DB2 update

156

October 2005

In this issue

A tc PUBLICATION

DB2 Update

Published by

Xephon Inc
PO Box 550547
Dallas, Texas 75355
USA

Phone: 214-340-5690
Fax: 214-341-7081

Editor

Trevor Eddolls
E-mail: trevore@xephon.com

Publisher

Colin Smith
E-mail: info@xephon.com

Subscriptions and back-issues

A year's subscription to *DB2 Update*, comprising twelve monthly issues, costs $380.00 in the USA and Canada; £255.00 in the UK; £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 2000 issue, are available separately to subscribers for $33.75 (£22.50) 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.

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.

Display DB2 subsystems

The SUBS (DB2 subsystems) REXX procedure shows the active DB2 systems on an MVS system. The SUBS procedure uses two commands to display this information:

- The first is the console command **DISPLAY OPDATA**:

```
D OPDATA
IEE6Ø3I Ø8.23.33 OPDATA DISPLAY 728
 PREFIX     OWNER      SYSTEM     SCOPE     REMOVE   FAILDSP
 .
 .
 -          DSNTMSTR   MB39Ø27    SYSTEM    NO       SYSPURGE
 %          DSNNMSTR   MB39Ø27    SYSTEM    NO       SYSPURGE
 .
 .
```

  This console command displays the active system services' address space start-up procedure (xxxxMSTR). The start-up procedure name must begin with the subsystem name (xxxx) and end with MSTR. In this example the subsystem names are DSNT and DSNN.

  You must also have TSO CONSOLE authority to run the **D OPDATA** command.

- The second is the DB2 **-DISPLAY GROUP** command:

```
-DISPLAY GROUP
DSN71ØØI  % DSN7GCMD
*** BEGIN DISPLAY OF GROUP(........) GROUP LEVEL(...)
                                     GROUP ATTACH NAME(....)
--------------------------------------------------------------------
DB2                                    DB2 SYSTEM    IRLM
MEMBER   ID  SUBSYS CMDPREF   STATUS   LVL NAME      SUBSYS IRLMPROC
-------- -- ----   --------  -------- -- --------  ----   --------
........   Ø DSNN   %         ACTIVE   71Ø MB39Ø27   IRLN   IRLMPRON
--------------------------------------------------------------------
*** END DISPLAY OF GROUP(........)
DSN9Ø22I  % DSN7GCMD 'DISPLAY GROUP ' NORMAL COMPLETION
```

  This DB2 command displays some additional DB2 subsystem information such as prefix, DB2 level, MVS system name, and IRLM address space. DB2 uses the

IRLM to manage locks. To run the **-DISPLAY GROUP** command, you must have the DISPLAY privilege.

The SUBS procedure collects the data from both commands and displays the following report:

```
------------------------  Active DB2 Systems  ---------- Row 1 to 2 of 2
Command ===>                                              Scroll ===> PAGE
--------------------------------------------------------------------------
Valid cmd: S Select DB2 SubSystem,                               F1 -> Help
or press END to Exit                                             F3 -> End
--------------------------------------------------------------------------
S DB2 System   Prefix  Status  DB2 Level  MVS Name  IRLM Subsys  Irlmproc
- DSNT            -    ACTIVE     71Ø     MB39Ø27      IRLM      IRLMPROC
- DSNN            %    ACTIVE     71Ø     MB39Ø27      IRLN      IRLMPRON
**************************** Bottom of data ****************************
```

SUBS REXX PROCEDURE

```
/*rexx*/
/* The D OPDATA command may be used to display JES3 system and     */
/* sysplex-scoped command prefix characters, as well as the        */
/* command prefixes in use by other subsystems.                    */
/*trace r */
zpfctl = 'OFF'
address ispexec 'vput (zpfctl) profile'
wait_time = 1Ø              /*  seconds to wait for reply */
"CONSOLE ACTIVATE"
lastrc = rc
if lastrc ¬= Ø then
   do
      say ""
      say "*** Unable to activate TSO CONSOLE services!"
      say "*** The return code from 'console activate' was:" lastrc
      say "*** Attempting to recover..."
      "CONSOLE DEACT"
      lastrc = rc
      say "*** CONSOLE DEACT return code was:" lastrc
      "CONSOLE ACTIVATE"
      lastrc = rc
      if lastrc = Ø
      then say "*** Recovery successful!"
      else
      do
         say "*** Recovery attempt failed (I issued CONSOLE DEACT)":,
         "return code was:" lastrc;
          say "*** Perhaps you don't have TSO CONSOLE authority?"
             exit(16)
          end
       end
 "CONSPROF SOLDISPLAY(NO) SOLNUM(1000)"
 /* Display OPDATA console command                                  */
 cmd="D O"
 address "TSO"
 "CONSOLE SYSCMD("cmd")"
 getcode = getmsg("msgs.","SOL",,,wait_time)
 if getcode ¬= 0 then
    do
       say "*** GETMSG return code was:" lastrc
       "CONSPROF SOLDISPLAY(YES) SOLNUM(1000)"
       "CONSOLE DEACTIVATE"
       exit
    end
 address "TSO"
 "CONSPROF SOLDISPLAY(YES) SOLNUM(1000)"
 "CONSOLE DEACTIVATE"
 address ispexec,
 'tbcreate "slist" names(subs,pref,pref,stat,db2l,mvsn,irlms,irlmp)'
 ind=0
 do i=1 to msgs.0
    /* Search DB2 address space xxxxMSTR                            */
    if substr(word(msgs.i,2),5)='MSTR'
    then do
       pref = word(msgs.i,1)
       pref = center(pref,7)
       subs = substr(word(msgs.i,2),1,4)
       /* More DB2 information using -DISPLAY GROUP command         */
       Call Detail
       address ispexec 'tbadd "slist"'
       ind=1
    end
 end
 /* Data not found                                                  */
 if ind=0
 then do
   address ispexec
   zedsmsg = 'Not found'
   zedlmsg = 'DB2 Subsystems not found'
   "setmsg msg(isrz001)"
 end
 address ispexec 'tbtop "slist"'
 cmd=''
 address ispexec 'tbdispl "slist" panel(SUBSP1)'
 item=''
 if cmd='S'
 then item=subs
address ispexec 'tbend "slist"'
 if item=''
 then return
 else return item
Exit
Detail:
  db2=subs
  y = outtrap('out.')
  queue '-display group'
  queue 'end'
  'dsn system('db2')'
  y = outtrap('off')
  stat=''; db2l=''; mvsn=''; irlms=''; irlmp=''
  do j=1 to out.Ø
     if db2 = word(out.j,3)
     then do
        stat = word(out.j,5)
        db2l = word(out.j,6)
        db2l = center(db2l,9)
        mvsn = word(out.j,7)
        irlms= word(out.j,8)
        irlms= center(irlms,11)
        irlmp= word(out.j,9)
     end
  end
Return
```

SUBSP1 ACTIVE DB2 SYSTEMS PANEL

```
)Attr Default(%+_)
   ! type(text)   intens(high) caps(on ) color(yellow)
   $ type(output) intens(high) caps(off) color(yellow)
   § type(output) intens(high) caps(off) color(green)
   # type(text)   intens(high) caps(off) hilite(reverse)
   } type(text)   intens(high) caps(off) color(white)
   { type(text)   intens(high) caps(off) color(green)
   ] type(input)  intens(high) caps(on ) just(left ) pad('-')
   [ type(output) intens(high) caps(off) color(white)
   ¬ type(output) intens(low ) caps(off) just(asis ) color(turquoise)
)Body  Expand(//)
%-/-/- $title             +%-/-/-
%Command ===>_zcmd                                / /%Scroll ===>_amt +
+-------------------------------------------------------------------------
+Valid cmd:!S+Select DB2 SubSystem,
}F1+->}Help
+or press{END+to Exit
}F3+->}End
+-------------------------------------------------------------------------
#S#DB2 System #Prefix #Status #DB2 Level #MVS Name #IRLM Subsys
#Irlmproc +
)Model
```

```
]z§z          ¬z       [z       [z          ¬z         ¬z            ¬z      +
)Init
  .ZVARS = '(cmd subs pref stat db2l mvsn irlms irlmp)'
  &amt = PAGE
  &title = 'Active DB2 Systems'
  .HELP = subsp2
)Reinit
)Proc
)End
```

SUBSP2 HELP PANEL

```
)attr default(/+")
  @ type(text) intens(high) color(red) caps(off) hilite(reverse)
  ~ type(text) intens(high) color(red) caps(off)
  } type(text) intens(high) color(white) hilite(reverse)
  { type(text) intens(high) color(white)
  [ type(text) intens(high) color(white) hilite(uscore)
  ( type(text) intens(high) color(green)
  ) type(text) intens(high) color(pink)
  \ type(text) intens(high) color(blue)
  ] type(text) intens(high) color(yellow)
)body window(69,19) expand ($$)
+  $_$@ H e l p +$_$
+
+ [Panel Explanation:+
+
+ (The~SUBS(procedure shows the{active DB2 systems(on this        +
+ (MVS system.~S(Select the DB2 system you wish to use from       +
+ (the list and press]Enter,(or press]END(to exit.                +
+                                                                  +
+ (You can call this procedure from another procedure like        +
+ (a subroutine by\Call SUBS(or\%SUBS(statement. The return       +
+ (result value is{subsystem name(or{blank.                       +
+                                                                  +
+ (The~SUBS(REXX procedure uses Console command{D OPDATA+         +
+ (and DB2{-DISPLAY GROUP(command for more information.           +
+ (You must have also)TSO CONSOLE(authority. Contact your         +
+ )RACF(administrator.                                             +
+
                                                        }F3: Return+
)init
  .HELP = subsp2
)proc
  .HELP = subsp2
  &zcont = subsp2
)end
```

Bernard Zver (bernard.zver@informatika.si)
DBA (Slovenia)

SQL analyser utility – part 2

This month we publish a sample input program and sample output from the analyser program.

SAMPLE INPUT

```
000100*****************************************************************
000200*       SCREEN DISPLAYING DRUG DETAILS FOR DOCTOR/PHARMACIST   *
000300*****************************************************************
000400 IDENTIFICATION DIVISION.
000500 PROGRAM-ID. ZM000572.
000600 ENVIRONMENT DIVISION.
000700 DATA DIVISION.
000800 WORKING-STORAGE SECTION.
000900     COPY ZM00057.
001000     COPY DFHAID.
001100     EXEC SQL
001200         INCLUDE MASTER
001300     END-EXEC.
001400     EXEC SQL
001500         INCLUDE INVENT
001600     END-EXEC.
001700     EXEC SQL
001800         INCLUDE PRICE
001900     END-EXEC.
002000     EXEC SQL
002100         INCLUDE SQLCA
002200     END-EXEC.
002300 77  DCNT                     PIC 9     VALUE 0.
002400 77  PCNT                     PIC 9     VALUE 0.
002500 77  WS-DATE-TIME             PIC S9(15) COMP-3.
002600 01  WS-DATE.
002700     02 DD                    PIC 9(2).
002800     02 FILLER                PIC X.
002900     02 MM                    PIC 9(2).
003000     02 FILLER                PIC X.
003100     02 YY                    PIC 9(2).
003200
003300 01 WS-TIME.
003400     02 HH                    PIC 9(2).
003500     02 FILLER                PIC X.
003600     02 MM                    PIC 9(2).
003700     02 FILLER                PIC X.
003800     02 SS                    PIC 9(2).
003900
```

```
 01  WS-CURR-PROG-ID       PIC X(8) VALUE IS 'ZM000572'.

     EXEC SQL
        SELECT T1.DRUG_CODE,T1.DRUG_NAME,COMP_NAME,SUP_NAME,
               DRUG_QTY,THRESHOLD_QTY,PRICE_PER_UNIT
        FROM LAX_TAB
        WHERE T1.DRUG_CODE > T2.DRUG_CODE
              AND T2.DRUG_CODE > T3.DRUG_CODE
              AND T3.DRUG_CODE > :I-DRUG-CODE
              AND T4.DRUG_CODE > :I-DRUG-CODE
              AND T4.DRUG_CODE > :I-DRUG-CODE
              AND T4.DRUG_CODE > :I-DRUG-CODE
     END-EXEC.

     EXEC SQL
       SELECT * FROM
       LAX_TAB
       WHERE T1.DRUG_CODE > T2.DRUG_CODE
            AND T2.DRUG_CODE > T3.DRUG_CODE
            AND T1.DRUG_CODE > :I-DRUG-CODE
     END-EXEC.

     END-EXEC.

 LINKAGE SECTION.
 01 DFHCOMMAREA.
     05 LK-PROG-ID        PIC X(8).

 PROCEDURE DIVISION.
*SCREEN FOR INQUIRING DRUG DETAILS
 MAIN-PARA.
        IF LK-PROG-ID NOT = WS-CURR-PROG-ID
            MOVE LOW-VALUES TO ZM005720
            MOVE "SELECT OPTION" TO MSG2O
            PERFORM 2000-SEND-PARA
        ELSE
           PERFORM 2000-RECEIVE-PARA.
 END-MAIN-PARA.

* TO DISPLAY DATE DETAILS
 2000-DATE-PARA.
          EXEC SQL
        SELECT DISTINCT COL1, COL2 , COUNT(*) FROM
        TRG3AUSR.MASTER57 WHERE COL1 > 20
        GROUP BY COL1,COL2  HAVING COUNT(*) > 50
        ORDER BY COL1
      END-EXEC.

      EXEC SQL
               DELETE
       FROM TRG3AUSR.INVENTORY57
       WHERE DRUG_CODE = :I-DRUG-CODE
     END-EXEC.

    EXEC CICS FORMATTIME
     ABSTIME(WS-DATE-TIME)
     DDMMYY(WS-DATE)
     DATESEP(':')
     TIME(WS-TIME)
     TIMESEP(':')
     NOHANDLE
    END-EXEC.
    MOVE WS-DATE TO DATE2O.
    MOVE WS-TIME TO TIME2O.
END-DATE-PARA.

2000-SEND-PARA.
    PERFORM 2000-DATE-PARA.

     EXEC SQL
         SELECT DISTINCT
         DRUG_CODE AS "DRUG",
         DRUG_QTY, AVG(DRUG), COUNT(*)
         INTO :M-DRUG-CODE,:M-DRUG-NAME,:M-COMP-NAME,:M-SUP-
              :I-DRUG-QTY,:I-THRESHOLD-QTY,:P-PRICE-PER-UN
         FROM QUAL.TABLE5 T1, TABLE2 T4
         WHERE  SUP_NAME=12 AND COL LIKE "A%"
       AND T2.DRUG_CODE NOT IN (M,F)
        GROUP BY DRUG_CODE ,DRUG_QTY
        HAVING COUNT(*) > 5
       ORDER BY PEOPLE,CODE,SUP_NAME
     END-EXEC.

     IF SQLCODE = +100 THEN
       MOVE 1 TO A
     ELSE
       MOVE 0 TO A
     END-IF.
     EXEC SQL
        SELECT * FROM
        SRIRAM
        WHERE A > B AND A <=D AND A=F
              AND T2.DRUG_CODE > T3.DRUG_CODE
                 AND T1.DRUG_CODE > :I-DRUG-CODE
        END-EXEC.
     MAPSET('ZM00057')
     ERASE
     NOHANDLE
    END-EXEC.
    EXEC CICS RETURN
```

```
       TRANSID('ZM57')
       COMMAREA(WS-CURR-PROG-ID)
       LENGTH(8)
     END-EXEC.
 END-SEND-PARA.

*DISPLAY THE MAP FOR DOCTOR OR PHARMACIST OPTION SELECTION
 2000-RECEIVE-PARA.
     EXEC CICS RECEIVE
         MAP('ZM00572')
         MAPSET('ZM00057')
         NOHANDLE
     END-EXEC.
     IF EIBAID = DFHPF3 THEN
         EXEC CICS
              XCTL PROGRAM('ZM000571')
         END-EXEC
     ELSE IF EIBAID = DFHPF5 THEN
        MOVE LOW-VALUES TO ZM00572O
        MOVE "SELECT OPTION" TO MSG2O
        PERFORM 2000-SEND-PARA
     ELSE IF EIBAID = DFHENTER THEN
         IF OPTION2I = 'D' THEN
              EXEC SQL
                OPEN CUR1
              END-EXEC
                 MOVE LOW-VALUES TO ZM00572O
       MOVE 'DOCTOR SELECTED AND PRESS F5 FOR REFRESH' TO MSG2O
                 PERFORM 2000-DOCTOR-PARA UNTIL SQLCODE = +100
                 PERFORM 2000-SEND-PARA
              EXEC SQL
                CLOSE CUR1
              END-EXEC
         ELSE IF OPTION2I = 'P' THEN
              EXEC SQL
                OPEN CUR2
              END-EXEC
                 MOVE LOW-VALUES TO ZM00572O
       MOVE 'PHARMACIST SELECTED & PRESS F5 FOR REFRESH' TO MSG2O
                 PERFORM 2000-PHARMA-PARA UNTIL SQLCODE = +100
                 PERFORM 2000-SEND-PARA
              EXEC SQL
               CLOSE CUR2
             END-EXEC
         ELSE
              MOVE LOW-VALUES TO ZM00572O
              MOVE 'OPTION ENTERED IS INVALID' TO MSG2O
              PERFORM 2000-SEND-PARA
         END-IF
     ELSE
```

```
         MOVE LOW-VALUES TO ZMØØ572O
         MOVE 'INVALID KEY PRESSED' TO MSG2O
         PERFORM 2ØØØ-SEND-PARA
     END-IF.
 END-RECEIVE-PARA.

*DISPLAY THE INQUIRY SCREEN FOR DOCTOR.
*THIS DISPLAYS DRUG CODE,DRUG NAME,AND DRUG QUANTITY AVAILABLE.
 2ØØØ-DOCTOR-PARA.
          EXEC SQL
             FETCH CUR1 INTO :M-DRUG-CODE,:M-DRUG-NAME,
                              :I-DRUG-QTY
          END-EXEC.
        IF SQLCODE NOT = 1ØØ
          COMPUTE DCNT = DCNT + 1
          EVALUATE TRUE
          WHEN DCNT = 1
              MOVE M-DRUG-CODE TO CODE21O
              MOVE M-DRUG-NAME TO NAME21O
              MOVE I-DRUG-QTY TO QUAN21O
          WHEN DCNT = 2
              MOVE M-DRUG-CODE TO CODE22O
              MOVE M-DRUG-NAME TO NAME22O
              MOVE I-DRUG-QTY TO QUAN22O
          WHEN DCNT = 3
              MOVE M-DRUG-CODE TO CODE23O
              MOVE M-DRUG-NAME TO NAME23O
              MOVE I-DRUG-QTY TO QUAN23O
          WHEN DCNT = 4
              MOVE M-DRUG-CODE TO CODE24O
              MOVE M-DRUG-NAME TO NAME24O
              MOVE I-DRUG-QTY TO QUAN24O
          WHEN DCNT = 5
              MOVE M-DRUG-CODE TO CODE25O
              MOVE M-DRUG-NAME TO NAME25O
              MOVE I-DRUG-QTY TO QUAN25O
          WHEN DCNT = 6
              MOVE M-DRUG-CODE TO CODE26O
              MOVE M-DRUG-NAME TO NAME26O
              MOVE I-DRUG-QTY TO QUAN26O
          WHEN OTHER
              MOVE Ø TO DCNT
          END-EVALUATE
       ELSE MOVE Ø TO DCNT.
 END-DOCTOR-PARA.

*DISPLAY THE INQUIRY FOR PHARMACIST
 2ØØØ-PHARMA-PARA.
          EXEC SQL
             FETCH CUR2 INTO :M-DRUG-CODE,:M-DRUG-NAME,
                             :I-DRUG-QTY,:I-THRESHOLD-QTY,
                             :M-DATE-OF-INTRO,M-SUP-NAME
          END-EXEC.
       IF SQLCODE NOT = +1ØØ
          COMPUTE PCNT = PCNT + 1
          EVALUATE TRUE
          WHEN PCNT = 1
              MOVE M-DRUG-CODE TO CODE210
              MOVE M-DRUG-NAME TO NAME210
              MOVE I-DRUG-QTY TO QUAN210
              MOVE I-THRESHOLD-QTY TO THRES210
              MOVE M-DATE-OF-INTRO TO EXP210
              MOVE M-SUP-NAME TO SUP210
          WHEN PCNT = 2
              MOVE M-DRUG-CODE TO CODE220
              MOVE M-DRUG-NAME TO NAME220
              MOVE I-DRUG-QTY TO QUAN220
              MOVE I-THRESHOLD-QTY TO THRES220
              MOVE M-DATE-OF-INTRO TO EXP220
              MOVE M-SUP-NAME TO SUP220
          WHEN PCNT = 3
              MOVE M-DRUG-CODE TO CODE230
              MOVE M-DRUG-NAME TO NAME230
              MOVE I-DRUG-QTY TO QUAN230
              MOVE I-THRESHOLD-QTY TO THRES230
              MOVE M-DATE-OF-INTRO TO EXP230
              MOVE M-SUP-NAME TO SUP230
          WHEN PCNT = 4
              MOVE M-DRUG-CODE TO CODE240
              MOVE M-DRUG-NAME TO NAME240
              MOVE I-DRUG-QTY TO QUAN240
              MOVE I-THRESHOLD-QTY TO THRES240
              MOVE M-DATE-OF-INTRO TO EXP240
              MOVE M-SUP-NAME TO SUP240
          WHEN PCNT = 5
              MOVE M-DRUG-CODE TO CODE250
              MOVE M-DRUG-NAME TO NAME250
              MOVE I-DRUG-QTY TO QUAN250
              MOVE I-THRESHOLD-QTY TO THRES250
              MOVE M-DATE-OF-INTRO TO EXP250
              MOVE M-SUP-NAME TO SUP250
          WHEN PCNT = 6
              MOVE M-DRUG-CODE TO CODE260
              MOVE M-DRUG-NAME TO NAME260
              MOVE I-DRUG-QTY TO QUAN260
              MOVE I-THRESHOLD-QTY TO THRES260
              MOVE M-DATE-OF-INTRO TO EXP260
              MOVE M-SUP-NAME TO SUP260
          WHEN OTHER
              MOVE Ø TO PCNT
```

```
Ø251ØØ           END-EVALUATE
Ø252ØØ       ELSE MOVE Ø TO PCNT.
Ø253ØØ  END-PHARMA-PARA.
```

SAMPLE OUTPUT

```
********************************* *********************************

PROGRAM NAME    : LAX1

SQL
---

       SELECT T1.DRUG_CODE,T1.DRUG_NAME,COMP_NAME,SUP_NAME,
              DRUG_QTY,THRESHOLD_QTY,PRICE_PER_UNIT
       FROM LAX_TAB
       WHERE T1.DRUG_CODE > T2.DRUG_CODE
             AND T2.DRUG_CODE > T3.DRUG_CODE
             AND T3.DRUG_CODE > :I-DRUG-CODE
             AND T4.DRUG_CODE > :I-DRUG-CODE
             AND T4.DRUG_CODE > :I-DRUG-CODE
             AND T4.DRUG_CODE > :I-DRUG-CODE


COLUMNS IN UNIQUE WHERE
-----------------------
T1.DRUG_CODE
T2.DRUG_CODE
T3.DRUG_CODE
T4.DRUG_CODE

TABLES ACCESSED
---------------
LAX_TAB

SQL_TYPE        : SELECT
NO OF COLUMNS   : 7
NO OF OPERATOR  : 6
NO OF DISTINCT  : 4
NO OF AND       : 5
NO OF OR        : Ø
NO OF >=        : Ø
NO OF <=        : Ø
NO OF =         : Ø
NO OF >         : 6
NO OF <         : Ø
DISTINCT        : NO
LIKE            : NO
SELECT *        : NO
```

```
NOT IN         : NO

GUIDELINES
----------



******************************** ********************************

PROGRAM NAME   : LAX1

SQL
---

      SELECT * FROM
      LAX_TAB
      WHERE T1.DRUG_CODE > T2.DRUG_CODE
           AND T2.DRUG_CODE > T3.DRUG_CODE
           AND T1.DRUG_CODE > :I-DRUG-CODE


COLUMNS IN UNIQUE WHERE
-----------------------
T1.DRUG_CODE
T2.DRUG_CODE

SQL_TYPE       : SELECT
NO OF COLUMNS  : 7
NO OF OPERATOR : 3
NO OF DISTINCT : 2
NO OF AND      : 2
NO OF OR       : Ø
NO OF >=       : Ø
NO OF <=       : Ø
NO OF =        : Ø
NO OF >        : 3
NO OF <        : Ø
DISTINCT       : NO
LIKE           : NO
SELECT *       : YES
NOT IN         : NO

GUIDELINES
----------
1) SELECT * IS USED IN THE SQL. SQL SHOULD ALWAYS LIST THE NAMED
COLUMNS TO BE RETURNED TO THE PROGRAM. SELECT * SHOULD NEVER
BE USED.
```

```
********************************  ********************************

PROGRAM NAME   : LAX1

SQL
---

       SELECT DISTINCT COL1, COL2 , COUNT(*) FROM
       TRG3AUSR.MASTER57 WHERE COL1 > 2Ø
       GROUP BY COL1,COL2  HAVING COUNT(*) > 5Ø
       ORDER BY COL1


COLUMNS IN ORDER BY
--------------------
COL1

  COLUMN FUNCTION
--------------------
COUNT(*)

COLUMNS IN GROUP BY
--------------------
COL1
COL2

COLUMNS IN UNIQUE WHERE
-----------------------
COL1

TABLES ACCESSED
---------------
TRG3AUSR.MASTER57

SQL_TYPE       : SELECT
NO OF COLUMNS  : 2
NO OF OPERATOR : 1
NO OF DISTINCT : 1
NO OF AND      : Ø
NO OF OR       : Ø
NO OF >=       : Ø
NO OF <=       : Ø
NO OF =        : Ø
NO OF >        : Ø
NO OF <        : Ø
DISTINCT       : YES
LIKE           : NO
SELECT *       : NO
NOT IN         : NO
```

```
GUIDELINES
----------
7) AVOID USING DISTINCT.IF DUPLICATES WILL NOT CAUSE A PROBLEM,DO
NOT CODE DISTINCT.
1Ø) TRY TO SORT ONLY ON INDEXED COLUMNS. WHEN USING ORDER BY GROUP
BY, DISTINCT. IT IS BEST TO USE ONLY INDEXED COLUMNS.



******************************** ********************************

PROGRAM NAME   : LAX1

SQL
---

         SELECT DISTINCT
         DRUG_CODE AS "DRUG",
         DRUG_QTY, AVG(DRUG), COUNT(*)
          INTO :M-DRUG-CODE,:M-DRUG-NAME,:M-COMP-NAME,:M-SUP-
               :I-DRUG-QTY,:I-THRESHOLD-QTY,:P-PRICE-PER-UN
         FROM QUAL.TABLE5 T1, TABLE2 T4
         WHERE  SUP_NAME=12 AND COL LIKE "A%"
       AND T2.DRUG_CODE NOT IN (M,F)
        GROUP BY DRUG_CODE ,DRUG_QTY
        HAVING COUNT(*) > 5
       ORDER BY PEOPLE,CODE,SUP_NAME


COLUMNS IN ORDER BY
--------------------
PEOPLE
CODE
SUP_NAME

  COLUMN FUNCTION
--------------------
AVG(DRUG)
COUNT(*)

COLUMNS IN GROUP BY
--------------------
DRUG_CODE
DRUG_QTY

COLUMNS IN UNIQUE WHERE
-----------------------
SUP_NAME

TABLES ACCESSED
```

```
---------------
QUAL.TABLE5
TABLE2

SQL_TYPE        : SELECT
NO OF COLUMNS   : 2
NO OF OPERATOR  : 1
NO OF DISTINCT  : 1
NO OF AND       : Ø
NO OF OR        : Ø
NO OF >=        : Ø
NO OF <=        : Ø
NO OF =         : Ø
NO OF >         : Ø
NO OF <         : Ø
DISTINCT        : YES
LIKE            : YES
SELECT *        : NO
NOT IN          : YES

GUIDELINES
----------
7) AVOID USING DISTINCT.IF DUPLICATES WILL NOT CAUSE A PROBLEM,DO
NOT CODE DISTINCT.
1Ø) TRY TO SORT ONLY ON INDEXED COLUMNS. WHEN USING ORDER BY GROUP
BY, DISTINCT. IT IS BEST TO USE ONLY INDEXED COLUMNS.
22) USE IN INSTEAD OF LIKE. IF YOU KNOW THAT ONLY CERTAIN
OCCURRENCES EXIST, USING IN WITH THE SPECIFIC LIST IS MORE
EFFICIENT THAN USING LIKE.
23) AVOID USING NOT (EXCEPT WITH EXISTS). NOT SHOULD ONLY BE USED AS
AN ALTERNATIVE TO VERY COMPLEX PREDICATES.



*********************************  *********************************

PROGRAM NAME    : LAX1

SQL
---

        SELECT * FROM
        SRIRAM
        WHERE A > B AND A <=D AND A=F
              AND T2.DRUG_CODE > T3.DRUG_CODE
                 AND T1.DRUG_CODE > :I-DRUG-CODE


COLUMNS IN UNIQUE WHERE
-----------------------
```

```
A
T2.DRUG_CODE
T1.DRUG_CODE

SQL_TYPE        : SELECT
NO OF COLUMNS   : 3
NO OF OPERATOR  : 5
NO OF DISTINCT  : 3
NO OF AND       : 4
NO OF OR        : Ø
NO OF >=        : Ø
NO OF <=        : 1
NO OF =         : 1
NO OF >         : 3
NO OF <         : Ø
DISTINCT        : NO
LIKE            : NO
SELECT *        : YES
NOT IN          : NO

GUIDELINES
----------
1) SELECT * IS USED IN THE SQL. SQL SHOULD ALWAYS LIST THE NAMED
COLUMNS TO BE RETURNED TO THE PROGRAM. SELECT * SHOULD NEVER
BE USED.
```

T S Laxminarayan (ts_laxminarayan@yahoo.com)
Systems Programmer
Tata Consultancy Services Ltd (India)

DB2 UDB for LUW 8.2 – how to list tablespace information

This article looks at how to get DB2 UDB for LUW tablespace information in an easy-to-read format.

A REXX EXEC was published in *DB2 Update* in March 2003 (see ‘Formatting the LIST TABLESPACE output’ in issue 125) to list tablespace information in a more readable format than the native **db2 list tablespace show detail** command. What happens if the output format changes? You will need to modify the program. There is another way to get the tablespace information and that is to use the snapshot table functions.

The following was run on a Windows 2000 Professional system running DB2 UDB V8.2 and using the SAMPLE database.

First let's look at the SNAPSHOT command. The SNAPSHOT command you would use to get tablespace information for the SAMPLE database is:

```
>db2 get snapshot for tablespaces on sample
```

This command produces lots of output, and you could write a program to accept this and format it for you. However, this is no better than issuing the LIST TABLESPACE command and processing the output – you still need to write and, more importantly, maintain a program.

So now let's look at the SNAPSHOT table functions. These table functions are documented in the *SQL Reference* volume 1 (Chapter 3 – 'Functions'). The main ones we are interested in are SNAPSHOT_TBS (activity from a tablespace) and SNAPSHOT_TBS_CFG (configuration information from a tablespace). These tables can be queried as normal tables, so to list all the columns in the tables we can use the following query:

```
>db2 connect to SAMPLE

>db2 describe select * from table(SNAPSHOT_TBS('SAMPLE',-2)) as s
```

where *SAMPLE* is the database we are connected to.

The columns for the SNAPSHOT_TBS and SNAPSHOT_TBS_CFG tables are shown below:

```
SNAPSHOT_TBS                      SNAPSHOT_TBS_CFG
SNAPSHOT_TIMESTAMP (TS)           SNAPSHOT_TIMESTAMP (TS)
POOL_DATA_L_READS (BI)            TABLESPACE_ID (BI)
POOL_DATA_P_READS (BI)            TABLESPACE_NAME (VARCHAR)
POOL_ASYNC_DATA_READS (BI)        TABLESPACE_TYPE (SI)
POOL_DATA_WRITES (BI)             TABLESPACE_STATE (BI)
POOL_ASYNC_DATA_WRITES (BI)       NUM_QUIESCERS (BI)
POOL_INDEX_L_READS (BI)           STATE_CHANGE_OBJ_ID (BI)
```

```
POOL_INDEX_P_READS (BI)         STATE_CHANGE_TBS_ID (BI)
POOL_INDEX_WRITES (BI)          MIN_RECOVERY_TIME (TS)
POOL_ASYNC_INDEX_WRITES (BI)    TBS_CONTENTS_TYPE (SI)
POOL_READ_TIME (BI)             BUFFERPOOL_ID (BI)
POOL_WRITE_TIME (BI)            NEXT_BUFFERPOOL_ID (BI)
POOL_ASYNC_READ_TIME (BI)       PAGE_SIZE (BI)
POOL_ASYNC_WRITE_TIME (BI)      EXTENT_SIZE (BI)
POOL_ASYNC_DATA_READ_REQS (BI)  PREFETCH_SIZE (BI)
DIRECT_READS (BI)               TOTAL_PAGES (BI)
DIRECT_WRITES (BI)              USABLE_PAGES (BI)
DIRECT_READ_REQS (BI)           USED_PAGES (BI)
DIRECT_WRITE_REQS (BI)          FREE_PAGES (BI)
DIRECT_READ_TIME (BI)           PENDING_FREE_PAGES (BI)
DIRECT_WRITE_TIME (BI)          HIGH_WATER_MARK (BI)
UNREAD_PREFETCH_PAGES (BI)      REBALANCER_MODE (BI)
POOL_ASYNC_INDEX_READS (BI)     REBALANCER_EXTENTS_REMAINING (BI)
POOL_DATA_TO_ESTORE (BI)        REBALANCER_EXTENTS_PROCESSED (BI)
POOL_INDEX_TO_ESTORE (BI)       REBALANCER_PRIORITY (BI)
POOL_INDEX_FROM_ESTORE (BI)     REBALANCER_START_TIME (TS)
POOL_DATA_FROM_ESTORE (BI)      REBALANCER_RESTART_TIME (TS)
FILES_CLOSED (BI)               LAST_EXTEND_MOVED (BI)
TABLESPACE_NAME (VARCHAR)       NUM_RANGES (BI)
                                NUM_CONTAINERS (BI)
```

where *BI* is bigint, *SI* is smallint, and *TS* is timestamp.

You can see that there are lots of columns here and lots of information! All the columns are described in detail in the *SQL Reference* manual, so I won't go through them again – what I will do is show some queries that you can run to replace the **LIST TABLESPACE(S)** commands.

The **LIST TABLESPACES** command returns the values *Tablespace ID*, *Name*, *Type*, *Contents*, *State*, and *Detailed explanation* (this is shown in the example output below):

```
>db2 list tablespaces

           Tablespaces for Current Database

 Tablespace ID                        = 0
 Name                                 = SYSCATSPACE
 Type                                 = System managed space
 Contents                             = Any data
 State                                = 0x0000
   Detailed explanation:
     Normal
```

The equivalent columns (there isn't an equivalent for *Detailed explanation*) in the SNAPSHOT tables are:

```
TABLESPACE_ID (BI)
TABLESPACE_NAME (VARCHAR)
TABLESPACE_TYPE (SI)
TABLESPACE_STATE (BI)
TBS_CONTENTS_TYPE (SI)
```

Therefore my query would look like this:

```
>db2 select snapshot_timestamp TABLESPACE_ID,
substr(tablespace_name,1,20), tablespace_type, tablespace_state,
tbs_contents_type as type from table(SNAPSHOT_TBS_CFG('sample',-2)) as s
```

Which would produce the following output:

```
TABLESPACE_ID    2          TABLESPACE_TYPE  TABLESPACE_STATE  TYPE
------------- ------------- ---------------- ----------------  ------
            0 SYSCATSPACE           1                    0      0
            1 TEMPSPACE1            1                    0      2
            2 USERSPACE1            1                    0      0
            3 SYSTOOLSPACE          1                    0      0
```

(not showing all the output and the timestamp as the first column).

Note that we get a snapshot timestamp as part of the output. So, to store the output, I could create a table called TAB_LIST_TBS as shown below:

```
>db2 create table tab_list_tbs (timestamp timestamp, TABLESPACE_ID
bigint, TABLESPACE_NAME VARCHAR(128), TABLESPACE_TYPE smallint,
TABLESPACE_STATE bigint, TBS_CONTENTS_TYPE smallint)
```

And I would populate it thus:

```
>db2 insert into tab_list_tbs select snapshot_timestamp, TABLESPACE_ID,
substr(tablespace_name,1,20), tablespace_type, tablespace_state,
tbs_contents_type from table(SNAPSHOT_TBS_CFG('sample',-2)) as s
```

This would be just the basics of what we would want in the table. For DMS tablespaces we might want to record a tablespace percentage full value, etc.

So let's look at the other LIST TABLESPACE commands and 'translate' those.

Say we want to find the container information for the two tablespaces shown below.

Tablespace (TS2C), which we defined with two SMS containers:

```
>db2 CREATE  REGULAR  TABLESPACE TS2C PAGESIZE 4 K  MANAGED BY SYSTEM
USING ('C:\cont1', 'C:\cont2' ) EXTENTSIZE 16 OVERHEAD 10.5 PREFETCHSIZE
16 TRANSFERRATE 0.14 BUFFERPOOL  IBMDEFAULTBP  DROPPED TABLE RECOVERY ON
```

and tablespace (TS1D), which we defined with one DMS container:

```
>db2 CREATE  REGULAR  TABLESPACE TS1D PAGESIZE 4 K  MANAGED BY DATABASE
USING ( FILE 'C:\Temp\dms01' 5120 ) EXTENTSIZE 16 OVERHEAD 10.5
PREFETCHSIZE 16 TRANSFERRATE 0.14 BUFFERPOOL  IBMDEFAULTBP  DROPPED
TABLE RECOVERY ON
```

First we have to get the *Tablespace ID* (using the **list tablespaces** command):

```
>db2 list tablespaces

………..

Tablespace ID                        = 10
 Name                                 = TS2C
 Type                                 = System managed space
 Contents                             = Any data
 State                                = 0x0000
   Detailed explanation:
     Normal

 Tablespace ID                        = 11
 Name                                 = TS1D
 Type                                 = Database managed space
 Contents                             = Any data
 State                                = 0x0000
   Detailed explanation:
     Normal
```

To get the container details for tablespaces 10 and 11 we would use the following commands:

```
>db2 list tablespace containers for 10

 Container ID                         = 0
 Name                                 = C:\cont1
 Type                                 = Path

 Container ID                         = 1
 Name                                 = C:\cont2
 Type                                = Path

>db2 list tablespace containers for 11

            Tablespace Containers for Tablespace 11

 Container ID                        = Ø
 Name                                = C:\Temp\dmsØ1
 Type                                = File
```

This query returns the *Container ID*, its **Name**, and its **Type**. The equivalent SNAPSHOT table columns (taken from the SNAPSHOT_CONTAINER) table are shown below:

```
>db2 select snapshot_timestamp TABLESPACE_ID,
substr(tablespace_name,1,11), container_id, substr(container_name,1,35),
container_type from table(SNAPSHOT_CONTAINER('sample',-2)) as s

TABLESPACE_ID  2        CONTAINER_ID         4       CONTAINER_TYPE
-------------- -------- -------------------- -------  --------------
            1Ø TS2C                        Ø C:\cont1           Ø
            1Ø TS2C                        1 C:\cont2           Ø
            11 TS1D                        Ø C:\Temp\dmsØ1      6
```

(The output above shows just the information for tablespaces 10 and 11 and without the timestamp information.)

You can see that the container type for an SMS tablespace is 0 and for a DMS file is 6.

We have seen how the SNAPSHOT table functions can replace the **list tablespace** command and make querying and recording tablespace information easier than writing and maintaining scripts.

C Leonard
Freelance Consultant (UK)

Submitting DB2 commands through IFI

You can use IFI (the Instrumentation Facility Interface) in a monitor program (a program or function outside DB2 that receives information about DB2) to perform the following tasks:

- Submit DB2 commands through IFI.
- Obtain trace data.
- Pass data to DB2 through IFI.

This article describes the first task. The first option on the selection panel creates all the necessary DDL statements for loading all DB2 commands into the table (tip: put all DROP commands into comments during the initial execution of the generated job). Option 2 populates the table with commands together with their full syntax from the sequential file. Option 3 displays all DB2 commands and after you choose one of them you will see its command syntax in the upper part of the panel. Next, in the command field having the required options (clauses), you should enter the command you want to execute. The result of the command (or error report) will be displayed on a separate panel.

A monitor program issuing IFI requests must be connected to DB2 at the thread level. If the program contains SQL statements, you must precompile the program and create a DB2 plan using the BIND process. If the monitor program does not contain any SQL statements, it does not have to be precompiled. However, as is the case in all the attachment environments, even though an IFI-only program (ie one with no SQL statements) does not have a plan of its own, it can use any plan to get the thread level connection to DB2. The monitor program can run in either 24- or 31-bit mode.

IFI can be accessed through any of the DB2 attachment facilities. Part of the call attachment facility is a DB2 load module, DSNALI, known as the call attachment facility language interface. DSNALI has the alias names DSNHLI2 and DSNWLI2. The module has five entry points – DSNALI, DSNHLI, DSNHLI2, DSNWLI, and DSNWLI2:

- Entry point DSNALI handles explicit DB2 connection service requests.

- DSNHLI and DSNHLI2 handle SQL calls (use DSNHLI if your application program link-edits CAF; use DSNHLI2 if your application program loads CAF).
- DSNWLI and DSNWLI2 handle IFI calls (use DSNWLI if your application program link-edits CAF; use DSNWLI2 if your application program loads CAF).

You can access the DSNALI module either by explicitly issuing LOAD requests when your program runs, or by including the module in your load module when you link-edit your program. The following example depicts an IFI call in a REXX program:

```
Address LINKPGM "DSNWLI2 COMMAND ifca returnarea outputarea"
```

A DB2 command resides in the output area; a monitor program can submit that command by issuing a COMMAND request to IFI. The DB2 command is processed and the output messages are returned to the monitor program in the return area. The program's IFCA (Instrumentation Facility Communication Area) is a communications area between the monitor program and IFI. IFCA is a required parameter on all IFI requests. It contains information about the success of the call in its return code and reason code fields. The monitor program is responsible for allocating storage for the IFCA and initializing it. The IFCA must be initialized to binary zeros and the eye catcher, 4-byte owner field, and length field, must be set by the monitor program. Failure to properly initialize the IFCA results in IFI requests being denied. The monitor program is also responsible for checking the IFCA return code and reason code fields to determine the status of the request. The IFCA fields are described in Figure 1. The IFCA is mapped by Assembler mapping macro DSNDIFCA.

You must specify a return area on all COMMAND requests. IFI uses the return area to return command responses, synchronous data, and asynchronous data to the monitor program – see Figure 2.

Data returned on a COMMAND request consists of varying-

Name	Hex offset	Data type	Description
IFCALEN	0	Hex, 2 bytes	Length of IFCA.
IFCAFLGS	2	Hex, 1 byte	Processing flags. - IFCAGLBL, X'80' This bit is on if an IFI request is to be processed on all members of a data sharing group.
	3	Hex, 1 byte	Reserved.
IFCAID	4	Character, 4 bytes	Eye catcher for block, IFCA.
IFCAOWNR	8	Character, 4 bytes	Owner field, provided by the monitor program. This value is used to establish ownership of an OPn destination and to verify that a requester can obtain data from the OPn destination. This is not the same as the owner ID of a plan.
IFCARC1	C	Four-byte signed integer	Return code for the IFI call. Binary zero indicates a successful call. See Part 3 of DB2 Messages and Codes for information about reason codes. For a return code of 8 from a COMMAND request, the IFCAR0 and IFCAR15 values contain more information.
IFCARC2	10	Four-byte signed integer	Reason code for the IFI call. Binary zero indicates a successful call. See Part 3 of DB2 Messages and Codes for information about reason codes.

Figure 1: Instrumentation Facility Communication Area

Name	Hex offset	Data type	Description
IFCABM	14	Four-byte signed integer	Number of bytes moved to the return area. A non-zero value in this field indicates information was returned from the call. Only complete records are moved to the monitor program area.
IFCABNM	18	Four-byte signed integer	Number of bytes that did not fit in the return area and still remain in the buffer. Another READA request will retrieve that data. Certain IFI requests return a known quantity of information. Other requests will terminate when the return area is full.
	1C	Four-byte signed integer	Reserved.
IFCARLC	20	Four-byte signed integer	Indicates the number of records lost prior to a READA call. Records are lost when the OP buffer storage is exhausted before the contents of the buffer are transferred to the application program via an IFI READA request. Records that do not fit in the OP buffer are not written and are counted as records lost.

Figure 1: Instrumentation Facility Communication Area (cont)

Name	Hex offset	Data type	Description
IFCAOPN	24	Character, 4 bytes	Destination name used on a READA request. This field identifies the buffer requested, and is required on a READA request. Your monitor program must set this field. The instrumentation facility fills in this field on START TRACE to an OPn destination from a monitor program. If your monitor program started multiple OPn destination traces, the first one is placed in this field. If your monitor program did not start an OPn destination trace, the field is not modified. The OPn destination and owner ID are used on subsequent READA calls to find the asynchronous buffer.
IFCAOPNL	28	Two-byte signed integer	Length of the OPn destinations started. On any command entered by IFI, the value is set to X'0004'. If an OPn destination is started, the length is incremented to include all OPn destinations started.
	2A	Two-byte signed integer	Reserved.
IFCAOPNR	2C	Character, 8 fields of 4 bytes each	Space to return 8 OPn destination values.

Figure 1: Instrumentation Facility Communication Area (cont)

Name	Hex offset	Data type	Description
IFCATNOL	4C	Two-byte signed integer	Length of the trace numbers plus 4. On any command entered by IFI the value is set to X'0004'. If a trace is started, the length is incremented to include all trace numbers started.
	4E	Two-byte signed integer	Reserved.
IFCATNOR	50	Character, 8 fields of 2 bytes each	Space to hold up to eight EBCDIC trace numbers that were started. The trace number is required if the MODIFY TRACE command is used on a subsequent call.
IFCADL	60	Hex, 2 bytes	Length of diagnostic information.
	62	Hex, 2 bytes	Reserved.
IFCADD	64	Character, 80 bytes	Diagnostic information. - IFCAFCI, offset 64, 6 bytes This contains the RBA of the first CI in the active log if IFCARC2 is 00E60854. See "Reading specific log records (IFCID 0129)" in topic APPENDIX1.3.3.2 for more information. - IFCAR0, offset 6C, 4 bytes

Figure 1: Instrumentation Facility Communication Area (cont)

Name	Hex offset	Data type	Description
IFCADD	64	Character, 80 bytes	For COMMAND requests, this field contains -1 or the return code from the component that executed the command. - IFCAR15, offset 70, 4 bytes For COMMAND requests, this field contains one of the following values: 0 The command completed successfully 4 Internal error. 8 The command was not processed because of errors in the command. 12 The component that executed the command returned the return code in IFCAR0. 16 An abend occurred during command processing. Command processing might be incomplete, depending on when the error occurred. See IFCAR0 for more information.

Figure 1: Instrumentation Facility Communication Area (cont)

Name	Hex offset	Data type	Description
IFCADD	64	Character, 80 bytes	Response buffer storage was not available. The command completed, but no response messages are available. See IFCAR0 for more information. 24 Storage was not available in the DSNMSTR address space. The command was not processed. 28 CSA storage was not available. If a response buffer is available, the command might have partially completed. See IFCAR0 for more information. 32 The user is not authorized to issue the command. The command was not processed. - IFCAGBPN, offset 74, 8 bytes

Figure 1: Instrumentation Facility Communication Area (cont)

Name	Hex offset	Data type	Description
IFCADD	64	Character, 80 bytes	This is the group buffer pool name in error if IFCARC2 is 00E60838 or 00E60860 - IFCABSRQ, offset 88, 4 bytes This is the size of the return area required when the reason code is 00E60864. - IFCAHLRS, offset 8C, 6 bytes This field can contain the highest LRSN or log RBA in the active log (when WQALLMOD is 'H'). Or, it can contain the RBA of the log CI given to the Log Exit when the last CI written to the log was not full, or an RBA of zero (when WQALLMOD is 'P').
IFCAGRSN	98	Four-byte signed integer	Reason code for the situation in which an calls requests data from members of a data sharing group, and not all the data is returned from group members. See Part 3 of DB2 Messages and Codes for information about reason codes.

Figure 1: Instrumentation Facility Communication Area (cont)

Name	Hex offset	Data type	Description
IFCAGBM	9C	Four-byte signed integer	Total length of data that was returned from other data sharing group members and fit in the return area.
IFCAGBNM	A0	Four-byte signed integer	Total length of data that was returned from other data sharing group members and did not fit in the return area.
IFCADMBR	A4	Character, 8 bytes	Name of a single data sharing group member on which an IFI request is to be executed. Otherwise, this field is blank. If this field contains a member name, DB2 ignores field IFCAGLBL.
IFCARMBR	AC	Character, 8 bytes	Name of the data sharing group member from which data is being returned. DB2 sets this field in each copy of the IFCA that it places in the return area, not in the IFCA of the application that makes IFI request.

Figure 1: Instrumentation Facility Communication Area (cont)

length segments (X'xxxxrrrr', where the length is two bytes and the next two bytes are reserved) followed by the message text. More than one record can be returned. The last character in the return area is a new-line character (X'15'). The output area is used by COMMAND and WRITE requests. The area can contain a DB2 command or information to be written to the

Hex offset	Data type	Description
0	Signed four-byte integer	The length of the return area, plus 4. This must be set by the monitor program. The valid range for READA requests is 100 to 1048576 (X'00000064' to X'00100000').
4	Character, varying-length	DB2 places as many varying-length DB2 places as many varying-length records as it can fit into the area following the length field. The monitor program's length field is not modified by DB2. Each varying-length trace record has a 2-byte length field. After a COMMAND request, the last character in the return area is a new-line character (X'15').

Figure 2: Return area

instrumentation facility. The first two bytes of an area contain the length of the monitor program's record to write or the DB2 command to be issued, plus four additional bytes. The next two bytes are reserved. You can specify any length from 10 to 4096 (X'000A0000' to X'10000000'). The rest of the area is the actual command or record text. The record returned from a command request can contain none or many message text segments. Each segment is a varying-length message (LLZZ, where LL is the 2-byte length and ZZ is a 2-byte reserved area) followed by message text. The IFCA's IFCABM field contains the total number of bytes moved.

CMDREXX0

```
/* rexx */
Address ISPEXEC 'select panel(CMDPNL1)'
```

```
SYSADM.CMDFILE.STAL (FB/8Ø/2792Ø)
```

```
-ALTER BUFFERPOOL
```

```
Alters attributes for the buffer pools

>>__ALTER BUFFERPOOL__(__bpname__)__ _________________ ______________>
                                    |_VPSIZE(integer)_|
>__ _______________________ __ _________________ ____________________>
   |_VPTYPE__ _PRIMARY___ _|  |_HPSIZE(integer)_|
             |_DATASPACE_|
>__ _________________ __ __________________ _________________________>
   |_VPSEQT(integer)_|  |_VPPSEQT(integer)_|
>__ ___________________ __ _________________ __ _______________ _____>
   |_VPXPSEQT(integer)_|  |_HPSEQT(integer)_|  |_DWQT(integer)_|
>__ __________________________ __ ____________________ ______________>
   |_VDWQT(integer1,integer2)_|  |_CASTOUT(_ _YES_ _)_|
                                            |_NO__|
>__ ___________________ _____________________________________________>
   |_PGSTEAL__ _LRU__ _|
              |_FIFO_|

-ALTER GROUPBUFFERPOOL
Alters attributes for the group buffer pools

>>__ALTER GROUPBUFFERPOOL__(__ _gbpname________ __)__________________>
                              |_structure-name_|
>__ _____________________ __ ____________________ ___________________>
   |_GBPCACHE(_ _YES_ _)_|  |_AUTOREC(_ _YES_ _)_|
               |_NO__|                 |_NO__|
>__ ______________ __ _________________ __ __________________ _______>
   |_RATIO(ratio)_|  |_CLASST(integer)_|  |_GBPOOLT(integer)_|
>__ ___________________ _____________________________________________>
   |_GBPCHKPT(integer)_|

-ALTER UTILITY
Alters parameter values of the REORG utility

>>__ALTER__UTILITY__(__utility-id__)__REORG__________________________>
>__ ___________________________ __ ______________________ ___________>
   |_DEADLINE(_ _NONE______ _)_|  |_MAXRO(_ _integer_ _)_|
               |_timestamp_|               |_DEFER___|
>__ _________________________ __ ______________________ _____________>
   |_LONGLOG(_ _CONTINUE_ _)_|  |_DELAY(___integer___)_|
              |_TERM_____|
              |_DRAIN____|

-ARCHIVE LOG
Enables a site to close a current active log and open the next
available log dataset

>>__ARCHIVE  LOG_____________________________________________________>
    _SCOPE(MEMBER)_____________________________________
>__|___________________________________________________|_____________>
```

```
   |_SCOPE(GROUP)_______________________________________|
   |_MODE(QUIESCE)__ ____________ __ __________________ _|
   |                |_TIME(nnn)_|  |         _NO__    | |
   |                               |_WAIT(_|_YES_|_)_| |
   |_CANCEL OFFLOAD_____________________________________|
```

```
-CANCEL THREAD
Cancels processing for specific local or distributed threads

>>__CANCEL__ _THREAD(token)___________ __ ______ __ ____________ _____>
            |_DDF THREAD(_ _luwid_ _)_|  |_DUMP_|  |_NOBACKOUT_|
                          |_token_|
```

```
-DISPLAY ARCHIVE
Displays information about archive log processing

>>__DISPLAY  ARCHIVE__________________________________________________>
```

```
-DISPLAY BUFFERPOOL
Displays information about the buffer pools

                         _(__ACTIVE__)_____
>>__DISPLAY  BUFFERPOOL__|__________________|_________________________>
                         |_(__*__)__________|
                         |    <_,______     |
                         |_(____bpname_|__)_|
>__ ________________________ __ ____________________ ________________>
   |          _INTERVAL_    |  |        _ACTIVE_    |
   |_DETAIL(_|__________|_)_|  |_LIST(_|________|_)_|
             |_*________|               |_*______|
                    _ACTIVE_
>__ ________ __(__|________|__)______________________________________>
   |_LSTATS_|     |_*______|
>__ _________________________________ _______________________________>
   |          _*_________________    |
   |          | <_,______________  |    |
   |_DBNAME(_|___database-name_|_|_)_|
             |_name1:name2_______|
             |_name*_____________|
>__ ________________________________ __ ___________________ _________>
   |            _*_______________    |  |_GBPDEP(_ _YES_ _)_|
   |           | <_,___________   |    |            |_NO__|
   |_SPACENAM(_|___space-name_|_|_)_|
               |_name1:name2____|
               |_name*__________|
>__ _____________________ ___________________________________________>
   |_CASTOWNR(_ _YES_ _)_|
               |_NO__|
```

```
-DISPLAY DATABASE
```

```
Displays status information about DB2 databases

                          <_,_____________
>>__DISPLAY DATABASE__(_ ___database-name_|_____ _)__________________>
                        |_*_____________________|
                        |_dbname1:dbname2_______|
                        |_dbname*_______________|
                        |_*dbname_______________|
                        |_*dbname*______________|
                        |_*dbstring1*dbstring2*_|
>__ _ __________ ____________________________________________________>
   | |_USE______|
   | |_CLAIMERS_|
   | |_LOCKS____|
   | |_LPL______|
   | |_WEPR_____|
   |                <_,__________
   |_SPACENAM__(__ ___space-name_|______________ __)_________________>
   |          |   |_*___________________________|
   |          |   |_spacename1:spacename2_______|
   |          |   |_spacename*__________________|
   |          |   |_*spacename__________________|
   |          |   |_*spacename*_________________|
   |          |   |_*spacestring1*spacestring2*_|
   |          |___ __________ __ ______ _____________________________>
   |              |_USE______|  |_ONLY_|
   |              |_CLAIMERS_|
   |              |_LOCKS____|
   |              |_LPL______|
   |              |_WEPR_____|
   |_ONLY____________________________________________________________>
>__ _______________________________ _________________________________>
   |         <_,_______            |
   |_PART(_ ___integer_|_______ _)_|
           |_integer1:integer2_|
>__ ______________________ __ _______ __ ________ ___________________>
   |         _5Ø______    |  |_AFTER_|  |_ACTIVE_|
   |_LIMIT(_|_integer_|_)_|
            |_*_______|
>__ _______________________________ _________________________________>
   |_RESTRICT__(__ ___________ __)_|
                  | <_,_____  |
                  |___ACHKP_|_|
                  |_CHKP______|
                  |_COPY______|
                  |_GRECP_____|
                  |_LPL_______|
                  |_RBDP______|
                  |_RECP______|
                  |_REORP_____|
                  |_RO________|
                  |_STOP______|
                  |_UT________|
                  |_UTRO______|
                  |_UTRW______|
                  |_UTUT______|
                  |_UT*_______|
                  |_WEPR______|
>__ _______________________________ _________________________________>
   |_ADVISORY__(__ ___________ __)_|
                  | <_,_____  |
                  |___ICOPY_|_|
                  |_AUXW______|

-DISPLAY DDF
Display the distributed data facility

>>__DISPLAY DDF__ ________ __________________________________________>
                 |_DETAIL_|

-DISPLAY FUNCTION SPECIFIC
Displays statistics about external user-defined functions

>>__DISPLAY FUNCTION SPECIFIC________________________________________>
    _(*.*)_____________________________________
>__|___________________________________________|_____________________>
   |   <_,_________________________________    |
   |_(___ _schema.specific-function-name_ _|_)_|
         |_schema.partial-name*__________|

-DISPLAY GROUP
Displays information about the data sharing group to which a DB2
subsystem belongs

>>__DISPLAY GROUP__ ________ ________________________________________>
                   |_DETAIL_|

-DISPLAY GROUPBUFFERPOOL
Displays status information about DB2 group buffer pools

>>__DISPLAY GROUPBUFFERPOOL__ ________________________________ ______>
                             |  _(*)________________________  |
                             | |   <_,___________________   | |
                             |_|_(___ _gbpname________ _|_)_|_|
                                     |_structure-name_|
>__ _________________________ __ _______________________________ ____>
   |           _*_______     |  |_MDETAIL__ __________________ _|
   |_TYPE__(__|_GCONN___|__)_|             |    _INTERVAL_    |
              |_MCONN___|                  |_(_|__________|_)_|
              |_NOCACHE_|                      |_*________|
>__ _______________________________ __ _________________________ ____>
   |_GDETAIL__ _________________ _|   |                _NO__     |
              |    _INTERVAL_     |   |_CONNLIST__(__|_YES_|__)_|
              |_(_|__________|_)_|
                  |_*________|
```

```
-DISPLAY LOCATION
Displays status information about distributed threads
```

```
                       _(*)______________________________
>>__DISPLAY  LOCATION__|_________________________________|____________>
                      |    <_,______________________     |
                      |_(____ _location-name_____ _|__)_|
                             |_partial-location*_|
                             |_<luname>__________|
                             |_ipaddr____________|
>__ ________ ________________________________________________________>
   |_DETAIL_|
```

```
-DISPLAY LOG
Displays log information and status of the offload task
```

```
>>__DISPLAY  LOG______________________________________________________>
```

```
-DISPLAY PROCEDURE
Displays status information about stored procedures
```

```
                        _(*.*)_____________________________
>>__DISPLAY  PROCEDURE__|___________________________________|_________>
                       |   <_,_________________________     |
                       |_(___ _schema.procedure-name_ _|_)_|
                             |_schema.partial-name*__|
                             |_procedure-name________|
                             |_partial-name*_________|
>__ _______________________ _________________________________________>
   |           _MEMBER_    |
   |_SCOPE__(_|________|_)_|
              |_GROUP__|
```

```
-DISPLAY RLIMIT
Displays status information about the resource limit facility
(governor)
```

```
>>__DISPLAY  RLIMIT___________________________________________________>
```

```
-DISPLAY THREAD
Displays information about DB2 threads
```

```
>>__DISPLAY THREAD___ ___________________________________ ___________>
                     |    <_,_______________________     |
                     |_(____ _connection-name_____ _|__)_|
                     |      |_partial-connection*_|      |
                     |_(__*__)___________________________|
>__ _______________________ __________________________________________>
   |        _ACTIVE____    |
   |_TYPE(_|_INDOUBT___|_)_|
           |_*_________|
           |_INACTIVE__|
           |_POSTPONED_|
>__ _______________________________________________ __ _______ _______>
   |               <_,_____________________        |  |_DETAIL_|
   |_ _LOCATION(_ ___ _location-name_____ _|_ _)_ _|
     |           |   |_partial-location*_|   |   |
     |           |_*_________________________|   |
     |        <_,________________                |
     |_LUWID(__ _luwid__________ |_)_____________|
               |_partial-luwid*_|
               |_token__________|
>__ _____________________________ ____________________________________>
   |             <_,______       |
   |_RRSURID_ _(__rrs-urid|_)_ __|
             |_( * )__________|
```

-DISPLAY TRACE
Displays information about DB2 traces

```
                     _(__*__)_______
>>__DISPLAY  TRACE__|_______________|_________________________________>
                    |_(__PERFM__)___|
                    |_(__ACCTG__)___|
                    |_(__STAT__)____|
                    |_(__AUDIT__)___|
                    |_(__MONITOR__)_|
>__  _________________________________________________________________>
   |      <_,______
   |_DEST(__ _GTF_ |_)________________________________________________>
            |_SMF_|
            |_SRV_|
            |_OPn_|
>__  _________________________________________________________________>
   |        _*_____________              _*____________
   |       | <_,__________ |            | <_,________  |
   |_ _PLAN(_____plan-name_|___)__AUTHID(_____auth-id_|__)____________>
      |      _*____________          _*____________
      |     | <_,_________ |        | <_,_________ |
      |_CLASS(_____integer_|___)__TNO(______integer_|__)______________>
      |            _*_________________
      |           | <_,_____________   |
      |_LOCATION(_|___location-name_|_|_)_____________________________>
                  |_<luname>__________|
                 |_nnn.nnn.nnn.nnn___|
>__ _____________________ __ _________________ ______________________>
   |_DETAIL(output-type)_|  |_COMMENT(string)_|
```

-DISPLAY UTILITY
Displays status information about a DB2 utility

```
>>__DISPLAY UTILITY__(_ _utility-id__________ _)_____________________>
                       |_partial-utility-id*_|
                       |_*___________________|
>__ _________________________ _______________________________________>
   |         <_,_________    |
   |_MEMBER(__member-name|_)_|
```

-MODIFY TRACE
Changes the IFCIDs (trace events) associated with a particular active
trace

```
                                           _*___________
                                          | <_,__________ |
>>__MODIFY TRACE__(_ _PERFM___ _)__CLASS(_|___integer_|_|_)__________>
                    |_ACCTG___|
                    |_STAT____|
                    |_AUDIT___|
                    |_MONITOR_|
                         _*_______________
                        | <_,_____________ |
>__TNO(integer)__IFCID(_|____ifcid_nbr__|_|_)________________________>
>__ _________________ _______________________________________________>
   |_COMMENT(string)_|
```

-RECOVER BSDS
Reestablishes dual bootstrap datasets

```
>>__RECOVER  BSDS____________________________________________________>
```

-RECOVER INDOUBT
Recovers threads left indoubt

```
>>__RECOVER INDOUBT__ ___________________ __ACTION(_ _COMMIT_ _)_____>
                     |_(connection-name)_|          |_ABORT__|
           <_,______________
>__ _ID(_ ___correlation-id_|_ _)_ __________________________________>
   |     |_*__________________|   |
   |      <_,__________           |
   |_NID(___network-id_|_)________|
   |        <_,_______            |
   |_LUWID(__ _luwid_ |_)_________|
             |_token_|
-RECOVER POSTPONED
Completes back-out processing for units of recovery left incomplete
during an earlier restart

>>__RECOVER POSTPONED__ ________ ____________________________________>
                       |_CANCEL_|

-RESET GENERICLU
Purges information stored by VTAM in the coupling facility

                         <_,________________
>>__RESET GENERICLU__ _(___ _luname_______ _|_)_ ____________________>
                     |     |_netid.luname_|     |
                     |_(*)______________________|

-RESET INDOUBT
Purges information displayed in the indoubt thread report generated by
the -DISPLAY THREAD command

>>__RESET  INDOUBT___________________________________________________>
               <_,______
>__ _LUNAME(_ ___luname_|_ _)_ _______ ________________ _____________>
   |         |_*__________|   |_FORCE_|                |
   |           <_,_____________                        |
   |_LOCATION(___location-name_|_)_____________________|
   |           <_,____________________                 |
   |_IPADDR(_ ___nnn.nnn.nnn.nnn:port_|_ _)_ _______ __|
   |         |_*________________________|   |_FORCE_|  |
   |        <_,_______                                 |
   |_LUWID(__ _luwid_ |_)_ _________________________ __|
             |_token_|    |_LOCATION(location-name)_|

-SET ARCHIVE
Controls the allocation of tape units and the deallocation time of the
tape units for archive log processing

>>__SET ARCHIVE__ __ ____________________ ____ __ ___________________>
                 |  |_COUNT(__integer__)_|    |  |
                 |____TIME(_ _minutes__ _)____| |
                 |          |_,seconds_|   |
                 |          |_1440_____|   |
                 |          |_NOLIMIT__|   |
                 |____DEFAULT____________________|

-SET LOG
Modifies the checkpoint frequency

>>__SET LOG__ _LOGLOAD(integer)_______ ______________________________>
             |_CHKTIME__(__integer__)_|
             |_SUSPEND________________|
             |_RESUME_________________|

-SET SYSPARM
Changes subsystem parameters online

>>__SET SYSPARM__ _LOAD(_ _DSNZPARM_________ _)_ ____________________>
                 |       |_load-module-name_|   |
                 |_RELOAD_______________________|
                 |_STARTUP______________________|

-START DATABASE
Makes the specified database available for use

                        <_,_____________
>>__START DATABASE__(_ ___database-name_|_____ _)____________________>
                      |_*_____________________|
                      |_dbname1:dbname2_______|
                      |_dbname*_______________|
                      |_*dbname_______________|
                      |_*dbname*______________|
                      |_*dbstring1*dbstring2*_|
>__ _________________________________________________________________>
   |             <_,__________
   |_SPACENAM(_ ___space-name_|______________ _)
               |_*___________________________|
               |_spacename1:spacename2_______|
               |_spacename*__________________|
               |_*spacename__________________|
               |_*spacename*_________________|
               |_*spacestring1*spacestring2*_|
>______________ _______________________________ _____________________>
               |         <_,________           |
               |_PART(_ ___integer_|________)__|
               |_integer1:integer2_|
>__ ______________________ __________________________________________>
   |          _RW____    |
   |_ACCESS(_|_RO____|_)_|
             |_UT____|
             |_FORCE_|

-START DB2
Initializes the DB2 subsystem (can be issued only from an MVS console)

>>__START DB2__ _________________________ ___________________________>
               |        _DSNZPARM____    |
               |_PARM(_|_module name_|_)_|
>__ _____________________ __ __________________ _____________________>
   |          _*_____    |  |         _NO__    |
   |_ACCESS(_|_MAINT_|_)_|  |_LIGHT(_|_YES_|_)_|
>__ ________________________ __ ________________________ ____________>
```

```
   |_MSTR(jcl-substitution)_|  |_DBM1(jcl-substitution)_|
>__ _________________________ _______________________________________>
   |_DIST(jcl-substitution)_|
```

```
-START DDF
Starts the distributed data facility

>>__START DDF_______________________________________________________>
```

```
-START FUNCTION SPECIFIC
Activates an external function that is stopped

>>__START FUNCTION SPECIFIC_________________________________________>
    _(*.*)_____________________________________
>__|___________________________________________|_____________________>
   |   <_,_________________________________    |
   |_(___ _schema.specific-function-name_ _|_)_|
         |_schema.partial-name*__________|
```

```
-START PROCEDURE
Activates the definition of stopped or cached stored procedures

                      _(*.*)_____________________________
>>__START PROCEDURE__|___________________________________|___________>
                     |   <_,_________________________    |
                     |_(___ _schema.procedure-name_ _|_)_|
                           |_schema.partial-name*__|
                           |_procedure-name________|
                           |_partial-name*_________|
```

```
-START RLIMIT
Starts the resource limit facility (governor)

>>__START RLIMIT__ _______ __________________________________________>
                  |_ID=id_|
```

```
-START TRACE
Initiates DB2 trace activity

>>__START TRACE__(_ _PERFM___ _)_____________________________________>
                   |_ACCTG___|
                   |_STAT____|
                   |_AUDIT___|
                   |_MONITOR_|
>___  _______________________________________________________________>
    |          <_,_____
    | >__DEST(__ _GTF_ |_)___________________________________________>
    |           |_SMF_|
    |           |_SRV_|
    |           |_OPn_|
    |           |_OPX_|
>___ ________________________________________________________________>
    |           _*_____________              _*____________________
    |          | <_,_________  |            | <_,________________  |
    | >__PLAN(_|___plan-name_|_|_)__AUTHID(_|___authorization-id_|_|)>
    |            _*___________             _*_________
    |           | <_,_______  |           | <_,_____  |
    | >__CLASS(_|___integer_|_|_)__IFCID(_|___ifcid_|_|_)____________>
    |              _*________           <_,______________
    | >__BUFSIZE(_|_k_bytes_|_)__TDATA(__ _CORRELATION_ |_)__________>
    |                                    |_TRACE_______|
    |                                    |_CPU_________|
    |                                    |_DISTRIBUTED_|
    | >__LOCATION(_ _*_________________ _)___________________________>
    |              | <_,______________ |
    |              |___location-name_|_|
    |              |_<luname>__________|
    |              |_ipaddr____________|
>___ _________________ ______________________________________________>
    |_COMMENT(string)_|

-STOP DATABASE
Makes specified databases unavailable for applications
```

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

Nikola Lazovic
DB2 System Administrator
Postal Savings Bank (Serbia and Montenegro)

November 2002 – October 2005 index

Items below are references to articles that have appeared in *DB2 Update* since issue 121, November 2002. References show the issue number followed by the page number(s). Subscribers can download copies of all issues in Acrobat PDF format from Xephon's Web site.

DB2 news

Mainstar Software has announced Version 1.8 of MS/VCR, its mirroring solutions suite. MS/VCR helps companies gain use of their online or batch data by augmenting fast data replication tools such as FlashCopy and SnapShot. The tool also uses splits of continuous mirroring tools to clone data either offline or online.

The product supports EMC TimeFinder, IBM PPRC, HDS ShadowImage, Softek Replicator (formerly TDMF), and Innovation's FDRPAS. The cloned data can then be accessed from the same MVS system. MS/VCR cloning reduces production downtime and the costs associated with cloning with traditional tools.

The automation in MS/VCR has been enhanced, particularly in the online DB2 area, to help resource-strapped data centres increase productivity. Support for DB2 Version 8 was formally released in MS/VCR 1.8. There's also support now to clone multiple DB2 subsystems that share the same disk within the same copy, and rename commands to lessen complexity.

For further information contact:
URL: www.mainstar.com/pdf/009-0123_VCR0108_PR.pdf.

* * *

ClearStory Systems has announced Version 2.1 of the Radiant Enterprise Media Server. The new release builds on the product's enterprise DAM capability, now offering integration with DB2 Content Manager and expanded standard database support to include DB2 Universal Database, Microsoft SQL Server, and additional versions of Oracle 9i.

Radiant EMS is a J2EE system, which manages rich media assets (digital video, graphics, multi-media presentations, and compound documents). Integration with DB2 Content Manager allows companies to leverage their existing ECM infrastructure, centralizing all enterprise content, while incorporating high value rich media assets into enterprise business processes.

For further information contact:
URL: www.clearstorysystems.com/company/news-details.asp?id=244.

* * *

Princeton Softech has announced Release 5.4 of Archive for DB2 and Relational Tools, which enables companies to implement Information Governance strategies, improve performance, and mitigate business risks.

Release 5.4 of Archive for DB2 and Relational Tools allows users to deploy their applications, data, and storage to meet evolving business needs. It also offers integration with IBM TotalStorage DR550; facilitating the segregation of DB2 application data for storage in an immutable format for long-term retention. Additional product enhancements continue to optimize batch performance and facilitate the discovery of archived data, say the company.

For further information contact:
URL: www.neonesoft.com/product_bind.htmlwww.princetonsoftech.com/news/press/AR4DB2-RT4DB25.4.asp.

* * *

Move2open has announced the general availability of its automated suite of conversion tools. Move2open was established to focus on legacy transformation, and in particular the needs of companies converting from CA-IDEAL to open languages such as COBOL and Java, and migrating from CA-DATACOM to open databases such as DB2, Oracle, CA-Ingres, and SQL Server.

For further information contact:
URL: www.move2open.com.