94 DB2

August 2000

In this issue

update

DB2 Update

Published by

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

North American office

Xephon
PO Box 350100
Westminster, CO 80035-0100
USA
Telephone: 303 410 9344

Subscriptions and back-issues

A year's subscription to *DB2 Update*, comprising twelve monthly issues, costs £255.00 in the UK; $380.00 in the USA and Canada; £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 1997 issue, are available separately to subscribers for £22.50 ($33.50) each including postage.

DB2 Update on-line

Code from *DB2 Update* can be downloaded from our Web site at http://www.xephon.com/db2update.html; you will need the user-id shown on your address label.

Editor

Trevor Eddolls

Disclaimer

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

Contributions

Articles published in *DB2 Update* are paid for at the rate of £170 ($260) per 1000 words and £100 ($160) per 100 lines of code for the first 200 lines of original material. The remaining code is paid for at the rate of £50 ($80) per 100 lines. In addition, there is a flat fee of £30 ($50) per article. To find out more about contributing an article, without any obligation, please contact us at any of the addresses above and we will send you a copy of our *Notes for Contributors*, or you can download a copy from www.xephon.com/contnote.html.

Printed in England.

Enabling your Linux workstation to run DB2 Version 6.1

The release of DB2 Universal Database Version 6.1 for Linux (from now on referred to as DB2 for Linux) brought a buzz to the Linux community worldwide. IT professionals around the world, already familiar with the benefits and robustness of the Linux operating system, now have the opportunity to run the world's most powerful database on it. Those familiar with the benefits and accolades of DB2, now have a chance to run it on one of the world's most respected operating systems.

Most of the DB2 for Linux marketing is aimed at the big guns in the game – support for Red Hat, Caldera, Turbo Linux, and SuSE Linux are often referred to in various advertisements. The truth is that DB2 for Linux will run on any distributions that adhere to the following requirements:

- Linux kernel 2.0.35 or higher.
- RPM (Red Hat package manager).
- pdksh package (public domain Korn shell).
- glibc Version 2.0.7 or higher.
- libstdc++ Version 2.8.0 (notice that this does not say 'or higher').

Despite the simplicity of the requirements, getting DB2 for Linux to run on a Linux-based workstation can sometimes be a painful and time-consuming task. Scanning various newsgroups, and responding to endless questions, I decided to write this article to help you get DB2 for Linux installed on your Linux-based workstation in a minimal amount of time, along with minimal frustration. If you are planning to install DB2 for Linux on a Linux-based workstation, save yourself time and potential frustrations by reading this article. As the Linux operating system matures and relationships between both the various Linux vendors and the companies that write applications for them strengthen, you should expect to see many of the frustrations associated with this installation disappear.

In this article I will show you how to avoid the pitfalls that many have spent hours trying to resolve or figure out, along with some time-saving techniques, and the location of the packages that you may need for your particular distribution.

You should bear in mind when you read this article that the Linux world is changing fast, so you may find that some of these URLs have become outdated or the levels of the distribution have changed. You can use the information here to install a DB2 Server or a DB2 Client on the Linux distributions described in this article.

DB2 AND CALDERA OPEN LINUX

This section covers the workarounds and steps that you need to perform on a workstation that is running Caldera Open Linux (or simply Caldera) Version 2.2 and Version 2.3. If you are running Caldera Version 2.2, I suggest that you upgrade to Caldera Version 2.3 or later, it makes the installation a lot easier.

The pdksh package that you require for DB2 is missing from the default Caldera Version 2.3 installation. The package is available from the Caldera Version 2.3 CD-ROM, but you will find that it does not work with DB2. Caldera claims to have fixed this problem with a downloadable pdksh package available from their FTP site at ftp.calderasystems.com/pub/. I have not had a chance to check out their latest version of the pdksh package, but if you cannot get Caldera's new pdksh package to work with the DB2 installer program, you can simply use the pdksh package that is available from Red Hat. After all, despite the distribution, it is Linux, right? A good Red Hat mirror site to check out is the MetaLab site (the old Sun site) at www.metalab.unc.edu/pub/Linux/distributions/redhat/redhat-6.1/i386/RedHat/RPMS/. Download the pdksh package and install it with the rpm command using the -nodeps option. You have to use this option if you are going to install Red Hat's pdksh package on a Caldera-based workstation. If you try to install it without this option, you will receive an error stating that this package has a requirement for the glibc package. The error that you receive is only a result of each vendor's naming conventions beneath the covers. By default, the

glibc package is installed on any Caldera installation, so don't worry about using this option.

If you list all of the rpms installed on your Caldera-based system, using the rpm -qa command, you will see that Caldera Version 2.3 installs libstdc++ 2.9.0. If you recall, DB2 or Linux will run only with libstdc++ 2.80. This doesn't mean that you cannot have libstdc++ 2.9.0 installed on your distribution, it just means that you have to also install libstdc++ 2.8.0. You can find the required libstdc++ 2.8.0 library in the /col/contrib/RPMS directory on the Caldera Version 2.3 CD-ROM. This package is called libstdc++-compat-2.8.0-1.i386.rpm.

Once you have take care of these requirements, your Caldera Version 2.3 workstation is ready for a DB2 installation.

ADDITIONAL CONSIDERATIONS FOR CALDERA VERSION 2.2

Again, if you are using Caldera Version 2.2, I recommend that you upgrade to Version 2.3, it will make your DB2 installation a lot easier. If you don't want to do this, or you cannot, there are some other issues that you need to be aware of.

Remember the required libstdc++ 2.8.0 requirement? Well, it isn't part of the Caldera Version 2.2 installation and you cannot get it from the CD-ROM either. You can get it from Caldera's ftp site, however, at ftp://ftp.calderasystems.com/pub/openlinux/2.3/contrib/RPMS/.

Finally, DB2 requires the libcrypt.so.1 library to work. I didn't mention this in the requirements section because for every other Linux distribution (including Caldera Version 2.3 or higher) you do not have to worry about it. From what I understand, Caldera ran into some problems with Federal export laws that resulted in Caldera's Version 2.2 shipping without this file. You can get this required library from the Linux Land ftp site at ftp://ftp.linuxland.de/pub/OpenLinux/crypto/2.2/RPMS/. I looked for this library on Caldera's ftp site, but I could not find it. I assume that since it now ships with Caldera, and Caldera Version 2.2 has been generally available for almost one year (which is a lifetime in the Linux world), that it isn't deemed necessary. If you want to run DB2 for Linux on Caldera Version 2.2, it is!

Once you have taken care of the Caldera Version 2.2 considerations, your workstation is ready for a DB2 installation.

DB2 AND TURBO LINUX

Getting DB2 for Linux to install on a workstation that is running Turbo Linux Version 3.6 can be a real headache. I worked with the Linux development team at IBM to create a fix for the problems that you would normally encounter. Trust me, download it! You can get this fix from IBM's service ftp site at ftp://ftp.software.ibm.com/ps/products/db2/tools/. Locate the fix called tl36_instfix.tar.Z (l is the letter 'L' in case the font is confusing you). With this fix, there is a README, which contains all the information you need to know to implement this fix.

Once you have downloaded and installed the fix for Turbo Linux Version 3.6, you need only add the required pdksh package to your workstation. This package is not part of the default installation, but it is readily available from the Turbo Linux Version 3.6 CD-ROM in the /TurboLinux/RPMS directory.

Once you have completed these tasks, your TurboLinux Version 3.6 workstation is ready for a DB2 installation.

DB2 AND SuSE LINUX

Installing DB2 for Linux on a SuSE Version 6.2 or Version 6.3 workstation is simple, if someone tells you about the 'gotcha' – and that is what I am going to do. SuSE uses a different naming convention for its Linux packages, and this causes some confusion for the DB2 Installer program. For example, the required glibc package is known as shlibs in the SuSE world. Don't ask me why this is, Linux is supposed to be standardized. This proprietary naming convention is anything but standard and painfully reminds me of a once-standardized technology called HTML. None the less, SuSE's naming convention will cause a problem when you try to install DB2 because the DB2 Installer program will fail to recognize the required glibc package and subsequently fail. The fix for this problem is simple. Install a 'dummy' package from the DB2 CD-ROM. This will trick the DB2 Installer

program into thinking that this package is actually installed. Of course, this package is installed, it's just called by a different name and therefore the DB2 Installer program fails to recognize it. The 'dummy' package is called glibc-2.0.7-0.i386.rpm and can be found in the /db2/install/dummyrpm directory on your DB2 product CD-ROM.

Once you have installed this 'dummy' rpm, your SuSE Version 6.2 or Version 6.3 workstation is ready for a DB2 installation.

ADDITIONAL STEPS FOR SuSE VERSION 6.1

SuSE Linux Version 6.1 has another twist to it. SuSE Version 6.1 shipped with DB2 for Linux Version 5.2. This version of Linux was never made generally available, it was a beta copy of the soon-to-be released DB2 for Linux Version 6. Consequently, when you go to install DB2, this causes problems with a default installation. I won't even go into some of the other problems that I found with this bundling of DB2, just remove the DB2 users (db2inst1, db2as, db2fenc1) that were created when you installed the SuSE Version 6.1 operating system. If any DB2 files where installed, remove them as well. You can search for any installed DB2 rpms by entering the rpm -qa | grep db2 command.

Once you have completed these tasks, your SuSE Version 6.1 workstation is ready for a DB2 installation.

DB2 AND RED HAT LINUX

Red Hat Version 5.2 and Version 6.0 are perhaps the easiest of all distribution to enable for DB2. Each version is missing only the required pdksh package. You can find this package on the Red Hat CD-ROM in the /RedHat/RPMS directory.

Once you have installed the pdksh package, your Red Hat Version 5.2 or Version 6.1 workstation is ready for a DB2 installation.

ADDITIONAL STEPS FOR RED HAT 6.1

If you are trying to install DB2 on a workstation that is running Red Hat Version 6.1, you are going to run into problems the moment that

you try to start an instance. If you install DB2 for Linux using the DB2 Installer program, and you decide to create an instance during the installation, your installation is going to fail. Why? By default, the DB2 Installer program sets an instance to auto-start when it creates an instance. Even if you change this default setting, DB2 will still eventually hang when you issue the db2start command, so you might as well fix it before the installation.

If you are planning to install DB2 for Linux on a workstation that is running Red Hat Version 6.1, there is a fix that you can download from ftp://ftp.software.ibm.com/ps/products/db2/tools. There are two fixes that are available for download and the one that you should use depends on where the DB2 code that you are planning to install came from. If you want to install the copy of DB2 that was bundled with the Red Hat Version 6.1 release, you need to download the db2rh61fix.tgz file. This copy of DB2 actually has FixPack 1 integrated into the installation. If you want to install the generally available copy of DB2 for Linux Version 6.1 on Red Hat Version 6.1, you must download the db2rh61gafix.tgz file. Make sure that you download the correct file, you cannot share them.

Once you download the fix that you need for your particular copy of DB2, you need to unzip and untar the file by entering the tar xvzf filename command, where filename is the name of the fix file that you downloaded. When you have unzipped and untarred the fix file, check out the readme.txt file. This file gives complete and detailed instructions on how to implement this fix.

Once you have addressed the fix situation, your Red Hat Version 6.1 workstation is ready for a DB2 installation.

INSTALLING DB2

If you closely followed the instructions that I have presented in this article, you should have no problems getting DB2 to run on any of the distributions outlined in this article. From here, just follow the instructions in the *DB2 Quick Beginning for Unix* book to install DB2 for Linux. If you downloaded your copy of DB2 for Linux from the Web, you can view this book at www.software.ibm.com/cgi-bin/

db2www/library/index.d2w/report. If you have implemented a fix (for Turbo Linux Version 3.6 or Red Hat Version 6.1) you must remember to start the DB2 Installer program where the fix is located on your drive. When you implement either of the fixes, some files are placed on your workstation. These files will interact with the DB2 CD-ROM, so that needs to be mounted as well.

I hope that this article helps you to install DB2 for Linux on your particular Linux distribution. Enjoy using the world's leading database on what some consider the world's best platform!

Paul Zikopoulos
Software Engineer
IBM Canada (Canada)

Hinting and the DB2 optimizer

ABSTRACT

Query optimization features have always been upgraded by incremental releases of DB2. DB2 UDB Version 6 has come up with a very effective technique by which users can give hints to the optimizer. For example, when a static SQL is bound, the access path is established by DB2. But it might not have taken the data pattern into consideration during execution time to influence the access path. This new feature of DB2, can help influence the access path by giving hints to the optimizer.

This is a powerful feature of DB2 , which could be dangerous if not used with a good knowledge of the data pattern and data distribution across instances and partitions.

During the migration of applications, it is recommended that you save the PLAN_TABLE information before proceeding with 'hinting the optimizer'.

In this article, I shall describe one way of 'hinting the DB2 optimizer'. The user of this technique must understand the importance and underlying danger of influencing the DB2 optimizer.

ANALYSIS

Let us analyse the following SQL:

```
EXPLAIN ALL SET QUERYNO=999 FOR
SELECT TACCONT.AC_NA , TACCONT.AC_TYP_CD
FROM TACCONT
WHERE TACCONT.AC_NR  = '123456' AND
TACCONT.CNY_CD = 'US' and
TACCONT.AC_TYP_CD = 'A';
```

Suppose that DB2 would pick up index IACCONT1, which is an index based on AC_NR.

In the event of DB2 picking up the normal index, the PLAN_TABLE entry for the above query would appear as follows:

```
Query # MTH TNAME   TABNO ACCESS MATCH ACCESS   INDEX
                          TYPE    COLS  NAME    ONLY
999      Ø  TACCONT   1      I     Ø1  IACCONT1    Y
```

But, based on our knowledge of the data, we want DB2 to pick up IACCONT2, a secondary index based on CNY_CD and AC_TYP_CD. In earlier versions of DB2 we could not influence the access path, but illustrated below is a feature provided by DB2 Version 6 by which we can give hints to the DB2 optimizer to pick up IACCONT2.

Some new columns to the PLAN_TABLE have been added in DB2 Version 6 to facilitate this feature of DB2. They are OPTHINT and HINT_USED.

Now, let's update the OPTHINT field to name our new access path. Let's call that new plan for accessing the secondary index PICKSECX. We need to update the PLAN_TABLE as follows:

```
UPDATE PLAN_TABLE
SET OPTHINT = 'PICKSECX'
WHERE OPTHINT = '' AND QUERYNO = 999;
```

Now let's update the PLAN_TABLE to indicate the access path for

our new plan,PICKSECX. In this case, you will update the ACCESSNAME column of PLAN_TABLE, with the following SQL statement:

```
UPDATE PLAN_TABLE
SET ACCESSNAME = 'IACCONT2'
WHERE OPTHINT = 'PICKSECX' AND QUERYNO = 999;
```

When updating the PLAN_TABLE for the second time, do make sure you are updating the right row, by predicating the OPTHINT column by the plan – which picks the secondary index.

After updating the PLAN_TABLE, the row of interest to us would look like:

```
Query # MTH TNAME   TABNO ACCESS MATCH ACCESS   INDEX OPT_HINT HINT_USED
                          TYPE    COLS  NAME    ONLY
999      Ø  TACCONT    1     I     Ø1  IACCONT2   Y   PICKSECX
```

Now after setting the hint to the optimizer in the PLAN_TABLE, we should bind our static SQL accordingly.

For static SQL, let's use the following parameter in our bind package or bind plan cards:

```
OPTHINT('PICKSECX').
```

For dynamic SQL, issue a new special DB2 register statement as follows:

```
SET CURRENT QUERY OPTIMIZATION HINT = 'PICKSECX' WHERE QUERYNO=999;
```

And then execute the same SQL as in the first example above as follows:

```
SELECT TACCONT.AC_NA , TACCONT.AC_TYP_CD
FROM TACCONT
WHERE TACCONT.AC_NR  = '123456' AND
TACCONT.CNY_CD = 'US' and
TACCONT.AC_TYP_CD = 'A'
QUERYNO 999;
```

The difference between the two SQL examples is the presence of QUERYNO in the second example.

After execution, the PLAN_TABLE row suggests the following:

```
Query # MTH TNAME   TABNO ACCESS MATCH ACCESS   INDEX OPT_HINT HINT_USED
                          TYPE    COLS  NAME    ONLY
999      Ø  TACCONT   1      I     Ø2  IACCONT2    Y            PICKSECX
```

After executing the second SQL with hints to the optimizer, if we then run an EXPLAIN, PLAN_TABLE shows that plan PICKSECX has moved from column OPT_HINT to HINT_USED. This confirms that DB2 has indeed picked the secondary index for the second SQL example. MATCHCOLS for the second SQL, on picking up the secondary index, IACCONT2, has become 02.

The BIND statement for static SQL and the PREPARE statement for dynamic SQL indicates a SQLCODE of +394, suggesting that the hint was successfully picked up for execution.

If IACCONT2 is a non-existent secondary index, then the BIND statement for static SQL and the PREPARE statement for dynamic SQL indicates a SQLCODE of +395, suggesting that the hint was found, but execution was found to be invalid.

RECOMMENDATIONS

This feature should be used if you are absolutely sure about the data patterns pertaining to the index you want DB2 to pick up. Before rebinding, save copies of your PLAN_TABLE, which you might need if the performance has worsened because of your hint. This concept can be extended to a program where you want to influence a selected number of embedded SQL statements.

No optimizing technique is absolute. But the power of controlling the optimizer has arrived in the hands of users with this new feature.

Hemanta Ranjan Panda
Computer Consultant
PricewaterhouseCoopers (USA)

Advanced SQL for fuzzy logic

My article, *Optimizing Dynamic SQL (DB2 Update*, Issue 93, July 2000), provided tips for optimizing performance when using Dynamic SQL for processing user fuzzy SELECT queries. It suggested using advanced SQL such as CASE, compound SQL, quantified predicate, etc. This article illustrates some usages. The following simplified logical table is used for the examples:

```
STUDENT(STUDENT_ID, SCHOOL_ID, surname, gpa, absence_days, fuzzy_rank,
fuzzy_description)
```

COMPOUND SQL

Compound SQL combines one or more SQL statements into an executable block in an application program. Fuzzy users frequently want more discrimination among groups such as finding the best and worst of the best students. Compound SQL provides better performance because it reduces the DB2 database manager overhead and reduces the number of requests transmitted over the network. The syntax is:

```
>>   BEGIN COMPOUND     ATOMIC          STATIC                          >
                        NOT ATOMIC
>    STOP AFTER FIRST   host-variable   STATEMENT   sql-statements      >

>    END COMPOUND                                                      ><
```

Assume users do want to discriminate between best students, which are defined as having a gpa >91.69 and absence_days <8.88. (See *Fuzzy SELECT, DB2 Update*, Issue 87, January 2000, for an explanation of how the values were calculated.)

```
EXEC SQL
     BEGIN COMPOUND ATOMIC STATIC     --atomic specifies rollback
                                      --on any failure
                                      --static retains input
                                      --variable original value
                                      --no stop after first
     CREATE TABLE      best_student
            LIKE       student

     INSERT INTO       best_student
```

```
            SELECT     student_id, surname, gpa, absence_days
            FROM       student
            WHERE      gpa > 91.69
            AND        absence_days < 8.88

--other sql statements to compute the gpa median and its Gaussian
--distribution as described in Fuzzy SELECT are placed here
--a case statement, as described below, will update fuzzy_description
--from fuzzy_rank

      COMMIT                          --must be last statement

END COMPOUND;
```

Restrictions:

- Can only be embedded in an application program.
- No host language code is allowed within EXEC and END.
- Cannot be nested.
- Cannot be dynamically prepared.
- Not supported by REXX.
- DB2 Connect does not support SELECT LOB columns within a compound SQL.
- DDCS accepts only NOT ATOMIC.

Benefits:

- Reduced database manager overhead.
- Reduced network requests for remote clients.

CASE EXPRESSIONS

CASE allows ‘result-expressions’ to be chosen based on condition evaluation. Typically the case-expression value is the ‘result-expression’ value of the first case that evaluates true. If all cases evaluate false then value is ‘ELSE result-expression’ or NULL. The syntax is:

```
>>CASE        searched-when-clause    ELSE NULL               END    ><
              simple-when-clause      ELSE result-expression
```

```
    |-----------------------------------------------|
>>  |WHEN     search-condition  THEN  result-expression|              ><
                                      NULL

             |--------------------------------------------|
>>expression |WHEN expression    THEN  result-expression|              ><
                                      NULL
```

Any ‘WHERE search-condition’ is allowed in ‘WHEN search-condition’ except quantified predicate, a fullselect IN, or EXISTS.

‘WHEN in expression’ means equality. The first expression must be a data type comparable to all evaluated expressions.

The next example is only the CASE block for updating the fuzzy_description; all cursor processing has been omitted for brevity.

```
CASE
      WHEN   :fuzzy-rank-numeric >= :verylow-low AND <= :verylow-high
            THEN      'very low'
      WHEN   :fuzzy-rank-numeric >= :low-low AND <= :low-high
            THEN      'low'
      WHEN   :fuzzy-rank-numeric >= :belowavg-low AND <=:belowavg-high
            THEN      'below average'
      WHEN   :fuzzy-rank-numeric >= :avg-low AND <= :avg-high
            THEN      'average'
      WHEN   :fuzzy-rank-numeric >= :aboveavg-low AND <=:aboveavg-high
            THEN      'above average'
      WHEN   :fuzzy-rank-numeric >= :high-low AND <= :high-high
            THEN      'high'
      WHEN   :fuzzy-rank-numeric >= :veryhigh-low AND <=:veryhigh-high
            THEN      'very high'
--no else since all domain values checked
END
```

An interesting CASE use is to prevent division by zero. Assume part of the fuzzy_rank equation includes dividing gpa by absence_day. A student with perfect attendance has an absence_day of zero.

```
SELECT       absence_days
FROM         student
WHERE        (CASE WHEN absence_days = 0
             THEN NULL
             ELSE gpa/absence_days
             END)
```

CASE has two scalar functions of NULLIF and COALESCE for the following subsets:

```
CASE WHEN e1=e2 THEN NULL ELSE e1 END          |    NULLIF(e1,e2)
CASE WHEN e1 IS NOT NULL THEN e1 ELSE e2 END   |    COALESCE(e1,e2)
CASE WHEN e1 IS NOT NULL THEN e1 ELSE
      COALESCE(e2,...,eN) END                  |    COALESCE(e1,e2,...eN)
```

QUANTIFIED PREDICATE

Some readers may disagree that quantified predicates are advanced SQL, but they are included here because of their usefulness in fuzzy SELECT. A quantified predicate compares a value or values with a collection of values. The simplified WHERE syntax is:

```
>>WHERE    expression  <|=|>*   SOME      (fullselect)                ><
                               ANY                 --equivalent to some
                               ALL
```

*other comparison operators are supported; '|' is OR

SOME/ANY is true if a specified relationship is true for any returned row.

SOME/ANY is false if no rows are returned or a specified relationship is false for every returned row.

SOME/ANY is unknown if a specified relationship is not true for any row and at least one comparison is unknown because of a NULL value.

ALL is true if no rows are returned or a relationship is true for every returned row.

ALL is false if a specified relationship is false for any returned row.

ALL is unknown if a specified relationship is not false for any returned row and at least one comparison is unknown because of a NULL value.

The following is a repeatable embedded SELECT:

```
SELECT      student_id, surname, fuzzy_rank, gpa
FROM        student
WHERE       fuzzy_rank > ALL
            (SELECT      AVG(fuzzy_rank), AVG(gpa)
            FROM         student)
```

It returns all students whose fuzzy rating and gpa are > AVG (an 'A' student who is absent for many days would not qualify). Changing > to < returns all 'poor' students. Changing > to = would produce no students since fuzzy_rank and gpa are integers and AVG is fractional. Changing ALL to ANY would have no effect because there is only one returned row.

Nested SELECT or correlated SELECT can be used to provide any combination including comparisons across schools using school_id.

SUMMARY

Advanced SQL statements can provide many benefits to fuzzy and standard applications including:

- Compound SQL can appreciably improve performance because:
  - The database manager overhead is reduced.
  - ATOMIC and COMMIT guarantee correctness.
  - It reduces requests transmitted over a network for remote clients.
- CASE provides a simplified construct for evaluating multiple conditions within a single block. Single constructs typically provide faster SELECT response times.
- A quantified predicate provides a simplified construct for evaluating condition subsets that otherwise would require nested expressions, sometimes many. Some say that interpretation is complex, but it is simpler than understanding multiple nests. It usually provides faster SELECT response times.

WHY

Fuzzy SELECT provides users with more relevant answers allowing them to make better decisions that can dramatically and positively affect the bottom line. Fuzzy logic and fuzzy SELECT should be in every enterprise toolbox!

Eric Garrigue Vesely
Principal
Analyst Workbench Consulting (Malaysia)

DB2 statistics report generator

The RUNSTATS on-line utility collects summary information about the characteristics of data in tablespaces, indexes, and partitions. DB2 records this information in the DB2 catalog and uses it to select access paths to data during the bind process. The RUNSTATS utility allows users to generate a report about these characteristics without actually updating them in the catalog. The option used is UPDATE NONE with REPORT YES. This helps a DBA in determining when to run REORG, RUNSTATS (with the UPDATE option), and REBIND programs. However, the output report generated by this option is not in a very readable format, in particular for a partitioned tablespace/ index.

This REXX EXEC takes the SYSPRINT dataset as input (output of RUNSTATS when run with option REPORT YES) and reformats the information into a more readable report. This report helps DBAs to understand and analyse the values like CARDF, COLCARDF, NEARINDREF, FAROFFPOS, etc. In this EXEC, the SYSPRINT dataset used is 'DBM.REXX.STATS.REPORT' and the formatted report is in dataset 'DBM.STATS.INPUT.DATA'. An example RUNSTATS report is shown at the end of this article.

EXEC

```
/*REXX*/
rec1=' '
rec2=' '
first_rec=0
ti_me=''
i=0
j=0
k=0
file_num=0
fileout =  'DBM.REXX.STATS.REPORT'
if sysdsn("'"fileout"'") <> "OK" then
do
  say 'Output file ' fileout ' does not exist.'
  say
  exit
end
"ALLOC DA('"||fileout||"')  F(DATAOUT) OLD REUSE"
```

```
"EXECIO Ø DISKW DATAOUT (OPEN"
call main_rtn
"EXECIO Ø DISKW DATAOUT (FINIS"
"FREE F(DATAOUT)"
exit
main_rtn:
file_char=''
eof='NO'
rec1=' '
rec2=' '
first_ind=Ø
n_pages=''
ts_name=''
perc_change=''
ti_me=''
i=Ø
j=Ø
k=Ø
eof='NO'
INLIST.   = ""
INLIST.Ø  = Ø
VAR.   = ""
VAR.Ø  = Ø
OUTLIST.   = ""
OUTLIST.Ø  = Ø
filename=''
filename='DBM.STATS.INPUT.DATA'
if sysdsn("'"filename"'") <> "OK" then
do
  say 'Input file ' filename ' does not exist.'
  say
  return
end
"ALLOC DA('"||filename||"') F(DATAIN)  SHR REUSE"
"EXECIO Ø DISKR DATAIN  (OPEN"
do while eof='NO'
    "EXECIO 1 DISKR DATAIN (STEM INLIST."
    if RC=2 then
        eof='YES'
    else
        call  process_rtn
end
call process_print
"EXECIO Ø DISKR DATAIN  (FINIS"
"FREE F(DATAIN)"
call dis_play
return
process_rtn:
    a=''
    b=''
    c=''
```

```
    d=''
    e=''
    f=''
    rec1=inlist.1
    rec1=substr(rec1,2)
    a=substr(rec1,1,8)
    if a='DSNUØ5ØI' then
      if first_rec=Ø   then
         do
           first_rec=1
           call process_init
         end
      else
         do
           call dis_play
           call process_init
         end
    if a<>'        '  then
      if first_ind=Ø   then
         first_ind=1
      else
         call process_detail
    select
      when a='DSNU613I' | rec_ind=613  then   call process_tb_part
      when a='DSNU624I' | rec_ind=613  then   call process_tb_part
      when a='DSNU617I' | rec_ind=617  then   call process_ix
      when a='DSNU615I' | rec_ind=615  then   call process_cl
      when a='DSNU618I' | rec_ind=618  then   call process_ix_part
      when a='DSNU627I' | rec_ind=627  then   call process_ix_part
      when a='DSNU612I' | rec_ind=612  then   call process_ts
      when a='DSNU614I' | rec_ind=614  then   call process_tb
      when (a='DSNU616I' & word(rec1,2)='DSNUSUDS') | rec_ind=616 then
           call process_kcard
      otherwise nop
    end
return Ø
process_print:
    rec2= ' '
return Ø
process_write:
    outlist.1 = rec2
    "EXECIO 1 DISKW DATAOUT (STEM OUTLIST."
return Ø

process_tb_part:
  rec_ind=613
  if  a='DSNU613I'  then
      do
        num_tbpts=num_tbpts+1
        tbpt_num.num_tbpts=word(rec1,11)
```

```
      end
  if word(rec1,1) = 'CARD'    then
     tbpt_card.num_tbpts=word(rec1,3)
  if word(rec1,1) = 'NEARINDREF'    then
     tbpt_nindref.num_tbpts=word(rec1,3)
  if word(rec1,1) = 'FARINDREF'    then
     tbpt_findref.num_tbpts=word(rec1,3)
  if word(rec1,1) = 'SPACE'    then
     tbpt_space.num_tbpts=word(rec1,3)
  if word(rec1,1) = 'NPAGES'    then
     tbpt_npages.num_tbpts=word(rec1,3)
  if word(rec1,1) = 'NACTIVE'    then
     tbpt_nactive.num_tbpts=word(rec1,3)
return Ø

process_ix:
  rec_ind=617
  if  npi_ind=1  then
      do
        call process_npi
        return Ø
      end
  if  a='DSNU617I'  then  ix_name=word(rec1,9)
  if word(rec1,1) = 'CLUSTERED'    then
     ix_clustered=word(rec1,3)
  if word(rec1,1) = 'CLUSTERRATIO'    then
     ix_clustratio=word(rec1,3)
  if word(rec1,1) = 'FIRSTKEYCARDF='   then
     ix_firstkeycardf=word(rec1,2)
  if word(rec1,1) = 'FULLKEYCARDF'   then
     ix_fullkeycardf=word(rec1,3)
  if word(rec1,1) = 'NLEAF'    then
     ix_nleaf=word(rec1,3)
return Ø

process_npi:
  if  a='DSNU617I'  then  npi_name.num_npis=word(rec1,9)
  if word(rec1,1) = 'CLUSTERED'    then
     npi_clustered.num_npis=word(rec1,3)
  if word(rec1,1) = 'CLUSTERRATIO'    then
     npi_clustratio.num_npis=word(rec1,3)
  if word(rec1,1) = 'FIRSTKEYCARDF='   then
     npi_firstkeycardf.num_npis=word(rec1,2)
  if word(rec1,1) = 'FULLKEYCARDF'   then
     npi_fullkeycardf.num_npis=word(rec1,3)
  if word(rec1,1) = 'NLEAF'    then
     npi_nleaf.num_npis=word(rec1,3)
return Ø

process_cl:
```

```
  rec_ind=615
  if ix_name<>'' | num_npis>Ø | num_ixpts >Ø then
     return Ø
  if  a='DSNU615I'  then
      do
        num_cols=num_cols+1
        cl_tbname.num_cols=tb_name.num_tbs
        cl_name.num_cols=word(rec1,9)
      end
  if word(rec1,1) = 'COLCARDF'    then
     cl_colcardf.num_cols=word(rec1,3)
  if word(rec1,1) = 'HIGH2KEY'    then
     cl_high2key.num_cols=word(rec1,3)
  if word(rec1,1) = 'LOW2KEY'    then
     cl_low2key.num_cols=word(rec1,3)
return Ø

process_ix_part:
  rec_ind=618
  if  a='DSNU618I'  then
      npi_ind=Ø
  if ( a='DSNU618I' & word(rec1,11) = 'Ø' ) | npi_ind=1  then
      do
        call process_npipt
        return Ø
      end
  if  a='DSNU618I'  then
      do
        num_ixpts=num_ixpts+1
        ixpt_num.num_ixpts=word(rec1,11)
      end
  if word(rec1,1) = 'CARDF'    then
     ixpt_cardf.num_ixpts=word(rec1,3)
  if word(rec1,1) = 'NEAROFFPOSF'    then
     ixpt_noffposf.num_ixpts=word(rec1,3)
  if word(rec1,1) = 'FAROFFPOSF'    then
     ixpt_foffposf.num_ixpts=word(rec1,3)
  if word(rec1,1) = 'LEAFDIST'    then
     ixpt_leafdist.num_ixpts=word(rec1,3)
  if word(rec1,1) = 'SPACE'    then
     ixpt_space.num_ixpts=word(rec1,3)
  if word(rec1,1) = 'CLUSTERRATIO'    then
     ixpt_clustratio.num_ixpts=word(rec1,3)
  if word(rec1,1) = 'FIRSTKEYCARD'    then
     ixpt_firstkeycard.num_ixpts=word(rec1,3)
  if word(rec1,1) = 'FULLKEYCARD'    then
     ixpt_fullkeycard.num_ixpts=word(rec1,3)
  if word(rec1,1) = 'NLEAF'    then
     ixpt_nleaf.num_ixpts=word(rec1,3)
return Ø
```

```
process_npipt:
  if  a='DSNU618I'  then
      do
        num_npis=num_npis+1
        npi_ind=1
      end
  if word(rec1,1) = 'NEAROFFPOSF'    then
     npi_noffposf.num_npis=word(rec1,3)
  if word(rec1,1) = 'FAROFFPOSF'    then
     npi_foffposf.num_npis=word(rec1,3)
  if word(rec1,1) = 'LEAFDIST'    then
     npi_leafdist.num_npis=word(rec1,3)
  if word(rec1,1) = 'SPACE'    then
     npi_space.num_npis=word(rec1,3)
return Ø

process_ts:
  rec_ind=612
  if  a='DSNU612I'  then  ts_name=word(rec1,9)
  if word(rec1,1) = 'NACTIVE'  then
     ts_nactive=word(rec1,3)
return Ø

process_tb:
  rec_ind=614
  if  a='DSNU614I'  then
      do
        num_tbs=num_tbs+1
        tb_name.num_tbs=word(rec1,9)
      end
  if word(rec1,1) = 'CARDF'    then
     tb_cardf.num_tbs=word(rec1,3)
  if word(rec1,1) = 'NPAGES'    then
     tb_npages.num_tbs=word(rec1,3)
  if word(rec1,1) = 'PCTPAGES'    then
     tb_pctpages.num_tbs=word(rec1,3)
return Ø

process_kcard:
  rec_ind=616
  if  a='DSNU616I'  then
      do
        num_kcard=num_kcard+1
        rec1=subword(rec1,7)
        rec1=translate(rec1,' ',',')
        kcard_cl_str.num_kcard=rec1
      end
  else
     if word(rec1,1) = 'CARDINALITY'    then
```

```
        kcard_cl_card.num_kcard=word(rec1,3)
     else
       do
         rec1=translate(rec1,' ',',')
         kcard_cl_str.num_kcard=kcard_cl_str.num_kcard||rec1
       end
return Ø

process_detail:
  if  a='DSNU624I' | a='DSNU627I'  then
    nop
  else
    rec_ind=Ø
return Ø

dis_play:
  rec2=''
  rec2='                                                RUNSTATS REPORT'
  call process_write
  rec2='                                                ________________'
  call process_write
  rec2=''
  call process_write
  rec2=''
  call process_write
  rec2=''
  call process_write
  if ts_name<>' '  then
    call print_ts
  if num_tbpts > Ø  then
    call print_tbpt
  rec2=''
  call process_write
  if num_tbs > Ø  then
    call print_tb
  rec2=''
  call process_write
  if ix_name<>' '  then
    call print_ix
  rec2=''
  call process_write
  if num_ixpts > Ø  then
    call print_ixpt
  rec2=''
  call process_write
  if num_npis > Ø  then
    call print_npis
  rec2=''
  call process_write
  if num_kcard > Ø  then
```

```
    call print_kcard
  rec2=''
  call process_write
return Ø

print_ts:
    rec2=' Tablespace name : '||ts_name
    call process_write
    rec2=' Number of Active pages : '||ts_nactive
    call process_write
return Ø

print_tb:
  do itbs=1 to num_tbs
    rec2=''
    call process_write
    rec2=' Table name : '||tb_name.itbs
    call process_write
    rec2=' Table cardinality: '||tb_cardf.itbs
    call process_write
    rec2=' Number of pages: '||tb_npages.itbs
    call process_write
    rec2=' Percentage of pages: '||tb_pctpages.itbs
    call process_write
    rec2=''
    call process_write
    if num_cols>Ø then
       call print_cl
    rec2=''
    call process_write
  end
return Ø

print_cl:
    rec2=' Column level Statistics:'
    call process_write
    rec2=' ------------------------------------------------------------'
    rec2=rec2||'-------------------'
    call process_write
    rec2=' Name                Cardinality       LOW2KEY      '
    rec2=rec2||'        HIGH2KEY'
    call process_write
    rec2=' ------------------------------------------------------------'
    rec2=rec2||'-------------------'
    call process_write
  do icols=1 to num_cols
    if cl_tbname.icols<>tb_name.itbs then
       iterate
    rec2=' '||substr(cl_name.icols,1,18)||'  '
    rec2=rec2||substr(cl_colcardf.icols,1,16)||'  '
```

```
    rec2=rec2||substr(cl_low2key.icols,1,19)||'  '
    rec2=rec2||substr(cl_high2key.icols,1,19)||'  '
    call process_write
  end
    rec2=' ------------------------------------------------------------'
    rec2=rec2||'------------------'
    call process_write
return Ø

print_ix:
    rec2=' Index name : '||ix_name
    call process_write
    rec2=' Clustered: '||ix_clustered
    call process_write
    rec2=' Clusterratio: '||ix_clustratio
    call process_write
    rec2=' First Key Cardinality: '||ix_firstkeycardf
    call process_write
    rec2=' Full Key Cardinality: '||ix_fullkeycardf
    call process_write
    rec2=' Number of leaf pages: '||ix_nleaf
    call process_write
return Ø

print_tbpt:
    if tbpt_num.1='Ø'  then
       do
         rec2=' Card : '||tbpt_card.1
         call process_write
         rec2=' NearIndRef : '||tbpt_nindref.1
         call process_write
         rec2=' FarIndRef : '||tbpt_findref.1
         call process_write
         rec2=' Space : '||tbpt_space.1
         call process_write
         return Ø
       end
    rec2=''
    call process_write
    rec2=' Tablespace Partition level Statistics:'
    call process_write
    rec2=' ------------------------------------------------------------'
    rec2=rec2||'------------------'
    call process_write
    rec2=' Part Num. Cardf      Nearindref Farindref   Space    Pages'
    rec2=rec2||'      Active pages'
    call process_write
    rec2=' ------------------------------------------------------------'
    rec2=rec2||'------------------'
    call process_write
```

```
  do itbpts=1 to num_tbpts
    rec2=' '||substr(tbpt_num.itbpts,1,3)||'       '
    rec2=rec2||substr(tbpt_card.itbpts,1,10)||' '
    rec2=rec2||substr(tbpt_nindref.itbpts,1,10)||' '
    rec2=rec2||substr(tbpt_findref.itbpts,1,10)||'  '
    rec2=rec2||substr(tbpt_space.itbpts,1,8)||' '
    rec2=rec2||substr(tbpt_npages.itbpts,1,8)||'   '
    rec2=rec2||substr(tbpt_nactive.itbpts,1,8)
    call process_write
  end
    rec2=' ------------------------------------------------------------'
    rec2=rec2||'-------------------'
    call process_write
return 0

print_ixpt:
    rec2=' Index Partition level Statistics:'
    call process_write
    rec2=' ------------------------------------------------------------'
    rec2=rec2||'------------------------------------------------------'
    call process_write
    rec2=' Part   ClustRatio  FirstKeyCard    FullKeyCard    Space     '
    rec2=rec2||' NearoffPos    FaroffPos      LeafDist     LeafPages'
    call process_write
    rec2=' ------------------------------------------------------------'
    rec2=rec2||'------------------------------------------------------'
    call process_write
  do iixpts=1 to num_ixpts
    rec2=' '||substr(ixpt_num.iixpts,1,3)||'    '
    rec2=rec2||substr(ixpt_clustratio.iixpts,1,3)||'         '
    rec2=rec2||substr(ixpt_firstkeycard.iixpts,1,10)||'      '
    rec2=rec2||substr(ixpt_fullkeycard.iixpts,1,10)||'     '
    rec2=rec2||substr(ixpt_space.iixpts,1,8)||'   '
    rec2=rec2||substr(ixpt_noffposf.iixpts,1,10)||'    '
    rec2=rec2||substr(ixpt_foffposf.iixpts,1,10)||'     '
    rec2=rec2||substr(ixpt_leafdist.iixpts,1,8)||'     '
    rec2=rec2||substr(ixpt_nleaf.iixpts,1,8)||'  '
    call process_write
  end
    rec2=' ------------------------------------------------------------'
    rec2=rec2||'------------------------------------------------------'
    call process_write
return 0

print_npis:
  do inpis=1 to num_npis
    rec2=' Index name : '||npi_name.inpis
    call process_write
    rec2=' Clustered: '||npi_clustered.inpis
    call process_write
```

```
    rec2=' Clusterratio: '||npi_clustratio.inpis
    call process_write
    rec2=' First Key Cardinality: '||npi_firstkeycardf.inpis
    call process_write
    rec2=' Full Key Cardinality: '||npi_fullkeycardf.inpis
    call process_write
    rec2=' Number of leaf pages: '||npi_nleaf.inpis
    call process_write
    rec2=' Space : '||npi_space.inpis
    call process_write
    rec2=' NearOffPos : '||npi_noffposf.inpis
    call process_write
    rec2=' FarOffPos : '||npi_foffposf.inpis
    call process_write
    rec2=' Leaf Distance : '||npi_leafdist.inpis
    call process_write
    rec2=''
    call process_write
    rec2=''
    call process_write
  end
return 0

print_kcard:
    rec2=' SYSCOLDIST - Key Cardinality Statistics: '
    call process_write
    rec2=' ------------------------------------------------------------'
    rec2=rec2||'------------------------------------------------------'
    rec2=rec2||'-------------'
    call process_write
    rec2=' Column Names                                                 '
    rec2=rec2||'              '
    rec2=rec2||'                                        Cardinality '
    call process_write
    rec2=' ------------------------------------------------------------'
    rec2=rec2||'------------------------------------------------------'
    rec2=rec2||'-------------'
    call process_write
  do ikcard=1 to num_kcard
    rec2=''
    ncols=0
    nwords=words(kcard_cl_str.ikcard)
    do ik=1 to nwords
       rec2=rec2||substr(word(kcard_cl_str.ikcard,ik),1,18)||' '
       ncols=ncols+1
       if ncols=6 then
         if ik=6 then
           do
             rec2=rec2||' '||substr(kcard_cl_card.ikcard,1,18)
             call process_write
```

```
            rec2=''
            ncols=Ø
          end
        else
          do
            rec2=rec2||'  '
            call process_write
            rec2=''
            ncols=Ø
          end
    end
    if ncols>Ø then
      do ik=(ncols+1) to 6
         rec2=rec2||'                    '
      end
    if ncols>Ø then
      if nwords < 6 then
         rec2=rec2||' '||substr(kcard_cl_card.ikcard,1,18)
    if ncols>Ø then
       call process_write
    rec2=''
    call process_write
    rec2=''
    call process_write
    rec2=''
    call process_write
  end
  rec2=' ------------------------------------------------------------'
  rec2=rec2||'-----------------------------------------------------'
  rec2=rec2||'-------------'
  call process_write
return Ø

process_init:
rec1=' '
rec2=' '
rec_ind=Ø
first_ind=Ø
ts_name=''
ts_nactive=''
num_tbs=Ø
itbs=Ø
tb_name.Ø=Ø
tb_name.=''
tb_cardf.Ø=Ø
tb_cardf.=''
tb_npages.Ø=Ø
tb_npages.=''
tb_pctpages.Ø=Ø
tb_pctpages.=''
```

```
num_tbpts=0
itbpts=0
tbpt_num.=''
tbpt_num.0=0
tbpt_card.=''
tbpt_card.0=0
tbpt_nindref.0=0
tbpt_nindref.=''
tbpt_findref.0=0
tbpt_findref.=''
tbpt_space.0=0
tbpt_space.=''
tbpt_npages.0=0
tbpt_npages.=''
tbpt_nactive.0=0
tbpt_nactive.=''
num_cols=0
icols=0
cl_tbname.0=0
cl_tbname.=''
cl_name.0=0
cl_name.=''
cl_colcardf.0=0
cl_colcardf.=''
cl_low2key.0=0
cl_low2key.=''
cl_high2key.0=0
cl_high2key.=''
npi_ind=0
num_npis=0
inpis=0
npi_name.0=0
npi_name.=''
npi_clustered.0=0
npi_clustered.=''
npi_clustratio.0=0
npi_clustratio.=''
npi_firstkeycardf.0=0
npi_firstkeycardf.=''
npi_fullkeycardf.0=0
npi_fullkeycardf.=''
npi_nleaf.0=0
npi_nleaf.=''
npi_noffposf.0=0
npi_noffposf.=''
npi_foffposf.0=0
npi_leafdist.0=0
npi_leafdist.=''
npi_space.0=0
npi_space.=''
```

```
ix_name=''
ix_clustered=''
ix_clustratio=Ø
ix_firstkeycardf=Ø
ix_fullkeycardf=Ø
ix_nleaf=Ø
num_ixpts=Ø
iixpts=Ø
ixpt_num.Ø=Ø
ixpt_num.=''
ixpt_cardf.Ø=Ø
ixpt_cardf.=''
ixpt_noffposf.Ø=Ø
ixpt_noffposf.=''
ixpt_foffposf.Ø=Ø
ixpt_foffposf.=''
ixpt_leafdist.Ø=Ø
ixpt_leafdist.=''
ixpt_space.Ø=Ø
ixpt_space.=''
ixpt_clustratio.Ø=Ø
ixpt_clustratio.=''
ixpt_firstkeycard.Ø=Ø
ixpt_firstkeycard.=''
ixpt_fullkeycard.Ø=Ø
ixpt_fullkeycard.=''
ixpt_nleaf.Ø=Ø
ixpt_nleaf.=''
num_kcard=Ø
ikcard=Ø
ncols=Ø
ik=Ø
nwords=Ø
kcard_cl_str.Ø=Ø
kcard_cl_str.=""
kcard_cl_card.Ø=Ø
kcard_cl_card.=""
return Ø
/************ End of REXX Program ********************************** /
```

RUNSTATS REPORT

```
 Tablespace name : DSNDBØ4.SDSTPKG
 Number of Active pages : 25Ø86

 Tablespace Partition level Statistics:
 ---------------------------------------------------------------------
 Part Num. Cardf   Nearindref Farindref  Space   Pages     Active pages
```

```
 ------------------------------------------------------------------------
 1         120000     0          0       50400    12000      12003
 2         119985     0          0       63360    11999      12003
 3         0          0          0       3600     0          360
 4         0          0          0       3600     0          360
 5         0          0          0       3600     0          360
 ------------------------------------------------------------------------

 Table name : DBM.TDSTPKG
 Table cardinality: 2.39985E+05
 Number of pages: 23999
 Percentage of pages: 95

 Column level Statistics:
 ------------------------------------------------------------------------
 Name               Cardinality   LOW2KEY              HIGH2KEY
 ------------------------------------------------------------------------
 PKG_TCK_NR         2.816E+04   X'C1F0F0F0F3F1F6F9'  X'D5F0F7F9F8F3F5F7'
 AC_NR              1.28E+04    X'C1D7F6F1F0F0F4F0'  X'F9F9F9F6F0F4F6F8'
 REC_STS_CD         1.0E+00     X'C140404040404040'  X'C140404040404040'
 INF_SRC_TYP_CD     1.0E+00     X'4040404040404040'  X'4040404040404040'
 PKG_ALT_REF_NR     1.0E+00     X'4040404040404040'  X'4040404040404040'
 PKG_DEL_DT         4.2E+01     X'1996100140404040'  X'1996111640404040'
 CNS_BTO_AC_NR      1.0E+00     X'4040404040404040'  X'4040404040404040'
 CUS_AD_NA          4.352E+03   X'5CE2D6E4E3C8C5D9'  X'F9F1F04040404040'
 PKG_RCV_CRF_SIG_TE 1.2416E+04  X'C140C1C2C4E4D3D3'  X'E9E8C4E9C9D24040'
 UDT_TS             1.0E+00     X'0001010100000000'  X'0001010100000000'
 PKG_CHA_TYP_CD     1.14E+02    X'F0F0F04040404040'  X'F8F7F14040404040'
 ------------------------------------------------------------------------

 Index name : DBM.IDSTPKG1
 Clustered: Y
 Clusterratio: 100
 First Key Cardinality: 2.0E+00
 Full Key Cardinality: 5.9996E+04
 Number of leaf pages: 1112


Index Partition level Statistics:
 ------------------------------------------------------------------------
----------------------------------------
 Part ClustRatio FirstKeyCard FullKeyCard Space NearoffPos FaroffPos
LeafDist     LeafPages
 ------------------------------------------------------------------------
----------------------------------------
 1      100         1              30000          5760       1.0E+00
0.0E0          0           556
 2      100         1              29996          4320       1.0E+00
0.0E0          0           556
```

```
 3      Ø           Ø               Ø               72Ø         Ø.ØEØ
Ø.ØEØ          Ø           Ø
 4      Ø           Ø               Ø               72Ø         Ø.ØEØ
Ø.ØEØ          Ø           Ø
 5      Ø           Ø               Ø               72Ø         Ø.ØEØ
Ø.ØEØ          Ø           Ø
 ------------------------------------------------------------------------
----------------------------------------

 Index name : DBM.IDSTPKG2
 Clustered: N
 Clusterratio: 1Ø
 First Key Cardinality: 1.ØE+ØØ
 Full Key Cardinality: 1.475E+Ø4
 Number of leaf pages: 5Ø7
 Space : 72Ø
 NearOffPos : 2.414E+Ø3
 FarOffPos : 3.3261E+Ø4
 Leaf Distance : 1

 Index name : DBM.IDSTPKG3
 Clustered: Y
 Clusterratio: 99
 First Key Cardinality: 1.ØE+ØØ
 Full Key Cardinality: 1.ØE+ØØ
 Number of leaf pages: 596
 Space : 28ØØØ
 NearOffPos : 2.ØE+ØØ
 FarOffPos : 1.ØE+ØØ
 Leaf Distance : Ø

 SYSCOLDIST - Key Cardinality Statistics:
 ------------------------------------------------------------------------
--------
 Column Names Cardinality
Cardinality
 ------------------------------------------------------------------------
--------
 TBL_PTN_NR         PKG_TCK_NR
5.9996E+Ø4

 ------------------------------------------------------------------------
--------
```

DB2 Version 6 Installation dialog

INTRODUCTION

We recently installed DB2 UDB Version 6.1 for OS/390 and migrated our existing DB2 subsystems from Version 5.1. IBM supply a DB2 Installation dialog for this. It reads/saves many values in SDSNSAMP members then generates batch JCL (and CLISTs) for the installation. Unfortunately, it does not save all the relevant values separately for each DB2 subsystem/version. The names of the BookManager books and the initial values from the first panel (DSNTIPA1) are saved into the profile dataset for that user-id, which is not sharable with other users.

To overcome that, we wrote a front-end to the standard IBM dialog. Our dialog saves the last-used variables in a shared table library. When any user starts our dialog, it allocates the necessary DB2/ISPF libraries, then starts the standard DB2 Installation dialog with the correct last-used values on the first couple of panels.

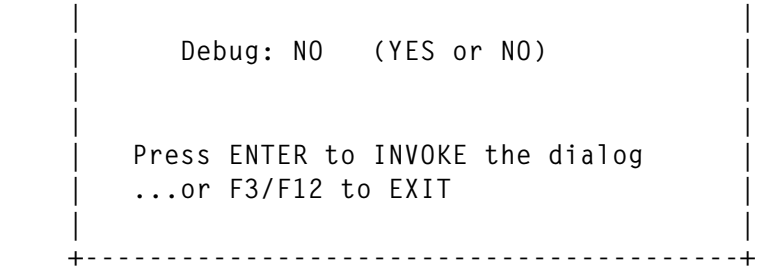
USAGE NOTES

It is invoked in the following way:

- 'TSO DB2INST' – then specify the ssid/version in the window.
- 'TSO DB2INST ssid' – then specify the version in the window.
- 'TSO DB2INST ssid ver' – to start the standard dialog immediately.

The initial window might look like this:

```
+-------- DB2 Installation Dialog --------+
| Command ==>                              |
|                                          |
|                                          |
|   DB2 SSID:      (DBxn eg DBT1)          |
|                                          |
|    Version: 610  (610, 510 or 410)       |
|                                          |
|   BookMngr: YES  (YES or NO)             |
```



By default it usually sets BookMngr=YES (to use BookManager for HELP) and Debug=NO (to NOT LIST the CLIST statements from the standard dialog), and you can change those in the initial window if you wish.

Sometimes it is useful to run the dialog for a previous version of DB2. When you MIGRATE a DB2 subsystem, the dialog uses the parameter values from the old release (stored in the old SDSNSAMP library). With the dialog you can check/update the stored values or generate a new ZPARM job (member DSNTIJUZ) to compare against your current DB2 parameters.

Warning: IBM sometimes supplies PTFs which enable new ZPARM parameters, but the dialog still generates a DSNTIJUZ job without them. So you must compare the generated parameters for the new release against your old parameters.

Our dialog allows multiple user-ids to use the Installation dialog, but it prevents two users from simultaneously customizing the same subsystem/version. It also prevents one user-id from running the dialog for more than one DB2 at a time.

IMPLEMENTATION

It consists of REXX EXEC DB2INST and ISPF panels DB2INSTP and DB2INSTH, which must be put into libraries allocated to SYSEXEC and ISPPLIB. It uses standard IBM message ISRZ002 for any messages. It needs a table output library to be allocated, which can be updated by all user-ids that will use the dialog.

You must update the sample DSNTINST EXEC to include your library names (near the start of the code). Note that if you set the BookManager prefix to null (ie bprf = ''), the BookMngr option is not shown in the window and BookManager will not be used.

The panel must be updated to have the correct list of valid DB2 ssids for your site. Update the lines with ‘IF (&SSID = DBT1,DBT2...’ and also ‘&ZERRLM = 'Only DBT1,DBT2...'’ for that. This panel is used to check the parameters even if it is not displayed.

When those updates are made the dialog will be ready to use.

DB2INST EXEC

```
/*============================>> REXX <<================================*/
/* DB2INST: front-end for IBM DB2 Installation dialog                 */
/*--------------------------------------------------------------------*/
/* The user chooses the DB2 ssid/version, and this EXEC allocates     */
/* the necessary DB2/ISPF libraries for the DB2 Installation dialog.  */
/* Then it gets the last-used values for that particular DB2 ssid,    */
/* and starts the normal DB2 installation dialog.  When that dialog   */
/* ends, the up-to-date values are saved in a shared table library.   */
/* The next time any user wants to use the DB2 Installation dialog    */
/* for the same ssid & version, they start with the last-used values. */
/* MVS enqueues prevent one user from running two DB2 Installation    */
/* dialogs at once, and they prevent two users from simultaneously    */
/* using it for the same DB2 ssid & version.                          */
/*====================================================================*/
  /*-----------------------*/
  /* specify library names */
  /*-----------------------*/
  tlib = 'DB2INST.ISPTLIB'      /* <------ shared table library       */
  v6prefix = 'DB2.DSN610'       /* DB2 V6 SMP/E target library prefix */
  v6suffix = ''                 /* DB2 V6 SMP/E target library suffix */
  v5prefix = 'DB2.DSN510'       /* DB2 V5 SMP/E target library prefix */
  v5suffix = ''                 /* DB2 V5 SMP/E target library suffix */
  v4prefix = 'DB2.DSN410'       /* DB2 V4 SMP/E target library prefix */
  v4suffix = ''                 /* DB2 V4 SMP/E target library suffix */
/*bprf = ''   <----------------- * BookManager not to be used         */
  bprf = 'EOY'                  /* BookManager target library prefix  */
  Address ISPEXEC
  "CONTROL ERRORS RETURN"
  /*------------------------------*/
  /* create serialization enqueue */
  /*------------------------------*/
  "TBCREATE SERIALIZ LIBRARY(ISPPROF) WRITE"
  If rc > 0 Then Do
     Call MSG('You are already using the DB2 Installation dialog',
         'in this TSO session.                                  ',
         'Only one at a time is allowed.')
     Exit
     End
  /*-----------------------------------------*/
  /* display Pop-up window to get SSID and VER */
```

```
/*-------------------------------------------*/
Arg SSID VER                          /* did user supply the values ? */
If SSID <> '' & VER <> '' Then
    "CONTROL NONDISPL ENTER"            /* do not display the panel */
 "ADDPOP"
 "DISPLAY PANEL(DB2INSTP)"          /* user can specify ssid/version */
 disrc = rc
 "REMPOP"
 If disrc > Ø Then Signal FINISH
/*-------------------------------------------*/
/* use appropriate DB2 library prefix & suffix */
/*-------------------------------------------*/
Select
  When VER = '61Ø' Then Do
     dprf = v6prefix
     If v6suffix <> ''
        Then dsuf = '.'v6suffix
        Else dsuf = ''
     End
  When VER = '51Ø' Then Do
     dprf = v5prefix
     If v5suffix <> ''
        Then dsuf = '.'v5suffix
        Else dsuf = ''
     End
  When VER = '41Ø' Then Do
     dprf = v4prefix
     If v4suffix <> ''
        Then dsuf = '.'v4suffix
        Else dsuf = ''
     End
  End
/*-------------------------------------------*/
/* copy any existing profile to user's library */
/*-------------------------------------------*/
Address TSO "ALLOC FI(DSNTLIB) DSN('"tlib"') SHR REUSE"
tabl = SSID||VER
"TBOPEN" tabl "LIBRARY(DSNTLIB) WRITE"      /* get last-used values */
If rc = Ø Then
   "TBSAVE" tabl "NAME(DSNTPROF)"      /* copy it to user's ISPTABL */
If rc = 8 Then
   Call MSG('Warning: this DB2 SSID/version was not previously',
            'customized.                                      ',
            'IBM defaults will be set as initial values')
If rc > 8 Then Do
   If rc = 12 Then
      Call MSG('Another user is already using the DB2 Installation',
               'dialog for' SSID 'version' VER '                  ',
               'Try again later')
   If rc = 16 Then
      Call MSG('Unable to allocate' tlib 'to ddname DSNTLIB')
```

```
   If rc = 2Ø Then
      Call MSG('Severe error in TBSAVE of table' tabl 'to name',
               'DSNTPROF')
   Signal FINIS
   End
/*-----------------------------------*/
/* make temporary dataset allocations */
/*-----------------------------------*/
dmlib = dprf'.SDSNSPFM'dsuf
dplib = dprf'.SDSNSPFP'dsuf
dslib = dprf'.SDSNSPFS'dsuf
dtlib = dprf'.SDSNSPFT'dsuf
lib_error = ''
If BKM = 'YES' Then Do       /* installation with HELP books */
   "LIBDEF ISPLLIB DATASET ID(          '"bprf".SEOYLOAD') STACK"
   If rc > Ø Then lib_error = lib_error 'ISPLLIB'
   "LIBDEF ISPMLIB DATASET ID('"dmlib"' '"bprf".SEOYMENU') STACK"
   If rc > Ø Then lib_error = lib_error 'ISPMLIB'
   "LIBDEF ISPPLIB DATASET ID('"dplib"' '"bprf".SEOYPENU') STACK"
   If rc > Ø Then lib_error = lib_error 'ISPPLIB'
   "LIBDEF ISPSLIB DATASET ID('"dslib"'                  ) STACK"
   If rc > Ø Then lib_error = lib_error 'ISPSLIB'
   "LIBDEF ISPTLIB DATASET ID('"dtlib"' '"bprf".SEOYTENU') STACK"
   If rc > Ø Then lib_error = lib_error 'ISPTLIB'
   Address TSO "ALTLIB ACT APPLICATION(EXEC) DA('"bprf".SEOYCLIB')"
   If rc > Ø Then lib_error = lib_error 'SYSEXEC'
   End
Else Do                      /* installation without HELP */
   "LIBDEF ISPMLIB DATASET ID('"dmlib"') STACK"
   If rc > Ø Then lib_error = lib_error 'ISPMLIB'
   "LIBDEF ISPPLIB DATASET ID('"dplib"') STACK"
   If rc > Ø Then lib_error = lib_error 'ISPPLIB'
   "LIBDEF ISPSLIB DATASET ID('"dslib"') STACK"
   If rc > Ø Then lib_error = lib_error 'ISPSLIB'
   "LIBDEF ISPTLIB DATASET ID('"dtlib"') STACK"
   If rc > Ø Then lib_error = lib_error 'ISPTLIB'
   End
If lib_error <> '' Then
   Call MSG('Library allocation error for ddname:'lib_error ,
            Copies(' ',3Ø) 'Check the library name prefix/suffix' ,
            'in DB2INST exec')
/*----------------------------*/
/* invoke the standard dialog */
/*----------------------------*/
If lib_error = '' Then Do       /* libraries successfully allocated */
   dclib = dprf'.SDSNCLST'dsuf
   If LST = 'YES' Then debug = 'CONTROL(LIST)'
                  Else debug = ''
   "SELECT CMD(EXEC '"dclib"(DSNTINST)' '"debug "FROMØ("BKM")' )",
               "NEWAPPL(DSNT) PASSLIB"
   dsntinst_rc = rc               /* rc=Ø (normal end); rc=8 (PF3) */
```

```
      If dsntinst_rc > 8 Then
         Call MSG('Error in DSNTINST invocation (IBM standard DB2',
                  'Installation dialog), rc='rc)
      End
    /*-----------------------------------*/
    /* end temporary dataset allocations */
    /*-----------------------------------*/
    If Pos('ISPMLIB',lib_error) = Ø Then "LIBDEF ISPMLIB"
    If Pos('ISPPLIB',lib_error) = Ø Then "LIBDEF ISPPLIB"
    If Pos('ISPSLIB',lib_error) = Ø Then "LIBDEF ISPSLIB"
    If Pos('ISPTLIB',lib_error) = Ø Then "LIBDEF ISPTLIB"
    If BKM = 'YES' Then Do
       If Pos('ISPLLIB',lib_error) = Ø Then "LIBDEF ISPLLIB"
       If Pos('SYSEXEC',lib_error) = Ø Then
          Address TSO "ALTLIB DEACT APPLICATION(EXEC)"
       End
    /*-------------------------------------------------*/
    /* save new profile in appropriate (shared) library */
    /*-------------------------------------------------*/
    "TBCLOSE" tabl                      /* close table, drop the enqueue */
    If dsntinst_rc = Ø | dsntinst_rc = 8 Then Do
       "TBOPEN DSNTPROF SHARE WRITE"
       "TBSAVE DSNTPROF NAME("tabl") LIBRARY(DSNTLIB)"
       If rc = Ø Then
          Call MSG('   ***  Profile saved in' tlib'('tabl')  ***')
       Else
          Call MSG('Error saving profile in' tlib'('tabl'), rc='rc)
       "TBEND  DSNTPROF"
       End
    /*----------------------*/
    /* clean up and go home */
    /*----------------------*/
FINIS:
    "TBOPEN XXXXX LIBRARY(XXX)"            /* this makes the FREE work  */
    Address TSO "FREE FI(DSNTLIB)"
FINISH:
    "TBEND SERIALIZ"               /* release the serialization enqueue */
    Exit
/*====================================================================*/
/* Subroutine to display ISPF message in window box                   */
/*--------------------------------------------------------------------*/
MSG:
    Parse Arg ZERRLM                    /* get the long message text    */
    ZERRSM = ''                         /* no short message             */
    ZERRHM = '*'                        /* use HELP specified in panel  */
    If Left(ZERRLM,1) = ' ' Then        /* Information message:         */
       ZERRALRM = 'NO'                  /* no alarm                     */
    Else                                /* Error message:               */
       ZERRALRM = 'YES .WINDOW=LRESP'   /* alarm, user must press ENTER */
    "SETMSG MSG(ISRZØØ2)"               /* standard IBM ISPF message    */
    Return
```

DB2INSTP PANEL

```
)PANEL KEYLIST(ISRSNAB,ISR)
)ATTR DEFAULT(%+_)
/*-------------------------------------------------------------------*/
/*  Panel for front-end to IBM DB2 Installation dialog               */
/*-------------------------------------------------------------------*/
 $ TYPE(TEXT)   COLOR(GREEN)
 ı TYPE(OUTPUT) COLOR(TURQ)   CAPS(OFF)
 · TYPE(OUTPUT) COLOR(GREEN)  CAPS(OFF)
 ? TYPE(OUTPUT) COLOR(BLUE)
 # TYPE(CHAR)   COLOR(GREEN)
 * TYPE(CHAR)   COLOR(WHITE)
 @ AREA(DYNAMIC)
)BODY WINDOW(40,15)
+Command ==>_ZCMD
$
$
$  DB2 SSID:_SSID+(DBxn eg. DBT1)
$
$   Version:_VER+ (610, 510 or 410)
$
$ ·BOOKMG   _BKM+?BOOKOPT       +
$
$     Debug:_LST+ (YES or NO)
$
$
$  Press%ENTER$to INVOKE the dialog
$  @TEXT,SHAD             @
)INIT
 &ZWINTTL = 'DB2 Installation Dialog'
 &ZCMD = &Z
 IF (&SSID = &Z)
     .CURSOR = SSID
 IF (&VER = &Z)
     &VER = 610                        /* 610 is default version */
     .CURSOR = VER
 IF (&BPRF = &Z)                       /* BookManager library prefix */
     &BOOKMG = &Z
     &BOOKOPT = &Z
     &BKM  = NO
     .ATTR(BKM) = 'TYPE(OUTPUT) INTENS(NON)'
 ELSE
     &BOOKMG = 'BookMngr'
     &BOOKOPT = '(YES or NO)'
     &BKM  = YES
 &LST  = NO
 &TEXT = '...or F3/F12 to EXIT'
 &SHAD = '########*#***#########'
 VGET ZKLUSE PROFILE
 &KLUSE = &ZKLUSE
```

```
 IF (&ZKLUSE = N)                       /* user has KEYLIST OFF */
     &ZKLUSE = Y                        /* turn it ON           */
     VPUT ZKLUSE PROFILE
 .HELP = DB2INSTH
)REINIT
 REFRESH(*)
)PROC
 IF (&ZCMD = CAN,CANCEL,EXIT)
     .RESP = END
 IF (.RESP = END)
     IF (&KLUSE = N)                    /* user had KEYLIST OFF */
         &ZKLUSE = N                    /* turn it OFF again    */
         VPUT ZKLUSE PROFILE
 IF (.RESP = ENTER)
     &VER1 = TRUNC(&VER,1)              /* DB2 version number   */
     IF (&VER1 = 6) &VER  = 610
     IF (&VER1 = 5) &VER  = 510
     IF (&VER1 = 4) &VER  = 410
     &VER2 = TRUNC(&VER,2)
     IF (&VER2 = V6) &VER  = 610
     IF (&VER2 = V5) &VER  = 510
     IF (&VER2 = V4) &VER  = 410
     &BKM1 = TRUNC(&BKM,1)              /* BookMngr = YES or NO */
     IF (&BKM1 = Y) &BKM  = YES
     ELSE           &BKM = NO
     &LST1 = TRUNC(&LST,1)              /* Debug = YES or NO    */
     IF (&LST1 = Y) &LST  = YES
     ELSE           &LST = NO
     VER (&SSID,NB)                     /* DB2 subsystem name   */
     IF (&SSID = DBT1,DBT2,DBR1,DBR2,DBP1,DBP2,DBX1,DBX2)
     ELSE
        &ZERRSM = ''
        &ZERRLM = 'Only DBT1,DBT2,DBR1,DBR2,DBP1,DBP2,DBX1,DBX2 allowed'
        &ZERRALRM = 'YES'
        &ZERRHM = '*'
        .MSG =ISRZ002
     IF (.MSG = &Z)                     /* DB2 version          */
         VER(&VER,NB,LIST,410,510,610)
)END
```

DB2INSTH PANEL

```
)PANEL KEYLIST(ISRHELP,ISR)
/*-------------------------------------------------------------------*/
/*  HELP Panel for DB2 Installation Dialog (exec DB2INST)            */
/*-------------------------------------------------------------------*/
)ATTR DEFAULT(%+_)
¬ TYPE(OUTPUT) COLOR(TURQ)
! TYPE(TEXT)   COLOR(GREEN)
? TYPE(TEXT)   COLOR(YELLOW)
```

```
$ TYPE(TEXT)   COLOR(TURQ)
# TYPE(CHAR)   COLOR(GREEN)
* TYPE(CHAR)   COLOR(TURQ)
@ AREA(DYNAMIC)
)BODY EXPAND(\\)
+HELP-\-\-?DB2 Installation Dialog+-\-\-HELP
+Command ===>_ZCMD
+
    %DB2INST!is a front-end to the standard IBM DB2 Installation dialog,
     to make it easier to invoke and to let multiple user-ids use it.
!
?   Invocation:
!     a)$DB2INST           !then specify the ssid/version on the window
      b)$DB2INST ssid      !then specify the version on the window
      c)$DB2INST ssid ver  !to start standard dialog immediately
!     The required ISPF libraries will be dynamically allocated for your
    chosen DB2 version @HTXT1,HSHD1
@
    optionally for BookManager @HTXT2,HSHD2
@
    Then it starts the standard IBM DB2 Instalation dialog.
      The names of the BookManager libraries and the values on the first
    installation panel are then the same as the last time the same DB2
    ssid/version was customized.  These values are stored in a shared
    table library:ıTLIB
+-\-\-
!
)INIT
 &HTXT1 = '(ie. SDSNSPFM, SDSNSPFP, SDSNSPFS and SDSNSPFT), and'
 &HSHD1 = '#####********##********##********#####********######'
 &HTXT2 = '(ie. SEOYLOAD, SEOYMENU, SEOYPENU and SEOYTENU).'
 &HSHD2 = '#####********##********##********#####********##'
)END
```

CONCLUSION

When you use the DB2 Installation dialog you must be careful about what you enter in the fields in the first couple of panels, particularly for the intial INSTALL or MIGRATE of a subsystem. We cannot change that.

But our dialog allocates the libraries you need, and ensures that any user-id will start the DB2 Installation dialog with the correct last-used values for the first couple of panels.

Ron Brown
Systems Programmer (Germany)

Using the LSTCAT output to generate ALTER SQL statements – revisited

The June 2000 issue of *DB2 Update* (Issue 92) published an article describing the GENALTR utility, which works on the output of the LSTCAT utility. It checks the old PRIQTY and the new PRIQTY and, if they are different, it will generate ALTER SQL statements for that particular object, which could be an indexspace or a tablespace.

The GENALTR REXX that goes with this article is published below. We apologize to readers for the delay in publishing the code.

GENALTR

```
/*********************************************************************/
/*     rexx                                                          */
/*********************************************************************/
trace o
clear
fnd=Ø
PREFX = SYSVAR(SYSPREF)
PARSE UPPER ARG P_dsname
if strip(P_dsname)='' then
do
   Call GETDBLST
end
else
do
   I_lstdsn = strip(P_dsname)
   I_lstdsn = strip(P_dsname,B,"'")
end
Call ALLOCDSN
MAINØØØ:
Call GETCRETOR
Call GETDEFSEC
k = Ø
do i = 3 to dbl.Ø
   strng = strip(dbl.i)
   parse VAR strng dbname.i 1Ø obname.i 18 dumy 65 oldspc.i 74 dum2
   parse VAR strng xdumy 39 opqty.i 5Ø xdumyr
   parse VAR strng dum3 9Ø newspc.i 99 nsec.i 1Ø8 d4 116 pno.i 121 dum5
   parse VAR strng dum6 132 ixmrkr.i 133 prtind.i 133 dum7
   pno.i = strip(pno.i)
   parse VAR dumy a volnam.i zap
end
```

```
Call GENALTER
say 'Results generated into 'ods_name
say
say 'Do you want to check associated volumes for space? Default (Y) '
pull vchek
upper vchek
vchek = strip(vchek)
if  vchek = 'Y' | vchek = '' then
   Call VOLCHEK
exit
GETDBLST:
Say  'Give the input dataset ...'
Say  '(It must be a PS )'
pull I_lstdsn
I_lstdsn = strip(I_lstdsn)
I_lstdsn = strip(I_lstdsn,Both,"'")
x = SYSDSN("'"I_lstdsn"'")
if x ¬= OK then
do
   say; say '*** ERROR ' x ; say
   SIGNAL  GETDBLST
end
return
ALLOCDSN:
Say 'Do you want to edit this dataset? ..Y/N (Default is Y): '
pull I_edresp
upper I_edresp
If strip(I_edresp) = '' | strip(I_edresp) = 'Y' then
   ADDRESS ISPEXEC "EDIT  DATASET('"I_lstdsn"')"
"ALLOCATE DD(lstdd) DSN('"I_lstdsn"') REUSE SHR"
if  rc>Ø then
do
   say  'Failed during allocation of 'I_lstdsn
   exit(8)
end
"execio * DISKR lstdd (FINIS STEM dbl."
address tso "free f(lstdd)"
return
GETCRETOR:
Say  'Give the CREATOR ID for index prefix'
pull I_creator
upper I_creator
if strip(I_creator) = ''  then
do
   say 'Error... Please enter CREATOR ID : '
   signal GETCRETOR
end
else
do
   I_creator = strip(I_creator)
end
```

```
return
GETDEFSEC:
defsec = 1Ø
Say  'Give the default percentage for secondary quantity as a number '
Say  ' Or Hit Enter for default (1Ø) ...'
pull I_defsec
upper I_defsec
I_defsec = strip(I_defsec)
if strip(I_defsec) = ''  then
   nop
else
   defsec = I_defsec
return
GENALTER:
cts= time(S)
cd = date(U)
us_date = substr(cd,7,2)||substr(cd,1,2)||substr(cd,4,2)
GETDBNODE:
defnode = substr(I_lstdsn,31,6)
say  'Give a DBname (upto 6 chars) to use as a node for output dataset'
say  ' Or Press Enter for default node ' defnode
pull dbnode
upper dbnode
dbnode=strip(dbnode)
if length(strip(dbnode)) > 6 then
do
   say '****   Error   ****'
   SIGNAL GETDBNODE
end
if strip(dbnode) = '' then
do
   dbnode = defnode
end
defsec = defsec/1ØØ
say 'Processing the input dataset...'
do i = 3 to dbl.Ø
   dbname.i = strip(dbname.i)
   obname.i = strip(obname.i)
   pno.i    = strip(pno.i)
   oldspc.i = strip(oldspc.i)
   newspc.i = strip(newspc.i)
   nsec.i   = strip(nsec.i)
/* say dbname.i obname.i pno.i oldspc.i newspc.i nsec.i       */
   if oldspc.i ¬= newspc.i then
   do
      diff = newspc.i - opqty.i
      if diff < Ø then diff = Ø
      cyldiff.i = trunc((diff+719)/72Ø)
      Call SUMVOLS
      if nsec.i = 'DEFAULT' then
         secqty.i =  trunc(((defsec*newspc.i)+719)/72Ø)*72Ø
      else
```

```
         secqty.i = nsec.i
      obid = right(obname.i,2)
      if ixmrkr.i = 'T' then
      do
         k=k+1
         out.k = ' ALTER TABLESPACE '||dbname.i||'.'||?bname.i
         if prtind.i = 'P' then
         do
            out.k = out.k || ' PART  '||pno.i
         end
      end
      else
      do
         k=k+1
         out.k = ' ALTER INDEX '||I_creator||'.'||?bname.i
         if prtind.i = 'P' then
         do
            out.k = out.k || ' PART  '||pno.i
         end
      end
      k = k+1
      out.k = '   PRIQTY '||>ewspc.i||' SECQTY '||secqty.i||' ;'
   end
end
/*   delete existing dataset and allocate new dataset  */
/*   and write the output to the new dataset           */
ods_name = PREFX||"."||USERID()||".ALTER."||dbnode||".D"||us_date
say;say 'Writing output members into 'ods_name
xx = outtrap("zap.","*")
address tso "delete '"ods_name"'"
address tso "alloc f(opds) new unit(hsm) space(2,2)",
            "cyl reuse dsname('"ods_name"')",
            "dsorg(ps) blksize(312Ø) lrecl(8Ø) recfm(f b)"
"execio * diskw opds (stem out. FINIS "
address tso "free f(opds)"
xx = outtrap("OFF")
return
SUMVOLS:
if fnd = Ø then
do
   fnd=fnd + 1
   vollst.fnd = volnam.i
   voltot.fnd = cyldiff.i
   return
end
else
do
   fndflg = Ø
   do p = 1 to fnd
      if  vollst.p = volnam.i then
      do
          voltot.p = voltot.p + cyldiff.i
```

```
            fndflg = 1
         end
      end
      if  fndflg = Ø then
      do
         fnd=fnd+1
         vollst.fnd = volnam.i
         voltot.fnd = cyldiff.i
         fndflg = 1
      end
end
return
VOLCHEK:
say 'Checking 'fnd' volumes for space information ...'
say 'Please wait ...'
trace o
xx = outtrap("zap.","*")
do c=1 to fnd
   Call CHKVTOC vollst.c
   address tso "ALLOC F(SYSP) SHR UNIT(SYSDA) space(2,2)",
               " DSNAME('"||PREFX||"."||USERID()||".SYSPRINT')"
   "execio * diskr SYSP (stem frcyl. FINIS"
   parse VAR frcyl.1 a1 a2 a3 a4 a99
   drop frcyl
   voltot.c = format(voltot.c,6)
   a4 = format(a4,6)
   volout.c = vollst.c||' '||voltot.c||'  '||/4
   address tso "FREE DDNAME(SYSP)"
   address tso "delete '"||PREFX||"."||USERID()||".SYSPRINT'"
   if  c // 2Ø = Ø then
      say 'Processed ' c ' volumes so far ...'
end
fsp_name = PREFX||"."||USERID()||".FRSPC."||dbnode||".D"||us_date
address tso "delete '"fsp_name"'"
address tso "alloc f(fpds) new unit(hsm) space(2,2)",
"cyl reuse dsname('"fsp_name"')",
"dsorg(ps) blksize(312Ø) lrecl(8Ø) recfm(f b)"
say;say 'The free space analysis is written into 'fsp_name
h.1 = '-------------------------'
h.2 = 'Volume  Cylreq    Cylfree'
h.3 = '-------------------------'
"execio * diskw fpds (stem h. "
"execio * diskw fpds (stem volout. "
f.1 = '-------------------------'
"execio * diskw fpds (stem f. FINIS "
address tso "free f(fpds)"
xx = outtrap("OFF")
return
```

Sybase has begun shipping Version 12.0 of each of its Replication Server, EnterpriseConnect, and MainframeConnect products, geared to moving and accessing information from mainframe sources and LAN datastores.

The new versions support access to both DRDA/MVS and international character sets and enable access to foreign datastores. There is now enhanced support for access to DB2 data stores.

Replication Server 12.0 provides near real-time bi-directional data replication across enterprise, client/server, desktop systems, and occasionally-connected systems over a geographically-distributed environment. It has a graphical management and monitoring interface for all components in a replication environment.

EnterpriseConnect 12.0, meanwhile, enables transactional replication between mixed data sources. It supports the necessary database connectivity, including Oracle, Informix, and SQL Server for data movement and AS/400 and DB2/DRDA for data access.

MainframeConnect, finally, provides connectivity between client/server databases and mainframe data, as well as access to DB2/MVS data and on-line production applications in CICS, IMS/TM, and MVS mainframe environments.

For further information contact:
Sybase, 6475 Christie Ave, Emeryville, CA 94608, USA.
Tel: (510) 922 3500.
URL: http://www.sybase.com/products/system11/repsrvr.html.

StorageTek and BMC have jointly announced COPY PLUS for DB2 with Instant Snapshot, RECOVER PLUS for DB2 with Instant SnapShot, and SVA Reporter.

BMC's Enterprise Snapshot and RESOLVE Storage Resource Manager (SRM) adds new functions to StorageTek's recently-announced 9500 Shared Virtual Array subsystem. Its COPY PLUS for DB2 with Instant SnapShot uses the StorageTek virtual disk architecture to create instant copies of a database for recovering information without first restoring from tape.

COPY PLUS for DB2 with Instant SnapShot and RECOVER PLUS for DB2 with Instant SnapShot are said to allow faster copies and data set-level recoveries. SVA Reporter provides reporting for the subsystem, including the ability to report at the file level.

StorageTek will sell the SVA Reporter, based on BMC's RESOLVE SRM, which delivers statistical reporting such as historical details and summary information across OS/390, Unix, and Windows NT systems. It supports both direct-attached storage and storage area networks.

For further information contact:
BMC Software, 2101 CityWest Boulevard, Houston, TX 77042-2827, USA.
Tel: (713) 918 8800.
BMC Software, Compass House, 207-215 London Road, Camberley, Surrey, GU15 3EY, UK.
Tel: (01276) 24622.
URL: http://www.bmc.com/products/prod_directory.html.