August 2003

In this issue

3 The DB2 V8 Health Monitor/Health Center
10 Consolidate messages in DB2 master log
18 How to verify DB2 UDB back-ups
22 CAF interface with caller in amode 24 or 31 and more – part 2
24 DB2 catalog/directory sizing
49 DB2 news

© Xephon plc 2003
The DB2 V8 Health Monitor/Health Center

What is the DB2 UDB V8.1 Health Monitor/Center? Well, it replaces the DB2 UDB V7 Performance Monitor function and is used to report abnormalities in the ‘health’ of your system. What do I mean by ‘health’? I will define that later on.

The Health Monitor/Health Center come bundled in with DB2 UDB 8.1, you do not have to install anything extra.

The following discussion on Health Monitor/Health Center functionality is based on a Windows 2000 system using the db2admin userid (install userid) running DB2 UDB 8.1 FP1.

The monitoring part is performed by the Health Monitor component, and the GUI visualization of the information returned by the Health Monitor is performed by the Health Center component. You can run the Health Monitor on all Linux, Unix, and Windows systems that you can run DB2 UDB 8.1 on. You can run the Health Center on Windows-based systems.

Let’s go though some terms that we will use throughout this article.

The Health Monitor checks system pre-defined characteristics called ‘Database Object Health Indicators’ or Indicators for short. These Indicators are listed later on. When I talked about ‘health’ above, I meant the state of these Indicators, and whether the values of these Indicators were within certain limits. The first of these limits is called the warning limit and the second limit is called the alarm limit. When an Indicator value reaches either of the warning or alarm values, then the Indicator is said to have tripped that alert. Each Indicator has an associated shortname, which is used in commands.

As mentioned earlier, the Health Monitor component is installed when you install DB2 UDB V8.1. You can check whether the Health Monitor component is active, by issuing the CLP command:

```
>db2 get dbm cfg | find /i "HEALTH_MON"
Monitor health of instance and databases (HEALTH_MON) = ON
```
You can turn off the Health Monitor by setting the DBM configuration parameter HEALTH_MON to OFF

>db2 update dbm cfg using HEALTH_MON OFF

You can access the Health Monitor component from either the start program (Start/Programs/IBM DB2/Monitoring Tools/Health Center) or by typing the command db2hc from the Command Line Processor (CLP).

The Health Monitor comes with 17 pre-defined Indicators that are monitored. You cannot add to this list. These Indicators are listed in alphabetical order in Figure 1 and the ‘W’ and ‘A’ columns show the default Warning and Alarm values.

<table>
<thead>
<tr>
<th>Indicator:</th>
<th>W</th>
<th>A</th>
<th>&lt;shortname&gt;</th>
</tr>
</thead>
<tbody>
<tr>
<td>Application control heap utilization</td>
<td>85</td>
<td>95</td>
<td>db.applctl_heap_util</td>
</tr>
<tr>
<td>Application heap utilization</td>
<td>85</td>
<td>95</td>
<td>db.appl_heap_util</td>
</tr>
<tr>
<td>Catalog cache hit ratio</td>
<td>80</td>
<td>70</td>
<td>db.catcache_hitratio</td>
</tr>
<tr>
<td>Database heap utilization</td>
<td>85</td>
<td>95</td>
<td>db.db_heap_util</td>
</tr>
<tr>
<td>Database operational state</td>
<td></td>
<td></td>
<td>db.db_op_status</td>
</tr>
<tr>
<td>Deadlock rate</td>
<td>5</td>
<td>10</td>
<td>db.deadlock_rate</td>
</tr>
<tr>
<td>Lock escalation rate</td>
<td>5</td>
<td>10</td>
<td>db.lock_escal_rate</td>
</tr>
<tr>
<td>Lock list utilization</td>
<td>75</td>
<td>85</td>
<td>db.locklist_util</td>
</tr>
<tr>
<td>Log filesystem utilization</td>
<td>75</td>
<td>85</td>
<td>db.log_fs_util</td>
</tr>
<tr>
<td>Log utilization</td>
<td>75</td>
<td>85</td>
<td>db.log_util</td>
</tr>
<tr>
<td>Long term shard sort memory utilization</td>
<td>60</td>
<td>30</td>
<td>db.max_sort_shrmem_util</td>
</tr>
<tr>
<td>Package cache hit ratio</td>
<td>80</td>
<td>70</td>
<td>db.pkgcache_hitratio</td>
</tr>
<tr>
<td>Percentage of applications waiting on locks</td>
<td>50</td>
<td>70</td>
<td>db.apps_waiting_locks</td>
</tr>
<tr>
<td>Percentage of sorts that overflow</td>
<td>30</td>
<td>50</td>
<td>db.spilled_sorts</td>
</tr>
<tr>
<td>Shared sort memory utilization</td>
<td>75</td>
<td>85</td>
<td>db.sort_shrmem_util</td>
</tr>
<tr>
<td>Shared workspace hit ratio</td>
<td>80</td>
<td>70</td>
<td>db.shrworkspace_hitratio</td>
</tr>
<tr>
<td>Utility heap utilization</td>
<td>85</td>
<td>95</td>
<td>db.utility_heap_util</td>
</tr>
</tbody>
</table>

Figure 1: Indicators

There are two ways of listing these Indicators and their corresponding warning and alarm values—via the Health Monitor screen or from the CLP. From the Health Center main screen (>db2hc), right-click on the <db-alias>, then select Configure and Database Object Health Indicator Settings. This brings up the Configure Database Object Health Indicator settings.
screen, which shows the Indicators and the respective values.

From the CLP issue:

>db2 get alert cfg for database on <db-alias>

As an example, if you issue the CLP command for the SAMPLE database, then the first few lines of the output will look like:

>db2 get alert cfg for database on sample

Alert Configuration

| Indicator Name                     | = db.locklist_util         |
| Type                               | = Threshold-based          |
| Warning                            | = 75                       |
| Alarm                              | = 85                       |
| Sensitivity                        | = 0                        |
| Formula                            | = (db.lock_list_in_use/(locklist*4096))*100; |
| Actions                            | = Disabled                 |
| Threshold or State checking        | = Enabled                  |

| Indicator Name                     | = db.db_op_status          |
| Type                               | = State-based              |
| Sensitivity                        | = 0                        |
| Formula                            | = db.db_status;            |
| Actions                            | = Disabled                 |
| Threshold or State checking        | = Enabled                  |

You can see, from the output above, the warning and alarm value for each Indicator. You can change either of these values from the Health Center screen or from the CLP. From the Health Center screen, click on the warning or alarm value that you want to change. A spin wheel allows you then to change the value. The CLP command is:

>db2 UPDATE ALERT CFG FOR DATABASE ON SAMPLE USING db.apps_waiting_locks SET ALARM 70, WARNING 50, SENSITIVITY 0, THRESHOLDSCHECKED NO, ACTIONSENABLED NO

As you can see, all the CLP commands use the Indicator shortnames rather than the full Indicator name. You can use the Indicator shortname to get a description of that Indicator from the CLP:

>db2 get description for health indicator <shortname>

For example:
So what happens when the Health Monitor detects that the warning/alarm value for an Indicator has been tripped? It really depends on your DB2 instance settings. ‘Out of the box’ settings mean that only one thing will happen – an entry will be written in the notification log.

So, if you want more, you have to set certain values.

If you want to get e-mail or pager notifications, then you need to set up an SMTP server. This is fully documented in the Administration manual, but I have not tested it.

If you want a ‘health beacon’ to appear in the Control Center when the Health Center is not open, you need to go to Health Center/Tools/Tools settings/Health center status beacon and tick the box Notify through status line. If you want to be notified via a pop-up message, then you also need to tick the Notify through pop-up message box. Then, if an alert is tripped and the Health Center is not open, but the Control Center is open, then a ‘health beacon’ appears at the bottom on the Control Center – in the status line. This ‘health beacon’ takes the form of an exclamation mark. Click on this to open the Health Center, or click on Tools/Health Center to open the Health Center.

If you want a script or task to be executed, then you need to update the Indicator parameters, either from the Health Center screen or from the CLP. I could not find a command that would allow me to list which scripts/tasks will be executed for which alert. The only way I found of doing it was to look at each Indicator through the Health Center screen. You can schedule a script/task for when a warning or alarm is generated, but not both (for the same Indicator).

If you are using the Health Center to get a GUI visualization of the state of the Indicators, then there are a couple of points to remember. The Health Center has something called a ‘refresh value’, which means that the Health Center screen is updated only at these intervals, irrespective of when an alert was tripped in the Health Monitor. The possible settings for this refresh value...
are: No automatic refresh, 1 minute (default), 5 minutes, 10 minutes, 30 minutes, 1 hour, 2 hours, and 4 hours. There is also a refresh now button.

As mentioned earlier, DB2 keeps a record of Indicator settings that have been tripped in the Health Center journal. You can examine this log by right clicking on the <instance name> and selecting Show Notification Log. The log is cleared down every time you stop/start the instance. The journal also contains information about DB2 Task history, DB2 Database history, and DB2 Messages issued. Note that the notification log does not allow you to see recommendations (this is covered later on).

You can use the GET HEALTH SNAPSHOT command to retrieve health information about your system. The values returned are those at the point in time when the command was issued. To issue the command for the database manager, use:

>db2 get health snapshot for dbm

A useful addition to the above command is the SHOW DETAIL parameter. This gives you the last ten values as well as an additional information line.

To get health information for a single database issue:

>db2 get health snapshot for database on <db-alias>

To get health information for all active databases in an instance issue:

>db2 get health snapshot for all databases.

So when an alert is tripped, what should I do? This is where the recommendations component of the Health Center comes into play. These recommendations are available from two places. The first place is from the Alerts pane of the Health Center main screen, the second place is from the CLP. When an alert is tripped, and you have the Health Center screen open, an Alert pane is displayed. If you double-click on the Indicator that has been tripped, you will see the relevant recommendations. However, if the alert is cleared, then the Alert pane is also cleared, so how can you get recommendations then? You would
use the `get recommendations` command. The command issued from the CLP is:

```bash
>db2 get recommendations for health indicator <shortname>
```

For example:

```bash
>db2 get recommendations for health indicator db.locklist_util
```

In addition to the database Indicators, DB2 also makes it possible to monitor Storage Management thresholds. These thresholds monitor the space usage of table spaces, table cluster ratios, and table data skews (only applicable for tables that are partitioned). As with the database Indicators you can set warning and alarm values for these three Storage Management thresholds. The space usage for table spaces is a figure between 0 and 100% (for DMS table spaces only), and for cluster ratio you also get a value between 0 and 100%.

Strangely, Storage Management thresholds are not managed from the Health Center but from the Control Center. You can access the Storage Management screens in two ways, firstly from the main Control Center screen where you can right-click on your database and select `Manage Storage`, or secondly from the main Control Center screen you can go to the table space that you want managed, right-click on it and select `Manage Storage`. Whichever way you go, you end up at the Storage Management Setup Launchpad.

Before setting any Storage Management thresholds, you need to create some management tables that DB2 will use when monitoring your storage. You do this by pressing the `Specify Snapshot Storage` button. Once you have created these management tables for your database, this option is greyed out the next time you go to the Storage Management launchpad. Once you have created the management tables, you can start specifying the threshold settings. You do this using the `Specify Threshold Settings` button. Once you have decided on your threshold values, you need to schedule a task to gather the information required to make the comparisons between the actual values and the threshold settings. You set up the scheduling
using the **Specify Snapshot Schedule** button. If you specify that you want the information collected, say, hourly (depending on the insert activity of your system), when a Storage Management threshold is tripped, the system will also try to give you an estimate of when the container for the table space, for instance, will reach 100%. This is a very neat feature, because it gives you a first indication of how urgently you need to fix the problem. This value, of course, is a simple calculation based on the history values obtained, and therefore depends on the frequency of collecting this data, assuming a ‘normal’ (or usual) mode of inserting data.

The default warning and alarm values for table space usage are 75% and 85%, and for cluster ratio 20% and 10% respectively. Yes, these are the right way around; we want a warning when the cluster ratio falls below 20% and an alarm when it falls below 10%, whereas for table space usage we want a warning at 75% and an alarm at 85%. These values may be too high for you, it really depends on the usage of your table spaces, but I think they are a good starting point.

The **get recommendations** command doesn’t apply to Storage Management thresholds.

You can turn off Storage Management monitoring by going to the **Specify Threshold Settings** screen and unticking the **Enable threshold detection for disk space usage** box.

For the database Indicators, I didn’t get the e-mail notification or the script/task execution on alert trip to work, but I think this is because of my system set-up. The e-mail notification requires an SMTP server, which your site may or may not have access to.

In general I think that the Health Monitor/Health Center is a big improvement on the Performance Monitor of V7, and it is free! It does provide an early warning system of something going wrong with your system. Although it requires a certain amount of effort in setting up the appropriate warning and alarm values, I think it is worthwhile; so give it a go and see what you think.

---

*C Leonard*

*Freelance Consultant (UK)*

© Xephon 2003
Consolidate messages in DB2 master log

This REXX program finds the DB2 messages in the DB2 master address space log. These messages are consolidated for each hour period and printed. Only the first message text line is assumed for the message code and this line is printed for the message description. In the final report, all these DB2 messages are summarized. If you want to use this utility, you must change DBP1 to your DB2 member name and change ‘SYSPDBA’ high-level qualifier to your HLQ.

DB2MSGA

/* rexx                                                             */
/* *********************************************************************/
/* main                                                             */
/* This program counts reports DB2 messages in the DB2 master log. */
/* *********************************************************************/
arg dbmid
$free  file(db2log)$
$alloc fi(db2log) da(syspdba.psO.$dbmid$.log.temp) shr$
eof = 'Ø'
i = Ø
j = Ø
k = Ø
tt = Ø
mf = 'Ø'
mcl = Ø
fd = '1'
call w_print_header
call w_init_array
$execio 1 diskr db2log$
do while eof = 'Ø'
pull line
wmsg = word(line,3)
wday = word(line,4)
whour = word(line,1)
wdes = substr(line,29,5Ø)
if wday = 'MONDAY,' then wd = '1'
if wday = 'TUESDAY,' then wd = '1'
if wday = 'WEDNESDAY,' then wd = '1'
if wday = 'THURSDAY,' then wd = '1'
if wday = 'FRIDAY,' then wd = '1'
if wday = 'SATURDAY,' then wd = '1'
if wday = 'SUNDAY,' then wd = '1'
if wd = '1' & fd = 'Ø' then call w_print_dtl
if wd = '1' then wdate = substr(line,35,12)
if wd = '1' & fd = '1' then fd = 'Ø'
if wd = '1' then wd = 'Ø'
if substr(wmsg,1,3) = 'DSN',
   & substr(wmsg,8,1) = 'I' then
   do
      wh = Ø
      wh = substr(whour,1,2)
      wh = wh + 1
      mf = 'Ø'
      do i = 1 to h.wh
         if mc.wh.i = wmsg then
            do
               mn.wh.i = mn.wh.i + 1
               mf = '1'
            end
         end
      if mf = 'Ø' then
         do
            h.wh = h.wh + 1
            k = h.wh
            mc.wh.k = wmsg
            mn.wh.k = 1
            md.wh.k = wdes
         end
   end
$execio 1 diskr db2log$
if word(line,2) = '//STARTING' then
   eof = '1'
if rc > Ø then eof = '1'
end
   call w_print_dtl
   call w_print_total
   $free file(db2log)$
exit
w_print_header :
say '      MEMBER FOR ' dbmid 'MESSAGE OUTPUT '
say ' D A T E ' 'HOUR ' 'MSG. CODE' 'COUNT' 'MESSAGE DESC.'
say '============' '=====' '========' '=====' '======================'
return
w_print_dtl :
wp1 = '1'
do i = 1 to 24
wp = '1'
if h.i > Ø then
   do k = 1 to h.i
      if i < 10 then
         w1h = 'Ø' || i-1 || '-' || 'Ø' || i
if i = 10 then
    w1h = 'Ø' || i-1 || '·' || i
else
    w1h = i-1 || '·' || i
if mn.i.k < 10 then w1mn = ' ' || mn.i.k
    else if mn.i.k < 100 then w1mn = ' ' || mn.i.k
        else if mn.i.k < 1000 then w1mn = ' ' || mn.i.k
            else if mn.i.k < 10000 then w1mn = ' ' || mn.i.k
if wp1 = '1' then
    say wdate w1h mc.i.k w1mn md.i.k
else if wp = '1' & wp1 = 'Ø' then
    say ' ' w1h mc.i.k w1mn md.i.k
else
    say '                  ' mc.i.k,
        w1mn md.i.k
wp = 'Ø'
wp1 = 'Ø'
end
end
call fill_result_array1
call w_init_array
return
w_print_total:
say '  
    Total Counts
say '  MSG.CODE' 'COUNT' 'MESSAGE DESC.'
say ' ==#' '===' '=================================='
do i = 1 to tt
    if t1_mn.i < 10 then w1mn = ' ' || t1_mn.i
    else if t1_mn.i < 100 then w1mn = ' ' || t1_mn.i
        else if t1_mn.i < 1000 then w1mn = ' ' || t1_mn.i
            else if t1_mn.i < 10000 then w1mn = ' ' || t1_mn.i
    say ' ' t1_mc.i w1mn t1_md.i
end
return
w_init_array:
do i = 1 to 24
    h.i = Ø
end
return
fill_result_array1:
do i = 1 to 24
    do k = 1 to h.i
        w_tt = 'Ø'
    if tt > Ø then
do j = 1 to tt
        if t1_mc.j = mc.i.k then
do
    t1_mn.j = t1_mn.j + mn.i.k
    w_tt = '1'
end
if w_tt = '0' then
    do
        tt = tt + 1
        t1_mc.tt = mc.i.k
        t1_mn.tt = mn.i.k
        t1_md.tt = md.i.k
    end
else
    do
        tt = tt + 1
        t1_mc.tt = mc.i.k
        t1_mn.tt = mn.i.k
        t1_md.tt = md.i.k
    end
end
end
return

JCL
//DB2MSGA JOB (ACCT), 'DB2-DB2LOGA',
//     CLASS=A, MSGCLASS=X, MSGLEVEL=(1, 1)
//*
//***************************************************************
//* DELETE SYSPDBA.PSØ.DBP1.LOG.TEMP DATASET                 ***
//***************************************************************
//DELPDS EXEC PGM=IEFBR14
//DELPLOG1 DD DSN=SYSPDBA.PSØ.DBP1.LOG.TEMP,DISP=(MOD,DELETE,DELETE),
//        SPACE=(CYL,(10,10))
//*
//***************************************************************
//* DEFINE SYSPDBA.PSØ.DBP1.LOG.TEMP DATASET                  ***
//***************************************************************
//DEFPDS EXEC PGM=IEFBR14
//DEFPLOG1 DD DISP=(NEW,CATLG,DELETE),
//        DSN=SYSPDBA.PSØ.DBP1.LOG.TEMP,
//        SPACE=(CYL,(50,50)), DCB=(RECFM=FB, LRECL=132)
//*
//***************************************************************
//* LOAD DBP1MSTR LOG RECORD FROM SDSF TO DATASET              ***
//***************************************************************
//SDSGET EXEC PGM=ISFAFD
//ISFOUT DD SYSOUT=*
//ISFIN DD *
SYSNAME PX*
PREFIX DBP1MSTR
OWNER *
DA
FIND 'DBP1MSTR'
++S
PRINT ODSN 'SYSPDBA.PSØ.DBP1.LOG.TEMP' * SHR
PRINT 1 9999999
PRINT CLOSE
/*
************************************************************
/* RUN LOG ANALYSE REXX PROGRAM FOR DBP1 */
************************************************************
//DB2DBP1 EXEC PGM=IKJEFTØ1, DYNAMNBR=3Ø, REGION=4Ø96K
//STEPLIB DD DSN=ISP.SISPLOAD, DISP=SHR
//SYSEXEC DD DSN=SYSPDBA.REXXLIB, DISP=SHR
//SYSTSPRT DD SYSOUT=*;
//SYSTSIN DD *
PROFILE NOPREFIX
%DB2MSGA DBP1
*/

OUTPUT
1READY
PROFILE NOPREFIX
READY
%DB2MSGA DBP1
IKJ56247I FILE DB2LOG NOT FREED, IS NOT ALLOCATED

MEMBER FOR DBP1 MESSAGE OUTPUT

DATE HOUR MSG.CODE COUNT MESSAGE DESC.
============== ===== ======== ===== ======================
30 MAR 2ØØ3 03-04 DSNZØ02I 1 -DBP1 DSNZINIT SUBSYSTEM DBP1 SYSTEM PARAMETERS L
DSNZ7507I 1 -DBP1 DSN7LSTK
DSNYØ01I 1 -DBP1 SUBSYSTEM STARTING
DSNJ127I 1 -DBP1 SYSTEM TIMESTAMP FOR BSDS= 03.Ø89 02:25:27.
DSNJØ01I 2 -DBP1 DSNJW007 CURRENT COPY 1 Active LOG
DSNJØ99I 1 -DBP1 LOG RECORDING TO COMMENCE WITH
DSNRØ01I 1 -DBP1 RESTART INITIATED
DSNRØ03I 1 -DBP1 RESTART...PRI OR CHECKPOIN T
RBA=076FD72A7AB4
DSNRØ04I 1 -DBP1 RESTART...UR STATUS COUNTS
DSNR005I  1  -DBP1 RESTART...COUNTS AFTER FORWARD
DSNR006I  1  -DBP1 RESTART...COUNTS AFTER BACKWARD
DSNR002I  1  -DBP1 RESTART COMPLETED
DSNL003I  1  -DBP1 DDF IS STARTING
DSNT704I  1  -DBP1 SYSIBM.DSNRLST01 HAS BEEN STARTED
          FOR THE R
DSN9022I  3  -DBP1 DSNTCSTR 'START RLIMIT' NORMAL
          COMPLETION
DSNV434I  1  -DBP1 DSNVRP NO POSTPONED ABORT THREADS
          FOUND
DSNL519I  2  -DBP1 DSNLILNR TCP/IP SERVICES
          AVAILABLE
DSNL004I  1  -DBP1 DDF START COMPLETE
DSNL510I  189 -DBP1 DSNLVPCS CONVLIMIT NEGOTIATED
DSNI031I  2  -DBP1 DSNILKES - LOCK ESCALATION HAS
DSN3201I  2  -DBP1 ABNORMAL EOT IN PROGRESS FOR
04-05  DSNJ002I  2  -DBP1 FULL ACTIVE LOG DATA SET
DSNJ001I  2  -DBP1 DSNJW307 CURRENT COPY 1 ACTIVE
          LOG
DSNJ311I  1  -DBP1 DSNJC005 ASYNCHRONOUS LOG ARCHIVE
DSNJ354I  1  -DBP1 DSNJC005 ARCHIVE LOG: ALL ACTIVE
DSNJ359I  1  -DBP1 MEMBER RESPONSES:
DSNJ003I  2  -DBP1 DSNJOFF3 FULL ARCHIVE LOG VOLUME
DSNJ139I  1  -DBP1 DSNLOGOFF TASK ENDED
DSN3201I  2  -DBP1 ABNORMAL EOT IN PROGRESS FOR
DSNB302I  1  -DBP1 DSNB1GC1 GROUP BUFFER POOL GBP4
          IS
DSNB315I  1  -DBP1 DSNB1GC1 GROUP BUFFER POOL GBP4
          IS
05-06  DSNW133I  3  -DBP1 DSNWVOPX - TRACE DATA LOST, OP1
          NOT ACCESS
DSNW123I  3  -DBP1 DSNWVOPX - TRACE RECORDING HAS
          BEEN RESUMED
DSNI031I  1  -DBP1 DSNILKES - LOCK ESCALATION HAS
DSNT376I  16 -DBP1 PLAN=HVLEPL01 WITH
DSNT501I  16 -DBP1 DSNILMCL RESOURCE UNAVAILABLE
DSNJ002I  14 -DBP1 FULL ACTIVE LOG DATA SET
DSNJ001I  14 -DBP1 DSNJW307 CURRENT COPY 1 ACTIVE
          LOG
DSNJ003I  14 -DBP1 DSNJOFF3 FULL ARCHIVE LOG VOLUME
DSNJ139I  6  -DBP1 DSNLOGOFF TASK ENDED
06-07  DSNJ002I  12 -DBP1 FULL ACTIVE LOG DATA SET
DSNJ001I  12 -DBP1 DSNJW307 CURRENT COPY 1 ACTIVE
          LOG
DSNJ003I  12 -DBP1 DSNJOFF3 FULL ARCHIVE LOG VOLUME
DSNJ139I  6  -DBP1 DSNLOGOFF TASK ENDED
07-08  DSNI031I  1  -DBP1 DSNILKES - LOCK ESCALATION HAS
09-10  DSNT376I  4  -DBP1 PLAN=BVIZPL01 WITH
DSNT501I  4  -DBP1 DSNILMCL RESOURCE UNAVAILABLE
10-11 DSNL5101 1 - DBP1 DSNLVPCS CONVLIMIT NEGOTIATED
   DSNJ002I 2 - DBP1 FULL ACTIVE LOG DATA SET
   DSNJ001I 2 - DBP1 DSNJW307 CURRENT COPY 1 ACTIVE
                LOG
   DSNJ003I 2 - DBP1 DSNJOFF3 FULL ARCHIVE LOG VOLUME
   DSNJ139I 1 - DBP1 LOG OFFLOAD TASK ENDED
11-12 DSNJ002I 2 - DBP1 FULL ACTIVE LOG DATA SET
   DSNJ001I 2 - DBP1 DSNJW307 CURRENT COPY 1 ACTIVE
                LOG
   DSNJ003I 2 - DBP1 DSNJOFF3 FULL ARCHIVE LOG VOLUME
   DSNJ139I 1 - DBP1 LOG OFFLOAD TASK ENDED
12-13 DSNJ002I 2 - DBP1 FULL ACTIVE LOG DATA SET
   DSNJ001I 2 - DBP1 DSNJW307 CURRENT COPY 1 ACTIVE
                LOG
   DSNJ003I 2 - DBP1 DSNJOFF3 FULL ARCHIVE LOG VOLUME
   DSNJ139I 1 - DBP1 LOG OFFLOAD TASK ENDED
14-15 DSNL5101 4 - DBP1 DSNLVPCS CONVLIMIT NEGOTIATED
19-20 DSNT376I 1 - DBP1 PLAN=ATMRP003 WITH
   DSNT501I 1 - DBP1 DSNI LMCL RESOURCE UNAVAILABLE
21-22 DSNW133I 29 - DBP1 DSNWVOPX - TRACE DATA LOST, OP1
                NOT ACCESS
   DSNW123I 28 - DBP1 DSNWVOPX - TRACE RECORDING HAS BEEN RESUMED
   DSNP007I 1 - DBP1 DSNPXTN0 - EXTEND FAILED FOR
22-23 DSNW123I 7 - DBP1 DSNWVOPX - TRACE RECORDING HAS BEEN RESUMED
   DSNW133I 6 - DBP1 DSNWVOPX - TRACE RECORDING HAS NOT ACCESS
   DSNJ002I 2 - DBP1 FULL ACTIVE LOG DATA SET
   DSNJ001I 2 - DBP1 DSNJW307 CURRENT COPY 1 ACTIVE
                LOG
   DSNJ003I 2 - DBP1 DSNJOFF3 FULL ARCHIVE LOG VOLUME
   DSNJ139I 1 - DBP1 LOG OFFLOAD TASK ENDED
23-24 DSNJ031I 1 - DBP1 DSNILKES - LOCK ESCALATION HAS
31 MAR 2003 00-01 DSNT376I 13 - DBP1 PLAN=POSPLN01 WITH
   DSNT501I 13 - DBP1 DSNIL LMCL RESOURCE UNAVAILABLE
   DSNJ002I 2 - DBP1 FULL ACTIVE LOG DATA SET
   DSNJ001I 2 - DBP1 DSNJW307 CURRENT COPY 1 ACTIVE
                LOG
   DSNJ311I 1 - DBP1 DSNJC005 ASYNCHRONOUS LOG ARCHIVE
   DSNJ354I 1 - DBP1 DSNJC005 ARCHIVE LOG: ALL ACTIVE
   DSNJ359I 1 - DBP1 MEMBER RESPONSES:
   DSNJ003I 2 - DBP1 DSNJOFF3 FULL ARCHIVE LOG VOLUME
   DSNJ139I 1 - DBP1 LOG OFFLOAD TASK ENDED
01-02 DSNT365I 1 - DBP1 NO DATABASES FOUND
   DSNT365I 1 - DBP1 DSNTDDIS 'DISPLAY DATABASE'
                NORMAL COMPLETI
   DSNJ031I 2 - DBP1 DSNILKES - LOCK ESCALATION HAS
   DSNW133I 18 - DBP1 DSNWVOPX - TRACE DATA LOST, OP1
                NOT ACCESS
<table>
<thead>
<tr>
<th>MSG. CODE</th>
<th>COUNT</th>
<th>MESSAGE DESC.</th>
</tr>
</thead>
<tbody>
<tr>
<td>DSNZ002I</td>
<td>1</td>
<td>-DBP1 DSNZINIT SUBSYSTEM DBP1 SYSTEM PARAMETERS LISTED</td>
</tr>
<tr>
<td>DSN7507I</td>
<td>1</td>
<td>-DBP1 DSN7LSTK</td>
</tr>
<tr>
<td>DSNY001I</td>
<td>1</td>
<td>-DBP1 SUBSYSTEM STARTING</td>
</tr>
<tr>
<td>DSNJ127I</td>
<td>1</td>
<td>-DBP1 SYSTEM TIMESTAMP FOR BSDS= 03.009 02:25:27.</td>
</tr>
<tr>
<td>DSNJ007I</td>
<td>54</td>
<td>-DBP1 DSNJW007 CURRENT COPY 1 ACTIVE LOG</td>
</tr>
<tr>
<td>DSNJ099I</td>
<td>1</td>
<td>-DBP1 LOG RECORDING TO COMMENCE WITH</td>
</tr>
<tr>
<td>DSNR001I</td>
<td>1</td>
<td>-DBP1 RESTART INITIATED</td>
</tr>
<tr>
<td>DSNR003I</td>
<td>1</td>
<td>-DBP1 RESTART...PRIOR CHECKPOINT RBA=076FD72A7AB4</td>
</tr>
<tr>
<td>DSNR004I</td>
<td>1</td>
<td>-DBP1 RESTART...UR STATUS COUNTS</td>
</tr>
<tr>
<td>DSNR005I</td>
<td>1</td>
<td>-DBP1 RESTART...COUNTS AFTER FORWARD</td>
</tr>
<tr>
<td>DSNR006I</td>
<td>1</td>
<td>-DBP1 RESTART...COUNTS AFTER BACKWARD</td>
</tr>
<tr>
<td>DSNR002I</td>
<td>1</td>
<td>-DBP1 RESTART COMPLETED</td>
</tr>
<tr>
<td>DSNL003I</td>
<td>1</td>
<td>-DBP1 DDF IS STARTING</td>
</tr>
<tr>
<td>DSNT704I</td>
<td>1</td>
<td>-DBP1 SYSIBM.DSNRLST01 HAS BEEN STARTED FOR THE RDSN902I</td>
</tr>
<tr>
<td>DSNV434I</td>
<td>1</td>
<td>-DBP1 DSNVRP NO POSTPONED ABORT THREADS FOUND</td>
</tr>
<tr>
<td>DSNL519I</td>
<td>2</td>
<td>-DBP1 DSNLILNR TCP/IP SERVICES AVAILABLE</td>
</tr>
<tr>
<td>DSNL004I</td>
<td>1</td>
<td>-DBP1 DDF START COMPLETE</td>
</tr>
<tr>
<td>DSNL510I</td>
<td>194</td>
<td>-DBP1 DSNLVCPS CONV LIMIT NEGOTIATED</td>
</tr>
<tr>
<td>DSNI031I</td>
<td>7</td>
<td>-DBP1 DSNLILKES - LOCK ESCALATION HAS OCCURRED</td>
</tr>
<tr>
<td>DSNI390I</td>
<td>4</td>
<td>-DBP1 ABNORMAL EOT IN PROGRESS</td>
</tr>
<tr>
<td>DSNJ002I</td>
<td>52</td>
<td>-DBP1 FULL ACTIVE LOG DATA SET</td>
</tr>
<tr>
<td>DSNJ391I</td>
<td>2</td>
<td>-DBP1 DSNJCOOS ASYNCHRONOUS LOG ARCHIVE</td>
</tr>
<tr>
<td>DSNJ354I</td>
<td>2</td>
<td>-DBP1 DSNJCOOS ARCHIVE LOG: ALL ACTIVE</td>
</tr>
</tbody>
</table>
| DSNJ359I | 2     | -DBP1 MEMBER RESPONSES:
How to verify DB2 UDB back-ups

It is all very well taking DB2 back-ups using the BACKUP command (>db2 backup db <db-alias> to <drive>), but how do you know that the back-up is usable? DB2 UDB (7.2 and 8.1) provides a utility called db2ckbkp to perform this task.

This article looks at the db2ckbkp utility. The DB2 UDB V8 Command Reference manual (SC09-4828-00) describes the function of this utility quite distinctly, as “… This utility can be used to test the integrity of a backup image and to determine whether or not the image can be restored.” This is a very important although often overlooked task.

I ran all the commands in this article on a Windows 2000 machine running DB2 8.1 FP1. I used the SAMPLE database as the test database.

To get a list of all possible parameters, use the command:

>db2ckbkp -h

Let’s look at some practical uses of this command. I took a back-up of the SAMPLE database to the C:\backups directory. Which
of the possible parameters is the best one to use? If you don’t specify any parameters, then what you get back when you invoke the command is a line saying that the image verification was successful (or not).

The utility can be invoked in one of two ways:

- By CDing to the directory containing the back-up image and issuing the command with the file name as a parameter:

  C:\backups\SAMPLE.Ø\DB2\NODEØØØØ\CATNØØØØ\2003Ø323>db2ckbkp 185549.ØØ1

- By issuing the command using the full back-up path name:

  C:\>db2ckbkp
  C:\backups\SAMPLE.Ø\DB2\NODEØØØØ\CATNØØØØ\2003Ø323\185549.ØØ1

Note that in this second example you need to specify the full path name even if you have only one back-up file in the directory (you can’t just specify the directory name).

Both of the above invocations produce a couple of lines of output, but the key line to look for is the last line, which says “Image Verification Complete - successful.”.

What about the other parameters? The **-h** parameter dumps the media header and performs an image verification. It produces the following output:

C:\>db2ckbkp -h
C:\backups\SAMPLE.Ø\DB2\NODEØØØØ\CATNØØØØ\2003Ø323\185549.ØØ1

=====================  
MEDIA HEADER REACHED:  
=====================  

<table>
<thead>
<tr>
<th>Server Database Name</th>
<th>SAMPLE</th>
</tr>
</thead>
<tbody>
<tr>
<td>Server Database Alias</td>
<td>SAMPLE</td>
</tr>
<tr>
<td>Client Database Alias</td>
<td>SAMPLE</td>
</tr>
<tr>
<td>Timestamp</td>
<td>20030323185549</td>
</tr>
<tr>
<td>Database Partition Number</td>
<td>0</td>
</tr>
<tr>
<td>Instance</td>
<td>DB2</td>
</tr>
<tr>
<td>Sequence Number</td>
<td>1</td>
</tr>
<tr>
<td>Release ID</td>
<td>AØØ</td>
</tr>
<tr>
<td>Database Seed</td>
<td>3E8E829E</td>
</tr>
<tr>
<td>DB Comment's Codepage (Volume)</td>
<td>0</td>
</tr>
<tr>
<td>DB Comment (Volume)</td>
<td></td>
</tr>
<tr>
<td>DB Comment's Codepage (System)</td>
<td>0</td>
</tr>
</tbody>
</table>
The proper image path would be:

SAMPLE.0\DB2\NODE0000\CATN0000\20030323\185549.001

[1] Buffers processed: #######

Image Verification Complete - successful.

This output might be useful if somebody just sends you the back-up file. You can then re-create the directory structure, because the output gives you the instance, back-up date, database name, and partition number, in fact everything you need to recreate the C:\backups\SAMPLE.0\DB2\NODE0000\CATN0000\20030323 path. You can then recreate the full path name and use the file in a recovery situation.

The -H parameter does the same as the -h parameter, except that it does not perform an image verification – all it does is display the media header.

If you use any of the -a, -c, -d, -l, -o parameters, you get back numerous pages of output, with the last line indicating whether the image verification was successful or not. I am not sure how useful the output produced by using these parameters is – it may be required by the IBM support centre if you report a problem with the back-up.

I think that it is vitally important that once I take a DB2 back-up I verify it. I would do this using the \texttt{db2ckbkp} command without any parameters. This does not substitute for performing regular disaster recovery tests, but should be part of that process. I hope I have shown how simple the invocation of the \texttt{db2ckbkp}
command is. If you do not currently verify you back-ups once you have taken them, I would strongly recommend that you do so — it is a quick step that could prove decisive if you ever find yourself in the situation where you need to use your back-ups in a real disaster scenario!

C Leonard
Freelance Consultant (UK)

Why not share your expertise and earn money at the same time? DB2 Update is looking for technical articles and hints and tips that experienced DB2 users have written to make their life, or the lives of their users, easier. We would also be interested in articles about performance and tuning, and information and tips for DB2 DBAs.

Articles can be of any length and can be sent to Trevor Eddolls at any of the addresses shown on page 2. Alternatively, they can be e-mailed to trevore@xophon.com. A copy of our Notes for contributors is available from www.xophon.com/nfc.
CAF interface with caller in amode 24 or 31 and more – part 2

This month we conclude the code that provides a CAF (Call Attachment Facility) in 24-bit and 31-bit mode.

```plaintext
CONSPLAN DS CL8 *
    DS CL1 *
CONSTYPE DS CL1 *
    DS CL1 * 20
CHECKMSG DS OCL25 * 00
CONSRETC DS CL8 *
    ORG *. 8 *
CONSSQLC DS CL8 *
    DS CL1 *
CONSREAS DS CL8 * 17
    ORG *. 8 *
CONSSQLS DS CL8 *
    DS CL1 * 18
    ORG *. 9 *
CONSABND DS CL4 *
    DS CL5 * 18
CONSSTPN DS CL16 *
    DS CL1 *
CONSFUNC DS CL12 *
    DS CL4 * 34
    ORG *
    LTORG
    EJECT
DB2CAFDS DSECT *
DB2CAF DS OF *
CAFRETC DS F RETURN CODE
CAFREAC DS CL8 REASON CODE
CAFSSSID DS CL4 DB2 SSID
CAFPLAN DS CL8 PLAN NAME
CAFTERM DS CL4 TERMINATION OPTION
CAFTERMA EQU C'A' CLOSE THREAD WITH ABRT
CAFTERMS EQU C'S' CLOSE THREAD WITH SYNC
CAFFUNC DS CL1 FUNCTION CODE TO EXECUTE
CAFCONNE EQU C'1' CONNECT TO DB2
CAFOPENE EQU C'2' OPEN THREAD
CAFCLOSE EQU C'3' CLOSE THREAD WITH CAFTERM
CAFDISCE EQU C'4' DISCONNECT FROM DB2
    EJECT
SQLCADS DSECT *
    ***$$$ *
```
* EXEC SQL INCLUDE SQLCA *
***$$$ SQLCA *
SQLCA    DS        ØF                  *
SQLCAI    DS        CL8                 ID
SQLCBC    DS        F                   BYTE COUNT
SQLCODE    DS        F                   SQLCODE, RETURN CODE FROM DB2
SQLERRL    DS        H                   LENGTH ERROR MSG
SQLERRM    DS        CL7Ø                TEXT ERROR MSG
SQLERRP    DS        CL8                 IMPLEMENTATION-DEPENDENT
SQLERRD    DS        6F                  *
SQLWARN    DS        ØCL8                WARNING FLAGS
SQLWARNO    DS        C'W' IF ANY *
SQLWARN1    DS        C'W' = WARNING *
SQLWARN2    DS        C'W' = WARNING *
SQLWARN3    DS        C'W' = WARNING *
SQLWARN4    DS        C'W' = WARNING *
SQLWARN5    DS        C'W' = WARNING *
SQLWARN6    DS        C'W' = WARNING *
SQLWARN7    DS        C'W' = WARNING *
SQLEXT    DS        ØCL8                  *
SQLWARN8    DS        C'W' = WARNING *
SQLWARN9    DS        C'W' = WARNING *
SQLWARNA    DS        C'W' = WARNING *
SQLSTATE    DS        CL5                  *
***$$$ EJECT *
SQLPLDS    DSECT                  *
***$$$ EXEC SQL INCLUDE SQLPLIST *
***$$$ SQLPLIST *
SQLPLIST    DS        ØF                  SQL PARAMETER LIST
SQLPLLEN    DS        H                   LENGTH
SQLFLAGS    DS        H                   FLAGS
SQLCTYPE    DS        H                   ID
SQLPROGN    DS        CL8                 PROGRAM NAME
SQLTIMES    DS        ØCL8                ID
SQLTIME1    DS        CL4                 ID
SQLTIME2    DS        CL4                 ID
SQLSECTN    DS        CL2                 ID
SQLCODEP    DS        CL4                 ADDR SQLCA
SQLVAPARM    DS        F                   ID
SQLAPARM    DS        F                   ID
SQLSTNUM    DS        H                   STMT NUMBER
SQLTYPE    DS        H                   ID
SQLAVARS    DS        F                   ID
SQLAVELT    DS        ØCL12                ID
SQLAVTYPE    DS        H                   ID
SQLAVLEN    DS        H                   ID
SQLAVADD    DS        A                   ID
SQLAVIND    DS        A                   ID

DB2 catalog/directory sizing

This is a process for resizing the VSAM datasets of a DB2 catalog/directory to an optimal size, allowing for a bit of growth. A batch job is run to produce a report and an output PDS with JCL to resize the datasets. The original version of this was written in 1997 by James Gill from Triton Consulting for a DB2 V3.1 system and since then I have refined it, enhanced it, and kept it valid for all versions up to DB2 V7.1.
HOW IT WORKS

The DB2SIZE EXEC gets MVS catalog information for all the VSAM datasets via LISTCAT commands. It uses the high-used-RBA to check how much space is currently used. Based on the percentage of the allocated space that is used or on the number of extents, it will decide whether each object needs to be resized or not. If an object is to be resized, a separate PDS member is created with JCL for that purpose. The generated JCL allocates new VSAM datasets and copies the data from the old datasets into them. The copying is done via the DB2 DSN1COPY program, and if that is successful the old datasets are deleted. When resizing, it will allow for a specified percentage of free space in each dataset, and if one is near the maximum size (2GB or 4GB) it will allocate an extra new dataset for DB2 to extend into.

SAMPLE JOB

There are various parameters to tailor it as you want, and they are explained in detail in the comments in this sample job.

```plaintext
// SIZEDSN1 JOB 'DB2 SIZE', CLASS=A, MSGCLASS=X,
//   NOTIFY=&SYSUID, MSGLEVEL=(1, 1), REGION=32M
// *
// * GENERATE CATALOG AND DIRECTORY RESIZING JOBS FOR DSN1
// *
// * INTO THE PDS ALLOCATED TO ISPFILE
// *
// SIZE EXEC PGM=IKJEFT01, REGION=4M
// SYSEXEC DD DS=SHR, DSN=ISPF.SISPEXEC <- DB2SIZE EXEC
// DD DS=SHR, DSN=SYS1.ISPF.ISPEXEC
// SYSPROC DD DS=SHR, DSN=SYS1.ISPF.ISPCLU1
// ISPMLIB DD DS=SHR, DSN=SYS1.ISPF.ISPMENU
// ISPPLIB DD DS=SHR, DSN=SYS1.ISPF.ISPPENU
// ISPBLIB DD DS=SHR, DSN=ISPF.ISPCLU1
// ISPSTLIB DD DS=SHR, DSN=ISPFILE
// ISPSTLIB DD DS=SHR, DSN=ISPF.ISPCLU1
// ISPCTL1 DD DS=NEW, UNIT=VIO, SPACE=(CYL, (1, 1)),
// DCB=(LRECL=80, BLKSIZE=800, RECFM=FB)
// ISPCTL2 DD DS=NEW, UNIT=VIO, SPACE=(CYL, (1, 1)),
// DCB=(LRECL=80, BLKSIZE=800, RECFM=FB)
// ISPLST1 DD DS=NEW, UNIT=VIO, SPACE=(CYL, (1, 1)),
// DCB=(LRECL=121, BLKSIZE=1210, RECFM=FB, DSORG=PO)
// ISPPROF DD DSN=&PROF, DS=NEW, PASS, UNIT=VIO, SPACE=(TRK, (5, 2, 3)),
// DCB=(LRECL=80, BLKSIZE=3120, RECFM=FB, DSORG=PO)
```
// SYSTSPRT DD SYSOUT=*  
// SYSTSIN DD *  
ISPSTART CMD(%DB2SIZE)  

/*  
** ** COL1 - AN ASTERISK IN COLUMN 1 INDICATES A COMMENT  
**  MODE = <MODE>  
**   HLQ = <HLQ>  
**   TSVOLS = <TS VOL1>  
**       <TS VOL2>  
**   IXVOLS = <IX VOL1>  
**       <IX VOL2>  
**   FORCE = <FORCE>  
**   INCREASE = <%AGE INCREASE>  
**   SECPERC = <SECONDARY %AGE>  
**   MINUSED = <MIN %AGE USED>  
**   MAXUSED = <MAX %AGE USED>  
**   MAXSIZE = <2 OR 4 GB>  
**    MAXPRIM = <MAX PRIM CYL>  
**  WHERE <MODE> IS "GENERATE" TO GENERATE CAT/DIR RESIZING JOBS  
**  "LOADINF" TO PRODUCE CURRENT SIZING STATS AS A  
**    DB2 LOAD UTILITY INPUT - SEE THE  
**    SAMPLE MEMBER SIZINGLD FOR MORE DETAILS  
**  "REPORT" TO PRODUCE CURRENT CAT/DIR SIZING REPORT  
**  <HLQ> IS THE HIGH-LEVEL QUALIFIER FOR THE DB2 CATALOG AND  
**    DIRECTORY DATASETS THAT YOU WISH TO WORK WITH  
**    THIS IS A REQUIRED PARAMETER.  
**  <TSVOLS> IS A CANDIDATE VOLUME THAT WILL BE PASSED TO THE  
**    VOLUME PARAMETER OF THE TABLESPACE IDCAMS DEFINE  
**    STATEMENTS (MAXIMUM OF 7 VOLUMES)  
**    THIS IS REQUIRED IN GENERATE MODE, BUT IF SMS  
**    MANAGES THEM IT WILL OVERRIDE THIS PARAMETER  
**  <IXVOLS> IS A CANDIDATE VOLUME THAT WILL BE PASSED TO THE  
**    VOLUME PARAMETER OF THE INDEXSPACE IDCAMS DEFINE  
**    STATEMENTS (MAXIMUM OF 7 VOLUMES)  
**    THIS IS REQUIRED IN GENERATE MODE, BUT IF SMS  
**    MANAGES THEM IT WILL OVERRIDE THIS PARAMETER  
**  <FORCE> IF SET TO "ON", THIS FORCES THE GENERATE PROCESS  
*/
TO Resize every object that it gets a match with, rather than just those in multiple extents or allocated in large numbers of tracks (>14). This is optional - the default = "off".

<INCREASE> This is the percentage increase over the required space to allocate in a single extent. EG if a tablespace is allocated as PRI = 70 cylinders, and SEC = 20 cylinders, and it is in three extents using 100 cylinders out of the 110 cylinders allocated.

If the increase parameter were specified as 25 (the default percentage) the new primary allocation would be 125 cylinders. Thus the resizing should allow for the normal growth of catalog & directory objects default = 25%.

<SECPERC> Secondary extent allocation percentage of the used space. Default = 10%.

In the example given for the INCREASE parameter with a SECPERC value of 10, secondary space allocation would be 13 cylinders (ie 12.5 rounded up).

<MINUSED> This is used to decide which tablespace/indexspace to resize, depending on the percentage of space used of the allocated space, (removing over-allocation).

If the tablespace/indexspace percentage used is greater than or equal to 'MINUSED' (and also less than or equal to 'MAXUSED'), a job will be created to re-allocate appropriately. The allocation will be for the used space plus the percent specified in the 'INCREASE' parameter (see above).

Default = 1% -> Empty datasets will not be resized.

<MAXUSED> SEE MINUSED

Default = 49% -> Datasets less than half used will be resized, even if they have only 1 extent.

Note: That DB2 can pre-format pages, usually up to a track/cylinder boundary. This can lead to some misleading high-used-RBA values (which is what we use to calculate the percentage used). Therefore be particularly sceptical about enlarging the small tablespaces/indexspaces which appear to be 100% full.

<MAXSIZE> Maximum total space possible (usually 2GB, but can be 4GB if SMS is set up for it). Default = 2 (GB)

<MAXPRIM> Maximum primary space to allocate (cylinder), but can't be bigger than MAXSIZE. Default 2913 cyl = 2GB

SET THIS TO SUIT YOUR TARGET DISKS.

/*
** //SYSIN DD *
** //** CATALOG AND DIRECTORY RESIZING PARAMETERS
** MODE = GENERATE - GENERATE RESIZING JOBS
** HLQ = DSN1 - DSN1 IS THE HLQ FOR SUB-SYSTEM DSN1
** //** SPECIFY VOLUMES TO BE USED BY CATALOG AND DIRECTORY TABLESPACES
** TSVOLS = TSVVV1 - SPECIFY ANY EXTRA VOLUME ON THE NEXT LINE
** TSVVV2 - (ONLY A COMMENT WHILE * IN COL 1)
** TSVVV3 - (ONLY A COMMENT, UP TO 7 VOLS CAN BE GIVEN)
** //** SPECIFY VOLUMES TO BE USED BY CATALOG AND DIRECTORY INDEXSPACES
** IXVOLS = IXVVV1 - THIS CAN BE OVERRIDEN BY SMS
** //** END OF PARAMETERS
** //<<NOTE>>>
** ISPFILE IS THE TARGET OF THE GENERATED JCL
** ISPFILE DD DSN=DB2.SIZEDSN1.Dyymmdd,DISP=(NEW,CATLG), <- NAME OK?
** UNIT=SYSDA,SPACE=(CYL,(1,1,45)),
** DCB=(LRECL=80,BLKSIZE=0,RECFM=FB)
**
** SAMPLE REPORT
** Here is a sample fragment from a report to show how it looks:
** *=*=*=*=*=*=*=*=*=*=*=*=*=*=*=*=*=*=*=*=*=*=*=*=*=*=*=*=*=*=*=*=*=*
** Options selected from parameters and defaults:-
** Established Mode : GENERATE
** High Level Qualifier : SDB0
** Tablespace Volumes : SSDXXX
** Indexspace Volumes : SSDXXX
** Setting of Force : NO
** % Increase : 25
** Sec % Allocation : 10
** Min % Used : 1
** Max % Used : 49
** Max Space Size (gb) : 2
** Max Prim Space (cyl) : 2913
** *=*=*=*=*=*=*=*=*=*=*=*=*=*=*=*=*=*=*=*=*=*=*=*=*=*=*=*=*=*=*=*=*=*
** 152 names retrieved...
** Generating new sizing for SDB0.DSNDBC.DSNDB01.DBD01.I0001.A001
**
** © 2003. Xephon UK telephone 01635 33848, fax 01635 38345. USA telephone (303) 410 9344, fax (303) 438 0290.
Generating new sizing for SDB0.DSNDBC.DSNDB01.DSNLLX01.10001.A001

Old size info - Units : CYLINDER Alloc : 1070 Used : 1068
Pri Alloc : 370 Volume : SSD020
Sec Alloc : 20
Extents : 52 Used % : 100

New size info - Units : CYLINDER
Pri Alloc : 1335
Sec Alloc : 134
Extents : 1 New % : 80

Generating new sizing for SDB0.DSNDBC.DSNDB01.SPT01.10001.A002

Old size info - Units : CYLINDER Alloc : 194 Used : 132
Pri Alloc : 97 Volume : SSD000
Sec Alloc : 97
Extents : 2 Used % : 68

New size info - Units : CYLINDER
Pri Alloc : 165
Sec Alloc : 17
Extents : 1 New % : 80

Generating new sizing for SDB0.DSNDBC.DSNDB01.SPT01.10001.A003

Old size info - Units : CYLINDER Alloc : 2913 Used : 2913
Pri Alloc : 1700 Volume : SSD005
Sec Alloc : 100
Extents : 39 Used % : 100

New size info - Units : CYLINDER
Pri Alloc : 2913
Sec Alloc : 292
Extents : 1 New % : 100

Allocating extra dataset SDB0.DSNDBC.DSNDB01.SPT01.10001.A004

New size info - Units : CYLINDER
Pri Alloc : 500 Volume : SSDXXX
Sec Alloc : 50
Extents : 1 New % : 0
Generating new sizing for SDBØ.DSNDBC.DSNDBØ6.DSNKLXØ1.IOØØ1.AØØ1

Old size info - Units : TRACK         Alloc : 2Ø       Used : 18
                  Pri Alloc : 4            Volume : SSDØØ2
                  Sec Alloc : 4            Extents : 5       Used % : 9Ø

New size info - Units : CYLINDER
                  Pri Alloc : 2            Extents : 1       New % : 6Ø
                  Sec Alloc : 1

Generating new sizing for SDBØ.DSNDBC.DSNDBØ6.SYSVIEWS.IOØØ1.AØØ1

Old size info - Units : CYLINDER         Alloc : 5ØØ      Used : 36
                  Pri Alloc : 5ØØ          Volume : SSDØ19
                  Sec Alloc : 2Ø            Extents : 1       Used % : 7

New size info - Units : CYLINDER
                  Pri Alloc : 45
                  Sec Alloc : 5
                  Extents : 1                   New % : 8Ø

Completion Report:
Space required to allocate new objects is 316899 tracks
or 21127 Cylinders
Largest dataset to be reallocated is
SDBØ.DSNDBC.DSNDBØ1.SPTØ1.IOØØ1.AØØ2
Space required to allocate this is 2913 Cylinders
Additional new space required is -1Ø774 tracks
or -718 Cylinders
-12Ø22 tracks for Tablespaces
1248 tracks for Indexes

DB2SIZE EXEC

/* REXX -----------------------------------------------*/
/* Acquire sizing information for the DB2 catalog and directory. */
/* Utilizing this information, optionally generate jobs and IDCAMS */
/* control cards to perform the resizing operation. */
/* */
/* Dependencies:- */
/* ISPSLIB - DB2SIZE1 skeleton for JOBCARD */
/* DB2SIZE2 skeleton for DEFINE, COPY & DELETE JCL steps */
/* DB2SIZE3 skeleton for DEFINE JCL step */
/* Parameters:- */
/* 1. MODE   The mode of operation of this EXEC. Valid values are: */
/* GENERATE - Performs data capture, analysis, and job */
/*   generation. A report of targetted objects */
/*   is produced to SYSTSPRT. */
/* LOADINF - Generates a DB2 Load utility file, to assist */
/*   in tracking changes to the size of the */
/*   catalog and directory. This is accompanied */
/*   by a report produced to SYSTSPRT. */
/* REPORT - Produces just the report of the current size */
/*   of the catalog and directory */
/* 2. HLQ    This is the high-level dataset name qualifier for the */
/*   catalog and directory. Just the VCAT name is required. */
/* 3. TSVOLS This is a list of candidate volumes for the catalog and */
/*   directory tablespaces. It should be specified in the */
/*   form: "TSVOLS = volser1", and any extra volumes go on */
/*   the following lines (one volume per line). */
/*   This list will be used in the IDCAMS define statements, */
/*   but may be overridden by DFSMS if the datasets are */
/*   under SMS control. */
/* 4. IXVOLS This is a list of candidate volumes for the catalog and */
/*   directory indexes. It should be specified in the form: */
/*   "IXVOLS = volser1", and any extra volumes go on the */
/*   following lines (one volume per line). */
/*   This list will be used in the IDCAMS define statements, */
/*   but may be overridden by DFSMS if the datasets are */
/*   under SMS control. */
/* 5. FORCE If set to "ON", this forces the GENERATE process */
/*   to resize every object that it gets a match with, */
/*   rather than just those in multiple extents or */
/*   allocated in large numbers of tracks (>14). */
/*   this is optional - the default is "OFF" */
/* 6. INCREASE This is the percentage increase over the required */
/*   space to allocate in a single extent. Eg if a */
/*   tablespace is allocated as pri = 70 cylinders, and */
/*   sec = 20 cylinders, and it is in three extents using */
/*   100 cylinders out of the 110 cylinders allocated. */
/*   If the INCREASE parameter were specified as 25 (the */
/*   default percentage) the new primary allocation would */
/*   be 125 cylinders. Thus the resizing should allow */
/*   for the normal growth of Catalog & Directory objects. */
/* 7. SECPERC Percentage of used space allocated for each secondary */
/*   extent (all used space is normally in primary alloc). */
/*   The default is 10 (percent). */
/*   In the example given for the INCREASE parameter, with */
/*   a SECPERC value of 10, the secondary space allocation */
/*   would be 13 cylinders (ie 12.5 rounded up). */
/* 8. MINUSED This is used to decide which tablespace/indexspace to */
/*   resize, depending on the percentage of space used out */
/*   of the allocated space, (removing over allocations). */
/*   If the tablespace/indexspace percentage used is */
/* greater than or equal to 'MINUSED' (and also less */
/* than or equal to 'MAXUSED'), a job will be created */
/* to re-allocate appropriately. The allocation will */
/* be for the used space plus the percent specified */
/* in the 'INCREASE' parameter (see above). */
/* Default = 1% hence empty datasets will not be resized. */
/* 9. MAXUSED  See 'MINUSED' above. */
/* Default = 49% hence datasets which are less than half */
/* used will be resized, even if they have only 1 extent. */
/* Note: that DB2 can pre-format pages, usually up to */
/* a track/cylinder boundary. This can lead to some */
/* misleading high-used-RBA values (which is what we */
/* use to calculate the percentage used). Therefore */
/* be particularly sceptical about enlarging the very */
/* small tablespaces/indexspaces which appear to be */
/* 100% full. */
/* 10. MAXSIZE Maximum total space possible per dataset (usually 2GB */
/* but can be 4GB if SMS is set up for it). */
/* 11. MAXPRIM Maximum primary space to allocate (cylinders), which */
/* should be no larger than 2GB (2913 cyl), see MAXSIZE */
/*-------------------------------------------------------------------*/
Call Process_Parms        /* Fetch and interpret input parameters */
Call Initialise           /* Establish parms setup variables */
Call Fetch_DSNames        /* Query ICF catalog for DB2 Catalog and */
                        /* Directory dataset names */
/* Acquire current sizing information, and build JCL for resizing */
Out. = "" /*
Say Left("",79,"_") /* process each dataset from Fetch_DSNames */
Do i = 1 To No_Objs    /* get catalog info about the dataset */
    Call Extract_LISTC_Info /* determine which mode we are running in, and perform it's activity */
    Select
        When Mode = "GENERATE" Then Call Gen_Opts
        When Mode = "LOADINF" Then Call Load_Opts
        Otherwise Call Report_Opts
    End
End /* End of individual objects loop */
/* Depending on which Mode, produce the appropriate completion report */
Select
    When Mode = "GENERATE" Then Call Generate_Complete
    When Mode = "LOADINF" Then Call Load_Complete
    Otherwise Call Report_Complete
End
/* === Process complete === */
Exit
Bye_bye:
    Say "Object in error is" Sys_Obj.i
    Say "Volume   " Volser
    Say "Space in " SpacType
    Say "Primary  " SpacPri
Say "Secondary" SpacSec
Say "Extents  " NumExt
Say "Used %  " Format(UsedPerc, 3, Ø)
ZISPFRC = 12
Address IspExec "VPUT ZISPFRC"
Exit 12
Error: Procedure
   Arg Err_Str
   Say Left("", 79, "*")
   Say "*"||Center(Err_Str, 77)||"*"
   Say Left("", 79, "*")
   ZISPFRC = 12
   Address IspExec "VPUT ZISPFRC"
Exit 12
/* Define constants and initialize variables */
Initialise:
   True = 1
   False = Ø
   Out. = ""
   Sys_Obj. = ""
   New_Dset_Needed = Ø
   No_Objs = Ø
   Job_No = Ø
   Delta = Ø
   DAT_Delta = Ø
   IND_Delta = Ø
   DASD = Ø
   Largest_Size = Ø
   Largest_Name = ""
If Mode = "LOADINF" Then Do
   t_date = Date("S")
   t_date = SubStr(t_date, 1, 4)||"-"||SubStr(t_date, 5, 2)||","||SubStr(t_date, 7, 2)||"."
   t_time = Translate(Time(), ".", ":")||".000000"
   Load_TS = t_date||t_time
   NewStack
End
Return
/* Determine the names of all of the catalog and directory objects */
Fetch_DSNames:
   x = OutTrap("Out.", "*", "CONCAT")
   Address TSO "LISTC LVL("HLQ".DSNDBC.DSNDB01) NAME"
   Address TSO "LISTC LVL("HLQ".DSNDBC.DSNDB06) NAME"
   x = OutTrap("Off")
   Do i = 1 To Out.Ø
      If Word(Out.i, 1) = "CLUSTER" Then Do
         No_Objs = No_Objs + 1
         Sys_Obj.No_Objs = Word(Out.i, 3)
      End
   End
End
Say No_Obj "names retrieved..."
Return

/* Extract various values about a VSAM dataset from LISTCAT output */

Extract_LISTC_Info:
  x = OutTrap("Out." , "*" , "NOCONCAT") /* LISTC -> Out.x */
  Address TSO "LISTC ENT('Sys_Obj.i'') ALL"
  x = OutTrap("OFF")
  Do j = 8 To Out.Ø /* Extract LISTC information */
  Select
   When Left(Word(Out.j,4),5) = "CI/CA" Then Do
     T_Str = Word(Out.j,4)
     T_Str = Translate(T_Str," ", "-"
     ci_ca = Word(T_Str,2)
   End
   When SubStr(Word(Out.j,3),1,7) = "EXTENTS" Then Do
     T_Str = Word(Out.j,3)
     T_Str = Translate(T_Str," ", "-"
     NumExt = Word(T_Str,2)
   End
   When SubStr(Word(Out.j,1),1,10) = "SPACE-TYPE" Then Do
     T_Str = Word(Out.j,1)
     T_Str = Translate(T_Str," ", "-"
     SpacType = Word(T_Str,3)
     T_Str = Word(Out.j,2)
     T_Str = Translate(T_Str," ", "-"
     Hi_Alloc_RBA = Word(T_Str,4)
   End
   When SubStr(Word(Out.j,1),1,9) = "SPACE-PRI" Then Do
     T_Str = Word(Out.j,1)
     T_Str = Translate(T_Str," ", "-"
     SpacPri = Word(T_Str,3)
     T_Str = Word(Out.j,2)
     T_Str = Translate(T_Str," ", "-"
     Hi_Used_RBA = Word(T_Str,4)
   End
   When SubStr(Word(Out.j,1),1,9) = "SPACE-SEC" Then Do
     T_Str = Word(Out.j,1)
     T_Str = Translate(T_Str," ", "-"
     SpacSec = Word(T_Str,3)
   End
   When SubStr(Word(Out.j,1),1,6) = "VOLSER" Then Do
     T_Str = Word(Out.j,1)
     T_Str = Translate(T_Str," ", "-"
     Volser = Word(T_Str,2)
   End
   When Left(Word(Out.j,2),11) = "PHYRECS/TRK" Then Do
     T_Str = Word(Out.j,2)
     T_Str = Translate(T_Str," ", "-"
     pages_trk = Word(T_Str,2)        /* equals 12 for 339Ø */
   End
When Left(Word(Out.j,2),9) = "TRACKS/CA" Then Do
  T_Str = Word(Out.j,2)
  T_Str = Translate(T_Str,"","-")
  tracks_ca = Word(T_Str,2)  /* equals 1 or 15 */
  Leave j /* We now have all the info we need - breakout */
End
Otherwise Nop
End

UsedPerc = 100 * Hi_Used_RBA / Hi_Alloc_RBA
Return

/* This code section performs the individual item processing for the
Generate mode. Note that the FORCE option causes every catalog and
directory dataset to be processed by this routine, which is used to
generate a job to resize the object concerned. */

Gen_Opts:
/* Any dataset name ending in ".OLD" to be ignored for resizing */
If Right(Sys_Obj.i,4) = '.OLD' Then Do
  If i = No_Objs Then /* if this is the last object dataset */
    Call Close_File_Tailoring
  Return
End

/* Write member for previous object's dataset(s) to disk */
Parse Var Sys_Obj.i 'DSNDBO' . '.' Item '.' .
If Item ≠ Member_Name Then
  Call Close_File_Tailoring
/* Determine if resizing is required for this object */
High_Track = (SpacType = "TRACK") & (SpacPri > 14)
/* UsedPerc = 100 * Hi_Used_RBA / Hi_Alloc_RBA */
Used_Space = Hi_Used_RBA / 4096 / ci_ca * tracks_ca /* tracks */
To_Be_Resized = Force_Indicated | (NumExt > 1) | High_Track |,
(UsedPerc >= Min_Used & UsedPerc <= Max_Used &,
  Used_Space < Max_Prim_Trk)
If To_Be_Resized Then Do
  Space = Calc_Space() /* Calculate existing allocation in tracks */
  If SpacType = "TRACK"
    Then AlSpace = Space
    Else AlSpace = Space / 15 /* total allocated space in CYL */
  Resize = Space * UsedPerc * (100 + Increase)
  Ind = Resize // 10000 /* remainder after division */
  Resize = Resize % 10000 /* integer result of division */
  If Ind > 100 ThenResize = Resize + 1 /* Always round up */
  If Resize > Max_Prim_Trk Then New_Dset_Needed = 1
  Else New_Dset_Needed = 0
  Resize = Min(Resize,Max_Prim_Trk) /* enforce 2 or 4 GB limit */
/* If any of the new objects are going to be allocating more than 14
tracks, then we'll switch to cylinder allocation */
Pri = Resize
Type = "TRACK"
If Resize > 14 Then Do /* convert to cylinder allocation */
Pri = Resize % 15
If Resize // 15 > Ø Then Pri = Pri + 1
Type = "CYLINDER"
End
Sec = Pri * Sec_Perc % 100 /* secondary space default is 10% */
If Pri * Sec_Perc // 100 > Ø Then Sec = Sec + 1 /* rounded up */
Pri = Min(Pri, Max_Prim) /* reduce Pri iff over the limit */
If Type = "CYLINDER"
Then multi = 15
Else multi = 1
Pri_Trks = Pri * multi /* primary allocation in tracks */
Sec_Trks = Sec * multi /* secondary allocation in tracks */
/* Calculate the old space used */
If SpacType = "CYLINDER"
Then multi = 15 /* multiplier is 15 for cylinder alloc */
Else multi = 1
Used_Units = Used_Space % multi
If Used_Space // multi > Ø Then Used_Units = Used_Units + 1
UsedPerc = Format(UsedPerc,,Ø) /* round it off for reporting */
/* Calculate the new percentage used */
Sec_Exts = Ø /* should all be allocated in primary extent */
NewPerc = 100 * Used_Space / Pri_Trks
If NewPerc > 100 Then Do /* didn't all fit in primary ext */
Req_Sec_Trks = Used_Space / Pri_Trks
Sec_Exts = Req_Sec_Trks % Sec_Trks
If Req_Sec_Trks // Sec_Trks > Ø Then Sec_Exts = Sec_Exts + 1
Total_Trks = Min(Max_Prim_Trk, Pri_Trks + Sec_Trks * Sec_Exts)
NewPerc = 100 * Used_Space / Total_Trks
End
NewPerc = Format(NewPerc,,Ø)
NewExts = Sec_Exts + 1
/* Report the information relating to this object resize */
Say "Generating new sizing for" Sys_Obj.i
Say "Old size info - Units :" Left(SpacType,8),
" Alloc :" Left(AlSpace,4),
" Used :" Used_Units
Say "Pri Alloc :" Left(SpacPri,4),
" Volume :" Volser
Say "Sec Alloc :" SpacSec
Say "Extents :" Left(NumExt,4),
" Used % :" UsedPerc
/* If the new allocation size is the same as before - don't re-alloc */
If Type = SpacType & Pri = SpacPri & NumExt =~ NewExts Then Do
Say "New size same as old, so no re-allocation JCL created"
Say Left("",79,"")
If i = No_Objs Then /* if this is the last object dataset */
Call Close_File_Tailoring
Return
End
Say "New size info - Units :" Type
Say " Pri Alloc :" Pri
Say " Sec Alloc :" Sec
Say " Extents :" Left(NewExts,3),
       " New % :" NewPerc

Say Left("",79,"_")

/* Tell the skeleton that performs dataset initialisation, that this system object should be included */
Obj = Sys_Obj.i
Parse Var Obj 'DSNDBO' . '.' Item '.'.
Interpret Item["" = 'Y'

/* We have the cluster component name, we must generate the data component dataset name */
T_Sw = Index(Obj,"DSNDBC") + 4
T_Ln = Length(Obj)
ObjD = SubStr(Obj,1,T_Sw)||"D"||,
       SubStr(Obj,(T_Sw + 2),(T_Ln - (T_Sw + 1)))
L_HLQ = Length(HLQ)
DsNum = Right(Obj,1) /* last char of dataset */

/* Generate the dataset names to be used for renaming the original catalog/directory datasets */
NObj = HLQ||SubStr(Obj,(L_HLQ + 1),(T_Ln - L_HLQ))||".OLD"
NObjD = HLQ||SubStr(ObjD,(L_HLQ + 1),(T_Ln - L_HLQ))||".OLD"

/* Build job card information */
Address IspExec "VGet ZACCTNUM Shared"
Acct = ZACCTNUM
Job_No = Right(Job_No + 1,3,"Ø") /* add 1 to Job_No */
User = Userid() /* get current user's id */
JobNo_Length = 8 - Length(User)
Job = Right(Job_No,JobNo_Length,"Ø") /* number for jobcard */

/* Split volume assignments by whether the item in question is a tablescape or an index */
ISATS = Is_A_Tablespace(Item)

/* DSN1COPY pgm (specified in the DB2SIZE2 skeleton) needs */
a) PARM='LOB' for LOB tablespaces
b) PARM='SEGMENT' for segmented spaces (-> no 'zero page' msgs) */
If ISATS Then Do /* Tablespaces */
   ISALOB = Is_A_LOB(Item)
   ISASEG = Is_A_Segment(Item)
End
Else Do /* Indexes */
   ISALOB = Ø
   ISASEG = Ø
End

/* Keep track of how much DASD we will need, and the increase in size */
DASD = DASD + Resize
Delta = Delta + Resize - Space /* ie the increase */
If ISATS = 1 then
   DAT_Delta = DAT_Delta + Resize - Space
If ISATS = Ø then
   IND_Delta = IND_Delta + Resize - Space
If Resize > Largest_Size Then Do
   Largest_Size = Resize
   Largest_Name = Obj
End

/* Build the IDCAMS control cards to define this object */
If Item ≠ Member_Name Then Do
   Member_Name = Item
   Address lspExec "FTOpen"
   Address lspExec "FTIncl DB2SIZE1" /* Jobcard */
End
Address lspExec "FTIncl DB2SIZE2" /* JCL steps */
/* Write the last member to disk */
If i = No_Obs Then /* only if this is the last object dataset */
   Call Close_File_Tailoring
End /* End of resizing processing for this item */
Return

/* Finish file tailoring for the last member, writing the job to that
member in the output dataset with DDNAME = ISPFILE */
Close_File_Tailoring:
If New_Dset_Needed Then Do
   DsNum = Right(Obj,1) + 1 /* last char of prev d'set plus 1 */
   T_Ln = Length(Obj)
   NewObj = Left(Obj,T_Ln - 1) || DsNum
   T_Sw = Index(NewObj,"DSNDBC") + 4
   NewObjd = Left(NewObj,T_Sw)||'D'||Substr(NewObj,T_Sw +2)
   Address lspExec "FTIncl DB2SIZE3" /* add new dataset */
   NewTrks = NewPrim * 15 /* NewPrim = primary cyls from DB2SIZE3 */
   DASD = DASD + NewTrks
   Delta = Delta + NewTrks /* ie the increase */
   If ISATS = 1 then
      DAT_Delta = DAT_Delta + NewTrks
   If ISATS = Ø then
      IND_Delta = IND_Delta + NewTrks
   /* Report the information relating to this object */
   Say "Allocating extra dataset" NewObj
   Say " New size info - Units :" Type
   Say " Pri Alloc :" Left(NewPrim,4),
       " Volume :" Vols
   Say " Sec Alloc :" NewSec
   Say " Extents : 1",
       " New % : Ø"
   Say Left("","",79,"_")
End
If Member_Name ≠ 'MEMBER_NAME' Then
   Address lspExec "FTClose Name("Member_Name")"
Return
/* Resizing processing has been performed for all of the individual
items, now produce a completion report and generate the job */
Generate_Complete:
Say ""
Say "Completion Report:" 
Say "Space required to allocate new objects is" DASD "tracks"
C_DASD = DASD % 15
If DASD // 15 > Ø Then C_DASD = C_DASD + 1
Say " or" C_DASD "Cylinders"
Say "Largest dataset to be reallocated is" Largest_Name
If Largest_Size > 14 Then Do
    C_Sp = Largest_Size % 15
    If Largest_Size // 15 > Ø Then C_Sp = C_Sp + 1
    Say "Space required to allocate this is" C_Sp "Cylinders"
End
Else Say "Space required to allocate this is" Largest_Size "tracks"
Say "Additional new space required is" Delta "tracks"
C_Delta = Delta % 15
If Delta // 15 > Ø Then C_Delta = C_Delta + 1
Say " or" C_Delta "Cylinders"
Say DAT_Delta "tracks for Tablespaces"
Say IND_Delta "tracks for Indexes"
Say "Return"

/* The following function returns True (1) if the item passed is a catalog or directory tablespace, and False (Ø) if it is an index */
Is_A_Tablespace: Procedure
Arg Item
Return (Item = "DBDØ1") | (Item = "SCTØ2") |,
      (Item = "SPTØ1") | (Item = "SYSLGRNG") |,
      (Item = "SYSUTIL") | (Item = "SYSLGRNX") |,
      (Item = "SYSCPY") | (Item = "SYSDBASE") |,
      (Item = "SYSPBTAUT") | (Item = "SYSPBAUT") |,
      (Item = "SYSGROUP") | (Item = "SYSPKAGE") |,
      (Item = "SYSSPLAN") | (Item = "SYSSTATS") |,
      (Item = "SYSSTR") | (Item = "SYSUSER") |,
      (Item = "SYSVIWES") | (Item = "SYSLGRNX") |,
      (Item = "SYSSDDF") | (Item = "SYSOBJ") |,
      (Item = "SYSSSEQ") | (Item = "SYSSSEQ2") |,
      (Item = "DSNPSM") | (Item = "SYSGRTNS") |,
      (Item = "SYSHIST") | (Item = "SYSJAVA") |,
      (Item = "SYSJAUXA") | (Item = "SYSJAUXB")

/* This function returns True (1) if item is a LOB or False (Ø) if not */
Is_A_LOB: Procedure
Arg Item
Return (Item = "SYSJAUXA") | (Item = "SYSJAUXB")

/* This returns True (1) for segmented tablespace, or False (Ø) if not */
Is_A_Segment: Procedure
Arg Item
Return (Item = "SYSSDDF") | (Item = "SYSGRTNS") |,
/* This function is used to calculate the size of an existing dataset in tracks. The variables used are those that are exposed... */

Calc_Space: Procedure Expose Hi_Alloc_RBA ci_ca tracks_ca
Return Hi_Alloc_RBA / 4096 / ci_ca * tracks_ca

/* This procedure builds an individual dataset load record */

Load_Opts:
Say "Build LOAD data for" Sys_Obj.i
DB_Str = Translate(Sys_Obj.i, "", "")
TS_Str = Left(Word(DB_Str, 4), 8)
DB_Str = Left(Word(DB_Str, 3), 8)
If Is_A_Tablespace(TS_Str) Then Obj_Type = "T"
Else Obj_Type = "I"

Al_Tp = SubStr(SpacType, 1, 1)
Al_Pr = Right(D2C(SpacPri), 4, X2C(00))
Al_Sc = Right(D2C(SpacSec), 4, X2C(00))
Xtnts = Right(D2C(NumExt), 2, X2C(00))
Space = Right(D2C(Calc_Space()), 4, X2C(00))
Op_Str = DB_Str || TS_Str || Obj_Type || Al_Tp || Al_Pr || Al_Sc, || Xtnts || Space || Load_TS
Queue Op_Str
Return

/* Produce completion report for the LOAD data build process */

Load_Complete:
Say "Build of LOAD data complete."
Airx = Queued()
Queue
Address TSO "ExecIO * DiskW SYSREC00 (Finis"
Say Airx "records fed to SYSREC00"
DelStack

Return

/* Produce an entry in the report for a cat/dir object */

Report_Opts:
Say "Report on" Sys_Obj.i
Space = Calc_Space()
Say "Current allocation status:"
Select
  When SpacType = "TRACK" Then Sp_Str = "tracks"
  When SpacType = "CYLINDER" Then Sp_Str = "cylinders"
  Otherwise Sp_Str = "unknown units - "||SpacType
End
Say "Allocation is in" Sp_Str".",
  "Primary:" SpacPri, "Secondary:" SpacSec
Say "Current extent count is" NumExt,
  "and Used %:" Format(UsedPerc,,0)
Say Left(",", 79, "_")
DASD = DASD + Space
If Space > Largest_Size Then Do
    Largest_Size = Space
    Largest_Name = Sys_Obj.i
End
Return
/* Reporting has been performed for all of the individual items, now produce a completion report */
Report_Complete:
Say ""
T_S = "Totals For DB2 Catalog And Directory ("HLQ")"
L_S = Length(T_S)
Say T_S
Say Left("",L_S,"=")
Say ""
Say "Total space in use by system objects is" DASD "tracks"
C_DASD = DASD % 15
If DASD // 15 > Ø Then C_DASD = C_DASD + 1
Say "or" C_DASD "cylinders"
Say ""
Say "Largest dataset within this is" Largest_Name
If Largest_Size > 14 Then Do
    C_Sp = Largest_Size % 15
    If Largest_Size // 15 > Ø Then C_Sp = C_Sp + 1
    Say "Space required to allocate this is" C_Sp "cylinders"
End
Else Say "Space required to allocate this is" Largest_Size "tracks"
Say ""
Return
/* Process the arguments supplied in the SYSIN ddname. These are */
/* MODE=GENERATE|REPORT|LOADINF */
/* HLQ=<catalog and directory HLQ> */
/* TSVOLS=<volsers for tablespaces> */
/* IXVOLS=<volsers for indexes> */
/* FORCE=ON|OFF */
/* INCREASE=<% increase in size> */
/* SECPERC=<secondary allocation % of used space> */
/* MAXUSED=<maximum % used> */
/* MINUSED=<minimum % used> */
/* MAXSIZE=<maximum dataset size> */
/* MAXPRIM=<maximum primary space allocation> */
/* */
Process_Parms:
/* Set up new stack, and go and get the parameter file */
NewStack
Address TSO "ExecIO * DiskR SYSIN (Finis"
/* Establish the default values for incoming parameters */
Mode = "REPORT"
HLQ = "DB2B"
TSVols = "XXXXXX"
IxVols = "YYYYYY"

Force_Indicated = Ø
Increase = 25
Sec_Perc = 10
Max_Used = 70
Min_Used = Ø1
Max_Size = 2       /* either 2 or 4 GB */
Max_Prim = 2913    /* 2 GB = 43695 trks = 2913 cyls */
/*/ Sift through records and establish parameters... */
Do While Queued() > Ø
   Pull Parm_Rec
   If Index(Parm_Rec,"=") > 1 Then Do
      Parm_Rec = Translate(Parm_Rec," ","=")
      Select
         When SubStr(Parm_Rec,1,1) = "*" Then Nop /* i.e. Comment */
         When Word(Parm_Rec,1) = "MODE" Then Call Set_Mode(Parm_Rec)
         When Word(Parm_Rec,1) = "HLQ" Then Call Set_HLQ(Parm_Rec)
         When Word(Parm_Rec,1) = "TSVOLS" Then
            Call Set_TS_Vols(Parm_Rec)
         When Word(Parm_Rec,1) = "IXVOLS" Then
            Call Set_IX_Vols(Parm_Rec)
         When Word(Parm_Rec,1) = "FORCE" Then
            Call Set_Force(Parm_Rec)
         When Word(Parm_Rec,1) = "INCREASE" Then
            Call Set_Increase(Parm_Rec)
         When Word(Parm_Rec,1) = "SECPERC" Then
            Call Set_Sec_Perc(Parm_Rec)
         When Word(Parm_Rec,1) = "MAXUSED" Then
            Call Set_Max_Used(Parm_Rec)
         When Word(Parm_Rec,1) = "MINUSED" Then
            Call Set_Min_Used(Parm_Rec)
         When Word(Parm_Rec,1) = "MAXSIZE" Then
            Call Set_Max_Size(Parm_Rec)
         When Word(Parm_Rec,1) = "MAXPRIM" Then
            Call Set_Max_Prim(Parm_Rec)
         Otherwise Call Illegal_Keyword(Parm_Rec)
      End
   End
End
Del Stack
bndry = ""
t_item = "*="
Do i = 1 to 39
   bndry = bndry||t_item
End
bndry = bndry||"**"
Say bndry
Say "Options selected from parameters and defaults:" 
Say "Established Mode :" Mode
Say "High Level Qualifier :" HLQ
Say "Tablespace Volumes :" TsVols
Say "Indexspace Volumes   :" IxVols
If Force_Indicated Then Say "Setting of Force   : YES"
   Else Say "Setting of Force   : NO"
Say "% Increase           :" Increase
Say "Sec % Allocation     :" Sec_Perc
Say "Min % Used           :" Min_Used
Say "Max % Used           :" Max_Used
Say "Max Space Size (gb)  :" Max_Size
Say "Max Prim Space (cyl) :" Max_Prim
Say bndry
Return
Set_Mode:
   Arg Parms
   Mode = Word(Parms,2)
   Select
      When Mode = "GENERATE" Then Nop
      When Mode = "LOADINF"  Then Nop
      When Mode = "REPORT"   Then Nop
      Otherwise Do
         Say "Mode ("Mode") is invalid - ending"
         Exit 12
      End
   End
Return
Set_HLQ:
   Arg Parms
   HLQ = Word(Parms,2)
Return
Set_TS_Vols:
   Arg Parms
   TsVols = Word(Parms,2)
   If Queued() > Ø Then Do
      Pull Parms
      Do While (Index(Parms,"=") < 1) & (SubStr(Parms,1,1) ≠ "*")
         TsVols = TsVols||" "||Word(Parms,1)
      If Queued() = Ø Then Leave
         Else Pull Parms
   End
   If Index(Parms,"=") > Ø Then Push Parms
Return
Set_IX_Vols:
   Arg Parms
   IxVols = Word(Parms,2)
   If Queued() > Ø Then Do
      Pull Parms
      Do While (Index(Parms,"=") < 1) & (SubStr(Parms,1,1) ≠ "*")
         IxVols = IxVols||" "||Word(Parms,1)
      If Queued() = Ø Then Leave
         Else Pull Parms
   End
End
   If Index(Parms, "=") > Ø Then Push Parms
End
Return
Set_Force:
   Arg Parms
   Select
      When Word(Parms, 2) = "ON" Then Force_Indicated = 1
      When Word(Parms, 2) = "OFF" Then Force_Indicated = Ø
      Otherwise Call Illegal_Keyword(Parms)
   End
Return
Set_Increase:
   Arg Parms
   t_inc = Word(Parms, 2)
   If DataType(t_inc) ≠ "NUM" Then Call Illegal_Keyword(Parms)
   Increase = t_inc
Return
Set_Sec_Perc:
   Arg Parms
   t_sec = Word(Parms, 2)
   If DataType(t_sec) ≠ "NUM" Then Call Illegal_Keyword(Parms)
   Sec_Perc = t_sec
Return
Set_Max_Used:
   Arg Parms
   t_max = Word(Parms, 2)
   If DataType(t_max) ≠ "NUM" Then Call Illegal_Keyword(Parms)
   Max_Used = t_max
Return
Set_Min_Used:
   Arg Parms
   t_min = Word(Parms, 2)
   If DataType(t_min) ≠ "NUM" Then Call Illegal_Keyword(Parms)
   Min_Used = t_min
Return
Set_Max_Size:
   Arg Parms
   t_max = Word(Parms, 2)
   If DataType(t_max) ≠ "NUM" Then Call Illegal_Keyword(Parms)
   Max_Size = t_max  
      /* should be either 2 or 4 (GB) */
   If Max_Size = 2 Then Do  
      /* either 2 or 4 GB */
      Max_Prim_Cyl = 2913
      Max_Prim_Trk = 43695
   End
   Else Do  
      /* Max_Size = 4 */
      Max_Prim_Cyl = 5826
      Max_Prim_Trk = 87390
   End
Return
Set_Max_Prim:
Arg Parms
  t_max = Word(Parms, 2)
  If DataType(t_max) ≠ "NUM" Then Call Illegal_Keyword(Parms)
  Max_Prim = Min(t_max, Max_Prim_Cyl) /* enforce 2 or 4 GB limit */
Return

Illegal_Keyword:
Arg Err_Parms
Say Left("", 79, "*")
Say "Error with the following input record..."
Say Err_Parms
Say Left("", 79, "*")
Exit 12
Return

DB2SIZE1 SKELETON

CM +---------------------------------+
| This is used by DB2SIZE EXEC to resize the Catalog & Directory |
| This skeleton is only a job card (using &USER and &JBNO values |
| from the DB2SIZE EXEC). DB2SIZE2 (& DB2SIZE3?) will follow. |
+---------------------------------+
// &USER. &JBNO JOB (&ACCT), 'REBUILD', CLASS=A, MSGCLASS=R,
// NOTIFY=&&SYSUID, TIME=1439
//*
CM /*JOBPARM SYSAFF=sysid JES2 control card
CM /*MAIN CLASS=class JES3 control card

DB2SIZE2 SKELETON

CM +---------------------------------+
| This is used by DB2SIZE EXEC to resize the Catalog & Directory |
| This skeleton DB2SIZE2 is used to rename an existing Catalog or |
| Directory VSAM dataset, create a new (resized) dataset and copy |
| the data from the old to the new via DSN1COPY program, then |
| finally delete the old dataset. |
+---------------------------------+
SET VO = &VOLSER
SET SPT = &SPACTYPE
SET SPP = &SPACPRI
SET SPS = &SPACSEC
SET NUE = &NUMEXT
SET NNE = &NEWEXTS
SEL &ISATS = 1
SET VOLS = &TSVOLS
ENDSel
) SEL &ISATS = Ø
) SET VOLS = &IXVOLS
) ENDSel
/*====================================================================*/
/* REALLOCATE '&OBJ'
/*====================================================================*/
) SEL &NUMEXT = 1
/* OLD: &TYPE(&SPP,&SPS) &USEDPERC.% Used from &ALSPACE allocated
/* in 1 Extent, on &VO
) ENDSel
) SEL &NUMEXT > 1
/* OLD: &SPT(&SPP,&SPS) &USEDPERC.% Used from &ALSPACE allocated
/* in &NUE Extents, on &VO
) ENDSel
) SEL &NEWEXTS = 1
/* NEW: &TYPE(&PRI,&SEC) &NEWPERC.% Used
/* in 1 Extent, on &VOLS
) ENDSel
) SEL &NEWEXTS > 1
/* NEW: &TYPE(&PRI,&SEC) &NEWPERC.% Used
/* in &NNE Extents, on &VOLS
) ENDSel
/*====================================================================*/
//DEFINE&DSNUM EXEC PGM=IDCAMS,COND=(Ø,LT)
//SYSPRINT DD SYSOUT=* 
//SYSIN DD *
ALTER '&OBJ' -
   NEWNAME(&NOBJ) -
ALTER '&OBJD' -
   NEWNAME(&NOBJD) -
DEFINE CLUSTER -
   ( NAME(&OBJ) -
      LINEAR -
      REUSE -
      VOLUMES(&VOLS) -
      &TYPE(&PRI &SEC)
      SHAREOPTIONS(3 3)
   ) -
DATA -
   ( NAME(&OBJD) -

/*====================================================================*/
) SEL &ISASEG = 1
//COPY&DSNUM EXEC PGM=DSN1COPY, PARM='SEGMENT', COND=(Ø,LT)
) ENDSel
) SEL &ISALOB = 1
//COPY&DSNUM EXEC PGM=DSN1COPY, PARM='LOB', COND=(Ø,LT)
) ENDSel
) SEL &ISASEG = 0 && &ISALOB = 0
//COPY&DSNUM EXEC PGM=DSN1COPY, COND=(Ø,LT)
//SYSPRINT DD  SYSOUT=*  
//SYSUT1 DD  DISP=MOD,                      (MOD REQUIRED FOR JES3)  
//   DSN=&NOBJ,  
//   AMP=('BUFND=20')  
//SYSUT2 DD  DISP=OLD,  
//   DSN=&OBJ,  
//   AMP=('BUFND=20')  
//*--------------------------------------------------------------------
//* DELETE&DSNUM  EXEC PGM=IDCAMS,COND=(Ø,LT)  
//*--------------------------------------------------------------------
//* SYSPRINT DD  SYSOUT=*  
//*--------------------------------------------------------------------
//* SYSIN DD  *  
//*--------------------------------------------------------------------
//* DELETE &NOBJ  
//*--------------------------------------------------------------------

DB2SIZE3 SKELETON

)CM +--------------------------------------------------------------------+
)CM | This is used by DB2SIZE EXEC to resize the Catalog & Directory |
)CM |                                                                 |
)CM | This skeleton DB2SIZE3 is used to create a new VSAM dataset to       |
)CM | be used if the existing dataset expands to the maximum size         |
)CM | allowed (2gb or 4gb), then DB2 would start to use this new one.     |
)CM +--------------------------------------------------------------------+
)CM | set new primary & secondary allocation size (cylinders)              |
)CM | SET NEWPRIM = 500  
)CM | SET NEWSEC = 500  
)CM | SEL &ISATS = 1  
)CM | SET VOLS = &TSVOLS  
)CM | ENDSEL  
)CM | SEL &ISATS = Ø  
)CM | SET VOLS = &IXVOLS  
)CM | ENDSEL  
)CM +--------------------------------------------------------------------+
)CM | ALLOCATE '&NEWOBJ'  
)CM +--------------------------------------------------------------------+
)CM | NEW: &TYPE(&NEWPRIM,&NEWSEC) Ø% Used  
)CM | in 1 Extent, on &VOLS  
)CM +--------------------------------------------------------------------+
)CM | DEFINE&DSNUM  EXEC PGM=IDCAMS,COND=(Ø,LT)  
)CM +--------------------------------------------------------------------+
)CM | SYSPRINT DD  SYSOUT=*  
)CM | SYSIN DD  *  
)CM | DEFINE CLUSTER  
)CM |   ( NAME('&NEWOBJ') -  
)CM |   LINEAR -  
)CM |   REUSE -  
)CM |   VOLUMES('&VOLS') -  
)CM |   CYLINDER('&NEWPRIM &NEWSEC') -  
)CM |   SHAREOPTIONS(3 3) ) -  

DATA

( NAME( &NEWOBJD ) )

/**

USAGE
To implement the resizing, do the following:
1 Run the batch job to generate the JCL library.
2 Review the generated jobs.
3 Shut down the DB2 system(s).
4 Run the resizing jobs (and check that they all run OK).
5 Restart the DB2 system(s).
And you can run it any time in REPORT or GENERATE mode to simply check whether a DB2 system needs resizing.

Ron Brown
Principal Consultant
Triton Consulting (Germany)  © Xephon 2003
Ascential Software has announced full support for IBM’s new DB2 Cube Views software through its Ascential Enterprise Integration Suite, which combines with Cube Views to help accelerate the development and, says the vendor, lower the total cost of ownership for OLAP and business intelligence applications by making them easier to deploy and manage.

This new capability works with DB2 Cube Views’ dimensional modelling to provide meta data management of information distributed throughout the enterprise.

The company has developed a bi-directional MetaBroker interface between MetaStage, the meta data management component of the Ascential Enterprise Integration Suite, and DB2 Cube Views. MetaStage provides a view of the data’s descriptive information and lineage, including the data being modelled with Cube Views.

For further information contact:
Ascential Software, 50 Washington Street, Westboro, MA 01581, USA.
Tel: (508) 366 3888.

UDF Solution AG has announced CARES (Comprehensive Analysis and Reporting System) for DB2.
CARES assists DBAs to keep their DB2 subsystems in top condition. CARES conducts many of the daily routine checks necessary for preventive maintenance. CARES identifies, reports, and, together with UBS KeyTools DB2ICF and CATEX, enables users to automatically eliminate latent inhibitors in the area of CPU and I/O performance, concurrency, space savings, Parallel Sysplex, etc.

For further information contact
KeyTools, IT Systems Engineering, UBS AG, Hochstrasse 16, 4002 Basel, Switzerland.
Tel: +41 61 271 65 50.
URL: http://www.ufd.ch/index.html.

It captures within a single repository all metadata associated with electronic assets, processes, issues, and relationships across mainframe and distributed platforms. Authorized users can view and manage assets, processes, and issues from anywhere in the organization.

The newly announced Dimensions 8 includes DB2 and Oracle support, integration with Visual Studio .NET 2003, improved client usability, enhanced administration tools, and integration with Merant’s process asset library.

For further information contact:
Merant, 3445 NW 211th Terrace, Hillsboro, OR 97124, USA.
Tel: (503) 645 1150.
URL: http://www.merant.com/Products/ECM/dimensions/home.asp.