69

DB2

update

July 1998

In this issue

3 DB2 security tip – almost a perfect crime

7 Accessing directory information – revisited

8 Collecting accounting information – part 2

16 Call for papers

17 How to read on-line DB2 statistics

20 Deadlock/timeout reporting tool

29 Rebind and convert plans and packages – part 2

48 DB2 news

DB2 Update

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

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

Contributions
Articles published in *DB2 Update* are paid for at the rate of £170 ($250) per 1000 words and £90 ($140) per 100 lines of code for original material. To find out more about contributing an article, without any obligation, please contact us at any of the addresses above and we will send you a copy of our *Notes for Contributors*.

DB2 Update on-line
Code from *DB2 Update* can be downloaded from our Web site at http://www.xephon.com; you will need the user-id shown on your address label.

Editor
Trevor Eddolls

Disclaimer
Readers are cautioned that, although the information in this journal is presented in good faith, neither Xephon nor the organizations or individuals that supplied information in this journal give any warranty or make any representations as to the accuracy of the material it contains. Neither Xephon nor the contributing organizations or individuals accept any liability of any kind howsoever arising out of the use of such material. Readers should satisfy themselves as to the correctness and relevance to their circumstances of all advice, information, code, JCL, and other contents of this journal before making any use of it.

Subscriptions and back-issues
A year's subscription to *DB2 Update*, comprising twelve monthly issues, costs £245.00 in the UK; $365.00 in the USA and Canada; £251.00 in Europe; £257.00 in Australasia and Japan; and £255.50 elsewhere. In all cases the price includes postage. Individual issues, starting with the January 1994 issue, are available separately to subscribers for £21.00 ($31.00) each including postage.

Printed in England.

DB2 security tip – almost a perfect crime

I went through an agonizing 'soul searching' exercise before I decided to write this article.

The issue: how is the security of computer data at an organization best realized? Is it realized through education, security awareness, and proliferation of knowledge, or is it best served by secretive suppression of knowledge?

The answer to the above issue, at least in my mind, is that the security of the computer data at an organization is enhanced through education and security awareness.

If I were a security officer at an organization, I would feel more secure if I knew the organization's security weak points and security exposures – so that I could watch them, guard them, and minimize their impact on my organization.

This type of thinking is particularly true when dealing with information systems or computer software that can be purchased, studied, and used.

DB2 DBMS is not an exception to this kind of thinking. DB2 MVS is one of the best DBMSs in the world. Information about it is readily available through IBM manuals, books, CD-ROMs, third-party vendors, magazines, Internet Web pages, regional and international user groups, seminars and conferences, etc.

DB2 is also rapidly becoming the 'fat server' choice of the information industry, particularly in the distributed data warehouse and datamarts environments.

Exposing security entry points in DB2 does not mean that the DBMS product is flawed; rather it exposes an area that a DB2 security administrator should be aware of and should monitor diligently on a regular basis.

With these security thoughts in my mind, I decided to write this article about a purely hypothetical, fictitious organization whose computer's

financial data has been violated. The intention is to educate and promote security interests and awareness in DB2 DBMS.

THE SETTING

Our hypothetical organization has DB2 MVS to house its strategic computer financial data.

Database administration in this hypothetical organization is divided into two areas:

- System programmers, who install DB2 subsystems, control its system parameters, and maintain its system performance.
- Application DBAs, who translate the entity relational model of applications into a physical schema and into DB2 physical tables. Application DBAs also advise developers on the best application design to allow the DB2 Optimizer to choose the optimum access path. Application DBAs also control DB2 internal security through GRANTs and REVOKEs of system and database privileges to users and production IDs.

In this fictitious organization, the MVS machine is IPLed once a week.

MANIFESTATION OF THE PROBLEM

The financial department of this organization informs the security department that critical financial data of the organization has been violated

The security department starts investigating the allegation. The security department starts examining QMF reports compiled from the DB2 catalog to see who has update access to these critical financial DB2 tables. The QMF/DB2 catalog reports list DBA IDs and a restricted number of bonded user-ids who are allowed update access to these sensitive DB2 tables.

The listed DBAs and users are investigated thoroughly and are eliminated from suspicion in this criminal affair.

The security department is stupefied and constantly asks the following question: “We have interviewed every person whose ID appears on the QMF/DB2 catalog report who has access to these sensitive tables, and have eliminated them as suspects. So how on earth can somebody go into these sensitive DB2 tables and violate them without being INSTALL SYSADM1 or INSTALL SYSADM2, or without being recorded in the DB2 catalog?”.

The problem was not solved and the case was not closed. The security department is nervous but alert.

Six months later the same criminal violation occurs with the same bewildering result!

FURTHER DEVELOPMENT

We all know that in a civilized society crime is never a substitute for honest decent living, and a criminal will, sooner or later, be caught and apprehended!

The continuation to our story took place during Christmas week. It also happened that our criminal persona has decided to act out his crime again in this Christmas week.

A legitimate user came to the application DBA on the first day of the week and requested data access to the same DB2 financial tables that were violated twice before.

The DBA was so busy thinking of Christmas shopping that he forgot to GRANT the legitimate user the requested GRANT.

A few days later in the Christmas week, the legitimate user sent a thank you note for the DBA for his prompt attention to the user request!

Aha, bingo! In reality the DBA did not GRANT the legitimate user the access he requested because the DBA was busy Christmas shopping and forgot to take care of the user’s request. So how is the user able to access this sensitive data without GRANTs?

The DBA informed his manager and the DBA manager alerted security.

The security personnel, the DBA, and his manager puzzled over this situation, and asked themselves the following questions: “There is something happening in this organization that renders useless the GRANTs and REVOKEs in the DB2 catalog and the QMF report produced from the catalog. What could that be? Under what circumstances could that happen?”

The shift in thinking was directed to the DSNZPARM values and its meaning. The DBA interrogated the DSNZPARM values one by one through an on-line DB2 monitor. Eureka! There is a DSNZPARM parameter specified in the DB2 installation panel DSNTIPP that deals with security matters. The macro name is DSN6SPRM. The name of the parm is AUTH. This parm was set to the value NO. This parm should always be set to the value YES. When this parm is changed to the value NO, it will render all the GRANTs and REVOKEs in the entire DB2 catalog and any report produced from the DB2 catalog totally useless. Actually changing the value of this parm to NO is akin to GRANTing all accesses on everything in the entire DB2 subsystem and making it PUBLIC. DB2 will not perform any authorization checking whatsoever. Thus the entire financial data of this organization that is stored in this DB2 subsystem is at jeopardy.

Security personnel went to the system programmer who controlled DSNZPARM values. The system programmer was interrogated and confessed promptly to his crime! Every time he wanted to act out his crime, he changed the value of this DSNZPARM value to NO discreetly and without a CHANGE MANAGEMENT RECORD. Once the weekly IPL occurs and DB2 comes up with the unsuspected covert DSNZPARM module, the system programmer commits his crime and, after that, resets the value of DSNZPARM back to YES. This will take effect at the next IPL. Nobody knows, notices, or suspects anything. Even if the DSNZPARM values were interrogated the week after the crime it would reflect the proper DSNZPARM value of YES because our system programmer resets the value of the parameter to YES immediately after he commits his crime. The entire process is almost bordering on the ‘perfect’ crime, if there is such a thing as a ‘perfect’ crime – except, there was one single act of kindness

by a legitimate user who sent an innocent thank-you note to our security conscious DBA that started him thinking about the situation.

MORAL OF THE STORY

The moral of the story is that one should monitor and control the changes in DSNZPARM values on a regular basis. This is particularly important when an organization possesses a third-party product such as Opertune from BMC Software that conveniently and dynamically manipulates and changes the DSNZPARM values on-the-fly without bouncing DB2 down and up.

Nick Nur
Senior DBA (Canada)

Accessing directory information – revisited

Issue 13 of DB2 Update, *November 1993, contained an article by Alberto Grassi called* Accessing directory information. *In Issue 66, April 1998, we published a request for help from J Naumovic. He asked why the technique, which worked perfectly with DB2 Version 4, did not work with DB2 Version 5. Jeff Schade sent us the following e-mail.*

The question is why does the technique from Issue 13 not work in Version 5 of DB2. I am sure you will hear from a lot of people about this one. The problem is that the DSNDB01.SYSLGRNG tablespace was renamed to DSNDB01.SYSLGRNX as part of the Version 4 migration.

Most shops have since dropped the old table once DB2 Version 4 was considered stable. I have not gone so far as to check and see if the table actually changed but hopefully this code can be updated accordingly.

I hope this helps.

Jeff Schade (USA)

Collecting accounting information – part 2

This month we conclude the code for SMF101P2, which produces a summary report based on the detail data extracted by SMF101P1 (published last month).

SMF101P2

```
//YOUR JOB CARD HERE
//ASM     EXEC PGM=ASMA90,PARM='OBJECT,NODECK'
//SYSLIB   DD  DSN=SYS1.MACLIB,DISP=SHR
//         DD  DSN=SSID.DB2.DSNMACS,DISP=SHR
//         DD  DSN=SSID.DB2.DSNSAMP,DISP=SHR
//SYSLIN   DD  DSN=&&LOADSET,DISP=(MOD,PASS),UNIT=SYSDA,
//             SPACE=(800,(1000,1000)),DCB=(BLKSIZE=800)
//SYSPRINT DD  SYSOUT=*
//SYSUDUMP DD  SYSOUT=*
//SYSUT1   DD  SPACE=(800,(1000,1000),,,ROUND),UNIT=SYSDA
//SYSIN    DD  *
*****************************************************************
* SMF101P2 :   SUMMARIZE SMF ACCOUNTING RECORDS
*
* FUNCTION :   SUMMARIZE THE RECORDS GENERATED BY THE
*              SMF101P1 PROGRAM.  THIS INVOLVES ACCUMULATING
*              THE FIELDS OF EACH PLAN AND DIVIDING BY THE
*              THREAD COUNT TO GET THE AVERAGE, WHICH IS THEN
*              PRINTED IN A REPORT.
*
* INPUT    :   SMFSUMM DD - SORTED SMF ACCOUNTING RECORDS
*
* OUTPUT   :   SYSPRINT DD - REPORT OUTPUT
*
* PROCESS FLOW:
*    INITIALIZE VALUES
*    PRINT HEADER LINES
*    READ SORTED RECORDS UNTIL EOF
*       FOR EACH PLAN ACCUMULATE THE VALUES
*                     DIVIDE BY THE THREAD COUNT
*                     DISPLAY COMPUTED AVERAGE TO A REPORT
*    FINALIZATIONS
*****************************************************************
SMF101P2 CSECT
         SAVE  (14,12),,SMF101P2-&SYSDATE
         LR    R11,R15
         USING SMF101P2,R11,R12
```

```
         GETMAIN RU,LV=WORKLEN    GET WORK AREA
         ST    R13,4(,R1)
         ST    R1,8(,R13)
         LR    R13,R1
         USING WORKAREA,R13
         LA    R12,2Ø48(,R11)     SET-UP SECOND BASE REGISTER
         LA    R12,2Ø48(,R12)     SET-UP CONTINUED
         B     INITIALZ
**       I N I T I A L I Z E
INITIALZ DS    ØD
         BAL   R14,INITRTN        DO INITIALIZE VALUES
         BAL   R14,HDRLRTN        DO PRINT HEADER LINES
         USING SMFSUMMD,R3
**       M A I N L I N E
MAINLINE DS    ØH
         BAL   R14,READRTN             DO READ SMF RECORD
         CLI   SWEOF,X'FF'             IF END OF FILE?
         BE    FINALIZE                Y. THEN END-IT
         CLC   I1ACPLPK,WPRVPLPK       IF PLAN THE SAME AS PREVIOUS?
         BE    ACCURTN                 Y. DO ACCUMULATE EM
         CLC   WPRVPLPK,=26CL1' '      IF NO PREVIOUS PLAN
         BE    ACCURTN                 Y. DO ACCUMULATE FIRST REC
         BAL   R14,FMTØRTN             N. PRINT LINE
         BAL   R14,RESERTN                RESET EM
         B     MAINLINE                READ NEXT
**       F I N A L I Z E
FINALIZE DS    ØH
         BAL   R14,FMTØRTN             PRINT LINE
         BAL   R14,CLOSRTN
         CLC   RETCODE,=H'16'
         BE    RETCOD16
         LR    R1,R13                 LOAD A SAVEAREA FOR FREEMAIN
         L     R13,4(,R13)            LOAD A CALLERS SAVEAREA
         FREEMAIN RU,LV=WORKLEN,A=(1) FREE WORK AREA
         RETURN (14,12),RC=Ø          RETURN TO OS RC=Ø
RETCOD16 DS    ØH
         LR    R1,R13                 LOAD A SAVEAREA FOR FREEMAIN
         L     R13,4(,R13)            LOAD A CALLERS SAVEAREA
         FREEMAIN RU,LV=WORKLEN,A=(1) FREE WORK AREA
         RETURN (14,12),RC=16         RETURN TO OS RC=16
*****************************************************
**       S U B R O U T I N E S
*****************************************************
** ACCURTN -   ACCUMULATE VALUES
ACCURTN  DS    ØH
         ST    R14,ACCUSAVE
         AP    WORKATHD,=P'1'
         AP    WORKSCT,I1ACSCT
         AP    WORKTJST,I1ACTJST
```

```
         AP    WORKAWTI,I1ACAWTI
         AP    WORKAWTL,I1ACAWTL
         AP    WORKAWTR,I1ACAWTR
         AP    WORKAWTE,I1ACAWTE
         MVC   WPRVPLPK,I1ACPLPK
         B     MAINLINE
ACCUEXIT EQU   *
         L     R14,ACCUSAVE
         BR    R14
ACCUSAVE DS    F'Ø'
*************************************************
** RESERTN -   RESET ACCUMULATORS
RESERTN  DS    ØH
         ST    R14,RESESAVE
         ZAP   WORKATHD,=P'1'
         ZAP   WORKSCT,I1ACSCT
         ZAP   WORKTJST,I1ACTJST
         ZAP   WORKAWTI,I1ACAWTI
         ZAP   WORKAWTL,I1ACAWTL
         ZAP   WORKAWTR,I1ACAWTR
         ZAP   WORKAWTE,I1ACAWTE
         MVC   WPRVPLPK,I1ACPLPK
RESEEXIT EQU   *
         L     R14,RESESAVE
         BR    R14
RESESAVE DS    F'Ø'
*************************************************
** FMTØRTN     FORMAT DETAIL LINE
FMTØRTN  DS    ØH
         ST    R14,FMTØSAVE
         MVC   O1REPORT(132),=132CL1' '       CLEAR OUTPUT LINE
FMTØØ1Ø  MVC   O1PLAN,WPRVPLPK           PLUG PLAN TO OUTPUT
         MVC   O1PKID,WPRVPLPK+8         PLUG PLAN TO OUTPUT
         MVC   O1TYPE,I1ACTYPE
         MVC   O1ACTHD,=X'2Ø2Ø2Ø2Ø2Ø2121Ø'
         ED    O1ACTHD,WORKATHD        PLUG THREAD COUNT
         ZAP   WORKDIVI,WORKSCT
         BAL   R14,FMT1RTN
         MVC   O1ACSCT,WORKFMT1
         ZAP   WORKDIVI,WORKTJST
         BAL   R14,FMT1RTN
         MVC   O1ACTJST,WORKFMT1
         ZAP   WORKDIVI,WORKAWTI
         BAL   R14,FMT1RTN
         MVC   O1ACAWTI,WORKFMT1
         ZAP   WORKDIVI,WORKAWTL
         BAL   R14,FMT1RTN
         MVC   O1ACAWTL,WORKFMT1
         ZAP   WORKDIVI,WORKAWTR
```

```
         BAL   R14,FMT1RTN
         MVC   O1ACAWTR,WORKFMT1
         ZAP   WORKDIVI,WORKAWTE
         BAL   R14,FMT1RTN
         MVC   O1ACAWTE,WORKFMT1
         MVC   PRNTLINE,O1REPORT       MOVE PRINT
         BAL   R14,WRITRTN             DO PRINT LINE
FMTØEXIT EQU   *
         L     R14,FMTØSAVE
         BR    R14
FMTØSAVE DS    F'Ø'
*************************************************
** FMT1RTN     FORMAT OUTPUT AND ROUND-UP
FMT1RTN  DS    ØH
         ST    R14,FMT1SAVE
         SRP   WORKDIVI,1,Ø            SHIFT LEFT 1
         DP    WORKDIVI,WORKATHD
         SRP   WORKDIVI(8),63,5        SHIFT RIGHT 1 AND ROUND .5
         MVC   WORKFMT1,=X'2020202020202120 4B202020'
         ED    WORKFMT1,WORKDIVI+2
FMT1EXIT EQU   *
         L     R14,FMT1SAVE
         BR    R14
FMT1SAVE DS    F'Ø'
*************************************************
** INITRTN     INITIALIZE VALUES
INITRTN  DS    ØH
         ST    R14,INITSAVE
         OPEN  (SMFSUMM,INPUT)
         OPEN  (SYSPRINT,OUTPUT)
         OPEN  (SNAPDUMP,OUTPUT)
         ZAP   WORKATHD,=P'Ø'
         ZAP   WORKSCT,=P'Ø'
         ZAP   WORKTJST,=P'Ø'
         ZAP   WORKAWTI,=P'Ø'
         ZAP   WORKAWTL,=P'Ø'
         ZAP   WORKAWTR,=P'Ø'
         ZAP   WORKAWTE,=P'Ø'
         MVC   WPRVPLPK,=26CL1' '
INITEXIT EQU   *
         L     R14,INITSAVE
         BR    R14
INITSAVE DS    F'Ø'
*************************************************
** HDRLRTN - PRINT HEADER LINES
HDRLRTN  DS    ØH
         ST    R14,HDRLSAVE
         MVC   PRNTLINE(132),STITLEL1
*                                    GET CURRENT TIME AND DATE
```

```
         TIME  DEC
         SRL   R0,4                   SHIFT RIGHT LOGICAL
         O     R0,=X'0000000F'        OR VALUE TO SIGN
         ST    R0,TIMEPD              SAVE TIME VALUE
         ED    TIMEPATT(10),TIMEPD    EDIT TIME VALUE
         MVC   TIMECURR(8),TIMEPATT+2 MOVE EDIT TIME TO CL8
         ST    R1,DATEPD              SAVE DATE VALUE
         ED    DATEPATT(8),DATEPD     EDIT DATE VALUE
         MVC   DATECURR(6),DATEPATT+2 MOVE EDIT DATE TO CL8
         MVC   PRNTLINE+124(8),TIMECURR   CURRENT TIME
         MVC   PRNTLINE+110(6),DATECURR   CURRENT DATE
         BAL   R14,WRITRTN
         MVC   PRNTLINE(132),SCOLHDR1
         BAL   R14,WRITRTN
         MVC   PRNTLINE(132),SCOLHDR2
         BAL   R14,WRITRTN
HDRLEXIT EQU   *
         L     R14,HDRLSAVE
         BR    R14
HDRLSAVE DS    F'0'
*****************************************************
** CLOSRTN     CLOSE FILES
CLOSRTN  DS    0H
         ST    R14,CLOSSAVE
         CLOSE SMFSUMM
         CLOSE SYSPRINT
         CLOSE SNAPDUMP
CLOSEXIT EQU   *
         L     R14,CLOSSAVE
         BR    R14
CLOSSAVE DS    F'0'
*****************************************************
** READRTN     READ INPUT CARD
READRTN  DS    0H
         ST    R14,READSAVE
         GET   SMFSUMM
         LR    R3,R1                   SAVE BUFFER AFTER READ
READEXIT EQU   *
         L     R14,READSAVE
         BR    R14
READEOF  EQU   *
         MVI   SWEOF,X'FF'
         B     READEXIT
READSAVE DS    F'0'
*****************************************************
** WRITRTN     WRITE FILES
WRITRTN  DS    0H
         ST    R14,WRITSAVE
         PUT   SYSPRINT,PRNTLINE
```

```
WRITEXIT EQU   *
         L     R14,WRITSAVE
         BR    R14
WRITSAVE DS    F'Ø'
*****************************************************
*****************************************************
** ABENDRTN    HANDLE SOFT ABEND
ABNDRTN  DS    ØH
**       SNAP  DCB=SNAPDUMP,STORAGE=(WSSTART,WSEND)
         MVC   RETCODE,=H'16'
         B     FINALIZE
*****************************************************
** SNAPRTN     DUMP A STORAGE FOR DEBUGGING
SNAPRTN  DS    ØH
         SNAP  DCB=SNAPDUMP,PDATA=REGS,LIST=(9)
         MVC   RETCODE,=H'16'
         B     FINALIZE
*****************************************************
**       DATA CONTROL BLOCKS
SMFSUMM  DCB   DSORG=PS,MACRF=GL,                                    X
               DDNAME=SMFSUMM,EODAD=READEOF,BFTEK=A
SYSPRINT DCB   DSORG=PS,RECFM=F,MACRF=(PM),LRECL=132,BLKSIZE=3Ø36,   X
               DDNAME=SYSPRINT
SNAPDUMP DCB   DSORG=PS,RECFM=VBA,MACRF=(W),LRECL=125,BLKSIZE=1632,  X
               DDNAME=SNAPDUMP
*****************************************************
**       STRING CONSTANTS
STITLEL1 DC    CL44'DB2 ACCOUNTING SMF RECORDS - PACKAGE/DBRM   '
         DC    CL44'                                            '
         DC    CL44'            RUN DATE: YY.DDD  TIME: HH:MM:SS'
SCOLHDR1 DC    CL44'PLANNAME PACKAGEID/DBRMNAME TYPE     COUNT  '
         DC    CL44'     ELAPSED       TCB-CPU      I/O-WAIT    '
         DC    CL44'  L/L-WAIT    ASYNC-READ    SYNCH-READ      '
SCOLHDR2 DC    CL44'-------- ------------------ ---- --------- -'
         DC    CL44'------------ ------------- ------------- ---'
         DC    CL44'---------- ------------- -------------      '
*****************************************************
**       OTHER STUFF
         DS    ØD
WORKCL16 DS    CL16
WORKPL8A DS    PLØ8
WORKPL8B DS    PLØ8
WPRVPLPK DS    CL26
         DS    CLØ2
WORKATHD DS    PLØ4'Ø'
WORKSCT  DS    PLØ8'Ø'
WORKTJST DS    PLØ8'Ø'
WORKAWTI DS    PLØ8'Ø'
WORKAWTL DS    PLØ8'Ø'
```

```
WORKAWTR DS    PL8'0'
WORKAWTE DS    PL8'0'
WORKDIVI DS    PL12'0'
         DS    CL4
WORKFMT1 DC    CL13' '
         DC    CL3' '
TIMEPD   DS    PL4                       PACKED TIME VALUE
TIMEPATT DC    X'402021207A20207A2020'   EDIT PATTERN FOR TIME
TIMECURR DS    CL8                       FINAL TIME VALUE
         DS    0D
DATEPD   DS    PL4                       PACKED DATE VALUE
DATEPATT DC    X'202120204B202020'       EDIT PATTERN FOR DATE
DATECURR DS    CL6                       FINAL DATE VALUE
SWEOF    DS    XL1
RETCODE  DS    H'0'
**       DSECTS
SMFSUMMD DSECT
I1RECORD DS    0CL80
I1ACPLPK DS    0CL26
I1ACPLAN DS    CL8
I1ACPKID DS    CL18
I1ACTYPE DS    CL2
I1ACSCT  DS    PL8                TOT ELAPSED TIME
I1ACTJST DS    PL8                TOT TCB CPU TIME
I1ACAWTI DS    PL8                TOT I/O WAIT TIME
I1ACAWTL DS    PL8                TOT LOCK/LATCH WAIT TIME
I1ACAWTR DS    PL8                TOT ASYNC READ
I1ACAWTE DS    PL8                TOT SYNCH READ
         DS    CL4
I1LENGTH EQU   *-I1RECORD          LENGTH OF RECORD
WORKAREA DSECT
SAVEAREA DC    18F'0'
DTIME    DS    F
DDATE    DS    F
WORK     DS    D
WORK2    DS    D
WORK3    DS    D
TDATE    DC    CL7' '
TTIME    DC    CL10' '
DAY      DC    XL1'00'            RELATIVE DAY OF THE WEEK
DATE     DC    C'  /  /  '        CURRENT DATE
TIME     DC    C'  :  :  '        CURRENT TIME
*************************************************
** OUTPUT RECORD
         DC    CL8'REPORT1:'       OUTPUT REPORT1 RECORD
O1REPORT DS    0H                  OUTPUT LINE 1
O1PLPK   DS    0CL26
O1PLAN   DS    CL8
         DS    CL1
O1PKID   DS    CL18
```

```
         DS    CLØ1
O1TYPE   DS    CLØ4
         DS    CLØ3
O1ACTHD  DS    CLØ7
         DS    CLØ2
O1ACSCT  DS    CL13
         DS    CLØ1
O1ACTJST DS    CL13
         DS    CLØ1
O1ACAWTI DS    CL13
         DS    CLØ1
O1ACAWTL DS    CL13
         DS    CLØ1
O1ACAWTR DS    CL13
         DS    CLØ1
O1ACAWTE DS    CL13
         DS    CLØ5
O1RECLEN EQU   *-O1REPORT           INPUT RECORD LENGTH
*************************************************
** PRINT LINE
PRNTLINE DS    ØCL132              OUTPUT DETAIL RECORD
         DS    CL132
WORKLEN  EQU   *-WORKAREA
RØ       EQU   Ø
R1       EQU   1
R2       EQU   2
R3       EQU   3
R4       EQU   4
R5       EQU   5
R6       EQU   6
R7       EQU   7
R8       EQU   8
R9       EQU   9
R1Ø      EQU   1Ø
R11      EQU   11
R12      EQU   12
R13      EQU   13
R14      EQU   14
R15      EQU   15
         END
//*
//LKED    EXEC PGM=IEWL,PARM='XREF',
//             COND=((4,LT,ASM))
//SYSLIB   DD  DISP=SHR,DSN=SSID.DB2.DSNLOAD
//SYSLIN   DD  DSN=&&LOADSET,DISP=(OLD,DELETE)
//         DD  DDNAME=SYSIN
//SYSLMOD DD DSN=MYTSOID.LOAD(SMF1Ø1P2),DISP=SHR
//SYSPRINT DD  SYSOUT=*
//SYSUDUMP DD  SYSOUT=*
//SYSUT1   DD  SPACE=(1Ø24,(5Ø,5Ø)),UNIT=SYSDA
```

```
//SYSIN   DD *
  NAME SMF101P2(R)
//*
//SORT   EXEC PGM=SORT,
//            COND=(4,LT)
//SORTIN   DD DSN=MYTSOID.SMFSUMM,DISP=SHR
//SORTOUT  DD DSN=MYTSOID.SMFSUMMS,DISP=OLD
//SORTWK01 DD UNIT=SYSDA,SPACE=(CYL,(5,20))
//SORTWK02 DD UNIT=SYSDA,SPACE=(CYL,(5,20))
//SORTWK03 DD UNIT=SYSDA,SPACE=(CYL,(5,20))
//SORTWK04 DD UNIT=SYSDA,SPACE=(CYL,(5,20))
//SYSUDUMP DD SYSOUT=*
//SYSOUT   DD SYSOUT=*
//SYSIN    DD *
 SORT FIELDS=(1,27,CH,A),FILSZ=E100000
//*
//RUN     EXEC PGM=SMF101P2,
//             COND=((4,LT,ASM),(4,LT,LKED))
//STEPLIB  DD  DISP=SHR,DSN=MYTSOID.LOAD
//SMFSUMM  DD  DSN=MYTSOID.SMFSUMMS,DISP=SHR
//SYSABEND DD  SYSOUT=*
//SYSUDUMP DD  SYSOUT=*
//SYSPRINT DD  SYSOUT=*
//SNAPDUMP DD  SYSOUT=*
//*
```

Larry Prestosa (USA)

Call for papers

Why not share your expertise and earn money at the same time? *DB2 Update* is looking for REXX EXECs, macros, program code, etc, that experienced DB2 users have written to make their life easier. We will publish them (after vetting by our expert panel) and send you a cheque when the article is published. Articles can be of any length and can be sent or e-mailed to Trevor Eddolls at any of the addresses shown on page 2. Why not call now for a free copy of our *Notes for contributors*?

How to read on-line DB2 statistics

The following program reads the RDS STATISTICS block and the SERVICE CONTROLLER STATISTICS block, and writes the blocks to a file. This we can read using REXX.

You can find the record map in the installation's DB2 macros library (DSNMACS) in the members DSNDQXST and DSNDQTST.

You can find the number of SQL commands executed since DB2 started, the number of datasets opened during the execution period, parallelism statistics, bind plan and package statistics, etc.

The program accesses the DB2 control blocks via the SSVT MVS control block.

The program is shown below:

```
DB2XSTAT TITLE 'PROGRAM READS STATISTICS DB2 CONTROL BLOCK'
         SPACE 3
DB2XSTAT CSECT
DB2XSTAT AMODE 31
DB2XSTAT RMODE 24
*************************************************************
*        Standard Entry Linkage.
*************************************************************
         SPACE
         STM   R14,R12,12(R13)    * Save Registers
         BALR  R12,Ø              * New Base Addressability
         USING *,R12              * Get Addressability
         LA    R3,SAVEAREA        * Get pgm Save Area Address
         ST    R3,8(R13)          * Forward Chain
         ST    R13,4(R3)          * Backward Chain
         LR    R13,R3             * Set pgm Save Area Pointer
         EJECT
*************************************************************
*        Point to  PARMLIST
*************************************************************
         SPACE
PARAMETR EQU   *
         LR    R7,R1              * Load parm address
         L     R7,Ø(R7)           * Point Parm list
         LH    R8,Ø(R7)           * Load Parm list length
         BCTR  R8,Ø               * Len -1 for EX
         EX    R8,MOVEPARM        * Move parameter (DB2name)
```

```
         EJECT
*****************************************************************
*        Start PGM
*****************************************************************
MAINPGM  EQU   *
         OPEN  (SYSTBCK,OUTPUT),MODE=31  * Open dataset output
         MODESET KEY=ZERO,MODE=SUP  * in supervisor STATE
         L     R5,16              * CVT pointer
         L     R5,296(R5)         * JESCT pointer
         L     R5,24(R5)          * SSCVT pointer
SSCYCLE  EQU   *
         CLC   8(4,R5),PARAM      * Check SS with DB2 name ...
         BE    CONTINUE           * ... if equal, OK
         L     R5,4(R5)           * Next Sscvt pointer
         LTR   R5,R5              * Check coherency pointer ...
         BNZ   SSCYCLE            * if good, load new SScvt
         MVC   RETCODE,=F'8'      * Move 8 in return code
         B     ENDPGM             * exit with error rcode
CONTINUE EQU   *
         L     R5,2Ø(R5)          * Erly pointer
         L     R5,56(R5)          * Scom pointer
         L     R5,144(R5)         * Acom pointer
         L     R6,112(R5)         * Qxst pointer...
         USING DSNDQXST,R6        * ...mapping DSECT
         MVC   RECORD(QXSTLEN),DSNDQXST * Move DSECT to output
PUT   SYSTBCK,RECORD           * in SYSTBCK
         L     R7,116(R5)              * Qtst pointer...
         USING DSNDQTST,R7             * ...mapping DSECT
         MVC   RECORD(QTSTLEN),DSNDQTST * Move DSECT to output
PUT   SYSTBCK,RECORD           * in SYSTBCK
         MVC   RETCODE,=F'Ø'            * Move Ø in rcode
         EJECT
*****************************************************************
*        Exit point
*****************************************************************
         SPACE
ENDPGM   EQU   *
         L     R13,SAVEAREA+4      * Restore Save Area
         XR    R15,R15             * Clear R15
         L     R15,RETCODE         * Load return code value
         RETURN (14,12),,RC=(15)   * Exit pgm
         EJECT
*****************************************************************
*        Istruction called by EXECUTE command
*****************************************************************
         SPACE
MOVEPARM MVC   PARAM(Ø),2(R7)      * LOAD DB2 name parm
         EJECT
*****************************************************************
*        Work Area and Constants
```

```
*****************************************************************
         SPACE
         DS    ØF
SAVEAREA DS    18F'Ø'              * My Save Area
RETCODE  DC    F'8'                * Return code
PARAM    DC    CL4' '              * DB2 Name from PARM list
RECORD   DS    CL255               * Output area
         EJECT
*****************************************************************
*        Data Control Block of dataset   - SYSTBCK -
*****************************************************************
         SPACE
SYSTBCK  DCB   DDNAME=SYSTBCK,DSORG=PS,MACRF=(PM),LRECL=255
         LTORG
*****************************************************************
*        Equate Register
*****************************************************************
RØ       EQU   Ø
R1       EQU   1
R2       EQU   2
R3       EQU   3
R4       EQU   4
R5       EQU   5
R6       EQU   6
R7       EQU   7
R8       EQU   8
R9       EQU   9
R1Ø      EQU   1Ø
R11      EQU   11
R12      EQU   12
R13      EQU   13
R14      EQU   14
R15      EQU   15
*****************************************************************
*        Include DB2 control block DSECT
*****************************************************************
*        DSNDQTST SERVICE CONTROLLER STATISTICS BLOCK
*****************************************************************
         DSNDQTST
QTSTLEN  EQU   *-DSNDQTST        * length DSECT
*****************************************************************
*        DSNDQXST RDS STATISTICS BLOCK
*****************************************************************
         DSNDQXST
QXSTLEN  EQU   *-DSNDQXST        * length DSECT
         EJECT
         END
```

Andrea Stocchero
DB2 System Administrator
Cariverona SPA (Italy)

Deadlock/timeout reporting tool

OBJECT

DB2 Version 4 and Type II indexes mean no more contention! Well yes, if the contention involves deadlocks on index pages. But, with compression this has in some cases translated to deadlocks on tablespaces. Unless the design, the subsystem configuration, and the workload are perfectly balanced, it is unlikely that the applications will be contention-free, and the system will suffer such contention at the worst possible time for the users. This article explains the tool set-up to ease the analysis of such contention.

SOURCE

The reporting of DEADLOCKs and TIMEOUTs in a DB2 subsystem is possible through four options – read the MVS JES log, read the JESMSGS of the DB2 system address space, use the IFI interface, or use the 196 and 172 IFCID data from SMF.

The IFI interface will not be mentioned further, because it is outside the scope of this article.

The option to browse either the system log or DB2 system address space is very laborious, particularly when large bursts of multiple contention messages are produced in a short space of time.

The use of 196 and 172 IFCID data is possible with the activation of the statistics trace. The output from these IFCID can be found in the type 102 SMF records.

IFCID 172 identifies the resource involved in the deadlock, but, there is no such information produced by 196 for a timeout.

For this reason, these reports are not useful in the long-term but excellent for short-term analysis, since they can be produced with the minimum of effort and without the time delay, as when using the MVS JESlog. Sample SAS code for this purpose is shown at the end of this article.

The messages produced by DB2 are captured into DB2 tables designed to reflect the message structure of DSNT501I, DSNT375I, and DSNT376I.

COLUMN NAME	TYPE	LENGTH	NL	UP	INDEX NAME	IXLCOL
DLTORTME	INTEGER	4			DLTO01X0DSNT501I	1
DLTOSSID	CHAR	4			DLTO01X0DSNT501I	2
DLTOSEQN	INTEGER	4			DLTO01X0DSNT501I	3
DLTOCORR	CHAR	12				
DLTOCONN	CHAR	8				
DLTOREAS	CHAR	10				
DLTOTYPE	CHAR	8				
DLTONAME	CHAR	44				

Figure 1: TDLTO01DSNT501I table

Figures 1 to 3 display both the table structure and the indexes defined on each table.

The tables capture the essential information from each of the contention messages that DB2 produces.

The messages that are destined for the MVS JES log and the DB2 system address space are different in format.

COLUMN NAME	TYPE	LENGTH	NL	UP	INDEX NAME	IXLCOL
DLTORTME	INTEGER	4			DLTO02X0DSNT375I	1
DLTOSSID	CHAR	4			DLTO01X0DSNT375I	2
DLTOSEQN	INTEGER	4			DLTO02X0DSNT375I	3
DLTOPLN1	CHAR	8				
DLTOCOR1	CHAR	12				
DLTOCON1	CHAR	8				
DLTOREAS	CHAR	10				
DLTOPLN2	CHAR	8				
DLTOCOR2	CHAR	12				
DLTOCON2	CHAR	8				

Figure 2: TDLTO02DSNT375I table

```
COLUMN NAME    TYPE        LENGTH  NL  UP   INDEX NAME         IXLCOL
-------------  --------  --------  --  --   ----------------- ------
DLTORTME        INTEGER        4             DLTØ3XØDSNT376I        1
DLTOSSID        CHAR           4             DLTØ1XØDSNT376I        2
DLTOSEQN        INTEGER        4             DLTØ3XØDSNT376I        3
DLTOVPLN        CHAR           8
DLTOVCOR        CHAR          12
DLTOVCON        CHAR           8
DLTOREAS        CHAR          1Ø
DLTOCPLN        CHAR           8
DLTOCCOR        CHAR          12
DLTOCCON        CHAR           8
```

Figure 3: TDLTO03DSNT376I table

A sample of the MVS JES log message DSNT501I looks like:

```
M 8ØØØØØØ MMIØ 96337 Ø7:ØØ:56.94 STCØØØ55 ØØØØØØ94  DSNT5Ø1I -
D                                  958 ØØØØØØ94     CORRELATION-I
D                                  958 ØØØØØØ94     CONNECTION-ID
D                                  958 ØØØØØØ94     LUW-ID=*
D                                  958 ØØØØØØ94     REASON ØØC9ØØ
D                                  958 ØØØØØØ94     TYPE ØØØØØ3Ø2
E                                  958 ØØØØØØ94     NAME HOMAØ1D
```

A sample of the DB2 system address space message looks like:

```
Ø6.15.39 STCØØØ55  DSNT5Ø1I - DSNILMCL RESOURCE UNAVAILABLE
                              CORRELATION-ID=ØØØ7COMA3
                              CONNECTION-ID=MIØ1
                              LUW-ID=*
                              REASON ØØC9ØØ88
                              TYPE ØØØØØ3Ø2
                              NAME HOMAØ1D .HOMA18S .X'ØØØ35E'
```

The message in the DB2 system address space does not, for example, have time in milliseconds, neither does it have a Julian date, and there is no subsystem/MVS identifier available.

Because of these differences, the process that extracts the messages from JESMSGS dataset of the DB2 system address space would have to create some dummy entries in the columns, DLTOSSID.

The column DLTOSEQN captures the Julian date when the extraction

is from the MVS JES log, but, when the extraction is from the DB2 system address space, this field is a sequential number generated in the program, and is normally used to match the messages when the contention burst is from many different applications.

The following is a sample SQL used to produce the contention report:

```
SELECT A.DLTOSEQN,A.DLTORTME,
' D '||A.DLTONAME||'   '||B.DLTOCOR1||' '||B.DLTOCOR2
FROM  TDLTO01DSNT501I A,
      TDLTO02DSNT375I B
WHERE A.DLTOCORR = B.DLTOCOR1
  AND A.DLTOSEQN = B.DLTOSEQN
  AND A.DLTOSSID = B.DLTOSSID
  AND A.DLTOSSID = ?
  AND A.DLTORTME =
      ( SELECT MIN(D.DLTORTME) FROM TDLTO01DSNT501I D
        WHERE D.DLTOCORR = B.DLTOCOR1
          AND D.DLTOSSID = B.DLTOSSSID
          AND D.DLTORTME >= B.DLTORTME )
UNION
SELECT A.DLTOSEQN,A.DLTORTME,
' T '||A.DLTONAME||' V-'||B.DLTOVCOR||' '||B.DLTOCCOR
FROM  TDLTO01DSNT501I A,
      TDLTO03DSNT376I B
WHERE A.DLTOCORR = B.DLTOVCOR
  AND A.DLTOSEQN = B.DLTOSEQN
  AND A.DLTOSSID = B.DLTOSSID
  AND A.DLTOSSID = ?
  AND A.DLTORTME =
      ( SELECT MIN(D.DLTORTME) FROM TDLTO01DSNT501I D
        WHERE D.DLTOCORR = B.DLTOVCOR
          AND D.DLTOSSID = B.DLTOSSID
          AND D.DLTORTME >= B.DLTORTME )
ORDER BY 1 , 2 ;
```

Output from the SQL above is shown in Figure 4.

When using the data extracted from the DB2 system address space, the SQL above has to be modified by removing any predicate involving DLTOSEQN.

The reference to the column DLTOSSID in all predicates should also be removed because it will contain a dummy SSID.

CONCLUSIONS

The procedure is simple and once set-up can provide a long-term

```
  +-------------------------------------------------------------------------------------------------------------+
  !    DLTOSEQN   !    DLTORTME    !                                                                            !
  +-------------------------------------------------------------------------------------------------------------+
1_!         96340 !       10473587 !  T DSNDB06.SYSPKAGE.X'003B39'               V-020.AUTBND06 CRNRDRIX    !
2_!         96340 !       11415578 !  T HOMA01D.HOMA19X1.X'000C7D'               V-0045HUDA8    M70494A     !
3_!         96340 !       13031023 !  D HOMA01D.HOMA19X3.X'0000B8'                 0043HUDA3    0036HUDA3   !
4_!         96340 !       15305824 !  D HOMA01D.HOMA19X3.X'0000B4'                 0050HUDA3    0043HUDA3   !
5_!         96340 !       17133101 !  T QQ5302D.HOCA06S .X'000002'               V-Q33237       Q36536      !
6_!         96340 !       17484168 !  D HODA02D.HODA03X1.X'0000D6'.X'03'           0002HUD2P    000HUD72P   !
7_!         96340 !       18193106 !  T DSNDB06.SYSDBASE                         V-M711222      LWFDB2A1    !
  +-------------------------------------------------------------------------------------------------------------+
```

Figure 4: Example output

facility. Capturing the contention information on a regular basis will automate the collection of the information from the MVS JES log.

The same table structures and technique can be used for quick analysis using the JESMSGS dataset of the DB2 system address space.

EXAMPLE SAS CODE

```
DATA WORK.DEADLCK1
 (KEEP = RPTDTIME QWHCPLAN QWHCAID  QWHCCN   QWHCCV   QWHSLUNM
         QW0196NU LOCKDURQ LOCKSTRQ LOCKFURQ QWHSLUCC QW0196HW
         LOCKDUH1 QW0196HN QW0196HP QW0196HR LOCKSTH1 QW0196HW
         LOCKDUH2 QW1196HN QW1196HP QW1196HR LOCKSTH2 QW1196HW
         LOCKDUH3 QW2196HN QW2196HP QW2196HR LOCKSTH3 QW2196HW
         QW0196WU QW0196WD QW0196WS
         QW0196HD QW1196HD QW2196HD
         QW0196HS QW1196HS QW2196HS
         SMFTIME  QWHSSTCK QWHSISEQ );
SET PDB5.T102S196 ;
RPTDTIME = TIMEPART(SMFTIME) ;
FORMAT RPTDTIME TIME11.3 ;
  SELECT (QW0196WU) ;
    WHEN ('02'X)  LOCKFURQ = 'LOCK  ' ;
    WHEN ('03'X)  LOCKFURQ = 'UNLOCK' ;
    WHEN ('04'X)  LOCKFURQ = 'CHANGE' ;
    WHEN ('06'X)  LOCKFURQ = 'QUERY ' ;
    OTHERWISE     LOCKFURQ = '------' ;
  END ;
  SELECT (QW0196WD) ;
    WHEN ('20'X)  LOCKDURQ = 'M  ' ;
    WHEN ('21'X)  LOCKDURQ = 'M+1' ;
    WHEN ('40'X)  LOCKDURQ = 'C  ' ;
    WHEN ('41'X)  LOCKDURQ = 'C+1' ;
    WHEN ('60'X)  LOCKDURQ = 'ALC' ;
    WHEN ('80'X)  LOCKDURQ = 'PLN' ;
    WHEN ('FF'X)  LOCKDURQ = 'FRA' ;
    OTHERWISE     LOCKDURQ = '---' ;
  END ;
  SELECT (QW0196HD) ;
    WHEN ('20'X)  LOCKDUH1 = 'M  ' ;
    WHEN ('21'X)  LOCKDUH1 = 'M+1' ;
    WHEN ('40'X)  LOCKDUH1 = 'C  ' ;
    WHEN ('41'X)  LOCKDUH1 = 'C+1' ;
    WHEN ('60'X)  LOCKDUH1 = 'ALC' ;
    WHEN ('80'X)  LOCKDUH1 = 'PLN' ;
    WHEN ('FF'X)  LOCKDUH1 = 'FRA' ;
    OTHERWISE     LOCKDUH1 = '---' ;
  END ;
```

```
SELECT (QW1196HD) ;
  WHEN ('20'X)  LOCKDUH2 = 'M  ' ;
  WHEN ('21'X)  LOCKDUH2 = 'M+1' ;
  WHEN ('40'X)  LOCKDUH2 = 'C  ' ;
  WHEN ('41'X)  LOCKDUH2 = 'C+1' ;
  WHEN ('60'X)  LOCKDUH2 = 'ALC' ;
  WHEN ('80'X)  LOCKDUH2 = 'PLN' ;
  WHEN ('FF'X)  LOCKDUH2 = 'FRA' ;
  OTHERWISE     LOCKDUH2 = '---' ;
END ;
SELECT (QW2196HD) ;
  WHEN ('20'X)  LOCKDUH3 = 'M  ' ;
  WHEN ('21'X)  LOCKDUH3 = 'M+1' ;
  WHEN ('40'X)  LOCKDUH3 = 'C  ' ;
  WHEN ('41'X)  LOCKDUH3 = 'C+1' ;
  WHEN ('60'X)  LOCKDUH3 = 'ALC' ;
  WHEN ('80'X)  LOCKDUH3 = 'PLN' ;
  WHEN ('FF'X)  LOCKDUH3 = 'FRA' ;
  OTHERWISE     LOCKDUH3 = '---' ;
END ;
SELECT (QW0196WS) ;
  WHEN ('00'X)  LOCKSTRQ = ' ANY' ;
  WHEN ('01'X)  LOCKSTRQ = ' USL' ;
  WHEN ('02'X)  LOCKSTRQ = '  IS' ;
  WHEN ('03'X)  LOCKSTRQ = '  IX' ;
  WHEN ('04'X)  LOCKSTRQ = '   S' ;
  WHEN ('05'X)  LOCKSTRQ = '   U' ;
  WHEN ('06'X)  LOCKSTRQ = ' SIX' ;
  WHEN ('07'X)  LOCKSTRQ = 'NSUL' ;
  WHEN ('08'X)  LOCKSTRQ = '   X' ;
  OTHERWISE     LOCKSTRQ = '----' ;
END ;
SELECT (QW0196HS) ;
  WHEN ('00'X)  LOCKSTH1 = ' ANY' ;
  WHEN ('01'X)  LOCKSTH1 = ' USL' ;
  WHEN ('02'X)  LOCKSTH1 = '  IS' ;
  WHEN ('03'X)  LOCKSTH1 = '  IX' ;
  WHEN ('04'X)  LOCKSTH1 = '   S' ;
  WHEN ('05'X)  LOCKSTH1 = '   U' ;
  WHEN ('06'X)  LOCKSTH1 = ' SIX' ;
  WHEN ('07'X)  LOCKSTH1 = 'NSUL' ;
  WHEN ('08'X)  LOCKSTH1 = '   X' ;
  OTHERWISE     LOCKSTH1 = '----' ;
END ;
SELECT (QW1196HS) ;
  WHEN ('00'X)  LOCKSTH2 = ' ANY' ;
  WHEN ('01'X)  LOCKSTH2 = ' USL' ;
  WHEN ('02'X)  LOCKSTH2 = '  IS' ;
  WHEN ('03'X)  LOCKSTH2 = '  IX' ;
  WHEN ('04'X)  LOCKSTH2 = '   S' ;
```

```
    WHEN ('Ø5'X)  LOCKSTH2 = '   U' ;
    WHEN ('Ø6'X)  LOCKSTH2 = ' SIX' ;
    WHEN ('Ø7'X)  LOCKSTH2 = 'NSUL' ;
    WHEN ('Ø8'X)  LOCKSTH2 = '   X' ;
    OTHERWISE     LOCKSTH2 = '----' ;
  END ;
  SELECT (QW2196HS) ;
    WHEN ('ØØ'X)  LOCKSTH3 = ' ANY' ;
    WHEN ('Ø1'X)  LOCKSTH3 = ' USL' ;
    WHEN ('Ø2'X)  LOCKSTH3 = '  IS' ;
    WHEN ('Ø3'X)  LOCKSTH3 = '  IX' ;
    WHEN ('Ø4'X)  LOCKSTH3 = '   S' ;
    WHEN ('Ø5'X)  LOCKSTH3 = '   U' ;
    WHEN ('Ø6'X)  LOCKSTH3 = ' SIX' ;
    WHEN ('Ø7'X)  LOCKSTH3 = 'NSUL' ;
    WHEN ('Ø8'X)  LOCKSTH3 = '   X' ;
    OTHERWISE     LOCKSTH3 = '----' ;
  END ;
  PROC PRINT DATA=WORK.DEADLCK1 NOOBS ;
   VAR RPTDTIME QWHCCV
       QWØ196NU LOCKFURQ LOCKDURQ LOCKSTRQ
       QWØ196HP QWØ196HR QWØ196HW LOCKDUH1 LOCKSTH1
                QW1196HR          LOCKDUH2 LOCKSTH2
                QW2196HR          LOCKDUH3 LOCKSTH3
       QWHSLUCC ;
 TITLE ' TIMEOUT INFORMATION - IFCID 196' ;
/*--------------------DEADLOCKS-------------------------*/;
DATA WORK.DEADLCKØ
 (KEEP = QWHCPLAN QWHCAID  QWHCCN   QWHCCV   QWHSLUNM
         QWØ172FR QWØ172HD QWØ172HF QWØ172HI QWØ172HL
         QWØ172HN QWØ172HO QWØ172HP QWØ172HR QWØ172HS
         QWØ172KD QWØ172KP QWØ172KR LOCKSTWR LOCKFUWR
         RPTDTIME LOCKSTHR QWØ172NR QWØ172RN QWØ172TD
         QWØ172WA QWØ172WD QWØ172WF QWØ172WI QWØ172WL
         QWØ172WN QWØ172WO QWØ172WP QWØ172WR QWØ172WS
         LOCKFUWR LOCKDUWR RESTYPE  LOCKDUHR
         QWØ172WW SMFTIME  );
SET PDB5.T1Ø2S172 ;
RPTDTIME = TIMEPART(QWØ172TD) ;
  SELECT (QWØ172FR) ;
    WHEN ('ØØ'X)  RESTYPE = 'DPG  ' ;
    WHEN ('Ø1'X)  RESTYPE = 'DB   ' ;
    WHEN ('Ø2'X)  RESTYPE = 'TSP  ' ;
    WHEN ('Ø3'X)  RESTYPE = 'PRTN ' ;
    WHEN ('Ø4'X)  RESTYPE = 'SKCT ' ;
    WHEN ('Ø5'X)  RESTYPE = 'IXPG ' ;
    WHEN ('Ø6'X)  RESTYPE = 'RSRV ' ;
    WHEN ('Ø7'X)  RESTYPE = 'SERL ' ;
    WHEN ('Ø8'X)  RESTYPE = 'UTIL ' ;
    WHEN ('Ø9'X)  RESTYPE = 'PS P ' ;
```

```
  WHEN ('ØA'X)  RESTYPE = 'DBDØ2' ;
  WHEN ('ØB'X)  RESTYPE = 'GP@AB' ;
  WHEN ('ØC'X)  RESTYPE = '32KPL' ;
  WHEN ('ØD'X)  RESTYPE = 'SYSLG' ;
  WHEN ('ØE'X)  RESTYPE = 'UTILS' ;
  WHEN ('ØF'X)  RESTYPE = 'MASSD' ;
  WHEN ('1Ø'X)  RESTYPE = 'TABLE' ;
  WHEN ('11'X)  RESTYPE = 'HASHA' ;
  WHEN ('12'X)  RESTYPE = 'SPT  ' ;
  WHEN ('13'X)  RESTYPE = 'COLEC' ;
  WHEN ('17'X)  RESTYPE = 'AUTBN' ;
  OTHERWISE     RESTYPE = '-----' ;
END ;
SELECT (QWØ172WF) ;
  WHEN ('Ø2'X)  LOCKFUWR = 'LOCK  ' ;
  WHEN ('Ø3'X)  LOCKFUWR = 'UNLOCK' ;
  WHEN ('Ø4'X)  LOCKFUWR = 'CHANGE' ;
  WHEN ('Ø6'X)  LOCKFUWR = 'QUERY ' ;
  OTHERWISE     LOCKFUWR = '------' ;
END ;
SELECT (QWØ172HS) ;
  WHEN ('ØØ'X)  LOCKSTHR = 'SERV' ;
  WHEN ('Ø1'X)  LOCKSTHR = 'RSVR' ;
  WHEN ('Ø2'X)  LOCKSTHR = '  IS' ;
  WHEN ('Ø3'X)  LOCKSTHR = '  IX' ;
  WHEN ('Ø4'X)  LOCKSTHR = '   S' ;
  WHEN ('Ø5'X)  LOCKSTHR = '   U' ;
  WHEN ('Ø6'X)  LOCKSTHR = ' SIX' ;
  WHEN ('Ø7'X)  LOCKSTHR = 'RSVR' ;
  WHEN ('Ø8'X)  LOCKSTHR = '   X' ;
  OTHERWISE     LOCKSTHR = '----' ;
END ;
SELECT (QWØ172WS) ;
  WHEN ('ØØ'X)  LOCKSTWR = 'SERV' ;
  WHEN ('Ø1'X)  LOCKSTWR = 'RSVR' ;
  WHEN ('Ø2'X)  LOCKSTWR = '  IS' ;
  WHEN ('Ø3'X)  LOCKSTWR = '  IX' ;
  WHEN ('Ø4'X)  LOCKSTWR = '   S' ;
  WHEN ('Ø5'X)  LOCKSTWR = '   U' ;
  WHEN ('Ø6'X)  LOCKSTWR = ' SIX' ;
  WHEN ('Ø7'X)  LOCKSTWR = 'RSVR' ;
  WHEN ('Ø8'X)  LOCKSTWR = '   X' ;
  OTHERWISE     LOCKSTWR = '----' ;
END ;
SELECT (QWØ172WD) ;
  WHEN ('2Ø'X)  LOCKDUWR = 'M  ' ;
  WHEN ('21'X)  LOCKDUWR = 'M+1' ;
  WHEN ('4Ø'X)  LOCKDUWR = 'C  ' ;
  WHEN ('41'X)  LOCKDUWR = 'C+1' ;
  WHEN ('6Ø'X)  LOCKDUWR = 'ALC' ;
```

```
    WHEN ('8Ø'X)  LOCKDUWR = 'PLN' ;
    WHEN ('FF'X)  LOCKDUWR = 'FRA' ;
    OTHERWISE     LOCKDUWR = '---' ;
  END ;
  SELECT (QWØ172HD) ;
    WHEN ('2Ø'X)  LOCKDUHR = 'M  ' ;
    WHEN ('21'X)  LOCKDUHR = 'M+1' ;
    WHEN ('4Ø'X)  LOCKDUHR = 'C  ' ;
    WHEN ('41'X)  LOCKDUHR = 'C+1' ;
    WHEN ('6Ø'X)  LOCKDUHR = 'ALC' ;
    WHEN ('8Ø'X)  LOCKDUHR = 'PLN' ;
    WHEN ('FF'X)  LOCKDUHR = 'FRA' ;
    OTHERWISE     LOCKDUHR = '---' ;
  END ;
FORMAT RPTDTIME TIME11.3 ;
  PROC SORT DATA=WORK.DEADLCKØ OUT=WORK.DEADLCK1 ;
   BY QWHSLUNM QWØ172KD QWØ172KP RPTDTIME QWØ172HP QWØ172HR ;
  RUN ;
  PROC PRINT DATA=WORK.DEADLCK1 NOOBS ;
    BY QWHSLUNM QWØ172KD QWØ172KP ;
   VAR RPTDTIME QWØ172KR QWØ172HP QWØ172HR LOCKSTHR
       RESTYPE  LOCKDUHR
       QWØ172NR QWØ172WA
       QWØ172WP QWØ172WR LOCKSTWR LOCKFUWR
       LOCKDUWR ;
XX
```

M K Mohan
DB2 Specialist (UK)

Rebind and convert plans and packages – part 2

This month we continue with the code that enables you to rebind plans and packages or convert plans to packages.

```
     WHEN(ff='4?') THEN DO
      Call Numeric 1
      head ="4-Rebind all plans since a given date and time"
      Call fields
      line4="where binddate >= '"vdate1"'"
      line5="and   bindtime >= '"vtime1"'"
      CUR='ff'
      Call Help
      ff=4
     END
     WHEN(ff='5') THEN DO
```

```
 Call Numeric 1
 if ind = '1' then SIGNAL top
 Call Numeric 2
 if ind = '1' then SIGNAL top
 Call fields
 title="Rebind plans within a given date and time range"
 line4="where binddate>='"vdate1"' and bindtime>='"vtime1"'"
 line5="and   binddate<='"vdate2"' and bindtime<='"vtime2"'"
 line5=line5||';'
 CUR='ff'
 Call Generate_jcl
END
WHEN(ff='5?') THEN DO
 Call Numeric 1
 Call Numeric 2
 head ="5-Rebind plans within a given date and time range"
 Call fields
 line4="where binddate>='"vdate1"' and bindtime>='"vtime1"'"
 line5="and   binddate<='"vdate2"' and bindtime<='"vtime2"'"
 CUR='ff'
 Call Help
 ff=5
END
WHEN(ff='6') THEN DO
 title="Rebind all invalid plans"
 line4="where valid='N'"||';'
 CUR='ff'
 Call Generate_jcl
END
WHEN(ff='6?') THEN DO
 head ="6-Rebind all invalid plans"
 line4="where valid='N'"
 CUR='ff'
 Call Help
 ff=6
END
WHEN(ff='7') THEN DO
 title="Rebind all inoperative plans"
 line4="where operative='N'"||';'
 CUR='ff'
 Call Generate_jcl
END
WHEN(ff='7?') THEN DO
 head ="7-Rebind all inoperative plans"
 line4="where operative='N'"
 CUR='ff'
 Call Help
 ff=7
END
WHEN(ff='8') THEN DO
 title="Rebind all plans bound with isolation level CS"
```

```
      line4="where isolation='S'"||';'
      CUR='ff'
      Call Generate_jcl
     END
     WHEN(ff='8?') THEN DO
      head ="8-Rebind all plans bound with isolation level CS"
      line4="where isolation='S'"
      CUR='ff'
      Call Help
      ff=8
     END
     OTHERWISE CUR='ff'
   END
   rst='NO'
   ans='NO'
   ADDRESS ISPEXEC "DISPLAY PANEL(REBPØ1) CURSOR("CUR")"
   Call dsn
END
EXIT Ø
Numeric:
   arg nf
   str1=value(time||nf)
   str2=value(date||nf)
   nn = verify(str1,'1234567890')
   ind=Ø
   IF nn > Ø THEN DO
      message=,
      'Btime'||nf||' or Bdate'||nf||,
      ' field only valid for numeric data'      ADDRESS ISPEXEC "SETMSG
MSG(PREBØØ1)"
      CUR='time'||nf
      ind=1
   END
   nn = VERIFY(str2,'1234567890')
   IF nn > Ø THEN DO
      message=,
      'Bdate'||nf||' or Btime'||nf||,
      ' field only valid for numeric data'      ADDRESS ISPEXEC "SETMSG
MSG(PREBØØ1)"
      CUR='date'||nf
      ind=1
   END
   IF length(str1) < 8 THEN DO
      message=,
      'Btime'||nf||' field must be in time format hhmmssth     .'
      ADDRESS ISPEXEC "SETMSG MSG(PREBØØ1)"
      CUR='time'||nf
      ind=1
   END
   IF length(str2) < 6 THEN DO
```

```
      message=,
      'Bdate'||nf||' field must be in date format yymmdd       .'
      ADDRESS ISPEXEC "SETMSG MSG(PREB001)"
      CUR='date'||nf
      ind=1
   END
Return
Fields:
   if date1 ¬= ''
   then vdate1=date1
   else vdate1='?'
   if time1 ¬= ''
   then vtime1=time1
   else vtime1='?'
   if vdate1='?' | vtime1='?'
   then text ="Enter required fields Bdate1 and Btime1"
   if ff='5?' | ff='5' then do
      if date2 ¬= ''
      then vdate2=date2
      else vdate2='?'
      if time2 ¬= ''
      then vtime2=time2
      else vtime2='?'
      if vdate2='?' | vtime2='?'
      then text ="Enter required fields Bdate2 and Btime2"
   end
Return
Generate_jcl:
   IF rst='YES' then Call Rstat
   ADDRESS ISPEXEC REMPOP ALL
   dat=DATE()
   tim=TIME(C)
   user=userid()
   tempfile=userid()||'.REBIND.PLAN'
   ADDRESS TSO
   "DELETE '"tempfile"'"
   "FREE DSNAME('"tempfile"')"
   "FREE DDNAME(ISPFILE)"
   "FREE ATTRLIST(FORMFILE)"
   "ATTRIB FORMFILE BLKSIZE(800) LRECL(80) RECFM(F B) DSORG(PS)"
   "ALLOC DDNAME(ISPFILE) DSNAME('"tempfile"')",
          "NEW USING (FORMFILE) UNIT(3390) SPACE(1 1) CYLINDERS"
   opt=1
   ADDRESS ISPEXEC
   "FTOPEN"
   "FTINCL PLANREB"
   "FTCLOSE"
   opt=0
   ZEDSMSG = "JCL shown"
   ZEDLMSG = "JCL for rebind shown"
```

```
   "SETMSG MSG(ISRZØØ1)"
   "EDIT DATASET('"tempfile"')"
   ADDRESS TSO
   "FREE DSNAME('"tempfile"')"
   if rst='YES' then ADDRESS ISPEXEC 'TBEND "PLPA"'
   ADDRESS ISPEXEC 'ADDPOP ROW(3) COLUMN(1)'
   EXIT 1
Return
Rstat:
   ADDRESS TSO "DELETE '"SYSVAR(SYSUID)".PREB.RUNSTAT'"
   "ALLOC DD(SYSPRINT) DSN('"SYSVAR(SYSUID)".PREB.RUNSTAT'),
   SPACE(24 8),   TRACK NEW UNIT(339Ø) RECFM(F,B) LRECL(8Ø)
BLKSIZE(8ØØ),   F(SYSPRINT) CATALOG REUSE "
   PARM = ff||SUBSTR(ppla,1,18)||date1||time1||date2||time2
   ADDRESS TSO
   QUEUE "RUN PROGRAM(PREBPL) PLAN(PREBPL),
          LIBRARY ('SKUPNI.BATCH.LOADLIB'),
          PARMS ('/"PARM"')"
   QUEUE "END "
   "DSN SYSTEM("DB2")"
   IF RC=12 | RC=8 THEN DO
      "delstack"
      CUR='DB2'
      message='Error.   'DB2||' ssid is not valid  |'
      ADDRESS ISPEXEC "SETMSG MSG(PREBØØ1)"
      "EXECIO Ø DISKR SYSPRINT (FINIS"
      ADDRESS TSO "FREE F(SYSPRINT)"
      SIGNAL top
   END
   "EXECIO * DISKR SYSPRINT (STEM ROW."
   IF SUBSTR(ROW.1,2) = 'NO CATALOG ENTRIES FOUND' THEN DO
      "EXECIO Ø DISKR SYSPRINT (FINIS"
      ADDRESS TSO "FREE F(SYSPRINT)"
      rst='NO'
      return
   END
   ADDRESS ISPEXEC 'TBCREATE "PLPA" NAMES(dbname tsname)'
   DO I=1 TO ROW.Ø
      dbname = STRIP(SUBSTR(ROW.I,2,8))
      tsname = SUBSTR(ROW.I,1Ø,8)
      ADDRESS ISPEXEC 'TBADD "PLPA"'
   END
   ADDRESS ISPEXEC 'TBTOP "PLPA"';
   "EXECIO Ø DISKR SYSPRINT (FINIS"
   ADDRESS TSO "FREE F(SYSPRINT)"
Return
Dsn:
   if db2=' ' then do
    CUR='DB2'
    message='Enter ssid |'
```

```
    ADDRESS ISPEXEC "SETMSG MSG(PREBØØ1)"
    SIGNAL top
   end
Return
Help:
   rcc=Ø
   ans='NO'
   ADDRESS ISPEXEC 'ADDPOP ROW(1) COLUMN(1)'
   ADDRESS ISPEXEC "DISPLAY PANEL(REBPØ1H)"
   rcc=rc
   ADDRESS ISPEXEC REMPOP ALL
   query=SUBSTR(line1,1,7Ø)||SUBSTR(line2,1,7Ø)||SUBSTR(line3,1,7Ø)||,
         SUBSTR(line4,1,7Ø)||SUBSTR(line5,1,7Ø)
   IF ans='YES' & rcc=Ø then Call rsql db2 query
   ADDRESS ISPEXEC 'ADDPOP ROW(3) COLUMN(1)'
   ans='NO'
Return
```

PAREB – REBIND PACKAGE PROCEDURE

```
/* REXX *//* trace r */X=MSG("OFF")ZPFCTL = 'OFF'ADDRESS ISPEXEC 'VPUT
(ZPFCTL) PROFILE'ADDRESS ISPEXEC 'VGET (db2) PROFILE'ADDRESS ISPEXEC
'ADDPOP ROW(3) COLUMN(1)'
rst='NO'
ans='NO'
CUR='ff'
top:
ADDRESS ISPEXEC "DISPLAY PANEL(REBPØ2) CURSOR("CUR")"
Call dsn
option='REBIND PACKAGE'
DO WHILE RC=Ø
   text=''
   line1="Select substr('REBIND PACKAGE('concat collid "
   line2="concat '.' concat name concat'.(*))',1,47)"
   line3="from sysibm.syspackage                    "
   line4=''
   line5=''
   SELECT
     WHEN(ff='1') THEN DO
      CUR='pack'
      title="Rebind package(s)"
      if pack = '' then do
         message="Enter package name-db2 wildcards supported"
         ADDRESS ISPEXEC "SETMSG MSG(PREBØØ1)"
         SIGNAL top
      end
      vname='%'
      if pack ¬= '' then vname=pack||'%'
      if length(pack) = 8 then vname=pack
```

```
 line4="where name like '"vname"'"||';'
 Call Generate_jcl
END
WHEN(ff='1?') THEN DO
 head ="1-Rebind package(s)"
 text ="Enter package name-db2 wildcards supported"
 vname='%'
 if pack ¬= '' then vname=pack||'%'
 if length(pack) = 8 then vname=pack
 line4="where name like '"vname"'"
 CUR='pack'
 Call Help
 ff=1
END
WHEN(ff='2') THEN DO
 title="Rebind all versions of the packages"
 line3=line3||';'
 CUR='ff'
 Call Generate_jcl
END
WHEN(ff='2?') THEN DO
 head ="2-Rebind all versions of the packages"
 CUR='ff'
 Call Help
 ff=2
END
WHEN(ff='3') THEN DO
 Call Numeric 1
 if ind = '1' then SIGNAL top
 title="Rebind packages before a given date and time"
 Call fields
 line4="where bindtime <= '"vtst1"'"||';'
 CUR='ff'
 Call Generate_jcl
END
WHEN(ff='3?') THEN DO
 Call Numeric 1
 head ="3-Rebind packages before a given date and time"
 Call fields
 line4="where bindtime <= '"vtst1"'"
 CUR='ff'
 Call Help
 ff=3
END
WHEN(ff='4') THEN DO
 Call Numeric 1
 if ind = '1' then SIGNAL top
 Call fields
 title="Rebind packages since a given date and time"
 line4="where bindtime >= '"vtst1"'"||';'
```

```
 CUR='ff'
 Call Generate_jcl
END
WHEN(ff='4?') THEN DO
 Call Numeric 1
 head ="4-Rebind packages since a given date and time"
 Call fields
 line4="where bindtime >= '"vtst1"'"
 CUR='ff'
 Call Help
 ff=4
END
WHEN(ff='5') THEN DO
 Call Numeric 1
 if ind = '1' then SIGNAL top
 Call Numeric 2
 if ind = '1' then SIGNAL top
 Call fields
 title="Rebind packages within a given date and time range"
 line4="where bindtime>='"vtst1"'"
 line5="and   bindtime<='"vtst2"'"
 line5=line5||';'
 CUR='ff'
 Call Generate_jcl
END
WHEN(ff='5?') THEN DO
 Call Numeric 1
 Call Numeric 2
 head ="5-Rebind packages within a given date and time range"
 Call fields
 line4="where bindtime>='"vtst1"'"
 line5="and   bindtime<='"vtst2"'"
 CUR='ff'
 Call Help
 ff=5
END
WHEN(ff='6') THEN DO
 title="Rebind all invalid versions of the packages"
 line4="where valid='N'"||';'
 CUR='ff'
 Call Generate_jcl
END
WHEN(ff='6?') THEN DO
 head ="6-Rebind all invalid versions of the packages"
 line4="where valid='N'"
 CUR='ff'
 Call Help
 ff=6
END
WHEN(ff='7') THEN DO
```

```
       title="Rebind all inoperative versions of the packages"
       line4="where operative='N'"||';'
       CUR='ff'
       Call Generate_jcl
      END
      WHEN(ff='7?') THEN DO
       head ="7-Rebind all inoperative versions of the packages"
       line4="where operative='N'"
       CUR='ff'
       Call Help
       ff=7
      END
      WHEN(ff='8') THEN DO
       title="Rebind all packages that allow CPU,I/O parallelism"
       line4="where degree='ANY'"||';'
       CUR='ff'
       Call Generate_jcl
      END
      WHEN(ff='8?') THEN DO
       head ="8-Rebind packages that allow CPU,I/O parallelism"
       line4="where degree='ANY'"
       CUR='ff'
       Call Help
       ff=8
      END
      OTHERWISE CUR='ff'
   END
   rst='NO'
   ans='NO'
   ADDRESS ISPEXEC "DISPLAY PANEL(REBPØ2) CURSOR("CUR")"
   Call dsn
END
EXIT Ø
Numeric:
   arg nf
   str1=value(tst||nf)
   nn = verify(str1,'1234567890-.')
   ind=Ø
   IF nn > Ø | length(str1) < 26 THEN DO
      message=,
      'Tst'||nf||' field only valid in timestamp format.'
      ADDRESS ISPEXEC "SETMSG MSG(PREBØØ1)"
      CUR='tst'||nf
      ind=1
   END
Return
Fields:
   if tst1 ¬= ''
   then vtst1=tst1
   else vtst1='?'
```

```
   if vtst1='?'
   then text ="Enter required field Tst1."
   if ff='5?' | ff='5' then do
      if tst2 ¬= ''
      then vtst2=tst2
      else vtst2='?'
      if vtst2='?'
      then text ="Enter required field Tst2."
   end
Return
Generate_jcl:
   IF rst='YES' then Call Rstat
   ADDRESS ISPEXEC REMPOP ALL
   dat=DATE()
   tim=TIME(C)
   user=userid()
   tempfile=userid()||'.REBIND.PACKAGE'
   ADDRESS TSO
   "DELETE '"tempfile"'"
   "FREE DSNAME('"tempfile"')"
   "FREE DDNAME(ISPFILE)"
   "FREE ATTRLIST(FORMFILE)"
   "ATTRIB FORMFILE BLKSIZE(800) LRECL(80) RECFM(F B) DSORG(PS)"
   "ALLOC DDNAME(ISPFILE) DSNAME('"tempfile"')",
          "NEW USING (FORMFILE) UNIT(3390) SPACE(1 1) CYLINDERS"
   opt=2
   ADDRESS ISPEXEC
   "FTOPEN"
   "FTINCL PLANREB"
   "FTCLOSE"
   opt=0
   ZEDSMSG = "JCL shown"
   ZEDLMSG = "JCL for rebind shown"
   "SETMSG MSG(ISRZ001)"
   "EDIT DATASET('"tempfile"')"
   ADDRESS TSO
   "FREE DSNAME('"tempfile"')"
   if rst='YES' then ADDRESS ISPEXEC 'TBEND "PLPA"'
   ADDRESS ISPEXEC 'ADDPOP ROW(3) COLUMN(1)'
   EXIT 1
Return
Rstat:
   ADDRESS TSO "DELETE '"SYSVAR(SYSUID)".PREB.RUNSTAT'"
   "ALLOC DD(SYSPRINT) DSN('"SYSVAR(SYSUID)".PREB.RUNSTAT'),
   SPACE(24 8),   TRACK NEW UNIT(3390) RECFM(F,B) LRECL(80)
BLKSIZE(800),   F(SYSPRINT) CATALOG REUSE "   PARM =
ff||SUBSTR(pack,1,8)||tst1||tst2   ADDRESS TSO   QUEUE "RUN
PROGRAM(PREBPA) PLAN(PREBPA),          LIBRARY ('SKUPNI.BATCH.LOADLIB'),
          PARMS ('/"PARM"')"
   QUEUE "END "
```

```
   "DSN SYSTEM("DB2")"
   IF RC=12 | RC=8 THEN DO
      "delstack"
      CUR='DB2'
      message='Error.   'DB2||' ssid is not valid  |'
      ADDRESS ISPEXEC "SETMSG MSG(PREBØØ1)"
      "EXECIO Ø DISKR SYSPRINT (FINIS"
      ADDRESS TSO "FREE F(SYSPRINT)"
      SIGNAL top
   END
   "EXECIO * DISKR SYSPRINT (STEM ROW."
   IF SUBSTR(ROW.1,2) = 'NO CATALOG ENTRIES FOUND' THEN DO
      "EXECIO Ø DISKR SYSPRINT (FINIS"
      ADDRESS TSO "FREE F(SYSPRINT)"
      message = 'No catalog entries found, check Search Field'
      CUR='pack'
      ADDRESS ISPEXEC "SETMSG MSG(PREBØØ1)"
      SIGNAL TOP
   END
   ADDRESS ISPEXEC 'TBCREATE "PLPA" NAMES(dbname tsname)'
   DO I=1 TO ROW.Ø
      dbname = STRIP(SUBSTR(ROW.I,2,8))
      tsname = SUBSTR(ROW.I,1Ø,8)
      ADDRESS ISPEXEC 'TBADD "PLPA"'
   END
   ADDRESS ISPEXEC 'TBTOP "PLPA"';
   "EXECIO Ø DISKR SYSPRINT (FINIS"
   ADDRESS TSO "FREE F(SYSPRINT)"
Return
Dsn:
   if db2=' ' then do
    CUR='DB2'
    message='Enter ssid |'
    ADDRESS ISPEXEC "SETMSG MSG(PREBØØ1)"
    SIGNAL top
   end
Return
Help:
   rcc=Ø
   ans='NO'
   ADDRESS ISPEXEC 'ADDPOP ROW(1) COLUMN(1)'
   ADDRESS ISPEXEC "DISPLAY PANEL(REBPØ1H)"
   rcc=rc
   ADDRESS ISPEXEC REMPOP ALL
   query=SUBSTR(line1,1,7Ø)||SUBSTR(line2,1,7Ø)||SUBSTR(line3,1,7Ø)||,
         SUBSTR(line4,1,7Ø)||SUBSTR(line5,1,7Ø)
   IF ans='YES' & rcc=Ø then Call rsql db2 query
   ADDRESS ISPEXEC 'ADDPOP ROW(3) COLUMN(1)'
   ans='NO'
Return
```

PPLPA – CONVERT PLAN TO PACKAGE

```
/* REXX *//* trace r */X=MSG("OFF")ZPFCTL = 'OFF'ADDRESS ISPEXEC 'VPUT
(ZPFCTL) PROFILE'ADDRESS ISPEXEC 'VGET (db2) PROFILE'ADDRESS ISPEXEC
'ADDPOP ROW(3) COLUMN(1)'rst='NO'ans='NO'
def='NO'
new=Ø
CUR='cplan'
msg='Enter to Continue'
top:
ADDRESS ISPEXEC "DISPLAY PANEL(REBPØ3) CURSOR("CUR")"
if rc¬=Ø then exit
msg=''
Call dsn
Call fields
if ind=1 then signal top
CUR1='iso'
top1:
if def='YES' then do
   ADDRESS ISPEXEC 'VGET (iso,val,rel,exp,cda,deg,dru) PROFILE'
   if iso=' ' then iso='SAME'
   if val=' ' then val='SAME'
   if rel=' ' then rel='SAME'
   if exp=' ' then exp='SAME'
   if cda=' ' then cda='SAME'
   if deg=' ' then deg='SAME'
   if dru=' ' then dru='SAME'
   ADDRESS ISPEXEC "DISPLAY PANEL(REBPØ4) CURSOR("CUR1")"
   if rc¬=Ø then do
      def='NO'
      new=Ø
      msg='Enter to Continue'
      signal top
   end
   Call fields1
   if ind1=1
   then signal top1
   else new=1
end
if new=Ø
then parm='ØSAMESAMESAME      SAMESAMESAMESAME'
else parm='1'||SUBSTR(iso,1,4)||SUBSTR(val,1,4)||SUBSTR(rel,1,1Ø)||,
     SUBSTR(exp,1,4)||SUBSTR(cda,1,4)||SUBSTR(deg,1,4)||SUBSTR(dru,1,4)
option='PLAN TO PACKAGE'
title ='CONVERSION PLAN TO PACKAGE'
IF rst='YES' then Call Rstat
Call Converse
ADDRESS ISPEXEC REMPOP ALL
dat=DATE()
```

```
tim=TIME(C)
user=userid()
tempfile=userid()||'.PLAN.PACKAGE'
ADDRESS TSO
"DELETE '"tempfile"'"
"FREE DSNAME('"tempfile"')"
"FREE DDNAME(ISPFILE)"
"FREE ATTRLIST(FORMFILE)"
"ATTRIB FORMFILE BLKSIZE(800) LRECL(80) RECFM(F B) DSORG(PS)"
"ALLOC DDNAME(ISPFILE) DSNAME('"tempfile"')",
       "NEW USING (FORMFILE) UNIT(3390) SPACE(1 1) CYLINDERS"
opt=1
ADDRESS ISPEXEC
"FTOPEN"
"FTINCL CONVERSE"
"FTCLOSE"
opt=0
ZEDSMSG = "JCL shown"
ZEDLMSG = "JCL for rebind shown"
"SETMSG MSG(ISRZ001)"
"EDIT DATASET('"tempfile"')"
ADDRESS TSO
"FREE DSNAME('"tempfile"')"
ADDRESS ISPEXEC 'ADDPOP ROW(3) COLUMN(1)'
IF rst='YES' then ADDRESS ISPEXEC 'TBEND "PLPA"'
ADDRESS ISPEXEC 'TBEND "CONV"'
EXIT 1
Fields:
   ind=0
   if rst='YES' | rst='NO'
   then ind=0
   else do
      ind=1
      CUR='rst'
      message='Enter YES or NO at the cursor position'
      ADDRESS ISPEXEC "SETMSG MSG(PREB001)"
      return
   end
   if ans='YES' | ans='NO'
   then ind=0
   else do
      ind=1
      CUR='ans'
      message='Enter YES or NO at the cursor position'
      ADDRESS ISPEXEC "SETMSG MSG(PREB001)"
      return
   end
   if cplan=' '
   then do
```

```
      ind=1
      CUR='cplan'
      message='Enter plan name at the cursor position'
      ADDRESS ISPEXEC "SETMSG MSG(PREBØØ1)"
      return
   end
   if def='YES' | def='NO'
   then ind=Ø
   else do
      ind=1
      CUR='def'
      message='Enter YES or NO at the cursor position'
      ADDRESS ISPEXEC "SETMSG MSG(PREBØØ1)"
      return
   end
Return
Fields1:
   ind1=Ø
   if iso='SAME' | iso='RR' | iso='CS' | iso='UR'
   then ind1=Ø
   else do
      ind1=1
      CUR1='iso'
      message=,
      'Valid are SAME, RR, CS, or UR at the cursor position'
      ADDRESS ISPEXEC "SETMSG MSG(PREBØØ1)"
      return
   end
   if val='SAME' | val='BIND' | val='RUN'
   then ind1=Ø
   else do
      ind1=1
      CUR1='val'
      message='Valid are SAME, BIND, or RUN at the cursor position'
      ADDRESS ISPEXEC "SETMSG MSG(PREBØØ1)"
      return
   end
   if rel='SAME' | rel='COMMIT' | rel='DEALLOCATE'
   then ind1=Ø
   else do
      ind1=1
      CUR1='rel'
      message=,
      'Valid are SAME, COMMIT, or DEALLOCATE at the cursor position'
      ADDRESS ISPEXEC "SETMSG MSG(PREBØØ1)"
      return
   end
   if exp='SAME' | exp='YES' | exp='NO'
   then ind1=Ø
```

```
   else do
      ind1=1
      CUR1='exp'
      message='Valid are SAME, YES, or NO at the cursor position'
      ADDRESS ISPEXEC "SETMSG MSG(PREBØØ1)"
      return
   end
   if cda='SAME' | cda='YES' | cda='NO'
   then ind1=Ø
   else do
      ind1=1
      CUR1='cda'
      message='Valid are SAME, YES, or NO at the cursor position'
      ADDRESS ISPEXEC "SETMSG MSG(PREBØØ1)"
      return
   end
   if deg='SAME' | deg='1' | deg='ANY'
   then ind1=Ø
   else do
      ind1=1
      CUR1='deg'
      message='Valid are SAME, 1, or ANY at the cursor position'
      ADDRESS ISPEXEC "SETMSG MSG(PREBØØ1)"
      return
   end
   if dru='SAME' | dru='BIND' | dru='RUN'
   then ind1=Ø
   else do
      ind1=1
      CUR1='dru'
      message='Valid are SAME, BIND, or RUN at the cursor position'
      ADDRESS ISPEXEC "SETMSG MSG(PREBØØ1)"
      return
   end
Return
Rstat:
   ADDRESS TSO "DELETE '"SYSVAR(SYSUID)".PREB.RUNSTAT'"
   "ALLOC DD(SYSPRINT) DSN('"SYSVAR(SYSUID)".PREB.RUNSTAT'),
   SPACE(24 8),   TRACK NEW UNIT(339Ø) RECFM(F,B) LRECL(8Ø)
BLKSIZE(8ØØ),   F(SYSPRINT) CATALOG REUSE "
   parmr = SUBSTR(cplan,1,8)
   ADDRESS TSO
   QUEUE "RUN PROGRAM(PREBRU) PLAN(PREBRU),
          LIBRARY ('SKUPNI.BATCH.LOADLIB'),
          PARMS ('/"parmr"')"
   QUEUE "END "
   "DSN SYSTEM("DB2")"
   Call Check
   ADDRESS ISPEXEC 'TBCREATE "PLPA" NAMES(dbname tsname)'
```

```
   DO I=1 TO ROW.Ø
      dbname = STRIP(SUBSTR(ROW.I,2,8))
      tsname = SUBSTR(ROW.I,1Ø,8)
      ADDRESS ISPEXEC 'TBADD "PLPA"'
   END
   ADDRESS ISPEXEC 'TBTOP "PLPA"';
   "EXECIO Ø DISKR SYSPRINT (FINIS"
   ADDRESS TSO "FREE F(SYSPRINT)"
Return
Converse:
   ADDRESS TSO "DELETE '"SYSVAR(SYSUID)".PLPA.CONV'"
   "ALLOC DD(SYSPRINT) DSN('"SYSVAR(SYSUID)".PLPA.CONV') SPACE(24 8),
   TRACK NEW UNIT(339Ø) RECFM(F,B) LRECL(8Ø) BLKSIZE(8ØØ),
   F(SYSPRINT) CATALOG REUSE "

   parm = SUBSTR(cplan,1,8)||parm
   ADDRESS TSO
   QUEUE "RUN PROGRAM(PREBCO) PLAN(PREBCO),
          LIBRARY ('SKUPNI.BATCH.LOADLIB'),
          PARMS ('/"parm"')"
   QUEUE "END "
   "DSN SYSTEM("DB2")"
   Call Check
   ADDRESS ISPEXEC 'TBCREATE "CONV" NAMES(line)'
   DO I=1 TO ROW.Ø
      line = SUBSTR(ROW.I,2,72)
      ADDRESS ISPEXEC 'TBADD "CONV"'
   END
   ADDRESS ISPEXEC 'TBTOP "CONV"';
   "EXECIO Ø DISKR SYSPRINT (FINIS"
   ADDRESS TSO "FREE F(SYSPRINT)"
Return
Dsn:
   if db2=' ' then do
    CUR='DB2'
    message='Enter ssid |'
    ADDRESS ISPEXEC "SETMSG MSG(PREBØØ1)"
    SIGNAL top
   end
Return
Check:
   IF RC=12 | RC=8 then do
    "delstack"
    CUR='DB2'
    message='Error.   'DB2||' ssid is not valid  |'
    ADDRESS ISPEXEC "SETMSG MSG(PREBØØ1)"
    "EXECIO Ø DISKR SYSPRINT (FINIS"
    ADDRESS TSO "FREE F(SYSPRINT)"
    SIGNAL top
```

```
   end
   "EXECIO * DISKR SYSPRINT (STEM ROW."
   IF SUBSTR(ROW.1,2) = 'NO CATALOG ENTRIES FOUND' then do
     "EXECIO Ø DISKR SYSPRINT (FINIS"
     ADDRESS TSO "FREE F(SYSPRINT)"
     message = 'No catalog entries found, check Search Field'
     CUR='cplan'
     ADDRESS ISPEXEC "SETMSG MSG(PREBØØ1)"
     SIGNAL TOP
   end
Return
```

RSQL – ON-LINE SQL PROCEDURE

```
/* REXX */
ARG db2 query
/* trace r */
X=MSG("OFF")
ADDRESS TSO "DELETE '"SYSVAR(SYSUID)".SQL.IN'"
ADDRESS TSO "DELETE '"SYSVAR(SYSUID)".SQL.OUT'"
"ALLOC DD(SYSIN) DSN('"SYSVAR(SYSUID)".SQL.IN') SPACE(1 1) ,
TRACK NEW UNIT(339Ø) RECFM(F,B) LRECL(8Ø) BLKSIZE(8ØØ) ,
F(SYSIN) CATALOG REUSE "
"ALLOC DD(SYSPRINT) DSN('"SYSVAR(SYSUID)".SQL.OUT') SPACE(2 1) ,
TRACK NEW UNIT(339Ø) RECFM(F,B) LRECL(8Ø) BLKSIZE(8ØØ) ,
F(SYSPRINT) CATALOG REUSE "

i=Ø
do j=1 to length(query) by 7Ø
   i=i+1
   NEWROW.i = substr(query,j,7Ø)
end
i=i+1
NEWROW.i = substr(query,j)

"EXECIO * DISKW SYSIN (STEM NEWROW. FINIS"

ADDRESS TSO
QUEUE "RUN PROGRAM(DSNTEP2) PLAN(DSNTEP41),
       LIBRARY ('DSN41Ø.RUNLIB.LOAD') ";
QUEUE "END "
"DSN SYSTEM("db2")"

"EXECIO * DISKR SYSPRINT (STEM ROW. FINIS"
ADDRESS ISPEXEC 'TBCREATE ISQLIST NAMES(V1)'
DO I=4 TO ROW.Ø
   IF SUBSTR(ROW.I,1,35)='                                    '
   THEN V1=SUBSTR(ROW.I,4Ø,72)
```

```
   ELSE V1='    '||SUBSTR(ROW.I,1,133)
   IF SUBSTR(ROW.I,1,11)='ØSUCCESSFUL'
   THEN V1='      '||SUBSTR(ROW.I,2,7Ø)
   ADDRESS ISPEXEC 'TBADD ISQLIST';
END
ADDRESS ISPEXEC 'TBTOP ISQLIST';
ADDRESS ISPEXEC 'TBDISPL ISQLIST PANEL(RSQLPAN)';
ADDRESS ISPEXEC 'TBEND ISQLIST';
"EXECIO Ø DISKR SYSIN    (FINIS"
"EXECIO Ø DISKR SYSPRINT (FINIS"
ADDRESS TSO "FREE F(SYSIN)"
ADDRESS TSO "FREE F(SYSPRINT)"
EXIT
```

REBP00 – MAIN MENU

```
)ATTR DEFAULT(%+_)   [ TYPE (OUTPUT) INTENS(LOW)  COLOR(GREEN) CAPS(OFF)
# TYPE (OUTPUT) INTENS(LOW)  COLOR(WHITE) CAPS(OFF)   ] TYPE (TEXT)
INTENS(LOW)  COLOR(WHITE) CAPS(OFF)  HILITE(REVERSE)
   _ TYPE (INPUT)  INTENS(LOW)  COLOR(YELLOW) CAPS(ON)  HILITE(BLINK)
   | TYPE (OUTPUT) INTENS(LOW)  COLOR(GREEN)  CAPS(OFF)
   + TYPE (TEXT)   INTENS(LOW)  COLOR(GREEN)
   / TYPE (TEXT)   INTENS(LOW)  COLOR(TURQ)
   @ TYPE (TEXT)   INTENS(HIGH) COLOR(RED)   CAPS(OFF)  HILITE(REVERSE)
)BODY WINDOW(41,15) EXPAND ($$)
+]                   + Date:|date       +
+] DB2 Rebind Action + Time:|time      +
+]                   + User: &zuser
/ ************************************
+
+ [row1                               +
+ [row2                               +
+ [row3                               +
+ [row4                               +
+
/ ************************************
+
+@==>+ _X+  #msg                       +
+
+] PF1 Help +                ] PF3 End +
)INIT
  .HELP = REBPØØH
  &row1= '1 - Rebind plan(s)'
  &row2= '2 - Rebind package(s)'
  &row3= '3 - Conversion plan(s) to package(s)'
  &row4= 'X - Exit'
  IF (&X = 1,2,3,X)
      &msg = ''
```

```
  ELSE
      .ATTR (msg) = 'COLOR (RED)'
      &msg = 'Enter 1, 2, 3 or X.'
  IF (&X = 1)
      .ATTR (row1) = 'COLOR (YELLOW) CAPS(ON)'
  IF (&X = 2)
      .ATTR (row2) = 'COLOR (YELLOW) CAPS(ON)'
  IF (&X = 3)
      .ATTR (row3) = 'COLOR (YELLOW) CAPS(ON)'
)PROC
  IF (.PFKEY = PFØ3) &PF3 = EXIT
)END
```

REBP00H – HELP FILE

```
)ATTR DEFAULT(#+")  @ TYPE (TEXT) INTENS(HIGH) COLOR(RED) CAPS(OFF)
HILITE(REVERSE)  ~ TYPE (TEXT) INTENS(HIGH) COLOR(RED) CAPS(OFF)  }
TYPE(TEXT)  COLOR(WHITE) HILITE(REVERSE) INTENS(HIGH)
  # TYPE(TEXT)  COLOR(WHITE) INTENS(LOW)
  \ TYPE(TEXT)  COLOR(GREEN) INTENS(HIGH)
)BODY WINDOW(69,15) EXPAND ($$)
+  $_$@ H e l p +$_$ 1 of 1
+
+ \The~PREB\service enables you to~rebind\plan(s) and package(s)  +
+ \or~conversion\plan(s) to package(s).                           +
+
+ #1,2:+The DSN subcommand REBIND PLAN/PACKAGE rebinds an
+       application plan/package when you make changes that
+       affect the plan/package (create a new index or RUNSTATS),
+       but do not change the SQL statements in the programs.
+
+ #3  :+Plan to Package Conversion
+       This service generates a batch job stream that will
+       convert your plans to packages.
+
                          }PF3: Return+
)INIT
  .HELP = REBPØØH
)PROC
  .HELP = REBPØØH
   &ZCONT = REBPØØH
)END
```

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

Bernard Zver
Database Administrator
Informatika Maribor (Slovenia)

DB2 news

Micro Focus has announced its new Mainframe Maintenance Solution, which provides transparent access to application code and data that are stored on the mainframe, LAN server, or workstation.

Programmers can compile, edit, debug, and test mainframe code on workstations without having to download the entire application. There are installation wizards and auto-configuration tools for setting up new maintenance projects. There's support for DB2, as well as COBOL, System/390 Assembler, CICS, IMS, CLIST, and JCL.

Mainframe compatibility means that applications can be tested in an emulated mainframe environment before they are put into final test and production on the host.

For further information contact:
Micro Focus, Speen Court, 7 Oxford Road, Newbury, Berks, RG14 1PB, UK.
Tel: (01635) 32646.
Micro Focus, 2465 E Bayshore Rd, Palo Alto, CA 94303, USA.
Tel: (650) 856 6134.
URL: http://www.microfocus.com.

* * *

Hitachi Data Systems has announced the Nucleus Exploration DB2 Accelerator, for carrying out *ad hoc* queries on DB2-based data warehouses without disrupting normal data warehouse operations.

According to HDS, running a typical DB2 workload's most resource-intensive queries on Nucleus can reduce the amount of mainframe system cycles used by up to 95%.

The accelerator is initially available for use on any OS/390-based mainframe running DB2 decision support systems.

For further information contact:
HDS, PO Box 54996, 750 E Central Expwy, Santa Clara, CA 95056-0996, USA.
Tel: (408) 970 1000.
HDS, Sefton Park, Stoke Poges, Bucks, SL2 4HD, UK.
Tel: (01753) 618000.
URL: http://www.hdshq.com.

* * *

IBM has announced plans for DB2 Universal Database for OS/390, including a customer beta programme set to begin in September.

The enhanced DB2 for OS/390 will get better Java support and access to multimedia data types such as image, text, audio, and video.

For further information contact your local IBM representative.

* * *

IBM and Baan are to port BaanERP for OS/390 running DB2, consolidating the ERP application and database servers on the one platform. First versions of BaanERP on OS/390 have already been installed as part of the beta test program.

For further information contact:
Baan, 4600 Bohannon Dr, Menlo Park, CA 94025, USA.
Baan International, Shield House, Elizabeth Way, Harlow, Essex, CM19 5AH, UK.
Tel: (01279) 445577.
URL: http://www.baan.com.