107 DB2 update

September 2001

In this issue

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*, and complete issues in Acrobat PDF format, can be downloaded from our Web site at http://www.xephon.com/db2update.html; you will need to supply a word from the printed issue.

Editor

Trevor Eddolls

Contributions

When Xephon is given copyright, 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 download a copy of our *Notes for Contributors* from www.xephon.com/contnote.html.

Reporting concurrent DDF thread usage

My company is currently using DB2 to hold the data generated from several related Web-based applications. The DB2 monitor that we use does not have the capability to historically report how many concurrent DDF threads are active at any one time or who originated the request. And, to add complexity, we have started using DB2 data sharing with the requestors connecting to several different members of the data sharing group that reside on different MVS systems.

I have developed several REXX EXECs and batch jobs to collect this information and store it in sequential files. Our job-scheduling package allows me to run the collect statistics job multiple times separated by a 1-minute or a 4-minute wait. This job is run ten times and then the job to summarize all the data is run. The final form of the data can be imported into MS Excel 97 and allows me to graph the DDF usage to assist in reporting current usage and capacity planning.

COLLECT STATISTICS JOB

The first job consists of four steps. The first step is the DB2 command to display active threads. The second step is a REXX EXEC to remove only the TCP/IP addresses and authorization IDs associated with the each address. It also adds the current date and time to each row.

```
//ADBDDFC1 JOB (1Ø19,XXXX,1),'JLV/S185/ADBDDFC1',
//    CLASS=J,MSGCLASS=T,REGION=6M
//*USERDATA=DDF COUNTS
//* ------------------------------------------------------------------
//* COLLECT DDF COUNTS FROM DBST ON SYS1
//* ------------------------------------------------------------------
//* DB2 DISPLAY THREAD
//* ------------------------------------------------------------------
//DISPLAY1 EXEC  PGM=IKJEFT1A
//SYSTSIN  DD *
  DSN SYSTEM (DBST)
  -DISPLAY THREAD(*)
  END
//SYSTSPRT DD DISP=(,PASS),DSN=&&DISPCMD,
//            SPACE=(TRK,(1,1),RLSE),UNIT=DISK,VOL=SER=DB2DØ3
//SYSPRINT DD SYSOUT=*
//SYSIN    DD *
//* ------------------------------------------------------------------
```

```
//* REMOVE AUTHIDS AND TCP/IP ADDRESSES
//* ------------------------------------------------------------------
//STRIP1   EXEC PGM=IKJEFT1A,COND=(Ø,NE)
//SYSPROC  DD DISP=SHR,DSN=auth5JLW.DB2.PROGRAMS
//         DD DISP=SHR,DSN=ISP.DATECH.CLIST
//SYSPRINT DD SYSOUT=*
//SYSTSPRT DD SYSOUT=*
//IFILE    DD DISP=(OLD,PASS),DSN=&&DISPCMD
//*FILE    DD SYSOUT=*
//OFILE    DD DISP=(,PASS),DSN=&&STRIP,
//         DCB=(RECFM=FB,LRECL=4Ø96),UNIT=DISK,
//         SPACE=(TRK,(1,1),RLSE),VOL=SER=DB2DØ3
//SYSTSIN  DD *
PROF NOPREFIX
DB2SRVR
/*
//* --------------------------------
//* SORT AUTHIDS AND TCP/IP ADDRESSES
//* --------------------------------
//SORT1   EXEC SORTD2,SORTSYS='*',COND=(Ø,NE)
//SORTIN  DD   DISP=(OLD,PASS),DSN=&&STRIP
//SORTOUT DD   DSN=&&SORTO,
//             DISP=(,PASS),
//             UNIT=DISK,SPACE=(TRK,(5,5),RLSE),VOL=SER=DB2DØ3,
//             DCB=(LRECL=8Ø,DSORG=PS)
//SYSIN     DD   *
  SORT FIELDS=(1,8,CH,A,11,12,CH,A)
/*
//* --------------------------------
//* CALL COUNT REXX
//* --------------------------------
//REXXCNT  EXEC PGM=IKJEFT1A,COND=(Ø,NE)
//SYSPROC  DD DISP=SHR,DSN=auth5JLW.DB2.PROGRAMS
//         DD DISP=SHR,DSN=ISP.DATECH.CLIST
//IFILE    DD DISP=(OLD,PASS),DSN=&&SORTO
//SYSTSPRT DD SYSOUT=*
//SYSPRINT DD SYSOUT=*
//*FILE    DD SYSOUT=*
//OFILE    DD DISP=MOD,DSN=DB2.DBST.DDF.COUNTS(Ø)
//SYSTSIN  DD *
PROF NOPREFIX
SRVRSUM
/*
//
```

DB2SRVR

The third step sorts the authorization ID and TCP/IP address pairs. And the last step is a REXX EXEC to summarize the number of

threads coming from each authid-TCP/IP address pairing. The output from the last step is stored in a sequential file for further processing. This job needs to be run separately for each member of the group since they run on separate MVS systems.

```
/* REXX
         db2srvr
         Strip server information from display thread output
         Req'd DD-Cards: IFILE (File with DB2 command output)
                         OFILE (File with server output)
*/
/* trace a */
maxcc = 0 ;
if retc > 0 then signal DB2CMD1_END ;
/* READ DD IFILE        */
retc = read_ifile() ;
if retc > 0 then signal DB2CMD1_END ;
/* INITIALIZE ARRAY VARIABLES */
retc = init_vars() ;
if retc > 0 then signal DB2CMD1_END ;
/* FORMAT OUTPUT RECORDS */
retc = format_orec() ;
if retc > 0 then signal DB2CMD1_END ;
/* WRITE DD OFILE       */
retc = write_ofile() ;
if retc > 0 then signal DB2CMD1_END ;
DB2CMD1_END:
exit(retc) ;                                  /* End program          */
/* ------------------------------------------------------------------
    Proc: read_ifile()
    Read DB2 command output file
   ------------------------------------------------------------------ */
read_ifile:
"Execio * DISKR IFILE (stem irec. finis)" ;
if rc <> 0 then,
do;
   say 'Error Reading Input File (DD IFILE) !!!' ;
   maxcc = 20 ;
   return maxcc ;
end;
say 'Number of Input-Records: 'irec.0 ;
return(0) ;
/* ------------------------------------------------------------------
    Proc: write_ofile()
    Write authid and tcpip output
   ------------------------------------------------------------------ */
write_ofile:
"Execio * DISKW OFILE (stem orec. finis)" ;
if rc <> 0 then,
do;
```

```
   say 'Error Writing Output File (DD OFILE) !!!' ;
   maxcc = 2Ø ;
   return maxcc ;
end;
say 'Number of Output-Records: 'orec.Ø ;
if orec.Ø = Ø then return(4)
else return(Ø);
/* ---------------------------------------------------------------
   Proc: init_vars()
   Initialize array variables that are needed for further processing
   --------------------------------------------------------------- */
init_vars:
do cc = 1 to 5ØØ;
   orec.cc = '' ;                       /* Array of output records   */
end cc ;
return(Ø) ;
/* ---------------------------------------------------------------
    Proc: format_orec()
    Pull authid and tcp/ip address off different input record rows,
    concatenate them, put them to the output record
   --------------------------------------------------------------- */
format_orec:
/* trace r */
cc = 1;
dd = Ø;
bumpflg = Ø;
do while cc <= irec.Ø
   word1 = word(irec.cc,1);
   word2 = word(irec.cc,2);
   /* CHECK AND ADJUST FOR PAGE FEED CHARACTER  */
   if word1 = '1' then do;
      word1 = word(irec.cc,2);
      word2 = word(irec.cc,3);
      bumpflg = 1;
      end;
   if substr(word1,1,1)='1' then word1=substr(word1,2,length(word1)-1);
   /* GET AUTHID */
   if word1 = 'SERVER' & word2 = ‘RA’ then do;
      dd = dd + 1;
      userid = substr(irec.cc,35,8)
      orec.dd = left(userid,1Ø);
      end;
   /* GET TCP/IP ADDRESS */
   if substr(word1,1,4) = 'V445' then do;
      if bumpflg then tcp_addr = word(irec.cc,6)
         else tcp_addr = word(irec.cc,5);
      if tcp_addr = '' then do;
         cc = cc + 1;
         tcp_addr = word(irec.cc,1);
         end;
      orec.dd = orec.dd' 'left(tcp_addr,15);
```

```
      end;
   bumpflg = 0;
   cc = cc + 1;
end;
orec.0 = dd;
/* CHECK FOR NO SERVERS CONNECTED AT THIS TIME */
if orec.0 = 0 then do;
   orec.0 = 1;
   orec.1 = left('no-srvr',10);
   orec.1 = orec.1' 'left('no-addr',15);
end;
return(0) ;
```

SRVRSUM

```
/* REXX
         srvrsum
         sum concurrent threads from distributed users
         Req'd DD-Cards:
             IFILE (sorted file with authids and tcp/ip addresses)
             OFILE (file with authids, tcp/ip addresses, counts)

*/
/* trace a */
maxcc = 0 ;
if retc > 0 then signal DB2CMD1_END ;
/* READ DD IFILE        */
retc = read_ifile() ;
if retc > 0 then signal DB2CMD1_END ;
/* INITIALIZE ARRAY VARIABLES    */
retc = init_vars() ;
if retc > 0 then signal DB2CMD1_END ;
/* COUNT CONCURRENT SERVER USERS  */
retc = count_orec() ;
if retc > 0 then signal DB2CMD1_END ;
/* WRITE DD OFILE       */
retc = write_ofile() ;
if retc > 0 then signal DB2CMD1_END ;
DB2CMD1_END:
exit(maxcc) ;                                  /* End program          */
/* ---------------------------------------------------------------
    Proc: read_ifile()
    Read authid - tcp/ip file
   ---------------------------------------------------------------- */
read_ifile:
"Execio * DISKR IFILE (stem irec. finis)" ;
if rc <> 0 then,
do;
   say 'Error Reading Input File (DD IFILE) !!!' ;
   maxcc = 20 ;
```

```
   return maxcc ;
end;
say 'Number of Input-Records: 'irec.0 ;
return(0) ;
/* ---------------------------------------------------------------
    Proc: write_ofile()
    Write authid, tcp/ip, count output
   --------------------------------------------------------------- */
write_ofile:
"Execio * DISKW OFILE (stem orec. finis)" ;
if rc <> 0 then,
do;
   say 'Error Writing Output File (DD OFILE) !!!' ;
   maxcc = 20 ;
   return maxcc ;
end;
say 'Number of Output-Records: 'orec.0 ;
return(0) ;
/* ---------------------------------------------------------------
   Proc: init_vars()
   Initialize array variables that are needed for further processing
   --------------------------------------------------------------- */
init_vars:
do cc = 1 to 100;
   orec.cc = '' ;                      /* Array of output records   */
end cc ;
return(0) ;
/* ---------------------------------------------------------------
    Proc: count_orec()
    Summarize authid and tcp/ip address pairs, add date and time
   --------------------------------------------------------------- */
count_orec:
/* trace r */
cc = 1;
dd = 0;
cnt = 1;
tcnt = 0;
date_st = left(substr(date('u'),1,10),10)
time_st = left(substr(time(),1,5),6)
rec1 = irec.cc;
do while cc <= (irec.0 - 1)
   cc = cc + 1;
   rec2 = irec.cc;
   if rec1 = rec2 then do;
      /* SAME ID AND ADDRESS, INCREASE COUNT */
      cnt = cnt + 1;
      end;
   else do;
      /* NEW ID OR ADDRESS, CREATE OUTPUT RECORD */
      dd = dd + 1;
      orec.dd = substr(rec1,1,26)' 'right(cnt,4)' 'date_st' 'time_st;
```

```
      rec1 = rec2;
      tcnt = cnt + tcnt;
      /* INIT COUNT FOR NEW ID */
      cnt = 1;
      end;
end;
/* FINISH UP LAST ID */
dd = dd + 1;
orec.dd = substr(rec1,1,26)' 'right(cnt,4)' 'date_st' 'time_st;
tcnt = cnt + tcnt;
if dd > 1 then do;
   dd = dd + 1;
   orec.dd = 'There were '||tcnt||' distributed server connections.'
   orec.Ø = dd;
end;
else if substr(word(orec.1,1),1,2) = 'no' then do;
   orec.dd = substr(rec1,1,26)' 'right('Ø',4)' 'date_st' 'time_st;
   orec.2= 'There were zero distributed server connections at 'time_st;
   end;
else
   orec.2 = 'There was one distributed server connection.'
return(Ø) ;
```

SUMMARIZE DATA JOB

The second job consolidates the information from each of the subsystems in three steps. The first step again sorts the data, this time by the date and time that it was recorded, then by the authid and the TCP/IP address. The second step executes a REXX EXEC that transforms the data from one row per date/time per authid/address into a table that has one row per authid/address with the times as column headings in the first row. All rows that do not have counts for a time slot are zero filled as the data is processed.

```
//ADBDDFG  JOB (1Ø19,XXXX,1),'JLV/S185/DDF GRAPH',
//    CLASS=J,MSGCLASS=T,REGION=6M
//*USERDATA=DDF GRAPH FOR DBST
//* ------------------------------------------------------------------
//* SORT DDF COUNT STATS AND PRODUCE FILE FOR EXPORT TO SPREADSHEET
//*  WITH TEMP FILES PASSED BETWEEN STEPS
//* ------------------------------------------------------------------
//* --------------------------------
//* SORT AUTHIDS AND TCP/IP ADDRESSES
//*      BY TIME THEN AUTHID, TCP/IP
//* --------------------------------
//SORT1   EXEC SORTD2,SORTSYS='*'
//SORTIN  DD   DISP=SHR,DSN=DB2.DBST.DDF.COUNTS(Ø)
//        DD   DISP=SHR,DSN=DB2.DBST.DDF.COUNTS2(Ø)
```

```
//SORTOUT DD   DSN=&&COMBO,
//             UNIT=DISK,SPACE=(TRK,(5,5),RLSE),VOL=SER=DB2DØ3,
//             DISP=(,PASS),DCB=(LRECL=8Ø,DSORG=PS)
//SYSIN     DD   *
  SORT FIELDS=(32,17,CH,A,1,26,CH,A)
/*
//* --------------------------------
//* CALL SUMMARIZE REXX
//* --------------------------------
//SUMRIZE  EXEC PGM=IKJEFT1A
//SYSPROC  DD DISP=SHR,DSN=auth5JLW.DB2.PROGRAMS
//         DD DISP=SHR,DSN=ISP.DATECH.CLIST
//IFILE    DD DISP=(OLD,PASS),DSN=&&COMBO
//SYSTSPRT DD SYSOUT=*
//SYSPRINT DD SYSOUT=*
//OFILE    DD DSN=&&TABL,
//            UNIT=DISK,SPACE=(TRK,(5,5),RLSE),VOL=SER=DB2DØ3,
//            DISP=(,PASS),DCB=(LRECL=2ØØ,DSORG=PS)
//*FILE    DD SYSOUT=*
//SYSTSIN  DD *
PROF NOPREFIX
DB2GRAPH
/*
//* ----------------------------------------
//* SORT TABLE BY AUTHID AND TCP/IP ADDRESSES
//* ----------------------------------------
//SORT2   EXEC SORTD2,SORTSYS='*'
//SORTIN  DD   DISP=(OLD,PASS),DSN=&&TABL
//SORTOUT DD   DSN=DB2.DBST.DDF.GRAPH(+1),
//             DISP=(,CATLG,DELETE),
//             UNIT=DISK,SPACE=(TRK,(5,5),RLSE),VOL=SER=DB2DØ3,
//             DCB=(GDG.MODEL,LRECL=2ØØ,DSORG=PS)
//SYSIN     DD   *
  SORT FIELDS=(1,26,CH,A)
/*
//* NEW GDG'S FOR NEXT RUN
//CNTS1  EXEC PGM=IEBGENER
//SYSPRINT DD SYSOUT=*
//SYSIN  DD DUMMY
//SYSUT1   DD  DUMMY,DCB=(RECFM=FB,LRECL=8Ø,BLKSIZE=2792Ø)
//SYSUT2   DD  DSN=DB2.DBST.DDF.COUNTS(+1),DISP=(,CATLG),
//             DCB=(GDG.MODEL,RECFM=FB,LRECL=8Ø,BLKSIZE=2792Ø,
//             DSORG=PS),UNIT=DISK,SPACE=(TRK,(1,1))
//*
//CNTS2  EXEC PGM=IEBGENER
//SYSPRINT DD SYSOUT=*
//SYSIN  DD DUMMY
//SYSUT1   DD  DUMMY,DCB=(RECFM=FB,LRECL=8Ø,BLKSIZE=2792Ø)
//SYSUT2   DD  DSN=DB2.DBST.DDF.COUNTS2(+1),DISP=(,CATLG),
//             DCB=(GDG.MODEL,RECFM=FB,LRECL=8Ø,BLKSIZE=2792Ø,
//             DSORG=PS),UNIT=DISK,SPACE=(TRK,(1,1))
```

```
//*
//
```

DB2GRAPH

```
/* REXX
         db2graph
         restructure ddf counts for excel graphs
         Req'd DD-Cards:
             IFILE (file with authids, tcp/ip addresses, counts,
                    dates, and times in chronological order)
             OFILE (summary file with one row per authid-tcp/ip
                    address and multiple time columns)
*/
/* trace a */
maxcc = Ø ;
if retc > Ø then signal DB2CMD1_END ;
/* READ DD IFILE        */
retc = read_ifile() ;
if retc > Ø then signal DB2CMD1_END ;
/* INITIALIZE ARRAY VARIABLES */
retc = init_vars() ;
if retc > Ø then signal DB2CMD1_END ;
/* SUMMARIZE DATA       */
retc = summary() ;
if retc > Ø then signal DB2CMD1_END ;
/* WRITE DD OFILE       */
retc = write_ofile() ;
if retc > Ø then signal DB2CMD1_END ;
DB2CMD1_END:
exit(maxcc) ;                                  /* End program          */
/* ---------------------------------------------------------------
    Proc: read_ifile()
    Read sorted authid, tcp/ip, time file
   --------------------------------------------------------------- */
read_ifile:
"Execio * DISKR IFILE (stem irec. finis)" ;
if rc <> Ø then,
do;
   say 'Error Reading Input File (DD IFILE) !!!' ;
   maxcc = 2Ø ;
   return maxcc ;
end;
say 'Number of Input-Records: 'irec.Ø ;
return(Ø) ;
/* ---------------------------------------------------------------
    Proc: write_ofile()
    Write matrix table output
   --------------------------------------------------------------- */
write_ofile:
```

```
"Execio * DISKW OFILE (stem orec. finis)" ;
if rc <> Ø then,
do;
   say 'Error Writing Output File (DD OFILE) !!!' ;
   maxcc = 2Ø ;
   return maxcc ;
end;
say 'Number of Output-Records: 'orec.Ø ;
return(Ø) ;
/* ---------------------------------------------------------------
   Proc: init_vars()
   Initialize array variables that are needed for further processing
   ---------------------------------------------------------------- */
init_vars:
do cc = 1 to 1ØØ;
   orec.cc = '' ;                        /* Array of output records   */
end cc ;
return(Ø) ;
/* ---------------------------------------------------------------
    Proc: summary()
    Rearrange sorted output for easy export to spreadsheet
   ---------------------------------------------------------------- */
summary:
/* trace a */
cc = 1;
orec.1 = '<authid>  <tcp/ip addr>    '
rec1 = irec.cc;
/* skip possible leading messages */
do while (substr(rec1,1,5) = 'There') && (substr(rec1,1,1) = ' ')
   cc = cc + 1;
   rec1 = irec.cc;
end;
/* INIT VARIABLES */
timecol = 3;
newtime = word(irec.cc,5);
oldtime = newtime;
auth_cnt = 1;
/* ADD FIRST TIME COLUMN TO OUTPUT RECORD */
orec.1 = orec.1' 'right(word(irec.cc,5),5);
orec.2 = substr(rec1,27)||newtime;
do while cc < (irec.Ø - 1)
   do while (oldtime = newtime)
      new_id = 1;
      iauth = substr(irec.cc,1,26);
      /* SEARCH FOR ID IN ARRAY */
      do loop = 2 to auth_cnt + 1;
         old_auth = substr(orec.loop,1,26);
         if old_auth = iauth then do;
            /* add count */
            orec.loop = orec.loop' 'right(word(irec.cc,3),5);
            new_id = Ø;
```

```
            leave;
            end;
         end;
      /* ADD NEW ID WITH ZEROS IN PREVIOUS TIME COLUMNS */
      if new_id then do;
         new_id = 0;
         newrow = auth_cnt + 1;
         orec.newrow = iauth;
         do y = 3 to timecol - 1;
            orec.newrow = orec.newrow' 'right(0,5);
            end;
         orec.newrow = orec.newrow' 'right(word(irec.cc,3),5);
         auth_cnt = auth_cnt + 1;
         end;
      oldtime = newtime;
      cc = cc + 1;
      newtime = word(irec.cc,5);
      end;
   /* FILL TIME COL W/ZERO FOR AUTHS NOT FOUND IN THIS TIME PERIOD */
   do loop = 2 to auth_cnt;
      if words(orec.loop) < timecol then do;
         /* ADD ZERO COUNT */
         orec.loop = orec.loop' 'right(0,5);
         end;
      end;
   if (substr(irec.cc,1,5) = 'There') then leave;
   timecol = timecol + 1;
   orec.1  = orec.1' 'right(newtime,5);
   oldtime = newtime;
end;
orec.0 = auth_cnt;
return(0) ;
```

The last step is to sort the data, once more, into authid/address order and put the data into a sequential file.

I then manually export this file to my PC where I import it into MS Excel 97. It creates the spreadsheet used by Chart Wizard to produce a bar graph of the concurrent usage.

This code was written and debugged with DB2 Version 5.1 and OS/390 2.5. During the development process we migrated to DB2 Version 6.1 and OS/390 2.8. Syncsort Version 3.6 was used to sort the sequential files. It does not count threads that are available to be assigned as type 2 inactive threads (new in DB2 Version 6.1) as active.

Jeannette L Vincent
Technical Support Consultant
First Allmerica Financial (USA)

DB2 parallel operations and query performance

Many of my *DB2 Update* articles have discussed using DB2 for data warehousing or OLAP. Each function can generate complex queries that require significant execution time. Parallel operations can noticeably improve performance.

DEFINITIONS

This article may use unfamiliar terms. Their definitions are:

- Degree of parallelism – number of parallel tasks or I/O operations that DB2 can use in a parallel group.
- Parallel group – unique ID for naming a specific parallel operation set. A query can have many groups but each will have the same ID.
- Task (CP or Sysplex query) – actual MVS execution unit.
- Task (parallel I/O streams) – processing of an identifiable concurrent I/O stream.
- Query I/O parallelism – concurrent I/O requests for a *single* query fetching pages into the buffer pool in parallel. (I/O parallelism is only used if CP or Sysplex is unavailable.)
- Query CP parallelism – *multi-tasking* within a large query by splitting it into smaller queries running simultaneously on multiple tightly-coupled processors.
- Sysplex query parallelism – *multi-tasking* across different members in a data sharing group.

WHAT PARALLELISM CAN BE USED FOR

Parallelism can be employed in static and dynamic queries, local and remote data access, queries using single table scans and multi-table joins, access via an index or table scan, or list prefetch, sort, and

inserts. Most parallel operations can be used on either partitioned or non-partitioned table spaces. Non-clustering indexes may be used.

WHY PARALLELISM SHOULD BE USED

Assume access to a table space with three partitions (P1, P2, and P3).

Sequential processing requires DB2 to complete P1 before starting P2, which must also be completed before starting P3. Sequential prefetch permits CP processing with I/O operations, but I/O operations do *not* overlap. Prefetchs can take longer if the processor is waiting for I/O.

Parallel I/O processing allows DB2 to prefetch data from all partitions *simultaneously*, but there is only *one* processing task.

Parallel CP processing lets DB2 use *multiple* parallel tasks; Sysplex improves parallel performance by permitting DB2 to cross a single CPC boundary.

WHAT ARE THE BEST QUERY TYPES FOR PARALLEL PROCESSING?

The best query types for parallel processing, in no particular order, are I/O intensive, intensive data scans with high selectivity, use aggregate functions, access long data rows particularly when there is one row per page, and complex read-only that is data-intensive or uses sort. Read-only cursor must be *unambiguous* so FOR FETCH/READ ONLY must be specified in the OPEN CURSOR declaration.

Queries that do *not* use parallelism include those accessing temporary tables, using merge scan join on more than one column, using direct row access (D in PRIMARY_ ACCESS_TYPE column in PLAN_TABLE), materializing views, or containing EXISTS within WHERE.

How do you enable parallel processing?

You can enable parallel processing by:

- Static SQL – specify DEGREE (ANY) on BIND or REBIND.
- Dynamic SQL – set CURRENT DEGREE special register to 'ANY'.
- Entire DB2 – set CURRENT DEGREE to ANY in panel DSNTIP4.

Warnings

The VPPSEQT (virtual buffer pool parallel sequential threshold) value must be sufficient to provide adequate space for parallel processing. CP or Sysplex can run only on a central processor complex containing two or more tightly-coupled CPs.

How do you disable or limit parallel processing?

- SQL – set DEGREE BIND or CURRENT DEGREE special register.
- Entire DB2 – set MAX DEGREE in panel DSNTIP4.

EXPLAIN

The PLAN_TABLE provides hints on whether DB2 plans to use parallelism:

- A non-null value in each query block (QBLOCKNO) in a query (QUERYNO) indicates that some degree of parallelism is planned.
- All steps (PLANNO) with the same value for ACCESS_PGROUP_ID, JOIN_ PGROUP_ID, JOIN_PGROUP_ID, SORTN_PGROUP_ID, or SORTC_PGROUP _ID specifies that there is an operation set in the same parallel group using various join methods and sort operations. (Warning: parallel group IDs can appear in the same row or in *different* rows.)
- PARALLELISM_MODE defines whether the planned parallelism is I, C, or X. A single query block cannot have a mixture. Some statements, such as UNION, can have more than one query block and therefore a mixture.

WHAT ARE THE BEST INSERT TYPES FOR PARALLEL PROCESSING?

Buffered INSERT can be used only in partitioned environments. It needs an application that uses a *single* INSERT with no other DML. The INSERT must be within a loop that adds many rows using the VALUES clause as the data source. The VALUES clause can specify single or multiple rows.

How do you enable buffered INSERT?

Use the PREP command to process the application program source file or use the BIND command on the resulting file to enable buffered INSERT. Either method requires specifying the INSERT BUF option. Buffered INSERT will *not* be used if VALUES has long fields or LOBs. A fullselect INSERT does *not* benefit from parallelism.

An application program *must be* used; CLP INSERT using EXECUTE IMMEDIATE does *not* work.

Warnings

Rows are inserted *asynchronously*. That means errors may be reported on a subsequent INSERT or any other statement that closes the buffer. Rows may *not* be visible immediately. Do episodic commits to avoid filling the transaction log.

OTHER PERFORMANCE ENHANCEMENTS FOR PARTITIONED ENVIRONMENTS

Unambiguous read-only cursors

Specifying FOR READ/FETCH ONLY in the OPEN CURSOR declaration allows the coordinator partition to retrieve multiple rows. Failure to stipulate it forces DB2 to treat the cursor as *updateable*, meaning it gets only a *single* row per FETCH. Specification will improve performance on *single* processors as well as *multi* processor parallelism.

Directed DSS (Distributed SubSection)

A Directed DSS uses the table partition key to send the query or fragment to the *single* partition needed for its processing. This avoids a query broadcast to *all* nodes. A typical SELECT template:

```
     SELECT    …
FROM      table1
     WHERE     PARTKEY=:hostvar
```

Directed DSS requires that complex queries be divided into multiple simple queries with a *single* host variable or a single SELECT using UNION:

```
     SELECT    …
     FROM      table1
     WHERE     PARTKEY=:hostvar1
     UNION
     SELECT    …
FROM      table1
     WHERE     PARTKEY=:hostvar2
```

Local Bypass

A *very* Directed DSS is called a Local Bypass. It involves sending the query to the coordinator partition thereby eliminating any partition intercommunication. Two techniques are:

- Have a remote client maintain connections to each partition.
- Group transactions by partition with each having a separate application server.

You can determine the partitions where the data resides by using the SQLUGRPN API (Get Row Partitioning Number) and code your program appropriately. You can also use the DB2ATLD utility to divide input by partition followed by running an application copy for each.

REFERENCES

The author suggests that each site has access to SBOF-8931. Useful books include:

- *Administration Guide: Planning*
- *Administration Guide: Performance*
- *Administrative API Reference*
- *Application Development Guide*
- *Command Reference*
- *Data Warehouse Center: Application Guide*
- *Data Warehouse Center: Application Integration Guide*
- *What's New.*

Eric Garrigue Vesely
Principal/Analyst
Workbench Consulting (Malaysia)

Simplifying occasional, regular, and periodic tasks of the DBA – revisited

In our article *Simplifying occasional, regular, and periodic tasks of the DBA* published in *DB2 Update*, Issues 105 and 106 (July and August 2001) we noticed that the code needs a correction in one line. In MAINTS2, in the SORT04 part (page 48, Issue 105), the line containing the SORTIN DD statement should be replaced with:

```
//SORTIN    DD  DISP=(OLD,PASS),DSN=&&EXTEXTR
```

We are sorry for the error.

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

Recover tablespaces

INTRODUCTION

This article is an implementation of the *Imagecopy generator procedure* article already published in *DB2 Update* Issue 106, August 2001. In this article I will describe a tool that provides an easy way to recover DB2 tablespaces.

PROCEDURE DESCRIPTION

The tool was developed to allow the user, with the right authority, to carry out the recovery of their own objects. The tool executes a query on the DB2 catalog and displays the ImageCopy available for recovery, then, through a simple selection, builds the recovey job.

CHECKLIST FOR INSTALLATION

Follow these steps to install the components of the REXX procedure.

Use the preallocated USER.LIBRARY (the USER.LIBRARY allocated for the *Imagecopy generator procedure* article).

Copy the following REXX, help panel, and panel members into the USER.LIBRARY:

- REXX – $DB2TL00, $DB2TL0A, and $DB2ACC1.
- Panel – $DB2P$$$, $DB2P000, $DB2P008, and $DB2P009.
- Help panel – $DB2H000, $DB2H001, $DB2H015, and $DB2H016.

Run the REXX $DB2TL00 and using the online help make your request.

The test environment is DB2 Version 5 in an OS/390 Version 8 environment.

$DB2TL00 REXX EXEC

```
/* REXX */
trace ?o
   /*---   Managment Tool Main menu            ---*/
 blk =  ; wrexit = ok ; cur00 ='@db2subs' ; acc = NON
 @db2subs = DSN?; @db2msg =   ; @db2data = date(e) ; @db2tim = time()
 RunEnv = 'ONLINE'
 address ispexec "display panel(@db2p$$$)"
 if rc = 8 then exit
 do forever
    @db2data = date(e) ; @db2tim  = time() ; wrexit = ok
    address ispexec "display panel(@db2p000) cursor("cur00")"
    @db2msg =
    if rc = 8 then leave ; call ParAssign
    if wrexit = ok then do
       acc = HIGH ; call ParAssign ; @db2ver = $db2ver
       end
    if wrexit  = ok & @db2opt = blk then do
       @db2msg = 'Select one Option' ; cur00 = '@db2opt' ; wrexit = ko
       iterate
       end
    if wrexit   = ok & @db2opt ¬= blk then do
       parmpass = @db2subs','@db2ver
       select
          when @db2opt = A then do      /*- Recover Tablespace  -*/
             call #DB2TL0A parmpass ; @db2msg = result ; @db2opt =
             end
    /*IMPL00*/
          otherwise
             @db2msg = 'Unpredictable error contact Support Staff  !!!'
             @db2opt =
             iterate
          end
       end
    end
 exit
ParAssign :
   /*---  Parameters assignment               ---*/
 call @db2par0 @db2subs RunEnv
 if word(result,1) = 99 then do
    $db2msg = 'Wrong DB2 subsystem '$db2subs
    cur00 = '$db2subs' ; acc = NON ; $db2ver = blk ; wrexit = ko
    return
    end
 $lpar    =word(result,1) ;$accn   =word(result,2) ;$class
=word(result,3)
 $msgcla  =word(result,4) ;$region =word(result,5) ;$msglvl
=word(result,6)
 $notif   =word(result,7) ;$user   =word(result,8) ;$unitda
=word(result,9)
```

```
 $unitta  =word(result,10);$esunit =word(result,11);$prt
=word(result,12)
 $hiwork  =word(result,13);$db2ver =word(result,14);$ctsubs
=word(result,15)
 $librexx =word(result,16);$parmlib=word(result,17);$proclib
=word(result,18)
 $jcllib  =word(result,19);$report =word(result,20);$libexec
=word(result,21)
 $isptenu =word(result,22);$isppenu=word(result,23);$ispmenu
=word(result,24)
 $ispslib =word(result,25);$plilink=word(result,26);$sibmlnk
=word(result,27)
 $sortlib =word(result,28);$runlib =word(result,29);$dsnload
=word(result,30)
 $tep2pgm =word(result,31);$tep2pln=word(result,32);$unlopgm
=word(result,33)
 $unlopln =word(result,34);$dunlopg=word(result,35);$dunlopl
=word(result,36)
 $dsnproc =word(result,37)
 return
```

$DB2TL0A REXX EXEC

```
/* REXX */
trace ?o
   /*-        Recover Tablespace TOCOPY          -*/
arg parmin ; parm   = translate(parmin,' ',',')
nparm  = words(parm) ; @db2subs = word(parm,1) ; @db2ver = word(parm,2)
   /*---  Parameters assignment                ---*/
 call @db2par0 @db2subs ; if word(result,1) = 99 then exit
 $lpar    =word(result,1) ;$accn   =word(result,2) ;$class
=word(result,3)
 $msgcla  =word(result,4) ;$region =word(result,5) ;$msglvl
=word(result,6)
 $notif   =word(result,7) ;$user   =word(result,8) ;$unitda
=word(result,9)
 $unitta  =word(result,10);$esunit =word(result,11);$prt
=word(result,12)
 $hiwork  =word(result,13);$db2ver =word(result,14);$ctsubs
=word(result,15)
 $librexx =word(result,16);$parmlib=word(result,17);$proclib
=word(result,18)
 $jcllib  =word(result,19);$report =word(result,20);$libexec
=word(result,21)
 $isptenu =word(result,22);$isppenu=word(result,23);$ispmenu
=word(result,24)
 $ispslib =word(result,25);$plilink=word(result,26);$sibmlnk
=word(result,27)
 $sortlib =word(result,28);$runlib =word(result,29);$dsnload
=word(result,30)
```

```
 $tep2pgm =word(result,31);$tep2pln=word(result,32);$unlopgm
=word(result,33)
 $unlopln =word(result,34);$dunlopg=word(result,35);$dunlopl
=word(result,36)
 $dsnproc =word(result,37)
   /*---   Work areas initialization           ---*/
 blk   =  ; lwork = Ø ; wrk = Ø ; delon = off ; curØØ = '@db2db'
 $yyyy = substr(date(s),1,4) ; $mm = substr(date(s),5,2)
 $mm  = $mm - 3 ; if $mm < 1 then $mm = 1
 $gg   = substr(date(s),7,2)
 @db2tims = $yyyy'-'right($mm,2,'Ø')'-'right($gg,2,'Ø')'-
ØØ.ØØ.ØØ.ØØØØØØ'
   /*---  Display Recover tablespace panel     ---*/
 call Free
 do forever
    @db2data = date(e) ; @db2ora  = time() ; wrexit = ok
    address ispexec "display panel(@db2pØØ8) cursor("curØØ")"
    @db2msg =
    if @db2db = 'DSNDBØ4'  | @db2db = 'DSNDBØ6'  | ,
       @db2db = 'DSNDBØ7'  | @db2db = 'DSQ1STBB' | ,
       @db2db = 'DSNRGFDB' | @db2db = 'DSNRLST'  | ,
       @db2db = 'DSQDBCTL' | @db2db = 'DSQDBDEF'  then do
       @db2msg = 'No SYSTEM DataBase allowed !!!! '
       curØØ = '@db2db' ; wrexit = ko
       end
    if rc = 8 then do
       if delon = on then call FreeØ
       curØØ = '@db2subs' ; @db2opt = ; @db2msg =  ; return  @db2msg
       end
    if wrexit = ok then call SelDB2
    end
 exit
   /*---   DB2 Routine                         ---*/
 SelDB2 :
   /*---  File bridge name                     ---*/
    delon = on
    outdstsin = $hiwork'.'@db2subs'.'$user'.SYSTSIN'
    outdsprt  = $hiwork'.'@db2subs'.'$user'.SYSPRINT'
    outdsrec  = $hiwork'.'@db2subs'.#DB2TLØA.SYSRECØØ'
   /*---  SYSIN file allocation                ---*/
    outdsin= $hiwork'.'@db2subs'.'$user'.SYSIN'
    dsnchk = sysdsn(''''outdsin'''')
    if dsnchk ¬= OK then do
       prmalloc = @db2subs' 'outdsin' Ø 5,1 f,b 8Ø 2792Ø sysin no'
       call @db2allØ prmalloc ; if word(result,1) = 99 then exit
       end
    else "alloc da('"outdsin"') f("sysin") shr reuse"
sk.1=' SELECT  TSNAME,DSNAME,DSNUM,FILESEQNO,DEVTYPE,SHRLEVEL        '
sk.2='        ,"TIMESTAMP",ICBACKUP,ICTYPE,ICUNIT,DSVOLSER           '
sk.3=' FROM SYSIBM.SYSCOPY                                           '
sk.4='      WHERE DBNAME =    '''@db2db'''                           '
```

```
    ldb2tbsp = length(@db2tbsp)
    silike = substr(@db2tbsp,ldb2tbsp,1)
    if silike ='%' | silike = '*' then do
       @db2tbsp = translate(@db2tbsp,'%','*')
sk.5='        AND TSNAME LIKE '''@db2tbsp'''                          '
       end
    else
sk.5='        AND TSNAME =    '''@db2tbsp'''                          '
sk.6='        AND ICTYPE =    ''F''                                   '
sk.7='        AND TIMESTAMP >= '''@db2tims'''                         '
sk.8='  ORDER BY TIMESTAMP DESC ;                                     '
    sk.0=8; jobw = sysin ; call WriteRec
    call @db2acc1 @db2subs',#DB2TL0A'
    RCdb2  = word(result,1) /* DB2 access RC                */
    okunlo = word(result,2) /* Accesso OK or Cursor Position */
    NRrec  = word(result,3) /* Nr. Record or Message         */
   /*---  DB2 access OK start process          ---*/
    if RCdb2 = 00 then do
       if NRrec > 0 then do
          call FillTab
          end
       else do
          @db2msg = 'No imagecopy available for Database/Tablespace/
Timestamp se
lected'
          cur00 = '@db2db' ; call Free ; return
          end
       end
    else do
       cur00 = okunlo ; @db2msg = translate(NRrec,' ','_') ; call Free
       end
    return
   /*---  Routine fill up ISPF/TAB             ---*/
 FillTab:
   /*---  Read extracted Records               ---*/
    jobw = sysrec00
    "alloc da('"outdsrec"') f("jobw") shr reuse"
    xx=outtrap(trpread01.)
       "execio * diskr sysrec00 (stem sysrec00. finis"
    xx=outtrap(off)
    if rc > 0 then do
       do #a = 1 to trpread01.0
          say trpread01.#a
       end
       say '' ; say '' ; say '>>>>>>>>'
       say '>>>>>>>>  Error reading file "'outdsrec'"    '
       say '>>>>>>>>  RC='rc'. Verify.                   '
       say '>>>>>>>>' ; say '' ; say '' ; call Free ; exit
       end
   /*---  TBcreate TDISIMG0                    ---*/
```

```
 address ispexec
"tbcreate tdisimg0 names($ts $tim $ds $fil $sh $icu $dsvo)
 nowrite replace"
 DO #d = 1 to sysrec00.0
    $ts     = strip(substr(sysrec00.#d,1,8))          /* Tsname    */
    $tim    = strip(substr(sysrec00.#d,70,26))        /* Timestamp */
    $ds     = strip(substr(sysrec00.#d,9,44))         /* Dsname    */
    $fil    = x2d(d2x(c2d(substr(sysrec00.#d,57,4)))) /* Filseqno  */
    $sh     = strip(substr(sysrec00.#d,69,1))         /* Shrlevel  */
    select
       when $sh = 'R' then $sh = 'Reference'
       when $sh = 'C' then $sh = 'Change'
       otherwise
          @db2msg= ' Shrlevel error. End elaboration'
          exit
          end
    $ictype = strip(substr(sysrec00.#d,98,1))         /* Ictype    */
    select
       when $ictype = 'F' then $ictype = 'Copy Full yes'
       when $ictype = 'I' then $ictype = 'Copy Full no'
       otherwise
          @db2msg = ' Ictype error. End elaboration'
          exit
          end
    $icu = strip(substr(sysrec00.#d,99,1))            /* Icunit    */
    select
       when $icu = 'T' then $icu = 'TAPE'
       when $icu = 'D' then $icu = 'DASD'
       otherwise
          @db2msg = ' Icunit error. End elaboration'
          exit
          end
    $ldsvolser = d2x(c2d(substr(sysrec00.#d,100,2)))  /* Dsvolser  */
    $dsvo      = translate(strip(substr(sysrec00.#d,102,20)),' ',',')
    $dsvo      = translate($dsvo,' ','_')
    "tbmod tdisimg0"
    end  /* END DO #d = 1 to sysrec00.0 */
 "tbsort tdisimg0 fields($ts,c,a,$tim,c,d)"
 cur00 =  '@db2sel'
 Do forever
    @db2sel = ' '
    "tbtop  tdisimg0"
    "tbdispl tdisimg0 panel(@db2p009) cursor("cur00")"
    if rc = 8 then do
       call Free ; cur00   = '@db2subs' ; @db2opt =  ; @db2msg =
       return  @db2msg
       end
    select
    when ZTDSELS = 1 then do
       data=substr($tim,9,2)'/'substr($tim,6,2)'/'substr($tim,1,4)
```

```
          ora =substr($tim,12,2)':'substr($tim,15,2)':'substr($tim,18,2)
          Call Wrrecjob
          if translate(@db2sel) ¬= E & translate(@db2sel) ¬= R then do
             @db2msg = 'Choice error ||| Enter E(Edit) R(Recover)'
             iterate
             end
          if translate(@db2sel) = E then do
             address tso "ispexec edit dataset('"outdsØ"')"
             @db2msg = 'Edit Recovery job ended successfully '
             end
          if translate(@db2sel) = R then do
             address tso "submit '"outdsØ"'"
             @db2msg = 'Recovery job for '@db2db'/'$ts' to 'data' 'ora'
                                                 has been submitted.'
             end
          call Free ; curØØ = '@db2db' ; return
          end
       when ZTDSELS = Ø then do
          @db2msg  = 'Select one row' ; curØØ = '@db2sel'
          end
       otherwise
          @db2msg = 'Select only one row !!!!'
          @db2sel = ' ' ; curØØ = '@db2sel'
       end
    end  /* End Do forever */
    return
   /*---  Routine write Recover TOcopy         ---*/
 Wrrecjob :
    outdsØ= $hiwork'.'@db2subs'.'$user'.#DB2TLØA.JOBREC'
    prmalloc = @db2subs' 'outdsØ' Ø 1,1 f,b 8Ø 2792Ø fijob no'
    call @db2allØ prmalloc ; if word(result,1) = 99 then exit
sk.1='//'userid()'Y  JOB ('$accn'),''DB2-Gestione'',CLASS='$class',   '
sk.2='//     MSGCLASS='$msgcla',REGION='$region',MSGLEVEL=('$msglvl'),'
sk.3='//     NOTIFY='$notif
sk.4='/*JOBPARM BYTES=999999,LINES=999999                             '
sk.5='//* ------------------------------------------------- *       '
sk.6='//* -------        Recover Tablespace          -------- *       '
sk.7='//* ------------------------------------------------- *       '
       lwork = length(@db2db) ; wrk = 3Ø - lwork
sk.8='//* -- DataBase     : '@db2db copies(' ',wrk)'-- *'
       lwork = length($ts) ; wrk = 3Ø - lwork
sk.9='//* -- Tablespace   : '$ts copies(' ',wrk)'-- *'
       lwork = length(data) ; wrk = 3Ø - lwork
sk.1Ø='//* -- Copy Date    : 'data copies(' ',wrk)'-- *'
       lwork = length(ora) ; wrk = 3Ø - lwork
sk.11='//* --   "  Time    : 'ora copies(' ',wrk)'-- *'
       lwork = length($ictype) ; wrk = 3Ø - lwork
sk.12='//* --   "  Type    : '$ictype copies(' ',wrk)'-- *'
       lwork = length($sh) ; wrk = 3Ø - lwork
sk.13='//* --   "  Shrlevel: '$sh copies(' ',wrk)'-- *'
       lwork = length($icu) ; wrk = 3Ø - lwork
```

```
sk.14='//* --   "  Unit    : '$icu copies(' ',wrk)'-- *'
       if $icu = TAPE then do
          lwork = length($dsvo) ; wrk = 3Ø - lwork
sk.15='//* --   "  Volser  : '$dsvo copies(' ',wrk)'-- *'
          end
       else
sk.15='//* ------------------------------------------------- *      '
sk.16='//* ------------------------------------------------- *      '
sk.17='//DB2PROC   JCLLIB ORDER=('$proclib')                        '
sk.18='//JOBLIB    DD  DISP=SHR,DSN='$dsnload
sk.19='//* ------------------------------------------------- *      '
sk.2Ø='//* -----------    Start  Tablespace "UT"   ----------- *      '
sk.21='//* ------------------------------------------------- *      '
sk.22='//STRTUT   EXEC PGM=IKJEFTØ1                                 '
sk.23='//SYSTSPRT  DD  SYSOUT=*                                     '
sk.24='//SYSTSIN   DD  *                                            '
sk.25='    DSN SYSTEM('@db2subs')                                   '
sk.26='  -START DB('@db2db') SPACENAM('$ts') ACCESS(UT)           '
sk.27='//* ------------------------------------------------- *      '
sk.28='//* -----------          RECOVER            ----------- *      '
sk.29='//* ------------------------------------------------- *      '
sk.3Ø='//LAB1     IF (STRTUT.RC EQ Ø) THEN                          '
sk.31='//RECOVER  EXEC '$dsnproc',PARM='''@db2subs','@db2db||$ts''''
sk.32='//SYSIN    DD  *                                             '
sk.33='RECOVER TABLESPACE '@db2db'.'$ts
sk.34='        TOCOPY '$ds
sk.35='REBUILD INDEX(ALL) TABLESPACE '@db2db'.'$ts
sk.36='        SORTDEVT '$esunit' SORTNUM 4                        '
sk.37='//* ------------------------------------------------- *      '
sk.38='//* -----------    Start  Tablespace "RW"   ----------- *      '
sk.39='//* ------------------------------------------------- *      '
sk.4Ø='//LAB2     IF (RECOVER.'$dsnproc'.RC LT 5) THEN              '
sk.41='//STRTRW   EXEC PGM=IKJEFTØ1                                 '
sk.42='//SYSTSPRT  DD  SYSOUT=*                                     '
sk.43='//SYSTSIN   DD  *                                            '
sk.44='    DSN SYSTEM('@db2subs')                                   '
sk.45='  -START DB('@db2db') SPACENAM('$ts') ACCESS(RW)           '
sk.46='//LAB1END  ENDIF                                             '
sk.47='//LAB2END  ENDIF                                             '
sk.48='//* ------------------------------------------------- *      '
    sk.Ø=48 ; jobw = fijob ; call WriteRec ; return
   /*---  Routine write record output           ---*/
 WriteRec :
    address tso "EXECIO * DISKW "jobw" (STEM sk. FINIS"
 ClearRec:
    DO f = 1 to sk.Ø
       sk.f = blk
       end
    return
   /*---  Free & Delete work DataSet            ---*/
```

```
 Free0 :
    xx=outtrap(trpdummy.)
       address tso
       "free fi(systsinp)"   ; "free fi(sysprint)" ; "free fi(sysrec00)"
       "free fi(syspunch)"   ; "free fi(sysin)"    ; "free fi(fijob)"
       "delete '"outdsin"'"  ; "delete '"outds0"'" ; "delete
'"outdsprt"'"
       "delete '"outdsrec"'" ; "delete '"outdstsin"'"
    xx=outtrap(off) ; return
   /*---  Free work DataSet                    ---*/
 Free  :
    xx=outtrap(trpdummy.)
       address tso
       "free fi(systsinp)"  ; "free fi(sysprint)"  ; "free fi(sysrec00)"
       "free fi(syspunch)"  ; "free fi(sysin)"  ; "free fi(fijob)"
    xx=outtrap(off) ; return
```

$DB2ACC1 REXX EXEC

```
/* REXX */
trace ?o
   /*-        DB2 Online Access Routine          -*/
arg parmin ; parm = translate(parmin,' ',',')
nparm    = words(parm) ; Subsys = word(parm,1) ; ComeFrom = word(parm,2)
   /*---  Test input parameters                ---*/
 if nparm < 2 then do
    say '' ; say '' ; say '>>>>>>>>'
    say '>>>>>>>>  Parameter string is incomplete !!!!'
    say '>>>>>>>>           'parmin
    say '>>>>>>>>' ; say '' ; say '' ; $exitc = 99 ; return $exitc
    end
   /*---  Parameters assignment                ---*/
 call @db2par0 Subsys
 if word(result,1) = 99 then do
    $exitc = 99 ; return $exitc
    end
 $lpar    =word(result,1) ;$accn   =word(result,2) ;$class
=word(result,3)
 $msgcla  =word(result,4) ;$region =word(result,5) ;$msglvl
=word(result,6)
 $notif   =word(result,7) ;$user   =word(result,8) ;$unitda
=word(result,9)
 $unitta  =word(result,10);$esunit =word(result,11);$prt
=word(result,12)
 $hiwork  =word(result,13);$db2ver =word(result,14);$ctsubs
=word(result,15)
 $librexx =word(result,16);$parmlib=word(result,17);$proclib
=word(result,18)
 $jcllib  =word(result,19);$report =word(result,20);$libexec
```

```
=word(result,21)
 $isptenu =word(result,22);$isppenu=word(result,23);$ispmenu
=word(result,24)
 $ispslib =word(result,25);$plilink=word(result,26);$sibmlnk
=word(result,27)
 $sortlib =word(result,28);$runlib =word(result,29);$dsnload
=word(result,3Ø)
 $tep2pgm =word(result,31);$tep2pln=word(result,32);$unlopgm
=word(result,33)
 $unlopln =word(result,34);$dunlopg=word(result,35);$dunlopl
=word(result,36)
 $dsnproc =word(result,37)
   /*---   Work areas initialization           ---*/
 blk  =  ; call free
   /*---  SYSTSIN file allocation              ---*/
 outdstsin= $hiwork'.'subsys'.'$user'.SYSTSIN'
 dsn = sysdsn('''outdstsin''')
 if dsn ¬= OK then do
    prmalloc = subsys' 'outdstsin' Ø 5,1 f,b 8Ø 2792Ø systsinp no'
    call @db2all0 prmalloc
    if word(result,1) = 99 then do
      $exitc = 99 ; return $exitc
      end
    end
 else "alloc da('"outdstsin"') f("systsinp") shr reuse"
sk.1='DSN SYSTEM('subsys')                                              '
sk.2='RUN PROGRAM('$dunlopg') PLAN('$dunlopl') LIB('''$runlib''') -     '
sk.3='LIB('''$runlib''') PARM(''SQL'')                                  '
sk.4='END                                                               '
 sk.Ø=4; jobw = systsinp ; call WriteRec
   /*---  SYSPRINT file allocation             ---*/
 outdsprt= $hiwork'.'subsys'.'$user'.SYSPRINT'
 dsn = sysdsn('''outdsprt''')
 if dsn ¬= OK then do
    prmalloc = subsys' 'outdsprt' Ø 1,15 f,b,a 121 121Ø sysprint no'
    call @db2all0 prmalloc
    if word(result,1) = 99 then do
       $exitc = 99 ; return $exitc
       end
    end
 else "alloc da('"outdsprt"') f("sysprint") shr reuse"
   /*---  SYSREC00 file allocation             ---*/
 outdsrec= $hiwork'.'subsys'.'ComeFrom'.SYSREC00'
 dsn = sysdsn('''outdsrec''')
 if dsn ¬= OK then do
    prmalloc = subsys' 'outdsrec' Ø 15,45 f,b Ø Ø sysrec00 no'
    call @db2all0 prmalloc
    if word(result,1) = 99 then do
       $exitc = 99 ; return $exitc
       end
```

```
    end
 else "alloc da('"outdsrec"') f("sysrec00") shr reuse"
   /*---  SYSPUNCH dummy file allocation       ---*/
 "alloc dummy f(syspunch)"
   /*---  SYSIN file allocation                ---*/
 outdsin= $hiwork'.'subsys'.'$user'.SYSIN'
 "alloc da('"outdsin"') f("sysin") shr reuse"
 if rc > 0 then do
    say '' ; say '' ; say '>>>>>>>>'
    say '>>>>>>>>  Error allocating file "'outdsin'"    '
    say '>>>>>>>>  RC='rc'. Verify.                     '
    say '>>>>>>>>' ; say '' ; say '' ; $exitc = 99 ; return $exitc
    end
   /*---  Call DB2                             ---*/
 xx=outtrap(trpdb00.)
    address tso "ex '"outdstsin"'"
 xx=outtrap(off)
 if rc > 0 then do
    analisi = substr(trpdb00.1,1,8)
   /*---  DB2 subsystem not active             ---*/
    select
    when analisi = 'DSNE100I' then do
       $exitc  = 99
       @db2msg =
'The_DB2_subsystem_'Subsys'_is_not_active.PF3_to_continue'
       cur00   = '@db2subs' ; return  $exitc cur00 @db2msg
       end
   /*---  DB2 subsystem not defined on LPAR    ---*/
    when analisi = 'DSNE110E' then do
       @db2msg = '
The_DB2_subsystem_'Subsys'_is_not_defined_on_'$lpar'_PF3_to_c
ontinue'
       $exitc = 99  ; cur00 = '@db2subs'
       return  $exitc cur00 @db2msg
       end
    otherwise
       xx=outtrap(trpdummy.)
          address tso "printoff ('"outdsprt"')  class(R)"
       xx=outtrap(off)
       do #a = 1 to trpdb00.0
          say trpdb00.#a
          end
       @db2msg = '
Unpredictable_error.End_elaboration._Contact_Support_Staff'
       $exitc = 99 ; cur00  = '@db2subs'
       return  $exitc cur00 @db2msg
       end
    end  /* if rc > 0 */
 else do
    xx=outtrap(trpread00.)
      "execio * diskr sysprint (stem sysprint. finis"
```

```
    xx=outtrap(off)
   /*---  Test Select RC                        ---*/
    do #b = 1 to sysprint.Ø
       okunlo = word(sysprint.#b,2)
       if okunlo = 'DSNT495I' then do
          okunlo = yes ; NRrec = word(sysprint.#b,5)
          end
       end
   /*---  DB2 access OK Start process           ---*/
    if okunlo = yes then do
       if NRrec > Ø then do
          xx=outtrap(trpdummy.)
             address tso "printoff ('"outdsprt"')  class(R)"
          xx=outtrap(off)
          call Free ; $exitc = ØØ ; return $exitc okunlo NRrec
          end
       else do
          $exitc = ØØ ; return $exitc okunlo NRrec
          end
       end
    else do
       xx=outtrap(trpdummy.)
          address tso "printoff ('"outdsprt"')  class(R)"
       xx=outtrap(off)
       cur ØØ  = '@db2subs'
       @db2msg= 'DB2 access KO. Verify the sysout '$user' to analyse the
failure reason !!'
       $exitc = 99 ; return  $exitc cur ØØ @db2msg
       end
    end /* if rc = Ø */
 return
   /*---  Write record routine                  ---*/
 WriteRec :
    "EXECIO * DISKW "jobw" (STEM sk. FINIS"
 ClearRec:
    DO #f = 1 to sk.Ø
       sk.#f = blk
       end ; return
   /*---  Free work DataSet                     ---*/
 Free  :
    xx=outtrap(trpdummy.)
       "free fi(sysprint)" ; "free fi(sysrecØØ)" ; "free fi(systsinp)"
       "free fi(sysin)"    ; "free fi(syspunch)"
    xx=outtrap(off) ; return
```

$DB2P$$$ PANEL

```
)ATTR
}  TYPE(TEXT)   COLOR(red)
```

```
¬  COLOR(turq)  TYPE(TEXT)
$  type(text)   intens(low) color(yellow)
?  COLOR(white) TYPE(TEXT)  hilite(blink)
>  type(text) color(green)
#  type(output) caps(off)
<  type(text) intens(low) skip(on)
%  type(text) color(turq)
)BODY DEFAULT();^)WINDOW(Ø8Ø,Ø24)
$                          ¬******     *******
$                         ¬**     *   ** ?> ¬*  ********
$                        ¬** ?> ¬*   *******    **    **
$                       ¬**     *   ** ?> ¬*        **
$                      ¬******     *******        **
$           }***                               ¬**
$         }*******                            ¬*********
}         **
}        ****       /   /  /   /  /***/  /***/ ***** /***/ **********
}          *****   /   /  /   /  /***/  /***/ /   / /«**/ **********
}             **  *****  *****  /      /     ***** /  «      **
}     ********                                              **
}      ****        $********************************** }** $*********
$                       **                          ***   }**
$                      **  *****  *****  /    ******* }**
$                     **  /   /  /   /  /    **
$                    **  *****  *****  **** ****
$                   **                        *****
$                  **                           $**
$                                            $********
$                                             $****
      #Z       $          >Press ENTER to continue<         #Z          $

)INIT
   .ZVARS = '@db2data,@db2tim'
)END
```

$DB2P000 PANEL

```
)ATTR FORMAT(MIX)
 >  type(text)    intens(&acc) color(white)
 «  type(text)    intens(high) color(yellow)
 %  type(text)    intens(high) color(white)  skip(on)
 +  type(text)    intens(low)  color(green)  skip(on)
 *  type(output)  intens(high) color(yellow) caps(off)
 #  type(output)  intens(low)  color(green)  skip(on)
 $  type(input)   intens(low)  color(red)    pad(_)
 @  type(input)   intens(low)  color(red)
)BODY expand(\\)
+ &ZUSER   + \-\- %DB2 Tools Main Menu +-\-\  #Z       +#Z       +
%COMMAND ===>$ZCMD                                                +
```

```
+*Z                                                                          +
    %DB2 Subsystem   ==>$Z   +             >DB2 Version         ==>#Z +

 +\-\   +
+
+
+
+                              «A+- Recover Tablespace TOCOPY
+
+
+
+
+
+              %Select option+$Z+
+
+
+
+
+
+
+                                             ---- %PF1+Help %PF3+End ----
)INIT
 .HELP  = @DB2H000
 .ZVARS = '@db2data,@db2tim,@db2msg,@db2subs,@db2ver,@db2opt'

)PROC
 &@db2data = '&zday/&zmonth/&zyear &ztime'
 ver(&@db2subs,nonblank)
 ver(&@db2opt,list,A)
)END
```

$DB2P008 PANEL

```
)ATTR FORMAT(MIX)
 %  type(text)    intens(high) color(&txhigh) skip(on)
 +  type(text)    intens(low)  color(green)   skip(on)
 ?  type(text)    intens(high) color(red)     skip(on)
 <  type(text)    intens(high) color(yellow)  skip(on) hilite(blink)
 *  type(output)  intens(high) color(yellow)  caps(off)
 #  type(output)  intens(low)  color(green)   skip(on)
 $  type(input)   intens(low)  color(red)     pad(_)
)BODY expand(\\)
+ &ZUSER + \-\- %DB2 Recover Tablespace TOCOPY +-\-\  #Z       +#Z     +
%COMMAND ===>$ZCMD                                                       +
+*Z                                                                      +
    %DB2 Subsystem   ==>#Z   +             %DB2 Version         ==>#Z +

   ------------------------------------------------------------------------
    %Database      ==>$Z       +        %Tablespace          ==>$Z     +
```

```
    %Timestamp GE  ==>$Z                          +

  +\-\- %Variables Description +-\-\ +

   +%Database          +- DataBase to be selected

   +%Tablespace name   +- Tablespace to be%Recovered Tocopy+

   +%Timestamp GE      +- Timestamp% Grest Equal+for select on SYSCOPY
                          table
                          Timestamp format yyyy-mm-dd-hh.mm.ss.zzzzzz




+                                          ---- %PF1+Help %PF3+End ----

)INIT
 .HELP  =  @db2hØ15
 .ZVARS = '@db2data,@db2ora,@db2msg,@db2subs,@db2ver, +
           @db2db,@db2tbsp,@db2tims,'
)PROC
 &@DB2DATA = '&ZDAY/&ZMONTH/&ZYEAR &ZTIME'
 ver(&@db2db,nonblank)
 ver(&@db2tbsp,nonblank)
 ver(&@db2tims,nonblank)
)END
```

$DB2P009 PANEL

```
)ATTR FORMAT(MIX)
 %  type(text)    intens(high) color(&txhigh) skip(on)
 +  type(text)    intens(low)  color(green)   skip(on)
 ?  type(text)    intens(high) color(red)     skip(on)
 *  type(output)  intens(high) color(yellow)  caps(off)
 #  type(output)  intens(high) color(green)   skip(on)
 <  type(output)  intens(high) color(white)   skip(on)
 $  type(input)   intens(low)  color(red)     pad(_)
)BODY expand(\\)
+ &ZUSER + \-\- %DB2 Recover Tablespace TOCOPY +-\-\  #Z       +#Z     +
%COMMAND ===>$ZCMD                                                      +
+*Z                                                                     +
    %DB2 Subsystem   ==>#Z   +             %DB2 Version         ==>#Z +
+                                           ---- %PF1+Help %PF3+End ----
   ------------------------------------------------------------------------
    %Database      ==>#Z       +         %Tablespace          ==>#Z    +
    %Timestamp GE  ==>#Z                           +
   ------------------------------------------------------------------------
%   TSname     Timestamp                Imagecopy                   File
```

```
%           yyyy mm dd hh:mm            Data set name              Number
+  -------- ---------------- ---------------------------------- --------
)MODEL
$Z#Z       #Z               #Z
#Z
)INIT
 .HELP  =  @db2hØ16
 .ZVARS = '@db2data,@db2tim,@db2msg,@db2subs,@db2ver,                 +
           @db2db,@db2tbsp,@db2tims,                                  +
           @db2sel,$ts,$tim,$ds,$fil'
)PROC
 &@DB2DATA = '&ZDAY/&ZMONTH/&ZYEAR &ZTIME'
)END
```

$DB2H000 HELP PANEL

```
)ATTR
>  type(text) intens(low) color(green)
?  type(text) intens(low) color(red)
%  type(text) intens(low) color(turq)
{  type(text)
$  type(text) intens(low)
}  type(et)
)BODY DEFAULT(<!*)EXPAND(ºº)
>---{ Managment Tool for DB2 environment Help >---$
$
$
}1.${Introduction$
$
$   >The{"DB2 Tools">procedure has the scope to make automatic a series
$   >of activities that are frequently carried out from customers DB2
$   >in a manual way; thus producing a big saving in time.
$
$   >The Tool executes a query on a DB2
$   >Catalog in order to obtain information, and creates some
$   >batch procedures.
$
}2.${Environment parameter descriptions$
$
$   >Before being able to approach the functions of the Tool, will be
$   >necessary to customize some Environment parameters:
$
$   %DB2 Subsystem   ==>?----         %DB2 Version         ==>?--
$
$ >-{DB2 Subsystem  >Is the DB2 subsystem name on which working
$
$
$                                              {ENTER>to continue{PF3>End>
)INIT
)PROC
```

```
&zcont = @db2hØØ1
)END
```

$DB2H001 HELP PANEL

```
)ATTR
>  type(text)   intens(low)  color(green)
$  type(text)   intens(low)
«  type(text)   intens(high) color(yellow)
{  type(text)
}  type(et)
)BODY DEFAULT(<!*)EXPAND(ºº)
>---{ Managment Tool for DB2 environment Help >---$
$
$  >-{DB2 Version   >Is the DB2 version valued in automatic way
$                   >depending on the chosen Subsystem.
$
$
$
}3.${Tool Functions$
$
$                          -«System Managment Tools>-
$
$
$                       «A>- Recover Tablespace TOCOPY
$                       «B>- SYSCOPY managment
$
$
$
$
$
$
$
$                                                {ENTER>to continue{PF3>End>
)INIT
)PROC
&zcont = @db2hØ15
)END
```

$DB2H015 HELP PANEL

```
)ATTR
>  type(text) intens(low) color(green)
?  type(text) intens(low) color(red)
%  type(text) intens(low) color(turq)
¬  type(text) intens(low) color(yellow)
{  type(text)
```

```
$  type(text) intens(low)
#  type(nt)
}  type(et)
)BODY DEFAULT(<!*)EXPAND(ºº)
>---{ Managment Tool for DB2 environment Help >---$
$
}3.A${Recover Tablespace TOCOPY$
$
$    >The function{"Recover Tablespace TOCOPY">allows to recover
$    >a TABLESPACE to one specific IMAGE COPY.
$    >The following fields are required :$
$
$    %Database      ==>?-------->     %Tablespace       ==>?-------->
$    %Timestamp GE  ==>?yyyy-mm-dd-hh.mm.ss.xxxxxx>
$
$  >-{Database$    >Is the Database name used in the WHERE-condition for
$                  >the query on SYSIBM.SYSCOPY
$
$  >-{Tablespace$  >Is the Tablespace name requiring a{RECOVER
$                  {TOCOPY.>The field Tablespace allows the
$                  character &varstar
$
$  >-{Timestamp GE$>Is the Timestamp used in the WHERE-condition for the
$             >query on SYSIBM.SYSCOPY. The query will be carried
$             >out for a Timestamp GE (great or equal) of that supplied.
$
$
$                                          {ENTER>to continue{PF3>End>
$
)INIT
&varstar = '* and %'
)PROC
  &zcont = @db2hØ16
)END
```

$DB2H016 HELP PANEL

```
)ATTR
>  type(text) intens(low) color(green)
?  type(text) intens(low) color(red)
%  type(text) intens(low) color(turq)
¬  type(text) intens(low) color(yellow)
{  type(text)
$  type(text) intens(low)
#  type(nt)
}  type(et)
)BODY DEFAULT(<!*)EXPAND(ºº)
>---{ Managment Tool for DB2 environment Help >---$
$
$    >After the selection of the{Database/Tablespace>and{Timestamp>the
```

```
$    >Clist shows the IMAGE COPY available for the Tablespace RECOVER.
$
$     ------------------------------------------------------------------
$     Database      ==>{DBANAGRA>       $Tablespace       ==>{TSSANAGR>
$       Timestamp GE  ==>{2001-01-19-00.00.00.000000>
$     ------------------------------------------------------------------
$                                  ---- {PF1>On Line Help{PF3>Exit ----
$      TSname     Timestamp              Imagecopy               File
$               yyyy mm dd hh:mm          Data set name           Number
$     -------- ---------------- -------------------------------- ------
$ ?_$TSSANAGR 2001-03-11-01.14 SYSG.DSNS.SAVEDATI.S03.COPYW.G0300V00  26
$ ?_$TSSANAGR 2001-03-04-01.10 SYSG.DSNS.SAVEDATI.S03.COPYW.G0299V00  26
$ ?E$TSSANAGR 2001-02-25-01.12 SYSG.DSNS.SAVEDATI.S03.COPYW.G0298V00  26
$ ?_$TSSANAGR 2001-02-18-01.17 SYSG.DSNS.SAVEDATI.S03.COPYW.G0297V00  26
$
$    >The value of the input field may be{E>or{R>:
$    > -{E>to create job{Tablespace RECOVER TOCOPY>&{REBUILD INDEX>and
$    >edit it
$    > -{R>to create job{Tablespace RECOVER TOCOPY>&{REBUILD INDEX>and
$    >submit it
$
$                                          {ENTER>to continue{PF3>End>
)INIT
)PROC
  &zcont = @db2h000
)END
```

Giuseppe Rendano
DB2 Systems Programmer (Italy)

If you have ever experienced any difficulties with DB2, or made an interesting discovery, you could receive a cash payment, a free subscription to any of our *Updates*, or a credit against any of Xephon's wide range of products and services, simply by telling us all about it.

If you're interested in writing an article, but not sure what on, then visit the *DB2 Update* Web site, http://www.xephon.com/db2update.html, and follow the link to *Opportunities for DB2 specialists*. Here you'll find a list of topics that readers have asked us for more articles about.

Articles can be e-mailed to Trevor Eddolls at trevore@xephon.com. A copy of our *Notes for Contributors* is available from www.xephon.com/nfc.

I like DB2 because...

In the dog-eat-dog world of database marketing, IBM has been very quick to attack its arch-rival Oracle. Oracle recently posted some disappointing results and followed it up by changing its pricing model from the hugely unpopular Universal Power Unit model, which was based on the type and speed of a processor, to a per-processor model – like DB2. IBM has published tables that suggest only users of Oracle's Enterprise Edition will save money, and everyone else will be paying more.

So how do Oracle9*i* and DB2 compare? Well the answer to that question is one that has had both companies' spin doctors working overtime.

Basically, Oracle9*i* is more expensive than DB2. However, Oracle9*i* comes with far more features than DB2 already in the box. It has message queueing, workflow, and file capabilities – DB2 users need to buy these. However, IBM claims that this is only of value in an all Oracle environment, and goes on to suggest that the products would never have any success in the marketplace on their own. IBM attacks the bundling approach saying that users have to pay for them whether they are ever going to use them or not.

DB2, of course, comes with MQSeries, VisualAge for Java, Websphere, OLAP, and warehousing. However, if a customer wants workflow or files capabilities there is an extra charge.

Oracle is claiming that its OLAP and mining solutions are cheaper than IBM's. IBM even admits that there are cases where this is true, but suggests that DB2's OLAP is based on industry standards, whereas Oracle's is proprietary and has little industry acceptance.

IBM admits that where there are only a few users, Oracle is the cheaper option. However, it claims that in most cases DB2 has a definite price advantage.

So it looks like sites that chose DB2 over Oracle made the right decision.

DB2 routines administration

This article describes my DB2 admistration routines. The DB2 routines are stored procedures, user-defined functions, and triggers. We have many SPs, UDFs, and triggers, so I have prepared an easy way to manage them. If you want to change *create* definitions for DB2 routines, you must retrieve all the necessary information from the DB2 catalog. My REXX procedure (UDF) generates these *create* statements (create procedure, create function, and create trigger) in an ISPF file. I used the IBM DSNREXX interface to retrieve data from the DB2 catalog.

To start the procedure, type TSO UDF in the command line. This gives the following main menu:

```
                     Date: 29 August 2001
 DB2 Routines        Time: 9:07am
                     User: NADI
**************************************

  1 - User-Defined Function UDF
  2 - Stored Procedure
  3 - Triggers
  X - Exit

**************************************

==>        Enter 1, 2, 3 or X.

 PF1 Help                    PF3 End
```

If you want to generate a trigger definition, type '3'. This produces the following:

```
-------------------------  Triggers             -------- Row 1 to 3 of 3
 Command ===>                                          Scroll ===> PAGE
 SSID DSNP
 ------------------------------------------------------------------------
 Valid cmd: S Continue                                        F3 -> End
 ------------------------------------------------------------------------
 S Schema   Name             Trigger action      DB2 Table

 - NADI     TRG999D            AFTER DELETE        NADI.A999
 - NADI     TRG999I            AFTER INSERT        NADI.A999
```

```
 - NADI     TRG999U            AFTER UPDATE        NADI.A999
 ***************************** Bottom of data ***************************
```

The last step generates the *create trigger* definition (after an update action):

```
****************************** Top of Data ************************
 SET CURRENT SQLID = 'NADI' ;

 --DROP TRIGGER NADI.TRG999U  RESTRICT;
 --COMMIT;

 CREATE TRIGGER TRG999U
 AFTER UPDATE ON NADI.A999
 REFERENCING NEW AS N FOR EACH ROW MODE DB2SQL
 UPDATE NADI.A999T SET A999USE=N.A999USE,
 A999PDS=N.A999PDS, A999DIS=N.A999DIS
 WHERE A999USE=N.A999USE;

 GRANT EXECUTE ON PACKAGE TRG999U.* TO PUBLIC;

 COMMENT ON TRIGGER NADI.TRG999U IS 'UPDATE ACTION';
***************************** Bottom of Data ************************
```

UDF REXX PROCEDURE

```
/* REXX */
/* UDF Clone                                              */
/* trace r */
 zpfctl = 'OFF'
 Y=MSG("OFF")
 address ispexec 'vput (zpfctl) profile'
 cur='X'
 Call Create_messg
 TOP:
 date=DATE()
 time=TIME(C)
 address ispexec "display panel(UDFMØ) cursor("CUR")"
 if rc=8 then do
    /* address ispexec "tbclose "messdb"" */
    exit
 end
 if x=1 | x=2 | x=3 then nop
 else if x='X'
      then exit
      else signal Top
 Title1='Specificname'
 Title2='External Name'
 if x=1 then Title ='User-Defined Function UDF'
```

```
if x=2 then Title ='Stored Procedure'
if x=3 then do
   Title ='Triggers'
   Title1='Trigger action'
   Title2='DB2 Table'
end
HVSC=' '; HVNA=' '; HVSP=' '; HVEN=' '; HVRE=' '; HVPK=' '
address ispexec "display panel(UDFM1)"
if rc=8 then signal Top
messg = "Accessing  db2 system "db2""
messg = time() || " " || messg
Call Send_messg
messg = 'Select     sysroutines    information'
if x=3 then messg = 'Select     systriggers    information'
messg = time() || " " || messg
Call Send_messg

/* DSNREXX Language Support                                */
Address TSO "SUBCOM DSNREXX"
IF RC THEN
S_RC = RXSUBCOM(ADD,DSNREXX,DSNREXX)

if x=1 then type='F'
if x=2 then type='P'

/* Part I . List of DB2 Routines                            */
SUB:
SSID = db2
ADDRESS DSNREXX "CONNECT" SSID

if x=1 | x=2 then do
SQLSTMT= "SELECT STRIP(SCHEMA), NAME, STRIP(SPECIFICNAME),         ",
"         EXTERNAL_NAME, STRIP(REMARKS), STRIP(COLLID)             ",
"FROM SYSIBM.SYSROUTINES                                           ",
"WHERE SCHEMA LIKE '"rsche"%'                                      ",
"  AND NAME   LIKE '"rname"%'                                      ",
"  AND SPECIFICNAME  LIKE '"rspec"%'                               ",
"  AND EXTERNAL_NAME LIKE '"rexte"%'                               ",
"  AND ROUTINETYPE='"type"'                                        ",
"  WITH UR                                                         "

Address DSNREXX "EXECSQL DECLARE C1 CURSOR FOR S1"
Address DSNREXX 'EXECSQL PREPARE S1 FROM :SQLSTMT'
Address DSNREXX "EXECSQL OPEN C1"
Address DSNREXX,
 "EXECSQL FETCH C1 INTO :HVSC, :HVNA, :HVSP, :HVEN, :HVRE, :HVPK"

address ispexec,
        'tbcreate "rlist" names(HVSC,HVNA,HVSP,HVEN,HVRE,HVPK)'
```

```
do while(sqlcode=Ø)
   address ispexec 'tbadd "rlist"'
   Address DSNREXX,
    "EXECSQL FETCH C1 INTO :HVSC, :HVNA, :HVSP, :HVEN, :HVRE, :HVPK"
end
Address DSNREXX "EXECSQL CLOSE C1"
address ispexec 'tbtop "rlist"'
address ispexec 'tbdispl "rlist" panel(UDFM1)'
if rc=8 then do
   address ispexec 'tbend "rlist"'
   address ispexec "tbend "messdb""
   Call Create_messg
   Signal Top
end
if rc<8 & cmd='S' then nop
else do
   address ispexec 'tbend "rlist"'
   address ispexec "tbend "messdb""
   Call Create_messg
   Signal Sub
end
address ispexec 'tbend "rlist"'
/* End Part I.    ************************************* */

function=hvna
specname=hvsp

messg = "Select     sysparms       information"
messg = time() || " " || messg
Call Send_messg
end

/* Part IIa. Create Function Statement                    */
if x=1 then do
SQLSTMT=,
"SELECT Ø, 'CREATE FUNCTION '||STRIP(SCHEMA,B)||'.'||STRIP(NAME,B) ",
"FROM SYSIBM.SYSROUTINES                                           ",
"WHERE NAME=UCASE('"FUNCTION"')                                    ",
" AND SPECIFICNAME=UCASE('"SPECNAME"')                             ",
" AND ROUTINETYPE='"type"'                                         ",
" UNION                                                            ",
" SELECT ORDINAL, SPACE(7)||                                       ",
"        CASE(ORDINAL)                                             ",
"         WHEN(1) THEN '( '                                        ",
"         ELSE ', '                                                ",
"        END CONCAT                                                ",
"        CASE(TYPENAME)                                            ",
"         WHEN('VARCHAR') THEN STRIP(TYPENAME,B) ||                ",
"                              '('||STRIP(CHAR(LENGTH),B)||')'     ",
"         WHEN('CHAR')    THEN STRIP(TYPENAME,B) ||                ",
```

```
"                            '('||STRIP(CHAR(LENGTH),B)||')'     ",
"         WHEN('DECIMAL') THEN STRIP(TYPENAME,B) ||               ",
"                            '('||STRIP(CHAR(LENGTH),B)||','||   ",
"                            STRIP(CHAR(SCALE),B)||')'           ",
"         ELSE STRIP(TYPENAME,B)                                 ",
"        END CONCAT                                              ",
"        CASE(ORDINAL)                                           ",
"         WHEN(PARM_COUNT) THEN ' )'                             ",
"         ELSE ''                                                ",
"        END                                                     ",
" FROM SYSIBM.SYSPARMS P,                                        ",
"      SYSIBM.SYSROUTINES R                                      ",
" WHERE P.NAME=UCASE('"FUNCTION"')                               ",
"   AND P.SPECIFICNAME=UCASE('"SPECNAME"')                       ",
"   AND P.ROUTINETYPE='"type"'                                   ",
"   AND P.ROWTYPE='P'                                            ",
"   AND P.NAME=R.NAME                                            ",
"   AND P.SPECIFICNAME=R.SPECIFICNAME                            ",
" UNION                                                          ",
" SELECT 99+ORDINAL,                                             ",
"        CASE(FUNCTION_TYPE)                                     ",
"         WHEN('T') THEN                                         ",
"                   CASE(ORDINAL)                                ",
"                    WHEN(1)                                     ",
"                    THEN 'RETURNS TABLE ( '||STRIP(PARMNAME)||' ' ",
"                    ELSE '              , '||STRIP(PARMNAME)||' ' ",
"                   END                                          ",
"         ELSE 'RETURNS  '                                       ",
"        END CONCAT                                              ",
"        CASE(TYPENAME)                                          ",
"         WHEN('VARCHAR') THEN STRIP(TYPENAME,B) ||              ",
"                            '('||STRIP(CHAR(LENGTH),B)||')'     ",
"         WHEN('CHAR')    THEN STRIP(TYPENAME,B) ||              ",
"                            '('||STRIP(CHAR(LENGTH),B)||')'     ",
"         WHEN('DECIMAL') THEN STRIP(TYPENAME,B) ||              ",
"                            '('||STRIP(CHAR(LENGTH),B)||','||   ",
"                            STRIP(CHAR(SCALE),B)||')'           ",
"                        ELSE STRIP(TYPENAME,B)                  ",
"        END CONCAT                                              ",
"        CASE(FUNCTION_TYPE)                                     ",
"         WHEN('T') THEN                                         ",
"                    CASE(R.RESULT_COLS)                         ",
"                     WHEN(ORDINAL) THEN ' )'                    ",
"                     ELSE ''                                    ",
"                   END                                          ",
"         ELSE ''                                                ",
"        END                                                     ",
" FROM SYSIBM.SYSPARMS P,                                        ",
"      SYSIBM.SYSROUTINES R                                      ",
" WHERE P.NAME=UCASE('"FUNCTION"')                               ",
"   AND P.SPECIFICNAME=UCASE('"SPECNAME"')                       ",
```

```
"   AND P.ROUTINETYPE='"type"'                                    ",
"   AND P.ROWTYPE IN ('C','R')                                    ",
"   AND P.NAME=R.NAME                                             ",
"   AND P.SPECIFICNAME=R.SPECIFICNAME                             ",
" UNION                                                           ",
" SELECT 201,'SPECIFIC '||SPECIFICNAME                            ",
" FROM SYSIBM.SYSROUTINES                                         ",
" WHERE NAME=UCASE('"FUNCTION"')                                  ",
"   AND SPECIFICNAME=UCASE('"SPECNAME"')                          ",
"   AND ROUTINETYPE='"type"'                                      ",
" UNION                                                           ",
" SELECT 202,'EXTERNAL NAME '''||                                 ",
"        STRIP(CHAR(EXTERNAL_NAME,18))||''''                      ",
" FROM SYSIBM.SYSROUTINES                                         ",
" WHERE NAME=UCASE('"FUNCTION"')                                  ",
"   AND SPECIFICNAME=UCASE('"SPECNAME"')                          ",
"   AND ROUTINETYPE='"type"'                                      ",
" UNION                                                           ",
" SELECT 203,'LANGUAGE '||LANGUAGE                                ",
" FROM SYSIBM.SYSROUTINES                                         ",
" WHERE NAME=UCASE('"FUNCTION"')                                  ",
"   AND SPECIFICNAME=UCASE('"SPECNAME"')                          ",
"   AND ROUTINETYPE='"type"'                                      ",
" UNION                                                           ",
" SELECT 204,'PARAMETER STYLE '||                                 ",
"            CASE(PARAMETER_STYLE)                                ",
"             WHEN('D') THEN 'DB2SQL'                             ",
"            END                                                  ",
" FROM SYSIBM.SYSROUTINES                                         ",
" WHERE NAME=UCASE('"FUNCTION"')                                  ",
"   AND SPECIFICNAME=UCASE('"SPECNAME"')                          ",
"   AND ROUTINETYPE='"type"'                                      ",
" UNION                                                           ",
" SELECT 205,CASE(DETERMINISTIC)                                  ",
"             WHEN('Y') THEN 'DETERMINISTIC'                      ",
"             WHEN('N') THEN 'NOT DETERMINISTIC'                  ",
"            END                                                  ",
" FROM SYSIBM.SYSROUTINES                                         ",
" WHERE NAME=UCASE('"FUNCTION"')                                  ",
"   AND SPECIFICNAME=UCASE('"SPECNAME"')                          ",
"   AND ROUTINETYPE='"type"'                                      ",
" UNION                                                           ",
" SELECT 206,CASE(SQL_DATA_ACCESS)                                ",
"             WHEN('N') THEN 'NO SQL'                             ",
"             WHEN('C') THEN 'CONTAINS SQL'                       ",
"             WHEN('M') THEN 'MODIFIES SQL DATA'                  ",
"             WHEN('R') THEN 'READS SQL DATA'                     ",
"            END                                                  ",
" FROM SYSIBM.SYSROUTINES                                         ",
" WHERE NAME=UCASE('"FUNCTION"')                                  ",
"   AND SPECIFICNAME=UCASE('"SPECNAME"')                          ",
```

```
"   AND ROUTINETYPE='"type"'                                          ",
" UNION                                                               ",
" SELECT 207,CASE(DBINFO)                                             ",
"             WHEN('N') THEN 'NO DBINFO'                              ",
"             WHEN('Y') THEN 'DBINFO'                                 ",
"            END                                                      ",
" FROM SYSIBM.SYSROUTINES                                             ",
" WHERE NAME=UCASE('"FUNCTION"')                                      ",
"   AND SPECIFICNAME=UCASE('"SPECNAME"')                              ",
"   AND ROUTINETYPE='"type"'                                          ",
" UNION                                                               ",
" SELECT 208,'FENCED'                                                 ",
" FROM SYSIBM.SYSROUTINES                                             ",
" WHERE NAME=UCASE('"FUNCTION"')                                      ",
"   AND SPECIFICNAME=UCASE('"SPECNAME"')                              ",
"   AND ROUTINETYPE='"type"'                                          ",
"   AND FENCED='Y'                                                    ",
" UNION                                                               ",
" SELECT 209,CASE(COLLID)                                             ",
"             WHEN(' ') THEN 'NO COLLID'                              ",
"             ELSE 'COLLID '||COLLID                                  ",
"            END                                                      ",
" FROM SYSIBM.SYSROUTINES                                             ",
" WHERE NAME=UCASE('"FUNCTION"')                                      ",
"   AND SPECIFICNAME=UCASE('"SPECNAME"')                              ",
"   AND ROUTINETYPE='"type"'                                          ",
" UNION                                                               ",
" SELECT 210,'CARDINALITY '||CHAR(INTEGER(CARDINALITY))               ",
" FROM SYSIBM.SYSROUTINES                                             ",
" WHERE NAME=UCASE('"FUNCTION"')                                      ",
"   AND SPECIFICNAME=UCASE('"SPECNAME"')                              ",
"   AND ROUTINETYPE='"type"'                                          ",
"   AND CARDINALITY > 0                                               ",
" UNION                                                               ",
" SELECT 211,'WLM ENVIRONMENT ' CONCAT                                ",
"            CASE(WLM_ENV_FOR_NESTED)                                 ",
"             WHEN('N') THEN WLM_ENVIRONMENT                          ",
"             WHEN('Y') THEN '('||STRIP(WLM_ENVIRONMENT,B)||',*)'     ",
"            END                                                      ",
" FROM SYSIBM.SYSROUTINES                                             ",
" WHERE NAME=UCASE('"FUNCTION"')                                      ",
"   AND SPECIFICNAME=UCASE('"SPECNAME"')                              ",
"   AND ROUTINETYPE='"type"'                                          ",
"   AND WLM_ENVIRONMENT <> ' '                                        ",
" UNION                                                               ",
" SELECT 212,CASE(STAYRESIDENT)                                       ",
"             WHEN('N') THEN 'STAY RESIDENT NO'                       ",
"             WHEN('Y') THEN 'STAY RESIDENT YES'                      ",
"            END                                                      ",
" FROM SYSIBM.SYSROUTINES                                             ",
" WHERE NAME=UCASE('"FUNCTION"')                                      ",
```

```
"   AND SPECIFICNAME=UCASE('"SPECNAME"')                                ",
"   AND ROUTINETYPE='"type"'                                            ",
" UNION                                                                 ",
" SELECT 213,CASE(PROGRAM_TYPE)                                         ",
"             WHEN('M') THEN 'PROGRAM TYPE MAIN'                        ",
"             WHEN('S') THEN 'PROGRAM TYPE SUB'                         ",
"            END                                                        ",
" FROM SYSIBM.SYSROUTINES                                               ",
" WHERE NAME=UCASE('"FUNCTION"')                                        ",
"   AND SPECIFICNAME=UCASE('"SPECNAME"')                                ",
"   AND ROUTINETYPE='"type"'                                            ",
" UNION                                                                 ",
" SELECT 214,CASE(EXTERNAL_ACTION)                                      ",
"             WHEN('E') THEN 'EXTERNAL ACTION'                          ",
"             WHEN('N') THEN 'NO EXTERNAL ACTION'                       ",
"            END                                                        ",
" FROM SYSIBM.SYSROUTINES                                               ",
" WHERE NAME=UCASE('"FUNCTION"')                                        ",
"   AND SPECIFICNAME=UCASE('"SPECNAME"')                                ",
"   AND ROUTINETYPE='"type"'                                            ",
" UNION                                                                 ",
" SELECT 215,CASE(NULL_CALL)                                            ",
"             WHEN('Y') THEN 'CALLED ON NULL INPUT'                     ",
"             WHEN('N') THEN 'RETURNS NULL ON NULL INPUT'               ",
"            END                                                        ",
" FROM SYSIBM.SYSROUTINES                                               ",
" WHERE NAME=UCASE('"FUNCTION"')                                        ",
"   AND SPECIFICNAME=UCASE('"SPECNAME"')                                ",
"   AND ROUTINETYPE='"type"'                                            ",
" UNION                                                                 ",
" SELECT 216,CASE(SCRATCHPAD)                                           ",
"             WHEN('Y') THEN 'SCRATCHPAD '||CHAR(SCRATCHPAD_LENGTH)",
"             WHEN('N') THEN 'NO SCRATCHPAD'                            ",
"            END                                                        ",
" FROM SYSIBM.SYSROUTINES                                               ",
" WHERE NAME=UCASE('"FUNCTION"')                                        ",
"   AND SPECIFICNAME=UCASE('"SPECNAME"')                                ",
"   AND ROUTINETYPE='"type"'                                            ",
" UNION                                                                 ",
" SELECT 217,CASE(FINAL_CALL)                                           ",
"             WHEN('Y') THEN 'FINAL CALL'                               ",
"             WHEN('N') THEN 'NO FINAL CALL'                            ",
"            END                                                        ",
" FROM SYSIBM.SYSROUTINES                                               ",
" WHERE NAME=UCASE('"FUNCTION"')                                        ",
"   AND SPECIFICNAME=UCASE('"SPECNAME"')                                ",
"   AND ROUTINETYPE='"type"'                                            ",
" UNION                                                                 ",
" SELECT 218,CASE(PARALLEL)                                             ",
"             WHEN('A') THEN 'ALLOW PARALLEL'                           ",
"             WHEN('D') THEN 'DISALLOW PARALLEL'                        ",
```

```
"            END                                                    ",
" FROM SYSIBM.SYSROUTINES                                           ",
" WHERE NAME=UCASE('"FUNCTION"')                                    ",
"   AND SPECIFICNAME=UCASE('"SPECNAME"')                            ",
"   AND ROUTINETYPE='"type"'                                        ",
" UNION                                                             ",
" SELECT 219,CASE                                                   ",
"             WHEN(ASUTIME=0) THEN 'ASUTIME NO LIMIT'               ",
"             WHEN(ASUTIME>0) THEN 'ASUTIME LIMIT '||CHAR(ASUTIME) ",
"            END                                                    ",
" FROM SYSIBM.SYSROUTINES                                           ",
" WHERE NAME=UCASE('"FUNCTION"')                                    ",
"   AND SPECIFICNAME=UCASE('"SPECNAME"')                            ",
"   AND ROUTINETYPE='"type"'                                        ",
" UNION                                                             ",
" SELECT 220,CASE(EXTERNAL_SECURITY)                                ",
"             WHEN('D') THEN 'SECURITY DB2'                         ",
"             WHEN('U') THEN 'SECURITY USER'                        ",
"             WHEN('C') THEN 'SECURITY DEFINER'                     ",
"            END                                                    ",
" FROM SYSIBM.SYSROUTINES                                           ",
" WHERE NAME=UCASE('"FUNCTION"')                                    ",
"   AND SPECIFICNAME=UCASE('"SPECNAME"')                            ",
"   AND ROUTINETYPE='"type"'                                        ",
" UNION                                                             ",
" SELECT 221,'RUN OPTIONS '||CHAR(RUNOPTS,50)                       ",
" FROM SYSIBM.SYSROUTINES                                           ",
" WHERE NAME=UCASE('"FUNCTION"')                                    ",
"   AND SPECIFICNAME=UCASE('"SPECNAME"')                            ",
"   AND ROUTINETYPE='"type"'                                        ",
"   AND RUNOPTS <> ' '                                              ",
" ORDER BY 1                                                        ",
" WITH UR                                                           "

Address DSNREXX "EXECSQL DECLARE C1 CURSOR FOR S1"
Address DSNREXX 'EXECSQL PREPARE S1 FROM :SQLSTMT'
Address DSNREXX "EXECSQL OPEN C1"
Address DSNREXX "EXECSQL FETCH C1 INTO :HVST, :HVFU"

address ispexec 'tbcreate "dlist" names(item)'

do while(sqlcode=0)
   item=hvfu
   address ispexec 'tbadd "dlist"'
   Address DSNREXX "EXECSQL FETCH C1 INTO :HVST, :HVFU"
end
code = sqlcode
Address DSNREXX "EXECSQL CLOSE C1"
item=';'
address ispexec 'tbadd "dlist"'
/* End Part IIa.   ************************************** */
```

```
Call Grant
end

/* Part IIb. Create Stored Procedure                          */
if x=2 then do
procname=function
SQLSTMT=,
"SELECT Ø,'CREATE PROCEDURE '||STRIP(SCHEMA,B)||                    ",
"       '.'||STRIP(NAME,B)                                          ",
"FROM SYSIBM.SYSROUTINES                                            ",
"WHERE NAME=UCASE('"procname"')                                     ",
"  AND ROUTINETYPE='"type"'                                         ",
"UNION                                                              ",
"SELECT ORDINAL, SPACE(7)||                                         ",
"       CASE(ORDINAL)                                               ",
"        WHEN(1) THEN '( '                                          ",
"        ELSE ', '                                                  ",
"       END CONCAT                                                  ",
"       CASE(ROWTYPE)                                              ",
"        WHEN('P') THEN CHAR('IN',6)                               ",
"        WHEN('O') THEN CHAR('OUT',6)                              ",
"        WHEN('B') THEN CHAR('INOUT',6)                            ",
"       END CONCAT                                                  ",
"       CASE(TYPENAME)                                             ",
"        WHEN('VARCHAR') THEN STRIP(TYPENAME,B) ||                 ",
"                             '('||STRIP(CHAR(LENGTH),B)||')'      ",
"        WHEN('CHAR')    THEN STRIP(TYPENAME,B) ||                 ",
"                             '('||STRIP(CHAR(LENGTH),B)||')'      ",
"        WHEN('DECIMAL') THEN STRIP(TYPENAME,B) ||                 ",
"                             '('||STRIP(CHAR(LENGTH),B)||','||    ",
"                             STRIP(CHAR(SCALE),B)||')'            ",
"        ELSE STRIP(TYPENAME,B)                                    ",
"       END CONCAT                                                 ",
"       CASE(ORDINAL)                                              ",
"        WHEN(PARM_COUNT) THEN ' )'                                ",
"        ELSE ''                                                   ",
"       END                                                        ",
"FROM SYSIBM.SYSPARMS P,                                           ",
"     SYSIBM.SYSROUTINES R                                         ",
"WHERE P.NAME=UCASE('"procname"')                                  ",
"  AND P.ROUTINETYPE='"type"'                                      ",
"  AND P.NAME=R.NAME                                               ",
"UNION                                                             ",
"SELECT 1ØØ,'LANGUAGE '||LANGUAGE                                  ",
"FROM SYSIBM.SYSROUTINES                                           ",
"WHERE NAME=UCASE('"procname"')                                    ",
"  AND ROUTINETYPE='"type"'                                        ",
"UNION                                                             ",
"SELECT 1Ø1,'RESULT SET '||CHAR(RESULT_SETS)                       ",
"FROM SYSIBM.SYSROUTINES                                           ",
```

```
"WHERE NAME=UCASE('"procname"')                                          ",
"  AND ROUTINETYPE='"type"'                                              ",
"UNION                                                                   ",
"SELECT 102,'EXTERNAL NAME '||CHAR(EXTERNAL_NAME,18)                     ",
"FROM SYSIBM.SYSROUTINES                                                 ",
"WHERE NAME=UCASE('"procname"')                                          ",
"  AND ROUTINETYPE='"type"'                                              ",
"UNION                                                                   ",
"SELECT 103,'PARAMETER STYLE '||                                         ",
"           CASE(PARAMETER_STYLE)                                        ",
"            WHEN('D') THEN 'DB2SQL'                                     ",
"            WHEN('G') THEN 'GENERAL'                                    ",
"            WHEN('N') THEN 'GENERAL WITH NULLS'                         ",
"           END                                                          ",
"FROM SYSIBM.SYSROUTINES                                                 ",
"WHERE NAME=UCASE('"procname"')                                          ",
"  AND ROUTINETYPE='"type"'                                              ",
"UNION                                                                   ",
"SELECT 104,CASE(SQL_DATA_ACCESS)                                        ",
"            WHEN('N') THEN 'NO SQL'                                     ",
"            WHEN('C') THEN 'CONTAINS SQL'                               ",
"            WHEN('R') THEN 'READS SQL DATA'                             ",
"            WHEN('M') THEN 'MODIFIES SQL DATA'                          ",
"           END                                                          ",
"FROM SYSIBM.SYSROUTINES                                                 ",
"WHERE NAME=UCASE('"procname"')                                          ",
"  AND ROUTINETYPE='"type"'                                              ",
"UNION                                                                   ",
"SELECT 105,CASE(DBINFO)                                                 ",
"            WHEN('N') THEN 'NO DBINFO'                                  ",
"            WHEN('Y') THEN 'DBINFO'                                     ",
"           END                                                          ",
"FROM SYSIBM.SYSROUTINES                                                 ",
"WHERE NAME=UCASE('"procname"')                                          ",
"  AND ROUTINETYPE='"type"'                                              ",
"UNION                                                                   ",
"SELECT 106,CASE(COLLID)                                                 ",
"            WHEN(' ') THEN 'NO COLLID'                                  ",
"            ELSE 'COLLID '||COLLID                                      ",
"           END                                                          ",
"FROM SYSIBM.SYSROUTINES                                                 ",
"WHERE NAME=UCASE('"procname"')                                          ",
"  AND ROUTINETYPE='"type"'                                              ",
"UNION                                                                   ",
"SELECT 107,'WLM ENVIRONMENT ' CONCAT                                    ",
"           CASE(WLM_ENV_FOR_NESTED)                                     ",
"            WHEN('N') THEN WLM_ENVIRONMENT                              ",
"            WHEN('Y') THEN '('||STRIP(WLM_ENVIRONMENT,B)||',*)'         ",
"           END                                                          ",
"FROM SYSIBM.SYSROUTINES                                                 ",
"WHERE NAME=UCASE('"procname"')                                          ",
```

```
"  AND ROUTINETYPE='"type"'                                          ",
"  AND WLM_ENVIRONMENT <> ' '                                        ",
"UNION                                                               ",
"SELECT 108,CASE(STAYRESIDENT)                                       ",
"            WHEN('N') THEN 'STAY RESIDENT NO'                       ",
"            WHEN('Y') THEN 'STAY RESIDENT YES'                      ",
"           END                                                      ",
"FROM SYSIBM.SYSROUTINES                                             ",
"WHERE NAME=UCASE('"procname"')                                      ",
"  AND ROUTINETYPE='"type"'                                          ",
"UNION                                                               ",
"SELECT 109,CASE(PROGRAM_TYPE)                                       ",
"            WHEN('M') THEN 'PROGRAM TYPE MAIN'                      ",
"            WHEN('S') THEN 'PROGRAM TYPE SUB'                       ",
"           END                                                      ",
"FROM SYSIBM.SYSROUTINES                                             ",
"WHERE NAME=UCASE('"procname"')                                      ",
"  AND ROUTINETYPE='"type"'                                          ",
"UNION                                                               ",
"SELECT 110,CASE(EXTERNAL_SECURITY)                                  ",
"            WHEN('D') THEN 'EXTERNAL SECURITY DB2'                  ",
"            WHEN('U') THEN 'EXTERNAL SECURITY USER'                 ",
"            WHEN('C') THEN 'EXTERNAL SECURITY DEFINER'              ",
"           END                                                      ",
"FROM SYSIBM.SYSROUTINES                                             ",
"WHERE NAME=UCASE('"procname"')                                      ",
"  AND ROUTINETYPE='"type"'                                          ",
"UNION                                                               ",
"SELECT 111,CASE(COMMIT_ON_RETURN)                                   ",
"            WHEN('N') THEN 'COMMIT ON RETURN NO'                    ",
"            WHEN('Y') THEN 'COMMIT ON RETURN YES'                   ",
"           END                                                      ",
"FROM SYSIBM.SYSROUTINES                                             ",
"WHERE NAME=UCASE('"procname"')                                      ",
"  AND ROUTINETYPE='"type"'                                          ",
"UNION                                                               ",
"SELECT 112,'RUN OPTIONS '||CHAR(RUNOPTS,50)                         ",
"FROM SYSIBM.SYSROUTINES                                             ",
"WHERE NAME=UCASE('"procname"')                                      ",
"  AND ROUTINETYPE='"type"'                                          ",
"  AND RUNOPTS <> ' '                                                ",
" ORDER BY 1                                                         ",
" WITH UR                                                            "
```

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

Bernard Zver
DBA (Slovenia)

DB2 news

IBM has announced Version 2.1 of its Tivoli Manager for DB2, promising improved database management through Tivoli Management Agent (TMA) support, which should increase scalability.

The software helps database administrators manage hundreds of DB2 servers from one console and focuses on monitoring DB2 servers, task scheduling, and automated task execution on many distributed platforms.

Also new is a client connectivity function for DB2 clients, a new task that lets Tivoli administrators test whether a specific client can access a given database server. There's also autodiscovery of DB2 instances on selected endpoints.

In addition, there's better integration with the Java version of DB2's control centre, to reference the appropriate database object directly from the Tivoli desktop. Also, users can now launch the Java DB2 control centre in context from Unix, Windows, and OS/2 endpoints.

Other new bits include support for Windows 2000, endpoints and gateways on OS/2, and nine international languages.

For further information contact your local IBM representative.
URL: http://www.tivoli.com/products.

* * *

Precise Software Solutions has announced its move into the IBM data management market with the launch of Precise/Indepth for DB2 UDB.

The new product joins Precise's suite of end-to-end application performance management solutions for the distributed IT infrastructure and gives DB2 administrators a solution for application performance problem detection and resolution.

Precise/Indepth for DB2 UDB provides customers with a built-in methodology to identify the symptoms and resolve the root causes of performance bottlenecks in their DB2 UDB databases.

Precise/Indepth for IBM DB2 UDB supports DB2 Universal Database running on Unix and is available immediately.

For further information contact:
Precise Software Solutions, 690 Canton Street Westwood, MA 02090, USA.
Tel: (781) 461 0700.
URL: http://www.precise.com/products/indepthDB2.asp.

* * *

Syncsort has launched its SyncSort for z/OS, described as a high performance sort, merge, and copy product to take advantage of the new features of z/OS and the underlying 64-bit architecture.

Users will also be able to specify a DB2 query, which SyncSort will use to access a DB2 database, thereby eliminating an extract step. It will support a maximum of 255 SORTWK datasets, up from 100. Finally, it will incorporate functionality to integrate the mainframe component of Visual SyncSort.

For further information contact:
Syncsort, 50 Tice Boulevard, Woodcliff Lake, NJ 07677, USA.
Tel: 201) 930 8200.
URL: http://www.syncsort.com