149

DB2 update

March 2005

In this issue

3 DB2 UDB V8 FP7 (V8.2) High Availability Disaster Recovery

10 Generating CREATE TABLE statements or DB2 LOADCARDs

20 Offline catalog runstats for online REORG – part 2

32 REXX/ISPF program to help programmers to resolve sqlcode -805

49 DB2 news

A xephon PUBLICATION

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.

Printed in England.

DB2 UDB V8 FP7 (V8.2) High Availability Disaster Recovery

High Availability Disaster Recovery (HADR) was introduced with DB2 UDB Fixpak 7. In a nutshell, it allows you to clone a database from one system to another and keep the databases in sync. I do not think this should be seen as a replacement for 'classic' High Availability (HACMP on AIX, Sun Cluster on Solaris, etc), but as a valuable addition – remember HACMP and HADR perform different functions. My thinking is that you would use HACMP for local recovery and HADR for remote recovery. (But you could, of course, use HADR for local recovery as well! I would use HADR for database recovery and HACMP for system recovery.)

Let's first look at the requirement and then how it works. The requirement is to be able to clone a database to another site and keep the two databases in sync as far as possible. What does this mean exactly? Well, HADR gives you three synchronization options – sync, nearsync, and async. The sync (synchronous) mode means that when a log flush to disk is performed on the primary system, it does not complete until it is processed on the standby system as well (which means that the log buffer is written to stable storage on the standby system). For nearsync (near synchronous) the process is just the same as for synchronous processing, except that the successful statement is sent back from the standby to the primary when the log has been received on the standby (but not yet written to stable storage). In async (asynchronous) mode, the primary does not wait for a confirmation from the standby that the log has been received, but just assumes that the log has been/will be sent when it has been put on the TCP/IP stack on the primary. (If the network is broken at this point, then the log buffer is lost.)

So which mode should you use? It all depends on how much data loss you are willing to tolerate if you need to switch

following a crash of the primary. If you want no data loss at all, you should choose sync mode. But there is no such thing as a free lunch! So if you choose this mode, you will have problems on the primary if the standby machine goes down or there is a problem with the network (because the primary will wait for confirmation that the standby has received the log and put it on stable storage). If you choose async mode, then if the standby crashes a fraction after the primary crashes, you have lost the information in the log that is being shipped. What are the chances of the primary and standby crashing at the same time? Only you can decide this and put the appropriate weight on your answer and then decide on the mode.

One more thing to remember is that the two systems containing the primary and standby databases have to be identical – for example two AIX 5.2 boxes. You cannot set up HADR between a Windows box and an AIX box, or an AIX box and a Sun box, or even a Windows 2000 box and a Windows 2003 box. The memory etc on both boxes should be roughly the same as well.

So how do you set up HADR? It is a very simple process. There is a GUI available from the Control Center (right-click on the database name and select High Availability Disaster Recovery), which takes you through the various steps. These are all detailed in the discussion below. The first time you set up HADR, try using the GUI because it takes you through all the steps and shows you all the commands. Then it is a relatively straightforward task to set up HADR on a system that doesn't have a GUI available.

The order of events is:

- Configure the primary database to have archive logging switched on.
- Make database configuration changes on the primary.
- Take a back-up of the primary database.
- Move the back-up file to the standby system.

- Restore the database on the standby system (keep it in roll forward mode).
- Make HADR database configuration changes on the primary.
- Make HADR database configuration changes on the standby.
- Update the services file on the standby.
- Update the services file on the primary.
- Define alternate servers on each system.
- Start HADR on the standby.
- Start HADR on the primary.

And that's it – 12 simple steps.

Let's go through the above steps in more detail and show the actual commands. Let's use two systems called SY1 (the primary) and SY2 (the standby) with a database called DB1 on SY1. The instance name on both systems is DB2 and the connection port is 50000. Also, the userid that applications will use to connect to the database is the same on both systems, as is the userid's password.

Configure the primary database to have archive logging switched on and make database configuration changes on the primary. The commands on SY1 are:

```
>db2 update db cfg for DB1 using logretain on
>db2 update db cfg for DB1 using logindexbuild on
>db2 update db cfg for DB1 using indexrec restart
```

You cannot set up HADR if you want to use circular or infinite logging.

Back up the primary database. The command on SY1 is:

```
>db2 backup db DB1 to <fs1>
```

Restore the database on SY2. You need to copy the back-up taken in the step above to SY2, or at least ensure that SY2 can

see <*fs1*>. Then the command on SY2 is:

```
>db2 restore db DB1 from <fs1> replace history file
```

Copy across any external UDFs and stored procedures from SY1 to SY2.

This is an important manual process – and don't forget that if you update/create any UDFs or stored procedures on the primary after this step then you have to manually copy them to the standby.

Catalog the database (which we restored on SY2) on SY1. The command on SY1 is:

```
>db2 catalog tcpip node hm remote <ipaddr-of-SY2> server 50000
remote_instance DB2

>db2 catalog db DB1 as DB1R at node hm
```

Make HADR database configuration changes on the primary/ standby.

The database configuration parameters you need to update are:

HADR_LOCAL_HOST, HADR_LOCAL_SVC, HADR_REMOTE_HOST, HADR_REMOTE_SVC, HADR_REMOTE_INST, HADR_SYNCMODE, and HADR_TIMEOUT.

Each of these is fully described in the *Performance Admin Guide*.

Update the services file on the primary/standby.

You need to update the services file on both systems with the values you specified in HADR_LOCAL_SVC and HADR_REMOTE_SVC.

Define alternate servers on each system. The command is:

```
>db2 update alternate server for database DB1 using hostname <ipaddr>
port 50000
```

On SY1 you would specify <*ipaddr*> as the IP address of SY2

and on SY2 you would specify the IP address of SY1.

Start HADR on the standby. Do this by using the command:

```
>db2 start hadr on database DB1 as standby
```

Start HADR on the primary. Do this by using the command:

```
>db2 start hadr on database DB1 as primary
```

You have now finished setting up HADR – but how can we tell that it's running? One of the ways of doing this is to issue a **get snapshot** command and look at the last section of the output. You can do this on both systems.

```
>db2 get snapshot for db on DB1
```

The output below is produced when the command is issued on the primary (SY1) system. You can see that the *Role* says Primary and the *State* is Peer, which means that HADR is up and running. On the standby system (SY2) the *Role* will say Standby and the *State* will say Peer.

```
HADR Status
  Role                   = Primary
  State                  = Peer
  Synchronization mode   = Nearsync
  Connection status      = Connected , xxxxxxxx
  Heartbeats missed      = 0
  Local host             = yyyyyyyyy
  Local service          = DB2_HADR1
  Remote host            = zzzzzzzzz
  Remote service         = DB2_HADR2
  Remote instance        = DB2
  timeout(seconds)       = 120
  Primary log position(file, page, LSN) = S0000000.LOG, 0,
00000000007D0000
  Standby log position(file, page, LSN) = S0000000.LOG, 0,
00000000007D0000
  Log gap running average(bytes) = 0
```

You can now perform updates etc on the primary system and the logs will be shipped to the standby system.

At some point you will want to switch (failover) the primary and standby. You do this using the TAKEOVER HADR command. In a 'normal' situation where you want a controlled failover,

simply issue the following command on the standby system (SY2):

```
>db2 takeover hadr on db DB1
```

You should get the following message back:‘DB20000I The TAKEOVER HADR ON DATABASE command completed successfully.’ If you now issue the **get snapshot** command on both systems you will see that SY2 is now the primary and SY1 is the standby.

So if we had a client connected to the primary system SY1, do we have to do anything after we failover? The answer is no! We don’t have to do anything on the client because we are using automatic client reroute, we defined alternate servers on each system (pointing to each other), and the connecting userid and password are the same on both systems. If a client was connected to DB1 on SY1 and was issuing **select** commands when the TAKEOVER HADR command was issued, we do not have to issue another **connect** command to DB1, but just have to re-issue the **select** command for it to connect to the new primary on SY2. Your application would have to be coded to detect SQL30108N messages (these say that a connection has failed but has been re-established) and then re-issue the **select** command.

If you now check the status of the database on each system, you will see that the database on SY1 is now in standby mode and on the SY2 it is in primary mode. If you want to failover again, simply issue the TAKEOVER HADR command on the ‘new primary’, which in our case would be SY2, and you will be back to the original set-up.

What happens if you pull the plug on the SY1 system? There is no automatic switch-over to the standby – you have to issue the TAKEOVER HADR command manually on the standby with the BY FORCE option. If you do not use this option, the standby will wait for the primary to send a message saying that it can take over the database (which it never will because you pulled the plug!); hence the use of the BY FORCE option. If

you issue the TAKEOVER HADR command and forget the BY FORCE option, you can simply re-issue the command with the option.

Should you write a script to poll the primary and if it isn't there issue the TAKEOVER HADR command automatically? The simple answer is no. If there were to be a network glitch between the primary and standby, thus making the standby think that the primary was no longer available, but the network between the client and primary (on a different network) was fine, and if the script issued the TAKEOVER HADR BY FORCE command, then SY2 would think that *it* was the primary database – but SY1 would still be running as the primary. In this situation you would have two primary databases – not good because updates will be happening against both! For this reason I would suggest that the TAKEOVER HADR command be issued manually – the downtime that will occur until someone makes the decision to switch and issue the command will be a lot less than the time required to sort out the situation with two primary databases!

If you use the BY FORCE option, the database on SY1 is no longer in standby status after the takeover – you have to resync the databases before switching back.

Another point to remember is that when you set up the HADR environment, the database on the standby system is in roll forward pending state. Therefore you cannot check on your data until you switch over. It is therefore important to monitor the LOG GAP value of the **get snapshot** command to check that it is at a value you are comfortable with.

One other point to remember is that if you load a table on the primary with the COPY NO option, that table is marked as non-recoverable on the standby – you need to specify the COPY YES option. So what can you do to stop people running load jobs with COPY NO? DB2_LOAD_COPY_NO_OVERRIDE is a new registry variable that will override a COPY NO and treat it as a COPY YES. If you are going to use LOAD with COPY YES on the primary, you need to make available to the standby

the directory to which you write the copy. What happens is this: when the standby processes the log with the LOAD command, it looks for the directory specified in the LOAD statement. If it cannot find it, it will place the tablespace on the standby in non-recoverable mode. You will only find this out when you failover! Therefore you need tight procedures in place if you are going to use the LOAD command on the primary database.

You should also have procedures in place to copy any UDFs and stored procedures from the primary to the standby systems when they are created. Failure to do this will mean that they are unavailable when you switch systems!

HADR is a new and valuable tool in the high availability game, but its implementation and continued use must be carefully planned. This is especially true with regard to how UDFs and stored procedures are maintained on both systems and whether the LOAD process should be allowed on the primary. But check it out!

C Leonard
Freelance Consultant (UK)

Generating CREATE TABLE statements or DB2 LOADCARDs

How often do you have to make a DB2 copytable from a VSAM or SAM file for error tracking or *ad hoc* reporting purposes?

This REXX generates a CREATE TABLE statement or a DB2 LOADCARD from a COBOL or PL/I copybook.

To generate a CREATE TABLE statement:

1 Place the copybook in TSO edit mode.

2 Type @DB2LOAD TBL in the command line.

To generate a LOADCARD:

1 Place the copybook in TSO edit mode.

2 Type @DB2LOAD in the command line

When you enter **@DB2LOAD ?** you will see an explanation of the parameters.

This application uses:

- @DB2LOAD – a REXX program
- DB2MLOAD – a REXX program
- POPUP – a panel.

Bear in mind the following:

- Occurs clauses must be solved manually by repeating the field *n* times.
- Only the first 18 positions of a fieldname will be used in the column-name.
- Hyphens in the field-name will be changed to underscores in the column name.
- COMP-3 results in DECIMAL.
- COMP results in DECIMAL.
- PIC X(n) results in CHAR(n).
- PIC 9(n) results in DECIMAL(n).
- When you want to convert datefields to DB2 DATE:
  - change variable DB2_DATE to ‘Y’
  - fieldname must contain DAT
  - fields have the following layout: CCYY-MM-DD.

@DB2LOAD

```
/* Rexx ************************************************************/
/*                                                                 */
```

```
/* Name   : @DB2LOAD                                                  */
/*                                                                    */
/* Purpose: Generate DB2 LOADCARD or CREATE TABLE statement from a    */
/*          PL/I or COBOL record definition.                          */
/*          Program is based on concern standards.                    */
/*                                                                    */
/* Author   : Loet Polkamp                                            */
/*                                                                    */
/* Sub-routines : DB2MLOAD                                            */
/*            * Change - to _                                         */
/*            * Delete Filler statements                              */
/*                                                                    */
/* Parameter: TBL=Generate CREATE TABLE statement                     */
/*            HIS=Generate LOAD STMT for partitioned ts               */
/*            no parm=Generate LOAD STMT for non-partitioned ts       */
/*                                                                    */
/* Change history:                                                    */
/********************************************************************/
Trace o
"ISREDIT MACRO (parm)"
upper parm
If parm='?' Then do
   Call help
   Exit
   End
/* ************ MAIN line ************************************ */
Call Init        /* initialize                        */
Call Der_name    /* derive table(space)name           */
Call PoPup 'DB2-LOAD statements will be generated for 'mbrname
Call Language    /* PL/I or COBOL source ?            */
Call Alloctmp    /* allocate tempfile                 */
Call Gen_stmt    /* generate LOAD/CREATE statements   */
Call Read_fields /* Read fields from copybook         */
Exit
/* ************ SUB routines ********************************** */
Init:
   DB2_date='N' /* convert date fields to DB2-DATE format?  */
   packed='N'
   dbname='MYDBASE'
   tbcr='MYCREATOR'
   count=Ø
   prefix='     ('
   linenmbr=Ø
   firstline=1
   recpos=1
   comment=''
return
Language:   /* Determine language: COBOL or PL/I  */
   "Isredit reset"
   "Isredit bnds"
```

```
   "Isredit find ' dcl '"
   If rc=Ø Then do
      language='PL1'
      /* convert decimal scale */
      "Isredit cha all 9V99 ',2)'"
      "Isredit cha all ''' ''  "
      End
      Else language='COB'
return
Der_name:  /* derive table(space)name */
   "ISREDIT (mbrname)=MEMBER"
   "ISREDIT (dsname)=DATASET"
   If mbrname=''
      Then do
           mydot=pos('.',dsname,1)
           x=pos('.',dsname,mydot+1)
           if x=Ø Then x=length(dsname)
                    Else x=x - mydot - 1
           mbrname=substr(dsname,mydot+1,x)
           tbcr= userid()
           tsname=substr(mbrname,1,2) || 'SØØ1'
           tbname=substr(mbrname,1,2) || 'TØØ1'
      End;
      Else do
         tsname=substr(mbrname,1,2) || 'S' || substr(mbrname,4,2) || 'Ø'
         tbname=substr(mbrname,1,2) || 'T' || substr(mbrname,4,2) || 'Ø'
      End;
Return
Conv_fld:   /* Convert field to column  */
   /*  Ignore comment line   */
   startpos=pos('*',datarec,7)      /* COBOL  */
   if startpos=7 Then return
   startpos=pos('/*',datarec,1)     /* PL/I   */
   if startpos=2 Then return
   datarec=substr(datarec,7,74)
   if language='PL1' Then call Read_pl1
   if language='COB' Then
      Parse var datarec level colname pic def comp
   if def='' Then return            /* no datadef record  */
   level=word(level,1);
   if level=88 then return
   pic=word(pic,1);
   pic=strip(pic);
   if pic=''               Then return  /* subdivision */
   if pic='.'              Then return  /* subdivision */
   pos=pos('REDEF',datarec,1)
   if pos>Ø Then call redefine
   if substr(colname,1,6)='FILLER' Then return /* free space*/
   pos=pos('OCCUR',datarec,1)
   comment=''
```

```
 if pos>0 Then Queue '      >>>>>>>> OCCURS 'def' statement <<<<<<<<'
 comp=word(comp,1);
 comp=strip(comp);
 colname=word(colname,1);
 colname=left(colname,18);
 def=strip(word(def,1));
 If substr(def,length(def),1)='V'
    Then def=substr(def,1,length(def)-1)
 If substr(def,length(def),1)='.'
    Then def=substr(def,1,length(def)-1)
 packed='N'
 If (substr(comp,1,4)='COMP' | substr(comp,1,4)='PACK')
    Then packed='Y'
 type=substr(def,1,1)
 If DB2_date='Y'
 Then do
      pos=pos('DT',colname)
      if (pos>0) Then type='D'
      pos=pos('DAT',colname)
      if (pos>0) Then type='D'
 End;
 call calc_length
 Select
   when type='D' Then call date
   when type='A' Then call alfa
   when type='X' Then call alfa
   when type='9' Then call num
   when type='S' Then call num
   otherwise call unknown
 End;
 /* Compile together LOAD-statement     */
count=count+1
If count>1 Then prefix='     ,'
recpos=right(recpos,5,' ')
txtlen=right(txtlen,8,' ')
If parm='TBL'
Then do    /* generate CREATE TABLE   */
    Queue  prefix || colname || ' ' || ,
           datatype || txtlen || ,
           ' NOT NULL WITH DEFAULT'
End;
Else do    /* generate LOAD DATA      */
    If packed='Y' Then
    Queue  prefix || colname' POSITION('recpos') ' ,
           datatype
    Else do
         if type='D' Then txtlen=' (10)'
         Queue  prefix || colname' POSITION('recpos') ' ,
                datatype || external || txtlen || ,
                comment
```

```
      End;
   End;
   recpos=recpos+itemlen
   If linenmbr=lregel Then Call closing
Return
Calc_length:
   If type='D'
   Then do
        txtlen='    '
        itemlen=10
        return
   End;
   /* split precision/scale  */
   scale_pos=pos('V',def)
   If   scale_pos>0 Then do  /*  definition contains decimals  */
        beg_prec=1
        aan_prec=scale_pos-1
        prec=substr(def,beg_prec,aan_prec)
        /* determine scale length */
        beg_scale=scale_pos+1
        aan_scale=length(def)-scale_pos
        scale=substr(def,beg_scale,aan_scale)
        pos=pos('(',scale)
        If  pos>0 Then do   /* scale between hooks */
            begin=pos+1
            eind end=pos(')',scale)-begin
            scale=substr(scale,begin,eind);
        End;
        Else scale=length(scale)    /* scale not between hooks */
   End;
   Else do
        prec=def /* no decimals */
        scale=0
   End;
   /* determine length precision  */
   pos=pos('(',prec)
   If  pos>0 Then do    /* precision between hooks */
       begin=pos+1
       eind=pos(')',prec)-begin
       prec= substr(prec,begin,eind);
   End;
   Else prec=length(prec) /* precision not between hooks */
   If language='PL1' Then nop
                     Else prec=prec+scale
   itemlen=prec
   txtlen=itemlen
   If packed='Y' Then itemlen=trunc((prec) / 2) + 1
   If scale_pos>0
      Then txtlen='(' || prec || ',' || scale || ')'
      Else txtlen='(' || txtlen || ')'
```

```
return
Alfa:
   external=''
   datatype='CHARACTER'
return
Num:
   datatype='DECIMAL  '
   If packed='Y' Then external=''
                 Else external='EXTERNAL'
return
Date:
   datatype='DATE     '
   external='EXTERNAL'
return
Unknown:  /* unknown datatype */
   If language='PL1'
   Then do
      nop
   End;
   Else datatype='Unknown  '
   external=''
return
Popup:
  parse arg text
  text=Center(Strip(text),5Ø)
  "ISPEXEC CONTROL DISPLAY LOCK"
  "ISPEXEC ADDPOP  ROW(9)  COLUMN(12)"
  "ISPEXEC DISPLAY PANEL(POPUP)"
  "ISPEXEC REMPOP"
return
Alloctmp:
 "ALLOC FI(DB2LOAD) SPACE(5 5) TRACKS BLKSIZE(616Ø) LRECL(8Ø) REUSE",
        "RECFM(F B) DSORG(PS)"
 "ISPEXEC LMINIT DATAID(DID1) DDNAME(DB2LOAD)"
return
Edittmp:
 "ISPEXEC EDIT DATAID("DID1") macro(DB2MLOAD)"
 "ISPEXEC LMFREE DATAID("DID1")"
return
Redefine: /* Reuse the definition of the previous field */
   firstline=firstline+1
   linenmbr=linenmbr+1
   "Isredit (datarec)="line firstline
   Parse var datarec xxxname pic def comp
   Queue '      >>>>>>>> Redefines statement <<<<<<<<'
return
Read_pl1:   /* pl1 syntax converteren naar cobol */
  Parse var datarec level colname def init last
  fix=def
  If def='' Then return
```

```
  upper def
  If strip(def)='FIXED' Then def=init
  datatype=strip( substr(def,1,3 ) )
  beg=pos(" ",def,1)
  last=strip( substr(def,beg+1,length(def)) )
  /*   remove rubbish  */
  beg=pos(";",def,1)
  If beg > 0 Then def=substr(def,1,beg-1)
  beg=pos("(",def,1)
  If beg > 0 Then do
     def=substr(def,beg,length(def))
     def=strip(def)
     end=pos(')',def,1)
     def=substr(def,1,end)
     def=strip(def)
     comma=pos(',',def,1)  /* position comma   */
     If comma>0 Then do /* convert decimals */
        scale=substr(def,comma+1,length(def)-comma-1)
        scale=strip(scale)
        scale='V9(' || scale || ')'
        def=substr(def,1,comma-1) || ')' || scale
     End;
  End;
  Else do
       beg=pos("'",def,1)
       If beg>0 Then do
          def=strip(def)
          def=substr(def,beg+1,length(def))
          end=pos("'",def,1)
          def=strip( substr(def,1,end-1) )
          def='(' || length(def) || ')'
       End;
  End;
/*  FIXED BIN(15): SMALLINT */
/*  FIXED BIN(31): INTEGER  */
  Select
    when datatype ='' Then return      /*  no datadef record  */
    when fix='FIXED' & init='BIN(31),' Then do
         def='X(4)'
         End;
    when fix='FIXED' Then do
         def='S9' || def
         comp='COMP-3'
         End;
    when datatype='DEC' Then do
         def='9' || def
         End;
    when datatype='PIC' Then do
         datatype='DEC'
         def='S9' || def
```

```
          End;
     when datatype ='CHA' Then do
          def='X' || def
          End;
     otherwise datatype='Unknown datatype '
  End;
Return;
Closing:
  ltoken='     )'
  If parm='TBL'
  Then do    /* generate CREATE TABLE   */
      Queue '     ) IN 'dbname'.'tsname
      ltoken=';'
  End;
  Queue ltoken
  If parm='TBL'
  Then do    /* generate COMMENT statement */
    Queue 'COMMENT ON TABLE ' || tbcr || '.' || tbname || ' IS'
    Queue '   ''' || mbrname || ': Generated by @DB2LOAD  '''
    Queue ';'
  End;
  /* Write queue to tempfile */
  "execio" queued() "diskw db2load (finis)"
  Call edittmp
  exit
Gen_stmt:
   "Isredit (current) =cursor";
   "Isredit cursor=.zlast";
   "Isredit (lregel) =cursor";
   "Isredit cursor=.zfirst";
   If parm='TBL'
   Then Queue  'CREATE TABLE ' || tbcr || '.' || tbname
   Else do
   If parm='HIS'
      Then ,
    Queue 'LOAD DATA RESUME YES LOG NO SORTDEVT WORK SORTNUM 8 SORTKEYS'
      Else ,
    Queue 'LOAD DATA REPLACE    LOG NO SORTDEVT WORK SORTNUM 8 SORTKEYS'
   Queue '     ENFORCE NO INTO TABLE ' || tbcr || '.' || tbname
   If parm='HIS' Then Queue    '     PART ** REPLACE'
   End;
Return
Read_fields:  /*  READ RECORD   */
   Do firstline=1 to lregel
      comp=''
      linenmbr=linenmbr+1
      "Isredit (datarec)="line firstline
      Call Conv_fld firstline
      If firstline>=lregel Then Call closing
      End;
```

```
Return
Help:
  "ISREDIT reset";
  "ISREDIT LINE_BEFORE 1=MSGLINE ",
  "'Usable parameters:'";
  "ISREDIT LINE_BEFORE 1=MSGLINE ",
  "'  @DB2LOAD    : generate 'DB2-loadcard' for non-partitioned space'
";
  "ISREDIT LINE_BEFORE 1=MSGLINE ",
  "'  @DB2LOAD HIS: generate 'DB2-loadcard' for partitioned space'";
  "ISREDIT LINE_BEFORE 1=MSGLINE ",
  "'  @DB2LOAD TBL: generate 'CREATE TABLE' statement'";
Return
Error:
Say 'Recordnumber: ' linenmbr
Say 'Returncode ' rc ' on line: 'SIGL
exit;
```

DB2MLOAD

```
/* Rexx ************************************************************/
/*                                                                 */
/* Name   : DB2MLOAD                                               */
/* Purpose: Change - to _                                          */
/* Author : Loet Polkamp                                           */
/* Main program: @DB2LOAD                                          */
/* Parameter :no                                                   */
/*                                                                 */
/* Change history:                                                 */
/*******************************************************************/
trace o
"ISREDIT MACRO"
"Isredit x filler all";
"Isredit del all x";
"Isredit cha '-' '_' all";
"Isredit reset";
"Isredit cursor = .zfirst 0";
```

POPUP

```
)ATTR
 [ TYPE(OUTPUT) INTENS(HIGH) CAPS(OFF) JUST(ASIS)
)BODY WINDOW(50,3)
+
[text
)END
```

Loet Polkamp
Database Administrator (The Netherlands)

Offline catalog runstats for online REORG – part 2

This month we conclude the code for this article.

```
   /*--  Write output on dasd and sysout      --*/
       partloop = Ø ; £g = Ø
       do £i = 1 to £f618 ; w1a = left(a1a.£i,18,' ') ;  w1b =
left(a1b.£i,5,' ') ;
          w1c = right(a1c.£i,3,' ') ;  w1d = ' | '
          w2a = left(a2a.£i,27,' ') ;  w2b = left(a2b.£i,5,' ')
          w2c = right(a2c.£i,3,' ') ;  w2d = ' | '
          w3 = left(a3.£i,2,' ')    ;  w4 = left(a5.£i,2,' ')
          w5 = left(a6.£i,2,' ')    ;  w5a = '| '
          w6 = left(a4.£i,2,' ')    ;  w7 = right(a7.£i,11,' ')
          TSopc  = translate(w1a,' ','.') ; TSopc1 = word(Tsopc,2)
          TSopc2 = strip(substr(TSopc1,4,5)) ; sj.£i  = TSopc2
sk.£i =
'|'w1a||w1b||w1c||w1d||w2a||w2b||w2c||w2d||w3||w4||w5||w5a||w6'|'w7'  |'
          say sk.£i
          if w1c ¬= blk then do;partloop = Partloop + 1 ; TempTS =
TSopc1 ; end ; end
   /*--  Write Reorg members                  --*/
       if Sireo = 'Y' then do
        ctreo = ctreo + 1 ; Wctreo = right(ctreo,3,'Ø') ; jobw = fireoin
          sk.Ø = £i ; Nrec = '*' ; call WriteRec
           sk.1='+-------------------------+--------------------------
---------+----------+-----------+'
 sk.Ø = 1 ; jobw = fireoin ; Nrec = '*' ; call WriteRec ; jobw = fireoop
       "alloc da('"outreopc"(J"AcrPro"R"Wctreo")') f("jobw") shr reuse"
          TSopc3 = sj.1 ; if Partloop = 1 then do
sk.1='//J'AcrPro'R'Wctreo' JOB ('$accn'),''Reorg
Online'',CLASS='$class', '
sk.2='//      MSGCLASS='$msgcla',USER='$user',REGION='$region',       '
sk.3='//      MSGLEVEL=('$msglvl'),NOTIFY='$notif',                   '
sk.4='//      COND=(5,LT)                                             '
sk.5='/*JOBPARM BYTES=999999,LINES=9999                               '
sk.6='//* ------------------------------------------------- *       '
sk.7='//*  Table : 'left(Dsnu614_TbName,27,' ')'             *       '
sk.8='//*  Card  : 'left(Dsnu614_Card,21,' ')'               *       '
sk.9='//* ------------------------------------------------- *       '
sk.1Ø='//DB2PROC  JCLLIB ORDER=('$proclib')                           '
sk.11='//* ------------------------------------------------- *      '
             if Dsnu614_Card > 15ØØØØØØ then
sk.12='//REO'TSopc3' EXEC 'AcrProg'REOT,TAB='Tempts
             else
sk.12='//REO'TSopc3' EXEC 'AcrProg'REOD,TAB='Tempts
sk.13='//* ------------------------------------------------- *      '
```

```
          jobw = fireoop;Nrec = '13' ; call WriteRec ; temptab =
substr(tempts,4,5)
        "ispexec lminit   dataid("inp1") dataset('"outreopc"') enq(shr)"
          "ispexec lmmstats dataid("inp1")
                      member(J"AcrPro"R"Wctreo") user("temptab")" ; end
          else do
   /*--  Partitioned Tablespace               --*/
              do £c = 1 to Partloop
                 if sireo6181.£c = 'Y' |,
                    sireo6182.£c = 'Y' |,
                    sireo613.£c  = 'Y' |,
                    sireo617.£c  = 'Y' then nop
                 else iterate ; £g = £g + 1
   /*--  Partitioned TS without sec. index    --*/
                 if £f618 = Partloop then do
   /*--  Max number of job for partitioned TS --*/
                    if ctreom > 3 then ctreom = Ø
                    ctreom = ctreom + 1 ; Wctreom = right(ctreom,2,'Ø')
sk.£g ='//J'AcrPro'RP'Wctreom' JOB ('$accn'),''Reorg
Online'',CLASS='$class', '
                    end
   /*--  Partitioned TS with secondary index  --*/
                 else
sk.£g ='//J'AcrPro'R'Wctreo' JOB ('$accn'),''Reorg
Online'',CLASS='$class', '
                 £g = £g + 1
sk.£g ='//       MSGCLASS='$msgcla',USER='$user',REGION='$region',
'
                 £g = £g + 1
sk.£g ='//       MSGLEVEL=('$msglvl'),NOTIFY='$notif',
'
                 £g = £g + 1
sk.£g ='//       COND=(5,LT)
'
                 £g = £g + 1
sk.£g ='/*JOBPARM BYTES=999999,LINES=9999
'
                 £g = £g + 1
sk.£g ='//* ------------------------------------------------    *   '
                 £g = £g + 1
sk.£g ='//*  Table : 'left(Dsnu614_TbName,27,' ')'               *   '
                 £g = £g + 1
sk.£g ='//*  Card  : 'left(Dsnu618_Card.£c,21,' ')'              *   '
                 £g = £g + 1
sk.£g ='//*  Part  : 'left(£c,3,' ')' di 'left(Partloop,4,' ',)'  *   '
                 £g = £g + 1
sk.£g ='//* ------------------------------------------------ *       ,
                 £g = £g + 1
sk.£g ='//DB2PROC  JCLLIB ORDER=('$proclib')
'
```

```
                £g = £g + 1
sk.£g ='//* ------------------------------------------------ *       '
                £g = £g + 1
sk.£g ='//REO'TSopc3' EXEC 'AcrProg'REDP,TAB='Tempts',PART='£c
                £g = £g + 1
sk.£g ='//* ------------------------------------------------ *       '
                end             /* End DO £c */
              if £f618 ¬= Partloop then do
                £hPunt = Partloop + 1 ; do £h = £hPunt to £f618
           £hctr = £hctr + 1 ; Wctrix = right(£hctr,2,'Ø') ;£g = £g + 1
sk.£g ='//J'AcrPro'RI'Wctrix' JOB ('$accn'),''Reorg
Online'',CLASS='$class', '
                   £g = £g + 1
sk.£g ='//       MSGCLASS='$msgcla',USER='$user',REGION='$region',
'
                   £g = £g + 1
sk.£g ='//       MSGLEVEL=('$msglvl'),NOTIFY='$notif',TYPRUN=HOLD,
'
                   £g = £g + 1
sk.£g ='//       COND=(5,LT)
'
                   £g = £g + 1
sk.£g ='/*JOBPARM BYTES=999999,LINES=9999
'
                   £g = £g + 1
sk.£g ='//* ------------------------------------------------ *       '
                   £g = £g + 1
sk.£g ='//*  Index : 'left(a2a.£h,27,' ')'                *'
                   £g = £g + 1
sk.£g ='//*  Card  : 'left(Dsnu614_Card,21,' ')'                     *'
                   £g = £g + 1
sk.£g ='//* ------------------------------------------------ *       '
                   £g = £g + 1
sk.£g ='//DB2PROC  JCLLIB ORDER=('$proclib')
'
                   £g = £g + 1
sk.£g ='//* ------------------------------------------------ *       '
                   £g = £g + 1 ; IXfull = translate(a2a.£h,' ','.')
                   IXsub  = word(IXfull,2)
sk.£g ='//REO'TSopc3' EXEC 'AcrProg'REIX,TAB='IXsub
                   £g = £g + 1
sk.£g ='//* ------------------------------------------------ *       '
                 end ; end ; jobw = fireoop ; Nrec = £g ; call WriteRec
              temptab = substr(tempts,4,5)
          "ispexec lminit   dataid("inp1") dataset('"outreopc"') enq(shr)"
               "ispexec lmmstats dataid("inp1")
                        member(J"AcrPro"R"Wctreo") user(**"temptab")"
              end               /* End Else do Partloop */
            end
         Sireo  = 'N' ; £d = £d - 2
```

```
       end                        /* End if MsgSel = 'DSNU050I        */
    end                           /* End DO £d */
 say '+-------------------------+-----------------------------------+-
------+--+------------+'
 say '' ;  call Free ;  say '' ; say 'Tablespace to be Reorg n° 'ctreo
 say '' ; say '' ; jobw = fireoop
 "alloc da('"outreopc"(J"AcrPro"RERU)') f("jobw") shr reuse"
 Anno  = substr(date(s),1,4) ; Mese  = substr(date(s),5,2)
 Giorno= substr(date(s),7,2) ; Wday  = Anno'-'Mese'-'Giorno
sk.1='//J'AcrPro'WRUN JOB ('$accn'),''Reorg Online'',CLASS='$class',  '
sk.2='//      MSGCLASS='$msgcla',USER='$user',REGION='$region',       '
sk.3='//      MSGLEVEL=('$msglvl'),NOTIFY='$notif',                  '
sk.4='//      COND=(5,LT)                                            '
sk.5='/*JOBPARM BYTES=999999,LINES=9999                              '
sk.6='//* ------------------------------------------------- *       '
sk.7='//*  Runstats Tablespace not reorganized               *       '
sk.8='//* ------------------------------------------------- *       '
sk.9='//DB2PROC  JCLLIB ORDER=('$proclib')                           '
sk.10='//* ------------------------------------------------- *       '
sk.11='//STEP00   EXEC   DB2REXX1                                    '
sk.12='//REXX00.SYSTSIN  DD  *                                       '
sk.13='  ISPSTART CMD($DB2RUN1 +                                    '
wr=subsys','datab',*,NO,'Wday'-
00.00.00.000000,'AcrProg','Parallel','Crbind')'
sk.14='  'wr
sk.15='//* ------------------------------------------------- *      '
 jobw = fireoop ;  Nrec = '15' ;  call WriteRec
 if CRbind ¬= '*' then do
   /*--  Rebind job                          --*/
    jobw = fireoop ; "alloc da('"outreopc"(J"AcrPro"RBND)') f("jobw")
shr reuse"
sk.1='//J'AcrPro'WRBN JOB ('$accn'),''Rebind Totale'',CLASS='$class', '
sk.2='//      MSGCLASS='$msgcla',USER='$user',REGION='$region',       '
sk.3='//      MSGLEVEL=('$msglvl'),NOTIFY='$notif
sk.4='/*JOBPARM BYTES=999999,LINES=9999                              '
sk.5='//* ------------------------------------------------- *       '
sk.6='//*  Create Rebind job                                 *       '
sk.7='//* ------------------------------------------------- *       '
sk.8='//JOBLIB   DD DISP=SHR,DSN='$dsnload
sk.9='//DB2PROC  JCLLIB ORDER=('$proclib')                           '
sk.10='//* ------------------------------------------------- *      '
sk.11='//DELVIEW  EXEC PGM=IKJEFT01,DYNAMNBR=20                      '
sk.12='//SYSTSPRT  DD  SYSOUT=*                                      '
sk.13='//SYSPRINT  DD  SYSOUT=*                                      '
sk.14='//SYSTSIN   DD  *                                             '
sk.15='  DSN SYSTEM('subsys')                                        '
sk.16='  RUN PROGRAM('$tep2pgm') PLAN('$tep2pln') LIB('''$runlib''')  '
sk.17='//SYSIN    DD  *                                              '
sk.18='  DROP VIEW V'CRbind'_RBPLAN  ;                               '
sk.19='  DROP VIEW V'CRbind'_RBPACK  ;                               '
```

```
sk.20='//* ------------------------------------------------- *      '
sk.21='//LAB0     IF (DELVIEW.RC LT 9) THEN                        '
sk.22='//CREVIEW  EXEC PGM=IKJEFT01,DYNAMNBR=20                    '
sk.23='//SYSTSPRT  DD  SYSOUT=*                                    '
sk.24='//SYSPRINT  DD  SYSOUT=*                                    '
sk.25='//SYSTSIN   DD  *                                           '
sk.26='  DSN SYSTEM('subsys')                                      '
sk.27='  RUN PROGRAM('$tep2pgm') PLAN('$tep2pln') LIB('''$runlib''')  '
sk.28='//SYSIN     DD  *                                           '
sk.29='                                                            '
sk.30='   CREATE VIEW V'CRbind'_RBPLAN (PLAN) AS                   '
sk.31='   SELECT  NAME                                             '
sk.32='           FROM SYSIBM.SYSPLAN                              '
sk.33='           WHERE NAME     LIKE ''%''      AND               '
sk.34='                 CREATOR  =    '''CRbind''' ;              '
sk.35='                                                            '
sk.36='   CREATE VIEW V'CRbind'_RBPACK (PACKAGE,COLLECT,VERS) AS    '
sk.37='   SELECT  NAME,COLLID,VERSION                              '
sk.38='           FROM SYSIBM.SYSPACKAGE                           '
sk.39='           WHERE COLLID   LIKE ''%''      AND               '
sk.40='                 NAME     LIKE ''%''      AND               '
sk.41='                 OWNER    =    '''CRbind''' ;              '
sk.42='//* ------------------------------------------------- *      '
sk.43='//LAB1      IF (CREVIEW.RC EQ 0) THEN                       '
sk.44='//REXX00   EXEC   DB2REXX1                                  '
sk.45='//REXX00.SYSTSIN  DD  *                                     '
sk.46='  ISPSTART CMD($DB2RBND
'subsys','CRbind',NO,J'AcrPro'RBND,'datab')'
sk.47='//LAB0END  ENDIF                                            '
sk.48='//LAB1END  ENDIF                                            '
sk.49='//* ------------------------------------------------- *      '
    Nrec = ,49' ;  call WriteRec ; end
   /*--  Job $$$COIBM Member                  --*/
 jobw = fireoop ; "alloc da('"outreopc"($$$COIBM)') f("jobw") shr reuse"
sk.1='   DO NOT ERASE THIS MEMBER                                  '
 Nrec = '1' ;  call WriteRec
   /*--  Job Ipoupdte                         --*/
 jobw = fireoop ; "alloc da('"outreopc"(J"AcrPro"UPDT)') f("jobw") shr
reuse"
sk.1='//J'AcrPro'UPDT JOB ('$accn'),''Ipoupdate'',CLASS='$class',     '
sk.2='//      MSGCLASS='$msgcla',USER='$user',REGION='$region',      '
sk.3='//      MSGLEVEL=('$msglvl'),NOTIFY='$notif',                  '
sk.4='//      COND=(5,LT)                                          '
sk.5='/*JOBPARM BYTES=999999,LINES=9999                            '
sk.6='//* ------------------------------------------------- *      '
sk.7='//*  Ipoupdate library job                           *       '
sk.8='//* ------------------------------------------------- *      '
sk.9='//STEP01  EXEC PGM=IPOUPDTE,                                 '
sk.10='//            PARM='UPDATE'                                 '
sk.11='//SYSPRINT DD  SYSOUT=*                                     '
```

```
sk.12='//$INST    DD  DISP=SHR,DSN='outreopc
sk.13='//SYSIN    DD  DATA,DLM=$$                                          '
sk.14='</*                                                                 '
sk.15='USER='$user'<+                                                      '
sk.16=' USER=???????<                                                      '
sk.17='NOTIFY='$user'<+                                                    '
sk.18=' NOTIFY=???????<                                                    '
sk.19='MSGCLASS=2<+                                                        '
sk.2Ø=' MSGCLASS=?<                                                        '
sk.21='CLASS=A<+                                                           '
sk.22=' CLASS=?<                                                           '
sk.23='</*                                                                 '
 Nrec = '23'
 call WriteRec
   /*--  Delete file outreoin for no Reorg    --*/
 if ctreo = Ø then do
    "free fi(fireoin)" ; address tso "delete '"outreoin"'" ; end ; exit
   /*--   Routine for msg. DSNU613I           --*/
 Msg_DSNU613I :
    £d = £d + 1 ; Dsnu613_Card = word(outruns.£d,1)
    if Dsnu613_Card = 'CARD' then Dsnu613_Card = word(outruns.£d,3)
    £d = £d + 1 ; Dsnu613_Nearindref = word(outruns.£d,1)
    if Dsnu613_Nearindref = 'NEARINDREF' then Dsnu613_Nearindref =
word(outruns.£d,3)
    £d = £d + 1 ; Dsnu613_Farindref = word(outruns.£d,1)
    if Dsnu613_Farindref = 'FARINDREF' then Dsnu613_Farindref =
word(outruns.£d,3)
    £d = £d + 2 ; Dsnu613_Percdrop = word(outruns.£d,1)
    if Dsnu613_Percdrop = 'PERCDROP' then Dsnu613_Percdrop =
word(outruns.£d,3)
    reo613 = 'N' ; if Dsnu613_Card > Ø then do
       reo613 = (Dsnu613_Nearindref + Dsnu613_Farindref) * 1ØØ
       reo613 = reo613 % Dsnu613_Card
       if reo613 > 1Ø or | Dsnu613_Percdrop > 1Ø then do
    reo613 = 'Y' ; Sireo  = 'Y' ; end ; else reo613 = 'N' ; end ; return
   /*--  Routine for msg. DSNU614I         --*/
 Msg_DSNU614I :
    £d = £d + 1 ; npCard = word(outruns.£d,1)
    if npCard = 'CARD' then npCard = word(outruns.£d,3)
    £d = £d + 1 ; npCardf = word(outruns.£d,1)
    if npCardf = 'CARDF' then npCardf = word(outruns.£d,3)
   /*--  Test for Large TableSpace           --*/
    if npCard = -1 then npcard = FORMAT(npCardf,,Ø,Ø); return
   /*--  Routine for msg. DSNU617I           --*/
 Msg_DSNU617I :
    £d = £d + 1 ; npClustered = word(outruns.£d,1)
    if npClustered = 'CLUSTERED' then npClustered = word(outruns.£d,3)
    £d = £d + 1 ; npClusterratio = word(outruns.£d,1)
    if npClusterratio = 'CLUSTERRATIO' then npClusterratio =
word(outruns.£d,3)
```

```
    reo617 = 'N' ; if npClusterratio < 9Ø then do
       DO £e = 1 to sysrecØØ.Ø ; $creator  =
strip(substr(sysrecØØ.£e,1,8))
          $ixname   = strip(substr(sysrecØØ.£e,11,18))
          if $creator||'.'||$ixname = wrecØØ then do
             reo617 = 'Y' ; Sireo  = 'Y' ; return ; end
          else reo617 = 'N' ; end ; end ; return
   /*--  Routine for msg. DSNU618I            --*/
 Msg_DSNU618I :
    £d = £d + 1 ; npCard = word(outruns.£d,1)
    if npCard = 'CARD' then npCard = word(outruns.£d,3)
    £d = £d + 1 ; npCardf = word(outruns.£d,1)
    if npCardf = 'CARDF' then npCardf = word(outruns.£d,3)
   /*--  Test for Large TableSpace            --*/
    if npCard = -1 then npcard = FORMAT(npCardf,,Ø,Ø)
    £d = £d + 3 ; npFaroffpos = word(outruns.£d,1)
    if npFaroffpos = 'FAROFFPOS' then npFaroffpos = word(outruns.£d,3)
    £d = £d + 1 ; npFaroffposF = word(outruns.£d,1)
    if npFaroffposF = 'FAROFFPOSF' then npFaroffposF =
word(outruns.£d,3)
   /*--  Test for Large TableSpace            --*/
    if npFaroffpos =  -1 then npFaroffpos = FORMAT(npFaroffposF,,Ø,Ø)
    £d = £d + 1 ; npLeafdist = word(outruns.£d,1)
    if npLeafdist = 'LEAFDIST' then npLeafdist = word(outruns.£d,3)
    reo6182 = 'N' ; f npLeafdist > 5ØØ then do ; reo6182 = 'Y' ; Sireo
= 'Y' ; end
    else reo6182 = 'N' ; reo6181 = 'N'
    if npCard > Ø then do ; reo6181 = (npFaroffpos * 1ØØ) % npCard
       if reo6181 > 1Ø then do; DO £e = 1 to sysrecØØ.Ø
             $creator  = strip(substr(sysrecØØ.£e,1,8))
             $ixname   = strip(substr(sysrecØØ.£e,11,18))
             if $creator||'.'||$ixname = wrecØØ then do
                reo6181 = 'Y' ; Sireo  = 'Y' ; return ; end
             else reo6181 = 'N' ; end ; end
       else reo6181 = 'N' ; end ; return
   /*--  Routine for write output record      --*/
 WriteRec :
    "EXECIO "nrec" DISKW "jobw" (STEM sk. FINIS"
 ClearRec :
    DO f = 1 to sk.Ø ; sk.f = blk ; end ; return
   /*--  Routine for clean record             --*/
 ClRecOut :
    ctreom = Ø ; £hctr  = Ø ; £f613  = Ø ; £f617  = Ø ; £f618  = Ø
    DO £z = 1 to 3ØØ
       a1a.£z = blk ; a1b.£z = blk ; a1c.£z = blk ; a2a.£z = blk ;
a2b.£z = blk
 a2c.£z = blk ; a2.£z  = blk ; a3.£z  = blk ; a4.£z  = blk ; a5.£z = blk
       a6.£z = blk  ; sk.£z = blk  ; sj.£z = blk ; end
    return
   /*--  Free work data-set                   --*/
```

```
 Free :
    xx=outtrap(trpdummy.)
       "free fi(outruns)" ; "free fi(fireoop)" ; "free fi(sysrec00)"
       address tso "delete '"outdsprt"'" ; "free fi(sysprint)"
       address tso "delete '"outdsrec"'" ; address tso "delete
'"outdstsin"'"
       "free fi(systsinp)" ; "free fi(sysin)" ;address tso "delete
'"outdsin"'"
       "free fi(syspunch)" ; address tso "delete '"inpds01"'"
    xx=outtrap(off)
    return
```

$DB2PAR0 REXX EXEC

The code was published in *DB2 Update*, issue 106, August 2001. Article title: 'Imagecopy generator procedure'.

$DB2ALL0 REXX EXEC

The code was published in *DB2 Update*, issue 106, August 2001. Article title: 'Imagecopy generator procedure'.

$DB2ACC0 REXX EXEC

The code was published in *DB2 Update*, issue 106, August 2001. Article title: 'Imagecopy generator procedure'.

$DB2RBND EXEC

The code was published in *DB2 Update,* issue 73, November 1998. Article title: 'Plan and package management'.

$MDB2007 MACRO

```
/* REXX */
isredit macro
isredit exclude '''MEMBER NAME''' 1 all
isredit exclude '''DSNU000I'''    2 all
isredit exclude '''DSNU010I'''    2 all
isredit find    '''DSNU000I'''    2 First
isredit find    '''DSNU010I'''    2 Last
isredit delete x all
isredit change P'''=''' ''' '''   1 all
```

```
isredit save
isredit end
```

$MDB2019 MACRO

```
/* REXX */
isredit macro
isredit change X'''00''' x'''40''' all
isredit save
isredit end
```

SAMPLE JOB TO SUBMIT PROCEDURE

```
//&SYSUID.0 JOB (xxxxxxx),CLASS=I,NOTIFY=&SYSUID,
//            MSGCLASS=L,REGION=4M,MSGLEVEL=(1,1),TIME=1440
/*JOBPARM BYTES=999999,LINES=999999
//* --     DB2 Subsystem -------:  dsnx                        -- *
//* --     DataBase ------------:  DBNAME00                    -- *
//* --     TSname --------------:  *                           -- *
//* --     Submit JOB  ---------:  NO                          -- *
//* --     Statstime filt ------:  *                           -- *
//* --     Acronimo Progetto ---:  axxx                        -- *
//* --     Parallel Runstats ---:  9                           -- *
//* --     Creator Rebind ------:  CREAT00                     -- *
//DB2PROC  JCLLIB ORDER=(user.library)
//STEP00   EXEC   DB2REXX1
//REXX00.SYSTSIN  DD  *
    ISPSTART CMD($DB2RUN1 dsnZ,dbname00,*,NO,*,axxx,9,creat00)
```

DB2REXX1 PROC

The code was published in *DB2 Update*, issue 106, August 2001. Article title: ‘Imagecopy generator procedure’.

AXXXREOD PROC

```
//* --   Online Reorg with dasd IMAGECOPY                      -- *
//STEP005  EXEC PGM=EDITSYS,
//         PARM=('$X$X,&TAB')
//SYSUDUMP DD SYSOUT=*
//INSYS    DD DISP=SHR,DSN=USER.LIBRARY(AXXXREOD)
//OUTSYS   DD DISP=(NEW,PASS,DELETE),DSN=&&TEMP1,
//         SPACE=(CYL,(1,1),RLSE),UNIT=SYSDA,
//         DCB=(LRECL=80,RECFM=FB)
```

```
//STEPØØ5A EXEC PGM=IEFBR14
//DELETE1  DD DSN=AXXX.&TAB..SYSERR,
//         DISP=(MOD,DELETE,DELETE),SPACE=(TRK,(1,1))
//STEPØØ5B EXEC PGM=ARUUMAIN,
//         PARM='DSNX,REO&TAB,NEW/RESTART,,MSGLEVEL(1)'
//STEPLIB  DD DISP=SHR,DSN=BMC.LOADLIB
//         DD DISP=SHR,DSN=SYS1.DSN51Ø.SDSNLOAD
//SYSPRINT DD SYSOUT=*
//SYSERR   DD DSN=AXXX.&TAB..SYSERR,DISP=(,CATLG),
//            SPACE=(CYL,(2Ø,3Ø),RLSE),UNIT=(SYSDA,2)
//UTPRINT  DD SYSOUT=*
//SYSUDUMP DD SYSOUT=*
//KSDEBUG$ DD SYSOUT=*
//SYSIN    DD DISP=(OLD,PASS),DSN=&&TEMP1
//STEPØ1Ø  EXEC PGM=IEFBR14
//DELETE1  DD DSN=AXXX.&TAB..SYSERR,
//         DISP=(MOD,DELETE,DELETE),SPACE=(TRK,(1,1))
//STEPØ15  EXEC PGM=EDITSYS,
//         PARM=('$X$X,&TAB')
//SYSUDUMP DD SYSOUT=*
//INSYS    DD DISP=SHR,DSN=USER.LIBRARY(AXXXRUNS)
//OUTSYS   DD DISP=(NEW,PASS,DELETE),DSN=&&TEMP2,
//         SPACE=(CYL,(1,1),RLSE),UNIT=SYSDA,
//         DCB=(LRECL=8Ø,RECFM=FB)
//* --   RUNSTATS TABLESPACE                                    -- *
//STEPØ15A EXEC PGM=DSNUTILB,PARM='DSNX,RUN&TAB'
//STEPLIB  DD DISP=SHR,DSN=SYS1.DSN51Ø.SDSNLOAD
//SYSPRINT DD SYSOUT=*
//UTPRINT  DD SYSOUT=*
//SYSIN    DD DISP=(OLD,PASS),DSN=&&TEMP2
```

AXXXREOT PROC

```
//* --   Online Reorg with tape IMAGECOPY                       -- *
//STEPØØ5  EXEC PGM=EDITSYS,
//         PARM=('$X$X,&TAB')
//SYSUDUMP DD SYSOUT=*
//INSYS    DD DISP=SHR,DSN=USER.LIBRARY(AXXXREOT)
//OUTSYS   DD DISP=(NEW,PASS,DELETE),DSN=&&TEMP1,
//         SPACE=(CYL,(1,1),RLSE),UNIT=SYSDA,
//         DCB=(LRECL=8Ø,RECFM=FB)
//STEPØØ5A EXEC PGM=IEFBR14
//DELETE1  DD DSN=AXXX.&TAB..SYSERR,
//         DISP=(MOD,DELETE,DELETE),SPACE=(TRK,(1,1))
//STEPØØ5B EXEC PGM=ARUUMAIN,
//         PARM='DSNX,REO&TAB,NEW/RESTART,,MSGLEVEL(1)'
//STEPLIB  DD DISP=SHR,DSN=BMC.LOADLIB
//         DD DISP=SHR,DSN=SYS1.DSN51Ø.SDSNLOAD
//SYSPRINT DD SYSOUT=*
```

```
//SYSERR   DD DSN=AXXX.&TAB..SYSERR,DISP=(,CATLG),
//            SPACE=(CYL,(2Ø,3Ø),RLSE),UNIT=(SYSDA,2)
//UTPRINT  DD SYSOUT=*
//SYSUDUMP DD SYSOUT=*
//KSDEBUG$ DD SYSOUT=*
//SYSIN    DD DISP=(OLD,PASS),DSN=&&TEMP1
//STEPØ1Ø  EXEC PGM=IEFBR14
//DELETE1  DD DSN=AXXX.&TAB..SYSERR,
//         DISP=(MOD,DELETE,DELETE),SPACE=(TRK,(1,1))
//STEPØ15  EXEC PGM=EDITSYS,
//         PARM=('$X$X,&TAB')
//SYSUDUMP DD SYSOUT=*
//INSYS    DD DISP=SHR,DSN=USER.LIBRARY(AXXXRUNS)
//OUTSYS   DD DISP=(NEW,PASS,DELETE),DSN=&&TEMP2,
//         SPACE=(CYL,(1,1),RLSE),UNIT=SYSDA,
//         DCB=(LRECL=8Ø,RECFM=FB)
//* --   RUNSTATS TABLESPACE                                    -- *
//STEPØ15A EXEC  PGM=DSNUTILB,PARM='DSNX,RUN&TAB'
//STEPLIB  DD DISP=SHR,DSN=SYS1.DSN51Ø.SDSNLOAD
//SYSPRINT DD SYSOUT=*
//UTPRINT  DD SYSOUT=*
//SYSIN    DD DISP=(OLD,PASS),DSN=&&TEMP2
```

AXXXREDP PROC

```
//*       Online Reorg of a Partition with DASD IMAGECOPY       -- *
//STEPØØ5  EXEC PGM=EDITSYS,
//         PARM=('$X$X,&TAB|$Y$Y,&PART')
//SYSUDUMP DD SYSOUT=*
//INSYS    DD DISP=SHR,DSN=USER.LIBRARY(AXXXREDP)
//OUTSYS   DD DISP=(NEW,PASS,DELETE),DSN=&&TEMP1,
//         SPACE=(CYL,(1,1),RLSE),UNIT=SYSDA,
//         DCB=(LRECL=8Ø,RECFM=FB)
//STEPØØ5A EXEC PGM=IEFBR14
//DELETE1  DD DSN=AXXX.&TAB..SYSERR.P&PART,
//         DISP=(MOD,DELETE,DELETE),SPACE=(TRK,(1,1))
//STEPØØ5B EXEC PGM=ARUUMAIN,
//         PARM='DSNX,REO&TAB&PART,NEW/RESTART,,MSGLEVEL(1)'
//STEPLIB  DD DISP=SHR,DSN=BMC.LOADLIB
//         DD DISP=SHR,DSN=SYS1.DSN51Ø.SDSNLOAD
//SYSPRINT DD SYSOUT=*
//SYSERR   DD DSN=AXXX.&TAB..SYSERR.P&PART,DISP=(,CATLG),
//            SPACE=(CYL,(2Ø,3Ø),RLSE),UNIT=(SYSDA,2)
//UTPRINT  DD SYSOUT=*
//SYSUDUMP DD SYSOUT=*
//KSDEBUG$ DD SYSOUT=*
//SYSIN    DD DISP=(OLD,PASS),DSN=&&TEMP1
//STEPØ1Ø  EXEC PGM=IEFBR14
//DELETE1  DD DSN=AXXX.&TAB..SYSERR.P&PART,
//         DISP=(MOD,DELETE,DELETE),SPACE=(TRK,(1,1))
```

AXXXREIX PROC

```
//* --   Online Reorg of an INDEX                                  -- *
//STEP005  EXEC PGM=EDITSYS,
//         PARM=('$X$X,&TAB')
//SYSUDUMP DD SYSOUT=*
//INSYS    DD DISP=SHR,DSN=USER.LIBRARY(AXXXREIX)
//OUTSYS   DD DISP=(NEW,PASS,DELETE),DSN=&&TEMP1,
//         SPACE=(CYL,(1,1),RLSE),UNIT=SYSDA,
//         DCB=(LRECL=80,RECFM=FB)
//STEP005B EXEC PGM=ARUUMAIN,
//         PARM='DSNX,REO&TAB,NEW/RESTART,,MSGLEVEL(1)'
//STEPLIB  DD DISP=SHR,DSN=BMC.LOADLIB
//         DD DISP=SHR,DSN=SYS1.DSN510.SDSNLOAD
//SYSPRINT DD SYSOUT=*
//UTPRINT  DD SYSOUT=*
//SYSUDUMP DD SYSOUT=*
//KSDEBUG$ DD SYSOUT=*
//SYSIN    DD DISP=(OLD,PASS),DSN=&&TEMP1
```

AXXXREOD PARM

```
REORG TABLESPACE DBNAME00.$X$X
SHRLEVEL CHANGE
COPY YES
INLINE YES
DDTYPE UNLOAD ACTIVE YES SIZEPCT(50,200)
DSNPAT  'AXXX.&TSIX..D&DATE..T&TIME.&DDNAME'
*
```

AXXXREOT PARM

```
REORG TABLESPACE DBNAME00.$X$X
SHRLEVEL CHANGE
COPY YES
INLINE YES
DDTYPE UNLOAD ACTIVE YES SIZEPCT(50,200)
DSNPAT  'AXXX.&TSIX..D&DATE..T&TIME.&DDNAME'
COPYDDN (FC)
DDTYPE LOCPFCPY ACTIVE YES
DSNPAT('AXXX..&TSIX..D&DATE..T&TIME4.&DDNAME')
UNIT CART
*
```

AXXXREDP PARM

```
REORG TABLESPACE DBNAME00.$X$X PART $Y$Y
SHRLEVEL CHANGE
```

```
COPY YES
INLINE YES
COPYDDN (BMCPY)
ICDDN (BMICY,BMICZ)
RECOVERYICDDN (BMIRY,BMIRZ)
UNLDDN SYSRC
DDTYPE UNLOAD ACTIVE YES SIZEPCT(50,200)
DSNPAT 'AXXX.&TSIX..D&DATE..T&TIME.&DDNAME'
*
```

AXXXREIX PARM

```
REORG INDEX AXXX.$X$X
SHRLEVEL CHANGE
*
```

AXXXRUNS PARM

```
RUNSTATS TABLESPACE DBNAME00.$X$X
INDEX(ALL) SHRLEVEL CHANGE REPORT YES
```

Massimo Balzano
DB2 Systems Programmer (Italy)

REXX/ISPF program to help programmers to resolve sqlcode -805

In our shop many sqlcode -805s are produced by the execution of DB2 plans with package lists that do not contain the location of the package to execute. Another cause of sqlcode -805s is the attempted invocation of a module from an incorrect load library or a mismatch between the contoken of the module and the package.

To overcome these problems, I developed a REXX/ISPF program, DB2PLN, which I tested on z/OS V1.4 and DB2 V6.1.

At the start of its execution, program DB2PLN searches the

DB2 subsystems defined on z/OS. The results are shown in an initial panel, so the user can choose a DB2 subsystem.

Then a panel is displayed for the user to enter a plan name to view or modify. Next a window is presented with the collections included in the plan. Here the user can choose to add (option A) a collection, delete (option D) a collection from the package list of the plan, or display all the packages in one collection (option P).

If the user chooses to add or delete a collection, the program verifies whether the plan is in use by any process or thread. If it is in use, another window is displayed with all the threads (ie CICS, job, TSO, etc) using the plan, and the rebind is not performed (a message, 'Plan in use, try later' is displayed). Otherwise, the plan is rebound and the collections refreshed (more or fewer collections according to the action executed).

If the user chooses option P, the program displays a window with all the packages belonging to the collection, bindtime, pre-compiled time, contoken, and identifies whether the package is in use or not.

At this point the user has two options available: view all the collections containing the chosen package (option C), or review which load libraries contain a COBOL module with equal contoken to the selected package (option V).

The result of option C is shown in a window (with at least one row, a collection belonging to the plan entered at the beginning) and allows the user to identify duplicates of a package in other collections (may be a cause of sqlcode -805).

Where option V is selected, a panel is displayed where the user can enter up to eight dataset names of load libraries to search for a COBOL module with the same name and contoken as the package chosen.

In the event that the package and module are synchronized (RESULT = CORRECT), the sqlcode -805 will not be present.

It's assumed that routines are dynamically linked (ie there is

one COBOL module for each routine to compare the contoken of the module against the contoken of the package).

The results of this search are displayed on the same panel in the column RESULT. The possible values of the RESULT column are:

- Correct – the module exists in the library and the contoken is equal (program will return sqlcode 000 to the connection package-module if it is executed with the package chosen). The answer is displayed in a row coloured turquoise.
- Incorrect – the module exists in the library but the contoken is different. The answer is displayed in a row coloured pink.
- Dset error – the dataset is not found in the catalog. The answer is displayed in a row coloured red.
- Open error – the library cannot be opened. The answer is displayed in a row coloured red.
- No loadlib – the library is not a load library. The answer is displayed in a row coloured red.
- Not found – the module does not exist in the library. The answer is displayed in a row coloured red.

In the last five cases the package and the module are not synchronized (program will return sqlcode -805 if it is executed with the package chosen).

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

The REXX program also maintains a log in which it writes all user actions. It stores information like the options chosen, users who execute DB2PLN, DB2 subsystem, name of the collection added or deleted, package view, day and hour of the DB2PLN execution, etc.

For the set of the libraries (panels, windows, and messages) the program assumes (and you must replace these values) that:

- Subsystem DB2P is a production environment (dsnames begin with PREFIX1).
- Subsystem DB2T is a test environment (dsnames begin with PREFIX2).
- Subsystem DB2D is a development environment (dsnames begin with PREFIX3).

Prefix0.public.paneli is an initial and common panel library.

I installed the program DB2PLN as an option of the DB2 panel DSNEPRI (beside SPUFI and QMF).

DB2PLN

```
/* REXX DB2PLN                CARLOS-OSORIO@EXCITE.COM               */
TRACE OFF
NUMERIC   DIGITS 12;
CVT     = C2X(STORAGE(1Ø,4))
CVTJESCT= D2X((X2D(CVT))+296)
JESCT   = C2X(STORAGE(CVTJESCT,4))
JESSSCT = D2X((X2D(JESCT))+24)
SSCVT   = C2X(STORAGE(JESSSCT,4))
J = Ø
DO I = 1  WHILE (SSCVT <> ØØØØØØØØ)
          SSCTSNAM=D2X((X2D(SSCVT))+8)
          ERLY  = D2X((X2D(SSCVT))+2Ø)
          ERLYAD= C2X(STORAGE(ERLY,4))
          ERLYSCOM= D2X((X2D(ERLYAD))+56)
          IF SUBSTR(STORAGE(ERLYSCOM,64),29,8) = 'DSN3EPX ' THEN
             DO
             J = J + 1
             SSIDDB2.J = STORAGE(SSCTSNAM,4)
             END
          SSCTSCTA = D2X((X2D(SSCVT))+4)
          SSCVT    = C2X(STORAGE(SSCTSCTA,4))
END
"ISPEXEC LIBDEF ISPPLIB DATASET ID('PREFIXØ.PUBLIC.PANELI')"
"ISPEXEC TBCREATE TSSIDDB2",
"NAMES(O SDB2)",
"NOWRITE REPLACE"
O = ' '
```

```
DO J = 1 TO J
SDB2 = SSIDDB2.J
"ISPEXEC TBADD   TSSIDDB2"
END
"ISPEXEC TBTOP   TSSIDDB2"
"ISPEXEC TBDISPL TSSIDDB2 PANEL(DB2PLNSS)"
RCI = RC
"ISPEXEC LIBDEF ISPPLIB"
IF RCI = 8 THEN EXIT;
"ISPEXEC TBEND  TSSIDDB2"
SSID = SDB2
SELECT
  WHEN SSID = 'DB2P' THEN
       DO
      "ISPEXEC LIBDEF ISPPLIB DATASET ID('PREFIX1.PUBLIC.PANELLIB')"
      "ISPEXEC LIBDEF ISPMLIB DATASET ID('PREFIX1.PUBLIC.MSGLIB')"
       LOGPREFIX = 'PREFIX1'
       END
  WHEN SSID = 'DB2T' THEN
       DO
      "ISPEXEC LIBDEF ISPPLIB DATASET ID('PREFIX2.PUBLIC.PANELLIB')"
      "ISPEXEC LIBDEF ISPMLIB DATASET ID('PREFIX2.PUBLIC.MSGLIB')"
       LOGPREFIX = 'PREFIX2'
       END
  WHEN SSID = 'DB2D' THEN
       DO
      "ISPEXEC LIBDEF ISPPLIB DATASET ID('PREFIX3.PUBLIC.PANELLIB')"
      "ISPEXEC LIBDEF ISPMLIB DATASET ID('PREFIX3.PUBLIC.MSGLIB')"
       LOGPREFIX = 'PREFIX3'
       END
  OTHERWISE
       DO
      "ISPEXEC LIBDEF ISPPLIB DATASET ID('PREFIX1.PUBLIC.PANELLIB')"
      "ISPEXEC LIBDEF ISPMLIB DATASET ID('PREFIX1.PUBLIC.MSGLIB')"
       LOGPREFIX = 'PREFIX1'
       END
END
DO FOREVER
   DISPKLIST_EXIT = 'N'
   ADDRESS ISPEXEC "DISPLAY PANEL(DB2PLNIN)"
   IF   RC = 8 THEN CALL FINAL
               ELSE DO
                    ADDRESS ISPEXEC "TBCREATE TPLN",
                   "NAMES(O COLLID SEQNO)",
                   "NOWRITE REPLACE"
                    CALL GETPKLIST
                    CALL LIBDEFWINDJ
                      DO   WHILE  DISPKLIST_EXIT = 'N'
                      CALL DISPKLIST
                      END
```

```
                    CALL LIBDEFPANEL
                    ADDRESS ISPEXEC 'TBEND TPLN'
                    END
END
/*---------------- R O U T I N E S ---- CARLOS-OSORIO@EXCITE.COM--*/
GETPKLIST:
 CALL ALLOCSYS;
 IF ALLOCSYSRC = 1 THEN RETURN;
 O = ' '
 DO I = 1 TO 4
    SYSIN.I = ''
 END
 SYSIN.1  = "SELECT COLLID,SEQNO "
 SYSIN.2  = "FROM SYSIBM.SYSPACKLIST"
 SYSIN.3  = "WHERE PLANNAME = '"PLAN"'"
 SYSIN.4  = "ORDER BY SEQNO;"
"EXECIO  *   DISKW SYSIN (STEM SYSIN."
"EXECIO  Ø   DISKW SYSIN (FINIS"
 ADDRESS TSO "ALLOC FILE(SYSIN)    DATASET('"DSNIN"')    OLD REUSE"
 ADDRESS TSO "ALLOC FILE(SYSRECØØ) DATASET('"DSNREC"')   OLD REUSE"
 ADDRESS TSO "ALLOC FILE(SYSPRINT) DATASET('"DSNPRINT"') OLD REUSE"
 ADDRESS TSO "ALLOC FILE(SYSPUNCH) DUMMY"
 PUSH "END"
 PUSH "RUN PROGRAM(DSNTIAUL) PLAN(DSNTIAUL) PARMS('SQL')"
 ADDRESS TSO "DSN SYSTEM("SSID")"
 ADDRESS TSO "EXECIO * DISKR SYSRECØØ (STEM REC. FINIS"
 ADDRESS TSO "FREE FILE(SYSIN)"
 ADDRESS TSO "FREE FILE(SYSRECØØ)"
 ADDRESS TSO "FREE FILE(SYSPRINT)"
 ADDRESS TSO "FREE FILE(SYSPUNCH)"
 CALL DELTMP;
 SELECT
    WHEN REC.Ø = Ø THEN DO
                         ADDRESS ISPEXEC "SETMSG MSG(DBC7Ø1)"
                         ADDRESS ISPEXEC "DISPLAY PANEL(DB2PLNIN)"
                        CALL FINAL
                        RETURN
                        END
    OTHERWISE           DO I = 1 TO REC.Ø
                         COLLID  = STRIP(SUBSTR(REC.I,1,18))
                         SEQNO   = C2D(SUBSTR(REC.I,19,2))
                        ADDRESS ISPEXEC 'TBADD TPLN'
                        COLLIDR = COLLID
                        CALL GRABALOG
                        END
 END
RETURN
/*-------------------------------------------------------------------*/
DISPKLIST:
 COLLIDTG = '';
```

```
 ADDRESS ISPEXEC 'TBTOP   TPLN'
 ADDRESS ISPEXEC 'ADDPOP  ROW(3) COLUMN(3)'
 ADDRESS ISPEXEC 'TBDISPL TPLN PANEL(DB2TPLNS)'
 ADDDELCOLLID_EXIT = 'N';
 IF RC = 8 THEN          DISPKLIST_EXIT = 'S'
           ELSE DO WHILE ADDDELCOLLID_EXIT = 'N'
                SELECT
                   WHEN O = 'A' THEN CALL ADDCOLLID;
                   WHEN O = 'I' THEN CALL ADDCOLLID;
                   WHEN O = 'D' THEN DO
                                     COLLIDTG = COLLID
                                     CALL DELCOLLID
                                     END
                   WHEN O = 'P' THEN CALL PACKAGES;
                   OTHERWISE    NOP
                END
                END
 ADDRESS ISPEXEC 'REMPOP'
RETURN
/*---------------------------------------CARLOS-OSORIO@EXCITE.COM--*/
ADDCOLLID:
 ADDRESS ISPEXEC 'TBCREATE TTHD',
                 'NAMES(ID NAME AUTHID)',
                 'NOWRITE REPLACE'
 CALL CHK_PLAN_EN_USO;
 IF   PLAN_EN_USO = 'S' THEN
      DO
      ADDRESS ISPEXEC 'TBSORT  TTHD FIELDS(ID,C,A,NAME,C,A)'
      ADDRESS ISPEXEC 'TBTOP   TTHD'
      ADDRESS ISPEXEC 'ADDPOP  ROW(-4) COLUMN(7)'
      ADDRESS ISPEXEC 'TBDISPL TTHD PANEL(DB2TPLUS)'
      ADDRESS ISPEXEC 'REMPOP'
      ADDDELCOLLID_EXIT = 'S'
      RETURN
      END
 ADDRESS ISPEXEC 'TBEND TTHD'
 CALL LIBDEFWINDO;
 ADDRESS ISPEXEC "ADDPOP ROW(2) COLUMN(7)"
 ADDRESS ISPEXEC "DISPLAY PANEL(DB2PLNAD)"
 IF   RC = 8 THEN ADDDELCOLLID_EXIT = 'S'
 ELSE DO
      CALL CHK_COLLID_EN_PLAN;
      IF COLLID_EN_PLAN = 'S' THEN DO
                                   ADDRESS ISPEXEC 'SETMSG MSG(DBC7Ø2)'
                                   ADDRESS ISPEXEC "REMPOP"
                                   ADDDELCOLLID_EXIT = 'S'
                                   CALL LIBDEFWINDJ;
                                   RETURN
                                   END
      COLLIDR = COLLIDTG
```

```
      CALL REBIND_PLAN
      END
 ADDDELCOLLID_EXIT = 'S'
 ADDRESS ISPEXEC "REMPOP"
 CALL LIBDEFWINDJ;
RETURN
/*-------------------------------------------------------------------*/
DELCOLLID:
 ADDRESS ISPEXEC 'TBCREATE TTHD',
                 'NAMES(ID NAME AUTHID)',
                 'NOWRITE REPLACE'
 CALL CHK_PLAN_EN_USO;
 IF   PLAN_EN_USO = 'S' THEN
      DO
      ADDRESS ISPEXEC 'TBSORT  TTHD FIELDS(ID,C,A,NAME,C,A)'
      ADDRESS ISPEXEC 'TBTOP   TTHD'
      ADDRESS ISPEXEC 'ADDPOP  ROW(-4) COLUMN(7)'
      ADDRESS ISPEXEC 'TBDISPL TTHD PANEL(DB2TPLUS)'
      ADDRESS ISPEXEC 'REMPOP'
      ADDDELCOLLID_EXIT = 'S'
      RETURN
      END
 ADDRESS ISPEXEC 'TBEND TTHD'
 C = ' '
 CALL LIBDEFWINDO;
 ADDRESS ISPEXEC "ADDPOP ROW(2) COLUMN(7)"
 ADDRESS ISPEXEC "DISPLAY PANEL(DB2PLNDE)"
 IF   RC = 8 THEN ADDDELCOLLID_EXIT = 'S'
 ELSE DO
      CALL CHK_COLLID_EN_PLAN
      IF COLLID_EN_PLAN = 'N' THEN DO
                                   ADDRESS ISPEXEC 'SETMSG MSG(DBC7Ø9)'
                                   ADDRESS ISPEXEC "REMPOP"
                                   ADDDELCOLLID_EXIT = 'S'
                                   CALL LIBDEFWINDJ
                                   RETURN
                                   END
      CALL CHK_ANY_COLLID
      IF ANY_COLLID = 'N'     THEN DO
                                   ADDRESS ISPEXEC "SETMSG MSG(DBC71Ø)"
                                   ADDRESS ISPEXEC "REMPOP"
                                   ADDDELCOLLID_EXIT = 'S'
                                   CALL LIBDEFWINDJ
                                   RETURN
                                   END
      IF C = 'Y'              THEN DO
                                   COLLIDR = COLLIDTG
                                   CALL REBIND_PLAN
                                   END
      END
```

```
 ADDDELCOLLID_EXIT = 'S'
 ADDRESS ISPEXEC "REMPOP"
 CALL LIBDEFWINDJ
RETURN
/*--------------------------------------------------------------------*/
CHK_PLAN_EN_USO:
W = OUTTRAP('THREAD.')
    QUEUE "-DIS THD(*)"
    QUEUE "END"
   "DSN SYSTEM("SSID")"
W = OUTTRAP('OFF')
    DO I = 1   UNTIL (SUBSTR(THREAD.I,1,8) = 'NAME    ' | ,
                      SUBSTR(THREAD.I,1,8) = 'DSNV41ØI' | ,
                      SUBSTR(THREAD.I,1,8) = 'DSNV411I' )
    END
    PROCES = Ø
    DO I = I+1 WHILE SUBSTR(THREAD.I,1,7) <> 'DISPLAY'
    IF SUBSTR(THREAD.I,1Ø,1) = 'T' THEN
       DO
       IF SUBSTR(THREAD.I,43,8) = PLAN THEN
          DO
          PROCES = PROCES + 1
          NAME   = SUBSTR(THREAD.I,1,8)
          ID     = SUBSTR(THREAD.I,21,1Ø)
          AUTHID = SUBSTR(THREAD.I,34,8)
         "ISPEXEC TBADD TTHD"
          END
       END
    END
    IF PROCES = Ø THEN PLAN_EN_USO = 'N'
                  ELSE PLAN_EN_USO = 'S';
RETURN
/*--------------------------------------------------------------------*/
CHK_COLLID_EN_PLAN:
 COLLID_EN_PLAN = 'N'
 DO I = 1 TO REC.Ø WHILE COLLID_EN_PLAN = 'N'
 IF STRIP(SUBSTR(REC.I,1,18)) = STRIP(COLLIDTG)
    THEN COLLID_EN_PLAN = 'S'
 END
RETURN
/*---------------------------------------CARLOS-OSORIO@EXCITE.COM--*/
CHK_ANY_COLLID:
 CALL ALLOCSYS;
 IF ALLOCSYSRC = 1 THEN RETURN;
 DO I = 1 TO 4
    SYSIN.I = ''
 END
 SYSIN.1  = "SELECT COLLID,SEQNO "
 SYSIN.2  = "FROM SYSIBM.SYSPACKLIST"
 SYSIN.3  = "WHERE COLLID   = '"COLLIDTG"';"
```

```
"EXECIO  *   DISKW SYSIN (STEM SYSIN."
"EXECIO  Ø   DISKW SYSIN (FINIS"
 ADDRESS TSO "ALLOC FILE(SYSIN)    DATASET('"DSNIN"')    OLD REUSE"
 ADDRESS TSO "ALLOC FILE(SYSRECØØ) DATASET('"DSNREC"')   OLD REUSE"
 ADDRESS TSO "ALLOC FILE(SYSPRINT) DATASET('"DSNPRINT"') OLD REUSE"
 ADDRESS TSO "ALLOC FILE(SYSPUNCH) DUMMY"
 PUSH "END"
 PUSH "RUN PROGRAM(DSNTIAUL) PLAN(DSNTIAUL) PARMS('SQL')"
 ADDRESS TSO "DSN SYSTEM("SSID")"
 ADDRESS TSO "EXECIO * DISKR SYSRECØØ (STEM RECPK. FINIS"
 ADDRESS TSO "FREE FILE(SYSIN)"
 ADDRESS TSO "FREE FILE(SYSRECØØ)"
 ADDRESS TSO "FREE FILE(SYSPRINT)"
 ADDRESS TSO "FREE FILE(SYSPUNCH)"
 CALL DELTMP;
 IF RECPK.Ø = Ø THEN ANY_COLLID = 'N'
                ELSE ANY_COLLID = 'S'
RETURN
/*-------------------------------------------------------------------*/
REBIND_PLAN:
 IF (O = 'A') | (O = 'I') | (O = 'D') THEN
    DO
    PKLIST = ''
      DO J = 1 TO REC.Ø
         IF (O = 'D') & (COLLIDR = STRIP(SUBSTR(REC.J,1,18)))
            THEN NOP
            ELSE PKLIST = PKLIST || STRIP(SUBSTR(REC.J,1,18)) || '.*,'
      END
    IF (O = 'D')
       THEN PKLIST = SUBSTR(PKLIST,1,(LENGTH(PKLIST)-1))
       ELSE PKLIST = PKLIST || COLLIDR ||'.*'
   "NEWSTACK"
    W = OUTTRAP('REB.')
    QUEUE "REBIND PLAN("PLAN") PKLIST("PKLIST")"
    QUEUE "END"
   "DSN SYSTEM("SSID") RETRY(Ø)"
    RCB = RC
    W = OUTTRAP('OFF')
   "DELSTACK"
    IF RCB = Ø THEN DO
                    SELECT
                      WHEN O = 'A' THEN
                           ADDRESS ISPEXEC "SETMSG MSG(DBC7Ø4)";
                      WHEN O = 'I' THEN
                           ADDRESS ISPEXEC "SETMSG MSG(DBC7Ø4)";
                      WHEN O = 'D' THEN
                           ADDRESS ISPEXEC "SETMSG MSG(DBC7Ø3)";
                      OTHERWISE NOP
                    END
                    CALL GRABALOG
```

```
                    ADDRESS ISPEXEC 'TBEND  TPLN'
                    ADDRESS ISPEXEC "TBCREATE TPLN",
                   "NAMES(O COLLID SEQNO)",
                   "NOWRITE REPLACE"
                    CALL GETPKLIST
                    END
                ELSE DO
                       SELECT
                         WHEN O = 'A' THEN
                              ADDRESS ISPEXEC "SETMSG MSG(DBC7Ø5)";
                         WHEN O = 'I' THEN
                              ADDRESS ISPEXEC "SETMSG MSG(DBC7Ø5)";
                         WHEN O = 'D' THEN
                              ADDRESS ISPEXEC "SETMSG MSG(DBC7Ø6)";
                         OTHERWISE NOP
                       END
                     END
    END
RETURN
/*-------------------------------------------------------------------*/
PACKAGES:
 ADDRESS ISPEXEC "TBCREATE TPKG",
"NAMES(P PACKAGE BINDTIME PCTIME VAL OPE CONTOKEN PDSNAME)",
"NOWRITE REPLACE"
 CALL GETPACKAGES
 ADDRESS ISPEXEC 'TBSORT   TPKG',
"FIELDS(PACKAGE,C,A,BINDTIME,C,D)"
 ADDRESS ISPEXEC 'TBTOP    TPKG'
 DISPACKAGES_EXIT = 'N'
   DO WHILE  DISPACKAGES_EXIT = 'N'
      CALL DISPACKAGES;
   END
 ADDRESS ISPEXEC 'TBEND TPKG'
 ADDDELCOLLID_EXIT = 'S'
RETURN
/*-------------------------------------------------------------------*/
GETPACKAGES:
DO K = 1 TO 4
   SYSIN.K = ''
END
 CALL ALLOCSYS;
 IF ALLOCSYSRC = 1 THEN RETURN;
 SYSIN.1  = "SELECT NAME,BINDTIME,PCTIMESTAMP,"
 SYSIN.2  = "       VALID,OPERATIVE,HEX(CONTOKEN),PDSNAME"
 SYSIN.3  = "FROM SYSIBM.SYSPACKAGE"
 SYSIN.4  = "WHERE COLLID = '"COLLID"';"
"EXECIO  *   DISKW SYSIN (STEM SYSIN."
"EXECIO  Ø   DISKW SYSIN (FINIS"
 ADDRESS TSO
 ADDRESS TSO "ALLOC FILE(SYSIN)    DATASET('"DSNIN"')    OLD REUSE"
```

```
 ADDRESS TSO "ALLOC FILE(SYSREC00) DATASET('"DSNREC"')   OLD REUSE"
 ADDRESS TSO "ALLOC FILE(SYSPRINT) DATASET('"DSNPRINT"') OLD REUSE"
 ADDRESS TSO "ALLOC FILE(SYSPUNCH) DUMMY"
 PUSH "END"
 PUSH "RUN PROGRAM(DSNTIAUL) PLAN(DSNTIAUL) PARMS('SQL')"
 ADDRESS TSO "DSN SYSTEM("SSID")"
 ADDRESS TSO "EXECIO * DISKR SYSREC00 (STEM PKG. FINIS"
 ADDRESS TSO "FREE FILE(SYSIN)"
 ADDRESS TSO "FREE FILE(SYSREC00)"
 ADDRESS TSO "FREE FILE(SYSPRINT)"
 ADDRESS TSO "FREE FILE(SYSPUNCH)"
 CALL DELTMP;
 SELECT
    WHEN PKG.0 = 0 THEN
         DO
         DISPACKAGES_EXIT = 'S'
         RETURN
         END
    OTHERWISE
         DO K = 1 TO PKG.0
         PACKAGE  = STRIP(SUBSTR(PKG.K,1,8))
         BINDTIME = SUBSTR(PKG.K,09,4) || SUBSTR(PKG.K,14,2) ||,
                    SUBSTR(PKG.K,17,2) || '.' || ,
                    SUBSTR(PKG.K,20,2) || SUBSTR(PKG.K,23,2)
         PCTIME   = SUBSTR(PKG.K,35,4) || SUBSTR(PKG.K,40,2) ||,
                    SUBSTR(PKG.K,43,2) || '.' || ,
                    SUBSTR(PKG.K,46,2) || SUBSTR(PKG.K,49,2)
         VAL      = SUBSTR(PKG.K,61,1)
         OPE      = SUBSTR(PKG.K,62,1)
         CONTOKEN = STRIP(SUBSTR(PKG.K,63,16))
         PDSNAME  = STRIP(SUBSTR(PKG.K,81,44))
         ADDRESS ISPEXEC 'TBADD TPKG'
         CALL GRABALOG
         END
 END
RETURN
/*-------------------------------------------------------------------*/
DISPACKAGES:
 DISCOLCONTOKEN_EXIT = 'N'
 LIB1 = ' ' ; RESULT1 = ' ' ; COLOR1 = 'WHITE'
 LIB2 = ' ' ; RESULT2 = ' ' ; COLOR2 = 'WHITE'
 LIB3 = ' ' ; RESULT3 = ' ' ; COLOR3 = 'WHITE'
 LIB4 = ' ' ; RESULT4 = ' ' ; COLOR4 = 'WHITE'
 LIB5 = ' ' ; RESULT5 = ' ' ; COLOR5 = 'WHITE'
 LIB6 = ' ' ; RESULT6 = ' ' ; COLOR6 = 'WHITE'
 LIB7 = ' ' ; RESULT7 = ' ' ; COLOR7 = 'WHITE'
 LIB8 = ' ' ; RESULT8 = ' ' ; COLOR8 = 'WHITE'
 ADDRESS ISPEXEC "ADDPOP ROW(1) COLUMN(2)"
 ADDRESS ISPEXEC 'TBDISPL TPKG PANEL(DB2TPKGS)'
 IF RC = 8 THEN DISPACKAGES_EXIT = 'S'
```

```
           ELSE DO WHILE DISCOLCONTOKEN_EXIT = 'N'
                   SELECT
                      WHEN P = 'C' THEN CALL SEARCH_COLLECTIONS
                      WHEN P = 'V' THEN CALL VERIFY_CONTOKEN
                      OTHERWISE    NOP
                   END
                END
 ADDRESS ISPEXEC 'REMPOP'
RETURN
/*---------------------------------------CARLOS-OSORIO@EXCITE.COM--*/
SEARCH_COLLECTIONS:
 ADDRESS ISPEXEC "TBCREATE TPKGC",
"NAMES(COLLIDC BINDTIMC PCTIMEC VALC OPEC CONTOKEC)",
"NOWRITE REPLACE"
 CALL GETCOLLIDS
 ADDRESS ISPEXEC 'TBSORT  TPKGC',
"FIELDS(COLLIDC,C,A,CONTOKEC,C,A)"
 ADDRESS ISPEXEC 'TBTOP   TPKGC'
 DISCOLLIDS_EXIT = 'N'
   DO WHILE  DISCOLLIDS_EXIT = 'N'
      CALL DISCOLLIDS;
   END
 ADDRESS ISPEXEC 'TBEND TPKGC'
 DISCOLCONTOKEN_EXIT = 'S'
RETURN
/*--------------------------------------------------------------------*/
GETCOLLIDS:
DO L = 1 TO 4
   SYSIN.K = ''
END
 CALL ALLOCSYS;
 IF ALLOCSYSRC = 1 THEN RETURN;
 SYSIN.1  = "SELECT COLLID,BINDTIME,PCTIMESTAMP,"
 SYSIN.2  = "       VALID,OPERATIVE,HEX(CONTOKEN),PDSNAME"
 SYSIN.3  = "FROM SYSIBM.SYSPACKAGE"
 SYSIN.4  = "WHERE NAME  = '"PACKAGE"';"
"EXECIO  *   DISKW SYSIN (STEM SYSIN."
"EXECIO  Ø   DISKW SYSIN (FINIS"
 ADDRESS TSO
 ADDRESS TSO "ALLOC FILE(SYSIN)    DATASET('"DSNIN"')    OLD REUSE"
 ADDRESS TSO "ALLOC FILE(SYSRECØØ) DATASET('"DSNREC"')   OLD REUSE"
 ADDRESS TSO "ALLOC FILE(SYSPRINT) DATASET('"DSNPRINT"') OLD REUSE"
 ADDRESS TSO "ALLOC FILE(SYSPUNCH) DUMMY"
 PUSH "END"
 PUSH "RUN PROGRAM(DSNTIAUL) PLAN(DSNTIAUL) PARMS('SQL')"
 ADDRESS TSO "DSN SYSTEM("SSID")"
 ADDRESS TSO "EXECIO * DISKR SYSRECØØ (STEM PKGC. FINIS"
 ADDRESS TSO "FREE FILE(SYSIN)"
 ADDRESS TSO "FREE FILE(SYSRECØØ)"
 ADDRESS TSO "FREE FILE(SYSPRINT)"
```

```
 ADDRESS TSO "FREE FILE(SYSPUNCH)"
 CALL DELTMP;
 SELECT
    WHEN PKGC.Ø = Ø THEN
         DO
         DISCOLCONTOKEN_EXIT = 'S'
         RETURN
         END
    OTHERWISE
         DO L = 1 TO PKGC.Ø
         COLLIDC  = STRIP(SUBSTR(PKGC.L,1,18))
         BINDTIMC = SUBSTR(PKGC.L,19,4) || SUBSTR(PKGC.L,24,2) ||,
                    SUBSTR(PKGC.L,27,2) || '.' || ,
                    SUBSTR(PKGC.L,3Ø,2) || SUBSTR(PKGC.L,33,2)
         PCTIMEC  = SUBSTR(PKGC.L,45,4) || SUBSTR(PKGC.L,5Ø,2) ||,
                    SUBSTR(PKGC.L,53,2) || '.' || ,
                    SUBSTR(PKGC.L,56,2) || SUBSTR(PKGC.L,59,2)
         VALC     = SUBSTR(PKGC.L,71,1)
         OPEC     = SUBSTR(PKGC.L,72,1)
         CONTOKEC = STRIP(SUBSTR(PKGC.L,73,16))
         ADDRESS ISPEXEC 'TBADD TPKGC'
         CALL GRABALOG
         END
 END
RETURN
/*-------------------------------------------------------------------*/
DISCOLLIDS:
 ADDRESS ISPEXEC 'ADDPOP  ROW(-5) COLUMN(+1)'
 ADDRESS ISPEXEC 'TBDISPL TPKGC PANEL(DB2TCOLS)'
 IF RC = 8 THEN DISCOLLIDS_EXIT = 'S'
 ADDRESS ISPEXEC 'REMPOP'
RETURN
/*-------------------------------------------------------------------*/
VERIFY_CONTOKEN:
ADDRESS ISPEXEC "ADDPOP ROW(1) COLUMN(2)"
ADDRESS ISPEXEC "DISPLAY PANEL(DB2TLOAS)"
IF RC = 8 THEN DISCOLCONTOKEN_EXIT = 'S'
          ELSE DO
               RESULT1 = ' ' ; COLOR1 = 'WHITE'
               RESULT2 = ' ' ; COLOR2 = 'WHITE'
               RESULT3 = ' ' ; COLOR3 = 'WHITE'
               RESULT4 = ' ' ; COLOR4 = 'WHITE'
               RESULT5 = ' ' ; COLOR5 = 'WHITE'
               RESULT6 = ' ' ; COLOR6 = 'WHITE'
               RESULT7 = ' ' ; COLOR7 = 'WHITE'
               RESULT8 = ' ' ; COLOR8 = 'WHITE'
                CONTOKEN_PKG=SUBSTR(CONTOKEN,9,8)||SUBSTR(CONTOKEN,1,8)
               CALL COMPARA_CONTOKEN
               END
ADDRESS ISPEXEC 'REMPOP'
```

```
RETURN
/*---------------------------------------CARLOS-OSORIO@EXCITE.COM--*/
COMPARA_CONTOKEN:
LIB.1 = LIB1
LIB.2 = LIB2
LIB.3 = LIB3
LIB.4 = LIB4
LIB.5 = LIB5
LIB.6 = LIB6
LIB.7 = LIB7
LIB.8 = LIB8
DO K = 1 TO 8
   RESULTK = RESULT || K
   COLORK  = COLOR  || K
   KRESULT = Ø
   IF LIB.K = ' ' THEN
                    DO
                    INTERPRET COLORK  '= WHITE'
                    ITERATE K
                    END
   ADDRESS ISPEXEC "LMINIT DATAID(DD1) DATASET('"LIB.K"') ENQ(SHR)"
   IF RC <> Ø THEN  DO
                    INTERPRET RESULTK '= DSET ERROR'
                    INTERPRET COLORK  '= RED'
                    ITERATE K
                    END

   ADDRESS ISPEXEC "LMOPEN DATAID("DD1") OPTION(INPUT)"
   IF RC <> Ø THEN  DO
                    INTERPRET RESULTK '= OPEN ERROR'
                    INTERPRET COLORK  '= RED'
                    ADDRESS ISPEXEC "LMFREE DATAID("DD1")"
                    ITERATE K
                    END

   DSNL = "'" || LIB.K || "'"
   LISTDSIRC = LISTDSI(DSNL)
   IF LISTDSIRC <= 4
      THEN DO
           IF SYSDSORG = 'PO' & SYSRECFM = 'U' & SYSLRECL = Ø
              THEN NOP
              ELSE DO
                   INTERPRET RESULTK '= NO LOADLIB'
                   INTERPRET COLORK  '= RED'
                   ADDRESS ISPEXEC "LMCLOSE DATAID("DD1")"
                   ADDRESS ISPEXEC "LMFREE DATAID("DD1")"
                   ITERATE K
                   END
           END
```

```
   ADDRESS ISPEXEC "LMMFIND DATAID("DD1") MEMBER("PACKAGE")"
   IF RC <> Ø THEN  DO
                    INTERPRET RESULTK '= NOT FOUND'
                    INTERPRET COLORK  '= RED'
                    ADDRESS ISPEXEC "LMCLOSE DATAID("DD1")"
                    ADDRESS ISPEXEC "LMFREE DATAID("DD1")"
                    ITERATE K
                    END
   LMGETRC = Ø
   KRESULT = Ø
   DO G = 1 BY 1 UNTIL LMGETRC >= 8
      ADDRESS ISPEXEC "LMGET DATAID("DD1") MODE(INVAR) DATALOC(LOADREC),
                       DATALEN(LEN) MAXLEN(28ØØØ)"
      IF RC <> Ø THEN  DO
                       LMGETRC = RC
                       KRESULT = 2
                       LEAVE
                       END
      LOADREC_HEX = C2X(LOADREC)
      IF POS(CONTOKEN_PKG,LOADREC_HEX,1)>Ø
         THEN DO
              INTERPRET RESULTK '= CORRECT'
              INTERPRET COLORK  '= TURQ'
              KRESULT = 1
              LEAVE
              END
   END
   SELECT
      WHEN KRESULT = 1 THEN NOP
      WHEN KRESULT = 3 THEN NOP
      OTHERWISE             DO
                            INTERPRET RESULTK '= INCORRECT'
                            INTERPRET COLORK  '= PINK'
                            END
   END
   ADDRESS ISPEXEC "LMCLOSE DATAID("DD1")"
   ADDRESS ISPEXEC "LMFREE DATAID("DD1")"
END
RETURN
/*-------------------------------------------------------------------*/
GRABALOG:
ADDRESS TSO "ALLOC FILE(LOG) DSNAME('"LOGPREFIX".DB2PLN.LOG') OLD REUSE"
IF RC = Ø THEN
   DO
   ADDRESS TSO "EXECIO * DISKR  LOG (STEM LOG. FINIS"
   IF RC = Ø THEN
      DO
      ULTIMO = LOG.Ø + 1
      ADDRESS TSO "EXECIO Ø DISKW LOG (OPEN FINIS"
      END
```

```
   END
ELSE
   DO
   ADDRESS TSO "ALLOC FILE(LOG) DSNAME('"LOGPREFIX".DB2PLN.LOG')" ,
               "NEW CAT REUSE UNIT(SYSDA)" ,
               "LRECL(12Ø) BLKSIZE(2796Ø) RECFM(F B) SPACE(1,1) CYL"
                IF RC <> Ø THEN
                   DO
                   ADDRESS ISPEXEC "SETMSG MSG(DBCØ41)"
                   RETURN
                END
   END
OPC       = ''
COLLIDRL  = COLLIDR
SELECT
   WHEN O = 'A' THEN OPC = 'ADD'
   WHEN O = 'I' THEN OPC = 'ADD'
   WHEN O = 'D' THEN OPC = 'DELETE'
   WHEN O = 'P' THEN DO
                     OPC = 'LIST'
                     COLLIDRL = '____________________'
                     SELECT
                         WHEN P = 'C' THEN DO
                                           OPC = 'COLLID'
                                           COLLIDRL = COLLID
                                           END
                         WHEN P = 'V' THEN OPC = 'REVIEW'
                         OTHERWISE       NOP
                     END
                     END
   OTHERWISE         DO
                     OPC = 'LIST'
                     COLLIDRL = '____________________'
                     END
END
SSIDL      = LEFT(SSID,4);
USERIDL    = LEFT(USERID(),8);
LOG.ULTIMO = LEFT(OPC,6) || SSIDL || USERIDL,
                         || LEFT(PLAN,8) || LEFT(COLLIDRL,18),
                         || DATE('S')    || TIME()
ADDRESS TSO "EXECIO  *  DISKW LOG  (STEM LOG."
ADDRESS TSO "EXECIO  Ø  DISKW LOG  (FINIS"
"FREE DDNAME(LOG)"
RETURN
```

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

Carlos Osorio Montoya
Database and MQSeries Administrator (Peru)

Axios Products has announced V4.2.0 of its SmartProduction batch analysis and tuning software.

SmartProduction helps customers reduce batch run times and increase system online time. The new version provides enhanced system analysis, allowing users to identify areas for improvement not previously available. New global reports identify, for example, COBOL programs compiled using inefficient compiler options, DB2 programs that had executed inefficiently, and programs extensively accessing a particular dataset.

For further information contact:
URL: www.axios.com/whatprod.html.

* * *

Minq Software has announced Version 4.2 of DbVisualizer, its cross-platform database tool for development, testing, and administration.

Running on Windows, Mac OS X, Linux, and Solaris, the product works on DB2, MySQL, PostgreSQL, PointBase, HSQL, Sybase ASA/ASE, McKoi, SQL Server, MaxDB, Oracle, Mimer, Informix DaffodilDB, JDataStore, Cache, FrontBase, and Pervasive databases.

Users point and click to browse the database structure, view detailed characteristics of database objects, edit table data graphically, execute arbitrary SQL statements or SQL scripts, reverse-engineer referential integrity rules graphically, or chart the database.

URL: www.dbvis.com/products/dbvis/changelog.html.

* * *

For sites converting from Informix to DB2, Ispirer Systems has released Version 3.8 of its automated migration software SQLWays.

The new release of SQLWays allows the migration of stored procedures and triggers from Informix Dynamic Server to DB2, Oracle PL/SQL, Microsoft SQL Server, Sybase Transact-SQL, and MySQL.

Besides migrating from Informix, SQLWays easily transfers data, database schema, and business-logic between DB2, Oracle, Microsoft SQL Server, Sybase, MySQL, and other databases.

For further information contact:
URL: www.ispirer.com/pr/sqlways38_informix.

* * *

IBM has renamed DB2 Information Integrator products. In future they will be known as IBM WebSphere Information Integrator.

There are no product changes: ie no technology change, nor changes in product prerequisites, nor any changes in product packaging, licensing, or pricing.

DB2 Information Integrator Content Edition becomes WebSphere Information Integrator Content Edition; DB2 Information Integrator Classic Federation becomes WebSphere Information Integrator Classic Federation, and so on.

For further information contact your local IBM representative.

* * *