116

DB2

June 2002

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: trevore@xephon.com

North American office

Xephon
PO Box 350100
Westminster, CO 80035-0100
USA
Telephone: 303 410 9344

Subscriptions and back-issues

A year's subscription to *DB2 Update*, comprising twelve monthly issues, costs £255.00 in the UK; $380.00 in the USA and Canada; £261.00 in Europe; £267.00 in Australasia and Japan; and £265.50 elsewhere. In all cases the price includes postage. Individual issues, starting with the January 1999 issue, are available separately to subscribers for £22.50 ($33.75) each including postage.

DB2 Update on-line

Code from *DB2 Update*, and complete issues in Acrobat PDF format, can be downloaded from our Web site at http://www.xephon.com/db2; you will need to supply a word from the printed issue.

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

When Xephon is given copyright, articles published in *DB2 Update* are paid for at the rate of £170 ($260) per 1000 words and £100 ($160) per 100 lines of code for the first 200 lines of original material. The remaining code is paid for at the rate of £50 ($80) per 100 lines. In addition, there is a flat fee of £30 ($50) per article. To find out more about contributing an article, without any obligation, please download a copy of our *Notes for Contributors* from www.xephon.com/nfc.

Printed in England.

Generating DB2 utility jobs ASAP

In my current data warehouse project I frequently need to copy or move data to different DB2 regions, mostly to testing subsystems but sometimes to production as well.

It has to be done as soon as possible and the objects that are involved in the migration work are numerous, usually 50 and sometimes more than 100.

Often the request comes in for different groups of objects, the JCL that I had set up previously may not work immediately, and it would be cumbersome to edit it for the many different object names.

The following REXX program will generate DB2 utility jobs – IBM copy, unload, load or BMC unload, BMC load – as soon as possible, using an input file that contains a list of object names such as table names and tablespace names.

```
/************************ REXX ***************************************/
/*    Generates DB2 utility jobs :                                  */
/*    IBM copy, IBM unload, IBM load, BMC unload, BMC load.         */
/*    This REXX program becomes handy when there are numerous       */
/*    tablespaces or tables to process - copy or unload/load        */
/*    and the JCL has to be prepared as soon as possible.           */
/*    This program will create the JCL immediately with different   */
/*    variations.                                                   */
/*********************************************************************/
/**   Input dataset needed                          */
/**     this program does not use REXX/SQL interface, even though   */
/**     it can be changed to do so.                                 */
/**     Therefore the input dataset should provide the following    */
/**     information for the objects (tables or tablespace) to       */
/**     process the utility such as copy, unload or load.           */
/**        DATABASE NAME, TABLESPACE NAME,  TABLE NAME              */
/**                                                                 */
/**     The information can be retrieved from the DB2 catalog,      */
/**     SYSIBM.SYSTABLES by running SPUFI or QMF.                   */
/**                                                                 */
/**   Output dataset                                                */
/**     the JCL is created in the output dataset, which can be      */
/**     verified and edited before submited for execution.          */
/*********************************************************************/
/***  Specify the number of tables to unload or load in each job    */
/***  or specify the number of tablespace to copy in each job and   */
```

```
/***  the program will generate multiple jobs, which can run         */
/***  concurrently, reducing the processing time.                    */
/***  Other than that there is a limit on the number of files        */
/***  that can reside on a tape volume, 254.                         */
/***                                                                 */
/***  It will be beneficial to have the unload datasets on           */
/***  separate tape volumes in case multiple load jobs are running   */
/***  concurrently and requesting the unload files on the same       */
/***  volume, causing contention.                                    */
/*********************************************************************/
/**   Arguments ((k)eyword or (o)ptional)                            */
/**                                                                  */
/*  util  : (K) Utilities to run - IC(IBM copy), IU(IBM unload),     */
/*                 IL(IBM load), BU(BMC unload), BL(BMC load)        */
/*  ssn   : (K) DB2 subsystem name. (Ex. DB2A)                      */
/*  dateq : (K) Qualifier of the output datasets. Arbitrary.        */
/*          (Ex. DØ1151 for Jan. 15)                                */
/*  indsn : (K) Input dataset name.                                 */
/*          Will have dbname, tsname and table name to process.     */
/*  nt_job : (K) Number of tables or tablespaces to be processed    */
/*           in a job. If there are more objects to process than    */
/*           this parameter, then the program will generate         */
/*           multiple jobs.                                         */
/*  creator : (K) Creator of the tables. This information should    */
/*            not be in the input dataset.                          */
/*            Will be needed for IBMUL or BMCUL                     */
/*  media : (P) 'D'(Disk) or 'T'(tape) where the output is created. */
/*          BMCUL routines need this argument.                      */
/*  dual : (P) Option for IBM copy.                                 */
/*         Will make dual copy if specified as 'dual'.              */
/*         If blank or otherwise then it will be single.            */
/*  ic_num : (P)                                                    */
/*           Option for BMC unload utility with imagecopy input.    */
/*           X means direct unload from the table, not imagecopy.   */
/*           Ø is the most recent imagecopy to unload from.         */
/*           -1 is the previous than the most recent, etc.          */
/*  slvl : (P)                                                      */
/*         Option for BMC unload, shrlevel.                         */
/*         If 'CHANGE' then it will accept imagecopied that were    */
/*         taken with shrlevel change.                              */
/*********************************************************************/
/*   optional parameters :                                          */
/*     ibmcp - dual (DUAL or anything)                              */
/*     bmcul - media (T or D), ic_num (X, Ø, -1 etc),               */
/*             slvl (CHANGE or anything)                            */
/*********************************************************************/
Arg util ssn dateq indsn nt_job creator optional
Address TSO
"ALLOC DDNAME(tnamedd) DSNAME('"indsn"') SHR REUSE"
x = sysdsn('myid.genjcl.out')
If x <> 'OK' then do
```

```
   Address TSO
   "ALLOC DDNAME(outdd) DSNAME('myid.genjcl.out') NEW CATALOG ",
   "SPACE(15,15) TRACKS UNIT(DISK) RECFM(F B) LRECL(8Ø) BLKSIZE(312Ø)"
End
Else do
   Address TSO
   "ALLOC DDNAME(outdd) DSNAME('myid.genjcl.out') SHR REUSE"
End

If RC then do
  Say " File allocation error ..."
  exit -16
End

/*************************************************************/
/***  Read in the input -  dbname, tsname, table_name       */
/*  The table_name is not qualified.                         */
/*************************************************************/
"EXECIO * DISKR tnamedd (FINIS"
Number_of_obj = queued()

Select
   When util = 'IC' then
      Call IBMCP
   When util = 'IL' then
      Call IBMLD
   When util = 'IU' then
      Call IBMUL
   When util = 'BL' then
      Call BMCLD
   When util = 'BU' then
      Call BMCUL
   Otherwise
      Say 'Error, use valid util - IC, IL, IU, BL or BU'
End

Address tso
"EXECIO * DISKW outdd (finis"
"FREE FI(tnamedd)"
"FREE FI(outdd)"
Exit Ø

IBMUL:
/*********************************************************************/
/*    IBM unload job                                                 */
/****    The DDNAME SYSREC does not allow more than 1ØØ tables    */
/****    in the range of ØØ thru 99.                                */
/****    Generates multiple jobs that can run concurrently          */
/*********************************************************************/
Do i = 1 to Number_of_obj
   Pull word1 word2 word3
```

```
   dbname.i = word1
   tsname.i = word2
   tblname.i = word3
   tblname.i = strip(tblname.i)
   dd2.i = '//   DISP=(NEW,CATLG),UNIT=AFF=SYSREC' || i-1 ||','
   dd3.i = '//   LABEL=,VOL=(PRIVATE,RETAIN,,5Ø,REF=*.' || ,
         'SYSREC'|| i-1 || '),'
   End
   dd2.1 = '//   DISP=(NEW,CATLG),UNIT=CART,'
   dd3.1 = '//   LABEL=1,VOL=(PRIVATE,RETAIN,,5Ø),'

remain = Number_of_obj // nt_job
If remain > Ø then
   Number_of_jobs = (Number_of_obj % nt_job) + 1
   Else Number_of_jobs = Number_of_obj % nt_job

/*********************************************************/
/*  Create the JCL                                       */
/*********************************************************/
Do j = 1 to Number_of_jobs
   Queue '//myidIU'||j||" JOB (SG,XWW,11ØØØ,4,12345),'SAM 12345',"
   Queue "//     CLASS=F,MSGCLASS=T,MSGLEVEL=(1,1),NOTIFY=myid"
   Queue "//*"
   Queue '/*JOBPARM SYSAFF=HSYS'
   Queue '/*ROUTE XEQ' ssn
   Queue '//*****************************************************'
   Queue '//UL1   EXEC   PGM=IKJEFTØ1,REGION=2Ø48K'
   Queue '//STEPLIB  DD  DSN=SAA.TS1.DB2APF,DISP=SHR'
   Queue '//SYSTSPRT DD  SYSOUT=*'
   Queue '//SYSPRINT DD  SYSOUT=*'
   Queue '//SYSUDUMP DD  SYSOUT=*'

   If j < Number_of_jobs then
      Number_of_tbl_4thejob = nt_job
      Else Number_of_tbl_4thejob = remain

   k = (j - 1) * nt_job
   Do i = Ø to Number_of_tbl_4thejob - 1
      dd_sysrec_num = right(i,2,'Ø')
      n = k + i
      dsn_sysrec_num = right(n,3,'Ø')
      sysrec_dsn = 'SYSRC' || dsn_sysrec_num
      Queue '//SYSREC'||dd_sysrec_num||' DD DSN=TEST.VV1.IBMUL.'|| ,
            ssn ||'.'|| sysrec_dsn ||'.'|| dateq
      Queue '//   DISP=(MOD,CATLG),UNIT=SYSDA,' || ,
            'SPACE=(TRK,(15,15Ø),RLSE)'
      End
   Queue '//SYSPUNCH DD DSN=TEST.VV1.IBMUL.'||ssn|| ,
         '.SYSPNCH'|| j || '.' || dateq ||','
   Queue '//    DISP=(MOD,CATLG),UNIT=SYSDA,' || ,
         'SPACE=(TRK,(15,15),RLSE)'
```

```
   Queue '//SYSTSIN  DD  *'
   Queue ' DSN S('|| ssn ||') '
   Queue ' RUN  PROGRAM(DSNTIAUL) PLAN(DSNTIAUL) - '
   Queue "      LIB('SAA.TS1.DB2APF') "
   Queue '  END'
   Queue '/*'
   Queue '//SYSIN  DD  *'
   l = k
   Do Number_of_tbl_4thejob
      l = l + 1
      Queue '  ' creator || '.' || tblname.l
      end
   Queue '/*'
   Queue '//*'
   End
Return

IBMCP:
/*********************************************************************/
/*    IBM copy utility jobs                                          */
/*********************************************************************/
dual = optional
Do i = 1 to Number_of_obj
   Pull word1  word2 word3
   dbname.i = word1
   tsname.i = word2
   tblname.i = word3
   dd1.i='//CP'||i||' DD DSN=TEST.VV1.'||ssn||'.'||dbname.i||'.' ,
        || tsname.i || '(+1)' ||','
   ddx1.i='//CPX'||i||' DD DSN=TEST.VV1.'||ssn||'.'||dbname.i||'.' ,
        || tsname.i || 'X.' || dateq || ','
   dd2.i = '//     DISP=(NEW,CATLG,UNCATLG),UNIT=AFF=CP' || i-1 ||','
   ddx2.i ='//     DISP=(NEW,CATLG,UNCATLG),UNIT=AFF=CPX' || i-1 ||','
   dd3.i = '//     LABEL=,VOL=(PRIVATE,RETAIN,,50,REF=*.' ,
        ||'CP'|| i-1 || '),'
   ddx3.i ='//     LABEL=,VOL=(PRIVATE,RETAIN,,50,REF=*.' ,
        ||'CPX'|| i-1 || '),'
   cpyctl.i = 'COPY TABLESPACE '|| dbname.i || '.' || tsname.i || ,
      ' FULL YES COPYDDN (CP' || i || ')'
   if dual = 'DUAL' then
     cpyctl.i = 'COPY TABLESPACE '|| dbname.i ||'.'|| tsname.i || ,
      ' FULL YES COPYDDN (CP' || i || ',CPX' || i || ')'
   quectl.i = '      TABLESPACE  ' || dbname.i || '.' || tsname.i
   End

dd2.1 = '//     DISP=(NEW,CATLG,UNCATLG),UNIT=CART,'
dd3.1 = '//     LABEL=1,VOL=(PRIVATE,RETAIN,,50),'
dd4 = '//     DCB=(SYS3.DSCB,BLKSIZE=28672,BUFNO=20)'

remain = Number_of_obj // nt_job
If remain > 0 then
```

```
   Number_of_jobs = (Number_of_obj % nt_job) + 1
   Else do
       Number_of_jobs = Number_of_obj % nt_job
       remain = nt_job
       end

/*********************************************************/
/*             Create the JCL                            */
/*********************************************************/
Do j = 1 to Number_of_jobs
   Queue "//myidCP"||j||" JOB (SG,XWW,11000,4,12345),'SAM 12345',"
   Queue "//     CLASS=F,MSGCLASS=T,MSGLEVEL=(1,1),NOTIFY=myid "
   Queue "//*"
   Queue '/*JOBPARM SYSAFF=HSYS'
   Queue '/*ROUTE XEQ ' || ssn
   Queue '//*****************************************************'
   Queue '//CPY EXEC DB2UTIL,SYSTEM='||ssn|| ,
         ",UID='''myid IC"||j||"'''"
   Queue '//*'
   n = (j-1) * nt_job + 1
   Queue dd1.n
   Queue dd2.1
   Queue dd3.1
   Queue dd4
   If dual = 'DUAL' then do
      Queue ddx1.n
      Queue dd2.1
      Queue dd3.1
      Queue dd4
      End
   If j < Number_of_jobs then
      Number_of_obj_4thejob = nt_job
      Else Number_of_obj_4thejob = remain
   n = n + 1
   Do i = 2 to Number_of_obj_4thejob
        Queue dd1.n
        Queue dd2.n
        insert_dd3 = insert(i,dd3.n,13)
        Queue insert_dd3
        Queue dd4
        If dual = 'DUAL' then do
            Queue ddx1.n
            Queue ddx2.n
            insert_ddx3 = insert(i,ddx3.n,13)
            Queue insert_ddx3
            Queue dd4
            End
        n = n + 1
        End
   Queue '//SYSIN  DD * '
   l = (j-1) * nt_job
```

```
   Do Number_of_obj_4thejob
      l = l + 1
      Queue cpyctl.l
      End
   Queue '//*'
   Queue '//QUIES EXEC DB2UTIL,SYSTEM=' || ssn || ,
         ",UID='''myid QU" ||j|| "''''"
   Queue '//SYSIN   DD  *'
   Queue '  QUIESCE'
   k = (j - 1 ) * nt_job
   Do Number_of_obj_4thejob
      k = k + 1
      Queue quectl.k
      End
   Queue '/*'
   Queue '//*'
   End

Return

BMCUL:
/*********************************************************************/
/***  BMC unload jobs from imagecopy or directly from the table.    */
/*********************************************************************/
Parse var optional media ic_num slvl
If media = 'T' then do
   exec1var = ',LBL='
   exec2 = "//       VOLREF="
   procvar = "//BMCUL  PROC TNAME=,UNIT='CART',LBL=,VOLREF="
   unitvar = '//    UNIT=&UNIT,DISP=(,CATLG),'
   tplablvar =  '//    LABEL=&LBL,VOL=(PRIVATE,RETAIN,,50&VOLREF.)'
   end
Else if media = 'D' then do
   exec1var = ''
   procvar = "//BMCUL  PROC TNAME="
   unitvar = '//    UNIT=SYSDA,DISP=(,CATLG),SPACE=(CYL,(1,10),RLSE)'
   tplablvar = '//*'
   end
   Else do
       say 'Invalid media parameter. Use D or T'
       exit (8)
       end

If ic_num = 'X' then unldvar = ' UNLOAD '
Else unldvar = ' UNLOAD  INFILE IMAGECOPY FULL ' || ic_num

Do i = 1 to Number_of_obj
   Pull word1  word2 word3
   dbname.i = word1
   tsname.i = word2
   tblname.i = word3
```

```
   exec1.i = "//ST  EXEC  BMCUL,TNAME=" || tsname.i || exec1var
   tblname.i = strip(tblname.i)
   sysin5.i = '   SELECT * FROM ' || creator || '.' || tblname.i
End

remain = Number_of_obj // nt_job
If remain > Ø then
   Number_of_jobs = (Number_of_obj % nt_job) + 1
   Else do
      Number_of_jobs = Number_of_obj % nt_job
      remain = nt_job
      end

/*********************************************************/
/*  Create the JCL                                       */
/*********************************************************/
Do j = 1 to Number_of_jobs
   Queue '//myidBU'||j||" JOB (SG,XWW,11ØØØ,4,12345),'SAM 12345',"
   Queue "//     CLASS=F,MSGCLASS=T,MSGLEVEL=(1,1),NOTIFY=myid "
   Queue "//*"
   Queue '/*JOBPARM SYSAFF=HSYS'
   Queue '/*ROUTE XEQ 'ssn
   Queue '//*****************************************************'
   Queue procvar
   Queue '//UL     EXEC PGM=ADUUMAIN,COND=(8,LT,UNCAT),REGION=ØM,'
   Queue "//  PARM='"||ssn||',myid BU'||j||',NEW/RESTART,,' || ,
         "MSGLEVEL(1)'"
   Queue '//*'
   Queue '//STEPLIB  DD  DSN=TTAT.TS2.DB2.APFLIB,DISP=SHR'
   Queue '//SYSPRINT DD  SYSOUT=*'
   Queue '//SYSOUT   DD  SYSOUT=*'
   Queue '//UTPRINT  DD  SYSOUT=*'
   Queue '//SYSUDUMP DD  SYSOUT=*'
   Queue '//SYSREC   DD  DSN=TEST.VV1.'||ssn||'.&TNAME..'|| ,
         'BMCUL.' || dateq || ','
   Queue unitvar
   Queue tplablvar
   Queue '//SYSCNTL  DD  DSN=TEST.VV1.'||ssn||'.LOADCNTL.' || ,
          dateq ||'(&TNAME.),'
   Queue '//    DISP=SHR '
   Queue '//SORTWKØ1 DD  UNIT=SYSDA,DISP=NEW,SPACE=(CYL,(1,1))'
   Queue '//SORTWKØ2 DD  UNIT=SYSDA,DISP=NEW,SPACE=(CYL,(1,1))'
   Queue '//SORTWKØ3 DD  UNIT=SYSDA,DISP=NEW,SPACE=(CYL,(1,1))'
   Queue '//SORTWKØ4 DD  UNIT=SYSDA,DISP=NEW,SPACE=(CYL,(1,1))'
   QUeue '//*'
   Queue '// PEND '
   Queue '//*'
   Queue '//*********************************************'
   Queue '//**   Insure the datasets do not exist.     ***'
   Queue '//*********************************************'
   Queue '//UNCAT  EXEC  PGM=IDCAMS,REGION=ØM'
```

```
   Queue '//SYSPRINT DD  SYSOUT=*'
   Queue '//SYSIN    DD  *'
   i = (j-1) * nt_job
   If j < Number_of_jobs then
      Number_of_tbl_4thejob = nt_job
   Else Number_of_tbl_4thejob = remain

   Do k = 1 to Number_of_tbl_4thejob
      i = i + 1
      Queue "  DEL 'TEST.VV1."||ssn||'.'||tsname.i|| ,
            '.BMCUL.' || dateq || "'" 'NONVSAM NOSCRATCH'
   End
   Queue '//**'
   If ic_num = '' then ic_num = 0
                    else nop
   n = (j-1) * nt_job
   If media = 'T' then
      Do i = 1 to Number_of_tbl_4thejob
         n = n + 1
         Queue insert(i,exec1.n,4) || i || ','
         If i > 1 then
            Queue exec2 || "',REF=*.ST" || i-1 || ".UL.SYSREC'"
            Else Queue exec2
         Queue '//SYSIN DD *'
         Queue unldvar
         If slvl = 'CHANGE' then
             Queue '    SHRLEVEL ' || slvl || ,
                   '   ORDER NO  LIMIT 0  INTERVAL 0  DISCARDS 0'
             Else queue  '   ORDER NO  LIMIT 0  INTERVAL 0  DISCARDS 0'
         Queue '   AUTOTAG NO  FORMAT STANDARD  FIXEDVARCHAR NO'
         Queue sysin5.n
         Queue '/*'
         Queue '//*'
      End
   Else  do i = 1 to Number_of_tbl_4thejob
         n = n + 1
         Queue insert(i,exec1.n,4)
         Queue '//SYSIN DD *'
         Queue unldvar
         If slvl = 'CHANGE' then
             Queue '    SHRLEVEL ' || slvl || ,
                   '   ORDER NO  LIMIT 0  INTERVAL 0  DISCARDS 0'
             Else queue  '   ORDER NO  LIMIT 0  INTERVAL 0  DISCARDS 0'
         Queue '   AUTOTAG NO  FORMAT STANDARD  FIXEDVARCHAR NO'
         Queue sysin5.n
         Queue '/*'
         Queue '//*'
      End
End
Return
```

```
IBMLD:
/*********************************************************************/
/***    Generate IBM load jobs using the table names as input,     ***/
/***    useful tool especially when there are numberous tables     ***/
/***    to process, more than hundreds.                            ***/
/***    Specify the number of tables to load in each job and       ***/
/***    the program will generate multiple jobs, which can run     ***/
/***    concurrently.                                              ***/
/****   If the load works in tandem with unload then the order     ***/
/****   of the table names should be the same as in the unload     ***/
/****   in case the unload datasets are in tapes, otherwise        ***/
/****   the load jobs will contend for tapes.                      ***/
/*********************************************************************/
Do i = 1 to Number_of_obj
   Pull word1 word2 word3
   tsname.i = word2
End
remain = Number_of_obj // nt_job
If remain > Ø then
   Number_of_jobs = (Number_of_obj % nt_job) + 1
   Else Number_of_jobs = Number_of_obj % nt_job
/*********************************************************/
/*  Create the JCL                                       */
/*********************************************************/
Do j = 1 to Number_of_jobs
   Queue '//myidLD'||j||" JOB (SG,XWW,11ØØØ,4,12345),'SAM 12345',"
   Queue '//             CLASS=L,MSGCLASS=T,'
   Queue '//             MSGLEVEL=(1,1),NOTIFY=myid '
   Queue '//*'
   Queue '/*ROUTE XEQ 'ssn
   Queue '//IBMLD  PROC TNAME='
   Queue '//LD EXEC DB2UTIL,SYSTEM='|| ssn || ,
         ",UID='''myid LD" || j || "''''"
   Queue '//*'
   Queue '//SYSREC DD DSN=TEST.VV1.'||ssn || ,
         '.&TNAME..BMCUL.' || dateq || ',DISP=SHR'
   Queue '//SYSUT1   DD DSN=TEST.VV1.'||ssn || ,
         '.&TNAME..SYSUT1,'
   Queue '//      DISP=(NEW,DELETE,DELETE),UNIT=SYSDA,'
   Queue '//      SPACE=(CYL,(5,2ØØ),RLSE)'
   Queue '//SORTOUT  DD DSN=TEST.VV1.'||ssn || ,
         '.&TNAME..SORTOUT,'
   Queue '//      DISP=(NEW,DELETE,DELETE),UNIT=SYSDA,'
   Queue '//      SPACE=(CYL,(5,2ØØ),RLSE)'
   Queue '//SYSIN    DD DSN=TEST.VV1.'||ssn|| ,
         '.LOADCNTL.'|| dateq || '(&TNAME.),'
   Queue '//     DISP=SHR'
   Queue '// PEND'
   Queue '//*'
   Queue '//************************************************'
   If j < Number_of_jobs
```

```
      Then k = nt_job
      Else k = remain
   n = (j - 1) * nt_job
   Do i = 1 to k
      n = n + 1
      Queue '//ST'||i||"  EXEC IBMLD,TNAME='"||tsname.n||"'"
   End
   Queue '/*'
   Queue '//*'
End
Return

BMCLD:
/*********************************************************************/
/***    Generate BMC load jobs                                     ***/
/*********************************************************************/
Do i = 1 to Number_of_obj
   Pull word1 word2 word3
   tsname.i = word2
End

remain = Number_of_obj // nt_job
If remain > Ø then
   Number_of_jobs = (Number_of_obj % nt_job) + 1
   Else Number_of_jobs = Number_of_obj % nt_job

/*********************************************************/
/*  Create the JCL                                       */
/*********************************************************/
Do j = 1 to Number_of_jobs
   Queue '//myidBL'||j||" JOB (SS,XWW,11ØØØ,4,12345),'SAM 12345',"
   Queue '//             CLASS=L,MSGCLASS=T,'
   Queue '//             MSGLEVEL=(1,1),NOTIFY=myid '
   Queue '//*'
   Queue '/*ROUTE XEQ 'ssn
   Queue '//BMCLD  PROC TNAME='
   Queue '//LOAD1  EXEC PGM=AMUUMAIN, '
   Queue "//       PARM='"||ssn||',myid BL' || j || ,
         ",NEW/RESTART,,MSGLEVEL(1)'"
   Queue '//STEPLIB  DD DSN=SAA.TS1.DB2APF,DISP=SHR '
   Queue '//*'
   Queue '//SYSRECØØ DD DSN=TEST.VV1.'||ssn||'.&TNAME..BMCUL.' || ,
          dateq || ',DISP=SHR'
   Queue '//SYSPRINT DD SYSOUT=*'
   Queue '//UTPRINT DD SYSOUT=*'
   Queue '//SYSUDUMP DD SYSOUT=*'
   Queue '//SYSERR   DD DISP=(NEW,CATLG,CATLG),SPACE=(CYL,(1,1),RLSE),'
   Queue '//    DSN=TEST.BMCUL.&TNAME..SYSERR.'||dateq||',UNIT=SYSDA'
   Queue '//SYSDISC  DD DISP=(NEW,CATLG,CATLG),SPACE=(CYL,(2,3),RLSE),'
   Queue '//    DSN=TEST.BMCUL.&TNAME..SYSDISC.'||dateq||',UNIT=SYSDA'
   Queue '//SORTWKØ1 DD  UNIT=SYSDA,DISP=NEW,SPACE=(CYL,(1Ø,5Ø))'
```

```
   Queue '//SORTWKØ1 DD  UNIT=SYSDA,DISP=NEW,SPACE=(CYL,(1Ø,5Ø))'
   Queue '//SORTWKØ2 DD  UNIT=SYSDA,DISP=NEW,SPACE=(CYL,(1Ø,5Ø))'
   Queue '//SORTWKØ3 DD  UNIT=SYSDA,DISP=NEW,SPACE=(CYL,(1Ø,5Ø))'
   Queue '//SORTWKØ4 DD  UNIT=SYSDA,DISP=NEW,SPACE=(CYL,(1Ø,5Ø))'
   Queue '//SORTWKØ5 DD  UNIT=SYSDA,DISP=NEW,SPACE=(CYL,(1Ø,5Ø))'
   Queue '//SORTWKØ6 DD  UNIT=SYSDA,DISP=NEW,SPACE=(CYL,(1Ø,5Ø))'
   Queue '//SORTWKØ7 DD  UNIT=SYSDA,DISP=NEW,SPACE=(CYL,(1Ø,5Ø))'
   Queue '//SORTWKØ8 DD  UNIT=SYSDA,DISP=NEW,SPACE=(CYL,(1Ø,5Ø))'
   Queue '//SORTWKØ9 DD  UNIT=SYSDA,DISP=NEW,SPACE=(CYL,(1Ø,5Ø))'
   Queue '//SORTWK1Ø DD  UNIT=SYSDA,DISP=NEW,SPACE=(CYL,(1Ø,5Ø))'
   Queue '//SORTWK11 DD  UNIT=SYSDA,DISP=NEW,SPACE=(CYL,(1Ø,5Ø))'
   Queue '//SORTWK12 DD  UNIT=SYSDA,DISP=NEW,SPACE=(CYL,(1Ø,5Ø))'
   Queue '//SYSIN    DD  DSN=TEST.VV1.'||ssn|| ,
         '.LOADCNTL.'|| dateq || '(&TNAME.),DISP=SHR'
   Queue '// PEND'
   Queue '//*'
   Queue '//************************************************'
   If j < Number_of_jobs
      Then k = nt_job
      Else k = remain
   n = (j - 1) * nt_job
   Do i = 1 to k
      n = n + 1
      Queue '//ST'||i||"  EXEC BMCLD,TNAME='"||tsname.n||"'"
   End
   Queue '/*'
   Queue '//*'
End
Return
```

Sam Park
DBA (USA)

Compiling programs with the NODYNAM option

INTRODUCTION

In some cases it is necessary to compile programs with the NODYNAM option.

At our site CASE-tool-generated programs are affected by this, because they might call non-recursive (also generated) programs, which themselves might be called recursively through further subprograms, which have been called dynamically. This is the reason why those non-

recursive programs have to be linked statically to their callers and the callers therefore have to be compiled using NODYNAM. This prevents recursion from happening because the statically-linked subprogram is always a part of its own caller.

Unfortunately, the DB2 precompiler transfers DB2 statements within application programs to a **CALL "DSNHLI"** statement. This, with the NODYNAM compile option, causes the link-editor to link DSNHLI statically, too.

In most cases, the load modules produced are only executable in an environment that fits with the version of DSNHLI which was statically linked to the load module. Others are tolerated, at best, but might fail – for example DSNHLI000(IMS) linked programs will abend S0C4 in a TSO environment, where DSNELI is required.

That's why we actually keep differently-linked versions of those programs – one which was linked with the DB2 language interface from the IMS library (DFSLI000, which includes DSNHLI) in one load-library, and a second version with the DB2 language interface from TSO (DSNHLI, which is an alias of DSNELI) in another load-library. In addition, setting up JCL STEPLIB statements has to be done very accurately to make sure that the load module fits the environment (IMS or TSO) it should be executed in.

WHY MYOWNHLI MAY BE USEFUL

Using MYOWNHLI will satisfy the need for static linking of DSNHLI to the application program load module; the link-editor will produce an executable load module, which may be executed in a TSO, CAF, or IMS environment, or even in an environment which requires a special third-party version of the DB2 language interface! There's no need to keep differently linked versions of the same application program in different load libraries!

HOW TO USE MYOWNHLI

Compile and link-edit MYOWNHLI; make sure to copy the resulting MYOWNHLI load module to a library which is used as SYSLIB by the

link-editor. Use MYOWNHLI at link-edit time instead of DSNHLI (or DFSLI000, DSNELI, or third-party versions of DSNHLI) by qualifying an **INCLUDE SYSLIB(MYOWNHLI)** statement.

Alternatively, include MYOWNHLI directly in your COBOL source program as a nested subprogram. You can do so either by copying MYOWNHLI's statements directly to the end of your COBOL source or by inserting a **COPY MYOWNHLI.** statement instead, to cause your COBOL (pre-)compiler to copy it from the library where your copybooks etc reside.

In either case, make sure to insert an additional **END PROGRAM xyz.** statement after the nested MYOWNHLI (if not already present – assuming 'xyz' is the name of your COBOL source) and comment the ENVIRONMENT DIVISION statements and the ENTRY statements as mentioned in MYOWNHLI source code.

HOW MYOWNHLI WORKS

Now that MYOWNHLI will be executed every time when DSNHLI is called (that is to say when your DB2 statements were transferred to a **CALL "DSNHLI"** statement by the DB2 precompiler). It is able to call the appropriate DSNHLI dynamically from a LINKLIST or STEPLIB library, by calling a nested subprogram 'INTERFACE-TO-DYN-DB2-LI' which will do the call. As I mentioned previously this will not mean a recursive call (to the DSNHLI which is included in MYOWNHLI)!

Though MYOWNHLI should work with every program needing static linking, I would like to point out that, with languages other than COBOL, easier or more efficient work-arounds might be available. As I learned from a colleague who mainly uses PL/I instead of COBOL, you may cause PL/I modules to load DSNHLI dynamically by coding **FETCH DSNHLI ;**.

In MYOWNHLI's ENTRY statements, I've included those entry-names in the same order as they appeared as alias names of DSNHLI-load modules in several system libraries (which libraries I've searched you will find in MYOWNHLI's comment lines), without being sure whether they might all be really necessary!!!

MYOWNHLI

```
 ID DIVISION.
 PROGRAM-ID.     MYOWNHLI.
 AUTHOR.         FRANZ-JOSEF SCHREIBER.
 DATE-WRITTEN.   20.12.2001.
 DATE-COMPILED.
*================================================================*
* DESCRIPTION - MYOWNHLI                                         *
*================================================================*
* ---------------------------------------
*   MYOWNHLI - HOW-TO-USE AND WHY-TO USE
* ---------------------------------------
*
* 'OPENED' DB2-LANGUAGE-INTERFACE
*
* USING 'MYOWNHLI' AT LINK-EDIT ('INCLUDE SYSLIB(MYOWNHLI)'
* INSTEAD OF
* 'INCLUDE SYSLIB(DSNHLI)'    (FOR CAF-ATTACHMENT) OR
* 'INCLUDE SYSLIB(DSNELI)'    (FOR TSO-ATTACHMENT) OR
* 'INCLUDE SYSLIB(DFSLI000)') (FOR IMS-ATTACHMNET)
* WILL RESOLVE THE CALLS TO 'DSNHLI' WHICH WERE INSERTED INTO
* DB2 APPLICATION PROGRAMS BY THE DB2 PRECOMPILER;
* EXECUTABLE LOAD MODULES WILL BE OBTAINED.
*
* THUS, THE INTERFACE TO DB2 IS 'OPENED' AND A LOAD MODULE MAY BE
* USED EITHER WITH CAF, TSO, OR IMS;
* NO DIFFERENTLY LINKED VERSIONS OF THE SAME MODULE NEED TO BE
* STORED FOR USE UNDER THOSE DIFFERENT ENVIRONMENTS.
*
* IN ADDITION, THE SUITABLE SETTING UP OF STEPLIB-CONCATENATION IS
* EASED SINCE ONLY ONE APPLICATION PROGRAM LIBRARY IS NEEDED, TOO.
*
*================================================================*
* --------------------------
*   MYOWNHLI - HOW-IT-WORKS
* --------------------------
*
* AT RUNTIME ANY CALLS TO 'DSNHLI' WITHIN AN APPLICATION PROGRAM
* LEAD TO 'MYOWNHLI'.
* BY CALLING ITS NESTED SUBPROGRAM 'INTERFACE-TO-DYN-DB2-LI'
* 'MYOWNHLI' ITSELF NOW CAN CALL 'DSNHLI' DYNAMICALLY FROM THERE.
* THE APPROPRIATE VERSION OF THE 'DSNHLI' SHOULD BE FOUND WITHIN
* THE STEPLIB- OR LINKLIST-CONCATENATION FOR EXAMPLE.
*
*================================================================*
* ----------------------
*   MYOWNHLI - ATTENTION
* ----------------------
*
* TO MAKE SURE THAT PROGRAMS LINKED WITH 'MYOWNHLI' WILL ALSO WORK
```

```
* IN A CICS ENVIRONMENT ONE HAS TO VERIFY THAT ALIASES 'DSNHLI'
* AND 'DSNWLI' OF 'DSNCLI' EXIST IN THE CICS INSTALLATION LIBRARY.
* OTHERWISE THE APPLICATION PROGRAMS NEED TO BE LINKED WITH
* 'DSNCLI' INSTEAD OF 'MYOWNHLI' FOR USE UNDER CICS!
*
*===============================================================*
* VERSIONS OF THE DB2-LANGUAGE-INTERFACE USED AT OUR SITE:
*
* IBM-SOFTWARE:
* =============
* - SYS1.DB2.LINKLIB      OUR DB2-INSTALLATION
*   ----------------      --------------------
*    NAME     ALIAS-OF
*   DSNALI                 DB2-CALL ATTACHMENT FACILITY
*   DSNHLI2   DSNALI
*   DSNWLI2   DSNALI
*   DSNELI                 DB2-TSO ATTACHMENT
*   DSNHLI    DSNELI
*   DSNRLI                 RRSAF-DB2 LANGUAGE INTERFACE ???
*   DSNHLIR   DSNRLI
*   DSNWLIR   DSNRLI
*
* - IMPG.RESLIB           OUR IMS-INSTALLATION
*   ----------------      --------------------
*    NAME     ALIAS-OF
*   DFSLI000               IMS-DB2 ATTACHMENT
*   DSNHLI    DFSLI000
*   DSNWLI    DFSLI000
*
* - CICS.P.SDFHLOAD       OUR CICS-INSTALLATION
*   ----------------      --------------------
*    NAME     ENTRY-POINTS
*   DSNCLI                 CICS/DB2 ATTACHMENT FACILITY
*            DSNHLI
*            DSNWLI
*
* 3RD-PARTY-SOFTWARE:
* ===================
* - DB2.P.QSTART.LOADLIB  OUR BMC'S QUICKSTART INSTALLATION
*   --------------------  ---------------------------------
*    NAME     ENTRY-POINTS
*   BATCHCAF               DB2-CALL ATTACHMENT FACILITY
*            DSNHLI
*            QSTART
*
* - IMPG.PPRESLIB         OUR BMC'S AR/CTL-INSTALLATION
*   --------------------  ---------------------------------
*    NAME     ALIAS-OF
*   ARCLI000               IMS-DB2 ATTACHMENT
*   DSNHLI    ARCLI000
*   DSNWLI    ARCLI000
```

```
*
*================================================================*
*                                                                *
* CHANGE-LOG:                                                    *
*----------------------------------------------------------------*
*                                                                *
 ENVIRONMENT DIVISION.
*----------------------------------------------------------------*
 CONFIGURATION SECTION.
*----------------------------------------------------------------*
 SPECIAL-NAMES.
     DECIMAL-POINT   IS COMMA
*    CLASS DIGIT     IS '1' '2' '3' '4' '5' '6' '7' '8' '9' '0'
     .
/
*----------------------------------------------------------------*
 DATA DIVISION.
*----------------------------------------------------------------*
 WORKING-STORAGE SECTION.
*----------------------------------------------------------------*
*
 01  PROGRAM-CONSTANTS-ALPHA.
     05  MY-PGM-ID               PIC  X(08)  VALUE 'MYOWNHLI'.
     05  MY-PGM-ENTRY            PIC  X(08)  VALUE SPACE.
     05  DSNALI                  PIC  X(08)  VALUE 'DSNALI '.
***  05  DSNCLI                  PIC  X(08)  VALUE 'DSNCLI '.
     05  DSNELI                  PIC  X(08)  VALUE 'DSNELI '.
     05  DSNHLI                  PIC  X(08)  VALUE 'DSNHLI '.
     05  DSNHLIR                 PIC  X(08)  VALUE 'DSNHLIR'.
     05  DSNHLI2                 PIC  X(08)  VALUE 'DSNHLI2'.
     05  DSNRLI                  PIC  X(08)  VALUE 'DSNRLI '.
     05  DSNWLI                  PIC  X(08)  VALUE 'DSNWLI '.
     05  DSNWLIR                 PIC  X(08)  VALUE 'DSNWLIR'.
     05  DSNWLI2                 PIC  X(08)  VALUE 'DSNWLI2'.
*----------------------------------------------------------------*
 LINKAGE SECTION.
*----------------------------------------------------------------*
*    DATA TO PASS TO THE 'REAL' DB2 LANGUAGE INTERFACE
*----------------------------------------------------------------*
 01  MYOWNHLI-DATA             PIC  X.
/
 PROCEDURE DIVISION                   USING  MYOWNHLI-DATA.
*    MOVE MY-PGM-ID                     TO  MY-PGM-ENTRY
     MOVE 'ERROR'                       TO  MY-PGM-ENTRY
     GO TO STEUERUNG.
*
     ENTRY 'DSNALI'                     USING  MYOWNHLI-DATA.
     MOVE DSNALI                        TO  MY-PGM-ENTRY
     GO TO STEUERUNG.
*
***  ENTRY 'DSNCLI'                     USING  MYOWNHLI-DATA.
```

```
***  MOVE DSNCLI                          TO  MY-PGM-ENTRY
***  GO TO STEUERUNG.
*
     ENTRY 'DSNELI'                    USING  MYOWNHLI-DATA.
     MOVE DSNELI                          TO  MY-PGM-ENTRY
     GO TO STEUERUNG.
*
     ENTRY 'DSNHLI'                    USING  MYOWNHLI-DATA.
     MOVE DSNHLI                          TO  MY-PGM-ENTRY
     GO TO STEUERUNG.
*
     ENTRY 'DSNHLIR'                   USING  MYOWNHLI-DATA.
     MOVE DSNHLIR                         TO  MY-PGM-ENTRY
     GO TO STEUERUNG.
*
     ENTRY 'DSNHLI2'                   USING  MYOWNHLI-DATA.
     MOVE DSNHLI2                         TO  MY-PGM-ENTRY
     GO TO STEUERUNG.
*
     ENTRY 'DSNRLI'                    USING  MYOWNHLI-DATA.
     MOVE DSNRLI                          TO  MY-PGM-ENTRY
     GO TO STEUERUNG.
*
     ENTRY 'DSNWLI'                    USING  MYOWNHLI-DATA.
     MOVE DSNWLI                          TO  MY-PGM-ENTRY
     GO TO STEUERUNG.
*
     ENTRY 'DSNWLIR'                   USING  MYOWNHLI-DATA.
     MOVE DSNWLIR                         TO  MY-PGM-ENTRY
     GO TO STEUERUNG.
*
     ENTRY 'DSNWLI2'                   USING  MYOWNHLI-DATA.
     MOVE DSNWLI2                         TO  MY-PGM-ENTRY
     GO TO STEUERUNG.
*
     CONTINUE.
*
 STEUERUNG SECTION.
*
*    STATIC CALL TO NESTED SUB-PROGRAM
     CALL "INTERFACE-TO-DYN-DB2-LI"    USING  MYOWNHLI-DATA
                                              MY-PGM-ENTRY
     END-CALL.
*
 STEUERUNG-EXIT.
     GOBACK.
 ID DIVISION.
 PROGRAM-ID.     INTERFACE-TO-DYN-DB2-LI.
 AUTHOR.         FRANZ-JOSEF SCHREIBER.
 DATE-WRITTEN.   20.12.2001.
 DATE-COMPILED.
```

```
*                                                                 *
 ENVIRONMENT DIVISION.
*-----------------------------------------------------------------*
*CONFIGURATION SECTION.
* ____________________________________ *
*SPECIAL-NAMES.
*    DECIMAL-POINT   IS COMMA
**** CLASS DIGIT     IS '1' '2' '3' '4' '5' '6' '7' '8' '9' '0'
**** .
/
*-----------------------------------------------------------------*
 DATA DIVISION.
*-----------------------------------------------------------------*
 LINKAGE SECTION.
 01  MYOWNHLI-DATA             PIC  X.
 01  DYN-CALL-PGM-ID           PIC  X(08).
/
 PROCEDURE DIVISION               USING  MYOWNHLI-DATA
                                         DYN-CALL-PGM-ID.
*
 STEUERUNG SECTION.
*
*    DYNAMIC CALL OF DB2-LANGUAGE-INTERFACE
**** MOVE "DSNHLI"                   TO  DYN-CALL-PGM-ID
     CALL DYN-CALL-PGM-ID          USING  MYOWNHLI-DATA
     END-CALL.
*
 STEUERUNG-EXIT.
 END PROGRAM     INTERFACE-TO-DYN-DB2-LI.
 END PROGRAM     MYOWNHLI.
```

Franz-Josef Schreiber
System Programmer
R+V Allgemeine Versicherung AG (Germany)

Back-up and restore

The new back-up feature of UDB 7.2 is the introduction of incremental and delta back-ups in addition to the online/offline back-ups that currently can be used. The concept of incremental and delta back-ups has been around in the mainframe world for many years, firstly in IMS and then in DB2. It is only with V7.2 that this new back-up technology is available in UDB DB2. Let's look at why you would want to use these new types of back-up.

Basically, you would want to use the new back-up features to speed up the back-up process and reduce the amount of storage media required for the back-up. The amount of time/space you save depends on lots of factors (volatility of data being just one), and the only way to find out just how much on your system is to perform each type of back-up and compare the results.

Now let's look at the new back-up features in detail. There are two new types of back-up, namely incremental and delta. The difference between the two is: an incremental back-up will back up all changes to a database since the last full (offline or online) back-up; a delta back-up will back up all changes since the last back-up of any type (offline/online/incremental/delta).

The 'price' you have to pay for using incremental/delta back-ups is that the restore process needs more planning than if you were just using the usual offline/online back-ups. This is shown in the example.

Be aware that currently you cannot back-up long field or large object data using incremental back-ups.

The DB2 processing involved is fully explained in the *Command Reference* manual, so I won't repeat it here. I suggest reading the chapter first, and then going through the example to see how it all fits together.

Let's now look at an actual example, to see how it works. In the example below (running UDB EE 7.2 FP4 on Windows 2000) I used the sample database, and all commands were issued from the same CLP session.

There is a new database configuration parameter (trackmod), which tracks the database updates. It has a default value of NO, which has to be changed to YES. You obviously need to use archive logging and not circular logging.

Note: if you are on FP3, then you need to ACTIVATE the database (at the points indicated below) to use incremental back-ups.

Create a directory to store the back-ups (I will be using c:\backups):

```
>DB2 UPDATE DB CFG FOR sample USING TRACKMOD ON
>DB2 UPDATE DB CFG FOR sample USING LOGRETAIN YES
```

Before you can implement incremental back-ups, you need to take a full

offline back-up of the database:

```
>DB2 BACKUP DB sample TO c:\backups

Backup successful. The timestamp for this backup image is :
20020113092902
Only issue if you are below FP4 >DB2 ACTIVATE DATABASE sample
```

Now do the following:

```
>DB2 CONNECT TO sample

>DB2 SELECT empno,edlevel FROM employee WHERE empno
IN('000010','000020','000030','000050')

EMPNO  EDLEVEL
-----  -------
000010      18
000020      18
000030      20
000050      16

>DB2 UPDATE employee SET edlevel=58 WHERE empno='000010'

>DB2 SELECT empno,edlevel FROM employee WHERE empno
IN('000010','000020','000030','000050')

EMPNO  EDLEVEL
-----  -------
000010      58
000020      18
000030      20
000050      16

>DB2 BACKUP DB sample ONLINE INCREMENTAL TO c:\backups

Backup successful. The timestamp for this backup image is :
20020113093125

>DB2 CONNECT TO sample

>DB2 UPDATE employee SET edlevel=58 WHERE empno='000020'

>DB2 SELECT empno,edlevel FROM employee WHERE empno
IN('000010','000020','000030','000050')

EMPNO  EDLEVEL
-----  -------
000010      58
000020      58
000030      20
000050      16
```

```
>DB2 BACKUP DB sample ONLINE INCREMENTAL DELTA TO c:\backups

Backup successful. The timestamp for this backup image is :
20020113093337

>DB2 CONNECT TO sample

>DB2 UPDATE employee SET edlevel=58 WHERE empno='000030'

>DB2 SELECT empno,edlevel FROM employee WHERE empno
IN('000010','000020','000030','000050')

EMPNO  EDLEVEL
-----  -------
000010      58
000020      58
000030      58
000050      16

>DB2 SELECT (CURRENT TIME) FROM sysibm.sysdummy1

1
--------
09:35:38

>DB2 BACKUP DB sample ONLINE INCREMENTAL DELTA TO c:\backups

Backup successful. The timestamp for this backup image is :
20020113093619

>DB2 CONNECT TO sample

>DB2 UPDATE employee SET edlevel=58 WHERE empno='000050'

>DB2 SELECT empno,edlevel FROM employee WHERE empno
IN('000010','000020','000030','000050')

EMPNO  EDLEVEL
-----  -------
000010      58
000020      58
000030      58
000050      58
```

Now, say we want to restore to a point just after the update of employee '000030', which is at 093538. We need to recover to the back-up, which is just before this point, and roll forward to the timestamp at point 093538.

If you are below FP4, issue:


```
>DB2 DEACTIVATE DATABASE sample
```

The automatic keyword in the **restore db** command below reduces much of the pain of issuing all the individual restore commands (as detailed in the *Command Reference* manual):

```
>DB2 RESTORE DB sample INCREMENTAL AUTOMATIC FROM c:\backups TAKEN AT
20020113093337 WITHOUT PROMPTING

SQL2540W  Restore is successful, however a warning "2539" was
encountered during Database Restore while processing in No Interrupt
mode.
```

Now we have to roll forward the database to the desired point. Don't forget that the back-up timestamp is a local timestamp, whereas the rollforward command requires a timestamp in CUT format.

On Windows, you can tell if there is a difference between local time and 'DB2 current time' by comparing the results of the output of the current time query with the result of the Windows **time** command.

At the time of writing, and taking into account my geographical location, the CUT time is the same as the local time. Therefore I can issue:

```
>DB2 ROLLFORWARD DB sample TO 2002-01-13-09.35.38 AND STOP


                                Rollforward Status

 Input database alias                   = sample
 Number of nodes have returned status   = 1

 Node number                            = 0
 Rollforward status                     = not pending
 Next log file to be read               =
 Log files processed                    = S0000004.LOG - S0000007.LOG
 Last committed transaction             = 2002-01-13-09.34.50.000000

DB20000I  The ROLLFORWARD command completed successfully.
```

Now connect to the database and check that the restore and rollforward have worked, and that we have gone back to the point just after employee number 000030 was updated:

```
>DB2 CONNECT TO sample

>DB2 SELECT empno,edlevel FROM employee WHERE empno
IN('000010','000020','000030','000050')
```

```
EMPNO  EDLEVEL
-----  -------
000010      58
000020      58
000030      58
000050      16
```

This shows that the restore has worked, and has demonstrated the use of incremental and delta back-ups.

Note that if you get the result shown below:

```
EMPNO  EDLEVEL
-----  -------
000010      18
000020      18
000030      58
000050      16
```

it means that you are on FP3 and didn't do the activate database command after switching TRACKMOD ON.

I hope I have explained the difference between incremental and delta back-ups and have demonstrated how to implement them. There is potential for a huge reduction in both back-up time and storage medium required, so try it out and see what gains you get.

C Leonard
Freelance Consultant (UK)

DB2 data toolkit

INTRODUCTION

This article is an implementation of the DB2 data toolkit procedure, published in *DB2 Update*, Issues 112 and 113, February and March 2002, in an article called *Data tool for database management*.

PROCEDURE DESCRIPTION

The functions implemented are described below:

- Create a job to Delete/Define an Alias for the stogroup vcat definition.
- Run stats builder for the target database.
- Create a detailed report of the selected object.
- Create a Free Plan/Package job to release all the information from the DB2 catalog.
- Create a job to Drop the tablespace objects, in order to cancel your database after you have completed the download phase.

CHECKLIST FOR INSTALLATION

Follow these steps to install the components of the REXX procedure:

- Use the preallocated USER.LIBRARY (the user.library allocated for DB2 data toolkit procedure).
- Copy all REXX, macro, and source enhancements into the USER.LIBRARY in the following members:
  - REXX: $DB2ALI0, $DB2RUNS, $DB2UTIL, $DB2FREE, UPDATE00.
  - Macro: $MDB2001, $MDB2010, $MDB2011, $MDB2014, $MDB2028, $MDB2030.
  - Source enhancement: IMPL00, IMPL01, IMPL03, IMPL04, IMPL05, IMPL06, IMPL07.
- Run macro UPDATE00 on the original REXX source code $DB2UN00.
- After installation delete members: IMPL00, IMPL01, IMPL03, IMPL04, IMPL05, IMPL06, IMPL07, UPDATE00.

The test environment is DB2 V5 in OS/390 Version 8 environment.

IMPL00 SOURCE CODE

```
/*—  Write vector for $Report0           —*/
            Call WrReport
```

IMPL01 SOURCE CODE

```
/*—  Write $Report0                          —*/
       jobw = firep
       "alloc da('"outdsrep"($REPORT0)') f("jobw") shr reuse"
       sj.0 = £p1
       "EXECIO * DISKW "jobw" (STEM sj. FINIS"
/*—  Delete/Define Alias job allocation         —*/
       say '' ; prmalloc = subsys','DBname','$vcatname',@DB2UN00'
       call @db2ali0 prmalloc
       if word(result,1) = 99 then exit
```

IMPL03 SOURCE CODE

```
/*—  Del./Def. Alias file allocation      —*/
       outdsali= $hiwork'.'subsys'.'DBname'.@DB2UN00.JOBALIA'
       prmalloc = subsys' 'outdsali' 0 1,15 f,b 80 2720 fiali yes'
       call @db2all0 prmalloc ; if word(result,1) = 99 then exit
```

IMPL04 SOURCE CODE

```
/*—  Report file allocation                  —*/
       outdsrep= $hiwork'.'subsys'.'DBname'.@DB2UN00.REPORT'
       prmalloc = subsys' 'outdsrep' 20 300,90 f,b 133 1330 firep yes'
       call @db2all0 prmalloc ; if word(result,1) = 99 then exit
       #t0  = 'Creator '              ; #s0  = '________'
       #t1  = 'TBLspace'              ; #s1  = '________'
       #t2  = 'TStype'                ; #s2  = '______'
       #t3  = 'Prt'                   ; #s3  = '___'
       #t4  = '    Table Name    '    ; #s4  = '____________________'
       #t5  = ' Record  '             ; #s5  = '_________'
       #t6  = '  Last Runstat  '      ; #s6  = '__________________'
       #t7  = 'Encod.'                ; #s7  = '______'
       #t8  = 'DBID'                  ; #s8  = '____'
       #t9  = 'PSID'                  ; #s9  = '____'
       #t10 = 'OBID'                  ; #s10 = '____'
       #t11 = '  Table Created '      ; #s11 = '________________'
       #t12 = '  Table Altered '      ; #s12 = '________________'
sj.1 = '                      ----------------------------       —->'
sj.2 = ' Date : 'data'        ___   Archive  Report                ___'
sj.3 = ' Time : 'ora'         ___   DataBase 'left(DBname,8,' ')'  ___'
sj.4 = '                      ------------------------------------ '
sj.5 = '                                                           '
sj.6 = #t0' '#t1' '#t2' '#t3' '#t4' '#t5' '#t6' '#t7' '#t8' '#t9' '#t10'
'#t11' '#t12
sj.7 = #s0' '#s1' '#s2' '#s3' '#s4' '#s5' '#s6' '#s7' '#s8' '#s9' '#s10'
'#s11' '#s12
```

IMPL05 SOURCE CODE

```
/*—  Drop file allocation                  —*/
       outdsdrp= $hiwork'.'subsys'.'DBname'.@DB2UN00.JOBDROP'
       prmalloc = subsys' 'outdsdrp' 0 1,15 f,b 80 27920 fidrp yes'
       call @db2all0 prmalloc ; if word(result,1) = 99 then exit
sk.1='//'jna'.A JOB ('$accn'),''Drop Tablespace'',CLASS='$class',     '
sk.2='//      MSGCLASS='$msgcla',USER='$user',REGION='$region',       '
sk.3='//      MSGLEVEL=('$msglvl'),NOTIFY='$notif
sk.4='/*JOBPARM BYTES=999999,LINES=9999                               '
sk.5='//* ------------------------------------ *            '
sk.6='//* ——    DROP Tablespace procedure    —— *            '
sk.7='//* ------------------------------------ *            '
sk.8='//DB2PROC  JCLLIB ORDER=('$proclib')                            '
sk.9='//REXX00   EXEC   DB2REXX1                                      '
sk.10='//REXX00.SYSTSIN  DD  *                                        '
sk.11='  ISPSTART CMD(@DB2UTIL 'subsys','DBname',*,*,drop,si,*)      '
sk.12='//* ------------------------------------ *            '
       sk.0=12 ; jobw = fidrp ; Call WriteRec
```

IMPL06 SOURCE CODE

```
/*—  Free Plan/Pack, file allocation      —*/
       outdsfre= $hiwork'.'subsys'.'DBname'.@DB2UN00.JOBFREE'
       prmalloc = subsys' 'outdsfre' 0 1,15 f,b 80 27920 fifre yes'
       call @db2all0 prmalloc ;if word(result,1) = 99 then exit
sk.1='//'jna'.F JOB ('$accn'),''Free Plan/Pack'',CLASS='$class',     '
sk.2='//      MSGCLASS='$msgcla',USER='$user',REGION='$region',       '
sk.3='//      MSGLEVEL=('$msglvl'),NOTIFY='$notif
sk.4='/*JOBPARM BYTES=999999,LINES=9999                               '
sk.5='//* ------------------------------------------ *            '
sk.6='//* --------    Free Catalog Extractor    ------- *            '
sk.7='//* ------------------------------------------ *            '
sk.8='//DB2PROC   JCLLIB ORDER=('$proclib')                           '
sk.9='//JOBLIB    DD  DISP=SHR,DSN='$dsnload
sk.10='//          DD  DISP=SHR,DSN='$plilink
sk.11='//          DD  DISP=SHR,DSN='$sibmlnk
sk.12='//* ------------------------------------------ *            '
sk.13='//DELVIEW  EXEC PGM=IKJEFT01,DYNAMNBR=20                       '
sk.14='//SYSTSPRT  DD  SYSOUT=*                                       '
sk.15='//SYSPRINT  DD  SYSOUT=*                                       '
sk.16='//SYSTSIN   DD  *                                              '
sk.17=' DSN SYSTEM('subsys')                                          '
sk.18=' RUN PROGRAM('$tep2pgm') PLAN('$tep2pln') LIB('''$runlib''')   '
sk.19='//SYSIN    DD  *                                               '
sk.20='  DROP VIEW V'ownerb'_FRPLAN  ;                                '
sk.21='  DROP VIEW V'ownerb'_FRPACK  ;                                '
sk.22='//* ------------------------------------------ *            '
sk.23='//LAB0     IF (DELVIEW.RC LT 9) THEN                           '
sk.24='//CREVIEW  EXEC PGM=IKJEFT01,DYNAMNBR=20                       '
```

```
sk.25='//SYSTSPRT  DD  SYSOUT=*                                          '
sk.26='//SYSPRINT  DD  SYSOUT=*                                          '
sk.27='//SYSTSIN   DD  *                                                 '
sk.28=' DSN SYSTEM('subsys')                                             '
sk.29=' RUN PROGRAM('$tep2pgm') PLAN('$tep2pln') LIB('''$runlib''')      '
sk.30='//SYSIN    DD  *                                                  '
sk.31='                                                                  '
sk.32='   CREATE VIEW V'ownerb'_FRPLAN (PLAN) AS                         '
sk.33='   SELECT  NAME                                                   '
sk.34='           FROM 'catnam'.SYSPLAN                                  '
sk.35='           WHERE NAME     LIKE ''%''    AND                       '
sk.36='                 CREATOR  =    '''ownerb'''  ;                   '
sk.37='                                                                  '
sk.38='   CREATE VIEW V'ownerb'_FRPACK (PACKAGE,COLLECT,VERS) AS         '
sk.39='   SELECT  NAME,COLLID,VERSION                                    '
sk.40='           FROM 'catnam'.SYSPACKAGE                               '
sk.41='           WHERE COLLID   LIKE ''%''    AND                       '
sk.42='                 NAME     LIKE ''%''    AND                       '
sk.43='                 OWNER    =    '''ownerb''' ;                    '
sk.44='                                                                  '
sk.45='//* ------------------------------------------- *                 '
sk.46='//LAB1      IF (CREVIEW.RC EQ 0) THEN                             '
sk.47='//REXX00   EXEC   DB2REXX1                                        '
sk.48='//REXX00.SYSTSIN  DD  *                                           '
sk.49='  ISPSTART CMD(@DB2FREE 'subsys','ownerb',*,'$user'F)             '
sk.50='//LAB0END  ENDIF                                                  '
sk.51='//LAB1END  ENDIF                                                  '
sk.52='//* ------------------------------------------- *                 '
       sk.0=52 ; jobw = fifre ; Call WriteRec
```

IMPL07 SOURCE CODE

```
/*—  write routine $Report0              —*/
 WrReport :
     select
        when $encoding = 'E' then $encoding  = 'EBCDIC'
        when $encoding = 'A' then $encoding  = 'ASCII'
        otherwise
           $encoding  = 'Errore'
        end
     select
        when $partition = 0 & $ntable = 1 then wTStype = 'Simple'
        when $partition > 0 then wTStype = 'Partiz'
        when $partition = 0 & $ntable > 1 then wTStype = 'Segmen'
        otherwise
           wTStype = 'Errore'
        end
     if $alteredts = $createdts then $alteredts = '               '
    #a0 = left($creator,8,' ')   ; #a1 = left($tsname,8,' ')
    #a2 = left(wTStype,6,' ')    ;  #a3 = right($partition,3,' ')
```

```
    #a4 = translate(left($tblname,18,' '),' ','_')
    #a5 = right($card,9,' ')     ; #a6 = left($statstime,16,' ')
    #a7 = left($encoding,6,' ')  ; #a8 = right($dbid,4,'0')
    #a9 = right($psid,4,'0')     ; #a10= right($obid,4,'0')
    #a11= left($createdts,16,' '); #a12= left($alteredts,16,' ')
    #p1 = #p1 + 1
    sj.#p1 =  #a0' '#a1' '#a2' '#a3' '#a4' '#a5' '#a6' '#a7' '#a8' '#a9'
'#a10'
'#a11' '#a12
    return
```

UPDATE00 REXX EXEC

```
/* REXX */
trace ?o
/*--   Update Macro Enhancement   --*/
say ''
say '           @DB2UN00 REXX update is running.......               '
say ''
isredit macro
isredit find ''impl00''
call VerRc     impl00
isredit copy   impl00 after .zcsr
call VerRc     impl00
call Exlimpl   impl00
isredit find ''impl01''
call VerRc     impl01
isredit copy   impl01 after .zcsr
call VerRc     impl01
call Exlimpl   impl01
isredit change '"//*ISPSTART"' '"  ISPSTART"' first
call VerRc     chan00
call Exlimpl   impl02
isredit find ''impl03''
call VerRc     impl03
isredit copy   impl03 after .zcsr
call VerRc     impl03
call Exlimpl   impl03
isredit find ''impl04''
call VerRc     impl04
isredit copy   impl04 after .zcsr
call VerRc     impl04
call Exlimpl   impl04
isredit find ''impl05''
call VerRc     impl05
isredit copy   impl05 after .zcsr
call VerRc     impl05
call Exlimpl   impl05
isredit find ''impl06''
call VerRc     impl06
```

```
isredit copy   impl06 after .zcsr
call VerRc     impl06
call Exlimpl   impl06
isredit find ''impl07''
call VerRc     impl07
isredit copy   impl07 after .zcsr
call VerRc     impl07
call Exlimpl   impl07
say ''
say '    +-----------------------------------------------------------+'
say '    |      @DB2UN00 REXX update successfully executed           |'
say '    +-----------------------------------------------------------+'
say ''
isredit end
exit
VerRc :
   arg parmRc
   if rc > 0 then do
      say '    +-------------------------------------------------+  '
      say '    |  !!!!!!!!!!!   A T T E N T I O N     !!!!!!!!!!   |  '
      say '    |      Error during Code Update Function            |  '
      say '    |         Return Code 'rc' into Step 'parmRc'       |  '
      say '    | Activate trace ?r function and redo the UPDATE00  |  '
      say '    | Macro on the original Source Code downloaded from |  '
      say '    | Web Site                                          |  '
      say '    +-------------------------------------------------+  '
      isredit find parmRc
      exit
      end
   return
Exlimpl :
   arg parmRc
   if rc = 0 then do
      isredit exclude parmRc all
      isredit delete x all
      isredit save
      end
   return
```

$DB2ALI0 REXX EXEC

```
/* REXX */
   /*-------------------------------------------------------------------*/
   /*-                  Data Tool for DB management                    -*/
   /*-                    Delete/Define Alias                          -*/
   /*-------------------------------------------------------------------*/
arg parmin ; parm = translate(parmin,' ',',') ; nparm = words(parm)
subsys = word(parm,1) ; datab  = word(parm,2) ;alias  = word(parm,3)
rexxfrom = word(parm,4)
   /*-  Test input parameters                -*/
```

```
 if nparm < 4 then do
    say '' ; say '' ; say '>>>>>>>>'
    say '>>>>>>>>  Parameter string is incomplete !!!!'
    say '>>>>>>>>           'parmin
    say '>>>>>>>>' ; say '' ; say ''  ; $exitc = 99 ;
    return $exitc
    exit ; end
   /*-  Parameters assignment                -*/
 call @db2par0 subsys ;if word(result,1) = 99 then do
    $exitc = 99 ; return $exitc
    exit ; end
 $lpar =word(result,1) ; $accn =word(result,2) ; $class =word(result,3)
 $msgcla = word(result,4);$region  = word(result,5) ;$msglvl =
word(result,6)
 $notif  = word(result,7);$user    = word(result,8) ;$unitda =
word(result,9)
 $unitta = word(result,10);$esunit = word(result,11);$prt    =
word(result,12)
 $hiwork = word(result,13);$db2ver = word(result,14);$ctsubs =
word(result,15)
 $librexx= word(result,16);$parmlib= word(result,17);$proclib=
word(result,18)
 $jcllib = word(result,19);$report = word(result,20);$libexec=
word(result,21)
 $isptenu= word(result,22);$isppenu= word(result,23);$ispmenu=
word(result,24)
 $ispslib= word(result,25);$plilink= word(result,26);$sibmlnk=
word(result,27)
 $sortlib= word(result,28);$hilvlDB= word(result,29);$runlib =
word(result,30)
 $dsnload= word(result,31);$tep2pgm= word(result,32);$tep2pln=
word(result,33)
 $unlopgm= word(result,34);$unlopln= word(result,35);$dunlopg=
word(result,36)
 $dunlopl= word(result,37);$dsnproc= word(result,38)
   /*-   Work areas initialization          -*/
 blk     =  ; $notif = '&SYSUID'
   /*-  SYSPRINT file allocation            -*/
 xx=outtrap(trpdummy.)
    "free fi(sysprint)" ; "free fi(sysin)"
 xx=outtrap(off)
 outds1= $hiwork'.'subsys'.'datab'.@DB2ALI0.SYSPRINT'
 prmalloc = subsys' 'outds1' 0 60,30 f,b,a 121 1210 sysprint no'
 call @db2all0 prmalloc
 if word(result,1) = 99 then do
    $exitc = 99 ; return $exitc
    exit ; end
   /*-  SYSIN file allocation               -*/
 outds2 = $hiwork'.'subsys'.'datab'.@DB2ALI0.SYSIN'
 prmalloc = subsys' 'outds2' 0 1,1 f,b 80 27920 sysin no'
 call @db2all0 prmalloc
```

```
 if word(result,1) = 99 then do
    $exitc = 99 ; return $exitc
    exit ; end
   /*-  Listcat file allocation              -*/
 outds3 = alias'.'$hiwork
 prmalloc = subsys' 'outds3' Ø 1,1 f,b 8Ø 2792Ø fids3 no'
 call @db2all0 prmalloc
 if word(result,1) = 99 then do
    $exitc = 99 ; return $exitc
    exit ; end ; Call cidcams
 if word(result,1) = 99 then do
    $exitc = 99 ; return $exitc
    exit ; end
   /*-  Alias job file allocation            -*/
 outdsali= $hiwork'.'subsys'.'datab'.'rexxfrom'.JOBALIA'
 jobw = FIALI ; "alloc da('"outdsali"') f("jobw") mod reuse"
sk.1='//&SYSUID.D JOB ('$accn'),''Delete/Define Alias'',CLASS='$class','
sk.2='//      MSGCLASS='$msgcla',USER='$user',REGION='$region',       '
sk.3='//      MSGLEVEL=('$msglvl'),NOTIFY='$notif',TYPRUN=HOLD        '
sk.4='/*JOBPARM BYTES=999999,LINES=9999                               '
sk.5='//* ------------------------------------------- *               '
sk.6='//*           Delete/Define Alias               *               '
sk.7='//* ------------------------------------------- *               '
sk.8='//IDCAMSØØ EXEC PGM=IDCAMS                                      '
sk.9='//SYSPRINT  DD  SYSOUT=X                                        '
sk.1Ø='//SYSIN     DD  *                                              '
 walias= word(sysprint.1,3)
 wmcat = word(sysprint.2,3)
sk.11='  DELETE 'walias'  ALIAS  CAT ('wmcat'/......)                 '
sk.12='                                                               '
 sk.Ø=12 ; Call WriteRec ; alias = outds3 ; call cidcams
 wsubs = substr(subsys,4,1) ; wucat = word(sysprint.1,3)
 jobw = FIALI ; "alloc da('"outdsali"') f("jobw") mod reuse"
sk.1='  IF LASTCC > Ø THEN DO                                       '
sk.2='  DEFINE ALIAS                                    -           '
 lline = length('(NAME   ('walias')') ; wrk = 4Ø - lline
sk.3='        (NAME   ('walias')' copies(' ',wrk)'  -                 '
 lline = length('RELATE('wucat'))') ; wrk = 4Ø - lline
sk.4='         RELATE('wucat'))' copies(' ',wrk)' -                   '
sk.5='         CAT   ('wmcat'/......)                                 '
sk.6='         END                                                    '
sk.7='//* ———————————————————— *            '
       sk.Ø=7 ; Call WriteRec ; call free ; exit
   /*-  Call Idcams Routine                 -*/
 Cidcams :
    jobw = sysin
    sk.1=' LISTC ENTRY('alias')'
    sk.Ø = 1 ; call WriteRec
    xx=outtrap(trpØØ.) ; "ispexec select pgm(IDCAMS) "
       if rc > Ø then do
          say '' ; say '>>>>>>'
```

```
         say '>>>>>> RC listcat = 'rc
         say '>>>>>> Verify output. End elaboration '
         say '>>>>>>' ; say ''
         address tso "printoff ('"outds1"') class(X)"
         "free fi(sysin)" ; "free fi(sysprint)"
         $exitc = 99 ; return $exitc
         exit ; end ; xx=outtrap(off)
   xx=OUTTRAP(trp01.)
      "ispexec edit dataset('"outds1"') macro(@MDB2014)"
   xx=OUTTRAP(OFF)
   if rc > 0 then do
      do #a = 1 to trp01.0 ; say trp01.#a ; end
      $exitc = 99 ; return $exitc
      exit ; end
   xx=OUTTRAP(trp02.)
      "alloc da('"outds1"') f(sysprint) shr reuse"
      "execio * diskr sysprint (stem sysprint. finis"
   xx=OUTTRAP(OFF)
   if rc > 0 then do
      do #a = 1 to trp02.0 ; say trp02.#a ; end
      say '' ; say ''
      say '>>>>>>>>  Error reading file "'outds1'"    '
      say '>>>>>>>>  RC='rc'. Verify.                   '
      say '' ; say '' ; $exitc = 99 ; return $exitc
      exit ; end ; return
  /*-  Write record routine                -*/
WriteRec :
   "EXECIO * DISKW "jobw" (STEM sk. FINIS"
ClearRec:
   DO #f = 1 to sk.0 ; sk.#f = blk ; end ; return
  /*-  Free Work areas                     -*/
Free :
   xx=outtrap(trpdummy.)
      "free fi(fiali)" ; "free fi(fids3)"
      "free fi(sysin)" ; "free fi(sysprint)"
      address tso "delete '"outds1"'"
      address tso "delete '"outds2"'"
      address tso "delete '"outds3"'"
   xx=outtrap(off) ; return
```

$DB2RUNS REXX EXEC

```
/* REXX */
   /*-------------------------------------------------------------*/
   /*-                  Data Tool for DB management              -*/
   /*-                  DataBase Runstats generator              -*/
   /*-------------------------------------------------------------*/
 arg parmin ; parm = translate(parmin,' ',',') ;nparm = words(parm)
 subsys = word(parm,1) ;datab = word(parm,2)   ;autosub = word(parm,3)
 nextjob = word(parm,4)
```

```
   /*—  Test input parameters                —*/
 if nparm < 4 then do
    say '' ; say '' ; say '>>>>>>>>'
    say '>>>>>>>>  Parameter string is incomplete !!!!'
    say '>>>>>>>>           'parmin
    say '>>>>>>>>' ; say '' ; say '' ; exit ; end
 if autosub = '*' then autosub = NO
 if autosub ¬= NO & autosub ¬= YES then do
    say '' ; say '' ; say '>>>>>>>>'
    say '>>>>>>>>  Wrong Autosub 'autosub' variable !!!!!'
    say '>>>>>>>>  Specify : */NO/YES
    say '>>>>>>>>' ; say '' ; say '' ; exit ; end
 if nextjob = '*' then nextjob =  NO
   /*—  Parameters assignment                —*/
 call @db2parØ subsys ; if word(result,1) = 99 then exit
 $lpar =word(result,1) ; $accn =word(result,2) ; $class =word(result,3)
 $msgcla = word(result,4);$region  = word(result,5) ;$msglvl =
word(result,6)
 $notif  = word(result,7);$user    = word(result,8) ;$unitda =
word(result,9)
 $unitta = word(result,1Ø);$esunit = word(result,11);$prt    =
word(result,12)
 $hiwork = word(result,13);$db2ver = word(result,14);$ctsubs =
word(result,15)
 $librexx= word(result,16);$parmlib= word(result,17);$proclib=
word(result,18)
 $jcllib = word(result,19);$report = word(result,2Ø);$libexec=
word(result,21)
 $isptenu= word(result,22);$isppenu= word(result,23);$ispmenu=
word(result,24)
 $ispslib= word(result,25);$plilink= word(result,26);$sibmlnk=
word(result,27)
 $sortlib= word(result,28);$hilvlDB= word(result,29);$runlib =
word(result,3Ø)
 $dsnload= word(result,31);$tep2pgm= word(result,32);$tep2pln=
word(result,33)
 $unlopgm= word(result,34);$unlopln= word(result,35);$dunlopg=
word(result,36)
 $dunlopl= word(result,37);$dsnproc= word(result,38)
   /*—   Work areas initialization          —*/
 blk     = ; #jnr  = Ø ;#numjob = Ø ;sidispl = no ;comdb = datab
 if datab ¬= '*' then comdb1 = datab
 else comdb1 = SYSTEM
   /*—  SYSIN file allocation                —*/
 say '' ;outdsØ= $hiwork'.'subsys'.'comdb1'.SYSIN'
 prmalloc = subsys' 'outdsØ' Ø 5,1 f,b 8Ø 2792Ø sysin YES'
 call @db2allØ prmalloc ;if word(result,1) = 99 then exit
 jobw = sysin ;"alloc da('"outdsØ"') f("jobw") mod reuse"
 if datab = '*' then do
   sk.1='SELECT * FROM VDB'subsys' ORDER BY 1,2 ; '
   sk.Ø=1
```

```
   sidispl = YES ; end
 else do
   sk.1='SELECT DBNAME,NAME FROM SYSIBM.SYSTABLESPACE     '
   sk.2='       WHERE DBNAME LIKE '''datab'%'' ORDER BY 1,2 ; '
   sk.0=2 ; end ;Call WriteRec
   /*-  SYSTSIN file allocation              -*/
 outds1= $hiwork'.'subsys'.'comdb1'.SYSTSINP'
 prmalloc = subsys' 'outds1' 0 5,1 f,b 80 2720 systsinp YES'
 call @db2all0 prmalloc ;if word(result,1) = 99 then exit
sk.1='DSN SYSTEM('subsys')                                          '
sk.2='RUN PROGRAM('$tep2pgm') PLAN('$tep2pln') LIB('''$runlib''')    '
sk.3='END                                                           '
 sk.0=3 ;"execio * diskw systsinp (stem sk. finis" ;Call ClearRec
   /*-  SYSPRINT file allocation             -*/
 outds2= $hiwork'.'subsys'.'comdb1'.SYSPRINT'
 prmalloc = subsys' 'outds2' 0 45,15 f,b 121 1210 sysprint YES'
 call @db2all0 prmalloc ;if word(result,1) = 99 then exit
   /*-  Runstat file allocation              -*/
 outdsrun= $hiwork'.'subsys'.'comdb1'.JOBRUNS'
 prmalloc = subsys' 'outdsrun' 0 15,15 f,b 80 2720 firun YES'
 call @db2all0 prmalloc ;if word(result,1) = 99 then exit
 say ''
 say ' ********************************************************** '
 say '*                                            'time()'     *'
 say '*       DB2 query in progress                              *'
 say '*                                                          *'
 say '*                        wait please .......               *'
 say '*                                                          *'
 say ' ********************************************************** '
 say ''
 xx=outtrap(trp05.) ;address tso "ex '"outds1"'" ;xx=outtrap(off)
 if rc > 0 then do
    analisi = substr(trp05.1,1,8)
    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 trp05.0
          say trp05.#a  ; end
       end ; exit ; end
 say '' ; say '-->>>  DB2 query ended. Start elaboration ' ;say ''
 xx=outtrap(trpdummy.) ; address tso "printoff ('"outds2"')  class(R)"
 xx=outtrap(off)
 xx=OUTTRAP(trp06.) ;"ispexec edit dataset('"outds2"') macro(@mdb2028)"
 xx=OUTTRAP(OFF)
 if rc > 0 then do
    do #a = 1 to trp06.0
       say trp06.#a ; end ; exit ; end
```

```
   /*—  Output elaboration                    —*/
 xx=outtrap(trpØ7.)  ; "execio * diskr sysprint (stem sysprint. finis"
    "free fi(sysprint)"
 xx=outtrap(off)
 if rc > Ø then do
    do #a = 1 to trpØ7.Ø
       say trpØ7.#a ; end
    say '' ; say '' ; say '>>>>>>>>'
    say '>>>>>>>>  Error reading file "'outds2'"'
    say '>>>>>>>>  RC='rc'. Verify              '
    say '>>>>>>>>' ; say '' ; say '' ; exit ;  end
 if sysprint.Ø = Ø then do
    say '' ;say '' ; say '>>>>>>>>'
    say '>>>>>>>>  file "'outds2'" is empty        '
    say '>>>>>>>>  probably DB2 K.O.  access !!!!  '
    say '>>>>>>>>' ; say '' ; say '' ; exit ; end
 if substr(sysprint.1,33,8) = ' Ø ROW(S)' then do
    say '' ; say '' ; say '>>>>>>>>'
    say '>>>>>>>>  There are no Tablespace in DataBase selected.'
    say '>>>>>>>>' ; say '' ; say '' ; call free ; exit ; end
   /*—  Write jcl                             —*/
 DO #i = 1 to sysprint.Ø
    datab  = word(sysprint.#i,1) ; tsspa  = word(sysprint.#i,2)
    jobw = firun ; "alloc da('"outdsrun"') f("jobw") mod reuse"
    if #jnr > 5 then #jnr = Ø ; call hdrjob ; #jnr = #jnr + 1
sk.1=' RUNSTATS TABLESPACE ('datab'.'tsspa') INDEX(ALL)
SHRLEVEL(CHANGE)'
    sk.Ø=1 ; Call WriteRec ;#k = #i + 1
    datab_com = word(sysprint.#k,1) ; #g = Ø
    do while datab = datab_com
       #i = #i + 1 ; #g = #g + 1
       datab  = word(sysprint.#i,1) ; tsspa  = word(sysprint.#i,2)
sk.#g=' RUNSTATS TABLESPACE ('datab'.'tsspa') INDEX(ALL)
SHRLEVEL(CHANGE)'
       #k = #i + 1 ; datab_com = word(sysprint.#k,1) ; end
    sk.Ø=#g ; Call WriteRec
sk.1='//* ------------------------------------------------- *      '
    sk.Ø=1 ; Call WriteRec
   /*-  Stop/Start DataBase                  -*/
    if comdb1 = SYSTEM then do
       db  = center(datab,8)
sk.1='//* ---              Stop/Start   'db'          --- *          '
sk.2='//* ------------------------------------------------- *      '
sk.3='//STOPSTAR EXEC PGM=IKJEFTØ1,DYNAMNBR=2Ø                     '
sk.4='//SYSTSPRT  DD  SYSOUT=X                                     '
sk.5='//SYSTSIN   DD    *                                          '
sk.6='  DSN SYSTEM('subsys')                                      '
sk.7=' -STOP   DB('datab') SPACENAM(*)                             '
sk.8=' -START  DB('datab') SPACENAM(*)                             '
sk.9='//* ------------------------------------------------- *      '
       sk.Ø=9 ; Call WriteRec ; end ; end
```

```
 if nextjob ¬= NO then do
 sk.1 ='//* ------       Start prossimo job              ----- *      '
 sk.2 ='//* ------------------------------------------------ *      '
 sk.3 ='//LAB99    IF (RC LT 5) THEN                                  '
 sk.4 ='//START   EXEC PGM=IEBGENER                                   '
 sk.5 ='//SYSUT1   DD  DISP=SHR,DSN='$jcllib'('nextjob')              '
 sk.6 ='//SYSUT2   DD  SYSOUT=(,INTRDR)                               '
 sk.7 ='//SYSIN    DD  DUMMY                                          '
 sk.8 ='//SYSPRINT DD  SYSOUT=X                                       '
 sk.9 ='//LAB99END  ENDIF                                             '
 sk.1Ø='//* ------------------------------------------------ *      '
    sk.Ø=1Ø ; Call WriteRec ; end
 say '' ; say '>>>>>>>>  +----------------------------------------+'
 say '>>>>>>>>    TOTAL Runstat job     :' right(#numjob,6,' ')
 say '>>>>>>>>    Total Tablespace      :' right(sysprint.Ø,6,' ')
 say '>>>>>>>>  +----------------------------------------+' ; say ''
 if sidispl = YES then Call DBdispl
   /*—  Submit Runstat job                    —*/
 if autosub =  YES then do
    xx=outtrap(trpØ8.) ; address tso "submit '"outdsrun"'"
    xx=outtrap(off)
    if rc > Ø then do
       do a = 1 to trpØ8.Ø ; say trpØ8.a ; end ; exit ; end
    say '' ; say time()
    say time() '——>> '#numjob' Runstats job ....'
    say time() ; say '' ; say ''
    xx=outtrap(trpdummy.) ; address tso "delete '"outdsrun"'"
    xx=outtrap(off) ; end ; call Free ; exit
   /*—  Write header job                      —*/
 Hdrjob:
    #numjob = #numjob + 1 ; dbcen = center(datab,1Ø)
    jn = RUN || subsys || #jnr
    if comdb1 ¬= SYSTEM then jn = userid() || R
sk.1='//'jn' JOB ('$accn'),''Runstats database'',CLASS='$class',      '
sk.2='//      MSGCLASS='$msgcla',USER='$user',REGION='$region',       '
sk.3='//      MSGLEVEL=('$msglvl'),NOTIFY='$notif
sk.4='/*JOBPARM BYTES=999999,LINES=9999                              '
sk.5='//* ------------------------------------------------ *      '
sk.6='//* ——     R u n s t a t s  'dbcen'       —— *          '
sk.7='//* ------------------------------------------------ *      '
sk.8='//JOBLIB   DD DISP=SHR,DSN='$dsnload
sk.9='//'datab' EXEC PGM=DSNUTILB,PARM='''subsys',RU'time('s')datab'''
sk.1Ø='//SYSPRINT DD SYSOUT=*                                        '
sk.11='//SYSIN    DD *                                               '
    sk.Ø=11 ; Call WriteRec ; return
   /*—  Display DataBase                      —*/
 DBdispl :
    jobw = sysprint ; "alloc da('"outds2"') f("jobw") shr reuse"
    jobw = sysin ; "alloc da('"outdsØ"') f("jobw") shr reuse"
    sk.1='SELECT DISTINCT DBNAME FROM VDB'subsys' ; '
    sk.Ø=1 ; Call WriteRec ; address tso "ex '"outds1"'"
```

```
    "ispexec edit dataset('"outds2"') macro(@mdb2Ø3Ø)"
    xx=outtrap(trpdummy.)
       address tso "printoff ('"outds2"')  class(R)"
    xx=outtrap(off)
    "execio * diskr sysprint (stem sysprint. finis"
 say '' ; say '          +------------------------------+'
 say '              Summary DataBase               '
 say '          +------------------------------+'
    DO #d = 1 to sysprint.Ø
       dbdis  = word(sysprint.#d,2)
       say   '                      -' dbdis
       end ; say   '' ; say  '' ; return
   /*—  Write record routine                —*/
 WriteRec :
    "EXECIO * DISKW "jobw" (STEM sk. FINIS"
 ClearRec:
    DO #f = 1 to sk.Ø ; sk.#f = blk ; end ; return
   /*—  Free Work areas                     —*/
 Free    :
    xx=outtrap(trpdummy.) ; "free fi(sysin)"
        address tso "delete '"outdsØ"'" ; "free fi(systsinp)"
        address tso "delete '"outds1"'"
        address tso "delete '"outds2"'" ; "free fi(firun)"
    xx=outtrap(off);  return
```

$DB2UTIL REXX EXEC

```
/* REXX */
   /*-------------------------------------------------------------------*/
   /*-                  Data Tool for DB management                    -*/
   /*-                   DB2 command generator                         -*/
   /*-------------------------------------------------------------------*/
arg parmin
parm  = translate(parmin,' ',',') ; nparm  = words(parm)
subsys = word(parm,1) ;datab = word(parm,2) ;creat = word(parm,3)
creatp = word(parm,4) ;tip   = word(parm,5) ;hold  = word(parm,6)
jn     = word(parm,7)
   /*—  Test input parameters               —*/
 if nparm < 7 then do
    say '' ; say '' ; say '>>>>>>>>'
    say '>>>>>>>>  Parameter string is incomplete !!!!'
    say '>>>>>>>>           'parmin
    say '>>>>>>>>' ; say '' ; say '' ; exit ; end
   /*—  Parameters assignment               —*/
 call @db2parØ subsys ;if word(result,1) = 99 then  exit
 $lpar =word(result,1) ; $accn =word(result,2) ; $class =word(result,3)
 $msgcla = word(result,4);$region  = word(result,5) ;$msglvl =
word(result,6)
 $notif  = word(result,7);$user    = word(result,8) ;$unitda =
word(result,9)
```

```
 $unitta = word(result,10);$esunit = word(result,11);$prt    =
word(result,12)
 $hiwork = word(result,13);$db2ver = word(result,14);$ctsubs =
word(result,15)
 $librexx= word(result,16);$parmlib= word(result,17);$proclib=
word(result,18)
 $jcllib = word(result,19);$report = word(result,20);$libexec=
word(result,21)
 $isptenu= word(result,22);$isppenu= word(result,23);$ispmenu=
word(result,24)
 $ispslib= word(result,25);$plilink= word(result,26);$sibmlnk=
word(result,27)
 $sortlib= word(result,28);$hilvlDB= word(result,29);$runlib =
word(result,30)
 $dsnload= word(result,31);$tep2pgm= word(result,32);$tep2pln=
word(result,33)
 $unlopgm= word(result,34);$unlopln= word(result,35);$dunlopg=
word(result,36)
 $dunlopl= word(result,37);$dsnproc= word(result,38)
   /*—   Work areas initialization          —*/
 blk =  ;jna = userid()
 if jn   = '*' then jn =  jna || U
   /*—  SYSIN file allocation               —*/
 say '' ;outds0= $hiwork'.'subsys'.'datab'.SYSIN'
 prmalloc = subsys' 'outds0' 0 5,1 f,b 80 2720 sysin si'
 call @db2all0 prmalloc ; if word(result,1) = 99 then exit
 jobw = sysin ; "alloc da('"outds0"') f("jobw") mod reuse"
 if tip = DROP then do
sk.1='   SELECT DBNAME,NAME                                          '
sk.2='          FROM SYSIBM.SYSTABLESPACE                            '
sk.3='          WHERE DBNAME  =  '''datab'''' ';                     '
    sk.0=3 ; Call WriteRec ; end
 if tip = GRANT | tip = SINON then do
sk.1='   SELECT CREATOR,NAME                                         '
sk.2='          FROM SYSIBM.SYSTABLES                                '
sk.3='          WHERE CREATOR =  '''creat'''' ' AND                  '
sk.4='                DBNAME  =  '''datab'''' ';                     '
    sk.0=4 ; Call WriteRec ; end
   /*—  SYSTSINP file allocation            —*/
 outds1= $hiwork'.'subsys'.'datab'.SYSTSINP'
 prmalloc = subsys' 'outds1' 0 5,1 f,b 80 2720 systsinp si'
 call @db2all0 prmalloc ; if word(result,1) = 99 then exit
sk.1='DSN SYSTEM('subsys')                                          '
sk.2='RUN PROGRAM('$tep2pgm') PLAN('$tep2pln') LIB('''$runlib''')   '
sk.3='END                                                           '
 sk.0=3 ; "execio * diskw systsinp (stem sk. finis" ; Call ClearRec
   /*—  SYSPRINT file allocation            —*/
 outds2= $hiwork'.'subsys'.'datab'.SYSPRINT'
 prmalloc = subsys' 'outds2' 0 45,15 f,b,a 121 1210 sysprint si'
 call @db2all0 prmalloc ;if word(result,1) = 99 then exit
   /*—  Utility file allocation             —*/
```

```
outdsutl= $hiwork'.'subsys'.'datab'.'tip
prmalloc = subsys' 'outdsutl' 0 30,15 f,b 80 27920 fiutl si'
call @db2all0 prmalloc ;if word(result,1) = 99 then exit
  /*-  DB2 select                              -*/
say ''
say ' ********************************************************** '
say '*                                              'time()'     *'
say '*       DB2 query is running                                 *'
say '*                                                          *'
say '*                         wait please .........            *'
say ' ********************************************************** '
say ''
xx=outtrap(trp05.) ; address tso "ex '"outds1"'" ;xx=outtrap(off)
if rc > 0 then do
   analisi = substr(trp05.1,1,8)
   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 trp05.0
         say trp05.a ; end ; end ; exit ; end
say '-->>>  DB2 query ended. Start elaboration ' ; say ''
  /*-  Print DB2 output query               -*/
xx=outtrap(trpdummy.) ; address tso "printoff ('"outds2"')  class(R)"
xx=outtrap(off)
xx=OUTTRAP(trp06.)
   "ispexec edit dataset('"outds2"') macro(@mdb2001)"
   if tip = DROP then
      "ispexec edit dataset('"outds2"') macro(@mdb2011)"
   if tip = GRANT | tip = SINON then
      "ispexec edit dataset('"outds2"') macro(@mdb2010)"
xx=OUTTRAP(OFF)
if rc > 0 then do
   do a = 1 to trp06.0
      say trp06.a ; end ; exit ; end
xx=outtrap(trp07.) ; "execio * diskr sysprint (stem sysprint. finis"
   "free fi(sysprint)" ; xx=outtrap(off)
if rc > 0 then do
   do a = 1 to trp07.0
      say trp07.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 '>>>>>>>>  file "'outds2'" is empty        '
   say '>>>>>>>>  probably DB2 K.O.  access !!!!  '
   say '>>>>>>>>' ; say '' ; say '' ; exit ; end
```

```
   /*—  Write header job                        —*/
 jobw = fiutl ; "alloc da('"outdsutl"') f("jobw") mod reuse"
sk.1='//'jn' JOB ('$accn'),''DB2-Gestione'',CLASS='$class',           '
sk.2='//      MSGCLASS='$msgcla',USER='$user',REGION='$region',       '
       if hold = 'SI' then
sk.3='//      MSGLEVEL=('$msglvl'),NOTIFY='$notif',TYPRUN=HOLD        '
       else
sk.3='//      MSGLEVEL=('$msglvl'),NOTIFY='$notif
 db = center(datab,8) ; tp = center(tip,8)
sk.4='/*JOBPARM BYTES=999999,LINES=9999                               '
sk.5='//* ------------------------------------------------ *         '
sk.6='//* ------     '   tp   '    '   db   '          ------ *       '
sk.7='//* ------------------------------------------------ *         '
sk.8='//STEP00   EXEC PGM=IKJEFT01,DYNAMNBR=20                        '
sk.9='//STEPLIB  DD  DISP=SHR,DSN='$dsnload
sk.10='//         DD  DISP=SHR,DSN='$plilink
sk.11='//         DD  DISP=SHR,DSN='$sibmlnk
sk.12='//SYSTSPRT DD  SYSOUT=*                                        '
sk.13='//SYSPRINT DD  SYSOUT=*                                        '
sk.14='//SYSTSIN  DD  *                                               '
sk.15=' DSN SYSTEM('subsys')                                          '
sk.16=' RUN PROGRAM('$tep2pgm') PLAN('$tep2pln') LIB(''$runlib''')    '
sk.17='//SYSIN    DD  *                                               '
sk.18='                                                               '
 sk.0=18 ; Call WriteRec
 select
 when tip = DROP  then call DROPR
 when tip = GRANT then call GRANTR
 when tip = SINON then call SINONR
 otherwise
    say '' ;say '>>>>>>>>'
    say '>>>>>>>>  Function not recognized  !!!  '
    say '>>>>>>>>' ; say '' ; call free ; exit ; end
   /*—  Submit job                            —*/
 xx=outtrap(trp00.) ; address tso "submit '"outdsutl"'"
 xx=outtrap(off)
 if rc > 0 then do
    do a = 1 to trp00.0
       say trp00.a ; end ; exit ; end
 mess  = substr(trp00.1,1,22)
 say time()
 say time() '——>> 'mess' has been submitted ...... '
 say time() ; say '' ; say '' ; call Free ; exit
   /*—  Write record routine                  —*/
 WriteRec :
    "EXECIO * DISKW "jobw" (STEM sk. FINIS"
 ClearRec:
    DO f = 1 to sk.0
       sk.f = blk ; end ; return
   /*—  Free Work areas                       —*/
 Free    :
```

```
    xx=outtrap(trpdummy.)
       "free fi(sysin)" ; address tso "delete '"outdsØ"'"
       "free fi(systsinp)" ; address tso "delete '"outds1"'"
        address tso "delete '"outds2"'" ;"free fi(fiutl)"
    xx=outtrap(off) ; return
   /*—  Drop DataBase                        —*/
 Dropr   :
    jobw = fiutl ; "alloc da('"outdsutl"') f("jobw") mod reuse"
sk.1='                                                          '
    sk.Ø=1 ; Call WriteRec
    DO d = 1 to sysprint.Ø
       recin = translate(sysprint.d,' ','|')
       cre   = word(recin,1) ; tssp  = word(recin,2)
      jobw = fiutl  ; "alloc da('"outdsutl"') f("jobw") mod reuse"
sk.1='        DROP TABLESPACE 'cre'.'tssp ' ;                   '
sk.2='        COMMIT;                                           '
       sk.Ø=2 ; Call WriteRec ; end
    say '>>>>>>>>  +-------------------------------------+'
    say '>>>>>>>>   TOTAL  Tablespace          :' sysprint.Ø
    say '>>>>>>>>  +-------------------------------------+'
    say '' ; return
   /*—  Grant job                            —*/
 Grantr  :
    jobw = fiutl ; "alloc da('"outdsutl"') f("jobw") mod reuse"
sk.1=' SET CURRENT SQLID = '''creat''' ';                       '
sk.2='       GRANT ALL ON TABLE                                 '
    sk.Ø=2 ; Call WriteRec
    DO d = 1 to sysprint.Ø
       recin = translate(sysprint.d,' ','|')  ;tab   = word(recin,2)
       jobw = fiutl ; "alloc da('"outdsutl"') f("jobw") mod reuse"
       if d < sysprint.Ø then
sk.1='                    'tab'  ,                              '
       else
sk.1='                    'tab'  TO  'creatp ';                 '
       sk.Ø=1 ; Call WriteRec ; end
    say '>>>>>>>>  +-------------------------------------+'
    say '>>>>>>>>   TOTAL  Tables              :' sysprint.Ø
    say '>>>>>>>>  +-------------------------------------+'
    say '' ; return
   /*—  Synonyms job                         —*/
 Sinonr  :
    jobw = fiutl ; "alloc da('"outdsutl"') f("jobw") mod reuse"
sk.1=' SET CURRENT SQLID = '''creatp''' ';                      '
    sk.Ø=1 ; Call WriteRec
    DO d = 1 to sysprint.Ø
       recin = translate(sysprint.d,' ','|') ; tab = word(recin,2)
       jobw = fiutl ; "alloc da('"outdsutl"') f("jobw") mod reuse"
sk.1=' CREATE SYNONYM  'tab' FOR 'creat'.'tab' ;                '
       sk.Ø=1 ; Call WriteRec ;end
    say '>>>>>>>>  +-------------------------------------+'
    say '>>>>>>>>   TOTAL Synonyms             :' sysprint.Ø
```

```
    say '>>>>>>>>  +-------------------------------------+'
    say '' ; return
```

$DB2FREE REXX EXEC

```
/* REXX */
   /*-------------------------------------------------------------------*/
   /*-                  Data Tool for DB managment                     -*/
   /*-                    Free statement Generator                     -*/
   /*-------------------------------------------------------------------*/
arg parmin ; parm = translate(parmin,' ',',') ; nparm  = words(parm)
subsys = word(parm,1) ; creat  = word(parm,2) ; autosub= word(parm,3)
jn     = word(parm,4)
   /*—  Test input parameters                —*/
 if nparm < 4 then do
    say '' ; say '' ; say '>>>>>>>>'
    say '>>>>>>>>  Parameter string is incomplete !!!!'
    say '>>>>>>>>           'parmin
    say '>>>>>>>>' ;say '' ;say '' ;exit ; end
 if autosub = '*' then autosub = NO
  lcreat = length(creat) ; ljn = length(jn)
  if jn = '*' & lcreat < 6 then jn = userid()'R'
  if jn ¬= '*' & ljn < 8 then jn = userid()'R'
  if jn   = '*' then
     jna = FRE||substr(creat,1,2)||substr(creat,6,1)||substr(creat,4,2)
  else jna = jn
   /*—  Parameters assignment               —*/
 call @db2par0 subsys ;if word(result,1) = 99 then exit
 $lpar =word(result,1) ; $accn =word(result,2) ; $class =word(result,3)
 $msgcla = word(result,4);$region  = word(result,5) ;$msglvl =
word(result,6)
 $notif  = word(result,7);$user    = word(result,8) ;$unitda =
word(result,9)
 $unitta = word(result,10);$esunit = word(result,11);$prt    =
word(result,12)
 $hiwork = word(result,13);$db2ver = word(result,14);$ctsubs =
word(result,15)
 $librexx= word(result,16);$parmlib= word(result,17);$proclib=
word(result,18)
 $jcllib = word(result,19);$report = word(result,20);$libexec=
word(result,21)
 $isptenu= word(result,22);$isppenu= word(result,23);$ispmenu=
word(result,24)
 $ispslib= word(result,25);$plilink= word(result,26);$sibmlnk=
word(result,27)
 $sortlib= word(result,28);$hilvlDB= word(result,29);$runlib =
word(result,30)
 $dsnload= word(result,31);$tep2pgm= word(result,32);$tep2pln=
word(result,33)
 $unlopgm= word(result,34);$unlopln= word(result,35);$dunlopg=
```

```
word(result,36)
 $dunlopl= word(result,37);$dsnproc= word(result,38)
   /*-   Work areas initialization           -*/
blk  =  ; wrk =  ;sisub = no ; build = YES ; okdel = 0
   /*-  Free Plan file allocation            -*/
 say '' ; outdsrbn= $hiwork'.'subsys'.'creat'.FREE'
 prmalloc = subsys' 'outdsrbn' 0 15,15 f,b 80 27920 firbn YES'
 call @db2all0 prmalloc ; if word(result,1) = 99 then exit
   /*-  SYSTSINP file allocation             -*/
 outds1= $hiwork'.'subsys'.'creat'.FREE.SYSTSINP'
 prmalloc = subsys' 'outds1' 0 5,1 f,b 80 27920 systsinp YES'
 call @db2all0 prmalloc ;if word(result,1) = 99 then exit
sk.1='DSN SYSTEM('subsys')                                              '
sk.2='RUN PROGRAM('$tep2pgm') PLAN('$tep2pln') LIB('''$runlib''')    '
sk.3='END                                                               '
 sk.0=3; "execio * diskw systsinp (stem sk. finis" ;call ClearRec
 do i = 1 to 2
   /*-  PlanName select                   ---*/
    if i = 1 then do
       tipsel = FRPLAN ; call Allsysp ; call Allsysi
       jobw = sysin ; "alloc da('"outds0"') f("jobw") shr reuse"
sk.1=' SELECT * FROM V'creat'_FRPLAN ;                                  '
       sk.0=1;call WriteRec ;call Rundb2; macnr = @mdb2025 ;
       call Exmacro ;call Testout
       if build = YES then do
          sisub = YES ; call WFRPLAN
          say '' ; say '>>>>>>>>'
          say '>>>>>>>>  +------------------------------+'
          say '>>>>>>>>    Total Free   Plan  n>' e - 1
          say '>>>>>>>>  +------------------------------+'
          say '>>>>>>>>' ; say '' ; say '' ; end
       else do
          build = YES ; okdel = okdel + 1
          say '' ; say '>>>>>>>>'
          say '>>>>>>>> FRPLAN select has no rows !!! '
          say '>>>>>>>> No Plan to be free.'
          say '>>>>>>>>' ; say '' ; end ; end
    if i = 2 then do
       tipsel = FRPACK ;call Allsysp ;"free fi(sysin)" ;call Allsysi
       jobw = sysin ; "alloc da('"outds0"') f("jobw") shr reuse"
sk.1=' SELECT * FROM V'creat'_FRPACK ;                                  '
       sk.0=1 ;call WriteRec;call Rundb2;macnr = @mdb2026
       call Exmacro ;call Testout
       if build = YES then do
          sisub = YES ; call WFRPACK
          say '' ;say '>>>>>>>>'
          say '>>>>>>>>  +------------------------------+'
          say '>>>>>>>>    Total Free   Package n>' f - 1
          say '>>>>>>>>  +------------------------------+'
          say '>>>>>>>>' ; say '' ; say '' ; end
       else do
```

```
         build = YES ; okdel = okdel + 1
         say '' ; say '>>>>>>>>'
         say '>>>>>>>> FRPACK select has no rows !!! '
         say '>>>>>>>> No Package to be free.'
         say '>>>>>>>>' ; say '' ; end ; end ; end
   /*—  Automatic job submit                —*/
if autosub = YES then do
   if sisub = YES then do
      xx=OUTTRAP(trpØ8.) ;address tso "submit '"outdsrbn"'"
      xx=OUTTRAP(OFF)
      if rc > Ø then do
         do a = 1 to trpØ8.Ø
            say trpØ8.a ; end ; exit ; end
      mess  = substr(trpØ8.1,1,22)
      say '' ; say time()
      say time() '——>> 'mess' has been submitted .... '
      say time() ; say '' ; say '' ; end
   else do
      say '' ; say '>>>>>>>>'
      say '>>>>>>>> No Plan/Package for user 'creat '!!!!'
      say '>>>>>>>>' ; say '' ; end
   xx=outtrap(trpdummy.) ; address tso "delete '"outdsrbn"'"
   xx=outtrap(off) ; end ; call Free ;exit
  /*—  SYSIN file allocation               —*/
Allsysi:
   outdsØ= $hiwork'.'subsys'.'creat'.'tipsel'.SYSIN'
   prmalloc = subsys' 'outdsØ' Ø 5,1 f,b 8Ø 2792Ø sysin YES'
   call @db2allØ prmalloc ;if word(result,1) = 99 then exit
   say '' ;return
  /*—  SYSPRINT file allocation            —*/
Allsysp:
   outds2= $hiwork'.'subsys'.'creat'.'tipsel'.SYSPRINT'
   prmalloc = subsys' 'outds2' Ø 15Ø,6Ø f,b,a 121 121Ø sysprint si'
   call @db2allØ prmalloc ;if word(result,1) = 99 then  exit ; return
Rundb2 :
   wrk = center(tipsel,8)
   say ' ********************************************************** '
   say '* ' i '                                        'time()'     *'
   say '*          Query 'wrk    'in progress                      *'
   say '*                                                          *'
   say '*               wait please ......                         *'
   say '*                                                          *'
   say ' ********************************************************** '
   say ''
   xx=outtrap(trp11.) ;address tso "ex '"outds1"'"
   xx=outtrap(off)
   if rc > Ø then do
      analisi = substr(trp11.1,1,8)
      if analisi = 'DSNE1ØØI' then do
         subs = center(subsys,1Ø)
```

```
         say '' ; say '' ; say '>>>>>>>>' ;say '>>>>>>>>'
       say '>>>>>>>>   The DB2 subsystem –>'subs'<– is not active !!!'
         say '>>>>>>>>'; say '>>>>>>>>' ; say '' ; say '' ; end
      else do
         do a = 1 to trp11.Ø
            say trp11.a ;end ;end ;exit ;end
      say '>>>>>>>>  DB2 query ended' ; say ''
  /*–  Print DB2 output query              –*/
   xx=outtrap(trpdummy.);address tso "printoff ('"outds2"')  class(R)"
   xx=outtrap(off) ; return
Exmacro:
   xx=OUTTRAP(trp12.)
      "ispexec edit dataset('"outds2"') macro("macnr")"
   xx=OUTTRAP(OFF)
   if rc > Ø then do
      do a = 1 to trp12.Ø
         say trp12.a ;end ;exit ;end ;return
Testout:
   xx=outtrap(trp13.)
      "execio * diskr sysprint (stem sysprint. finis"
      "free fi(sysprint)" :xx=outtrap(off)
   if rc > Ø then do
      do a = 1 to trp13.Ø
         say trp13.a ;end
      say '' ;say '' ;say '>>>>>>>>'
      say '>>>>>>>>  Error reading file "'outds2'"'
      say '>>>>>>>>  RC='rc'. Verify'
      say '>>>>>>>>' ;say '' ;say '' ;exit ;end
   if sysprint.Ø = Ø then do
      say '' ;say '' ;say '>>>>>>>>'
      say '>>>>>>>>  file "'outds2'" is empty        '
      say '>>>>>>>>  probably DB2 K.O.  access !!!!  '
      say '>>>>>>>>' ;say '' ;say '' ;exit ;end
   if substr(sysprint.1,33,8) = ' Ø ROW(S)' then build = no ; return
  /*–  Free Plan statment building          –*/
WFRPLAN:
   say '>>>>>>>>  Building job of free   Plan '
   jobw = firbn ;"alloc da('"outdsrbn"') f("jobw") mod reuse"
   call Hdrbnd
   DO e = 1 to sysprint.Ø
      pla = word(sysprint.e,1) ;lpla = length('FREE PLAN('''pla''')')
      wrk = 5Ø - lpla
sk.e='FREE PLAN('pla')'copies(' ',wrk)
      end
sk.e='//* ------------------------------------------------ *       '
   sk.Ø = e ;call WriteRec ;return
  /*–  Free Package statment building       –*/
WFRPACK:
   say '>>>>>>>>  Building job of free   Package '
   jobw = firbn ; "alloc da('"outdsrbn"') f("jobw") mod reuse"
```

```
    call Hdrbnd
    DO f = 1 to sysprint.Ø
       pak = word(sysprint.f,1) ;col = word(sysprint.f,2)
       ver = word(sysprint.f,3)
       if ver = blk then do
          lpak = length('FREE PACKAGE('''col''''.''''pak''')')
          wrk = 5Ø - lpak
sk.f='FREE PACKAGE('col'.'pak')'copies(' ',wrk)
          end
       else do
          lpak=length('FREE PACKAGE('col'.'pak'.('ver'))')
          wrk = 5Ø - lpak
sk.f='FREE PACKAGE('col'.'pak'.('ver'))'copies(' ',wrk)
          end ;end
sk.f='//* ------------------------------------------------- *       '
    sk.Ø = f ;call WriteRec ;return
   /*—  Write header bind job               —*/
 Hdrbnd:
    jobw = firbn ; "alloc da('"outdsrbn"') f("jobw") mod reuse"
sk.1='//'jna' JOB ('$accn'),''Bind job'',CLASS='$class',              '
sk.2='//      MSGCLASS='$msgcla',USER='$user',REGION='$region',       '
sk.3='//      MSGLEVEL=('$msglvl'),NOTIFY='$notif
sk.4='/*JOBPARM BYTES=999999,LINES=9999                               '
sk.5='//* ------------------------------------------------- *       '
sk.6='//*                  Free    JOB                      *       '
sk.7='//* ------------------------------------------------- *       '
sk.8='//JOBLIB    DD  DISP=SHR,DSN='$dsnload
sk.9='//BIND     EXEC  PGM=IKJEFTØ1,DYNAMNBR=1ØØ                     '
sk.1Ø='//SYSTSPRT  DD  SYSOUT=*                                     '
sk.11='//REPORT    DD  SYSOUT=*                                     '
sk.12='//SYSTSIN   DD  *                                            '
sk.13='DSN SYSTEM('subsys')'
    sk.Ø = 13 ;call WriteRec ;return
   /*—  Write record routine                —*/
 WriteRec :
    "EXECIO * DISKW "jobw" (STEM sk. FINIS"
 ClearRec:
    DO g = 1 to sk.Ø
       sk.g = blk ; end ;return
   /*—  Free Work areas                     —*/
 Free    :
     xx=outtrap(trpdummy.)
        tipsel = FRPLAN
        outdsØ= $hiwork'.'subsys'.'creat'.'tipsel'.SYSIN'
        outds2= $hiwork'.'subsys'.'creat'.'tipsel'.SYSPRINT'
        address tso "delete '"outdsØ"'"
        address tso "delete '"outds2"'"
        tipsel = FRPACK
        outdsØ= $hiwork'.'subsys'.'creat'.'tipsel'.SYSIN'
        outds2= $hiwork'.'subsys'.'creat'.'tipsel'.SYSPRINT'
```

```
        address tso "delete '"outdsØ"'"
        address tso "delete '"outds2"'"
       "free fi(systsinp)" ;address tso "delete '"outds1"'"
        "free fi(firbn)"
        if okdel = 2 then address tso "delete '"outdsrbn"'"
     xx=outtrap(off) ;return
```

$MDB2001 EDIT MACRO

```
/* REXX */
trace ?o
/*-----------------------------------------*/
/*——    Used in REXX @DB2UTIL            ——*/
/*-----------------------------------------*/
isredit macro
isredit exclude all
isredit find '''_|''' all
isredit delete x all
isredit end
```

$MDB2010 EDIT MACRO

```
/* REXX */
trace ?o
/*-----------------------------------------*/
/*——    Used in REXX @DB2UTIL            ——*/
/*-----------------------------------------*/
isredit macro
isredit change P'''=====''' '''     ''' 49 all
isredit save
isredit end
```

$MDB2011 EDIT MACRO

```
/* REXX */
trace ?o
/*-----------------------------------------*/
/*——    Used in REXX @DB2UTIL            ——*/
/*-----------------------------------------*/
isredit macro
isredit change P'''=====''' '''     ''' 54 all
isredit save
isredit end
```

$MDB2014 EDIT MACRO

```
/* REXX */
```

```
trace ?o
/*-----------------------------------------*/
/*——    Used in REXX @DB2ALIØ            ——*/
/*-----------------------------------------*/
isredit macro
isredit exclude all
isredit find ALIAS   2
isredit find '''IN-CAT'''  7
isredit delete x all
isredit save
isredit end
```

$MDB2028 EDIT MACRO

```
/* REXX */
trace ?o
/*-----------------------------------------*/
/*——    Used in REXX @DB2RUNS            ——*/
/*-----------------------------------------*/
isredit macro
isredit exclude all
isredit find '''_|''' all
isredit find ''' Ø ROW'''
isredit delete x all
isredit change P'''======='''  '''      '''  52 all
isredit change '''|'''  ''' '''  all
isredit save
isredit end
```

$MDB2030 EDIT MACRO

```
/* REXX */
trace ?o
/*-----------------------------------------*/
/*——    Used in REXX @DB2RUNS            ——*/
/*-----------------------------------------*/
isredit macro
isredit exclude all
isredit find '''_|''' all
isredit find ''' Ø ROW'''
isredit delete x all
isredit change ''' | '''  '''   '''  all
isredit change '''_| '''  '''   '''  all
isredit save
isredit end
```

IBM has announced a new on-line demo for software developers showing off its DB2 database support of XML and Web Services for information integration. The preview is based on the company's Xperanto information integration project and is part of the company's efforts to combine emerging XML and XQuery standards with DB2's federation capabilities to address integration challenges in a real customer scenario.

The federated data management approach enables customers to search, access, analyse, and aggregate data and pose queries to it across a range of information ranging from relational databases, XML documents, flat files, spreadsheets, and Web services.

The demo provides a scenario of how a newly-merged bank and financial services company could use XML standards as a single interface to provide integrated views of multiple databases and Web services to users.

Using XQuery, the Xperanto demo can deliver relational data to the XML world, access real-time data on the Internet using Web services, view and query diverse databases and Web services as if they were a single database, and apply text search uniformly across federated data.

For further information contact your local IBM representative.
URL: http://www.ibm.com/software/data/developer.

* * *

IBM has announced DB2 OLAP Server Version 8.1. Based on Hyperion Essbase 6.5, it is said to incorporate scalability, performance, ease of use, and administration improvements over Version Version 7.1.

A logical cube can now be defined to have a lower portion of the cube residing in the relational database. The OLAP Server will retrieve the cells from the relational database as if they physically reside on the cube storage. Hybrid analysis is said to eliminate the need to load and store members and their data with the physical cube itself.

Meanwhile, OLAP Miner combines IBM data mining tools with OLAP to help discover anomalies and special segments from current data.

For further information contact your local IBM representative.
URL: http://www.ibm.com/software.

* * *

Candle has announced immediate support for z/OS 1.3 and reaffirmed its support of IBM's Workload Licence Charges software pricing structure for DB2, CICS, and IMS.

Day-one support includes OMEGAMON II for MVS, CICS, DB2, IMS, DBCTL, SMS and mainframe networks; OMEGAMON XE for Sysplex, CICSplex, DB2plex, IMSplex, UNIX Systems Services, OS/390, CICS and DB2; OMEGAMON DE; DB/EXPLAIN, DB/DASD, DB/SMU, DB/WORKBENCH, DB/QUICKCHANGE and DB/QUICK COMPARE; OMEGAVIEW TN3270, OMEGACENTER Gateway, AF/OPERATOR and AF/REMOTE; OMEGAMON XE Management Pac for MQSeries; CL/SUPERSESSION and CL/CONFERENCE; and PQEdit for MQSeries.

For further information contact:
Candle, 201 N Douglas St, El Segundo, CA 90245, USA.
Tel: (310) 535 3600.
URL: http://www.candle.com.