125

DB2 update

March 2003

In this issue

3 Formatting the LIST TABLESPACE output

8 DB2 log inventory report

26 Issuing a SELECT from a Windows BAT file

27 The importance of PCTFREE and FREEPAGE

44 An environment-independent DB2 language interface

54 DB2 news

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

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.

Formatting the LIST TABLESPACE output

The DB2 UDB **list tablespace show detail** command provides useful output on how tablespaces in a database have been defined, but this output is cumbersome to read if there are lots of tablespaces. The following code was written in Object REXX to run on a Windows 2000 machine to produce a more readable version of the output. It was run against DB2 UDB V8.1 and produced the output shown below. I also ran it successfully against a V7.2 database on AIX. The output is suitable for importing into a spreadsheet or Word document (using the vertical bar character (|) as a delimiter).

The column descriptions are: TSi, tablespace ID; Name, tablespace name; T, tablespace type (this will be 'S' for SMS and 'D' for DMS); Sta, tablespace state (see the *Command Reference* manual under the list tablespaces command for a list of all the different states); TP, total pages allocated; UsePg, total usable pages; UsedPg, used pages; %Us, percentage of pages used; FreePg, free pages available; HighPg, high water mark (pages); Pag, page size (bytes); Ext, extent size (pages); Pre, prefetch size (pages); NC, number of containers.

If the tablespace has only one container, the information is displayed in the main table. If it consists of more than one container, the container information is written out in a separate section, with a pointer from the main table.

You have to be in the CLP environment and connected to the database for which you want the information before running the REXX.

```
>rexx tslistØ3.txt
- Have issued list tablespace show detail command. Number of lines read:
141
=========================================================================
----------------- This is the container information ------------------
============ Type: P-path (sms), F-file (dms), D-disk (raw) ============
=========================================================================

Container path for tsid   Ø is accessible? Y and is type: P - route:
```

```
C:\DB2\NODEØØØØ\SQLØØØØ1\SQLTØØØØ.Ø
Container path for tsid   1 is accessible? Y and is type: P - route:
C:\DB2\NODEØØØØ\SQLØØØØ1\SQLTØØØ1.Ø
Container path for tsid   2 is accessible? Y and is type: P - route:
C:\DB2\NODEØØØØ\SQLØØØØ1\SQLTØØØ2.Ø
Container path for tsid   3 is accessible? Y and is type: F - route:
C:\dmsØ3

=======================================================================
----------------- This is the tablespace information -----------------
=======================================================================

|TSi|--Name----|T|-Sta--|--TP--|-UsePg-|UsedPg-|%Us|FreePg-|HighPg-|-
Pag-|Ext|Pre|NC-|
|  Ø|SYSCATSPACE|S|ØxØØØØ|   4656|   4656|   4656|1ØØ|    N/A|    N/A|
4Ø96| 32| 16|  1|C:\DB2\NODEØØØØ\SQLØØØØ1\SQLTØØØØ.Ø| P| Y|
|  1| TEMPSPACE1|S|ØxØØØØ|      1|      1|      1|1ØØ|    N/A|    N/A|
4Ø96| 32| 16|  1|C:\DB2\NODEØØØØ\SQLØØØØ1\SQLTØØØ1.Ø| P| Y|
|  2| USERSPACE1|S|ØxØØØØ|   9199|   9199|   9199|1ØØ|    N/A|    N/A|
4Ø96| 32| 16|  1|C:\DB2\NODEØØØØ\SQLØØØØ1\SQLTØØØ2.Ø| P| Y|
|  3|      DMSØ1|D|ØxØØØØ|  512ØØ|  51184|   1952|  4|  49232|   1952|
4Ø96| 16| 16|  1|C:\dmsØ3                           | F| Y|
```

The %Us column is the percentage of space used in the tablespace. Obviously, for SMS tablespaces this figure will always be 100%, but, for DMS tablespaces, it is worth monitoring this column for values nearing 100%.

```
/*********************************************************************/
/* This EXEC will produce a more readable version of                */
/* the list tablespace output.                                      */
/*********************************************************************/
 /* You run this EXEC from the DB2 CLP and you must be      */
 /* connected to the database you want the information for. */
  /* Issue the command to get the tablespace detail. */
  i = Ø
  queue_name =rxqueue('Create')
  Call rxqueue 'Set',queue_name
  interpret "'db2 list tablespaces show detail | RXQUEUE' queue_name"
  jk = Ø
  Do queued() - 1
    jk = jk + 1
    parse pull linn.jk
  End /* Do queued() - 1  */
  Call rxqueue 'DELETE',queue_name
  linn.Ø = jk
  say DATE() TIME() "- Have issued list tablespace show detail command.
Number of lines read:" linn.Ø
 /* When you issue the list tablespaces show detail command         */
 /* you get the following information. These are the fields we will */
 /* be looking for.                                                  */
 possv.1  = "Tablespace"
 possv.2  = "Name"
 possv.3  = "Type"
 possv.4  = "State"
 possv.5  = "Total"
 possv.6  = "Useable"
 possv.7  = "Used"
 possv.8  = "%Used"
 possv.9  = "Free"
 possv.1Ø  = "High"
 possv.11 = "Page"
 possv.12 = "Extent"
 possv.13 = "Prefetch"
 possv.14 = "Number"
 possv.Ø  = 14
 js = Ø
 Do jk = 1 to linn.Ø
   Do jk2 = 1 to possv.Ø
     If (Subword(linn.jk,1,1) = possv.jk2) then Do
       If (jk2 = 1) then Do
         js = js + 1
       End /* If (jk2 = 1) then Do */
       parse var linn.jk rubb "=" val.js.jk2
       If (subword(val.js.jk2,1,1) = "System") then Do
         val.js.jk2 = "S"
       End /* If (subword(val.js.jk2,1,1) = "System") then Do */
       If (subword(val.js.jk2,1,1) = "Database") then Do
         val.js.jk2 = "D"
       End /* If (subword(val.js.jk2,1,1) = "Database") then Do */
       If (substr(val.js.jk2,1,2) = "Øx") then Do
         val.js.jk2 = substr(val.js.jk2,3)
       End /* If (substr(val.js.jk2,1,2) = "Øx") then Do */
       If (subword(val.js.jk2,1,1) = "Not") then Do
         val.js.jk2 = "N/A"
       End /* If (subword(val.js.jk2,1,1) = "Not") then Do */
       leave jk2
     End /* If (Subword(linn.jk,1,1) = possv.jk2) then Do */
   End /* Do jk2 = 1 to possv.Ø */
 End /* Do jk = 1 to linn.Ø */
 /* Get the container information for each tablespace id. */
 say copies("=",72)
 say center(" This is the container information ",72,"-")
 say center(" Type: P-path (sms), F-file (dms), D-disk (raw) ",72,"=")
 say copies("=",72)
 say " "
 /* js is the number of tablespaces */
 /* The format of the path array is:                                   */
/* path.<tablespace-number>.<container-number>.<container-
```

```
path>.<container-accessible> */
 /* For each tablepsace issue a show detail command. */
 Do jt = 1 to js
   tsid = val.jt.1
   queue_name =rxqueue('Create')
   Call rxqueue 'Set',queue_name
  interpret "'db2 list tablespace containers for" tsid "show detail |
RXQUEUE' queue_name"
  jk = Ø
  Do queued() - 1
    jk = jk + 1
    parse pull linn.jk
  End /* Do queued() - 1 */
  Call rxqueue 'DELETE',queue_name
  linn.Ø = jk
  pi = Ø  /* The number of containers for each tablespace. */
  /* Work out and print the container information. */
  Do jk = 1 to linn.Ø
    If (subword(linn.jk,1,1) = 'SQL1ØØ8N') then Do
      path.jt.1.1 = "Invalid T/S id"
      path.jt.1.2 = "--"
    End /* If (subword(linn.jk,1,1) = 'SQL1ØØ8N') then Do */
    If (subword(linn.jk,1,1) = 'Name') then Do
      pi = pi + 1
      parse var linn.jk "=" path.jt.pi.1
      path.jt.pi.1 = strip(path.jt.pi.1,B,' ')
    End /* If (subword(linn.jk,1,1) = 'Name') then Do */
    If (subword(linn.jk,1,1) = 'Type') then Do
      /* The possible values are: path, file, disk */
      parse var linn.jk "="  typv
      typv = strip(typv,B," ")
      path.jt.pi.2 = substr(typv,1,1)
    End /* If (subword(linn.jk,1,1) = 'Type') then Do */
    If (subword(linn.jk,1,1) = 'Accessible') then Do
      parse var linn.jk "=" path.jt.pi.3
      path.jt.pi.3 = strip(path.jt.pi.3,B,' ')
      path.jt.pi.3 = substr(path.jt.pi.3,1,1)
    End /* If (subword(linn.jk,1,1) = 'Accessible') then Do */
  End /* Do jk = 1 to linn.Ø */
  /* For each container, print out the information. */
  Do jk = 1 to pi
    If (jk = 1) then Do
      say "Container path for tsid" right(tsid,3,' ') "is accessible?"
path.jt.jk.3 "and is type:" path.jt.jk[2 "- route: "path.jt.jk.1
    End /* If (jk = 1) then Do */
    Else Do
      say copies(" ",23) right(tsid,3,' ') "is accessible?" path.jt.jk.3
"and is type:" path.jt.jk.2 "path: "path.jt.jk.1
    End /* If (jk = 1) then Do */
  End /* Do jk = 1 to pi */
 End /* Do jt = 1 to js */
 /* This is the header line information. */
 hed.1  = "TSi"    ; len.1   =  3 /* Tablespace ID           */
 hed.2  = "Name"   ; len.2   = 11 /* Name                    */
 hed.3 = "T"       ; len.3   =  1 /* Type                    */
 hed.4 = "Sta"     ; len.4   =  6 /* State                   */
 hed.5 = "TP"      ; len.5   =  7 /* Total pages             */
 hed.6 = "UsePg"   ; len.6   =  7 /* Useable pages           */
 hed.7 = "UsedPg"  ; len.7   =  7 /* Used pages              */
 hed.8 = "%Us"     ; len.8   =  3 /* Percent used pages      */
 hed.9 = "FreePg"  ; len.9   =  7 /* Free pages              */
 hed.1Ø = "HighPg" ; len.1Ø  =  7 /* High water mark (pages) */
 hed.11 = "Pag"    ; len.11  =  5 /* Page size (bytes)       */
 hed.12 = "Ext"    ; len.12  =  3 /* Extent size (pages)     */
 hed.13 = "Pre"    ; len.13  =  3 /* Prefetch size (pages)   */
 hed.14 = "NC"     ; len.14  =  3 /* Number of containers    */
 hed.Ø = 14
 /* Write out the header line. */
 say " "
 say copies("=",72)
 say center(" This is the tablespace information ",72,"-")
 say copies("=",72)
 say " "
 tot.1 = "|"
 Do jk = 1 to hed.Ø
   tot.1 = tot.1 || center(hed.jk,len.jk,'-') || "|"
 End /* Do jk = 1 to hed.Ø */
 say tot.1
 /* Write out the main body of the text. */
 maxl = Ø
 Do jk = 1 to js
   len = length(path.jk.1.1)
   If (len > maxl) then Do
     maxl = len
   End /* If (len > maxl) then Do */
 End /* Do jk = 1 to js */
 Do jk = 1 to js
   lino.jk = "|"
   /* Work out the percent space used. */
   val.jk.8 = (val.jk.7 / val.jk.6) * 1ØØ
   val.jk.8 = format(val.jk.8,3,Ø)
   Do jk2 = 1 to hed.Ø
     val.jk.jk2 = strip(val.jk.jk2,B,' ')
     If (length(val.jk.jk2) > len.jk2) then Do
       val.jk.jk2 = substr(val.jk.jk2,1,len.jk2-1) || "*"
     End /* If (length(val.jk.jk2) > len.jk2) then Do */
     lino.jk = lino.jk|| right(val.jk.jk2,len.jk2,' ') || "|"
   End /* Do jk2 = 1 to hed.Ø */
   If (val.jk.14 = 1) then Do
     /* There is only 1 container for the tablespace. */
```

```
     lino.jk = lino.jk || left(path.jk.1.1,maxl,' ') || "|" path.jk.1.2
|| "|" path.jk.1.3 || "|"
   End /* If (val.jk.14 = 1) then Do */
   Else Do
     /* There is more than 1 container. */
     lino.jk = lino.jk || "See above for container information"
   End /* If (val.jk.14 = 1) then Do */
   say lino.jk
 End /* Do jk = 1 to js */
 exit
```

C Leonard
Freelance Consultant (UK)

DB2 log inventory report

INTRODUCTION

A DB2 database administrator may need to use the recovery process. In order to make things easier, I have created an on-line REXX tool that performs an inquiry on the bootstrap dataset and returns information that can be very useful for this process.

PARAMETER DESCRIPTION

The following parameters describe how you can customize the REXX EXEC on-line procedure:

- Subsys – the DB2 subsystem name.
- LOG TYPE – this field selects the type of DB2 log:
  - A to select the active log.
  - R to select the archived log dataset.

CHECKLIST FOR INSTALLATION

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

- Allocate a USER.LIBRARY.
- Copy all REXX, macro, panel, and help panels into the USER.LIBRARY:
  - REXX – $DB2PAR0, $DB2ALL0, $DB2TL00, $DB2TL0C.
  - Macro – $MDB2002.
  - Panel – $DB2P000, $DB2P012, $DB2P013, $DB2P014.
  - Help panel – $DB2H000, $DB2H00C.
- Customize $DB2PAR0 REXX for the global environment variables.

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

$DB2TL00 REXX EXEC

```
/* REXX */
trace ?o
   /*---------------------------------------------*/
   /*-     Management Tool for DB2 environment   -*/
   /*-               Main menu                   -*/
   /*---------------------------------------------*/
   /*--   Work areas initialization            ---*/
 blk      =
 wrexit   = ok
 curØØ    = '@db2subs'
 acc      = NON
 @db2subs = DSN?
 @db2msg  =
 @db2data = date(e)
 @db2tim  = time()
 RunEnv = 'ONLINE'
 address ispexec "display panel(@db2p$$$)"
 if rc = 8 then
   exit
 do forever
    @db2data = date(e)
    @db2tim  = time()
    wrexit = ok
    address ispexec "display panel(@db2pØØØ) cursor("curØØ")"
    @db2msg =
    if rc = 8 then
       leave
    call ParAssign
    if wrexit = ok then do
       acc = HIGH
       call ParAssign
       @db2ver = $db2ver
       end
    if wrexit = ok & @db2opt = blk then do
       @db2msg = 'Select one Option'
       curØØ = '@db2opt'
       wrexit = ko
       iterate
       end
    if wrexit = ok & @db2opt ¬= blk then do
       parmpass = @db2subs','@db2ver
       select
          when @db2opt = C then do      /*- Log Inventory Report  -*/
             call #DB2TLØC parmpass
             @db2msg = result
             @db2opt =
             end
          otherwise
             @db2msg = 'Unpredictable error contact Support Staff  !!!'
             @db2opt =
             iterate
          end
       end
    end
 exit
ParAssign :
   /*--  Parameters assignment                      */
 call @db2parØ @db2subs RunEnv
 if word(result,1) = 99 then do
    @db2msg = 'Wrong DB2 subsystem '@db2subs
    curØØ = '@db2subs'
    acc = NON
    @db2ver = blk
    wrexit = ko
    return
    end
 $lpar    = word(result,1)                 /*  LPAR                 */
 $accn    = word(result,2)                 /*  Account name         */
 $class   = word(result,3)                 /*  Class                */
 $msgcla  = word(result,4)                 /*  Message class        */
 $region  = word(result,5)                 /*  Region               */
 $msglvl  = word(result,6)                 /*  Message level        */
 $notif   = word(result,7)                 /*  Notify               */
 $user    = word(result,8)                 /*  User                 */
/**/
 $unitda  = word(result,9)                 /*  Type of unit dasd    */
 $unitta  = word(result,1Ø)                /*  Type of unit tape    */
 $esunit  = word(result,11)                /*  Esoteric work name   */
 $prt     = word(result,12)                /*  Printer name         */
 $hiwork  = word(result,13)                /*  Hi-level work areas  */
/**/
 $db2ver  = word(result,14)                /*  DB2 version          */
 $ctsubs  = word(result,15)                /*  Carattere subsystem  */
/**/
 $librexx = word(result,16)                /*  REXX       library   */
 $parmlib = word(result,17)                /*  Parmlib       "      */
 $proclib = word(result,18)                /*  Proclib       "      */
 $jcllib  = word(result,19)                /*  JCL           "      */
 $report  = word(result,2Ø)                /*  Report out    "      */
 $libexec = word(result,21)                /*  SysExec       "      */
/**/
 $isptenu = word(result,22)                /*  ISPF       library   */
 $isppenu = word(result,23)                /*   "            "      */
 $ispmenu = word(result,24)                /*   "            "      */
 $ispslib = word(result,25)                /*   "            "      */
/**/
 $plilink = word(result,26)                /*  PLI           "      */
 $sibmlnk = word(result,27)                /*   "            "      */
 $sortlib = word(result,28)                /*  Sort          "      */
/**/
 $runlib  = word(result,29)                /*  DB2 Runlib library   */
 $dsnload = word(result,3Ø)                /*  DB2 system    "      */
/**/
 $tep2pgm = word(result,31)                /*  DSNTEP2  (program)   */
 $tep2pln = word(result,32)                /*     "     (plan)      */
 $unlopgm = word(result,33)                /*  DSNTIAUL User.(pgm.) */
 $unlopln = word(result,34)                /*     "     (plan)      */
 $dunlopg = word(result,35)                /*  DSNTIAUL IBM  (pgm.) */
 $dunlopl = word(result,36)                /*     "     (plan)      */
 $dsnproc = word(result,37)                /*  Procname DB2         */
 return
```

$DB2TL0C REXX EXEC

```
/* REXX */
trace ?o
    /*-------------------------------------------*/
    /*-     Management Tool for DB2 environment    -*/
    /*-          Log Inventory Report              -*/
    /*-------------------------------------------*/
    /*-          P A R A M E T E R S               -*/
    /*-   - @Db2subs : DB2 Subsystem               -*/
    /*-   - @db2ver: : DB2 version                 -*/
    /*-------------------------------------------*/
arg parmin
parm     = translate(parmin,' ',',')
```

```
nparm    = words(parm)
@db2subs = word(parm,1)
@db2ver  = word(parm,2)
   /*--  Parameters assignment                --*/
 call @db2parØ @db2subs
 if word(result,1) = 99 then
    exit
 $lpar    = word(result,1)            /*  LPAR                */
 $accn    = word(result,2)            /*  Account name        */
 $class   = word(result,3)            /*  Class               */
 $msgcla  = word(result,4)            /*  Message class       */
 $region  = word(result,5)            /*  Region              */
 $msglvl  = word(result,6)            /*  Message level       */
 $notif   = word(result,7)            /*  Notify              */
 $user    = word(result,8)            /*  User                */
/**/
 $unitda  = word(result,9)            /*  Type of unit dasd   */
 $unitta  = word(result,1Ø)           /*  Type of unit tape   */
 $esunit  = word(result,11)           /*  Esoteric work name  */
 $prt     = word(result,12)           /*  Printer name        */
 $hiwork  = word(result,13)           /*  Hi-level work areas */
/**/
 $db2ver  = word(result,14)           /*  DB2 version         */
 $ctsubs  = word(result,15)           /*  Carattere subsystem */
/**/
 $librexx = word(result,16)           /*  REXX      library   */
 $parmlib = word(result,17)           /*  Parmlib      "      */
 $proclib = word(result,18)           /*  Proclib      "      */
 $jcllib  = word(result,19)           /*  JCL          "      */
 $report  = word(result,2Ø)           /*  Report out   "      */
 $libexec = word(result,21)           /*  SysExec      "      */
/**/
 $isptenu = word(result,22)           /*  ISPF      library   */
 $isppenu = word(result,23)           /*   "           "      */
 $ispmenu = word(result,24)           /*   "           "      */
 $ispslib = word(result,25)           /*   "           "      */
/**/
 $plilink = word(result,26)           /*  PLI          "      */
 $sibmlnk = word(result,27)           /*   "           "      */
 $sortlib = word(result,28)           /*  Sort         "      */
/**/
 $hilvlDB = word(result,29)           /*  Hi-level DB2        */
 $runlib  = word(result,3Ø)           /*  DB2 Runlib library  */
 $dsnload = word(result,31)           /*  DB2 system   "      */
/**/
 $tep2pgm = word(result,32)           /*  DSNTEP2  (program)  */
 $tep2pln = word(result,33)           /*     "     (plan)     */
 $unlopgm = word(result,34)           /*  DSNTIAUL User.(pgm.) */
 $unlopln = word(result,35)           /*     "     (plan)     */
 $dunlopg = word(result,36)           /*  DSNTIAUL IBM  (pgm.) */
 $dunlopl = word(result,37)                /*     "     (plan)      */
 $dsnproc = word(result,38)                /*  Procname DB2         */
   /*--   Work areas initialization           --*/
 blk      =
 #b       = Ø
 #c       = Ø
 curØØ    = '@db2logt'
 @db2bsds = @db2subs'ØØ1.BSDSØ1'
 @db2Logt = 'A'
   /*--  Display Log Inventory Report Panel   --*/
 call Free
 do forever
    @db2data = date(e)
    @db2tim  = time()
    wrexit = ok
    address ispexec "display panel(@db2pØ12) cursor("curØØ")"
    @db2msg =
    if rc = 8 then do
       curØØ   = '@db2subs'
       @db2opt =
       @db2msg =
       return  @db2msg
       end
    if wrexit = ok then
       call RunjuØØ4
    end
 exit
   /*--  Call DSNJUØØ4                        --*/
 RunjuØØ4 :
    xx=outtrap(trpdummy.)
       "free fi(sysprint)"
       "free fi(sysin)"
    xx=outtrap(off)
   /*--  SYSPRINT file allocation             --*/
    outds1= $hiwork'.'@db2subs'.@DB2BSDS.SYSPRINT'
    dsn = sysdsn(''''outds1'''')
    if dsn ¬= OK then do
       prmalloc = @db2subs' 'outds1' Ø 6Ø,3Ø v,b,a 125 27998 sysprint
no'
       call @db2allØ prmalloc
       if word(result,1) = 99 then
          exit
       end
    else do
       jobw = sysprint
       "alloc da('"outds1"') f("jobw") shr reuse"
       end
   /*--  SYSIN file allocation                --*/
    "alloc dummy f(SYSIN)"
   /*--  SYSUT1 file allocation               --*/
    BSDSinØ = @db2subs'ØØ1.BSDSØ1'
    xx=outtrap(trpØ.)
       "alloc da('"BSDSinØ"') f(sysut1) shr reuse"
    xx=outtrap(off)
    if rc > Ø then do
       do #a = 1 to trpØ.Ø
          say trpØ.#a
       end
       say ''
       say ''
       say '>>>>>>>>  Error reading file "'BSDSinØ'"    '
       say '>>>>>>>>  RC='rc'. Verify.                  '
       say '>>>>>>>>'
       say ''
       say ''
       call Free
       exit
       end
    xx=outtrap(trpØØ.)
       "ispexec select pgm(DSNJUØØ4) "
       if rc > Ø then do
          say ''
          say '>>>>>>'
          say '>>>>>> Call DSNJUØØ4 RC = 'rc
          say '>>>>>> Verify output. Stop elaboration. '
          say '>>>>>>'
          say ''
          address tso "printoff ('"outds1"') class(X)"
          "free fi(sysin)"
          "free fi(sysprint)"
          exit
          end
    xx=outtrap(off)
   xx=OUTTRAP(trpØ1.)
      "ispexec edit dataset('"outds1"') macro(@mdb2ØØ2)"
   xx=OUTTRAP(OFF)
   /*--  Start process                        --*/
    call FillTab
    return
   /*--  Routine fill up ISPF/TAB             --*/
 FillTab:
   /*--  Read DSNJUØØ4 output                 --*/
    jobw = sysprint
    "alloc da('"outds1"') f("jobw") shr reuse"
    xx=outtrap(trpreadØ1.)
       "execio * diskr sysprint (stem sysprint. finis"
    xx=outtrap(off)
    if rc > Ø then do
       do #a = 1 to trpreadØ1.Ø
          say trpreadØ1.#a
       end
       say ''
       say ''
       say '>>>>>>>>'
       say '>>>>>>>>  Error reading file "'outds1'"    '
       say '>>>>>>>>  RC='rc'. Verify.'
       say '>>>>>>>>'
       say ''
       say ''
       call Free
       exit
       end
   /*--  TBcreate tdislogØ                    --*/
    address ispexec
    "tbcreate tdislogØ names($A $B $C $D $E $F $G $F $H $I $L $M $N)
    nowrite replace"
    DO #d = 1 to sysprint.Ø
       analisi = substr(sysprint.#d,2,29)
       select
          when analisi = 'ACTIVE LOG COPY 1 DATA SETS ' &,
             @db2Logt = 'A' then do
             $B = '1'
             #b = #d + 2
             Do while analisi ¬= 'ARCHIVE LOG COPY 1 DATA SETS'
                #b = #b + 1
/* Logtype */     $A = 'ACTIVE'
/* Frba    */     $E = word(sysprint.#b,1)
/* Trba    */     $H = word(sysprint.#b,2)
/* Dsname  */     $I = strip(substr(sysprint.#b,68,12Ø))
/* Created */     $L = word(sysprint.#b,3)'  'word(sysprint.#b,4)
                #b = #b + 1
/* Fdate   */     $C = word(sysprint.#b,1)
/* Ftime   */     $D = substr(sysprint.#b,13,8)
/* Tdate   */     $F = word(sysprint.#b,3)
/* Ttime   */     $G = substr(sysprint.#b,35,8)
/* Status  */     $M = strip(substr(sysprint.#b,7Ø,12Ø))
                "tbmod tdislogØ"
                #c = #b + 1
                analisi = substr(sysprint.#c,2,29)
                end
             #d = #b
             end
          when analisi = 'ACTIVE LOG COPY 2 DATA SETS '  &,
             @db2Logt = 'A' then do
             #c = #d + 1
             analisi = substr(sysprint.#c,2,29)
             if analisi = 'NO ACTIVE DATA SETS DEFINED F' then do
                @db2msg = 'Dual Active Log not implemented '
                leave
                end
```

```
              $B = '2'
              #b = #d + 2
              Do while analisi ¬= 'ARCHIVE LOG COPY 2 DATA SETS'
                 #b = #b + 1
/* Logtype */    $A = 'ACTIVE'
/* Frba    */    $E = word(sysprint.#b,1)
/* Trba    */    $H = word(sysprint.#b,2)
/* Dsname  */    $I = strip(substr(sysprint.#b,68,12Ø))
/* Created */    $L = word(sysprint.#b,3)'  'word(sysprint.#b,4)
                 #b = #b + 1
/* Fdate   */    $C = word(sysprint.#b,1)
/* Ftime   */    $D = substr(sysprint.#b,13,8)
/* Tdate   */    $F = word(sysprint.#b,3)
/* Ttime   */    $G = substr(sysprint.#b,35,8)
/* Status  */    $M = strip(substr(sysprint.#b,7Ø,12Ø))
                 "tbmod tdislogØ"
                 #c = #b + 1
                 analisi = substr(sysprint.#c,2,29)
                 end
              #d = #b
              end
           when analisi = 'ARCHIVE LOG COPY 1 DATA SETS'  &,
              @db2Logt = 'R' then do
              $B = '1'
              #b = #d + 2
              Do while analisi ¬= 'ACTIVE LOG COPY 2 DATA SETS '
                 #b = #b + 1
/* Logtype */    $A = 'ARCHIVE'
/* Frba    */    $E = word(sysprint.#b,1)
/* Trba    */    $H = word(sysprint.#b,2)
/* Dsname  */    $I = strip(substr(sysprint.#b,68,12Ø))
/* Created */    $L = word(sysprint.#b,3)'  'word(sysprint.#b,4)
                 #b = #b + 1
/* Fdate   */    $C = word(sysprint.#b,1)
/* Ftime   */    $D = substr(sysprint.#b,13,8)
/* Tdate   */    $F = word(sysprint.#b,3)
/* Ttime   */    $G = substr(sysprint.#b,35,8)
/* Vol     */    $M = strip(translate(word(sysprint.#b,6),'','VOL='))
/* Unit    */    $N = substr(word(sysprint.#b,7),6,1Ø)
                 "tbmod tdislogØ"
                 #b = #b + 1
                 #c = #b + 1
                 analisi = substr(sysprint.#c,2,29)
                 end
              #d = #b
              end
           when analisi = 'ARCHIVE LOG COPY 2 DATA SETS'  &,
              @db2Logt = 'R' then do
              #c = #d + 1
              analisi = substr(sysprint.#c,2,29)
```

```
             if analisi = 'NO ARCHIVE DATA SETS DEFINED ' then do
                @db2msg = 'Dual Archive Logging not implemented '
                leave
                end
             $B = '2'
             #b = #d + 2
             Do while analisi ¬= '                 CONDITIONAL' &,
                      analisi ¬= 'DSNJ4Ø1I DSNRJPCR RESTART CO'
                #b = #b + 1
/* Logtype */   $A = 'ARCHIVE'
/* Frba    */   $E = word(sysprint.#b,1)
/* Trba    */   $H = word(sysprint.#b,2)
/* Dsname  */   $I = strip(substr(sysprint.#b,68,12Ø))
/* Created */   $L = word(sysprint.#b,3)'  'word(sysprint.#b,4)
                #b = #b + 1
/* Fdate   */   $C = word(sysprint.#b,1)
/* Ftime   */   $D = substr(sysprint.#b,13,8)
/* Tdate   */   $F = word(sysprint.#b,3)
/* Ttime   */   $G = substr(sysprint.#b,35,8)
/* Vol     */   $M = strip(translate(word(sysprint.#b,6),'','VOL='))
/* Unit    */   $N = substr(word(sysprint.#b,7),6,1Ø)
                "tbmod tdislogØ"
                #b = #b + 1
                #c = #b + 1
                analisi = substr(sysprint.#c,2,29)
                end
             #d = #b
             leave
             end
          otherwise
             nop
          end
       end  /* END DO #d = 1 to sysprint.Ø */
    "tbsort tdislogØ fields($I,c,a)"
    Do forever
       "tbtop  tdislogØ"
       if @db2Logt = 'A' then
          "tbdispl tdislogØ panel(@db2pØ13) cursor("curØØ")"
       else
          "tbdispl tdislogØ panel(@db2pØ14) cursor("curØØ")"
       if rc = 8 then do
          call Free
          curØØ   = '@db2Logt'
          @db2opt =
          @db2msg =
          return  @db2msg
          end
       end
    return
   /*--  Free work dataset                   --*/
```

```
 Free  :
    xx=outtrap(trpdummy.)
       address tso
       "delete '"outds1"'"
       "free fi(sysin)"
       "free fi(sysprint)"
       "free fi(sysut1)"
    xx=outtrap(off)
    return
```

$DB2PAR0 REXX EXEC

The code is contained in ‘Imagecopy generator procedure’, *DB2 Update*, issue 106, August 2001.

$DB2ALL0 REXX EXEC

The code is contained in ‘Imagecopy generator procedure’, *DB2 Update*, issue 106, August 2001.

$DB2P000 PANEL

```
)ATTR FORMAT(MIX)
 [  type(text)    intens(&acc) color(white)
 ‡  type(text)    intens(high) color(yellow)
 %  type(text)    intens(high) color(white)  skip(on)
 +  type(text)    intens(low)  color(green)  skip(on)
 *  type(output)  intens(high) color(yellow) caps(off)
 #  type(output)  intens(low)  color(green)  skip(on)
 $  type(input)   intens(low)  color(red)    pad(_)
 @  type(input)   intens(low)  color(red)
)BODY expand(\\)
+ &ZUSER   + \-\- %DB2 Tools Main Menu +-\-\  #Z       +#Z       +
%COMMAND ===>$ZCMD
+
+*Z                                                                  +
    %DB2 Subsystem   ==>$Z   +             [DB2 Version         ==>#Z +

 +\-\   +
+
+
+
+
+
+
+                              ‡C+- Log Inventory Report
```

```
+
+
+              %Select option+$Z+
+
+
+
+
+
+
+                                                ---- %PF1+Help %PF3+End
----
)INIT
 .HELP  = @DB2HØØØ
 .ZVARS = '@db2data,@db2tim,@db2msg,@db2subs,@db2ver,@db2opt'

)PROC
 &@db2data = '&zday/&zmonth/&zyear &ztime'
 ver(&@db2subs,nonblank)
 ver(&@db2opt,list,C)
)END
```

$DB2P012 PANEL

```
)ATTR FORMAT(MIX)
 %  type(text)    intens(high) color(&txhigh) skip(on)
 +  type(text)    intens(low)  color(green)   skip(on)
 ^  type(text)    intens(low)  color(white)   skip(on)
 ?  type(text)    intens(high) color(red)     skip(on)
 <  type(output)  intens(high) color(yellow)  caps(off)
 #  type(output)  intens(low)  color(green)   skip(on)
 $  type(input)   intens(low)  color(red)     pad(_)
)BODY expand(\\)
+ &ZUSER + \-\- %DB2 Log Inventory Report +-\-\  #Z      +#Z      +
%COMMAND ===>$ZCMD
+
+<Z                                                                +
    %DB2 Subsystem   ==>#Z   +            %DB2 Version        ==>#Z +

  +---------------------------------------------------------------------
    %BSDS Name ==>#Z               +
    %Log Type  ==>$Z+   (%A+- Active,%R+- Archive)

  +\-\- %Variables Description +-\-\

   +%BSDS Name         +- Name of Bootstrap data-set

   +%Log Type          +-%Active+Log data-set or%Archived+Log data-set
```

```
+                                           ---- %PF1+Help %PF3+End ----

)INIT
 .HELP  =  @db2hØØC
 .ZVARS = '@db2data,@db2tim,@db2msg,@db2subs,@db2ver, +
          @db2BSDS,@db2logt'
)PROC
 &@DB2DATA = '&ZDAY/&ZMONTH/&ZYEAR &ZTIME'
 ver(&@db2logt,nonblank)
 ver(&@db2logt,list,A,R)
)END
```

$DB2P013 PANEL

```
)ATTR FORMAT(MIX)
 %  type(text)    intens(high) color(&txhigh) skip(on)
 +  type(text)    intens(low)  color(green)   skip(on)
 ^  type(text)    intens(low)  color(yellow)  skip(on)
 ?  type(text)    intens(high) color(red)     skip(on)
 <  type(output)  intens(high) color(yellow)  caps(off)
 #  type(output)  intens(low)  color(green)   skip(on)
 $  type(input)   intens(low)  color(red)     pad(_)
)BODY expand(\\)
+ &ZUSER + \-\- %DB2 Log Inventory Report +-\-\  #Z       +#Z       +
%COMMAND ===>$ZCMD
+
+<Z                                                                  +
    %DB2 Subsystem   ==>#Z   +             %DB2 Version         ==>#Z +
  +---------------------------------------------------------------------
    %BSDS Name ==>#Z                +
    %Log Type  ==>#Z+   (%A+- Active,%R+- Archive)
  +---------------------------------------------------------------------
+                                           ---- %PF1+Help %PF3+End ----
%   Log    Copy          DATE        Time       Log RBA
+  ---------------------------------------------------------------------
)MODEL
  #Z       #Z  ^From: #Z          #Z         #Z
               ^To  : #Z          #Z         #Z
  ^DSN     :#Z
  ^Created :#Z                      ^Status  :#Z

)INIT
 .HELP  =  @db2hØØC
 .ZVARS = '@db2data,@db2tim,@db2msg,@db2subs,@db2ver, +
          @db2BSDS,@db2logt,                         +
          $A,$B,$C,$D,$E,$F,$G,$H,$I,$L,$M'
)PROC
 &@DB2DATA = '&ZDAY/&ZMONTH/&ZYEAR &ZTIME'
)END
```

$DB2P014 PANEL

```
)ATTR FORMAT(MIX)
 %  type(text)    intens(high) color(&txhigh) skip(on)
 +  type(text)    intens(low)  color(green)   skip(on)
 ^  type(text)    intens(low)  color(yellow)  skip(on)
 ?  type(text)    intens(high) color(red)     skip(on)
 <  type(output)  intens(high) color(yellow)  caps(off)
 #  type(output)  intens(low)  color(green)   skip(on)
 $  type(input)   intens(low)  color(red)     pad(_)
)BODY expand(\\)
+ &ZUSER + \-\- %DB2 Log Inventory Report +-\-\  #Z       +#Z       +
%COMMAND ===>$ZCMD
+
+<Z                                                                  +
    %DB2 Subsystem   ==>#Z   +             %DB2 Version        ==>#Z +
  +-------------------------------------------------------------------
    %BSDS Name ==>#Z               +
    %Log Type  ==>#Z+   (%A+- Active,%R+- Archive)
  +-------------------------------------------------------------------
+                                                    ---- %PF1+Help
%PF3+End ----
%   Log    Copy          DATE        Time       Log RBA
+  -------------------------------------------------------------------
)MODEL
  #Z       #Z  ^From: #Z          #Z         #Z
               ^To  : #Z          #Z         #Z
  ^DSN     :#Z
  ^Created :#Z                   ^Vol :#Z     ^Unit :#Z

)INIT
 .HELP  =  @db2hØØC
 .ZVARS = '@db2data,@db2tim,@db2msg,@db2subs,@db2ver, +
          @db2BSDS,@db2logt,                         +
          $A,$B,$C,$D,$E,$F,$G,$H,$I,$L,$M,$N'
)PROC
 &@DB2DATA = '&ZDAY/&ZMONTH/&ZYEAR &ZTIME'
)END
```

$DB2H000 HELP PANEL

```
)ATTR
[  type(text) intens(low) color(green)
```

```
?  type(text) intens(low) color(red)
‡  type(text)   intens(high) color(yellow)
%  type(text) intens(low) color(turq)
{  type(text)
$  type(text) intens(low)
}  type(et)
'  area(scrl)
)BODY DEFAULT(<]*)EXPAND(||)
[-|-|{ DB2 Tools Help [|-|-$
'OPNL                                                                  '
'                                                                      '
'                                                                      '
'                                                                      '
'                                                                      '
'                                                                      '
'                                                                      '
'                                                                      '
'                                                                      '
'                                                                      '
'                                                                      '
'                                                                      '
'                                                                      '
'                                                                      '
'                                                                      '
'                                                                      '
'                                                                      '
'                                                                      '
'                                                                      '
'                                                                      '
'                                                                      '
'                                                                      '
'                                                                      '
)AREA OPNL
$
$
}1.${Introduction$
$
$   [The{"DB2 Tools"[procedure has the ability to automatically perform
$   [a series of activities that are frequently carried out
$   [manually by DB2 customers; thus saving lots of time.
$
$   [The function below performs a query on the bootstrap dataset in
$   [order to obtain information about DB2 Active and Archived DB2 log.
$
}2.${Environment parameter descriptions$
$
$   [Before first using the functions of the Tool, it will be
$   [necessary to customize some Environment parameters:
$
$   %DB2 Subsystem   ==>?----         %DB2 Version         ==>?--
```

```
$
$ [-{DB2 Subsystem  [Is the DB2 subsystem name on which you work
$
$ [-{DB2 Version    [Is the DB2 version value selected in an automatic
$                   [way depending on the chosen subsystem.
$
$
$
}3.${Tool Functions$
$
$                         -‡System Managment Tools[-
$
$
$                       ‡C[- Log Inventory Report
$
$
$
$
$
$
$                                           {ENTER[to continue{PF3[End[
$
)PROC
&zcont = @DB2HØØØ
)END
```

$DB2H00C HELP PANEL

```
)ATTR
[  type(text) intens(low) color(green)
?  type(text) intens(low) color(red)
‡  type(text)   intens(high) color(yellow)
%  type(text) intens(low) color(turq)
{  type(text)
$  type(text) intens(low)
}  type(et)
'  area(scrl)
)BODY DEFAULT(<]*)EXPAND(||)
[-|-|{ DB2 Tools Help [|-|-$
'OPNL                                                                       '
'                                                                           '
'                                                                           '
'                                                                           '
'                                                                           '
'                                                                           '
'                                                                           '
'                                                                           '
'                                                                           '
'                                                                           '
'                                                                       '
'                                                                       '
'                                                                       '
'                                                                       '
'                                                                       '
'                                                                       '
'                                                                       '
'                                                                       '
'                                                                       '
'                                                                       '
'                                                                       '
'                                                                       '
'                                                                       '
)AREA OPNL
$
}3.A${Log Inventory Report$
$
$    [The function{"Log Inventory Report"[ allow the user to retrive
$    [information about Active and Archive Log.
$
$    [The following field is required :$
$
$    %Log Type  ==>?-[
$
$  [-{Log Type$ [Is the type of log to be selected
$                    {A$for all Active Log
$                    {R$for all Active Log
$
$    [For choice{A[the following report will be produced :
$
$ [ZZDBØØØ  ---------- {DB2 Log Inventory Report[ -------- Ø4/Ø2/Ø1
16:27:Ø3
$ ‡Dual Active Log not implemented
$    {DB2 Subsystem   ==>[DSNS             {DB2 Version        ==>[51
$  [-------------------------------------------------------------------
------
$    {BSDS Name ==>[DSNSØØ1.BSDSØ1
$    {Log Type  ==>[A    ({A[- Active,{R[- Archive)
$  [-------------------------------------------------------------------
------
$  [                                             ----  PF1 Help  PF3
End ----
$   {Log    Copy          DATE        Time       Log RBA
$  [-------------------------------------------------------------------
------
$  [ACTIVE   1  ‡From:[ 2ØØ1.286    23:2Ø:44   Ø1C62E25ØØØØ
$  [            ‡To  :[ 2ØØ1.288    2Ø:3Ø:52   Ø1C649D47FFF
$  ‡DSN     :[DSNSØØ1.LOGCOPY1.DSØ1
$  ‡Created :[1994.129  19:Ø8       ‡Status  :[REUSABLE
$
```

```
$  [ACTIVE   1  ‡From:[ 2ØØ1.288    2Ø:3Ø:52   Ø1C649D48ØØØ
$               ‡To  :[ 2ØØ1.29Ø    21:34:38   Ø1C66583FFFF
$  ‡DSN     :[DSNSØØ1.LOGCOPY1.DSØ2
$  ‡Created :[1994.129  19:Ø8        ‡Status  :[REUSABLE
$
$  [ACTIVE   1  ‡From:[ 2ØØ1.29Ø    21:34:38   Ø1C66584ØØØØ
$               ‡To  :[ ........    ........   Ø1C681337FFF
$  ‡DSN     :[DSNSØØ1.LOGCOPY1.DSØ3
$  ‡Created :[1994.352  13:18        ‡Status  :[NOTREUSABLE
$
$    [For choice{R[the follwing report will be produced :
$
$ [ZZDBØØØ  ---------- {DB2 Log Inventory Report[ -------- Ø4/Ø2/Ø1
16:27:Ø3
$ ‡Dual Archive Log not implemented
$    {DB2 Subsystem   ==>[DSNS              {DB2 Version         ==>[51
$  [-----------------------------------------------------------------
------
$    {BSDS Name ==>[DSNSØØ1.BSDSØ1
$    {Log Type  ==>[R    ({A[- Active,{R[- Archive)
$  [-----------------------------------------------------------------
------
$  [                                             ----  PF1 Help  PF3
End ----
$  { Log    Copy          DATE        Time       Log RBA
$  [-----------------------------------------------------------------
------
$  [ARCHIVE  1  ‡From:[ 2ØØ1.261    2Ø:59:48   Ø1C4E1EBØØØØ
$               ‡To  :[ 2ØØ1.264    2Ø:34:55   Ø1C4FD9A7FFF
$  ‡DSN     :[DSNSØØ1.ARCHLOG1.AØØØ9711
$  ‡Created :[2ØØ1.264  22:38     ‡Vol :[Ø71447‡Unit :[ROBOT
$
$  [ARCHIVE  1  ‡From:[ 2ØØ1.264    2Ø:34:55   Ø1C4FD9A8ØØØ
$               ‡To  :[ 2ØØ1.266    Ø3:14:15   Ø1C51949FFFF
$  ‡DSN     :[DSNSØØ1.ARCHLOG1.AØØØ9712
$  ‡Created :[2ØØ1.266  5:16      ‡Vol :[Ø71299‡Unit :[ROBOT
$
$  [ARCHIVE  1  ‡From:[ 2ØØ1.266    Ø3:14:15   Ø1C5194AØØØØ
$               ‡To  :[ 2ØØ1.268    2Ø:56:18   Ø1C534F97FFF
$  ‡DSN     :[DSNSØØ1.ARCHLOG1.AØØØ9713
$  ‡Created :[2ØØ1.268  22:58     ‡Vol :[Ø69465‡Unit :[ROBOT
$
$  [ARCHIVE  1  ‡From:[ 2ØØ1.268    2Ø:56:18   Ø1C534F98ØØØ
$               ‡To  :[ 2ØØ1.271    22:12:Ø6   Ø1C55ØA8FFFF
$  ‡DSN     :[DSNSØØ1.ARCHLOG1.AØØØ9714
$  ‡Created :[2ØØ1.272  Ø:15      ‡Vol :[Ø71419‡Unit :[ROBOT
$
$
$                                            {ENTER[to continue{PF3[End[
$
)INIT
)PROC
&zcont = @db2hØØC
)END
```

$MDB2002 MACRO

```
/* REXX */
trace ?o
/*-----------------------------------------------*/
/*----    Used in Rexx #DB2TLØC            ----*/
/*-----------------------------------------------*/
isredit macro
isredit change P'''=''' ''' ''' 1 all
isredit save
isredit end
```

Giuseppe Rendano
DB2 Systems Programmer (Italy)

Issuing a SELECT from a Windows BAT file

If you want to run a SELECT with a LIKE clause from a Windows BAT file, you need to use two per cent signs (as shown below). Put the lines below in a file called SM.BAT and then, from the CLP, type SM.

```
@REM Get data from the EXPLAIN table in sample db
db2 connect to sample
db2 select total_cost, substr(statement_text,1,9Ø)  from
explain_statement where statement_text like 'SELECT%%'
```

C Leonard
Freelance Consultant (UK)

The importance of PCTFREE and FREEPAGE

DB2 has recognized and made provision for database growth using two parameters that define the tablespace in which a table resides. They are FREEPAGE and PCTFREE. The former indicates the number of pages after which a free page must be left and the latter defines what percentage of each page must be left free to accommodate this growth. Typically, every time a tablespace is reorganized, these parameters will be applied and the pages will be filled accordingly.

The values of these two parameters are critical for several reasons:

- More free space would mean that DB2 would have to retrieve more pages to get the data. It would also mean longer times to run utility jobs like image copies, REORGs, RUNSTATS, etc.
- Less free space would mean that the data may not be accessed in a sequential manner because of a large number of relocated rows, and hence data access may be more resource-intensive. Also INSERT operations will become more time-consuming and performance will degrade gradually.

An optimal value for these parameters should therefore provide enough room for growth without affecting performance in any way. This can be achieved to some extent through the use of mathematical modelling and iterative analysis using utilities.

THE MATHEMATICAL MODEL

The mathematical model for estimating the optimal free space parameters will be discussed briefly and the utilities used for this purpose will be discussed. The mathematical model is actually developed from previously published material. Unfortunately I do not remember the title or the author because it is from notes that go back some time. Hence acknowledgements are due as applicable to the original author.

The estimated value for PCTFREE can be calculated as follows:

```
PCTFREE =   g / (1+g),
```

where *g* is the net growth rate of the tablespace (on a weekly or monthly basis. Ideally it should coincide with the frequency of REORGs).

If the growth rate is 50%, setting PCTFREE to 50 would mean that half the page is left free, which means that the page can really accommodate 100% growth. If we have 100 rows on a page and we expect it to accommodate another 50 rows (50% growth rate), then we would really need room for 150 rows. Hence, if the growth rate is 50%, we could say that PCTFREE = 0.5/1.5 or 33%.

Even though the above formula appears simple we would need to modify it to consider several factors. The number of rows that a page can hold depends on the average row length and the page size. With a 4KB page, after providing for overheads, we are left with 4074 bytes. The row length can be derived from the RECLENGTH column from the SYSTABLES table. (However, we do need to bear in mind that, for compressed tables, the row length will be significantly smaller and that would depend on the compression ratio, the number of rows, etc.)

The 4074 bytes available in a page may be evenly divisible into the row length without any wasted space. If not, the wasted space is calculated as:

```
Wasted space =   4Ø74 - ( max-rows  *  row-length )
```

where:

```
max-rows = MIN(INT(4074/row-length),255)
```

since each page can hold a maximum of 255 rows only.

The usable space may be calculated as:

```
Usable space =  max-rows * row-length,
```

Hence the PCTFREE formula would have to reserve g/(1+g)%

of the usable space and the wasted space and can be re-written as:

```
     PCTFREE = (g/(1+g)  * usable space  +  wasted space ) / 4Ø74
```

Sometimes this formula will result in PCTFREE values that are inadequate for even a single row. Hence the percentage of usable space that would be left as free must be increased to the row length to accommodate at least a full row. That is achieved by introducing a factor called row-ratio (RR). It is the ratio of the size of a row to the usable portion of a page and is expressed as:

```
     RR = row-length / (max-rows * row-length) = 1 / max-rows
```

Now the final formula for PCTFREE would be:

```
PCTFREE = (RR * CEILING(g/(1+g)/RR) * usable space + wasted space)/4Ø74
```

Note: *CEILING* is a function that rounds off a decimal number to the next higher integer.

AN EXAMPLE CALCULATING PCTFREE

Let us consider an example to see how this formula works.

Given that a table has a row length of 350 and an expected growth of 10%:

```
     Maximum rows per page is = MIN ( INT  ( 4Ø74 / 35Ø ), 255 ) = 11
     The row ratio (RR) = 1 / max rows = 1 / 11 = Ø.Ø9Ø9
     Usable space = max.rows * row-length =  11 * 35Ø = 385Ø
     Wasted space =  4Ø74 - usable space = 4Ø74 - 385Ø = 224

     PCTFREE = ( RR * CEILING( g/ (1+g) / RR ) * usable space
                                                 + wasted space) / 4Ø74
     = ( Ø .Ø9Ø9 * CEILING( Ø.1 / 1.1 / Ø.Ø9Ø9 ) * 385Ø + 224 ) / 4Ø74
     = Ø.226789 or 23%
```

CALCULATION OF TOTAL SPACE

In most cases, the PCTFREE value itself would help us to achieve optimum space usage. However, sometimes the value calculated as above would be meaningless, especially for tables with large row-lengths. Hence we try to achieve the best value by an iterative approach.

Using the growth percentage and the number of rows, it is possible to calculate the required room to accommodate the expected growth. It is given as the cardinality or number of rows times the growth. If the cardinality is 5,000,000 and the growth is 2%, then we must provide room for 100,000 rows.

Given a PCTFREE and a FREEPAGE value, we can compute the total rows that can be accommodated as follows:

```
available space =  INT(4Ø74*(1ØØ-PCTFREE)/1ØØ)
rows-per-page = MIN(INT(available space/row-length),255)
unused space = 4Ø74 - (rows-per-page * row-length)
```

Assuming a uniform distribution of growth, the number of additional rows that each page can hold based on the PCTFREE value can be given by:

```
room-for-rows = INT(unused space / row-length)
```

The number of pages required to hold the rows is:

```
no-pgs-used =  INT(cardinality / rows per page) + 2
```

The total number of pages with FREEPAGES can be approximated to:

```
total-pgs-used = INT(no-pgs-used*(FREEPAGE+1)/(MAX(FREEPAGE,1)))

The total-freepages   = total-pgs-used  -  no-pgs-used
```

If we had decided not to provide any free space on any page, then we would have provided total room for growth as:

```
total-room-for-rows =  total-freepages * rows-per-page
```

where:

```
total-freepages = total-pgs-used - no-pgs-used
```

If there is room to hold new rows on each page, then:

```
total-room-for-rows = (no-pgs-used * room-for-rows)
                               +  (total-freepages *  rows-per-page)
The page difference = total-pgs-used  -  no-pgs-used
```

The page difference specifies the amount of free space that is available to accommodate future growth.

A PROGRAMMING SOLUTION BASED ON THIS APPROACH

In the discussion that follows, the program variable will be indicated in lower case in parentheses and the output headings will be all capitals in parentheses, except where indicated otherwise.

The procedure CALCREQMT in the program SPCCAL performs the above calculations. It receives new values of PCTFREE (PCF) and FREEPAGE (FREPG) for each invocation and calculates the total-room-for-rows (ntotrm), total-pgs-used (ntotpg), and the page difference (npgdif) that is above what is required.

The required room for growth is calculated as the product of growth and cardinality and is represented as REQRM in the report. Using the existing set of PCTFREE and FREEPAGE values, we invoke CALCREQMT and compute how many additional rows can be held (OTOTRM) and the pages required to hold that many additional rows (OTOTPG). The number of pages that are required to hold the difference between OTOTRM and REQRM is calculated and represented as OPGDIF in the output.

The new PCTFREE that is computed using the formula described above is then passed to CALCREQMT along with the existing FREEPAGE value. The values computed by CALCREQMT are stored in temporary variables and they are checked as follows:

There are three possible scenarios when we start comparing the old and new values:

1 If the room-for-rows is equal to the REQRM (actual required room based on growth), then we accept these as the values for FREEPAGE and PCTFREE.

2 If the new room-for-rows is greater than the required room-for-rows, the FREEPAGE is decreased by 1 and the process will call CALCREQMT. This process iterates until the new room-for-rows just exceeds the required room-for-rows.

3 If the new room-for-rows is less than the required room-for-rows, the FREEPAGE is increased by 1 and the process will call CALCREQMT iteratively as in *Step 2* above until the new value falls just under the required value.

It is possible that we might end up with more free space than we had when we started. Hence we do the second set of iterations that keep the FREEPAGE at 0 and decrement the PCTFREE by 1 until the free space is more reasonable.

We indicate threshold values for FREEPAGE and PCTFREE when we use these iterative processes, otherwise they may become infinitely large.

A SIMULATOR

Despite all these efforts, sometimes it will be preferable to have certain specific values for FREEPAGE and PCTFREE based on experience. Or, you may want to tweak it further and see the end result. For this purpose, the second program, RCSPCCAL, is handy. We alter the PCTFRE and FREEPAGE values in the output generated by the SPCCAL program or by the RCSPCCAL program and run this against it and review the results. A high negative value in the PGDIF column of the output is undesirable and calls for revision of FREEPAGE and PCTFREE parameters.

INPUT TO THE PROGRAM

The line of code reproduced below would define the input structure. The input data is given as a dataset or file containing several rows. Within each row, the data must be separated by spaces. Each line can represent only one tablespace object and there is no limit to the number of tablespaces that you can have in the input file.

```
parse var inrec dbn tsn prt grn rl crl fp pct card  dumy
```

The key data that is required for the process is:

- inrec – the entire input record that is read in.

- dbn – database name.
- tsn – tablespace name.
- prt – partition number.
- grn – net growth rate in percentage rounded to the next highest integer.
- rl – row length.
- crl – compressed row length (computed as: row-length * (page saved by compression minus 5).
- fp – free page at present.
- pct – percentage free at present.
- card – number of rows.
- dumy – whatever follows after the last significant input.

A sample input row would look like this:

```
dbn       tsn        prt   grn   rl     crl   fp    pct      card
----------------------------------------------------------------------
DBADBØØ1  DBATSØØ1   ØØ1   4     226    4Ø    1Ø    3Ø       35765483
```

A description of some of the fields in the sample output are:

- NGRTH – net growth.
- NRECLEN – revised record length based on page-save.
- PF and NPF – old and new PCTFREE values.
- FP and NFP – old and new FREEPAGE values.
- CARD – cardinality.
- REQRM – expected number of new rows in table based on growth.
- OLDRM – additional rows that can be accommodated based on old values of PCTFREE and FREEPAGE.
- NEWRM – additional rows that can be accommodated

based on new (calculated) values of PCTFREE and FREEPAGE.

- OPGDIF – the additional pages free beyond the expected growth based on old values.
- NPGDIF – the additional pages free beyond the expected growth based on new values (0 is the desirable value here).
- OTOTPG – total pages as per old values (PF and FP)
- NTOTPG – total pages as per new values (NPF and NFP)
- PGDIF – OTOTPG minus NTOTPG. A negative value here indicates that we are over-allocating FREEPAGE and may be even PCTFREE. Revise the NPF and NFP values and run RCSPCCAL again.

INPUT TO RCSPCCAL

The output from SPCCAL or from RCSPCCAL will be the input for RCSPCCAL. However, each execution of this generates an output dataset with a timestamp so that it can be distinct from any previous runs.

ASSUMPTIONS

The assumption for this entire methodology is that the rows being added will follow a uniform distribution. In some cases, there will be more full pages as the database grows and hence more relocated rows. If the application's data access is predominantly random, then it should not matter that several rows are relocated. Another assumption is that the growth rate is computed based on the frequency of REORGs. It is also desirable to keep the partitioned tablespaces well balanced so as to balance the insert activity.

CONCLUSION

There are situations where this process will not yield very good results. The PGDIF column in the output will help identify those cases. We can substitute our values for PCTFREE and FREEPAGE and repeat the calculations using the simulator, RCSPCCAL, until we get a satisfactory value. Hence the output needs to be studied with caution before implementing the suggested values. This model will not work well for some tables like application trigger tables because their record counts keep increasing and decreasing all the time. For such tables, we fix the values based on experience and knowledge of the application.

This is not a solution tool, but rather an analysis technique. When used with understanding and close monitoring, this process should generate desirable benefits and help to set optimum free space values.

SPCCAL

```
/*********************************************************************/
/*     rexx                                                          */
/*  SPCCAL                                                           */
/* This is to calculate the optimum pctfree values based on         */
/* growth. Use this with care because it is extremely sensitive      */
/*********************************************************************/
trace o
numeric digits 15
cnt=Ø
clear
pref =strip(sysvar(syspref))
PARSE UPPER ARG P_dsname
if strip(P_dsname)='' then
do
   Call GETDBLST
end
else
do
   I_lstdsn = strip(P_dsname)
   I_lstdsn = strip(P_dsname,B,"'")
end
Call ALLOCDSN
Call ALLOCOUT
MAINØØØ:
hdr.1=,
'DBNAME    TSNAME   PRT  GRTH  RECLEN NRECLN   PF   FP',
'      CARD  NPF  NFP  REQRM   OLDRM  NEWRM   OPGDIF   NPGDIF',
'    TSNAME   OTOTPG   NTOTPG   PGDIF'
hdr.2=,
'-------------------------------------------------------------------',
'-----------------------------------------------------------------------
--',
'----------'
 "execio * diskw opds (stem hdr. "
do forever
  trace o
  "execio 1 diskr lstdd"
  if rc=2 then leave
  pull inrec
  if pos('DBNAME',inrec) > Ø | pos('------',inrec) > Ø then
     iterate
  parse var inrec dbn tsn prt grn rl crl fp pct card  dumy
  cnt = cnt+1
  if cnt//35 = Ø then
  do
     say 'Processing ' tsn prt
    "execio * diskw opds (stem hdr. "
  end
  crl = trunc(crl+Ø.9)
  if crl > rl then crl=rl
  ngrn = trunc(grn+Ø.9)
  maxrows = MIN(TRUNC(4Ø74/crl),255)
  grn = ngrn/1ØØ
  RR = 1/maxrows
  usblspc = maxrows * crl
  wstdspc = 4Ø74 - usblspc
  fac1   =  TRUNC( ( (grn/(1+grn)) / RR ) + Ø.9)
  fac2   =  RR * fac1 * usblspc
  pctfree=  ( fac2 + wstdspc ) / 4Ø74
  pctfree = TRUNC((pctfree * 1ØØ) + Ø.5)
  nreqrm = trunc(card*grn)      /* 1 more than required */
  /* calc of space based on old pctfree */
  PCF = pct
  FREPG = fp
  Call CALCREQMT
  ototrm = ntotrm
  opgdif = npgdif
  ototpg = ntotpg
/** check for low card and pages used < 32 ***/
  xavlspc = 4Ø74
  xrpg = MIN(TRUNC(xavlspc/crl),255)
  if xrpg <= Ø then xrpg = 1
  xpgusd = 2+trunc(card/xrpg)
  if xpgusd <= 32 then
  do
     FREPG = Ø
     pctfree = Ø
     Call WRITEOUT
     iterate
  end
```

```
/**  end of check for low card and low pages used **/
/* calc of space based on new pctfree */
  FREPG = fp   /* we start off with existing free pages */
  PCF = pctfree
  Call CALCREQMT
  rtotrm = ntotrm
  rpgdif = npgdif
  rtotpg = ntotpg
  tpgdif = ototpg-rtotpg
  if rtotrm = nreqrm | nreqrm=Ø then
  do
     rpgdif = npgdif
     Call WRITEOUT
     iterate
  end
  if rtotrm > nreqrm then
  do
     do while (rtotrm > nreqrm)   & FREPG >= Ø
/*      say 'Calc. dnward freepg for 'tsn prt FREPG PCF         */
        FREPG = FREPG - 1
        Call CALCREQMT
        rtotrm = ntotrm
     end
     FREPG = FREPG + 1
     Call CALCREQMT
  end
  else  /* rtotrm < nreqrm  */
  do
     do while(rtotrm < nreqrm) & FREPG <= 64
/*      say 'Calc. upward freepg for 'tsn prt FREPG PCF         */
        FREPG = FREPG + 1
        Call CALCREQMT
        rtotrm = ntotrm
     end
     FREPG = FREPG - 1
     Call CALCREQMT
     rtotrm = ntotrm
  end
  rtotrm = ntotrm
  rpgdif = npgdif
  rtotpg = ntotpg
  tpgdif = ototpg-rtotpg
/*  based on the above iteration, we expect the tpgdif to be
|   greater than or equal to Ø. If that were not the case,
|   and if pctfree is less than 25, then we discard the FREEPAGE
|   value, set it to Ø, and calculate a new PCTFREE that gives
|   tpgdif >= Ø.
*/
  do while tpgdif < Ø & pctfree > Ø
     FREPG = Ø
     pctfree = pctfree-1
     PCF = pctfree
     Call CALCREQMT
     rpgdif = npgdif
     rtotpg = ntotpg
     tpgdif = ototpg-rtotpg
     rtotrm = ntotrm
  end
/*  if the calculated total room is less than the reqd. room
|   then we fix FREEPG to Ø and increase PCTFREE till we get the
|   results we need or PCTFREE reaches 25
|
*/
  do while rtotrm < nreqrm  & pctfree < 26
     FREPG = Ø
     pctfree = pctfree+1
     PCF = pctfree
     Call CALCREQMT
     rpgdif = npgdif
     rtotpg = ntotpg
     tpgdif = ototpg-rtotpg
     rtotrm = ntotrm
  end
  Call WRITEOUT
end
Say 'Press Enter to continue '
"execio Ø diskr lstdd (FINIS "
"execio * diskw opds (FINIS "
address tso "free f(lstdd)"
address tso "free f(opds)"
say 'See output in 'ods_name
exit
GETDBLST:
Say  'Give the input dataset ...'
Say  '(It must be a PS )'
pull I_lstdsn
/* I_lstdsn = HRXS.$$PAJKJ.OUTPUT.DMPPDBP5.D99Ø2Ø8 */
I_lstdsn = strip(I_lstdsn)
I_lstdsn = strip(I_lstdsn,Both,"'")
x = SYSDSN("'"I_lstdsn"'")
if x ¬= OK then
do
   say; say '*** ERROR ' x ; say
   SIGNAL  GETDBLST
end
return
ALLOCDSN:
"ALLOCATE DD(lstdd) DSN('"I_lstdsn"') REUSE SHR"
if  rc>Ø then
do
```

```
      say  'Failed during allocation of 'I_lstdsn
      exit(8)
   end
   return
   ALLOCOUT:
   cts= time(S)
   cd = date(U)
   us_date = substr(cd,7,2)||substr(cd,1,2)||substr(cd,4,2)
   odsn = pref||'.'||userid()||'.SPCCAL.D'||us_date
   ods_name = odsn
   xx = outtrap("zap.","*")
   address tso "delete '"ods_name"'"
   xx = outtrap("OFF")
   address tso "alloc f(opds) new unit(hsm) space(2,2)",
             "tracks reuse dsname('"ods_name"')",
             "dsorg(ps) blksize(2000) lrecl(200) recfm(f b)"
   return
   WRITEOUT:
     grn =  trunc((grn*100),2)
     out.1 = left(dbn,10)||left(tsn,10)||right(prt,2)
     out.1 = out.1' 'right(ngrn,5)
     out.1 = out.1' 'right(rl,6)' 'right(crl,6)
     out.1 = out.1' 'right(pct,4)' 'right(fp,4)' 'right(card,10)
     out.1 = out.1||right(pctfree,5)||right(FREPG,5)
     out.1 = out.1' 'right(nreqrm,6)'  'right(ototrm,6)
     out.1 = out.1' 'right(rtotrm,6)
     out.1 = out.1'  'right(opgdif,7)'  'right(rpgdif,7)' 'right(tsn,10)
     out.1 = out.1'  'right(ototpg,7)'  'right(rtotpg,7)' 'right(tpgdif,7)
     "execio * diskw opds (stem out. "
     drop out.
   return
   CALCREQMT:
    navlspc = trunc(4074*(100-PCF)/100)
    nrpg = MIN(TRUNC(navlspc/crl),255)
    if nrpg <= 0 then nrpg = 1
    nunusd = 4074-(crl*nrpg)
    if nunusd = 0 then
       nroomfor = 0
    else
       nroomfor = trunc(nunusd/crl)
    npgusd = 2+trunc(card/nrpg)
    ntotpg = trunc(npgusd*(FREPG+1)/(MAX(FREPG,1)))
    totfpg = ntotpg - npgusd
    nKB = ntotpg*4
    if nroomfor = 0 then
    do
       ntotrm = totfpg*nrpg
    end
    else
       ntotrm = (npgusd*nroomfor) + (totfpg*nrpg)
```

```
 npgdif = ntotpg - npgusd
 return
```

RCSPCCAL

```
/*********************************************************************/
/*     rexx                                                          */
/* RCSPCCAL                                                          */
/* This is to recalculate the optimum free space values based on     */
/* manual inputs of these values.                                    */
/*********************************************************************/
trace o
numeric digits 15
cnt=Ø
clear
pref =strip(sysvar(syspref))
PARSE UPPER ARG P_dsname
if strip(P_dsname)='' then
do
   Call GETDBLST
end
else
do
   I_lstdsn = strip(P_dsname)
   I_lstdsn = strip(P_dsname,B,"'")
end
Call ALLOCDSN
Call ALLOCOUT
MAINØØØ:
do forever
  trace o
  "execio 1 diskr lstdd"
  if rc=2 then leave
  pull inrec
  if pos('DBNAME',inrec) > Ø | pos('------',inrec) > Ø then
  do
     hdr.1 = inrec
     "execio * diskw opds (stem hdr. "
     drop hdr.
     iterate
  end
  parse var inrec dbn tsn prt grn rl crl pct fp card npct nfp rest
  parse var rest reqrm oldrm newrm opgdf npgdf tsn2 dmy
  cnt = cnt+1
  if cnt//35 = Ø then
  do
     say 'Processing ' tsn prt
    "execio * diskw opds (stem hdr. "
  end
  nreqrm = trunc(card*grn/100)      /* 1 more than required */
  /* calc of space based on old pctfree */
  PCF = pct
  FREPG = fp
  Call CALCREQMT
  ototrm = ntotrm
  opgdif = npgdif
  ototpg = ntotpg
  /* calc of space based on new pctfree */
  PCF = npct
  FREPG = nfp
  Call CALCREQMT
  rtotrm = ntotrm
  rpgdif = npgdif
  rtotpg = ntotpg
  tpgdif = ototpg-rtotpg
  if rtotrm = nreqrm | nreqrm=0 then
  do
     rpgdif = npgdif
  end
  Call WRITEOUT
end
Say 'Press Enter to continue '
"execio 0 diskr lstdd (FINIS "
"execio * diskw opds (FINIS "
address tso "free f(lstdd)"
address tso "free f(opds)"
say 'See output in 'ods_name
exit
GETDBLST:
Say  'Give the input dataset ...'
Say  '(It must be a PS )'
pull I_lstdsn
/* I_lstdsn = HRXS.$$PAJKJ.OUTPUT.DMPPDBP5.D990208 */
I_lstdsn = strip(I_lstdsn)
I_lstdsn = strip(I_lstdsn,Both,"'")
x = SYSDSN("'"I_lstdsn"'")
if x ¬= OK then
do
   say; say '*** ERROR ' x ; say
   SIGNAL  GETDBLST
end
return
ALLOCDSN:
"ALLOCATE DD(lstdd) DSN('"I_lstdsn"') REUSE SHR"
if  rc>0 then
do
   say  'Failed during allocation of 'I_lstdsn
   exit(8)
end
```

```
DBNAME    TSNAME   PRT  GRTH  RECLEN NRECLN   PF   FP       CARD  NPF  NFP
---------------------------------------------------------------------------
DBADBØØ1  TSPACE1    1     2      87     87   3Ø   1Ø    6149338    4    Ø
DBADBØØ1  TSPACE1    2     2      87     87   3Ø   1Ø    6149338    4    Ø
DBADBØØ1  TSPACE1    3     2      87     87   3Ø   1Ø    6149338    4    Ø
DBADBØØ1  TSPACE1    4     2      87     87   3Ø   1Ø    6179434    4    Ø
DBADBØØ2  TSPACE1    Ø     1     312     63    Ø   1Ø     856578    3    Ø
DBADBØØ2  TSPACE2    Ø     Ø      33     33    Ø   1Ø       3Ø55    Ø    Ø
DBADBØØ2  TSPACE3    Ø     2      86     86    Ø   1Ø      22196    3    Ø
DBADBØØ2  TSPACE4    Ø     1      39     39    Ø   1Ø      4315Ø    2    Ø
DBADBØØ2  TSPACE5    Ø     1      18     18    Ø   1Ø      49661    2    Ø
DBADBØØ2  TSPACE6    Ø     1     23Ø     74    Ø   1Ø    1392867    2    Ø


 REQRM   OLDRM  NEWRM   OPGDIF   NPGDIF    TSNAME   OTOTPG   NTOTPG   PGDIF
-----------------------------------------------------------------------------
122986  3Ø5264 279518    19216        Ø   TSPACE1   211384   139759   71625
122986  3Ø5264 279518    19216        Ø   TSPACE1   211384   139759   71625
122986  3Ø5264 279518    19216        Ø   TSPACE1   211384   139759   71625
123588  321446 28Ø886    1931Ø        Ø   TSPACE1   212419   14Ø443   71976
  8565   85632  27634     1338        Ø   TSPACE1    14724    13817     9Ø7
     Ø     246  27634        2        Ø   TSPACE2       28    13817     9Ø7
   443    22Ø9    99Ø       47        Ø   TSPACE3      521      495      26
   431    4264    85Ø       41        Ø   TSPACE4      457      425      32
   496    4972   113Ø       22        Ø   TSPACE5      243      226      17
 13928  13926Ø  52564     2532        Ø   TSPACE6    27858    26282    1576
```

Figure 1: SPCCAL sample output

```
return
ALLOCOUT:
cts= time(S)
cd = date(U)
us_date = substr(cd,7,2)||substr(cd,1,2)||substr(cd,4,2)
odsn = pref||'.'||userid()||'.RCSPCCAL.D'||us_date
ods_name = odsn
xx = outtrap("zap.","*")
```

```
address tso "delete '"ods_name"'"
xx = outtrap("OFF")
address tso "alloc f(opds) new unit(hsm) space(2,2)",
          "tracks reuse dsname('"ods_name"')",
          "dsorg(ps) blksize(2000) lrecl(200) recfm(f b)"
return
WRITEOUT:
  grn =  trunc((grn*100),2)
  out.1 = left(dbn,10)||left(tsn,10)||right(prt,2)
  out.1 = out.1' 'right(ngrn,5)
  out.1 = out.1' 'right(rl,6)' 'right(crl,6)
  out.1 = out.1' 'right(pct,4)' 'right(fp,4)' 'right(card,10)
  out.1 = out.1||right(npct,5)||right(FREPG,5)
  out.1 = out.1' 'right(nreqrm,6)'  'right(ototrm,6)
  out.1 = out.1' 'right(rtotrm,6)
  out.1 = out.1'  'right(opgdif,7)'  'right(rpgdif,7)' 'right(tsn,10)
  out.1 = out.1'  'right(ototpg,7)'  'right(rtotpg,7)' 'right(tpgdif,7)
  "execio * diskw opds (stem out. "
  drop out.
return
CALCREQMT:
 navlspc = trunc(4074*(100-PCF)/100)
 nrpg = MIN(TRUNC(navlspc/crl),255)
 if nrpg <= 0 then nrpg = 1
 nunusd = 4074-(crl*nrpg)
 if nunusd = 0 then
    nroomfor = 0
 else
    nroomfor = trunc(nunusd/crl)
 npgusd = 2+trunc(card/nrpg)
 ntotpg = trunc(npgusd*(FREPG+1)/(MAX(FREPG,1)))
 totfpg = ntotpg - npgusd
 nKB = ntotpg*4
 if nroomfor = 0 then
 do
    ntotrm = totfpg*nrpg
 end
 else
    ntotrm = (npgusd*nroomfor) + (totfpg*nrpg)
 npgdif = ntotpg - npgusd
 return
```

SAMPLE INPUT FOR SPCCAL

```
--------------------------------------------------------------
DBNAME    TSNAME   PRT  GRTH  RECLEN NRECLN PF   FP       CARD
--------------------------------------------------------------
DBADB001  TSPACE1    1      2     87      87   30   10    6149338
DBADB001  TSPACE1    2      2     87      87   30   10    6149338
```

```
DBADBØØ1  TSPACE1    3      2      87      87   3Ø   1Ø    6149338
DBADBØØ1  TSPACE1    4      2      87      87   3Ø   1Ø    6179434
DBADBØØ2  TSPACE1    Ø      1     312      63    Ø   1Ø     856578
DBADBØØ2  TSPACE2    Ø      Ø      33      33    Ø   1Ø       3Ø55
DBADBØØ2  TSPACE3    Ø      2      86      86    Ø   1Ø      22196
DBADBØØ2  TSPACE4    Ø      1      39      39    Ø   1Ø      4315Ø
DBADBØØ2  TSPACE5    Ø      1      18      18    Ø   1Ø      49661
DBADBØØ2  TSPACE6    Ø      1     23Ø      74    Ø   1Ø    1392867
-----------------------------------------------------------------
```

SPCCAL SAMPLE OUTPUT

SPCCAL sample output is illustrated in Figure 1.

Jaiwant K Jonathan
DB2 DBA
QSS Inc (USA)

An environment-independent DB2 language interface

PROBLEM

When link editing a DB2 application it is a requirement to include the correct DB2 language interface (DSNELI, DSNCLI, DFSLI000 etc) in your application.

This requirement assumes that when you link edit you are aware of under which TP monitor your application will execute. For example, if your application will run under CICS you will have to include DSNCLI; if under TSO you will have to include DSNELI. This can make things rather complicated. Often developers include the wrong interface. Another scenario is that the module may execute under multiple TP monitors. In an on-line environment you may use CICS as your TP monitor and your batch may be IMS BMPs as well as some TSO batch. This scenario would require an application to be link edited into three different load libraries with each load module including its TP monitor- specific language interface. This assumes that your developer knows which environments the module will execute in and that, if the link edit is indeed done correctly, the STEPLIB/ JOBLIB concatenation is correct. A far simpler solution is to have a DB2 language interface that is TP monitor independent.

SOLUTION

This solution allows your developers to be oblivious of which TP monitor the module may execute under and does not require the application to be link edited with different language interfaces.

SOME TECHNICAL INSIGHT

When you code static SQL in an application it results in a V-type address constant for DSNHLI being generated. The external reference to DSNHLI is resolved (at link edit time) by the inclusion of modules such as DSNELI, DSNCLI, or DFSLI000. All of these modules happen to have a csect called DSNHLI which then resolves the external reference.

This solution proposes that at link edit time, instead of including a TP monitor-specific language interface, a generic language interface is included. To avoid confusion at this stage of the document we will call this DSNXLI. DSNXLI will determine at run-time which TP monitor it is currently executing under and in turn invoke the correct language interface.

This is a simple solution to implement. It is as simple as changing the SYSLIB concatenation on the linkedit JCL of your applications. It is also a simple solution to remove. This is achieved by relinking the load modules and replacing the new DSNXLI with DSNELI, DSNCLI, or DFSLI000 depending on the target execution environment.

Here is a simple diagram showing how the DB2 language interface is currently used:

Under CICS:

```
+----------------------------+
|           PGMA             |
|                            |
| EXEC SQL creates statement |
| v(DSNHLI)                  |
| linkedit include DSNCLI    |
|             +--------------+
|             |    DSNCLI    |
|             |    +---------+
|             |    | DSNHLI  |
+-------------+----+---------+
```

Under TSO:

```
+----------------------------+
|           PGMA             |
|                            |
| EXEC SQL creates statement |
| v(DSNHLI)                  |
| linkedit include DSNELI    |
|             +--------------+
|             |    DSNELI    |
|             |    +---------+
|             |    | DSNHLI  |
+-------------+----+---------+
```

Note that a different load module is required for each execution environment.

Here is a simple diagram showing how the new DSNXLI DB2 language interface could be implemented:

Under CICS/TSO/IMS:

```
+----------------------------+
|           PGMA             |
|                            |
| EXEC SQL creates statement |
| v(DSNHLI)                  |
| linkedit include DSNHLI    |
|             +--------------+
|             |    DSNXLI    |
|             | which TPM?   |
|             | CALL --------|------------+
+-------------+--------------+            |
                                          |
                                          |           +-------------+
                                          |  CICS?    | DSNDFHLI    |
                                          |           +-------------|
                                          |           |    +--------+
                                          |           |    | DSNCLI |
|           +----+--------+
|
|           +------------+
|  TSO?     | DSNIKJLI   |
|           +------------|
|           |    +-------+
|           |    | DSNELI|
|           +----+-------+
|
|           +------------+
|  IMS?     | DSNDFSLI   |
|           +------------|
            |   +--------+
            |   |DFSLIØØØ|
            +---+--------+
```

In order to determine which TP monitor is active, at run-time, the name of the program specified on the EXEC PGM= statement in the JCL is used. This program name is found in field JSCBPGMN in the JSCB. The JSCB can be found from the TCBJSCB pointer in the TCB control block.

The actual implementation of this solution is as follows. Assemble and link DSNHLI. This is the new interface, previously referred to as DSNXLI. Nothing too special here. It is re-entrant but you will be linking this into your application anyway so your application's linkedit attributes will take precedence.

Now assemble and link DSNCOMLI. This program is called from DSNHLI and it returns to DSNHLI the name of the language interface to use. You may be wondering why this logic was not incorporated into DSNHLI. It is because I decided the less logic that is linked into the application program the better. This way, if changes are necessary, there is no need for a relink of all applications using this solution. As you can see in the diagram above, the names of the language-specific interfaces are DSNDFHLI, DSNIKJLI, and DSNDFSLI for CICS, TSO, and IMS respectively. You could choose different names if you would like to avoid any future potential clash with modules delivered with the DB2 product.

The next step is to create the three TP-specific language interface modules as seen in the diagram above. These modules are identical apart from the fact that they have different IBM-delivered language interface modules linked into them. Because the code is identical for the three modules, I created a macro called DSNGENLI. DSNGENLI expects one parameter, which is the name of the program that the macro builds. The code is identical because, remember, that irrespective of the environment, the call is always to DSNHLI. In this case, this is the real IBM DSNHLI we are invoking.

The assembly JCL for these three language interfaces is the standard assembly/link. The only tricky part is making sure that you link the correct IBM language interface into your new TP-specific interface. Your JCL should look like this:

```
//       EXEC ASSCL,MBR=DSNDFSLI,PARM.LKED='RENT,XREF'
//LKED.SYSIN DD *
             INCLUDE INCLLIB(DFSLIØØØ)
//       EXEC ASSCL,MBR=DSNDFHLI,PARM.LKED='RENT,XREF'
//LKED.SYSIN DD *
             INCLUDE INCLLIB(DSNCLI)
//       EXEC ASSCL,MBR=DSNIKJLI,PARM.LKED='RENT,XREF'
//LKED.SYSIN DD *
             INCLUDE INCLLIB(DSNELI)
//*
```

Your INCLLIB DD statement on your LKED step should have both your IMS Reslib (IMS*.RESLIB) and the DB2 loadlib(DSN*.SDSNLOAD). The source for the three language interface modules is shown below.

For module DSNDFSLI:

```
*---------------------------------------------------------------------*
*-  CALLER OF DSNHLI UNDER IMS - LINKED WITH WITH INCL(DFSLIØØØ)    -*
*---------------------------------------------------------------------*
         DSNGENLI DSNDFSLI
```

For module DSNIKJLI:

```
*---------------------------------------------------------------------*
*-  CALLER OF DSNHLI UNDER TSO - LINKED WITH WITH INCL(DSNELI)      -*
*---------------------------------------------------------------------*
         DSNGENLI DSNIKJLI
```

For module DSNDFHLI:

```
*---------------------------------------------------------------------*
*-  CALLER OF DSNHLI UNDER CICS - LINKED WITH WITH INCL(DSNCLI)     -*
```

```
*---------------------------------------------------------------------*
         DSNGENLI DSNDFHLI
```

So, in summary, we have replaced the IBM-delivered language interface with a generic interface called DSNHLI. This calls DSNCOMLI to determine the TP monitor currently active and which language interface to use. DSNHLI then calls one of these interfaces, which has DSNHLI linked into it.

Just remember that, if you need to change DSNHLI or DSNGENLI, R1 on entry to the new DSNHLI has to be passed on unchanged to the real DSNHLI and to preserve the return code when returning.

Using this solution means that your developers no longer have to be concerned at link edit time which environment the module is destined for, and at run-time they are less likely to get abends caused by the wrong steplib concatenations. This, in addition to keeping three copies of the load module in three separate load libraries, makes for far fewer headaches.

```
DSNHLI   TITLE 'DB2 LANGUAGE INTEFACE(STUB)'
         EJECT
*----------------------------------------------------------------------*
*- THIS IS A REPLACEMENT DSNHLI MODULE                                -*
*- IT WILL CALL DSNCOMLI WHICH WILL PASS BACK A NAME OF A TP MONITOR -*
*- SPECIFIC LANGUAGE INTERFACE. THIS LANGUAGE INTERFACE WILL THEN    -*
*- BE LOADED AND INVOKED.                                             -*
*- REENTRANT NON REFRESHABLE SUBPOOL 252                              -*
*- NOT ELIGIBLE FOR LPA                                               -*
*----------------------------------------------------------------------*
DSNHLI   ESANTRY
MLAØØØ   DS    ØH
         ST    R1,A#PARMS                |SAVE PARMS ADDR
MLA1ØØ   DS    ØH                        |
         LOAD  EP=DSNCOMLI               |DSNCOMLI WILL RETURN TO US
         ST    RØ,DSNCOM@                |THE LANGUAGE INTERFACE MODULE
         LA    R2,DB2LI                  |NAME THAT SHOULD BE USED UNDER
         ST    R2,COMPARMS               |THE CURRENT TP MONITOR
         OI    COMPARMS,X'8Ø'            |
         LA    R1,COMPARMS               |
         L     R15,DSNCOM@               |
         BASSM R14,R15                   |IT WILL CHECK WHICH TPM ACTIVE
         DELETE EP=DSNCOMLI              |PREVENT USECNT>32K - S9Ø6 ABEN
         LOAD  EPLOC=DB2LI               |LANGUAGE INTERFACE MODULE
         ST    RØ,DB2LI@                 |WHICH WE LOAD AND INVOKE WITH
MLA2ØØ   DS    ØH                        |THE ORIGINAL PARMS
```

```
         L     R15,DB2LI@               |CALL TPM SPECIFIC INTERFACE
         L     R1,A#PARMS               |WITH PARMS PASSED TO US
         BASSM R14,R15                  |CALL OUR TP SPECIFIC LI
         ST    R15,RC                   |SAVE RC FROM REAL DSNHLI
         DELETE EPLOC=DB2LI             |LANGUAGE INTERFACE MODULE
MLA99Ø   DS    ØH                       |RETURN WITH POSSIBLE NON ZERO
         L     R15,RC                   |RC FROM REAL DSNHLI
         PRINT GEN
EXIT     ESAEXIT                        |
         LTORG                          |
*------------------WORKING STORAGE----------------------------*
** THIS AREA IS FOR STATIC DATA - THE SAME FOR EVERY INVOCATION
         SPACE 3
** THIS AREA IS FOR VOLATILE DATA AS IT IS GETMAINED - NO DECL CONSTS
** WS GENREGS AND WSLEN ARE REQUIRED BY ESANTRY
WS       DSECT
GENREGS  DS    18F                      |
STORLEN  DS    F                        |
A#PARMS  DS    F
RC       DS    F
COMPARMS DS    F
DB2LI    DS    CL8
DSNCOM@  DS    F
DB2LI@   DS    F
WSLEN    EQU   *-WS
         END   DSNHLI

DSNCOMLI TITLE 'DB2 COMMON LANGUAGE INTEFACE DRIVER'
         EJECT
*-------------------------------------------------------------------*
*- INSTRUCTIONS FOR USE:                                           -*
*- THIS MODULE SHOULD ONLY BE CALLED FROM DSNHLI(NON-IBM VERSION)  -*
*- THE NEW DSNHLI IS A "STUB" DESIGNED TO BE OPTIONALLY(BUT        -*
*- PREFERABLY) LINKED INTO THE APPLICATION PROGRAM.                -*
*- DSNHLI WILL LOAD AND INVOKE ME.                                 -*
*- HERE WE WILL DETERMINE WHICH TP MONITOR IS ACTIVE AND RETURN THE-*
*- NAME OF THIS TPM SPECIFIC LI MODULE ALL OF WHICH ARE CLONES     -*
*- DSNDFSLI FOR IMS/DL/I                                           -*
*- DSNDFHLI FOR CICS                                               -*
*- DSNIKJLI FOR TSO/E                                              -*
*- FACILITY IS PROVIDED FOR 2 OTHER INITIAL PROGRAMS AND ONE       -*
*- FURTHER INTERFACE (FOR ZAPPING) OTHERWISE YOU WILL HAVE TO      -*
*- HAND CODE                                                       -*
*- IF YOU CHOOSE TO NOT USE THIS MODULE SIMPLY LINK IN THE ORIGINAL-*
*- LANGUAGE INTERFACE.                                             -*
*-                                                                 -*
*- *-------------*                                                 -*
*- *- TP MONITOR -*                                                -*
*- *-------------*                                                 -*
*-        |                                                        -*
*-         |                                                      -*
*-         |                 *--------------*                     -*
*-         |                 *- DSNCOMLI   -*                     -*
*-         |                 *--------------*                     -*
*-         |                        |   LOAD+BR                   -*
*-         |                        |                             -*
*- *---------------*  BR    *--------------*                      -*
*- *- APPLICATION -*------->*- NEW DSNHLI -*                      -*
*- *---------------*        *--------------*                      -*
*-                                  |   LOAD+BR                   -*
*-              --------------------------------------            -*
*-              |                |                    |           -*
*-       *---------------* *---------------*  *---------------*   -*
*-       *- DSNDFSLI    -* *-  DSNDFHLI   -*  *-   DSNIKJLI  -*   -*
*-       *- LINKED WITH -* *- LINKED WITH -*  *- LINKED WITH -*   -*
*-       *- DFSLIØØØ    -* *- DSNCLI      -*  *-  DSNELI     -*   -*
*-       *---------------* *---------------*  *---------------*   -*
*-----------------------------------------------------------------*
         EJECT
         PRINT GEN
DSNCOMLI ESANTRY
*
MLAØØØ   DS    ØH
         ST    R1,A#PARMS                |SAVE PARMS ADDR
         MVC   RC,=F'Ø'                  |
MLAØØ1   NOP   MLA1ØØ                    |BYPASS INIT
         LOAD  EP=DSNCOMLI               |LOAD MYSELF TO ENSURE A
USECNT
         OI    MLAØØ1+1,X'FØ'            |>Ø (RESIDENT)
MLA1ØØ   DS    ØH
         USING IEZJSCB,R9                |JSCB
         L     R3,PSATOLD-PSA(Ø)         |TCB
         L     R9,TCBJSCB-TCB(R3)        |JSCBLOCK
         CLC   JSCBPGMN(6),=C'DFSRRC'    |DO WE HAVE IMS?
         BE    MLA11Ø                    |YES IMS ACTIVE
         CLC   JSCBPGMN(8),=C'DFHSIP  '  |CICS?
         BE    MLA12Ø                    |YES CICS IS OUR TPM
         CLC   JSCBPGMN(6),=C'IKJEFT'    |TSO IS TMP
         BE    MLA13Ø                    |YES TSO IS OUR TPM
         CLC   JSCBPGMN(6),=C'XPTSO '    |XPEDITER INTERACTIVE - IMS
         BE    MLA11Ø                    |YES IMS IS OUR TPM
         CLC   JSCBPGMN(7),=C'XPBATCH'   |XPEDITER UNATTENDED - IMS
         BE    MLA11Ø                    |YES IMS IS OUR TPM
         CLC   JSCBPGMN(8),ZAPIPGM1      |EMPTY ONE TO ZAP
         BE    MLA16Ø                    |YES XXXX IS OUR TPM
         CLC   JSCBPGMN(8),ZAPIPGM2      |EMPTY ONE TO ZAP
         BE    MLA17Ø                    |YES XXXX IS OUR TPM
*                                        |AT THIS STAGE WE DONT KNOW
WHO
*                                        |THE HELL IS OUR TPM
```

```
         B     MLA18Ø                    |SO WE DEFAULT ONE
MLA11Ø   DS    ØH                        |RUNNING UNDER IMS
         MVC   DSNLI,=C'DSNDFSLI'        |DL/1 DB2 LANGUAGE INTERFACE
         B     MLA2ØØ                    |GO COMMON
MLA12Ø   DS    ØH                        |RUNNING UNDER CICS
         MVC   DSNLI,=C'DSNDFHLI'        |CICS DB2 LANGUAGE INTERFACE
         B     MLA2ØØ                    |GO COMMON
MLA13Ø   DS    ØH                        |RUNNING UNDER TSO
         MVC   DSNLI,=C'DSNIKJLI'        |TSO DB2 LANGUAGE INTERFACE
         B     MLA2ØØ                    |GO COMMON
MLA16Ø   DS    ØH                        |RUNNING UNDER ???
         MVC   DSNLI,ZAPLI1              |EMPTY INTERFACE TO ZAP
         B     MLA2ØØ                    |GO COMMON
MLA17Ø   DS    ØH                        |RUNNING UNDER ???
         MVC   DSNLI,ZAPLI2              |EMPTY INTERFACE TO ZAP
         B     MLA2ØØ                    |GO COMMON
MLA18Ø   DS    ØH                        |RUNNING UNDER IMS
         MVC   DSNLI,=C'DSNHLI  '        |LAST CHANCE-UNKNOWN TPM
         B     MLA2ØØ                    |GO COMMON|KEEP THIS ONE!!
MLA2ØØ   DS    ØH                        |
         L     R1,A#PARMS                |GET PARMS ADDR
         L     R2,Ø(R1)                  |SECOND PARM
         MVC   Ø(8,R2),DSNLI             |AND POPULATE LI NAME
MLA9ØØ   DS    ØH                        |LEAVING
         L     R15,RC                    |GET RC BACK
MLA99Ø   DS    ØH                        |RETURN WITH POSSIBLE NON ZERO
         ESAEXIT
         DC    CL32'ZAP HERE TO ADD NEW PGM/LI->'
ZAPIPGM1 DC    CL8'XXXXXXXX'             |TP MONITOR-INIT PGM NAME
ZAPLI1   DC    CL8'XXXXXXXX'             |AND ITS LI MODULE
ZAPIPGM2 DC    CL8'XXXXXXXX'             |DITTO ABOVE
ZAPLI2   DC    CL8'XXXXXXXX'             |
         DS    ØF
*------------------WORKING STORAGE----------------------------*
WS       DSECT
GENREGS  DS 18F                             |
STORLEN  DS F                               |
A#PARMS  DS    F
RC       DS    F
DSNLI    DS    CL8
         DS    CL52
WSLEN    EQU   *-WS
         IHAPSA
         IKJTCB
         DSECT
ØØØ153ØØ
         IEZJSCB
ØØØ157ØØ
         END   DSNCOMLI
         MACRO
         DSNGENLI &NAME
&NAME    TITLE 'DB/2 LANGUAGE INTEFACE CALLER'
         EJECT
.*---------------------------------------------------------------*.
.*  DSNGENLI &NAME                                              -*.
.*  MACRO TO GENERATE GENERIC MODULES DSNDFSLI DSNDFHLI DSNIKJLI -*.
.*  EACH OF WHICH WILL BE LINKED WITH THEIR SPECIFIC TPMON DB2   -*.
.*  LANGUAGE INTERFACES - DSNELI DFSLIØØØ DSNCLI ETC             -*.
.*---------------------------------------------------------------*.
         PRINT ON
&NAME    ESANTRY
*
MLAØØØ   NOP   MLA1ØØ
         ST    R1,A#PARMS                |SAVE PARMS ADDR
MLAØØ1   NOP   MLA1ØØ
         OI    MLAØØ1+1,X'FØ'            |BYPASS INIT
         LOAD  EP=&NAME                  |FIX MYSELF-ONCE USECNT>Ø
MLA1ØØ   DS    ØH
         L     R1,A#PARMS                |GET PARMS
         CALL  DSNHLI                    |EVENTUALLY WE CALL THE REAL ONE
         ST    R15,RC                    |SAVE RC FROM REAL DSNHLI
MLA99Ø   DS    ØH                        |RETURN WITH POSSIBLE NON ZERO
         L     R15,RC                    |RC FROM REAL DSNHLI
         ESAEXIT
*-----------------WORKING STORAGE----------------------------*
WS       DSECT
GENREGS  DS    18F                       |
STORLEN  DS    F                         |
A#PARMS  DS    F
RC       DS    F
WSLEN    EQU   *-WS
         END   &NAME
         MEND
```

Nick Baguley
Applications Architect
Independent (South Africa)

DB2 news

Neon Systems has announced general availability of its Shadow Console interleaved management, monitoring, and debugging tool, designed to ensure the performance and availability of composite applications that integrate application platform suites with mainframe data and transactions.

Through its Windows-based GUI, it offers application developers and production operations staff a consolidated end-to-end view of the middleware component from the initial API adapter call in the application platform suite to the back-end data or transaction source on the mainframe.

Using diagnostic and control facilities inherent in other Shadow products, it consolidates information from the Shadow Client interfaces and data from the Shadow Server, which resides on z/OS.

It allows both mainframe and non-mainframe users to identify and resolve problems that may be affecting the performance and availability of an application, both during the development cycle and when the application is deployed in production environments.

The software can be deployed with leading J2EE and .NET application platform suites to provide customers with J2CA, JDBC, or ODBC access to mainframe data sources and transaction environments, supporting DB2, CICS/TS, IMS/TM, IMS/DB, VSAM, ADABAS, Natural/ACI, flat files, IDMS, and other z/OS data and transactional sources.

For further information contact:
NEON Systems, 14100 Southwest Freeway, Suite 500, Sugarland, TX 77478, USA.
Tel: (281) 491 4200.
URL: http://www.neonsys.com/solutions/shadow.

* * *

CONNX Solutions has announced Version 8.8 of its CONNX data access middleware, now with a range of performance and feature enhancements. Support for VSAM VSE data sources, which provides real-time high-performance access to VSAM files under CICS partitions on the VSE operating system, has also been included in the release. Also, direct support for Microsoft .NET technology has been added with the introduction of a pure CONNX OLE DB Provider.

The new release also includes a new product called the CONNX Data Synchronization Tool, which allows users to move enterprise data from any source location to any target data source. The Windows-based tool uses a technique of hash keys to detect when records are updated, deleted, or inserted.

It acts as a reusable data access framework, supporting DB2, Oracle, C-ISAM, VSAM, RMS, RDB, PostGreSQL, DBMS, Dataflex, POWERflex, SQL Server, Sybase, and Informix and any OLE DB, ODBC, or JDBC data source.

For further information contact:
CONNX Solutions, 1800 112th Avenue NE, Suite #150, Bellevue, WA 98004, USA.
Tel: (425) 519 6600.
URL: http://www.connx.com/products/products.html.