147

DB2

update

January 2005

In this issue

3 DB2 UDB V8.1 FP7 (V8.2) – what's new in the world of back-ups

10 DB2 for z/OS utilities with DISP=MOD

11 Preventing negative effects after changing table privileges from public

23 Automatic DB2 imagecopy

35 DB2 utility generation process

50 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 V8.1 FP7 (V8.2) – what's new in the world of back-ups

There are a couple of enhancements to the back-up process, which are very welcome in DB2 UDB V8.1 FP7 (V8.2). These are log incorporation into the back-up (for non-DPF databases only), back-up execution throttling, and automatic back-up parameter tuning. Let's look at each of these in turn. I ran all the SQL in this article on a Windows 2000 machine running DB2 UDB 8.1 FP7 using the db2admin userid.

It has long been a bugbear that if you take an on-line back-up, you need to take the appropriate logs with you – if, for example, you are going to a disaster recovery site. An earlier enhancement of closing the log at the end of the back-up process was a welcome first step, but we still had the problem of forgetting to take the logs with us when we went to our disaster recovery site! This has now been addressed with a new parameter with the back-up command called EXCLUDE LOGS/INCLUDE LOGS (the default being EXCLUDE LOGS to keep it compatible with the old version of the command). So let's look at an example. What we will do is create the SAMPLE database, switch on archive logging, take an initial back-up, perform an insert into the DEPARTMENT table, take an on-line back-up including the logs, restore the database to another database name, and roll the logs forward. We will then be ready to select from the new DEPARTMENT table in our new database and see the row we inserted.

So the commands are:

1 Create the sample database and configure for archive logging. Note we will back up the database to a directory called c:\db2_backups:

```
>db2sampl
>db2 update db cfg for sample using logretain on
>db2 backup db sample to c:\db2_backups
```

Delete the back-up file from C:\db2_backups\SAMPLE.0\DB2\NODE0000\CATN0000\<yyyymmdd> because we don't need it.

2 Insert a new row into the DEPARTMENT table:

```
>db2 connect to sample
>db2 insert into department values('x01','Test 1','000100','E01',' ')
```

3 Back up the database in on-line mode and include the logs:

```
>db2 backup db sample online to c:\db2_backups include logs
```

Note that we are using our back-up directory, c:\db2_backups, and that we have specified the **include logs** parameter along with the **online** parameter. The back-up file will contain the back-up image and the required log files.

4 Now restore the SAMPLE database to a new database called NEWDB:

```
>db2 restore db sample from c:\db2_backups into newdb logtarget
c:\db2_logs
```

Note that we have specified a new parameter called **logtarget**. This is where DB2 will restore the logs that were stored in the back-up file. We don't have to specify a back-up timestamp because we deleted the first back-up file, so the only back-up file in the back-up directory is our on-line back-up.

5 Issue the rollforward command:

```
>db2 rollforward db newdb to end of logs and stop
```

Note that we haven't manually copied any logs anywhere.

6 Connect to NEWDB and select from the DEPARTMENT table:

```
>db2 connect to newdb

>db2 select * from department

DEPTNO DEPTNAME                    MGRNO  ADMRDEPT LOCATION
```

```
------ -------------------------- ------ -------- ---------------
A00    SPIFFY COMPUTER SERVICE DIV.  000010 A00      -
B01    PLANNING                      000020 A00      -
C01    INFORMATION CENTER            000030 A00      -
D01    DEVELOPMENT CENTER            -      A00      -
D11    MANUFACTURING SYSTEMS         000060 D01      -
D21    ADMINISTRATION SYSTEMS        000070 D01      -
E01    SUPPORT SERVICES              000050 A00      -
E11    OPERATIONS                    000090 E01      -
E21    SOFTWARE SUPPORT              000100 E01      -
x01    Test 1                        000100 E01

  10 record(s) selected.
```

You can see the row we inserted in the DEPARTMENT table in Step 2. And, *voila,* we have finished! No deciding which logs we need and where they are.

Note, if you try to take an off-line database back-up and specify the INCLUDE LOGS parameter you get the rather cryptic SQL2032N message back:

```
>db2 backup db sample to c:\db2_backups include logs
SQL2032N The "iOptions" parameter is not valid.
```

Remember that it is meaningless to ask for logs when doing an off-line back-up.

How can I check whether the back-up file contains any logs? I would use the check backup command db2ckbkp with the –h option as shown below:

```
>db2ckbkp -h <backup-file-name> | find /i "Include"
```

An example of the command is:

```
C:\db2_backups\SAMPLE.0\DB2\NODE0000\CATN0000\20040925>db2ckbkp  -h
100605.001 | find /i "Include"
        Includes Logs                  -- 1
```

This shows a back-up that includes logs (the 1). If the value is 0, logs are not included.

I hope I have shown how the INCLUDE LOGS parameter works and how simple the restore process is now, with no necessity to remember to ship those logs!

Now let’s look at back-up throttling – what does this offer you?

Throttling is not new to FP7, but I think it is worth mentioning here for completeness. There was always a question about when was the best time to run back-ups (and runstats and rebalances) because these processes can be very resource hungry. When is the best time to run them, how long do they take, and will they impact the on-line day? The throttling facility was introduced to address these concerns. Throttling allows you to control how much resource the BACKUP, RUNSTATS, and REBALANCE utilities use. You do this by defining an impact policy at the instance level (using the DBM CFG parameter UTIL_IMPACT_LIM) and then using a new parameter of the BACKUP command, UTIL_IMPACT_PRIORITY. If you don't want to throttle the back-up, simply do not specify this parameter and it will run in un-throttled mode. The DBM CFG parameter UTIL_IMPACT_LIM tells DB2 not to let the utility impact on the system by more than this amount as a percentage. If you set the value to 100, no throttling will take place. If you set it to, say, 20, DB2 will try to limit the amount of system resources available to the utility to no more than 20% of the system resources. As your workload changes so the amount of resource available will change, and DB2 will actively monitor this to make sure that the utility stays within the required boundaries. The default value of UTIL_IMPACT_LIM is 10 (but remember this has no effect unless you specify the UTIL_IMPACT_PRIORITY parameter when you execute the utility).

So let's use the SAMPLE database again.

Let's check what value we have set for UTIL_IMPACT_LIM:

```
>db2 get dbm cfg | find /i "util"
 Workload impact by throttled utilities(UTIL_IMPACT_LIM) = 10
```

Now let's back up our SAMPLE database again, specifying the util_impact_priority parameter:

```
>db2 backup db sample online to c:\db2_backups util_impact_priority 75
include logs
```

Note that the lowest priority is 1 and the highest 100 and, if you don't specify a value but specify the util_impact_priority parameter, it assumes a value of 50.

We can use the LIST UTILITIES command to check on the progress of any utilities:

```
>db2 list utilities

ID                                   = 18
Type                                 = BACKUP
Database Name                        = SAMPLE
Partition Number                     = Ø
Description                          = online db
Start Time                           = mm/dd/yyyy hh:mm:ss.tht
Throttling:
   Priority                          = 75
Progress Monitoring:
   Estimated Percentage Complete = 76
```

We can see our job and the throttling priority we gave it (75). You can see more details if you append the SHOW DETAILS parameter to the LIST UTILITIES command.

Can we change the throttling priority value? Yes, we can, using the DB2 SET UTIL_IMPACT_PRIORITY command:

```
>db2 SET UTIL_IMPACT_PRIORITY FOR <utility-id> TO <priority>
```

And we get the <*utility-id*> from the first line of the LIST UTILITIES command.

If you want the back-up to run in unthrottled mode, set the priority to zero. So running another on-line back-up:

```
>db2 backup db sample online to c:\db2_backups util_impact_priority 75
include logs

>db2 list utilities

ID                                   = 2Ø
Type                                 = BACKUP
Database Name                        = SAMPLE
Partition Number                     = Ø
Description                          = online db
Start Time                           = mm/dd/yyyy hh:mm:ss.tht
Throttling:
   Priority                          = 75
```

```
Progress Monitoring:
   Estimated Percentage Complete = 1
```

And changing the UTIL_IMPACT_PRIORITY to zero:

```
>db2 set UTIL_IMPACT_PRIORITY for 20 to 0
DB20000I  The SET UTIL_IMPACT_PRIORITY command completed successfully.
```

And checking the utility again:

```
>db2 list utilities

ID                                   = 20
Type                                 = BACKUP
Database Name                        = SAMPLE
Partition Number                     = 0
Description                          = online db
Start Time                           = mm/dd/yyyy hh:mm:ss.tht
Throttling:
   Priority                          = Unthrottled
Progress Monitoring:
   Estimated Percentage Complete = 1
```

You can see that changing UTIL_IMPACT_PRIORITY to zero has unthrottled it.

Also new in FP7 is that on the BACKUP command there is now a vendor's parameter where you can specify vendor back-up options. Check the command manual for more information on this parameter.

As we are talking about back-ups, don't forget the Automatic Maintenance facility. This takes away the headache of deciding when to run a back-up. You tell the system how it should decide to run a back-up and then leave it up to DB2 to decide when to run it. You can, of course, specify optimum/blackout periods, which gives you a high level of control. In the Automatic Maintenance facility you cannot explicitly specify that you do/do not want to include the logs – DB2 will make that decision for you.

Finally, let's turn to automatic back-up parameter selection. What we are talking about is the number of buffers, the buffer size, and the parallelism value. In previous releases you had to calculate the most appropriate value for each of these

parameters, and if you got it wrong, then your back-up would take that bit longer! So with FP7 DB2 takes the problem out of your hands. What you do is simply not specify these parameters in the BACKUP command and let DB2 decide the best values – it's as simple as that! One way to test this is to run a back-up without specifying any of these parameters and then rerun the command specifying values for these parameters and see whether you can get the back-up to run faster. To test this, I created a copy of the EMPLOYEE table called EMP, which I copied into itself a number of times to give me 2,097,152 records. I then ran the following back-ups:

```
C:\temp>db2 backup db sample online to c:\db2_backups

C:\temp>db2 backup db sample online to c:\db2_backups with 12 buffers
buffer 1024
```

The first one (letting DB2 choose the parameter values) ran in 3 minutes 15 seconds – and the second run (where I specified values for the number of buffers and the buffer size) took 4 minutes and 3 seconds. A possible reason why the second back-up ran for longer is that I might have specified bad values for the parameters – but that is the whole point of letting DB2 choose the most appropriate values. (It would be nice if DB2 told me what values it was using!)

I think this has been an excellent step – the less I have to specify on any command the better, especially if DB2 then chooses the optimum values.

I hope I have shown you how the new improvements in the world of DB2 back-ups in FP7 will make life a lot easier – give it a try!

C Leonard
Freelance Consultant (UK)

DB2 for z/OS utilities with DISP=MOD

Don’t get caught out, like this site was!

They had been running daily REORG jobs to clean out certain rows from tables and save them in sequential files, like the following:

```
REORG TABLESPACE dbname.tsname LOG NO UNLOAD CONTINUE
  SHRLEVEL NONE COPYDDN(SYSCOPY1) DISCARDDN(SYSDISC1)
  STATISTICS TABLE (ALL) INDEX (ALL)
  DISCARD FROM TABLE table-name
  WHEN ( predicate )
```

They had the SYSDISC1 DD statement in the JCL with DISP=(MOD,CATLG) and the DSN set to a name including the date. The assumption was that any second run of a REORG would write the discarded rows at the end of the output from the first run that day, because of DISP=MOD.

When the jobs were run only once per day there was indeed no problem – a new output dataset was created each time. Then, one day, they were run a second time and the REORG utility used the existing output files, but OVERWROTE the output from the first run of that day! Thus, they lost their first DISCARD output.

Since then we have tested various utilities with DISP=MOD in the JCL and in (Version 7) templates to see how they treat those files. In both DB2 Versions 6 and 7, we have found that they mostly overwrite files that are specified with DISP=MOD. The DSNTIAUL program is an exception to this where DISP=MOD works as you would expect.

This ‘feature’ of DB2 utilities does not appear to be documented in any manuals or Redbooks. You have been warned!

Ron Brown
Principal Consultant
Triton Consulting (Germany)

Preventing negative effects after changing table privileges from public

Revoking privileges on a table can sometimes have undesirable consequences, such as invalidating packages and plans and dropping views. The need for the application presented in this article appeared when the access mode to some tables had to be changed from public to a limited number of users. If the owner of the plan or package does not have the privilege needed by the plan or package from another source, that plan or package is invalidated. So here we consider revoking privileges granted on one table or a set of tables from public, only after granting the necessary privileges to the owners of the plans and packages that would become invalid. Revoking grants sometimes also implies dropping view(s) based on the table(s), so special attention has to be given to the cascading effects of dropping a view in cases of views defined on views. Recursion is applied in order to find all the views dependent on the table in question. The corresponding create view statements for all views that would be dropped are generated in the same stream after the grant and revoke statements. Of course, some additional grants on the table(s) and view(s) can be included if required. The criterion used to determine which view would be dropped is the non-existence of the record in the catalog table SYSIBM.SYSTABAUTH relating to the specific view, which has GRANTEE = GRANTOR = TCREATOR. It is the consequence of the fact explained in the *DB2 Administration Guide*: 'If the SELECT privilege on the base table is revoked from the owner of the view, the view is dropped. However, if another grantor granted the SELECT privilege to the view owner before the view was created, the view is not dropped.'

The application consists of one panel and two REXX EXECs. The main REXX EXEC just selects the input panel. Required parameters are DB2 subsystem name (DSN or DBT at our installation) and the filtering criteria for the table or group of

tables (the table owner and full or partial table name). The name of the DB2 load modules library, where program DSNTIAD resides, depends on the chosen DB2 subsystem and is installation dependent.

Selecting option 1, the second REXX EXEC (which uses program SQLISPF, published in the July and August 2001 issues of *DB2 Update*) is invoked, and a job is generated with all the necessary grant, revoke, and, optionally, create view statements. Just after submitting a job from option 1, all plans and packages that would be affected by the revoke statements will be marked as invalid. If automatic rebind is allowed (the AUTO BIND option is set to YES on the installation panel DSNTIPO), they would be automatically rebound successfully when invoked next time, since their owners at that time would have explicit authorization.

Option 2 generates a job that rebinds all plans and packages that depend on the tables that satisfy the filtering criteria. So it can be used if automatic rebind is not allowed, you wish to do it yourself, or, entirely independently of option 1, when rebinding is needed (for example, when a new index on the table is created after RUNSTATS).

The generated jobs should be submitted by a user with SYSADM authority. If grants to public on the selected table(s) were previously given by another user with SYSADM authority, then in the REVOKE statements the BY clause has to be added manually.

REVOKEP1

```
)ATTR
%  TYPE(TEXT)
[  TYPE(TEXT) INTENS(LOW)
<  TYPE(INPUT) CAPS(ON)
+  TYPE(TEXT) INTENS(LOW)
!  TYPE(OUTPUT) INTENS(LOW) CAPS(OFF)
)BODY DEFAULT(]*;)EXPAND($$)
%-$-$- REVOKE ALL -$-$-[
```

```
+DB2 subsystem        ===><Z   [
[
                                                                        [
+Table(s) Owner       ===><Z       [
[

+Table or
[
+Tables prefix        ===><Z                [

+PLAN                 ===><Z       [

+DB2 load library     ===><Z                                  [


             1. GRANT, REVOKE, CREATE VIEW

             2. REBIND PACKAGE, PLAN



+Option: <Z[ +1 or 2+

)INIT
  .ZVARS = '(DSN8SSID mowner mtables mplan db2load ZCMD)'
  .CURSOR = DSN8SSID
  VGET (DSN8SSID mowner mtables mplan db2load) SHARED
  IF (&DSN8SSID = &Z)
    &DSN8SSID = DSN
    &mplan = DSNTIA71
    &db2load = DSN71Ø.RUNLIB.LOAD
  IF (&mplan = &Z)
    &mplan = DSNTIA71
  IF (&db2load = &Z)
    &db2load = DSN71Ø.RUNLIB.LOAD
)PROC
  VER (&ZCMD,NB,LIST,1,2)
  &S = &ZCMD
  VER (&DSN8SSID,NB,LIST,DSN,DBT)
  IF (&DSN8SSID = DSN)
    &db2load = DSN71Ø.RUNLIB.LOAD
  ELSE
    &db2load = DBT71Ø.RUNLIB.LOAD
  VER (&mowner,NONBLANK)
  VER (&mtables,NONBLANK)
  VER (&mplan,NONBLANK)
  VER (&db2load,NONBLANK)
  VPUT (DSN8SSID mowner mtables mplan db2load S) SHARED
  &ZSEL = TRANS(TRUNC(&ZCMD,'.')
```

```
               1,'CMD(REVOKER1)'
               2,'CMD(REVOKER1)'
               '  ',' '
               *,'?')
)END
```

REVOKER0

```
/* rexx REVOKERØ */
address ispexec 'select panel(revokep1)'
```

REVOKER1

```
/* REXX - REVOKER1 ****************************************************/
/* TITLE    :  REVOKE ALL ON TABLE(s) FROM PUBLIC                     */
/* ****************************************************************** */
/* TRACE I                                                            */
signal on error
TabN = Ø
ViewN = Ø
CrviewN = Ø
RepeatN = Ø
GrantN = Ø
GrantVN = Ø
RevokeN = Ø
RpackN = Ø
RplanN = Ø
ADDRESS ISPEXEC "VGET (DSN8SSID mowner mtables mplan db2load S) SHARED"
userid   = userid()
tick     = ''''
outdsn   = tick||userid||'.REVOKE.CNTL'||tick
DB2V = DSN8SSID
SQLQUERY = "SELECT CREATOR, NAME ",
           "FROM SYSIBM.SYSTABLES ",
           "WHERE CREATOR = '" || STRIP(mowner) || "' AND ",
                 "NAME LIKE '" || STRIP(mtables) || "%' AND ",
                 "TYPE = 'T' ",
           "ORDER BY 1, 2 ",
           "OPTIMIZE FOR 1 ROW";
ADDRESS ISPEXEC "SELECT PGM(SQLISPF)";
If  _nrows = Ø Then Do
  say 'No record found'
  Exit
End
If  _nrows > Ø Then Do
  nn = _nrows
  Do ii = 1 To nn
    TabN = TabN + 2
```

```
    LineForTab.Ø = TabN
    k = TabN - 1
    LineForTab.k = STRIP(CREATOR.ii)
    LineForTab.TabN = STRIP(NAME.ii)
    Call process_v CREATOR.ii NAME.ii 1
  End
End
if sysdsn(outdsn) = 'OK' Then
  "alloc fi(ispfile) da("outdsn") shr "
Else Do
  "alloc fi(ispfile) da("outdsn") new ",
  "  dsorg(ps) space(1,1) tracks",
  "  recfm(F B) lrecl(132) blksize(27984)"
End
s1 = "//"||userid||"X JOB 1,MSGLEVEL=(1,1),MSGCLASS=X,REGION=4M,"
s1 = s1 || "NOTIFY=" || userid
queue s1
"execio 1 diskw ispfile"
Select
When S = 1 Then Do
  queue "//STEPØØØ1 EXEC PGM=IKJEFTØ1,DYNAMNBR=2Ø"
  queue "//SYSTSPRT DD SYSOUT=*"
  queue "//SYSPRINT DD SYSOUT=*"
  queue "//SYSUDUMP DD SYSOUT=*"
  queue "//SYSTSIN DD *"
  queue "DSN SYSTEM(" || STRIP(DSN8SSID) || ")"
  queue "RUN  PROGRAM(DSNTIAD) PLAN(" || STRIP(mplan) || ") -"
  queue "     LIB('" || STRIP(db2load) || "')"
  queue "END"
  queue "//SYSIN   DD *"
  "execio 1Ø diskw ispfile"
  Do it = 1 To TabN By 2
    kt = it + 1
    Call process_1 LineForTab.it LineForTab.kt "TABLE"
  End
  Do iv = 1 To ViewN By 2
    kv = iv + 1
    Call process_1 LineForView.iv LineForView.kv "VIEW"
  End
  Do ig = 1 To GrantN
    queue LineForGrant.ig
   "execio 1 diskw ispfile"
  End
  Do ir = 1 To RevokeN
    queue LineForRevoke.ir
   "execio 1 diskw ispfile"
  End
  Do icrv = 1 To CrviewN
    queue LineForCrview.icrv
   "execio 1 diskw ispfile"
```

```
  End
  Do igv = 1 To GrantVN
    queue LineForGrantV.igv
   "execio 1 diskw ispfile"
  End
  queue "/*"
 "execio 1 diskw ispfile"
End /* when */
When S = 2 Then Do
  queue "//STEP000001 EXEC PGM=IKJEFT01,DYNAMNBR=20,COND=(4,LT)"
  queue "//SYSTSPRT DD SYSOUT=*"
  queue "//SYSPRINT DD SYSOUT=*"
  queue "//SYSUDUMP DD SYSOUT=*"
  queue "//SYSTSIN DD *"
  queue "DSN SYSTEM(" || STRIP(DSN8SSID) || ")"
 "execio 6 diskw ispfile"
  Do it = 1 To TabN By 2
    kt = it + 1
    Call process_2 LineForTab.it LineForTab.kt
  End
  Do iv = 1 To ViewN By 2
    kv = iv + 1
    Call process_2 LineForView.iv LineForView.kv
  End
  Do ip = 1 To RpackN
    queue LineForPack.ip
   "execio 1 diskw ispfile"
  End
  Do ipl = 1 To RplanN
    queue LineForPlan.ipl
   "execio 1 diskw ispfile"
  End
  queue "END"
 "execio 1 diskw ispfile"
End /* when */
Otherwise
End  /* select */
queue "//"
"execio 1 diskw ispfile"
"execio 0 diskw ispfile(finis"
signal off error
"free  fi(ispfile)"
"ispexec edit dataset("outdsn")"
signal on error
"ispexec lmerase dataset("outdsn")"
Exit
error:
  say 'error on line:' sigl '  ,rc:' rc
Exit 20
```

```
process_v:
  parse arg Tbcre Tbname Lvl
  SQLQUERY = "SELECT COUNT(*) ",
             "FROM SYSIBM.SYSVIEWDEP ",
             "WHERE BCREATOR = '" || Tbcre || "' AND ",
                   "BNAME = '" || Tbname || "'"
  ADDRESS ISPEXEC "SELECT PGM(SQLISPF)";
  NUMROW = _vn1.1
  If NUMROW = Ø Then Return;
  SQLQUERY = "SELECT DCREATOR, DNAME, DTYPE, BCREATOR, BNAME ",
             "FROM SYSIBM.SYSVIEWDEP ",
             "WHERE BCREATOR = '" || Tbcre || "' AND ",
                   "BNAME = '" || Tbname || "'"
  ADDRESS ISPEXEC "SELECT PGM(SQLISPF)";
  Do i = 1 To NUMROW
    Do j = 1 To RepeatN
      If LineForRepeat.j = DCREATOR.i||DNAME.i||DTYPE.i||,
                           BCREATOR.i||BNAME.i
      Then Return;
    End
    RepeatN = RepeatN + 1
    LineForRepeat.Ø = RepeatN
    LineForRepeat.RepeatN = DCREATOR.i||DNAME.i||DTYPE.i||,
                            BCREATOR.i||BNAME.i
    found = Ø
    Do j = 1 To ViewN By 2
      k = j + 1
      If LineForView.j = DCREATOR.i &,
         LineForView.k = DNAME.i Then found = 1;
    End
    if found = Ø Then Do
      ViewN = ViewN + 2
      LineForView.Ø = ViewN
      k = ViewN - 1
      LineForView.k = STRIP(DCREATOR.i)
      LineForView.ViewN = STRIP(DNAME.i)
    End
    Call process_v DCREATOR.i DNAME.i Lvl + 1
  End
  Return

process_1:
  parse arg Objcre Objname Obj
  If Obj = "VIEW" Then Do
    SQLQUERY = ,
   "SELECT COUNT(*) ",
   "FROM SYSIBM.SYSTABAUTH ",
   "WHERE TCREATOR = '" || Objcre || "' AND ",
         "TTNAME = '" || Objname || "' AND ",
         "GRANTOR = GRANTEE AND ",
```

```
        "GRANTEE = TCREATOR"
    ADDRESS ISPEXEC "SELECT PGM(SQLISPF)";
    If _vn1.1 = Ø Then Do
      SQLQUERY = ,
     "SELECT SEQNO, TEXT ",
     "FROM SYSIBM.SYSVIEWS ",
     "WHERE CREATOR = '" || Objcre || "' AND ",
           "NAME = '" || Objname || "' ",
     "ORDER BY 1"
      ADDRESS ISPEXEC "SELECT PGM(SQLISPF)";
      Do iw = 1 To _nrows
        If iw = 1 Then Do
          CrviewN = CrviewN + 1
          LineForCrview.Ø = CrviewN
          LineForCrview.CrviewN = " "
        End
        wN = WORDS(TEXT.iw)
        Do jw = 1 To wN
          If LENGTH(LineForCrview.CrviewN) +,
             LENGTH(WORD(TEXT.iw, jw)) > 6Ø Then Do
            CrviewN = CrviewN + 1
            LineForCrview.Ø = CrviewN
            If LENGTH(WORD(TEXT.iw, jw)) > Ø
            Then LineForCrview.CrviewN = "   " || WORD(TEXT.iw, jw)
          End
          Else Do
            If LENGTH(WORD(TEXT.iw, jw)) > Ø
            Then LineForCrview.CrviewN = LineForCrview.CrviewN ||,
                                         " " || WORD(TEXT.iw, jw)
          End
        End
      End
      If _nrows > Ø
      Then LineForCrview.CrviewN = LineForCrview.CrviewN || ";"
    End
  End
  SQLQUERY = ,
 "SELECT 'GRANT ' || CASE WHEN MAX(A.SELECTAUTH) = 'Y' ",
                          "THEN 'SELECT' ",
                          "ELSE '' ",
                     "END || ",
                     "CASE WHEN MAX(A.INSERTAUTH) = 'Y' AND ",
                               "MAX(A.SELECTAUTH) = 'Y' ",
                          "THEN ',INSERT' ",
                          "ELSE CASE WHEN MAX(A.INSERTAUTH) = 'Y' ",
                                     "THEN 'INSERT' ",
                                     "ELSE '' ",
                                "END ",
                     "END || ",
                     "CASE WHEN MAX(A.DELETEAUTH) = 'Y' AND ",
```

```
                          "(MAX(A.SELECTAUTH) = 'Y' OR ",
                           "MAX(A.INSERTAUTH) = 'Y') ",
                      "THEN ',DELETE' ",
                      "ELSE CASE WHEN MAX(A.DELETEAUTH) = 'Y' ",
                                "THEN 'DELETE' ",
                                "ELSE '' ",
                           "END ",
                 "END || ",
                 "CASE WHEN MAX(A.UPDATEAUTH) = 'Y' AND ",
                          "(MAX(A.SELECTAUTH) = 'Y' OR ",
                           "MAX(A.INSERTAUTH) = 'Y' OR ",
                           "MAX(A.DELETEAUTH) = 'Y') ",
                      "THEN ',UPDATE' ",
                      "ELSE CASE WHEN MAX(A.UPDATEAUTH) = 'Y' ",
                                "THEN 'UPDATE' ",
                                "ELSE '' ",
                           "END ",
                 "END || ",
 "' ON TABLE ' || STRIP(A.TCREATOR) || '.' || STRIP(A.TTNAME) || ",
   "' TO ' || STRIP(B.CREATOR) || ';' ",
"FROM SYSIBM.SYSTABAUTH A, SYSIBM.SYSPLAN B ",
"WHERE A.TCREATOR = '" || Objcre || "' AND ",
      "A.TTNAME = '" || Objname || "' AND ",
      "A.GRANTEETYPE = 'P' AND ",
      "A.GRANTEE = B.NAME ",
"GROUP BY B.CREATOR, A.TCREATOR, A.TTNAME ",
"UNION ",
"SELECT 'GRANT ' || CASE WHEN MAX(A.SELECTAUTH) = 'Y' ",
                      "THEN 'SELECT' ",
                      "ELSE '' ",
                 "END || ",
                 "CASE WHEN MAX(A.INSERTAUTH) = 'Y' AND ",
                           "MAX(A.SELECTAUTH) = 'Y' ",
                      "THEN ',INSERT' ",
                      "ELSE CASE WHEN MAX(A.INSERTAUTH) = 'Y' ",
                                "THEN 'INSERT' ",
                                "ELSE '' ",
                           "END ",
                 "END || ",
                 "CASE WHEN MAX(A.DELETEAUTH) = 'Y' AND ",
                          "(MAX(A.SELECTAUTH) = 'Y' OR ",
                           "MAX(A.INSERTAUTH) = 'Y') ",
                      "THEN ',DELETE' ",
                      "ELSE CASE WHEN MAX(A.DELETEAUTH) = 'Y' ",
                                "THEN 'DELETE' ",
                                "ELSE '' ",
                           "END ",
                 "END || ",
                 "CASE WHEN MAX(A.UPDATEAUTH) = 'Y' AND ",
                          "(MAX(A.SELECTAUTH) = 'Y' OR ",
```

```
                              "MAX(A.INSERTAUTH) = 'Y' OR ",
                              "MAX(A.DELETEAUTH) = 'Y') ",
                         "THEN ',UPDATE' ",
                         "ELSE CASE WHEN MAX(A.UPDATEAUTH) = 'Y' ",
                                   "THEN 'UPDATE' ",
                                   "ELSE '' ",
                              "END ",
                    "END || ",
 "' ON TABLE ' || STRIP(A.TCREATOR) || '.' || STRIP(A.TTNAME) || ",
   "' TO ' || STRIP(B.OWNER) || ';' ",
"FROM SYSIBM.SYSTABAUTH A, SYSIBM.SYSPACKAGE B ",
"WHERE A.TCREATOR = '" || Objcre || "' AND ",
      "A.TTNAME = '" || Objname || "' AND ",
      "A.GRANTEETYPE = 'P' AND ",
      "A.COLLID = B.COLLID ",
"GROUP BY B.OWNER, A.TCREATOR, A.TTNAME ",
"ORDER BY 1 ",
"OPTIMIZE FOR 1 ROW";
 ADDRESS ISPEXEC "SELECT PGM(SQLISPF)";
 If  _nrows > Ø Then Do
   Do i = 1 To _nrows
     found = Ø
     If Obj = "VIEW" Then Do
       Do j = 1 To GrantVN
         If LineForGrantV.j = value(_vn1"."strip(i,l,'Ø'))
         Then found = 1;
       End
     End
     Else Do
       Do j = 1 To GrantN
         If LineForGrant.j = value(_vn1"."strip(i,l,'Ø'))
         Then found = 1;
       End
     End
     If found = Ø Then Do
       If Obj = "VIEW" Then Do
         GrantVN = GrantVN + 1
         LineForGrantV.Ø = GrantVN
         LineForGrantV.GrantVN = value(_vn1"."strip(i,l,'Ø'))
       End
       Else Do
         GrantN = GrantN + 1
         LineForGrant.Ø = GrantN
         LineForGrant.GrantN = value(_vn1"."strip(i,l,'Ø'))
       End
     End
   End
 End
 SQLQUERY = ,
```

```
 "SELECT 'REVOKE ALL ON ' || STRIP(CREATOR) || '.' || STRIP(NAME) ||",
      " ' FROM PUBLIC;' ",
 "FROM SYSIBM.SYSTABLES ",
 "WHERE CREATOR = '" || Objcre || "' AND ",
       "NAME = '" || Objname || "' ",
 "ORDER BY 1 ",
 "OPTIMIZE FOR 1 ROW";
  ADDRESS ISPEXEC "SELECT PGM(SQLISPF)";
  If  _nrows > Ø Then Do
    Do i = 1 To _nrows
      found = Ø
      Do j = 1 To RevokeN
        If LineForRevoke.j = value(_vn1"."strip(i,l,'Ø'))
        Then found = 1;
      End
      If found = Ø Then Do
        RevokeN = RevokeN + 1
        LineForRevoke.Ø = RevokeN
        LineForRevoke.RevokeN = value(_vn1"."strip(i,l,'Ø'))
      End
    End
  End
  Return

process_2:
  parse arg Objcre Objname
  SQLQUERY = ,
 "SELECT 'REBIND PACKAGE (' || STRIP(COLLID) || '.' || ",
        "STRIP(NAME) || ')' ",
 "FROM SYSIBM.SYSPACKAGE A ",
 "WHERE EXISTS(SELECT * ",
              "FROM SYSIBM.SYSPACKDEP ",
              "WHERE DNAME = A.NAME AND ",
                    "BQUALIFIER = '" || Objcre || "' AND ",
                    "BNAME = '" || Objname || "') ",
 "UNION ",
 "SELECT 'REBIND PACKAGE (' || STRIP(COLLID) || '.' || ",
        "STRIP(NAME) || ')' ",
 "FROM SYSIBM.SYSPACKSTMT A ",
 "WHERE STMT LIKE '%"||Objcre||"%"||Objname||"%' AND ",
       "NOT EXISTS(SELECT * ",
                  "FROM SYSIBM.SYSPACKDEP ",
                  "WHERE DCOLLID = A.COLLID AND ",
                        "DNAME = A.NAME) ",
 "ORDER BY 1 ",
 "OPTIMIZE FOR 1 ROW";
  ADDRESS ISPEXEC "SELECT PGM(SQLISPF)";
  If  _nrows > Ø Then Do
    Do i = 1 To _nrows
```

```
      found = Ø
      Do j = 1 To RpackN
        If LineForPack.j = value(_vn1"."strip(i,l,'Ø'))
        Then found = 1;
      End
      If found = Ø Then Do
        RpackN = RpackN + 1
        LineForPack.Ø = RpackN
        LineForPack.RpackN = value(_vn1"."strip(i,l,'Ø'))
      End
    End
  End
  SQLQUERY = ,
"SELECT STRIP(A.NAME) AS NAME, ",
       "VALUE(STRIP(B.COLLID), ' ') AS COLLID, ",
       "VALUE(STRIP(B.NAME), ' ') AS NAME1 ",
"FROM SYSIBM.SYSPLAN A LEFT OUTER JOIN SYSIBM.SYSPACKLIST B ",
  "ON A.NAME = B.PLANNAME ",
"WHERE EXISTS(SELECT * ",
             "FROM SYSIBM.SYSPLANDEP ",
             "WHERE DNAME = A.NAME AND ",
                   "BCREATOR = '" || Objcre || "' AND ",
                   "BNAME = '" || Objname || "') ",
"UNION ",
"SELECT STRIP(A.PLNAME), ' ', ' ' ",
"FROM SYSIBM.SYSSTMT A ",
"WHERE TEXT LIKE '%"||Objcre||"%"||Objname||"%' AND ",
      "NOT EXISTS(SELECT * ",
                 "FROM SYSIBM.SYSPLANDEP ",
                 "WHERE DNAME = A.PLNAME) ",
"ORDER BY 1 ",
"OPTIMIZE FOR 1 ROW";
  ADDRESS ISPEXEC "SELECT PGM(SQLISPF)";
  If  _nrows > Ø Then Do
    MNAME = ""
    Do i = 1 To _nrows
      If MNAME <> NAME.i Then Do
        If i > 1 Then Do
          j = i - 1
          If COLLID.j <> ' ' Then stmt = stmt || ')'
          found = Ø
          Do jj = 1 To RplanN
            If LineForPlan.jj = stmt
            Then found = 1;
          End
          If found = Ø Then Do
            RplanN = RplanN + 1
            LineForPlan.Ø = RplanN
            LineForPlan.RplanN = stmt
          End
```

```
        End
        stmt = 'REBIND PLAN (' || NAME.i || ')'
        If COLLID.i <> ' '
        Then stmt = stmt || ' PKLIST(' || COLLID.i || '.' || NAME1.i
        MNAME = NAME.i
      End
      Else Do
        If LENGTH(stmt) + LENGTH(COLLID.i) + LENGTH(NAME1.i) > 72
        Then Do
          RplanN = RplanN + 1
          LineForPlan.Ø = RplanN
          LineForPlan.RplanN = stmt
          stmt = '             '
        End
        Else stmt = stmt || ',' || COLLID.i || '.' || NAME1.i
      End
    End
    j = i - 1
    If COLLID.j <> ' ' Then stmt = stmt || ')'
    found = Ø
    Do jj = 1 To RplanN
      If LineForPlan.jj = stmt
      Then found = 1;
    End
    If found = Ø Then Do
      RplanN = RplanN + 1
      LineForPlan.Ø = RplanN
      LineForPlan.RplanN = stmt
    End
  End
  Return
```

Nikola Lazovic and Gordana Kozovic
DB2 System Administrators
Postal Savings Bank (Serbia and Montenegro)

Automatic DB2 imagecopy

Our goal was to do an IMAGECOPY of DB2 tablespaces automatically (eg every Friday night). The criteria we set were:

- The imagecopy will be directed to DASD rather than to tape.

- A single START IMDB2 will launch a series of jobs using the IMDB CLIST, which will execute in batch mode. Please follow the instructions in the body of the IMDB CLIST and change the parameter values as indicated.
- The copies have the format ELE.COPIA.*3rd*.*4th*.AL*day*.T*hhmmss*, where *3rd* is the third qualifier of the original VSAM DB2 object, *4th* is the fourth qualifier, *day* is the day, and *hhmmss* is the time of the copy.
- The copies must be SMS managed. If yours are not, you should carefully modify the CLIST and JCL code in order to provide UNIT and VOLSER information where required.
- The size of the copies will be automatically calculated by IMDB CLIST.
- Only changed tablespaces will be saved (that is, CHBIT=YES objects).
- Old copies will be deleted, in order to maintain only the last four copies (you can modify this limit in the CLIST).
- As a final step, a RUNSTATS will be done for the successfully-copied object.

Are you ready? OK, first of all, place the IMDB2 JCL in a partitioned library (see IMDB2 below).

A simple trick to launch a batch job through a started task is as follows:

```
//          EXEC PGM=IEBGENER
//SYSPRINT  DD  DUMMY
//SYSIN     DD  DUMMY
//SYSUT1    DD  DSN=SYS1.RACF.PROCJOBS(IMDB2),DISP=SHR
//SYSUT2    DD  SYSOUT=(A,INTRDR),DCB=BLKSIZE=80
```

Put these five lines of JCL in SYS1.PROCLIB or in another proclib dataset. Then, schedule its automatic start through a scheduler (OPC, CA-7, etc) or through JES2 commands ($ta…), or directly by typing it on an operator console (S IMDB2).

Put this IMDB2 code in a partitioned dataset:

```
//IMDB2   JOB (4Ø,LØ421),'SOF  TTT',MSGCLASS=Q,MSGLEVEL=(1,1),
//             REGION=ØM,CLASS=B,USER=DB2UT,PASSWORD=DB2TEST
//*****************************************************************
//* IMAGECOPY OF MODIFIED DB2 TABLESPACES (CHIND=YES)
//* DB2 OPEN IN MAINT
//* change characters in lower case to reflect your installation
//*****************************************************************
//SAVELIST EXEC ACBJBAOB,TABL2=your.ISPTLIB2
//SYSTSIN  DD *
  DELETE 'EE.DATASET.REPORT.YES'
  PROFILE PREFIX(your-userid-prefix)
  ISPSTART CMD(ACBQBAI2 SAVE TABDB2 +
  SOURCENL(1) SOURCEGL(2) CHGIND(EQ YES) +
  VTOCVSER(your-volumes)  +
  DSN('**.DSNDBD.DB*.T*.**')) +
  NEWAPPL(DGT) BATSCRW(132) BATSCRD(27) BREDIMAX(3) BDISPMAX(99999999)
/*
//*****************************************************************
//ALCISPFL EXEC PGM=IEFBR14
//ISPFILE DD  DSN=EE.DATASET.REPORT.YES,DISP=(NEW,CATLG),
//     BLKSIZE=Ø,SPACE=(TRK,(3,1)),RECFM=FBA,LRECL=133,UNIT=SYSDA
//SYSPRINT DD SYSOUT=*
//SYSIN  DD *
/*
//*****************************************************************
//GENREP  EXEC ACBJBAOB,TABL2=your.ISPTLIB2
//ISPFILE DD  DSN=EE.DATASET.REPORT.YES,DISP=OLD
//SYSTSIN  DD *
PROFILE PREFIX(your-userid-prefix)
ISPSTART CMD(ACBQBAR1 TABDB2) +
BATSCRW(132) BATSCRD(27) BREDIMAX(3) BDISPMAX(99999999)
/*
//SYSIN DD  *
DSN
VOLSER
CHGIND
ALLOCSP
NUMEXT
/*
//*****************************************************************
//IMDB     EXEC   PGM=IKJEFTØ1,REGION=4ØØØK,DYNAMNBR=8Ø
//STEPLIB  DD   DISP=SHR,DSN=your.ISPLOAD             /* USER */
//         DD   DISP=SHR,DSN=ISP.SISPLOAD             /* ISPF */
//         DD   DISP=SHR,DSN=ISP.SISPLPA              /* ISPF */
//SYSPRINT DD   SYSOUT=*
//SYSTSPRT DD   SYSOUT=*
//SYSIN    DD   DUMMY
//SYSPROC  DD   DISP=SHR,DSN=your.ISPCLIB             /* USER */
```

```
//          DD   DISP=SHR,DSN=ISP.SISPCLIB             /* ISPF */
//ISPPLIB   DD   DISP=SHR,DSN=your.ISPPLIB             /* USER */
//          DD   DISP=SHR,DSN=ISP.SISPPENU             /* ISPF */
//ISPMLIB   DD   DISP=SHR,DSN=your.ISPMLIB             /* USER */
//          DD   DISP=SHR,DSN=ISP.SISPMENU             /* ISPF */
//ISPSLIB   DD   DISP=SHR,DSN=your.ISPSLIB             /* USER */
//          DD   DISP=SHR,DSN=ISP.SISPSLIB             /* ISPF */
//ISPTLIB   DD   DISP=SHR,DSN=your.ISPTLIB             /* USER */
//          DD   DISP=SHR,DSN=ISP.SISPTENU             /* ISPF */
//ISPTABL   DD   DISP=SHR,DSN=your.ISPTLIB             /* USER */
//ISPLOG    DD SYSOUT=*,DCB=(LRECL=125,BLKSIZE=129,RECFM=VA)
//ISPPROF   DD LIKE=userid.ISPF.ISPPROF,DISP=(,KEEP,DELETE),
//          DSN=&&PROF
//SYSTSIN   DD *
 PROF PREFIX(your-userid-prefix)
 ISPSTART CMD(IMDB)
//*******************************************************************
```

Set *your-userid-prefix* to a valid TSO userid prefix, and *your-volumes* to a set of volumes containing the tablespaces to be saved. Please supply your personal ISPF datasets.

The *VTOCVSER* parameter specifies the VOLSERs whose VTOCs are to be searched. You may specify from one to six alphanumeric characters and an asterisk for filtering. For example, DB20*E means all VOLSERs like DB201E, DB20ZE, and so on.

Job IMDB2 consists in four steps:

1 Step SAVELIST acquires and saves a list named TABDB2 using the NAVIQUEST facility (obviously, you may change the list name as you wish). The result is an ISPF list containing all DB2 VSAM objects modified since the last back-up (CHBIT=YES). I used the ACBJBAOB procedure (see ACBJBAOB) as documented in *DFSMS/MVS NAVIQUEST User's Guide* SC26-7194

2 Step ALCISPFL allocates the new sequential dataset to be used in order to obtain a detailed output from TABDB2 list.

3 Step GENREP generates the detailed report in EE.DATASET.REPORT.YES (if you modify this dsname, be careful to reflect the change in the IMDB CLIST!).

4 Step IMDB executes in batch the IMDB CLIST (see IMDB). If you wish to launch IMDB CLIST in foreground, it works fine with the last report produced; in other words, the contents of the EE.DATASET.REPORT.YES dataset.

Put this ACBJBAOB code in a PROCLIB dataset (eg SYS1.PROCLIB).

```
//ACBJBAOB PROC CLIST1='SYS1.DGTCLIB',
//             PLIB1='SYS1.DGTPLIB',
//             MLIB1='SYS1.DGTMLIB',
//             LOAD1='SYS1.DGTLLIB',
//             TABL2='your.ISPTLIB'
//*******************************************************************/
//* THIS IS A COPY OF SYS1.SACBCNTL(ACBJBAOB)                       */
//* PROC STEP STEP1 - INVOKES IKJEFT01                              */
//*******************************************************************/
//STEP1    EXEC PGM=IKJEFT01,DYNAMNBR=50,REGION=(6144K),TIME=(300)
//STEPLIB  DD DSN=&LOAD1,DISP=SHR
//         DD   DISP=SHR,DSN=ISP.SISPLPA
//         DD   DISP=SHR,DSN=ISP.SISPLOAD
//ISPPLIB  DD DSN=&PLIB1,DISP=SHR
//         DD DSN=ISP.SISPPENU,DISP=SHR            /* ISPF PANELS    */
//         DD DSN=SYS1.DGTMLIB,DISP=SHR            /* ISMF MESSAGES  */
//ISPMLIB  DD DSN=&MLIB1,DISP=SHR
//         DD DSN=ISP.SISPMENU,DISP=SHR            /* ISPF MESSAGES  */
//         DD DSN=SYS1.DGTMLIB,DISP=SHR            /* ISMF MESSAGES  */
//ISPSLIB  DD DSN=ISP.SISPSENU,DISP=SHR            /* ISPF SKELETONS */
//         DD DSN=SYS1.DGTSLIB,DISP=SHR            /* ISMF SKELETONS */
//ISPTLIB  DD DSN=&TABL2,DISP=SHR
//         DD DSN=ISP.SISPTENU,DISP=SHR            /* ISPF TABLES    */
//         DD DSN=SYS1.DGTTLIB,DISP=SHR            /* ISMF TABLES    */
//SYSPROC  DD DSN=&CLIST1,DISP=SHR
//         DD DSN=ISP.SISPCLIB,DISP=SHR            /* ISPF CLISTS    */
//         DD DSN=SYS1.DGTCLIB,DISP=SHR            /* ISMF CLISTS    */
//ISPTABL  DD DSN=&TABL2,DISP=SHR
//SYSPRINT DD SYSOUT=*
//SYSUDUMP DD SYSOUT=*
//SYSTSPRT DD SYSOUT=*
//ISPLOG   DD SYSOUT=*,DCB=(LRECL=125,BLKSIZE=129,RECFM=VA)
//ISPPROF  DD LIKE=userid.ISPF.ISPPROF,DISP=(,KEEP,DELETE),
//         DSN=&&PROF
//         PEND
```

Put this IMDB code in a CLIST dataset (DDNAME SYSPROC of logon proc).

```
PROC 0 DEBUG
/*- SETUP FOR DEBUG IF REQUESTED ----------------------------------*/
```

```
  CONTROL NOMSG NOLIST NOFLUSH END(ENDO) NOCONLIST NOPROMPT
  IF &DEBUG  =  DEBUG THEN +
   CONTROL MSG LIST NOFLUSH END(ENDO) PROMPT SYMLIST CONLIST
/*- END OF SETUP ------------------------------------------------*/
/* . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . */
/*                                                               */
/* IMDB: IMAGECOPY DB2, ONLY FOR MODIFIED TABLESPACE. THIS CLIST */
/*       PREPARES THE JCL TO BE RUN IN ORDER TO OBTAIN THE COPIES */
/*       LOOK AT ALL ">>>TO BE CHANGED<<<" PARMS AND ALTER THEM! */
/*                                                               */
/* . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . */
ISPEXEC VGET ZSYSID
/* . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . */
/* >>>TO BE CHANGED<<<                                           */
/* SET &DB2SYS TO THE VALUE IN SYS1.PARMLIB(IEFSSNØØ) THAT REFLECT */
/* THE SUBNAME IN USE FOR YOUR DB2 SUBSYSTEM.                    */
/* I ASSUME: 'DB2' PLUS THE LAST CHARACTER OF SYSID              */
/* . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . */
SET &DB2SYS=DB2&SUBSTR(4:4,&ZSYSID)
/* . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . */
/* >>>TO BE CHANGED<<<                                           */
/* SET &V TO THE VALUE YOU PREFER (I SET IT TO 25)               */
/* &V IS THE NUMBER OF TABLESPACES TO BE SAVED IN A SINGLE JOB   */
/* . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . */
SET &V=25
SET  &DT=&SUBSTR(7:8,&SYSDATE)&SUBSTR(1:2,&SYSDATE)&SUBSTR(4:5,&SYSDATE)
SET  &TM=&SUBSTR(1:2,&SYSTIME)&SUBSTR(4:5,&SYSTIME)&SUBSTR(7:8,&SYSTIME)
/* . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . */
/* >>>TO BE CHANGED<<<                                           */
/* SET &ISPOLIB TO YOUR PARTITIONED OUTPUT SKEL DATASET          */
/* SET &YESC TO THE OUTPUT OF NAVIQUEST LIST (STEP GENREP OF IMDB2) */
/* SET &INCL TO A PROCLIB DATASET                                */
/* SET &PALB TO A PARMLIB DATASET (PO-E TO AVOID ABENDS)         */
/* . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . */
  SET &ISPOLIB=ISP.USER.ISPOLIB
  SET &YESC=EE.DATASET.REPORT.YES
  SET &INCL=CASMEZ.PROCLIB
  SET &PALB=EE.PARMLIB
  ALLOC F(ISPFILE) DA('&ISPOLIB.') SHR REU
  ALLOC F(YESC) DA('&YESC') SHR REU
  ALLOC F(INCL) DA('&INCL(SYSCOP)') SHR REU
  OPENFILE YESC INPUT
  OPENFILE INCL OUTPUT
  SET &NN=&V
  ALLOC F(SYSC) DA('&PALB(SYSC&NN)') SHR REU
  ALLOC F(MODI) DA('&PALB(MODI&NN)') SHR REU
  ALLOC F(RUNS) DA('&PALB(RUNS&NN)') SHR REU
  OPENFILE SYSC OUTPUT
  OPENFILE RUNS OUTPUT
  OPENFILE MODI OUTPUT
```

```
 ERROR DO
  IF &LASTCC = 400 THEN DO
                        SET &FI=Y
                        GOTO FIN
                       ENDO
  RETURN
     ENDO
/* . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . */
/* READ INPUT FILE CREATED BY NAVIQUEST AND SEARCH FOR DB2 OBJECTS   */
/* I ASSUME THE SECOND QUALIFIER FOR THEM IS ALWAYS "DSNDBD"         */
/* . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . */
LEGGI: +
 GETFILE YESC
   SET &L = &SYSINDEX(.DSNDBD.,&YESC)
   IF &L EQ 0 THEN GOTO LEGGI
      SET &S=&L+8
   SET &L = &SYSINDEX(.I0001.,&YESC)
   IF &L EQ 0 THEN GOTO LEGGI
      SET TAB=&SUBSTR(&S:&L-1,&YESC)
   SET &L = &SYSINDEX(.TC,&TAB)
   IF &L NE 0 THEN GOTO TABOK
   SET &L = &SYSINDEX(.TE,&TAB)
   IF &L NE 0 THEN GOTO TABOK
   SET &L = &SYSINDEX(.TL,&TAB)
   IF &L NE 0 THEN GOTO TABOK
   SET &L = &SYSINDEX(.TR,&TAB)
   IF &L NE 0 THEN GOTO TABOK
   SET &L = &SYSINDEX(.TS,&TAB)
   IF &L NE 0 THEN GOTO TABOK
        GOTO LEGGI
TABOK: +
   SET &N = &N+1
      SET TRK=&SUBSTR(64:70,&YESC)
/* . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . */
/* >>>TO BE CHANGED<<<                                               */
/* SET &TV TO THE TRACK VALUE - I USE 3380 GEOMETRY, SO 47KB/TRK     */
/*                   IF YOU USE 3390 GEOMETRY, THEN PUT 57KB/TRK     */
/* . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . */
   SET &TV = 47
      SET TRK=&TRK/&TV
      IF &TRK=0 OR &TRK=    THEN SET &TRK=1
      SET SEK=&TRK/15
      IF &SEK=0 OR &SEK=    THEN SET &SEK=1
      SET &I = &I+1
/* . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . */
/* PREPARING CARDS FOR IMDBXXXX JCL                                  */
/* . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . */
      SET &INCL=&STR(//SYSCOP&I DD DSN=ELE.COPIA.&TAB..AL&DT..T&TM,)
      PUTFILE INCL
      SET &INCL=&STR//  DISP=(,CATLG),SPACE=(TRK,(&TRK,&SEK),RLSE))
```

```
      PUTFILE INCL
      SET &MODI=&STR(MODIFY RECOVERY TABLESPACE &TAB  DELETE AGE (90))
      PUTFILE MODI
      SET &SYSC=&STR(COPY TABLESPACE &TAB COPYDDN SYSCOP&I )
      PUTFILE SYSC
      SET &RUNS=&STR(RUNSTATS TABLESPACE &TAB  INDEX (ALL))
      PUTFILE RUNS
/* . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . */
/* >>>TO BE CHANGED<<<                                             */
/* LISTCAT TO SEARCH AND DELETE OLD COPIES                         */
/* I MAINTAIN FOUR COPIES (8 SYSOUTLINES), YOU MAY CHANGE IT       */
/* . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . */
SET &SYSOUTTRAP=100
LISTC LVL('ELE.COPIA.&TAB')
SET &SYSOUTTRAP=0
   IF &SYSOUTLINE < 8  THEN GOTO DOP
SET LIN=&SYSOUTLINE-7
   DO UNTIL &LIN < 2
   SET LS=&&SYSOUTLINE&LIN
   SET LL=&LENGTH(&STR(&LS))
   SET &DS=&SUBSTR(17:&LL,&LS)
   DEL '&DS'
 WRITE *** '&DS' OLD COPY DELETED ***
   SET &LIN=&LIN-2
 ENDO
DOP:  IF  &I = &V THEN DO
/* . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . */
/* WHEN &V NUMBER IS REACHED, THEN SUBMIT AN IMDBXXXX JOB          */
/* . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . */
FIN: +
          CLOSFILE INCL
          CLOSFILE SYSC
          CLOSFILE MODI
          CLOSFILE RUNS
          IF &I=0  THEN GOTO FINE
          SET &I = 0
          IF &FI=Y THEN SET &N=(&N+&V)/&V*&V
          SET &IMB=IMDB&N
          SET &MOD=MOD&N
          SET &RUN=RUN&N
          IF &LENGTH(&IMB)=5 THEN SET &IMB=IMDB000&N
          IF &LENGTH(&IMB)=6 THEN SET &IMB=IMDB00&N
          IF &LENGTH(&IMB)=7 THEN SET &IMB=IMDB0&N
          ISPEXEC FTOPEN
          ISPEXEC FTINCL IMDB
          ISPEXEC FTCLOSE NAME(&IMB)
          SET &NN=&NN+&V
          SUB '&ISPOLIB(&IMB)'
  WRITE   >>> &IMB SUBMITTED
          IF &FI=Y THEN GOTO FINE
```

```
  ALLOC F(SYSC) DA('&PALB(SYSC&NN)') SHR REU
  ALLOC F(MODI) DA('&PALB(MODI&NN)') SHR REU
  ALLOC F(RUNS) DA('&PALB(RUNS&NN)') SHR REU
          OPENFILE INCL OUTPUT
          OPENFILE SYSC OUTPUT
          OPENFILE RUNS OUTPUT
          OPENFILE MODI OUTPUT
                    ENDO
      GOTO LEGGI
FINE: +
 CLOSFILE YESC
 FREE F(YESC INCL SYSC MODI RUNS)
  END
```

Put the following IMDB SKELETON code in a member of an ISPSLIB concatenated dataset. It is used in the IMDB CLIST.

```
)CM   IMDB - DB2 IMAGECOPY
)CM   THIS SKELETON IS USED TO SUBMIT COPY OF MODIFIED TABLESPACE
)CM   (CHIND=YES) SEE "IMDB" CLIST
)CM
//&IMB     JOB (4Ø,LØ421),'DB2 DB2 DB2',MSGCLASS=Q,MSGLEVEL=(1,1),
//  REGION=3ØØØK,CLASS=B,USER=DB2UT,PASSWORD=DB2TEST,NOTIFY=LØ419Ø6
//*
//*-------------------------------------------------------------------*
//*  &DB2SYS IS PASSED BY IMDB CLIST                                  *
//*  &IMB    IS PASSED BY IMDB CLIST                                  *
//*  CHANGE JOBLIB TO REFLECT YOUR INSTALLATION                       *
//*-------------------------------------------------------------------*
//JOBLIB  DD  DSN=&DB2SYS..DSN.DSNLOAD,DISP=SHR
//IMAGDB2 EXEC PGM=DSNUTILB,PARM='&DB2SYS,&IMB'
// INCLUDE MEMBER=SYSCOP
//SYSPRINT DD SYSOUT=*
//SYSUDUMP DD SYSOUT=*
//SYSIN    DD  DSN=EE.PARMLIB(SYSC&NN),DISP=SHR
/*
//*-------------------------------------------------------------------*
//*  DELETE SYSIBM.SYSCOPY ENTRIES                                    *
//*  OLDER THAN 9Ø DAYS FROM EXECUTION DATE (PASSED BY IMDB CLIST)    *
//*  &MOD    IS PASSED BY IMDB CLIST                                  *
//*  &RUN    IS PASSED BY IMDB CLIST                                  *
//*-------------------------------------------------------------------*
//MODIFY EXEC DSNUPROC,SYSTEM=&DB2SYS,UID='&MOD',UTPROC='',
//           COND=(1,LT,IMAGDB2)
//DSNUPROC.SYSIN    DD   DSN=EE.PARMLIB(MODI&NN),DISP=SHR
/*
//RUNSTATS EXEC DSNUPROC,SYSTEM=&DB2SYS,UID='&RUN',UTPROC=''
//DSNUPROC.SYSIN    DD   DSN=EE.PARMLIB(RUNS&NN),DISP=SHR
```

This following is the result of the SKELETON prepared and submitted by the IMDB CLIST:

```
//IMDB0025     JOB (40,L0421),'DB2 DB2 DB2',MSGCLASS=Q,MSGLEVEL=(1,1),
//  REGION=3000K,CLASS=B,USER=DB2UT,PASSWORD=DB2TEST,NOTIFY=L041906
//*
//*-------------------------------------------------------------------*
//*  DB2A IS PASSED BY IMDB CLIST                                     *
//*  IMDB0025    IS PASSED BY IMDB CLIST
//*  CHANGE JOBLIB TO REFLECT YOUR INSTALLATION                       *
//*-------------------------------------------------------------------*
//JOBLIB  DD  DSN=DB2A.DSN.DSNLOAD,DISP=SHR
//IMAGDB2 EXEC PGM=DSNUTILB,PARM='DB2A,IMDB0025'
// INCLUDE MEMBER=SYSCOP
//SYSPRINT DD SYSOUT=*
//SYSUDUMP DD SYSOUT=*
//SYSIN    DD  DSN=EE.PARMLIB(SYSC25),DISP=SHR
/*
//*-------------------------------------------------------------------*
//*  DELETE SYSIBM.SYSCOPY ENTRIES                                    *
//*  OLDER THAN 90 DAYS FROM EXECUTION DATE                           *
//*  MOD25    IS PASSED BY IMDB CLIST                                 *
//*  RUN25    IS PASSED BY IMDB CLIST                                 *
//*-------------------------------------------------------------------*
//MODIFY EXEC DSNUPROC,SYSTEM=DB2A,UID='MOD25',UTPROC='',
//           COND=(1,LT,IMAGDB2)
//DSNUPROC.SYSIN    DD   DSN=EE.PARMLIB(MODI25),DISP=SHR
/*
//RUNSTATS EXEC DSNUPROC,SYSTEM=DB2A,UID='RUN25',UTPROC=''
//DSNUPROC.SYSIN    DD   DSN=EE.PARMLIB(RUNS25),DISP=SHR
```

If I set &V=25 in the IMDB CLIST, I run 25 imagecopies in a single job IMDB*xxxx*.

So, the sequence of the jobs submitted will be:

- IMDB0025 (with members SYSCOP, SYSC25, MODI25, and RUNS25).
- IMDB0050 (with members SYSCOP, SYSC50, MODI50, and RUNS50).
- IMDB0075 (with members SYSCOP, SYSC75, MODI75, and RUNS75).
- And so on, until we reach the end of the objects to be saved.

These are the contents of SYSCOP, SYSC25, MODI25, and RUNS25 members.

SYSCOP:

```
//SYSCOP1 DD DSN=ELE.COPIA.DB319501.TS319555.AL040831.T135044,
//  DISP=(,CATLG),SPACE=(TRK,(399,26),RLSE)
//SYSCOP2 DD DSN=ELE.COPIA.DB319501.TS319556.AL040831.T135044,
//  DISP=(,CATLG),SPACE=(TRK,(251,16),RLSE)
//SYSCOP3 DD DSN=ELE.COPIA.DB319501.TS319564.AL040831.T135044,
//  DISP=(,CATLG),SPACE=(TRK,(532,35),RLSE)
//SYSCOP4 DD DSN=ELE.COPIA.DB319501.TS31958D.AL040831.T135044,
//  DISP=(,CATLG),SPACE=(TRK,(40,2),RLSE)
//SYSCOP5 DD DSN=ELE.COPIA.DB319501.TS31958G.AL040831.T135044,
//  DISP=(,CATLG),SPACE=(TRK,(28,1),RLSE)
//SYSCOP6 DD DSN=ELE.COPIA.DB319501.TS31958L.AL040831.T135044,
//  DISP=(,CATLG),SPACE=(TRK,(22,1),RLSE)
//SYSCOP7 DD DSN=ELE.COPIA.DB319501.TS319585.AL040831.T135044,
//  DISP=(,CATLG),SPACE=(TRK,(799,53),RLSE)
//SYSCOP8 DD DSN=ELE.COPIA.DB319501.TS319588.AL040831.T135044,
//  DISP=(,CATLG),SPACE=(TRK,(102,6),RLSE)
//SYSCOP9 DD DSN=ELE.COPIA.DB319501.TS31959B.AL040831.T135044,
//  DISP=(,CATLG),SPACE=(TRK,(28,1),RLSE)
//SYSCOP10 DD DSN=ELE.COPIA.DB319501.TS319590.AL040831.T135044,
//  DISP=(,CATLG),SPACE=(TRK,(26,1),RLSE)
//SYSCOP11 DD DSN=ELE.COPIA.DB319501.TS319591.AL040831.T135044,
//  DISP=(,CATLG),SPACE=(TRK,(1982,132),RLSE)
//SYSCOP12 DD DSN=ELE.COPIA.DB319501.TS319592.AL040831.T135044,
//  DISP=(,CATLG),SPACE=(TRK,(70,4),RLSE)
```

Note: SYSCOP is replaced every time a new IMDB*xxxx* job is submitted, so we see only the last JCL preparation, relative to the last IMDB*xxxx* job.

SYSC25:

```
COPY TABLESPACE DB002001.TS002000 COPYDDN SYSCOP1
COPY TABLESPACE DB002001.TS002002 COPYDDN SYSCOP2
COPY TABLESPACE DB002001.TS002003 COPYDDN SYSCOP3
COPY TABLESPACE DB002001.TS002010 COPYDDN SYSCOP4
COPY TABLESPACE DB002001.TS002012 COPYDDN SYSCOP5
COPY TABLESPACE DB002001.TS002017 COPYDDN SYSCOP6
COPY TABLESPACE DB002001.TS002018 COPYDDN SYSCOP7
COPY TABLESPACE DB002001.TS002019 COPYDDN SYSCOP8
COPY TABLESPACE DB002001.TS002021 COPYDDN SYSCOP9
COPY TABLESPACE DB002001.TS002023 COPYDDN SYSCOP10
COPY TABLESPACE DB002001.TS002025 COPYDDN SYSCOP11
COPY TABLESPACE DB002001.TS002026 COPYDDN SYSCOP12
COPY TABLESPACE DB002001.TS002029 COPYDDN SYSCOP13
```

```
COPY TABLESPACE DB002001.TS002030 COPYDDN SYSCOP14
COPY TABLESPACE DB002001.TS002031 COPYDDN SYSCOP15
COPY TABLESPACE DB002001.TS002032 COPYDDN SYSCOP16
COPY TABLESPACE DB002001.TS002040 COPYDDN SYSCOP17
COPY TABLESPACE DB000402.TS000400 COPYDDN SYSCOP18
COPY TABLESPACE DB000402.TS000401 COPYDDN SYSCOP19
COPY TABLESPACE DB000402.TS000402 COPYDDN SYSCOP20
COPY TABLESPACE DB000402.TS000403 COPYDDN SYSCOP21
COPY TABLESPACE DB000402.TS000404 COPYDDN SYSCOP22
COPY TABLESPACE DB000402.TS000405 COPYDDN SYSCOP23
COPY TABLESPACE DB000402.TS000406 COPYDDN SYSCOP24
COPY TABLESPACE DB000402.TS000407 COPYDDN SYSCOP25
```

MODI25:

```
MODIFY RECOVERY TABLESPACE DB002001.TS002000  DELETE AGE (90)
MODIFY RECOVERY TABLESPACE DB002001.TS002002  DELETE AGE (90)
MODIFY RECOVERY TABLESPACE DB002001.TS002003  DELETE AGE (90)
MODIFY RECOVERY TABLESPACE DB002001.TS002010  DELETE AGE (90)
MODIFY RECOVERY TABLESPACE DB002001.TS002012  DELETE AGE (90)
MODIFY RECOVERY TABLESPACE DB002001.TS002017  DELETE AGE (90)
MODIFY RECOVERY TABLESPACE DB002001.TS002018  DELETE AGE (90)
MODIFY RECOVERY TABLESPACE DB002001.TS002019  DELETE AGE (90)
MODIFY RECOVERY TABLESPACE DB002001.TS002021  DELETE AGE (90)
MODIFY RECOVERY TABLESPACE DB002001.TS002023  DELETE AGE (90)
MODIFY RECOVERY TABLESPACE DB002001.TS002025  DELETE AGE (90)
MODIFY RECOVERY TABLESPACE DB002001.TS002026  DELETE AGE (90)
MODIFY RECOVERY TABLESPACE DB002001.TS002029  DELETE AGE (90)
MODIFY RECOVERY TABLESPACE DB002001.TS002030  DELETE AGE (90)
MODIFY RECOVERY TABLESPACE DB002001.TS002031  DELETE AGE (90)
MODIFY RECOVERY TABLESPACE DB002001.TS002032  DELETE AGE (90)
MODIFY RECOVERY TABLESPACE DB002001.TS002040  DELETE AGE (90)
MODIFY RECOVERY TABLESPACE DB000402.TS000400  DELETE AGE (90)
MODIFY RECOVERY TABLESPACE DB000402.TS000401  DELETE AGE (90)
MODIFY RECOVERY TABLESPACE DB000402.TS000402  DELETE AGE (90)
MODIFY RECOVERY TABLESPACE DB000402.TS000403  DELETE AGE (90)
MODIFY RECOVERY TABLESPACE DB000402.TS000404  DELETE AGE (90)
MODIFY RECOVERY TABLESPACE DB000402.TS000405  DELETE AGE (90)
MODIFY RECOVERY TABLESPACE DB000402.TS000406  DELETE AGE (90)
MODIFY RECOVERY TABLESPACE DB000402.TS000407  DELETE AGE (90)
```

RUNS25:

```
UNSTATS TABLESPACE DB002001.TS002000  INDEX (ALL)
RUNSTATS TABLESPACE DB002001.TS002002  INDEX (ALL)
RUNSTATS TABLESPACE DB002001.TS002003  INDEX (ALL)
RUNSTATS TABLESPACE DB002001.TS002010  INDEX (ALL)
RUNSTATS TABLESPACE DB002001.TS002012  INDEX (ALL)
RUNSTATS TABLESPACE DB002001.TS002017  INDEX (ALL)
RUNSTATS TABLESPACE DB002001.TS002018  INDEX (ALL)
```

```
RUNSTATS TABLESPACE DB002001.TS002019  INDEX (ALL)
RUNSTATS TABLESPACE DB002001.TS002021  INDEX (ALL)
RUNSTATS TABLESPACE DB002001.TS002023  INDEX (ALL)
RUNSTATS TABLESPACE DB002001.TS002025  INDEX (ALL)
RUNSTATS TABLESPACE DB002001.TS002026  INDEX (ALL)
RUNSTATS TABLESPACE DB002001.TS002029  INDEX (ALL)
RUNSTATS TABLESPACE DB002001.TS002030  INDEX (ALL)
RUNSTATS TABLESPACE DB002001.TS002031  INDEX (ALL)
RUNSTATS TABLESPACE DB002001.TS002032  INDEX (ALL)
RUNSTATS TABLESPACE DB002001.TS002040  INDEX (ALL)
RUNSTATS TABLESPACE DB000402.TS000400  INDEX (ALL)
RUNSTATS TABLESPACE DB000402.TS000401  INDEX (ALL)
RUNSTATS TABLESPACE DB000402.TS000402  INDEX (ALL)
RUNSTATS TABLESPACE DB000402.TS000403  INDEX (ALL)
RUNSTATS TABLESPACE DB000402.TS000404  INDEX (ALL)
RUNSTATS TABLESPACE DB000402.TS000405  INDEX (ALL)
RUNSTATS TABLESPACE DB000402.TS000406  INDEX (ALL)
RUNSTATS TABLESPACE DB000402.TS000407  INDEX (ALL)
```

Alberto Mungai
Senior Systems Programmer (Italy)

DB2 utility generation process

For those of us that work in environments where there is no automated method to generate DB2 utilities, I have created the following utility generation process.

I have worked in quite a few small shops that have DB2, but do not have BMC/Platinum utility generation tools. On quite a large number of occasions I have found it necessary to move data from one system to another and wrote this utility to assist me in this endeavour.

This is especially helpful when there are many tables in the database, or multiple databases in the system. Over time, this process has evolved to include most of the utilities I have ever needed. The REXX EXECs that create the utilities can be easily modified for a specific type of run of a utility (for example recovery to current/recovery to point in time). I have included

only the basic parameters here because the different combinations are unlimited.

This procedure begins with a SPUFI being run in DB2 to extract the relevant database information from the catalog. (The OUTLIST dataset should be allocated large enough to hold all the information extracted from the catalog.) The OUTLIST dataset is then used in the utility generation job. The dataset will be passed into a REFORMAT REXX, which removes all comments and unnecessary data. The reformatted OUTPUT file from the first REXX EXEC is then passed into a second REXX, which will use this data to create the utilities in separate datasets.

I would recommend using three separate libraries for the various members in this process – JCL in a CNTL library, SQL in a SPUFI library, and REXX in an EXEC library.

EXECUTION INSTRUCTIONS

Edit the UTILKEYS SPUFI member for the database information to be extracted.

Execute the UTILKEYS member in SPUFI.

Edit the JCL member UTIL to use the SPUFI OUTLIST dataset as input.

Edit the JCL member UTIL for <HLQ>s and <UID>s.

Edit the REFUTILO REXX EXEC member to extract the correct database.

Edit the UTIL REXX EXEC member to make any changes to the generated utilities.

Submit the batch job UTIL.

Verify that the job ran OK.

Each output dataset from STEP004 will contain a number of jobs. The utility JCL in each dataset is broken into 100 EXEC

steps to allow for job submission on smaller systems. Each dataset may need to be broken up into separate jobs. Again, this may depend on system limitations.

I usually create PDS members with each of the jobs in the output datasets; also the jobs can be broken up into utilities for each individual tablespace and index. This is purely a matter of choice with regard to system standards and operating procedures.

UTILKEYS

```
--
-- CHANGE <DBNAME> to database name to create Utilities for
--
SELECT TB.DBNAME, TB.TSNAME, TB.CREATOR, TB.NAME,
       IX.NAME, K.COLNAME, K.COLSEQ, MAX(TP.PARTITION)
  FROM SYSIBM.SYSTABLEPART TP
      ,SYSIBM.SYSTABLESPACE TS
      ,SYSIBM.SYSTABLES TB
      ,SYSIBM.SYSINDEXES IX
      ,SYSIBM.SYSKEYS K
 WHERE TB.DBNAME   = '<DBNAME>'
   AND TB.DBNAME   = TP.DBNAME
   AND TB.DBNAME   = TS.DBNAME
   AND TB.TSNAME   = TP.TSNAME
   AND TB.TSNAME   = TS.NAME
   AND TB.CREATOR  = IX.TBCREATOR
   AND TB.NAME     = IX.TBNAME
   AND K.IXCREATOR = IX.CREATOR
   AND K.IXNAME    = IX.NAME
   AND TS.TYPE     = ' '
 GROUP BY TB.DBNAME, TB.TSNAME, TB.CREATOR, TB.NAME,
       IX.NAME, K.COLNAME, K.COLSEQ
 ORDER BY TB.NAME, IX.NAME, K.COLSEQ;
```

REFUTILO

```
/*  REXX  */
/*  NAME:      REFORMAT                                          */
/*  FUNCTION:  REFORMATS THE INPUT FILE FROM SPUFI               */
/*                                                               */
/*  Change <DBNAME> to the required database name               */
/*---------------------------------------------------------------*/
/*TRACE I */
  PREV_TBN = ""
```

```
  PREV_IXN = ""
  DONE = "NO"
  OUT_DBN = ""
  OUT_TSN = ""
  OUT_TBC = ""
  OUT_TBN = ""
  OUT_IXN = ""
  OUT_TBK1 = ""
  OUT_TBK2 = "                    "
  OUT_TBK3 = "                    "
  OUT_TBK4 = "                    "
  OUT_TBK5 = "                    "
  OUT_TBK6 = "                    "
  OUT_TBK7 = "                    "
  OUT_NMP = ""
PROCESS_FILE:
   DO WHILE DONE="NO"
     "EXECIO 1 DISKR UTILKEYS"
     IF RC <> Ø
     THEN
         DONE = "YES"
     ELSE
       PULL RECORD
       DBN = SUBSTR(RECORD,1,8)
       TSN = SUBSTR(RECORD,11,8)
       TBC = SUBSTR(RECORD,21,8)
       TBN = SUBSTR(RECORD,31,18)
       IXN = SUBSTR(RECORD,51,18)
       TBK = SUBSTR(RECORD,71,18)
       KYN = SUBSTR(RECORD,96,1)
       NMP = SUBSTR(RECORD,1Ø4,1)
       IF DBN = '<DBNAME>'
       THEN
         DO
           IF PREV_TBN = TBN
           THEN
             DO
               IF PREV_IXN = IXN
               THEN
                 DO
                   CALL OUT_TBK
                 END
               ELSE
                 DO
                   CALL WRITE_REC
                   CALL POP_OUT
               END
             END
           ELSE
               IF PREV_TBN = ""
```

```
                THEN
                  DO
                    CALL POP_OUT
                  END
                ELSE
                  DO
                    CALL WRITE_REC
                    CALL POP_OUT
                  END
          END
   END
CALL WRITE_REC
"EXECIO Ø DISKR UTILKEYS (FINIS"
EXIT Ø
OUT_TBK:
   IF KYN = '2' THEN OUT_TBK2 = TBK
   IF KYN = '3' THEN OUT_TBK3 = TBK
   IF KYN = '4' THEN OUT_TBK4 = TBK
   IF KYN = '5' THEN OUT_TBK5 = TBK
   IF KYN = '6' THEN OUT_TBK6 = TBK
   IF KYN = '7' THEN OUT_TBK7 = TBK
RETURN
POP_OUT:
OUT_DBN=DBN
OUT_TSN=TSN
OUT_TBC=TBC
OUT_TBN=TBN
OUT_IXN=IXN
OUT_TBK1=TBK
OUT_TBK2="                    "
OUT_TBK3="                    "
OUT_TBK4="                    "
OUT_TBK5="                    "
OUT_TBK6="                    "
OUT_TBK7="                    "
OUT_NMP=NMP
PREV_TBN = TBN
PREV_IXN = IXN
RETURN
WRITE_REC:
QUEUE  OUT_DBN||OUT_TSN||OUT_TBC||OUT_TBN||OUT_IXN||OUT_TBK1||OUT_TBK2||,
  OUT_TBK3||OUT_TBK4||OUT_TBK5||OUT_TBK6||OUT_TBK7||OUT_NMP
"EXECIO 1 DISKW UTILFILE"
RETURN
```

UTIL

```
/* REXX */
/*********************************************************************/
```

```
/* READ A SEQUENTIAL FILE CONTAINING A LIST OF THE FOLLOWING:        */
/* DATABASE NAME      - 8  CHARACTERS                                */
/* TABLESPACE NAME    - 8  CHARACTERS                                */
/* TABLE CREATOR NAME - 8  CHARACTERS                                */
/* TABLE NAME         - 18 CHARACTERS                                */
/* INDEX NAME         - 18 CHARACTERS                                */
/* PRIMARY KEY COL 1  - 18 CHARACTERS                                */
/* PRIMARY KEY COL 2  - 18 CHARACTERS                                */
/* PRIMARY KEY COL 3  - 18 CHARACTERS                                */
/* PRIMARY KEY COL 4  - 18 CHARACTERS                                */
/* PRIMARY KEY COL 5  - 18 CHARACTERS                                */
/* PRIMARY KEY COL 6  - 18 CHARACTERS                                */
/* PRIMARY KEY COL 7  - 18 CHARACTERS (NO TABLES HAVE MORE THAN 7)   */
/* NUMBER OF PARTS    - 1  CHARACTER                                 */
/*                                                                   */
/* FOR EACH TABLESPACE IN THE INPUT DATASET, A SET OF UTILITIES      */
/* WILL BE CREATED. FOR EACH INDEX A UTILITY WILL BE CREATED         */
/*                                                                   */
/* EDIT THE FOLLOWING:                                               */
/*  <UID> - THIS IS THE USERID FOR THE JOBCARD                       */
/*  <HLQ> - DATASET HIGH LEVEL QUALIFIER                             */
/*  <DSN> - DB2 SUBSYSTEM ID                                         */
/*******************************************************************/
STATUS = MSG('OFF')
/*TRACE I*/
JC1="//<UID>X JOB (ACCT#),'* DB2 UTILITY *',USER=<UID>,"
JC2="//         MSGCLASS=X,CLASS=A,NOTIFY=<UID>,REGION=ØM,TIME=144Ø"
JC3="//*ROUTE PRINT U61"
QUEUE JC1
QUEUE JC2
QUEUE JC3
"EXECIO 3 DISKW UNLDFILE"
QUEUE JC1
QUEUE JC2
QUEUE JC3
"EXECIO 3 DISKW LOADFILE"
QUEUE JC1
QUEUE JC2
QUEUE JC3
"EXECIO 3 DISKW REPRFILE"
QUEUE JC1
QUEUE JC2
QUEUE JC3
QUEUE "//*"
QUEUE "//STEPØØ1 EXEC PGM=IDCAMS"
QUEUE "//SYSPRINT  DD SYSOUT=*"
QUEUE "//SYSIN     DD *"
"EXECIO 7 DISKW DELTFILE"
QUEUE JC1
QUEUE JC2
```

```
QUEUE JC3
"EXECIO 3 DISKW RUNSFILE"
QUEUE JC1
QUEUE JC2
QUEUE JC3
"EXECIO 3 DISKW TCPYFILE"
QUEUE JC1
QUEUE JC2
QUEUE JC3
"EXECIO 3 DISKW ICPYFILE"
QUEUE JC1
QUEUE JC2
QUEUE JC3
"EXECIO 3 DISKW RBLDFILE"
QUEUE JC1
QUEUE JC2
QUEUE JC3
"EXECIO 3 DISKW RCVIFILE"
QUEUE JC1
QUEUE JC2
QUEUE JC3
"EXECIO 3 DISKW RCVTFILE"
QUEUE JC1
QUEUE JC2
QUEUE JC3
"EXECIO 3 DISKW REOGFILE"
TSTEPN = Ø
ISTEPN = Ø
PREV_TBN = "                  "
PREV_IXN = "                  "
PROCESS_FILE:
   "EXECIO 1 DISKR UTILFILE"
   IF RC <> Ø
   THEN
       SIGNAL END_PROCESS
   CALL GET_REC
   IF PREV_TBN = "                  "
   THEN
     DO
       CALL BLANK_TBN
       SIGNAL PROCESS_FILE
     END
   IF PREV_TBN = TBN
   THEN
     DO
       CALL SAME_TBN
       SIGNAL PROCESS_FILE
     END
   CALL DIFF_TBN
   SIGNAL PROCESS_FILE
```

```
GET_REC:
 PULL RECORD
 DBN    = SUBSTR(RECORD,1,8)
 TSN    = SUBSTR(RECORD,9,8)
 TBC    = SUBSTR(RECORD,17,8)
 TBN    = SUBSTR(RECORD,25,18)
 IXN    = SUBSTR(RECORD,43,18)
 TBK1   = SUBSTR(RECORD,61,18)
 TBK2   = SUBSTR(RECORD,79,18)
 TBK3   = SUBSTR(RECORD,97,18)
 TBK4   = SUBSTR(RECORD,115,18)
 TBK5   = SUBSTR(RECORD,133,18)
 TBK6   = SUBSTR(RECORD,151,18)
 TBK7   = SUBSTR(RECORD,169,18)
 NMP    = SUBSTR(RECORD,187,1)
RETURN
BLANK_TBN:
 PREV_TBN = TBN
 PREV_IXN = IXN
 CALL TSTEP_NUM
 CALL ISTEP_NUM
 CALL DELT_FILE
 CALL UNLD_FILE
 CALL LOAD_FILE
 CALL REPR_FILE
 CALL RUNS_FILE
 CALL RCVT_FILE
 CALL REOG_FILE
 CALL TCPY_FILE
 CALL RCVI_FILE
 CALL ICPY_FILE
 CALL RBLD_FILE
RETURN
SAME_TBN:
 CALL ISTEP_NUM
 CALL RCVI_FILE
 CALL ICPY_FILE
 CALL RBLD_FILE
RETURN
DIFF_TBN:
 PREV_TBN = TBN
 PREV_IXN = IXN
 CALL TSTEP_NUM
 CALL ISTEP_NUM
 CALL DELT_FILE
 CALL UNLD_FILE
 CALL LOAD_FILE
 CALL REPR_FILE
 CALL RUNS_FILE
 CALL RCVT_FILE
```

```
 CALL REOG_FILE
 CALL TCPY_FILE
 CALL RCVI_FILE
 CALL ICPY_FILE
 CALL RBLD_FILE
RETURN
TSTEP_NUM:
TSTEPN = TSTEPN + 1
IF TSTEPN < 10
THEN
    TSTEPNUM="STEP00"||TSTEPN
ELSE
    IF TSTEPN < 100
    THEN
        TSTEPNUM = "STEP0"||TSTEPN
    ELSE
        IF TSTEPN < 1000
        THEN
            TSTEPNUM="STEP"||TSTEPN
        ELSE
            TSTEPNUM="STP"||TSTEPN
RETURN
ISTEP_NUM:
ISTEPN = ISTEPN + 1
IF ISTEPN < 10
THEN
    ISTEPNUM="STEP00"||ISTEPN
ELSE
    IF ISTEPN < 100
    THEN
        ISTEPNUM = "STEP0"||ISTEPN
    ELSE
        IF ISTEPN < 1000
        THEN
            ISTEPNUM="STEP"||ISTEPN
        ELSE
            ISTEPNUM="STP"||ISTEPN
RETURN
UNLD_FILE:
IF SUBSTR(TSTEPNUM,6,2) = "00"
THEN
  DO
    QUEUE JC1
    QUEUE JC2
    QUEUE JC3
   "EXECIO 3 DISKW UNLDFILE"
  END
QUEUE "//*"
"EXECIO 1 DISKW UNLDFILE"
QUEUE  "//*****************************************************************"
```

```
"EXECIO 1 DISKW UNLDFILE"
QUEUE "//*         UNLOAD TABLE "||STRIP(TBC,T,' ')||"."||TBN
"EXECIO 1 DISKW UNLDFILE"
QUEUE  "//*************************************************************"
"EXECIO 1 DISKW UNLDFILE"
QUEUE "//*"
"EXECIO 1 DISKW UNLDFILE"
QUEUE "//"TSTEPNUM||" EXEC PGM=IKJEFT01,REGION=4096K,DYNAMNBR=20"
"EXECIO 1 DISKW UNLDFILE"
QUEUE "//STEPLIB   DD DISP=SHR,DSN=DSN710.SDSNLOAD"
"EXECIO 1 DISKW UNLDFILE"
QUEUE "//          DD DISP=SHR,DSN=DSN710.SDSNEXIT"
"EXECIO 1 DISKW UNLDFILE"
QUEUE "//SYSTSPRT  DD SYSOUT=*"
"EXECIO 1 DISKW UNLDFILE"
QUEUE "//SYSPRINT  DD SYSOUT=*"
"EXECIO 1 DISKW UNLDFILE"
QUEUE "//SYSOUT    DD SYSOUT=*"
"EXECIO 1 DISKW UNLDFILE"
QUEUE "//SYSUDUMP  DD SYSOUT=*"
"EXECIO 1 DISKW UNLDFILE"
QUEUE "//SYSREC00  DD UNIT=3390,SPACE=(CYL,(5,10),RLSE),"
"EXECIO 1 DISKW UNLDFILE"
QUEUE "//             VOL=SER=TSOPK1,DISP=(,CATLG,DELETE),"
"EXECIO 1 DISKW UNLDFILE"
QUEUE "//             DSN=<UID>."||STRIP(DBN,T,' ')||"."||,
STRIP(TSN,T,' ')||".UNLD"
"EXECIO 1 DISKW UNLDFILE"
QUEUE "//SYSPUNCH  DD DISP=SHR,DSN=<UID>."||STRIP(DBN,T,' ')||,
".LOAD.CNTL("||STRIP(TSN,T,' ')||")"
"EXECIO 1 DISKW UNLDFILE"
QUEUE "//SYSTSIN   DD *"
"EXECIO 1 DISKW UNLDFILE"
QUEUE "  DSN SYSTEM(<DSN>) "
"EXECIO 1 DISKW UNLDFILE"
QUEUE "      RUN PROGRAM(DSNTIAUL) PLAN(DSNTIB71) -"
"EXECIO 1 DISKW UNLDFILE"
QUEUE "      LIB('DSN710.RUNLIB.LOAD') - "
"EXECIO 1 DISKW UNLDFILE"
QUEUE "      PARMS('SQL') "
"EXECIO 1 DISKW UNLDFILE"
QUEUE "/*"
"EXECIO 1 DISKW UNLDFILE"
QUEUE "//SYSIN     DD * "
"EXECIO 1 DISKW UNLDFILE"
QUEUE "  SELECT * "
"EXECIO 1 DISKW UNLDFILE"
QUEUE "    FROM "||STRIP(TBC,T,' ')||"."||TBN
"EXECIO 1 DISKW UNLDFILE"
QUEUE "   ORDER BY "||TBK1
```

```
"EXECIO 1 DISKW UNLDFILE"
IF TBK2 <> ""
THEN
  DO
    QUEUE "           ,"||TBK2
    "EXECIO 1 DISKW UNLDFILE"
  END
IF TBK3 <> ""
THEN
  DO
    QUEUE "           ,"||TBK3
    "EXECIO 1 DISKW UNLDFILE"
  END
IF TBK4 <> ""
THEN
  DO
    QUEUE "           ,"||TBK4
    "EXECIO 1 DISKW UNLDFILE"
  END
IF TBK5 <> ""
THEN
  DO
    QUEUE "           ,"||TBK5
    "EXECIO 1 DISKW UNLDFILE"
  END
IF TBK6 <> ""
THEN
  DO
    QUEUE "           ,"||TBK6
    "EXECIO 1 DISKW UNLDFILE"
  END
IF TBK7 <> ""
THEN
  DO
    QUEUE "           ,"||TBK7
    "EXECIO 1 DISKW UNLDFILE"
  END
QUEUE "  ;"
QUEUE "/*"
"EXECIO 2 DISKW UNLDFILE"
RETURN
LOAD_FILE:
IF SUBSTR(TSTEPNUM,6,2) = "ØØ"
THEN
  DO
    QUEUE JC1
    QUEUE JC2
    QUEUE JC3
    "EXECIO 3 DISKW LOADFILE"
  END
```

```
QUEUE "//*"
"EXECIO 1 DISKW LOADFILE"
QUEUE  "//*************************************************************"
"EXECIO 1 DISKW LOADFILE"
QUEUE "//*          LOAD TABLE "||STRIP(TBC,T,' ')||"."||TBN
"EXECIO 1 DISKW LOADFILE"
QUEUE  "//*************************************************************"
"EXECIO 1 DISKW LOADFILE"
QUEUE "//*"
"EXECIO 1 DISKW LOADFILE"
QUEUE "//"||TSTEPNUM||" EXEC PGM=DSNUTILB,REGION=4Ø96K,PARM='<DSN>'"
"EXECIO 1 DISKW LOADFILE"
QUEUE "//STEPLIB   DD DISP=SHR,DSN=DSN71Ø.SDSNLOAD"
"EXECIO 1 DISKW LOADFILE"
QUEUE "//          DD DISP=SHR,DSN=DSN71Ø.SDSNEXIT"
"EXECIO 1 DISKW LOADFILE"
QUEUE "//SYSPRINT  DD SYSOUT=*"
"EXECIO 1 DISKW LOADFILE"
QUEUE "//UTPRINT   DD SYSOUT=*"
"EXECIO 1 DISKW LOADFILE"
QUEUE "//SYSUDUMP  DD SYSOUT=*"
"EXECIO 1 DISKW LOADFILE"
QUEUE "//SYSREC ØØ  DD DISP=SHR,DSN=<UID>."||STRIP(DBN,T,' ')||"."||,
STRIP(TSN,T,' ')".UNLD"
"EXECIO 1 DISKW LOADFILE"
QUEUE "//SYSUT1    DD DISP=(,DELETE),UNIT=SYSDA,SPACE=(CYL,(5,5),RLSE)"
"EXECIO 1 DISKW LOADFILE"
QUEUE "//SORTLIB   DD DISP=SHR,DSN=SYS1.SORTLIB"
"EXECIO 1 DISKW LOADFILE"
QUEUE "//SORTOUT   DD DISP=(,DELETE),UNIT=SYSDA,SPACE=(CYL,(5,5),RLSE)"
"EXECIO 1 DISKW LOADFILE"
QUEUE "//SORTWKØ1  DD DISP=(,DELETE),UNIT=SYSDA,SPACE=(CYL,(5,5),RLSE)"
"EXECIO 1 DISKW LOADFILE"
QUEUE "//SORTWKØ2  DD DISP=(,DELETE),UNIT=SYSDA,SPACE=(CYL,(5,5),RLSE)"
"EXECIO 1 DISKW LOADFILE"
QUEUE "//SORTWKØ3  DD DISP=(,DELETE),UNIT=SYSDA,SPACE=(CYL,(5,5),RLSE)"
"EXECIO 1 DISKW LOADFILE"
QUEUE "//SORTWKØ4  DD DISP=(,DELETE),UNIT=SYSDA,SPACE=(CYL,(5,5),RLSE)"
"EXECIO 1 DISKW LOADFILE"
QUEUE "//SYSIN     DD DISP=SHR,DSN=<UID>."||STRIP(DBN,T,' ')||,
".LOAD.CNTL("||STRIP(TSN,T,' ')||")"
"EXECIO 1 DISKW LOADFILE"
QUEUE "//*"
"EXECIO 1 DISKW LOADFILE"
QUEUE  "//*************************************************************"
"EXECIO 1 DISKW LOADFILE"
QUEUE "//*        REPAIR TABLE "||STRIP(TBC,T,' ')||"."||TBN
"EXECIO 1 DISKW LOADFILE"
QUEUE  "//*************************************************************"
"EXECIO 1 DISKW LOADFILE"
```

```
QUEUE "//*"
"EXECIO 1 DISKW LOADFILE"
QUEUE "//"||TSTEPNUM||"A EXEC PGM=DSNUTILB,PARM='<DSN>'"
"EXECIO 1 DISKW LOADFILE"
QUEUE "//STEPLIB   DD DISP=SHR,DSN=DSN71Ø.SDSNLOAD"
"EXECIO 1 DISKW LOADFILE"
QUEUE "//          DD DISP=SHR,DSN=DSN71Ø.SDSNEXIT"
"EXECIO 1 DISKW LOADFILE"
QUEUE "//SYSPRINT  DD SYSOUT=*"
"EXECIO 1 DISKW LOADFILE"
QUEUE "//UTPRINT   DD SYSOUT=*"
"EXECIO 1 DISKW LOADFILE"
QUEUE "//SYSUDUMP  DD SYSOUT=*"
"EXECIO 1 DISKW LOADFILE"
QUEUE "//SYSIN     DD *"
"EXECIO 1 DISKW LOADFILE"
IF NMP = 'Ø'
THEN
  DO
    QUEUE "  REPAIR OBJECT LOG NO"
    QUEUE "          SET TABLESPACE "||STRIP(DBN,T,' ')||"."||TSN
    QUEUE "              NOCOPYPEND"
    "EXECIO 3 DISKW LOADFILE"
  END
ELSE
  DO
    NMPI = NMP
    DO WHILE NMPI > Ø
      QUEUE "  REPAIR OBJECT LOG NO"
      QUEUE "      SET TABLESPACE "||STRIP(DBN,T,' ')||"."||TSN
      QUEUE "          NOCOPYPEND PART " NMPI
      "EXECIO 3 DISKW LOADFILE"
      NMPI = NMPI - 1
    END
  END
QUEUE "/*"
"EXECIO 1 DISKW LOADFILE"
RETURN
REPR_FILE:
IF SUBSTR(TSTEPNUM,6,2) = "ØØ"
THEN
  DO
    QUEUE JC1
    QUEUE JC2
    QUEUE JC3
    "EXECIO 3 DISKW REPRFILE"
  END
QUEUE "//*"
"EXECIO 1 DISKW REPRFILE"
QUEUE  "//*************************************************************"
```

```
"EXECIO 1 DISKW REPRFILE"
QUEUE "//*        REPAIR TABLE "||STRIP(TBC,T,' ')||"."||TBN
"EXECIO 1 DISKW REPRFILE"
QUEUE  "//*************************************************************"
"EXECIO 1 DISKW REPRFILE"
QUEUE "//*"
"EXECIO 1 DISKW REPRFILE"
QUEUE "//"||TSTEPNUM||"A EXEC PGM=DSNUTILB,REGION=4Ø96K,PARM='<DSN>'"
"EXECIO 1 DISKW REPRFILE"
QUEUE "//STEPLIB   DD DISP=SHR,DSN=DSN71Ø.SDSNLOAD"
"EXECIO 1 DISKW REPRFILE"
QUEUE "//          DD DISP=SHR,DSN=DSN71Ø.SDSNEXIT"
"EXECIO 1 DISKW REPRFILE"
QUEUE "//SYSPRINT  DD SYSOUT=*"
"EXECIO 1 DISKW REPRFILE"
QUEUE "//UTPRINT   DD SYSOUT=*"
"EXECIO 1 DISKW REPRFILE"
QUEUE "//SYSUDUMP  DD SYSOUT=*"
"EXECIO 1 DISKW REPRFILE"
QUEUE "//SYSIN     DD *"
"EXECIO 1 DISKW REPRFILE"
IF NMP = 'Ø'
THEN
  DO
    QUEUE "  REPAIR OBJECT LOG NO"
    QUEUE "          SET TABLESPACE "||STRIP(DBN,T,' ')||"."||TSN
    QUEUE "              NOCOPYPEND"
    "EXECIO 3 DISKW REPRFILE"
  END
ELSE
  DO
    NMPI = NMP
    DO WHILE NMPI > Ø
      QUEUE "  REPAIR OBJECT LOG NO"
      QUEUE "     SET TABLESPACE "||STRIP(DBN,T,' ')||"."||TSN
      QUEUE "         NOCOPYPEND PART " NMPI
      "EXECIO 3 DISKW REPRFILE"
      NMPI = NMPI - 1
    END
  END
QUEUE "/*"
"EXECIO 1 DISKW REPRFILE"
RETURN
DELT_FILE:
QUEUE "  DELETE '<UID>."||STRIP(DBN,T,' ')||"."||STRIP(TSN,T,' ')||,
".UNLD'"
"EXECIO 1 DISKW DELTFILE"
RETURN
RUNS_FILE:
IF SUBSTR(TSTEPNUM,6,2) = "ØØ"
```

```
THEN
  DO
    QUEUE JC1
    QUEUE JC2
    QUEUE JC3
    "EXECIO 3 DISKW RUNSFILE"
  END
```

Editor's note: this article will be concluded in the next issue.

Tim Foster
DBA
Timian Systems (USA)

IPLocks has announced Version 4.2 of IPLocks, which is designed to further protect business-critical information assets from threat, fraud, or fraudulent abuse. Version 4.2 now supports DB2 on a mainframe.

The latest version offers an automated approach to monitoring and understanding user behaviour patterns associated with mainframe DB2.

Other enhancements include procedural language capabilities enabling users to create more sophisticated policies for security and compliance using PL/SQL and Transact-SQL scripts, to further refine information about usage, access patterns, and data integrity checking.

For further information contact:
IPLocks, 441-A West Trimble, San Jose, CA 95131, USA.
Tel: (408) 383 7513.
URL: www.iplocks.com/product.asp.

* * *

Arcplan has announced DB2 Cube Views connector for its dynaSight product. Support for DB2 Cube Views enables arcplan customers and partners to leverage IBM's latest generation of enhanced OLAP capabilities in DB2 Universal Database and better manage and deploy multidimensional data across the enterprise.

Arcplan's analytic development software, dynaSight 4.0, provides a business intelligence platform to create custom analytic applications for users at any level and across all enterprise departments. Arcplan's Cube Views query interface can directly tap into the OLAP BI features and content of DB2 Cube Views, allowing implementation of the Cube Views features, such as complex hierarchies, member attributes, and calculated measures.

For further information contact:
Arcplan, 12830 Hillcrest Road, Suite 111, Dallas, TX 75230, USA.
Tel: (972) 364 9007.
URL: www.dynasight.com.

* * *

HiT Software has announced new releases of its ODBC and OLE DB middleware products for DB2 databases running on the mid-range operating systems i5/OS and OS/400 (iSeries), DB2 Version 8 on the mainframe, as well as DB2 8.2 on AIX, Linux, Sun, HP-UX, and Windows.

Major features in these releases are support for the latest versions of the DB2 databases, updated security cipher suites, and performance enhancements for INSERT, UPDATE, and DELETE statements, as well as increased performance for standard SELECT statements.

The products are available for download from www.hitsw.com/products_services/downloads.html.

* * *

Software AG has unveiled software that allows mainframe customers to replicate Adabas information to DB2.

The software automatically publishes data changes on-the-fly to the target application, regardless of what language the target application is written in or where it is located.

For further information contact:
Software AG, 11190 Sunrise Valley Drive, Reston, VA 20191, USA.
Tel: (703) 860 5050.
URL: www1.softwareag.com/Corporate/products/adabas/default.asp.