In this issue

3  A REXX/ISPF program to automate the copying of rows in DB2 tables across DB2 subsystems using XCOM
23  SET operators
25  A program to fix tablespaces/indexes with RESTRICTed access – part 2
43  Using UDB DB2 utilities to generate explain output for dynamic SQL
50  DB2 news
Disclaimer
Readers are cautioned that, although the information in this journal is presented in good faith, neither Xephon nor the organizations or individuals that supplied information in this journal give any warranty or make any representations as to the accuracy of the material it contains. Neither Xephon nor the contributing organizations or individuals accept any liability of any kind howsoever arising out of the use of such material. Readers should satisfy themselves as to the correctness and relevance to their circumstances of all advice, information, code, JCL, and other contents of this journal before making any use of it.

Contributions
When Xephon is given copyright, articles published in DB2 Update are paid for at the rate of $160 (£100 outside North America) per 1000 words and $80 (£50) per 100 lines of code for the first 200 lines of original material. The remaining code is paid for at the rate of $32 (£20) per 100 lines. 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.
A REXX/ISPF program to automate the copying of rows in DB2 tables across DB2 subsystems using XCOM

Analysts and programmers are constantly asking for rows from production DB2 tables to be copied to maintenance systems to solve/simulate incidents and also to be used for testing. This work can consume a great deal of the DB2 support personnel’s time, besides being repetitive.

It is to help with this requirement that the program DB2COPIX was developed. It allows rows to be copied between DB2 tables by people who have no knowledge of DB2.

At our installation, it is used to copy rows of tables from the production environment (the first DB2 subsystem) to (a second DB2 subsystem) the development, quality, or test environments via XCOM, since neither environment has shared disks to facilitate the transmission of data. These target environments reside on another mainframe/partition.

The program allows us to copy rows between DB2 tables (with the same structure and name but in different DB2 subsystems), taking as input the DB2 table or a full imagecopy (any version) of this. It also allows all the rows in the table to be copied, or just a selection of them (by means of WHERE).

The main characteristics and functions of the program DB2COPIX are:

- The main program is REXX/ISPF, which invokes a routine in Assembler. It is executed in TSO with OS/390 V2.9, DB2 V6.1, XCOM V3.0, BMC Unload Plus (or DSNTIAUL), and BMC Load Plus (or DSNUTILB).

- When being executed, the program displays the main panel where the user enters the name of the DB2 table to copy – the input can be the DB2 table in production or a full image copy version (the default is Version 0, the most recent image.
copy), the output defines the environment it is copied to (the default is development), and optionally any parameters for the selection of lines using WHERE. The program builds and submits the jobs that unload the rows (it uses BMC Unload, but could use DSNTIAUL), transmit the files via XCOM, orders the execution of the jobs in maintenance via XCOM, and loads the rows in the target DB2 table (it uses BMC Load, but could use IBM Load).

- After the job is submitted, the user’s TSO enters a wait state (the program calls an Assembler routine, ESPERA, which uses STIMER to generate a wait of 20 seconds, which is amendable) looking for the return of the load job. If this doesn’t arrive, the following message is sent: ‘copying the DB2 table *tablename*, job *jobname* current hour (*hhmmss*)’. This cycle repeats until the load job is detected in the spool or the limit of cycles is completed (30 when the input is the tablespace, 60 when it is a full imagecopy, and 60 when the copy includes all the rows). After that the wait state concludes.

The program examines the return job, from where it obtains the quantity of rows copied, which is shown in the main panel. If the copy doesn’t conclude, there must have been some abnormal termination, so an explanatory message is shown in the same panel.

The program checks the validity (syntax and columns existence) of the WHERE in the case of a copy with rows selected, before generating the jobs, and it sends an explanatory message to the main panel in the event of not being valid.

Also, in the case of a copy operation with lines selected, the program generates the jobs with the appropriate deletion of lines in the target table to avoid the possible duplication of rows during the load. In the case of a copy of all the rows in a DB2 table, the program generates in the load job (before the job step) a step with ALTER PRIQTY, ALTER SECQTY for the tablespace, and indexes with the production values.

It creates its own log, where the actions of the users are recorded, such as the option chosen, the user’s RACF ID, the name of the
DB2 table, whether the input is a tablespace or a full imagecopy, whether the copy is for a selection of lines, the output environment, the date and time of the execution of DB2COPIX, the jobname and jobnumber of the load job, and the number of rows copied.

The program supplies help panels (pressing F1 key). In addition it has explanatory and error messages in two versions – short and long (accessible by pressing the F1 key when it shows a short message). It uses a library of messages (ISPMLIB).

The program needs to replace some variables (hardcoded in the program):

- SSIP – DB2 subsystem of the input table.
- SSID – DB2 subsystem of the output table.
- OWNERP – owner of the input table.
- OWNERP – owner of the output table.
- LUDEVE – LU of the target system (where the output DB2 subsystem resides). Included in the XCOM job transmission.
- NODEPROD – JES2 node identifier of the input system (where the output DB2 subsystem resides). Included in the statement /*ROUTE PRINT NODEPROD of the LOAD job.
- USERFTP – RACF user and password associated with the file transfer program.
- PASSWORD – included in the SYSIN01 file of the XCOM job transmission.
- USERFTP – USERID=USERFTP and PASSWORD=USERFTP.

The benefits from using the program DB2COPIX are:

- Faster response to incidents because non-specialized personnel can perform this task without the support of a DB2 specialist.
- Improves the quality of service.
- Greater productivity.
- Testing and development are speeded up in the maintenance partition.
- It frees specialized DB2 personnel from performing this task.

```rexx
/* REXX  DB2COPIX  CARLOS-OSORIO@EXCITE.COM */
/* REPLACE THE FOLLOWING VARIABLES (HARDCODE) */
/* .SSIP  DB2 SUBSYSTEM OF THE INPUT TABLE */
/* .SSID  DB2 SUBSYSTEM OF THE OUTPUT TABLE */
/* .OWNERP  OWNER OF THE INPUT TABLE */
/* .OWNERP  OWNER OF THE OUTPUT TABLE */
/* .LUDEVE  LU OF THE TARGET SYSTEM */
/* (WHERE THE OUTPUT DB2 SUBSYSTEM RESIDES) */
/* INCLUDED IN THE XCOM JOB TRANSMISSION */
/* .NODEPROD  JES2 NODE IDENTIFY OF THE INPUT SYSTEM */
/* (WHERE THE OUTPUT DB2 SUBSYSTEM RESIDES) */
/* INCLUDED IN STATEMENT: ROUTE PRINT NODEPROD */
/* OF THE LOAD JOB */
/* .USERFTP/ PASSWORD ASSOCIATED WITH FILE TRANSFER*/
/* PASSWORD INCLUDED IN THE XCOM JOB TRANSMISSION */
/* USERFTP USERID=USERFTP */
/* PASSWORD=USERFTP */
/*----------------------------------------------------------------------*/
TRACE OFF
"ISPEXEC LIBDEF ISPPLIB DATASET ID('YOUR.PROD.PANEL.LIB')"
"ISPEXEC LIBDEF ISPMLIB DATASET ID('YOUR.PROD.MESSAGE.LIB')"
TABLA = ''
VERSION = '000'
DO FOREVER
/* DEFAULTS */
DB2COP = Ø
ENTORNO = 'D'  /* TARGET ENVIRONMENT = DEVELOPMENT */
INPUT = 'I'  /* INPUT = IMAGECOPY */
HPQTY = 'N'  /* PRIQTY > 10000 TRACKS */
MODO = 'R'  /* LOAD REPLACE */
SQLWHERE = 'N'  /* NOT WHERE CONDITION */
WHERE1='' ; WHERE2='' ; WHERE3='' ; WHERE4='' ; WHERE5='' ; WHERE6='' ;
DO WHILE DB2COP = Ø
  "ISPEXEC DISPLAY PANEL(DB2COPIX)"
  IF RC = 8 THEN
    DO
      "ISPEXEC LIBDEF ISPPLIB"
      "ISPEXEC LIBDEF ISPMLIB"
    EXIT
```
ELSE DO
  MSG1 = '
  MSG2 = '
  HORARIORC = Ø
  CALL HORARIO
  IF HORARIORC = Ø THEN
    DO
      ALLOCSYSRC = Ø
      CALL ALLOCSYS
      IF ALLOCSYSRC = Ø THEN
        DO
          SYSIN.1 = "SELECT * FROM SYSIBM.SYSTABLES " ,
            "WHERE NAME = '"TABLA'";
          "EXECIO * DISKW SYSIN (STEM SYSIN."
          "EXECIO Ø DISKW SYSIN (FINIS"
          "FREE DDNAME(SYSIN)"
          CHKTABLARC = Ø
          CALL CHKTABLA
          IF CHKTABLARC = Ø THEN
            DO
              IF (WHERE1 <> '') THEN SQLWHERE = 'Y'
              CHKPQTYRC = Ø
              CALL CHKPQTY
              IF CHKPQTYRC = Ø THEN
                DO
                  CHKWHEREERC = Ø
                  CALL CHKWHERE
                  IF CHKWHEREERC = Ø THEN
                    DO
                      GENJOBRC = Ø
                      CALL GENJOB
                      IF GENJOBRC = Ø THEN
                        DO
                          MSG1 = 'COPYING THE TABLE 'TABLA
                          MSG2 = ''
                          VERSION = 'ØØØ'
                          MODO = 'R'
                          DB2COP = 1
                        END
                      END
                    END
                  END
                END
              END
            END
          END
        END
      END
    END
  END
END
EXIT
/*-------- R U T I N A S -------- CARLOS-OSORIO@EXCITE.COM --------*/
HORARIO:
AUTORIZ = ' '
IF INPUT = ' T ' THEN
DO
HORA = TIME(' H ')
DIA = SUBSTR(DATE(' W '), 1, 1)
IF ((HORA > 8 & HORA < 13) | (HORA > 15 & HORA < 18)) & DIA <> ' S ' THEN
DO
MSG1 = ' TIME NOT AUTHORIZED TO COPY DB2 TABLES '
MSG2 = ' RESTRICTION: MONDAY-FRIDAY 09:00-13:00 16:00-18:00 '
AUTORIZ = ' H '
SQLWHERE = ' C '
MODO = ' C '
JOBOUT = USERID() || '000(JOB0000)' 
FI LAS = Ø
CALL GRABALOG
HORARIO_RC = 1
END
END
RETURN
/*----------------------------------------------------------------*/
ALLOCSYS:
DSNIN = USERID() || '.TMP' || '.IN' || TIME('S ')
DSNREC = USERID() || '.TMP' || '.REC' || TIME('S ')
DSNPRINT = USERID() || '.TMP' || '.PRT' || TIME('S ')
ADDRESS TSO "ALLOC FILE(SYSIN) DATASET(''DSNIN'' ) ",
"NEW CAT REUSE UNIT(SYSDA) ",
"LRECL(80) BLKSIZE(27920) RECFM(F B) SPACE(1,1) CYL"
IF RC <> Ø THEN
DO
ADDRESS ISPSEXEC "SETMSG MSG(DBC001)"
ALLOCSYSRC = 1
RETURN
END
ADDRESS TSO "ALLOC FILE(SYSREC00) DATASET(''DSNREC'' ) ",
"NEW CAT REUSE UNIT(SYSDA) ",
"LRECL(100) BLKSIZE(27900) RECFM(F B) SPACE(1,1) CYL"
IF RC <> Ø THEN
DO
ADDRESS ISPSEXEC "SETMSG MSG(DBC002)"
ALLOCSYSRC = 1
RETURN
END
ADDRESS TSO "ALLOC FILE(SYSPRINT) DATASET(''DSNPRINT'' ) ",
"NEW CAT REUSE UNIT(SYSDA) ",
"LRECL(133) BLKSIZE(27930) RECFM(F B) SPACE(1,1) CYL"
IF RC <> Ø THEN
DO
ADDRESS ISPEXEC "SETMSG MSG(DBC003)"
ALLOCSYSRC = 1
RETURN
END

RETURN
*/

CHKTABLA:
ADDRESS TSO "ALLOC FILE(SYSIN)" "DATASET('"DSNIN")" OLD REUSE"
ADDRESS TSO "ALLOC FILE(SYSREC00)" "DATASET('"DSNREC")" OLD REUSE"
ADDRESS TSO "ALLOC FILE(SYSPRINT)" "DATASET('"DSNPRINT")" OLD REUSE"
ADDRESS TSO "ALLOC FILE(SYSPUNCH) DUMMY"
PUSH "END"
PUSH "RUN PROGRAM(DSNTIAUL) PLAN(DSNTIAUL) PARMS('SQL')"
ADDRESS TSO "DSN SYSTEM(SSIP)"
ADDRESS TSO "EXECIO * DISKR SYSREC00 (STEM REC00. FINIS"
ADDRESS TSO "FREE FILE(SYSIN)"
ADDRESS TSO "FREE FILE(SYSREC00)"
ADDRESS TSO "FREE FILE(SYSPRINT)"
ADDRESS TSO "FREE FILE(SYSPUNCH)"
CALL DELTMP
IF REC00.0 = 0 THEN
DO
ADDRESS ISPEXEC "SETMSG MSG(DBC010)"
CHKTABLARC = 1; MSG1 = ' '; MSG2 = ''
RETURN
END
IF REC00.0 > 1 THEN
DO
ADDRESS ISPEXEC "SETMSG MSG(DBC011)"
CHKTABLARC = 1; MSG1 = ' '; MSG2 = ''
RETURN
END
IF SUBSTR(REC00.1, 29, 1) = 'V' THEN
DO
ADDRESS ISPEXEC "SETMSG MSG(DBC012)"
CHKTABLARC = 1; MSG1 = ' '; MSG2 = ''
RETURN
END
IF SUBSTR(REC00.1, 29, 1) = 'A' THEN
DO
ADDRESS ISPEXEC "SETMSG MSG(DBC013)"
CHKTABLARC = 1; MSG1 = ' '; MSG2 = ''
RETURN
END
IF SUBSTR(REC00.1, 29, 1) = 'T' THEN
DO
DBNAME = SUBSTR(REC00.1, 30, 8)
IF SUBSTR(DBNAME, 1, 3) = 'BVB' THEN
DBNAME0 = SUBSTR(DBNAME, 1, 5) || ENTONO || SUBSTR(DBNAME, 7, 2)
ELSE DO

ADDRESS ISPSEXEC "SETMSG MSG(DBC014)"
CHKTABLARC = 1; MSG1 = ''; MSG2 = ''
RETURN
END
TSNAME = SUBSTR(REC00.1,38,8)
CHKTABLARC = 0
RETURN
END
RETURN
/
*---------------------------------------------------------------*/
CHKPQTY:
CALL ALLOCSYS
SYSIN.1 = "SELECT HEX(PQTY), HEX(PARTITION)"
SYSIN.2 = "FROM SYSIBM.SYSTABLEPART"
SYSIN.3 = "WHERE TSNAME = 'TSNAME'
SYSIN.4 = "  AND DBNAME = 'DBNAME'
SYSIN.5 = "ORDER BY PARTITION DESC;"
"EXECIO * DISKW SYSIN (STEM SYSIN.)
"EXECIO Ø DISKW SYSIN (FINIS"
ADDRESS TSO "ALLOC FILE(SYSIN)" "DATASET( "DSNI") OLD REUSE"
ADDRESS TSO "ALLOC FILE(SYSREC00)" "DATASET( "DSNREC") OLD REUSE"
ADDRESS TSO "ALLOC FILE(SYSPRINT)" "DATASET( "DSNPRINT") OLD REUSE"
ADDRESS TSO "ALLOC FILE(SYSPUNCH) DUMMY"
PUSH "END"
PUSH "RUN PROGRAM(DSNTIAUL) PLAN(DSNTIAUL) PARMS('SQL')"
ADDRESS TSO "DSN SYSTEM(SSIP)"
ADDRESS TSO "EXECIO * DISKR SYSREC00 (STEM REC00. FINIS"
ADDRESS TSO "FREE FILE(SYSIN)"
ADDRESS TSO "FREE FILE(SYSREC00)"
ADDRESS TSO "FREE FILE(SYSPRINT)"
ADDRESS TSO "FREE FILE(SYSPUNCH)"
CALL DELTMP
PRIQTY = (X2D(SUBSTR(REC00.1,1,8))) * 4
PART = X2D(SUBSTR(REC00.1,9,4))
IF PRIQTY > 5000 THEN HPQTY = 'Y' /* 10 000 TRKS */
CALL CHKPQTYI X
RETURN
/
*---------------------------------------------------------------*/
CHKPQTYIX:
SYSIN.1=''; SYSIN.2=''; SYSIN.3='';
CALL ALLOCYS
SYSIN.1 = "SELECT NAME FROM SYSIBM.SYINDEXES"
SYSIN.2 = "WHERE TBNAME = 'TABLA' AND DBNAME = 'DBNAME';"
"EXECIO * DISKW SYSIN (STEM SYSIN.)
"EXECIO Ø DISKW SYSIN (FINIS"
ADDRESS TSO "ALLOC FILE(SYSIN)" "DATASET( "DSNI") OLD REUSE"
ADDRESS TSO "ALLOC FILE(SYSREC00)" "DATASET( "DSNREC") OLD REUSE"
ADDRESS TSO "ALLOC FILE(SYSPRINT)" "DATASET( "DSNPRINT") OLD REUSE"
ADDRESS TSO "ALLOC FILE(SYSPUNCH) DUMMY"
PUSH "END"
PUSH "RUN PROGRAM(DSNTIAUL) PLAN(DSNTIAUL) PARMS('SQL')"
ADDRESS TSO "DSN SYSTEM(SSSIP)"
ADDRESS TSO "EXECIO * DISKR SYSRECØØ (STEM IX. FINIS"
ADDRESS TSO "FREE FILE(SYSIN)"
ADDRESS TSO "FREE FILE(SYSRECØØ)"
ADDRESS TSO "FREE FILE(SYSPRINT)"
ADDRESS TSO "FREE FILE(SYSPUNCH)"
CALL DELTMP
DO I = 1 TO IX.Ø
   IX.I = SUBSTR(IX.I, 3, 8)
   SYSIN.1 = ''; SYSIN.2 = '';
   CALL ALLOC SYS
   SYSIN.1 = "SELECT HEX(MAX(PQTY)) FROM SYSIBM.SYSINDEXPART"
   SYSIN.2 = "WHERE IXNAME = '"IX.I"';"
   "EXECIO * DISKW SYSIN (STEM SYSIN "
   "EXECIO Ø DISKW SYSIN FINIS"
ADDRESS TSO "ALLOC FILE(SYSIN)"
   "DATASET('DSNIN') OLD REUSE"
ADDRESS TSO "ALLOC FILE(SYSRECØØ)"
   "DATASET('DSNREC') OLD REUSE"
ADDRESS TSO "ALLOC FILE(SYSPRINT)"
   "DATASET('DSNPRINT') OLD REUSE"
ADDRESS TSO "ALLOC FILE(SYSPUNCH) DUMMY"
   PUSH "END"
   PUSH "RUN PROGRAM(DSNTIAUL) PLAN(DSNTIAUL) PARMS('SQL')"
ADDRESS TSO "DSN SYSTEM(SSSIP)"
ADDRESS TSO "EXECIO * DISKR SYSRECØØ (STEM RECØØ. FINIS"
ADDRESS TSO "FREE FILE(SYSIN)"
ADDRESS TSO "FREE FILE(SYSRECØØ)"
ADDRESS TSO "FREE FILE(SYSPRINT)"
ADDRESS TSO "FREE FILE(SYSPUNCH)"
CALL DELTMP
PRIQTYI.X.I = (X2D(SUBSTR(RECØØ.1, 1, 8))) * 4
END
RETURN

/*----------------------------------------------------------------*/
CHKWHER:
IF SQLWHERE = 'Y' THEN
DO
   DO I = 1 TO 8
      SYSIN.1 = ''
   END
MODO = 'S'
CALL ALLOC SYS
   SYSIN.1 = "SELECT COUNT(*) FROM OWNERP."TABLA" WHERE "
   SYSIN.2 = WHERE1
   IF WHERE2 <> '' THEN DO; SYSIN.3 = WHERE2;
      IF WHERE3 <> '' THEN DO; SYSIN.4 = WHERE3;
         IF WHERE4 <> '' THEN DO; SYSIN.5 = WHERE4;
            IF WHERE5 <> '' THEN DO; SYSIN.6 = WHERE5;
               IF WHERE6 <> '' THEN DO; SYSIN.7 = WHERE6;
                  SYSIN.8 = '';
         END;
   END;
END;
ELSE SYSIN.7 = ";"
END;
ELSE SYSIN.6 = ";"
END;
ELSE SYSIN.5 = ";"
END;
ELSE SYSIN.4 = ";"
END;
ELSE SYSIN.3 = ";"
"EXECIO * DISKW SYSIN STEM SYSIN."
"EXECIO Ø DISKW SYSIN FINIS"
ADDRESS TSO "ALLOC FILE(SYSIN)" DATASET("DSNIN") OLD REUSE
ADDRESS TSO "ALLOC FILE(SYSRECØØ)" DATASET("DSNREC") OLD REUSE
ADDRESS TSO "ALLOC FILE(SYSPRINT)" DATASET("DSNPRINT") OLD REUSE
ADDRESS TSO "ALLOC FILE(SYSPTHUNCH) DUMMY"
PUSH "END"
PUSH "RUN PROGRAM(DSNTIAUL) PLAN(DSNTIAUL) PARMS('SQL')"
ADDRESS TSO "DSN SYSTEM(SSIP)"
IF RC <= 4 THEN
DO
    CHKWHEREC = Ø
    RETURN
END
ELSE
DO
    WHERE = 'N'
    MODO = 'R'
    ADDRESS ISPEXEC "SETMSG MSG(DBCØ51)"
    CHKWHEREC = 1; MSG1 = "; MSG2 = ";
    RETURN
END
END
RETURN

/*------------------------------   CARLOS-OSORIO@EXCITE.COM   ----*/

GENJOB:
DSNJOB = USERID() || '.TMP' || '.JOB' || TIME('S')
ADDRESS TSO "ALLOC FILE(JOB) DATASET("DSNJOB")",
    "LRECL(Ø) BLKSIZE(2792Ø) RECFM(F B) SPACE(1,1) CYL"
IF RC <> Ø THEN
    ADDRESS ISPEXEC "SETMSG MSG(DBCØ21)"
    GENJOBRC = 1
    RETURN
END
BMCCNTL = "OWNERP.ªTABLA.ªSYSCNTL.ªENTORNO || TIME('S')"
BMCCNTLO = OVERLAY('D', BMCCNTL, 4, 1)
BMCREC = "OWNERP.ªTABLA.ªSYSREC.ªENTORNO || TIME('S')"
BMCRECO = OVERLAY('D', BMCREC, 4, 1)
JOBSUFX = SUBSTR(TIME('S'),3,3)
J O B S U F C  =  S U B S T R ( ( J O B S U F X + 1 ) , 1 , 3 )
J O B X C O M  =  U S E R I D ( )  ||  J O B S U F X
J O B C O P Y  =  U S E R I D ( )  ||  J O B S U F C
C A L L  G E N J O B L O A D
" N E W S T A C K"
Q U E U E  " / "  M S G L E V E L = ( 1 , 1 ) "
Q U E U E  " / "  S T E P U 1  E X E C  P G M = A D U U M A I N ,  R E G I O N = Ò M , "
Q U E U E  " / "  P A R M = ' S S I P , ,  N E W , ,  M S G L E V E L ( 1 ) ' "
Q U E U E  " / "  S Y S P R I N T  D D  S Y S O U T = "*
Q U E U E  " / "  U T P R I N T  D D  S Y S O U T = "*
Q U E U E  " / "  S Y S O U T  D D  S Y S O U T = "*
Q U E U E  " / "  S Y S I N  D D  *"
Q U E U E  " *"
Q U E U E  " U N L O A D"
I F  I N P U T  = ' I '  T H E N
  D O
    I F  V E R S I O N  = ' Ò '  T H E N  Q U E U E  "  I N F I L E  I M A G E C O P Y  F U L L  Ò '
    E L S E  Q U E U E  "  I N F I L E  I M A G E C O P Y  F U L L  ' "  V E R S I O N"
    Q U E U E  "  S H R L E V E L  C H A N G E"
    Q U E U E  "  S E L E C T  *  F R O M  O W N E R P . ' T A B L A"
  E N D
E L S E
  D O
    Q U E U E  "  S H R L E V E L  C H A N G E  C O N S I S T E N T  Y E S  X B M I D  X B M P"
    Q U E U E  "  D I R E C T  A U T O"
    Q U E U E  "  S E L E C T  *  F R O M  O W N E R P . ' T A B L A"
  E N D
I F  S Q L W H E R E  = ' Y '  T H E N
  D O
    Q U E U E  "  W H E R E"
    D O  I  =  2  T O  8  W H I L E  S Y S I N . I  <> ' '  Q U E U E  S Y S I N . I
  E N D
E N D
Q U E U E  " *"
I F  H P Q T Y  = ' N '  T H E N
  D O
    Q U E U E  " / /  S Y S R E C  D D  D S N = ' B M C R E C ' , "
    Q U E U E  " / /  S P A C E = ( C Y L , ( 1 5 0 0 , 1 0 0 ) , R L S E ) , "
    Q U E U E  " / /  U N I T = S Y S D A "
  E N D
E L S E
  D O
    Q U E U E  " / /  S Y S R E C  D D  D S N = ' B M C R E C ' , "
    Q U E U E  " / /  U N I T = T A P E ,  V O L = ( , , 3 0 ) ,  D I S P = ( N E W , C A T L G ) "
  E N D
Q U E U E  " / /  S Y S C N T L  D D  D S N = ' B M C C N T L ' , "

QUEUE " // DISP=(NEW, CATLG, DELETE), "
QUEUE " // SPACE=(TRK, (1, 1), RLSE), "
QUEUE " // UNIT=SYSDA"
QUEUE " */*
QUEUE "/* STEPU2 EXEC PGM=IKJEFT01, DYNAMNBR=30, REGION=4M"
QUEUE "/* SYSEXEC DD DSN=YOUR. PROD. REXX. LIB, DISP=SHR"
QUEUE "/* PLSBYPAS DD DUMMY"
QUEUE "/* SYSTSPRT DD SYSOUT=*"
QUEUE "/* SYSPRINT DD SYSOUT=*"
QUEUE "/* SYSTSIN DD *"
QUEUE " EXECUTI L SEARCHD(YES)"
QUEUE " /* ESPYC STL "BMCCNTL ENTORNO MODO"
QUEUE " */*
QUEUE " /* STEPU3 EXEC PGM=EBGENER"
QUEUE " /* SYSSIN DD DUMMY"
QUEUE " /* SYSPRINT DD SYSOUT=*"
QUEUE " /* SYSTIN DD DATA, DLM='{', INTRDR)"
QUEUE " /* SYSTOUT DD DATA, DLM='{', INTRDR)"
QUEUE " /* J OB XCOM" JOB OSORIO, DB2, CLASS=F, MSGCLASS=0, TIME=1440, "
QUEUE " /* MSGLEVEL=(1, 1)"
QUEUE " /* STEPX1 EXEC PGM=XCOMJOB, PARM= 'ACBNAME=XCOMP, TYPE=EXECUTE' "
QUEUE " /* STEPLIB DD DSN=XCOM.LINKLIB, DISP=SHR"
QUEUE " /* SYSTCPD DD DSN=SYS1.TCPPARMS(TCPI PDATA), DISP=SHR"
QUEUE " /* SYSUDUMP DD SYSOUT=*"
QUEUE " /* SYSPRINT DD SYSOUT=*"
QUEUE " /* SYSIN01 DD *"
QUEUE " LU=LUDEVE"
QUEUE " USERID=USERFTP"
QUEUE " PASSWORD=USERFTP"
QUEUE " TYPE=SEND"
QUEUE " FILETYPE=FILE"
QUEUE " FILEOPT=CREATE"
QUEUE " LFILE="BMCREC"
QUEUE " FILE="BMCRECO"
QUEUE " NEWXFER"
QUEUE " LFILE="BMCCNTL"
QUEUE " FILE="BMCCNTLO"
QUEUE " */*"
QUEUE " /* STEPX2 EXEC PGM=XCOMJOB, PARM= 'ACBNAME=XCOMP, TYPE=EXECUTE' "
QUEUE " /* STEPLIB DD DSN=XCOM.LINKLIB, DISP=SHR"
QUEUE " /* SYSTCPD DD DSN=SYS1.TCPPARMS(TCPI PDATA), DISP=SHR"
QUEUE " /* SYSUDUMP DD SYSOUT=*"
QUEUE " /* SYSPRINT DD SYSOUT=*"
QUEUE " /* SYSTCPD DD DSN=SYS1.TCPPARMS(TCPI PDATA), DISP=SHR"
QUEUE " /* /*
QUEUE " /* STEPX3 EXEC PGM=IDCAMS"
```
/* SYSPRINT DD SYSOUT=* */

QUEUE "//SYSPRINT DD SYSOUT=*"
QUEUE "//SYSIN DD *
QUEUE " DELETE "BMCCNTL" PURGE"
QUEUE " DELETE "BMCREC" PURGE"
QUEUE " DELETE "DSNJOBLD" PURGE"
QUEUE "//"
QUEUE "$$"
QUEUE ""
ADDRESS TSO "EXECIO * DISKW JOB (FINIS"
ADDRESS TSO "FREE DDNAME(JOB)
"DELSTACK"
  Y = OUTTRAP(DELSUB.)
  "SUBMIT 'DSNJOB''"
  Y = OUTTRAP('OFF')
IF RC <> Ø THEN
  DO
    ADDRESS ISPEXEC "SETMSG MSG(DBCØ22)"
    GENJOBRC = 1
    RETURN
  END
MODO = 'R'
DB2COP = 1
CALL DELTMP
CALL DELJOB
RETURN
*/

GENJOBLOAD:
DSNJOBLD = USERID() || '.TMP' || '.JLD' || TIME('S')
ADDRESS TSO "ALLOC FILE(JOBLD) DATASET('"DSNJOBLD"')",
"NEW CAT REUSE UNIT(SYSDA)",
"LRECL(8Ø) BLKSIZE(2792Ø) RECFM(F B) SPACE(1,1) CYL"
IF RC <> Ø THEN
  DO
    ADDRESS ISPEXEC "SETMSG MSG(DBCØ21)"
    GENJOBRC = 1
    RETURN
  END
"NEWSTACK"
QUEUE " //JOBCOPY" JOB OSORIO, DB2, CLASS=F, MSGCLASS=Ø, TIME=144Ø,
QUEUE " // MSGLEVEL=(1,1)"
QUEUE " //ROUTE PRINT NODEPROD"
QUEUE " //STEPS1 EXEC PGM=IKJEFTØ1, DYNAMNBR=2Ø"
QUEUE " // SYSTSPRT DD SYSOUT=*"
QUEUE " // SYSPRINT DD SYSOUT=*"
QUEUE " // SYSPUNCH DD SYSOUT=*"
QUEUE " // SYSPUNCH DD DUMMY"
QUEUE " // SYSPUNCH DD DUMMY"
QUEUE " // SYSTAI DD *
QUEUE " DSN SYSTEM(SSID)"
QUEUE " RUN PROGRAM(DSNTIAUL) PLAN(DSNTIAUL) PARM('SQL')"
```
QUEUE " -START DB("DBNAMEO") SPACE("TSNAME") ACCESS(RW)"
DO J = 1 TO IX.Ø
  QUEUE " -START DB("DBNAMEO") SPACE("IX.J") ACCESS(RW)"
END
QUEUE " END"
QUEUE " //SYSIN DD *"
IF SQLWHERE = 'Y' THEN
  DO
    QUEUE " DELETE FROM OWNERD."TABLA"
    QUEUE " WHERE"
    DO I = 2 TO 8 WHILE SYSIN.I <> ''
      QUEUE " | | SYSIN.I"
    END
    ADDRESS TSO "FREE FILE(SYSIN)"
    ADDRESS TSO "FREE FILE(SYSRECØØ)"
    ADDRESS TSO "FREE FILE(SYSPRINT)"
    ADDRESS TSO "FREE FILE(SYSPUNCH)"
    CALL DELTMP
  END
IFE MODO = 'R' THEN
  DO
    IF PART > Ø THEN
      DO I = 1 TO PART
        QUEUE " ALTER TABLESPACE "DBNAMEO"."TSNAME" PART "I"
        QUEUE " PRIQTY "PRIQTY" SECQTY "PRIQTY%1Ø";
      END
    ELSE
      DO
        QUEUE " ALTER TABLESPACE "DBNAMEO"."TSNAME"
        QUEUE " PRIQTY "PRIQTY" SECQTY "PRIQTY%1Ø";
      END
      DO J = 1 TO IX.Ø
        QUEUE " ALTER INDEX OWNERD"."IX.J"
        QUEUE " PRIQTY "PRIQTYIX.J" SECQTY "PRIQTYIX.J%1Ø";
      END
    END
  END
END
QUEUE " /*
QUEUE " //STEPL2 EXEC PGM=DSNUTILB,PARM='SSID,,'"
QUEUE " //SYSTSPRT DD SYSOUT=*"
QUEUE " //UTPRINT DD SYSOUT=*"
QUEUE " //SYSUDUMP DD SYSOUT=*"
QUEUE " //SYSPRINT DD SYSOUT=*"
QUEUE " //SYSIN DD *"
QUEUE " * REPAIR LOG NO SET TABLESPACE "DBNAMEO"."TSNAME" NOCOPYPEND"
QUEUE " */*
QUEUE " //STEPL3 EXEC PGM=AMUUMAIN,"
QUEUE " // PARM='SSID,NEW,MSGLEVEL(1)',"
QUEUE " // REGION=ØM"
QUEUE " //STEPLIB DD DISP=SHR,DSN='BMC.DEVELOP.LOAD'"
QUEUE " /*PLSLIGNR DD DUMMY"
QUEUE "//DSSPRINT DD SYSOUT=*"
QUEUE "//SYSPRINT DD SYSOUT=*"
QUEUE "//SYSDUMP DD SYSOUT=*"
QUEUE "//UTPRINT DD SYSOUT=*"
QUEUE "//SYSDM DD SYSOUT=*"
QUEUE "//SYSIN DD DSN="BMCCNTLO",DISP=SHR"
QUEUE "//SYSREC DD DSN="BMCRECO",DISP=SHR"
QUEUE "//*
QUEUE "//STEPL4 EXEC PGM=AMUUMAIN,"
QUEUE " // PARM='SSID, "USERID()", "JOBCOPY", TERM, MSGLEVEL(1)', "
QUEUE " // REGION=ØM, COND=(4, GT)"
QUEUE "//STEPLIB DD DSN= 'BMC.DEVELOP.LOAD'
QUEUE "//SYSOUT DD SYSOUT=*"
QUEUE "//*
QUEUE "//STEPL5 EXEC PGM=IDCAMS"
QUEUE "//SYSIN DD SYSOUT=*"
QUEUE "//SYSIN DD *
QUEUE " DELETE "BMCCNTLO" PURGE"
QUEUE " DELETE "BMCRECO" PURGE"
QUEUE "//
QUEUE " ADDRESS TSO "EXECIO * DISKW JOBLD (FINIS"
ADDRESS TSO "FREE DDNAME(JOBLD)"
"DELSTACK"
RETURN
/*-----------------------------------------------*/
CHKJOB:
"ISPEXEC LIBDEF ISPLLIB DATASET ID('YOUR.PROD.LOAD.LIB')"
IF INPUT = 'T' THEN VUELTAS = 30
IF INPUT = 'I' THEN VUELTAS = 60
IF HPQTY = 'Y' THEN VUELTAS = 60
JOBOUT = ''
DO I=1 TO VUELTAS WHILE JOBOUT = ''
   SAY 'COPIANDO LA TABLA 'TABLA', JOB 'JOBCOPY'........' TIME()
   ADDRESS TSO "ESPERA'';
   IF RC <> 0 THEN EXIT
   X = OUTTRAP(JST.)
   INTERPRET "STATUS "JOBCOPY;
   X = OUTTRAP('OFF')
   JSTMAX = JST.Ø
   IF SUBSTR(JST.JSTMAX, 1, 9) = 'IKJ56192I' THEN
      JOBOUT = WORD(JST.JSTMAX, 3)
   ELSE NOP
END
RETURN
/*-----------------------------------------------*/
DISRESUL:
VUELTAS = 15
IF JOBOUT = '' THEN
   DO
MSG1 = 'THE COPY OF THE TABLE 'TABLA' CONTINUE'
MSG2 = 'WAIT THE RETURN OF THE JOB 'JOBCOPY
END
ELSE
DO
    JDSN = USERID()'.'JOBCOPY'.OUTLIST';
    ADDRESS TSO "OUTPUT "JOBCOPY" KEEP PRINT('"JDSN"')"
    IF RC <> Ø THEN
        DO
            MSG1 = 'NOT FOUND OUTLIST RC=' RC
            MSG2 = 'REVIEW JOB OUTPUT: 'JOBOUT
            RETURN
        END
    ADDRESS TSO "ALLOC FILE(JO) DATASET('"JDSN"') OLD REUSE"
    ADDRESS TSO "EXECIO * DISKR JO (STEM JDSN. FINIS"
    IF JDSN.Ø = Ø THEN
        DO
            MSG1 = 'NOT FOUND REGISTERS IN OUTLIST'
            MSG2 = 'REVIEW JOB OUTPUT: 'JOBOUT
            RETURN
        END
    DO I=1 TO JDSN.Ø WHILE SUBSTR(JDSN.I,31,6) <> 'STEPL3'
    END
    IF SUBSTR(JDSN.1,52,2) = 'ØØ' THEN
        DO
            MSG1 = 'THE COPY OF THE TABLE 'TABLA' FINISHED OK'
            DO I=I+1 TO JDSN.Ø WHILE WORD(JDSN.I,1) ¬= 'BMC51475I'
            END
            MSG2 = 'THE NUMBER OF THE ROWS COPIED IS: 'WORD(JDSN.I,4)
            FILAS = WORD(JDSN.I,4)
            CALL GRABALOG
        END
    ELSE
        DO
            MSG1 = 'COPY JOB 'JOBCOPY 'FINISHED NOT OK'
            MSG2 = 'REVIEW JOB OUTPUT: 'JOBOUT
        END
        W = OUTTRAP(DELOUT.)
        ADDRESS TSO "DELETE ('"JDSN"')"
        W = OUTTRAP('OFF')
    END
RETURN
/*------------------------------*/
GRABALOG:
ADDRESS TSO "ALLOC FILE(LOG) DATASET('YOUR.DB2COPIX.LOG') OLD REUSE"
IF RC = Ø THEN
    DO
        ADDRESS TSO "EXECIO * DISKR LOG (STEM LOG. FINIS"
        ULTIMO = LOG.Ø + 1
        IF RC = Ø THEN
ADDRESS TSO "EXEClO Ø DISKW LOG (OPEN FINIS"
END
ELSE
ADDRESS TSO "ALLOC FILE (LOG) DATASET (YOUR.DB2COPIX.LOG)" , "NEW CAT REUSE UNIT (SYSDA)" , "LRECL (80) BLKSIZ E (27920) RECFM (F B) SPACE (1, 1) CYL"
IF RC <> Ø THEN
  DO
    ADDRESS ISPEXEC "SET MSG MSG (DBC041)"
    ALLOCSYSRC = 1
    RETURN
  END
USERIDL = LEFT (USERID (), 6)
TABLAL = LEFT (TABLA, 8)
LOG.ULTIMO = 'C' || USERIDL || TABLAL || AUTORIZ || ENTORNO || INPUT,
  || SQLWHERE || MODO || DATE ('S') || TIME () || JOBOUT || FILAS
ADDRESS TSO "EXEClO * DISKW LOG (STEM LOG."
ADDRESS TSO "EXEClO Ø DISKW LOG (FINIS"
"FREE DDNAME (LOG)"
RETURN
/* --------------------------------------------------------------- */
DELTMP:
W = OUTTRAP (DELTMP.)
ADDRESS TSO "DELETE ('"DSNIN"')"
ADDRESS TSO "DELETE ('"DSNREC"')"
ADDRESS TSO "DELETE ('"DSNPRINT"')"
W = OUTTRAP ('OFF')
RETURN
/* --------------------------------------------------------------- */
DELJOB:
W = OUTTRAP (DELJOB.)
ADDRESS TSO "DELETE ('"DSNJ0B"')"
W = OUTTRAP ('OFF')
RETURN

YOUR.PROD.PANEL.LIB (DB2COPIP)

attr
% TYPE (TEXT) INTENS (HIGH) SKIP (ON)
+ TYPE (TEXT) INTENS (LOW) SKIP (ON)
  _ TYPE (INPUT) HILITE (&VAHILITE) CAPS (ON)
) BODY EXPAND (\)
+\-%COPY OF THE DB2 TABLES FROM PRODUCTION TO MAINTENANCE+\-%
% Enter the name of the table to copy
% Press Enter Key.
+  1. + Table _ TABLA + 
+
MEMBER YOUR.PROD.MESSAGE.LIB(DBC00)

DBC001 'FAILURE OF SYSIN ALLOC' ALARM=YES
'CHECK FOR OUT OF SPACE IN SYSDA/RACF PERMISSIONS'

DBC002 'FAILURE OF SYSREC ALLOC' ALARM=YES
'CHECK FOR OUT OF SPACE IN SYSDA/RACF PERMISSIONS'

DBC003 'FAILURE OF SYSPRINT ALLOC' ALARM=YES
'CHECK FOR OUT OF SPACE IN SYSDA/RACF PERMISSIONS'

MEMBER 'YOUR.PROD.MESSAGE.LIB(DBC001)'

DBC010 'TABLE NOT FOUND' ALARM=YES
'THE TABLE ENTERED NOT EXIST. PLEASE ENTER A VALID TABLE NAME'
MORE THAN ONE TABLE FOUND . ALARM=YES
' MORE THAN ONE TABLE FOUND IN CATALOG DB2. CALL TO SUPPORT'

' YOU HAVE ENTERED A VIEW' . ALARM=YES
' YOU ARE TYPED A VIEW, NOT A TABLE. PLEASE ENTER A TABLE NAME'

' YOU HAVE ENTERED AN ALIAS' . ALARM=YES
' YOU ARE TYPED A ALIAS, NOT A TABLE. PLEASE ENTER A TABLE NAME'

' DATABASE NAME MUST BE BVB% (ONLY APPLICATIONAL DBNAME). CUSTOMIZE IT'

' THE SQL CONDITION(WHERE) IS NOT VALID. COLUMN NOT FOUND'

JCL AND SOURCE TO ASSEMBLE ROUTINE ESPERA

// ESPERASS JOB (OSORIO), 'ASS', CLASS=F, MSGCLASS=X, MSGLEVEL=(1,1),
// NOTIFY=&SYSUID
// ASM EXEC PGM=ASMA90,
// PARM=('NOOBJECT,DECK,XREF(SHORT)')
// SYSLIB DD DISP=SHR, DSN=SYS1.MACLIB
// SYSPUNCH DD DISP=(NEW,PASS), DSN=&&OBJ,
// SPACE=(3040,(40,40),,ROUND),
// UNIT=SYSALLDA,
// DCB=(BLKSIZE=3040,LRECL=80,RECFM=FBS,BUFNO=1)
// SYSUT1 DD DSN=&&SYSUT1,
// SPACE=(1024,(120,120),,ROUND),
// UNIT=SYSALLDA,
// DCB=BUFNO=1
// SYSLIB DD *
// TITLE 'ROUTINE ESPERA'

ESPERA CSECT
SAVE (14,12)
LR R12, R15
USING ESPERA, R12
ST R13, SAVEAREA+4
LA R2, SAVEAREA
ST R2, 8(, R13)
LR R13, R2
EXTRACT TIOTADR,'S', FIELDS=TIOT
L R8, TIOTADR
MVC JOBNAME, O(R8) TO SELECT TSO USERS WHO
* CAN EXECUTE THIS ROUTINE

CLC JOBNAME, =C'USER1' ' PERMIT TSO USER *USER1*
BE SIGUE
CLC JOBNAME, =C'USER2' ' PERMIT TSO USER *USER2*
BE SIGUE
CLC JOBNAME, =C'USER3' ' PERMIT TSO USER *USER3*
BE SIGUE
L R13, 4(R13)
RETURN (14, 12), RC=12
SIGUE EQU *

STIMER WAIT, DINTVL=WAITTIME
RETURN EQU *
L R13, 4(R13)
RETURN (14, 12), RC=Ø
USING ESPERA, R12

SAVEAREA DS 18F
EXITSAVE DS 18F
TIOTADR DS F
WTOLST1 DC AL2(WTOLST1X-WTOLST1)
DC H'Ø'
JOBNAME DS CL8
WTOLST1X EQU *
WAITTIME DC C'ØØØØ2ØØØ' 20 SECONDS
DEC DS D
LTORG

RØ EQU Ø
R1 EQU 1
R2 EQU 2
R3 EQU 3
R4 EQU 4
R5 EQU 5
R6 EQU 6
R7 EQU 7
R8 EQU 8
R9 EQU 9
R10 EQU 10
R11 EQU 11
R12 EQU 12
R13 EQU 13
R14 EQU 14
R15 EQU 15
PRINT GEN
IKJTCB
END ESPERA

/ *
//SYSLIN DD DUMMY
//SYSPRINT DD SYSOUT=* 
/ *
//LKED EXEC PGM=IEWL,
// PARM=('LIST,MAP,AC=1,AMODE=ANY,RMODE=24'),
SET operators

This article looks at the SET operators that are available in DB2 UDB for LUW. I ran the following SQL on a Windows 2000 machine running DB2 V8 FP2.

There are three SET operators available – UNION, EXCEPT, and INTERSECT. I will look at each of these in turn.

For all our examples we will use two tables, hm01 and hm02, created and populated as follows:

```sql
create table hm01 (id int, name char(10))
create table hm02 (id int, name char(10))

insert into hm01 values(1,'AAA'),(2,'BBB'),(3,'CCC')
insert into hm02 values(1,'AAA'),(4,'DDD'),(5,'EEE')
```

The UNION operator lets you join the results of two SELECT statements, with duplicate rows being eliminated (hence requiring a sort). The UNION ALL operator does not eliminate duplicate rows (and hence does not require a sort).
So if we use the UNION operator and select everything from our two tables, you can see that the operator returns only unique rows (one of the duplicate ‘1 AAA’ rows has been eliminated) and the number of rows returned is 5.

```sql
>db2 select * from hm01 union select * from hm02

<table>
<thead>
<tr>
<th>ID</th>
<th>NAME</th>
</tr>
</thead>
<tbody>
<tr>
<td>1</td>
<td>AAA</td>
</tr>
<tr>
<td>2</td>
<td>BBB</td>
</tr>
<tr>
<td>3</td>
<td>CCC</td>
</tr>
<tr>
<td>4</td>
<td>DDD</td>
</tr>
<tr>
<td>5</td>
<td>EEE</td>
</tr>
</tbody>
</table>
```

5 record(s) selected.

If we now use the UNION ALL operator, we can see we have two occurrences of the ‘1 AAA’ row and the number of rows returned is 6.

```sql
>db2 select * from hm01 union all select * from hm02

<table>
<thead>
<tr>
<th>ID</th>
<th>NAME</th>
</tr>
</thead>
<tbody>
<tr>
<td>1</td>
<td>AAA</td>
</tr>
<tr>
<td>4</td>
<td>DDD</td>
</tr>
<tr>
<td>5</td>
<td>EEE</td>
</tr>
<tr>
<td>1</td>
<td>AAA</td>
</tr>
<tr>
<td>2</td>
<td>BBB</td>
</tr>
<tr>
<td>3</td>
<td>CCC</td>
</tr>
</tbody>
</table>
```

6 record(s) selected.

If you did a visual explain on both of the above statements using the Control Center, you can see that the UNION version does a SORT whereas the UNION ALL version doesn’t.

Now let’s look at the EXCEPT operator. The EXCEPT operator returns rows in the table to the left of the operator, which do not exist in the table to the right of the operator. The operator is used as shown in the example below.

```sql
>db2 select * from hm01 except select * from hm02

<table>
<thead>
<tr>
<th>ID</th>
<th>NAME</th>
</tr>
</thead>
</table>
```

You can see that the result returned is the two rows in table hm01, which are not present in table hm02.

Finally, let’s look at the INTERSECT operator. As the name suggests, and in line with normal mathematical set theory, the INTERSECT operator will return rows that exist in only both tables specified either side of the operator. This is shown in the example below.

```plaintext
>db2 select * from hm01 intersect select * from hm02

<table>
<thead>
<tr>
<th>ID</th>
<th>NAME</th>
</tr>
</thead>
<tbody>
<tr>
<td>1</td>
<td>AAA</td>
</tr>
</tbody>
</table>

1 record(s) selected.
```

The only row that is present in both table hm01 and hm02 is the ‘1 AAA’ row, and this is the row that is returned.

You can use the additional ALL parameter for both the EXCEPT and INTERSECT operators. The manual says that this works in the same way as with UNION and UNION ALL. However, there is no difference as far as I can see between EXCEPT and EXCEPT ALL and INTERSECT and INTERSECT ALL. If you specified EXCEPT ALL in the third example, you would still get only the two rows back, and if you specified INTERSECT ALL in the fourth example, you would still get only the one row returned.

---

C Leonard  
Freelance Consultant (UK)  © Xephon 2004

---

**A program to fix tablespaces/indexes with RESTRICTed access – part 2**

This month we conclude the code to change the availability of tablespaces and indexes to RW from any restricted status.

```sql
IF ALLOCSYSRC = 1 THEN RETURN;
```
SYSIN.1 = "SELECT ICDATE, ICTIME, DSNAME 
SYSIN.2 = " FROM SYSBM.SYSCOPY"
SYSIN.3 = " WHERE DBNAME = 'DATABASE' 
SYSIN.4 = " AND TSNAME = 'NAME' 
SYSIN.5 = " AND ICTYPE = 'F' 
SYSIN.6 = " AND ICBACKUP = ' ' 
SYSIN.7 = " AND DSNUM = 0 
SYSIN.8 = " AND ICDATE LIKE '0%' 
SYSIN.9 = " ORDER BY ICDATE DESC, ICTIME DESC;" 
"EXECIO * DISKW SYSIN (STEM SYSIN."
"EXECIO 0 DISKW SYSIN (FINIS"
 ADDRESS TSO
 "ALLOC FILE(SYSIN) DATASET('DSNIN') OLD REUSE"
 "ALLOC FILE(SYSREC00) DATASET('DSNREC') OLD REUSE"
 "ALLOC FILE(SYSPRINT) DATASET('DSNPRINT') OLD REUSE"
 "ALLOC FILE(SYSPUNCH) DUMMY"
 PUSH "END"
 PUSH "RUN PROGRAM(DSNTIAUL) PLAN(DSNTIAUL) PARMS('SQL')"
 ADDRESS TSO "DSN SYSTEM('SSID')"
 ADDRESS TSO "EXECIO * DISKR SYSREC00 (STEM REC00. FINIS"
 ADDRESS TSO "FREE FILE(SYSIN)"
 ADDRESS TSO "FREE FILE(SYSREC00)"
 ADDRESS TSO "FREE FILE(SYSPRINT)"
 ADDRESS TSO "FREE FILE(SYSPUNCH)"
 CALL DELTMP;
 DO
 "NEWSTACK"
 ICDSNANE = ''
 V = ''
 "ISPEXEC TBCREATE TICS NAMES(V FECHA HORA VERSION DSNAME)",
 "NOWRITE REPLACE"
 DO I = 1 TO REC00.0
   FECHA = SUBSTR(REC00.I,1,6)
   HORA = SUBSTR(REC00.I,7,6)
   VERSION= (I-1)*(-1)
   DSNAME = SUBSTR(REC00.I,13,44)
   "ISPEXEC TBADD TICS"
 END
 "ISPEXEC ADDPOP ROW(5) COLUMN(5)"
 RESREC2.EXIT = 'N'
 "ISPEXEC TBTOP TICS"
 "ISPEXEC TBDISPL TICS PANEL(DB2TREIC)"
 IF RC = 8 THEN
 DO
   RESREC2.EXIT = 'S'
   R = NRES + 1
 END
 ELSE
 DO
   ICDSNANE = STRIP(DSNAME)
CALL RESRECPJ;  
END

"DESTACK"
"ISPEXEC REMPOP"
"ISPEXEC TBEND TICS"
END

IF E = '2' THEN
  DO
    CALL RESRECPJ;
  END

IF E = '3' THEN
  DO
    CALL RESREPAJ;
  END

END

"DESTACK"
"ISPEXEC REMPOP"
CALL LIBDEFPANEL;
RETURN

/*----------------------------------------------------------------*/
RESRECPJ:
"NEWSTACK"

DSNJOB = USERID() || '.RES' || '.JOB' || TIME('S')
ADDRESS TSO "ALLOC FILE(JOB) DATASET(DSNJOB)"
  "NEW CAT REUSE UNIT(SYSDA)"
  "LRECL(80) BLKSIZE(27920) RECFM(F B) SPACE(1,1) CYL"
IF RC <> 0 THEN
  DO
    ADDRESS ISPEXEC "SETMSG MSG(DBCØ21)"
    RETURN
  END

JOBNAME = SUBSTR((USERID() || 'RC'),1,8)
QUEUE "J O B N A M E = " JOB (DB2), 'OSORIO', MSGCLASS=X,
QUEUE "CLASS=S, MSGLEVEL=(1,1), NOTIFY=&SYSUID"
QUEUE "STEP1 EXEC PGM=AFRMAIN, REGION=ØM,"
QUEUE "PARM=" "SSID" , NEW, MSGLEVEL(1), , RDB2STAT(RW) " 
QUEUE "SYSPRINT DD SYSOUT=" 
QUEUE "UTPRINT DD SYSOUT=" 
QUEUE "SYSIN DD *
QUEUE "RECOVER TABLESPACE "DATABASE"."NAME" D NUM ALL"
IF E = 1 THEN
  IF THEN
  QUEUE "TOCOPY "ICDSNAME; 
  QUEUE "REDEFINE YES"
  QUEUE "RECOVER INDEX (ALL)"
  QUEUE "TABLESPACE "DATABASE"."NAME"
  QUEUE "ANALYZE YES"
  QUEUE "REDEFINE YES"
  QUEUE ""
ADDRESS TSO "EXECIO * DISK W JOB (FINIS"
ADDRESS TSO "EXECIO * DISKR JOB (STEM) JOBRECQ. FINIS"
"ISPEXEC TBCREATE TJOBREC",
"NAMES(JOBREC)",
"NOWRITE REPLACE"
DO I = 1 TO JOBRECQ.Ø
   JOBREC = JOBRECQ.I
   "ISPEXEC TBADD TJOBREC"
END
"ISPEXEC ADDPOP ROW(7) COLUMN(7)"
RESREC3_EXIT = 'N'
ADDRESS TSO "FREE DDNAME(JOB)"
"ISPEXEC TBTOP TJOBREC"
"ISPEXEC TBDISPL TJOBREC PANEL(DB2TRECJ)"
IF RC = 8 THEN
   DO
      RESREC3_EXIT = 'S'
      R = NRES + 1
   END
ELSE
   DO
      Y = OUTTRAP(DELSUB.)
      "SUBMIT 'DSNJOB'"
      Y = OUTTRAP('OFF')
      IF RC <> Ø THEN
         DO
            ADDRESS ISPEXEC "SETMSG MSG(DBC022)"
            RETURN
         END
      CALL GRABALOG
   END
"DELSTACK"
"ISPEXEC REMPOP"
"ISPEXEC TBEND TJOBREC"
CALL DELJOB;
RETURN

RESREPAJ:
"NEWSTACK"

DSNJOB = USERID() || '.RES' || '.JOB' || TIME('S')
ADDRESS TSO "ALLOC FILE(JOB) DATASET('DSNJOB') "
   "NEW CAT REUSE UNIT(SYSDA)"
   "LRECL(80) BLKSIZE(27920) RECFM(F B) SPACE(1,1) CYL"
IF RC <> Ø THEN
   DO
      ADDRESS ISPEXEC "SETMSG MSG(DBC021)"
      RETURN
   END
JOBNAME = SUBSTR((USERID() || 'RP'),1,8)
QUEUE "/" JOBNAME "JOB (DB2), 'OSORIO', MSGCLASS=X,"
QUEUE "// CLASS=S, MSGLEVEL=(1,1), REGION=ØM, NOTIFY=&SYSUID"
QUEUE "// REPAIR EXEC PGM=DSNUTILB, PARM=‘SSID’, , COND=EVEN"
QUEUE "// SYSPRINT DD SYSOUT=*"
QUEUE "// UTPRINT DD SYSOUT=*"
QUEUE "// SYSSR ERR DD SYSOUT=*"
QUEUE "// SYSIN DD *"
IF TYPE = 'TABLESPACE' THEN
QUEUE " REPAIR LOG NO SET TABLESPACE "DATABASE"."NAME" NORCVRPEND";
IF TYPE = 'INDEX' THEN
QUEUE " REPAIR LOG NO SET INDEX "IXCREATOR"."NAME" NORCVRPEND";
QUEUE ""
ADDRESS TSO "EXECIO * DISKW JOB (FINIS"
ADDRESS TSO "EXECIO * DISKR JOB (STEM JOBREPQ. FINIS"
"ISPEXEC TBCREATE TJOBREP",
"NAMES(JOBREP)",
"NOWRITE REPLACE"
DO I = 1 TO JOBREPQ.Ø
   JOBREP = JOBREPQ.I
   "ISPEXEC TBADD TJOBREP"
END
"ISPEXEC ADDPOP ROW(7) COLUMN(7)"
RESREC4_EXIT = 'N'
ADDRESS TSO "FREE DDNAME(JOB)"
"ISPEXEC TBTOP TJOBREP"
"ISPEXEC TBDISPL TJOBREP PANEL(DB2TREPJ)"
IF RC = 8 THEN
   DO
      RESREC4_EXIT = 'S'
      R = NRES + 1
   END
ELSE
   DO
      Y = OUTTRAP(DELSUB.)
      "SUBMIT ‘DSNJ OB’ ‘ ‘‘
      Y = OUTTRAP('OFF')
      IF RC <> Ø THEN
         DO
            ADDRESS ISPEXEC "SETMSG MSG(DBC022)"
            RETURN
         END
      CALL GRABALOG
   END
"DELSTACK"
"ISPEXEC REMPOP"
"ISPEXEC TBEND TJOBREP"
CALL DELJOB;
RETURN
="/*******************************************************************/
TBDELROWS:
"ISPEXEC TBSTATS TRES ROWCURR(TRESROWS)"

"ISPEXEC TBTOP TRES"
DO I = 1 TO TRESROWS
"ISPEXEC TBSKIP TRES"
"ISPEXEC TBDELETE TRES"
END
RETURN
/*----------------------------------------*/
GRABALOG:
ADDRESS TSO "ALLOC FILE(LOG) DSNAME('"LOGPREFIX".DB2RES.LOG') OLD REUSE"
IF RC = Ø THEN
DO
ADDRESS TSO "EXECIO * DISKR LOG (STEM LOG. FINIS"
IF RC = Ø THEN ULTIMO = LOG.Ø + 1
END
ELSE
DO
ADDRESS TSO "ALLOC FILE(LOG) DSNAME('"LOGPREFIX".DB2RES.LOG')",
"NEW CAT REUSE UNIT(SYSDA)"
,"LRECL(80) BLKSIZE(27920) RECFM(F B) SPACE(1,1) CYL"
IF RC <> Ø THEN
DO
ADDRESS ISPEXEC "SETMSG MSG(DBCØ41)"
RETURN
END
USERIDL = LEFT(USERID(),6)
SSID = LEFT(SSID,4)
SELECT
WHEN DISRES_ZERO = 1 THEN
LOG.ULTIMO = 'D' || USERIDL || SSID || LEFT(' ',18),
|| LEFT(' ',8) || LEFT(' ',8),
|| LEFT(' ',10) || LEFT(' ',4),
|| DATE('S') || TIME()
WHEN DISRES_ADD = 1 THEN
LOG.ULTIMO = 'D' || USERIDL || SSID || LEFT(STATUS,18),
|| LEFT(NAME,8) || LEFT(DATABASE,8),
|| LEFT(TYPE,10) || LEFT(PART,4),
|| DATE('S') || TIME()
OTHERWISE
LOG.ULTIMO = 'R' || USERIDL || SSID || LEFT(STATUS,18),
|| LEFT(NAME,8) || LEFT(DATABASE,8),
|| LEFT(TYPE,10) || LEFT(PART,4),
|| DATE('S') || TIME()
END
ADDRESS TSO "EXECIO * DISKW LOG (STEM LOG."
ADDRESS TSO "EXECIO Ø DISKW LOG (FINIS"
"FREE DDNAME(LOG)"
DISRES_ZERO = Ø
DISRES_ADD = Ø
RETURN
ALLOCYS:
ALLOCYSRC = 0
DSNI N = USER ID() || '.RES' || '.IN' || TIME('S')
DSNR EC = USER ID() || '.RES' || '.REC' || TIME('S')
DSNP RINT = USER ID() || '.RES' || '.PRT' || TIME('S')
ADDRESS TSO "ALLOC FILE(SYSIN) DATASET('"DSNI N"') " ,
"NEW CAT REUSE UNIT(SYSDA)" ,
"LRECL(80) BLKSIZE(27920) RECFM(F B) SPACE(1,1) CYL"
IF RC <> 0 THEN
DO
ADDRESS ISPSEXEC "SETMSG MSG(DBC001)"
ALLOCYSRC = 1
RETURN
END
ADDRESS TSO "ALLOC FILE(SYSREC00) DATASET('"DSNREC"') " ,
"NEW CAT REUSE UNIT(SYSDA)" ,
"LRECL(100) BLKSIZE(27900) RECFM(F B) SPACE(1,1) CYL"
IF RC <> 0 THEN
DO
ADDRESS ISPSEXEC "SETMSG MSG(DBC002)"
ALLOCYSRC = 1
RETURN
END
ADDRESS TSO "ALLOC FILE(SYSPRINT) DATASET('"DSNPRINT"') " ,
"NEW CAT REUSE UNIT(SYSDA)" ,
"LRECL(133) BLKSIZE(27930) RECFM(F B) SPACE(1,1) CYL"
IF RC <> 0 THEN
DO
ADDRESS ISPSEXEC "SETMSG MSG(DBC003)"
ALLOCYSRC = 1
RETURN
END
RETURN
LIBDEFWINDJ:
SELECT
WHEN SSID = 'DB2P' THEN
"ISPSEXEC LIBDEF ISPPLIB DATASET ID('YOUR.PROD.WINDOWSJ.USER')"
WHEN SSID = 'DB2D' THEN
"ISPSEXEC LIBDEF ISPPLIB DATASET ID('YOUR.DESA.WINDOWSJ.USER')"
WHEN SSID = 'DB2T' THEN
"ISPSEXEC LIBDEF ISPPLIB DATASET ID('YOUR.TEST.WINDOWSJ.USER')"
OTHERWISE
"ISPSEXEC LIBDEF ISPPLIB DATASET ID('YOUR.PROD.WINDOWSJ.USER')"
END
RETURN
*/
LIBDEFPANEL:
SELECT
WHEN SSID = 'DB2P' THEN
"ISPEXEC LIBDEF ISPPLIB DATASET ID('YOUR.PROD.PANELLIB.USER')"
WHEN SSID = 'DB2D' THEN
"ISPEXEC LIBDEF ISPPLIB DATASET ID('YOUR.DESA.PANELLIB.USER')"
WHEN SSID = 'DB2T' THEN
"ISPEXEC LIBDEF ISPPLIB DATASET ID('YOUR.TEST.PANELLIB.USER')"
OTHERWISE
"ISPEXEC LIBDEF ISPPLIB DATASET ID('YOUR.PROD.PANELLIB.USER')"
END
RETURN
/*----------------------------------------------------------------*/
DELTMP:
W = OUTTRAP(DELTMP.)
ADDRESS TSO "DELETE ('"DSNIN"')"
ADDRESS TSO "DELETE ('"DSNREC"')"
ADDRESS TSO "DELETE ('"DSNPRINT"')"
W = OUTTRAP('OFF')
RETURN
/*----------------------------------------------------------------*/
DELJOB:
W = OUTTRAP(DELJOB.)
ADDRESS TSO "DELETE ('"DSNJOB"')"
W = OUTTRAP('OFF')
RETURN
/*---------------------------------------------------------------*/
DISUTI BMC:
STATUSRBMC = '';
IF TYPE = 'TABLESPACE' THEN KIND = 'TP';
IF TYPE = 'INDEX' THEN KIND = 'IP';
DO I = 1 TO 9
  SYSIN.I = ''
END
CALL ALLOCSYS;
IF ALLOCSYSRC = 1 THEN RETURN;
SYSIN.1 = "SELECT UTILID,STATUS "
SYSIN.2 = "FROM BMCUTIL.CMN_BMCUTIL"
SYSIN.3 = "WHERE UTILID = ( SELECT DISTINCT UTILID "
SYSIN.4 = " FROM BMCUTIL.CMN_BMCSYNC"
SYSIN.5 = " WHERE NAME1 = 'DATABASE' "
SYSIN.6 = " AND NAME2 = 'NAME' "
SYSIN.7 = " AND KIND = 'KIND' )";
"EXECIO * DISKW SYSIN (STEM SYSIN.)"
"EXECIO 0 DISKW SYSIN (FINIS"
ADDRESS TSO "ALLOC FILE(SYSIN)" "DATASET('"DSNIN"') OLD REUSE"
ADDRESS TSO "ALLOC FILE(SYSRECØØ)" "DATASET('"DSNREC"') OLD REUSE"
ADDRESS TSO "ALLOC FILE(SYSPRINT)" "DATASET('"DSNPRINT"') OLD REUSE"
ADDRESS TSO "ALLOC FILE(SYSPPUNCH) DUMMY"
PUSH "END"
PUSH "RUN PROGRAM(DSNTIAUL) PLAN(DSNTIAUL) PARMS('SQL')"
ADDRESS TSO "DSN SYSTEM("SSID")"
ADDRESS TSO "EXECIO * DISKR SYSREC00 (STEM BMUCUTI. FINIS"
ADDRESS TSO "FREE FILE(SYSIN)"
ADDRESS TSO "FREE FILE(SYSREC00)"
ADDRESS TSO "FREE FILE(SYSPRINT)"
ADDRESS TSO "FREE FILE(SYSPPUNCH)"
CALL DELTMP
IF BMUCUTI.Ø = Ø THEN NOP               /* NO HAY UTILITARIOS BMC */
ELSE
   DO
      UTILIDR = SUBSTR(BMCUTI.1,1,16)
      STATUSR = SUBSTR(BMCUTI.1,17,1)
      IF STATUSR = 'X' THEN
      /* BMUCUTI UTILITARIO EN X */
         IF STATUSR = 'A' THEN
            /* BMUCUTI UTILITARIO EN A */
            IF STATUSR = 'P' THEN
               /* BMUCUTI UTILITARIO EN P */
               IF STATUSR = 'I' THEN
                  /* BMUCUTI UTILITARIO EN I */
                  DO
                     STATUSRBMC = 'X'
                     ADDRESS ISPEXEC "SETMSG MSG(DBC308)"
                  END
               ELSE CALL TERUTIBMC
               END
         ELSE CALL TERUTIBMC
      END
   END
RETURN
------------------------------------------------------------------
TERUTIBMC:
TERUTIBMCOK = Ø
DSNUT = USERID() || '.RES' || '.IN' || TIME('S')
ADDRESS TSO "ALLOC FILE(ZTEMPN) DATASET('"DSNUT"')",
   "NEW CAT REUSE UNIT(SYSDA)",
   "LRECL(80) BLKSIZE(27920) RECFM(F B) SPACE(1,1) CYL"
PLANNM = 'ABUD2300'
OPTNAME = ''
QUEUE '+TERM UTIL('UTILIDR')'
QUEUE 'END'
BMUCDSN = 'ABUDSN /SSID'/PLANNM'/OPTNAME'/

X = OUTTRAP("OUTPUT.",',',",NOCONCAT")
BMUCDSN
IF (RC = Ø) THEN TERUTIBMCOK = 1
ELSE
   ADDRESS ISPEXEC "SETMSG MSG(DBC206)"; /* FALLO TERM UTIL BMC*/
   "EXECIO" OUTPUT.Ø "DISKW" ZTEMPN "(OPEN STEM OUTPUT.)"
IF RC > Ø THEN SAY 'WRITE RC = ' RC
   "EXECIO" Ø "DISKW" ZTEMPN "(STEM OUTPUT FINIS)"
IF RC > Ø THEN SAY 'CLOSE RC = ' RC
Y = OUTTRAP(OFF)
IF TERUTIBMCOK = 1 THEN
   DO
      "FREE DDNAME(ZTEMPN)"

W = OUTTRAP('DELTZT."
ADDRESS TSO "DELETE ("DSNZT")"
W = OUTTRAP('OFF')
END
RETURN
/*-------------------------------------------------------------------------*/
TERUTI BMSTOP:

DISUTI:

W = OUTTRAP('UTI.')
_QUEUE "- DS N U T I L I T Y(*)"
_QUEUE "END"
"DSN SYSTEM("SSID")"
W = OUTTRAP('OFF')
IF RC > Ø THEN
  DO
    ADDRESS ISPEXEC "SETMSG MSG(DBC2ØØ)"        /* NO HAY DB2 */
    RETURN
  END
IF SUBSTR(UTI.1,1,8) = 'DSNU112I' THEN RETURN
I = 1;
DO I = I WHILE I < UTI.Ø
  IF (SUBSTR(UTI.I,1,8) = 'DSNU1Ø5I' |,         /* UTILITY ACTIVE */
     SUBSTR(UTI.I,1,8) = 'DSNU1ØØI' ) THEN     /* UTILITY STOPPED */
    DO J = 1 TO 7
      SELECT
        WHEN J = 1  THEN UTIUSERI = WORD(UTI.I,7)
        WHEN J = 2  THEN NOP
        WHEN J = 3  THEN UTILIDI  = WORD(UTI.I,3)
        WHEN J = 4  THEN NOP
        WHEN J = 5  THEN UTILITYI = WORD(UTI.I,3)
        WHEN J = 6  THEN PHASEI   = WORD(UTI.I,3)
        WHEN J = 7  THEN STATUSRIBM = WORD(UTI.I,3)
      END
      IF STATUSRIBM = 'STOPPED' THEN CALL TERUTI;
      IF J < 7 THEN I = I + 1;
    END
  END
RETURN
/*-------------------------------------------------------------------------*/
TERUTI:

W = OUTTRAP('TER.'
_QUEUE "- T E R M U T I L I T Y("UTILIDI")"
_QUEUE "END"
"DSN SYSTEM("SSID")"
W = OUTTRAP('OFF')
RETURN
/*-------------------------------------------------------------------------*/
TRAEIXCREATOR:

DO I = 1 TO 9
  SYSIN.I = ''
CALL ALLOCSYS
IF ALLOCSYSRC = 1 THEN RETURN;
SYSIN.1 = "SELECT CREATOR FROM SYSIBM.SYINDEXES"
SYSIN.2 = "WHERE INDEXSPACE = '"NAME"' AND DBNAME = '"DATABASE"';"
"EXECIO DISKW SYSIN (STEM SYSIN.)"
"EXECIO DISKW SYSIN (FINIS"
ADDRESS TSO "ALLOC FILE(SYSIN)" "DATASET('DSNIN')" OLD REUSE"
ADDRESS TSO "ALLOC FILE(SYSRECOO)" "DATASET('DSNREC')" OLD REUSE"
ADDRESS TSO "ALLOC FILE(SYSPRINT)" "DATASET('DSNPRINT')" OLD REUSE"
ADDRESS TSO "ALLOC FILE(SYSYPUNCH) DUMMY"
PUSH "END" PUSH "RUN PROGRAM(DSNTIAUL) PLAN(DSNTIAUL) PARMS('SQL')"
ADDRESS TSO "DSN SYSTEM('SSID')"
ADDRESS TSO "EXECIO DISKR SYSRECOO (STEM IX. FINIS"
ADDRESS TSO "FREE FILE(SYSIN)"
ADDRESS TSO "FREE FILE(SYSRECOO)"
ADDRESS TSO "FREE FILE(SYSPRINT)"
ADDRESS TSO "FREE FILE(SYSYPUNCH)"
CALL DELTMP
IXCREATOR = STRIP(SUBSTR(IX.1,1,8));
RETURN
/*@ -------------------- FIN END -- carlos-osorio@excite.com */

YOUR.PANELI(DB2RESSP)

)ATTR
% TYPE(TEXT) INTENS(HIGH) SKIP(ON)
  color(turquoise)
+ TYPE(TEXT) INTENS(LOW) SKIP(ON)
- TYPE(INPUT) INTENS(HIGH) CAPS(ON) HILITE(USCORE)
! TYPE(OUTPUT) INTENS(LOW) CAPS(ON) JUST(LEFT) HILITE(REVERSE)
  color(turquoise)
@ TYPE(TEXT) INTENS(HIGH) CAPS(ON) HILITE(REVERSE)
  color(turquoise)
)BODY EXPAND(\\)
@ SUBSYSTEM(S) DB2
@ %COMMAND ==>_ZCMD
%
% Opc Subsystem DB2
+ % - ---------------
)MODEL
+ _O+ !SDB2+
)INIT
  .help = db2ressh
  .cursor = 0
  &SCRO = PAGE
)REINIT

PROC
IF (.PFKEY = 'PFØ1')
    &pfkeyin = 's'
IF (&ZTDSELS > 0000)
    VER(&O, LIST, S)
END

YOUR.PANELI(DB2RESSH)

)attr
% TYPE(TEXT) INTENS(HIGH) SKIP(ON)
    color(turquoise)
+ TYPE(TEXT) INTENS(LOW) SKIP(ON)
- TYPE(INPUT) INTENS(HIGH) CAPS(ON) HILITE(USCORE)
! TYPE(OUTPUT) INTENS(LOW) CAPS(ON) JUST(LEFT) HILITE(REVERSE)
    color(turquoise)
@ TYPE(TEXT) INTENS(HIGH) CAPS(ON) HILITE(REVERSE)
    color(turquoise)
)BODY EXPAND(\)
@                            SUBSYSTEM(S) DB2
@
% Opc (OPTIONS) valids :+
+
+ %S+       - To select the subsystem DB2.
+
% Description of the columns: +
+
+ %Subsystem + Name/Identification of subsystem DB2
+                defined in MVS - OS/39Ø (SYS1.PARMLIB).
+
)END

YOUR.XXXX.PANELLIB.USER(DB2RESP)

)ATTR
% TYPE(TEXT) INTENS(HIGH) SKIP(ON)
    color(turquoise)
+ TYPE(TEXT) INTENS(LOW) SKIP(ON)
- TYPE(INPUT) INTENS(HIGH) CAPS(ON) HILITE(USCORE)
! TYPE(OUTPUT) INTENS(LOW) CAPS(ON) JUST(LEFT) HILITE(REVERSE)
    color(turquoise)
@ TYPE(TEXT) INTENS(HIGH) CAPS(ON) HILITE(REVERSE)
    color(turquoise)
)BODY EXPAND(\)
@                            TABLESPACE(S) / INDEX(ES) IN RESTRICT
@
### Opc  Status    Name    Database    Part    Type
+ %--  ------------------  --------  --------  ----  ----------

### INIT
```
help = db2resh
cursor = o
&SCRO = CSR
```

### PROC
```
if (.PFKEY = 'PF01')
  &pfkeyin = 's'
IF (&ZTDSELS > 0000)
  VER(&O, LIST, R)
END
```

### YOUR.XXXX.PANELLIB.USER(DB2RESH)

```
)attr
% TYPE(TEXT)  INTENS(HIGH)  SKIP(ON)  color(turquoise)
! TYPE(TEXT)  INTENS(HIGH)  SKIP(ON)  color(red)
+ TYPE(TEXT)  INTENS(LOW)   SKIP(ON)
_ TYPE(INPUT)  INTENS(HIGH)  CAPS(ON)  HILITE(USCORE)
@ TYPE(TEXT)  INTENS(HIGH)  CAPS(ON)  HILITE(REVERSE)
color(turquoise)
)BODY EXPAND(\)
```

**TABLESPACE(S)/INDEX(ES) IN RESTRICT**

```
@ @
% Opc (OPTIONS) valids :+
+
% R+  - Fix (repair). Unique option.
+ Generate commands and/or jobs for fix TS/IX in RESTRICT.
+
!  - Always begin for Fix (repair) the TS and then the IXs.
+
% Description of the columns :+
+
% Status+ - Status of +%TS/IX (Tablespace/Index).+
May be a combination of:

% RW, RO, STOP, STOP, STOP, UT, UTR, UTR, ACHKP, AUXW, CHK, COPY,
% GRECP, I COPY, LPL, LSTOP, PSRBD, RBDP, RBDP*, RECP, REORP, REST, REST.
+
% Name+ - Name of TS/IX in RESTRICT.
%Database+ Database of TS/IX en RESTRICT.
+ %Part+ Partition of TS/IX en RESTRICT.
+ %Type+ Type of object DB2. May be %TS/IX (Tablespace/Index).+

YOUR.XXXX.WINDOWSJ.USER(DB2TCOPJ)
LRECL 50 BLKSIZE 5000.

)ATTR
% TYPE(TEXT) INTENS(LOW) SKIP(ON)
   COLOR(TURQUOISE)
@ TYPE(TEXT) INTENS(HIGH) CAPS(ON)
   HILITE(REVERSE) COLOR(TURQUOISE)
$ TYPE(TEXT) INTENS(HIGH) CAPS(ON)
   HILITE(REVERSE) COLOR(BLUE)
# TYPE(OUTPUT) INTENS(LOW) CAPS(ON)
   HILITE(REVERSE) COLOR(WHITE)
! TYPE(OUTPUT) INTENS(LOW) CAPS(ON)
   HILITE(REVERSE) COLOR(TURQUOISE)
)BODY WINDOW(50,16)
%COMMAND ===>_ZCMD
$ JOB¿JOBNAME $SUBMITTED FOR RESET COPYPENDING
$-------------------------------------------------

MODEL
!JOBkop
 INIT
)PROC
)END

YOUR.XXXX.WINDOWSJ.USER(DB2TCHKJ)
LRECL 50 BLKSIZE 5000.

)ATTR
% TYPE(TEXT) INTENS(LOW) SKIP(ON)
   COLOR(TURQUOISE)
@ TYPE(TEXT) INTENS(HIGH) CAPS(ON)
   HILITE(REVERSE) COLOR(Blue)
$ TYPE(TEXT) INTENS(HIGH) CAPS(ON)
   HILITE(REVERSE) COLOR(White)
# TYPE(OUTPUT) INTENS(LOW) CAPS(ON)
   HILITE(REVERSE) COLOR(TURQUOISE)
YOUR.XXXX.WINDOWSJ.USER(DB2TRBDJ)
LRECL 50 BLKSIZE 5000.

)ATTR
% TYPE(TEXT)  INTENS(LOW) SKIP(ON)
  COLOR(TURQUOISE)
@ TYPE(TEXT)  INTENS(HIGH) CAPS(ON)
  HILITE(REVERSE) COLOR(BLUE)
$ TYPE(TEXT)  INTENS(HIGH) CAPS(ON)
  HILITE(REVERSE) COLOR(BLUE)
# TYPE(OUTPUT) INTENS(LOW) CAPS(ON)
  HILITE(REVERSE) COLOR(WHITE)
! TYPE(OUTPUT) INTENS(LOW) CAPS(ON)
  HILITE(REVERSE) COLOR(TURQUOISE)
)BODY WINDOW(50,16)
%COMMAND ===> ZCMD
$ JOB#JOBNAME $SUBMITED FOR RESET CHECKPENDING
$-----------------------------
)MODEL
!JOBCHK
)INIT
)PROC
)END

YOUR.XXXX.WINDOWSJ.USER(DB2TRECO)
LRECL 50 BLKSIZE 5000.

)ATTR
% TYPE(TEXT)  INTENS(LOW) SKIP(ON)
  COLOR(TURQUOISE)
@ TYPE(TEXT)  INTENS(HIGH) CAPS(ON)
  HILITE(REVERSE) COLOR(BLUE)
$ TYPE(TEXT)  INTENS(HIGH) CAPS(ON)
  HILITE(REVERSE) COLOR(TURQUOISE)
# TYPE(OUTPUT) INTENS(LOW) CAPS(ON)
  HILITE(REVERSE) COLOR(TURQUOISE)
$ TYPE(TEXT)  INTENS(HIGH) CAPS(ON)
  HILITE(REVERSE) COLOR(BLUE)
! TYPE(OUTPUT) INTENS(HIGH) CAPS(ON)
  HILITE(REVERSE) COLOR(WHITE)
- TYPE(INPUT) INTENS(HIGH) CAPS(ON)
) BODY WINDOW(37,08) AS IS
$ TYPE OF RECOVER : !DATABASE!NAME
@------------------------------------
% _E%1. TO FULL IMAGECOPY
%
% 2. TO ACTUAL POINT
%
% 3. REPAIR
) INIT
  CURSOR = E
  HELP  = DB2RECOH
) PROC
VER(&E, NONBLANK, LIST, 1, 2, 3)
) END

YOUR.XXXX.WINDOWSJ.USER(DB2TREIC)

LRECL 50 BLKSIZE 5000.

) ATTR
$ TYPE(TEXT) INTENS(HIGH) CAPS(ON)
  HILITE(REVERSE) COLOR(BLUE)
! TYPE(OUTPUT) INTENS(HIGH) CAPS(ON)
  HILITE(REVERSE) COLOR(WHITE)
% TYPE(TEXT) INTENS(HIGH) SKIP(ON)
+ TYPE(TEXT) INTENS(LOW) SKIP(ON)
- TYPE(INPUT) INTENS(HIGH) CAPS(ON)
@ TYPE(OUTPUT) INTENS(LOW) CAPS(ON) JUST(RIGHT)
  HILITE(REVERSE) COLOR(TURQUOISE)
? TYPE(OUTPUT) INTENS(LOW) CAPS(ON) JUST(LEFT)
  HILITE(REVERSE) COLOR(TURQUOISE)
) BODY WINDOW(44,18) AS IS
+
$ IMAGECOPY OF TABLESPACE!DATABASE!NAME
$-------------------------------------------
%COMMAND ===>_ZCMD       +%SCROLL ===>_SCRO+
%
% OPC    DATE    HOUR   VERSION
%--   ------  ------  -------
) MODEL
+  _V+  ?FECHA +?HORA  +@VERSION+
) INIT
  &FECHA  = '&ZDAY..&ZMONTH..&ZYEAR'
  &HORA  = &ZTIME
  &SCRO  = PAGE
PROC

IF (&ZTDSELS > 0000)
    VER(&V, LIST, R)
) END

YOUR.XXXX.WINDOWSJ.USER(DB2TRECJ)
LRECL 50 BLKSIZE 5000.

)ATTR
% TYPE(TEXT)   INTENS(LOW)  SKIP(ON)
    COLOR(TURQUOISE)
@ TYPE(TEXT)   INTENS(HIGH)  CAPS(ON)
    HILITE(REVERSE) COLOR(BLUE)
$ TYPE(TEXT)   INTENS(HIGH)  CAPS(ON)
    HILITE(REVERSE) COLOR(BLUE)
# TYPE(OUTPUT) INTENS(LOW)  CAPS(ON)
    HILITE(REVERSE) COLOR(WHITE)
! TYPE(OUTPUT) INTENS(LOW)  CAPS(ON)
    HILITE(REVERSE) COLOR(TURQUOISE)
)BODY WINDOW(50,16) ASIS

$ JOB#JOBNAME $SUBMITTED FOR RESET RECOVERYPENDING
$-------------------------------------------------

)MODEL
!JOBREC
)PROC
)INIT
)END

YOUR.XXXX.WINDOWSJ.USER(DB2TREPJ)
LRECL 50 BLKSIZE 5000.

)ATTR
% TYPE(TEXT)   INTENS(LOW)  SKIP(ON)
    COLOR(TURQUOISE)
@ TYPE(TEXT)   INTENS(HIGH)  CAPS(ON)
    HILITE(REVERSE) COLOR(BLUE)
$ TYPE(TEXT)   INTENS(HIGH)  CAPS(ON)
    HILITE(REVERSE) COLOR(BLUE)
# TYPE(OUTPUT) INTENS(LOW)  CAPS(ON)
    HILITE(REVERSE) COLOR(WHITE)
! TYPE(OUTPUT) INTENS(LOW)  CAPS(ON)
    HILITE(REVERSE) COLOR(TURQUOISE)
)BODY WINDOW(50,16) ASIS
$ JOB#JOBNAME $SUBMITTED FOR REPAIR RECOVERYPENDING

YOUR.XXXX.MSGLIB(DBC00)
DBC001 'FAILURE OF SYSIN ALLOC' . ALARM=YES
'CHECK FOR OUT OF SPACE IN SYSDA/RACF PERMISSIONS'
DBC002 'FAILURE OF SYSREC ALLOC' . ALARM=YES
'CHECK FOR OUT OF SPACE IN SYSDA/RACF PERMISSIONS'
DBC003 'FAILURE OF SYSPRINT ALLOC' . ALARM=YES
'CHECK FOR OUT OF SPACE IN SYSDA/RACF PERMISSIONS'

YOUR.XXXX.MSGLIB(DBC02)
DBC021 'FAILURE OF JOB ALLOC' . ALARM=YES
'CHECK FOR OUT OF SPACE IN SYSDA/RACF PERMISSIONS'
DBC022 'FAILURE WITH JOB SUBMIT' . ALARM=YES
'FAILURE WITH JOB SUBMIT FROM A REXX . CALL TO SUPPORT'

YOUR.XXXX.MSGLIB(DBC04)
DBC041 'REXX LOG NOT ACCESSIBLE' . ALARM=YES
'FAILURE IN ACCESS/DEFINE THE REXX LOG.'

YOUR.XXXX.MSGLIB(DBC20)
DBC200 'CONNECTION DB2 FAULT' . ALARM=YES . WINDOW=LNORESP
'THE CONNECTION DB2 IS NOT AVAILABLE. STARTUP THE DB2 SUBSYSTEM'
DBC206 'TERM UTIL BMC FAULT' . ALARM=YES . WINDOW=LNORESP
'FAILURE IN COMMAND TERM UTIL BMC (NOT IBM). CALL TO SUPPORT'

YOUR.XXXX.MSGLIB(DBC30)
DBC300 'CONNECTION DB2 FAULT' . ALARM=YES . WINDOW=LNORESP
'THE CONNECTION DB2 IS NOT AVAILABLE. STARTUP THE DB2 SUBSYSTEM'
DBC301 'NOT FOUND TS/IX RESTRICT' . ALARM=YES . WINDOW=LNORESP
'NOT FOUND TABLESPACES AND/OR INDEXS IN STATUS RESTRICT'
DBC302 'DISPLAY RESTRICT FAULT' . ALARM=YES . WINDOW=LNORESP
'PLAN BCT DB2 COMMANDS WAS VICTIM IN TIMEOUT/DEADLOCK. TRY AGAIN LATER'
DBC303 'START OF TS/IX FAULT' . ALARM=YES . WINDOW=LNORESP
Using UDB DB2 utilities to generate explain output for dynamic SQL

With more and more SQL being dynamic in nature, there is an ever-increasing need to be able to ‘explain’ such statements. This article discusses how to use the utilities available in the UDB DB2 environment to generate explain output for dynamic SQL. I ran all the examples on a Windows 2000 machine running UDB DB2 V8.1.

There are a couple of basic methods of obtaining explain information for a query:

- Using the EXPLAIN SQL function of the Control Center (or the Command Center). This gives you a visual representation of the access path and will also optionally populate the EXPLAIN tables. You can do this by firing up the Control
Center, right-clicking on the database name, selecting EXPLAIN SQL, and then typing in (or importing) the SQL you want to explain. If you are using the Command Center, then you need to check the Automatically generate access plan box under the Command Center/Options/Access Plan tab. You can then type in/run your SQL in the Interactive pane, and switch to the Access path pane to see the generated access path.

Both the Control Center and Command Center methods run the SQL before generating the access path. If you want just the access path information without running the SQL, then you need to use a green screen (see method 1 below), which uses the CURRENT EXPLAIN SNAPSHOT and CURRENT EXPLAIN MODE special register variables.

- If you don’t have access to a Windows session, there are three ways of using commands via a green screen session to get the access path/explain information. The first method also gives you the option of not running the SQL to obtain the access path information.

**GREEN SCREEN METHOD 1**

There are two special register variables in UDB that control whether EXPLAIN data is obtained for dynamic SQL – namely CURRENT EXPLAIN SNAPSHOT and CURRENT EXPLAIN MODE. Appendix K, Explain register values, in the SQL Reference manual describes both of these register variables so I won’t do it here. What I will do is show how they could both be set in order to obtain the EXPLAIN information. Let’s first look at the CURRENT EXPLAIN SNAPSHOT special register value. Its default is NO, which means do not take a snapshot or gather EXPLAIN data. You can set it to YES or EXPLAIN, depending on whether you want to execute the query and gather the EXPLAIN data (YES), or not execute the query, but gather the EXPLAIN data (EXPLAIN). The advantage of setting it to EXPLAIN is that then you don’t actually run the SQL – which can be very beneficial if it’s a long-running query! The downside is that you have to switch off the
special register before you can run any other SQL from your CLP session (strictly speaking, any other query will run; it’s just that you can see the result!).

You can find the value of the special register by using:

```
>db2 values current explain snapshot
```

and you can set the value of the special register by using:

```
>db2 set current explain snapshot <value>
```

where `<value>` is YES/EXPLAIN/NO.

The CURRENT EXPLAIN MODE special register determines whether you populate the EXPLAIN tables, and whether you want indexes evaluated/recommended or not. The possible values are NO/YES/EXPLAIN/RECOMMEND INDEXES/EVALUATE INDEXES. You query and set the value of CURRENT EXPLAIN MODE in the same way as you did for CURRENT EXPLAIN SNAPSHOT, namely by using the `db2 values` and the `db2 set` commands.

So if you just want to obtain the cost of a query (without populating the EXPLAIN tables), the sequence of commands would be:

a. Check the value of the CURRENT EXPLAIN MODE special register.

b. Check the value of the CURRENT EXPLAIN SNAPSHOT special register.

c. Set the CURRENT EXPLAIN SNAPSHOT special register value to EXPLAIN.

d. Run the SQL.

e. Run the db2exfmt utility to get the EXPLAIN output.

f. Set the special register value to NO.

This is shown below:

(a) `>db2 values current explain mode`

1
(b) >db2 values current explain snapshot

1

(c) >db2 set current explain snapshot explain

(d) >db2 select * from sysibm.sysdummy1

SQL0217W The statement was not executed as only Explain information requests are being processed. SQLSTATE=01604

(e) >db2exfmt

Type in the database name (in our case ‘sample’), and accept the defaults for all the other prompts:

IBM DATABASE 2 Explain Table Format Tool
Enter Database Name == sample
Connecting to the Database.
Connect to Database Successful.
Enter up to 26 character Explain timestamp (Default -1) ==>
Enter up to 8 character source name (SOURCE_NAME, Default %) ==> 
Enter source schema (SOURCE_SCHEMA, Default %) ==>
Enter section number (Ø for all, Default Ø) ==> 
Enter outfile name. Default is to terminal ==>
IBM DATABASE 2 Explain Table Format Tool

I won’t show the rest of the output, but it does give you the original SQL statement, the optimized one, and the total cost of the statement.

(f) >db2 set current explain snapshot no

The above sequence of commands gives me the cost of the query, and this is all I have to do if I want just the cost of the query (because the CURRENT EXPLAIN MODE was set to NO, the explain information was not written to the EXPLAIN tables).

You only get the message ‘SQL0217W The statement was not executed as only Explain information requests are being processed. SQLSTATE=01604’ from your CLP session; if you sign on to another CLP session with the same user id, you will not get this message.

If you want to populate the EXPLAIN tables, you would follow the
same steps as above, but in addition you would set the CURRENT EXPLAIN MODE special register value to EXPLAIN. This is shown below:

```db2 set current explain snapshot explain
>db2 set current explain mode explain
>db2 select * from sysibm.sysdummy1
SQL0217W The statement was not executed as only Explain information requests are being processed. SQLSTATE=01604
>db2 set current explain snapshot no
>db2 set current explain mode no
```

You can check on the contents of the EXPLAIN tables by using a query such as:

```db2 select int(a.total_cost), int(a.io_cost), int(a.cpu_cost), a.operator_type, a.stmtno, a.sectno, a.explain_time, substr(b.statement_text,1,70) from explain_operator a, explain_statement b where a.explain_time = b.explain_time and b.statement_text like 'SELECT%'
```

<table>
<thead>
<tr>
<th>1</th>
<th>2</th>
<th>3</th>
<th>OPERATOR_TYPE</th>
<th>STMTNO</th>
<th>SECTNO</th>
<th>EXPLAIN_TIME</th>
</tr>
</thead>
<tbody>
<tr>
<td>Ø</td>
<td>Ø</td>
<td>1</td>
<td>RETURN</td>
<td>1</td>
<td>120</td>
<td>201 2002-11-28 22:05:10.184001</td>
</tr>
<tr>
<td>Ø</td>
<td>Ø</td>
<td>1</td>
<td>TBSCAN</td>
<td>1</td>
<td>120</td>
<td>201 2002-11-28 22:05:10.184001</td>
</tr>
</tbody>
</table>

Column 1 gives you the total cost, and the last column gives you the SQL that was explained. I have added columns for io_cost and cpu_cost because they may be useful in comparing the costs of various SQL statements.

Once the EXPLAIN tables have been populated, you can use the visual explain feature of the Control Center to get a graphical representation of the access plan. Fire up the Control Center, right-click on the database name (‘sample’ in our case), and select Show Explained Statements history. You will then get a list
of statements that have been explained and are in the EXPLAIN tables. Simply click once on the statement you want to see, then select the Statements tab and click on Show Access Plan.

GREEN SCREEN METHOD 2

If you want to see a green screen access graph, you have to use the db2expln utility, as shown below. You need to specify the database name (-d), the fact that you want the output to go to the terminal (-t), and the SQL statement that you want to see the access path for (-q). You do not need to set either of the two special register values for this utility to work, and it does not store the explain information in the EXPLAIN tables. In fact, you don’t even need to be connected to the database (that is what the -d parameter does for you!). I have used a different query as an example, to show the format of the access path:

```sql
>db2expln -d sample -t -q "select empno from employee where workdept = 'A00'"
```

<table>
<thead>
<tr>
<th>Isolation Level</th>
<th>Cursor Stability</th>
</tr>
</thead>
<tbody>
<tr>
<td>Blocking</td>
<td>Block Unambiguous Cursors</td>
</tr>
<tr>
<td>Query Optimization Class</td>
<td>5</td>
</tr>
</tbody>
</table>

Partition Parallel = No
Intra-Partition Parallel = No

SQL Path = "SYSIBM", "SYSFUN", "SYSPROC", "DB2ADMN"

SQL Statement:

```sql
select empno from employee where workdept = 'A00'
```

Estimated Cost = 50.118404
Estimated Cardinality = 3.080000

Access Table Name = DB2ADMIN.EMPLOYEE ID = 2,5
| #Columns = 1 |
| Relation Scan |
| Prefetch: Eligible |
| Lock Intents |
| Table: Intent Share |
| Row: Next Key Share |
| Sargable Predicate(s) |
| #Predicates = 1 |
GREEN SCREEN METHOD 3

Perhaps a quicker method of populating the EXPLAIN tables is to use the EXPLAIN ALL WITH SNAPSHOT command – I guess this sets the special register values to YES for both under the covers! I don’t think the command is as flexible as setting the special registers yourself, but if you don’t need the index advisor part, then it is a quicker option than setting the special register values manually. It’s invoked as follows:

```
>db2 explain all with snapshot for "SELECT * from sysibm.sysdummy1"
```

You can then select from the EXPLAIN tables as discussed previously.

If you are using summary tables, then these can be used only with dynamic SQL. I would therefore use the `db2 explain all with snapshot for ...` command to do the explain, because other utilities produce a temporary package containing the SQL, which would be static SQL and therefore not compatible with summary tables.

_C Leonard_

Freelance Consultant (UK)
Software Engineering has announced RT-ExtentManager, which eliminates the problem of exploding DB2 spaces caused by unexpected amounts of data or unknown storage sizes needed by table spaces, for example those belonging to ERP systems like SAP.

By monitoring extents, RT-ExtentManager can automatically eliminate space problems by intelligent sizing of the secondary quantity of DB2 objects in real-time, even with a LOAD job running in parallel.

For further information contact:
Software Engineering, Robert-Stolz-Str 5, D-40470 Duesseldorf, Germany.
Tel: +49 (211) 96149 620
SEGUS Inc, 12007 Sunrise Valley Dr, Reston, VA 20191, USA.
Tel: (703) 391-9650.
URL: http://www.segus.com/

LSI Logic Storage Systems has announced a new back-up and recovery solution in its LogicStor Solutions Program that combines snapshot replication with IBM Tivoli Storage Manager to support IBM DB2 UDB.

Powered by LSI Logic Storage Systems SANtricity Snapshot, the LogicStor Backup and Recovery Solution provides instant back-up and recovery. SANtricity Snapshot enables a point-in-time copy of storage volumes that are then used by back-up software, such as IBM Tivoli Storage Manager, to nearly eliminate the back-up window and provide rapid recovery for DB2 users.

LSI Logic Storage Systems, 1621 Barber Lane, Milpitas, CA 95035, USA.
Tel: (408) 433-8000.

***

Lakeview Technology has announced the release of OmniReplicator Version 4.0.1.

OmniReplicator is a real-time, changed data replication solution that enables companies to move information across the most popular relational databases, including DB2, Informix, Oracle, MS SQL Server, and Sybase. It supports DB2 UDB on Unix, Linux, and Windows, and its streamlined model management process, incremental commit, were developed to meet increasing market needs.

Version 4.0.1 also includes incremental commit, which minimizes downtime by reducing the number of affected tables, to only those that have had a change since the last commit.

Lakeview Technology, 1901 South Meyers Rd, Suite 600, Oakbrook Terrace, IL 60181 USA.
Tel: (630) 282 8100.

***