106 DB2

August 2001

In this issue

update

DB2 Update

Published by

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

North American office

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

Subscriptions and back-issues

A year's subscription to *DB2 Update*, comprising twelve monthly issues, costs £255.00 in the UK; $380.00 in the USA and Canada; £261.00 in Europe; £267.00 in Australasia and Japan; and £265.50 elsewhere. In all cases the price includes postage. Individual issues, starting with the January 1997 issue, are available separately to subscribers for £22.50 ($33.50) each including postage.

DB2 Update on-line

Code from *DB2 Update*, 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.

How to recover a dropped table in UDB 7.1

This article describes how to test the recovery of a dropped table in UDB 7.1 (no fixpacks). This features gives the user the ability to 'recover' a dropped table – but this should be used as a last resort, because the whole database is recovered and rolled forward, which requires exclusive use of the database, and can be very time-consuming for a large database.

The example below was run on Windows NT 4.0 Service Level 5.

Create a directory called backups on the C: drive. This is where the database back-ups will go. If you already have a directory for your back-ups and want to use that, then that is fine (in the commands below, just replace c:\backups with your directory name).

The first thing to do is create a test database (testdb), change LOGRETAIN to ON, and perform a full offline back-up of it:

```
>db2 "CREATE DB testdb"
>db2 "UPDATE DB CFG FOR testdb USING LOGRETAIN ON"
>db2 "BACKUP DB testdb TO c:\backups"
Backup successful. The timestamp for this backup image is yyyymmddhhmmss
```

where: c:\backups is a directory created/defined in the first step and hhmmss is the local time.

Next create a test table with one column in the default tablespace:

```
>db2 "CONNECT TO testdb"
>db2 "CREATE TABLE test_table(name char(10)) IN USERSPACE1"
```

Alter the tablespace to be able to recover dropped tables:

```
>db2 "ALTER TABLESPACE USERSPACE1 DROPPED TABLE RECOVERY ON"
```

Insert three rows into the test table:

```
#db2 "INSERT INTO test_table VALUES('data1')"
#db2 "INSERT INTO test_table VALUES('data2')"
#db2 "INSERT INTO test_table VALUES('data3')"
```

Select from the test table:

```
#db2 "SELECT * FROM test_table"

NAME
----------
data1
data2
data3

  3 record(s) selected.
```

Drop the test table:

```
#db2 "DROP TABLE test_table"
```

Restore from the full back-up:

```
#db2 "RESTORE DB testdb FROM c:\backups TAKEN AT yyyymmddhhmmss"
```

Respond ‘y’ to the question: “Do you want to continue? (y/n)”.

You should eventually get the message:

```
DB20000I  The RESTORE DATABASE command completed successfully.
```

Find the back-up ID and the DDL of the test table.

To find the back-up ID of the test table, issue the list history dropped table command and write down the back-up ID:

```
>db2 "LIST HISTORY DROPPED TABLE ALL FOR testdb"

                    List History File for testdb

Number of matching file entries = 1


 Op Obj Timestamp+Sequence Type Dev Earliest Log Current Log  Backup ID
 -- --- ------------------ ---- --- ------------ ------------ ----------
  D  T  20010401175250                          0000000000006000000020002
 ------------------------------------------------------------------------
  "DB2ADMIN"."TEST_TABLE" resides in 1 tablespace(s):

  00001 USERSPACE1
 ------------------------------------------------------------------------
    Comment: DROP TABLE
 Start Time: 20010401175250
   End Time: 20010401175250
 ------------------------------------------------------------------------
  00001
```

```
 DDL: CREATE TABLE "DB2ADMIN"."TEST_TABLE" ( "NAME" CHAR(10) )  IN
"USERSPACE1" ;

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

Scan through the logs to get the data for the table:

```
>db2 "ROLLFORWARD DATABASE testdb RECOVER DROPPED TABLE
00000000000006000000020002 TO c:\backups
```

where 00000000000006000000020002 is the back-up ID.

```
                                Rollforward Status

 Input database alias                   = testdb
 Number of nodes have returned status   = 1

 Node number                            = 0
 Rollforward status                     = DB  pending
 Next log file to be read               = S0000000.LOG
 Log files processed                    =  -
 Last committed transaction             = 2001-04-01-16.46.46.000000

>db2 "ROLLFORWARD DATABASE testdb TO END OF LOGS AND STOP RECOVER
DROPPED TABLE 00000000000006000000020002 TO c:\backups

                               Rollforward Status

Input database alias                   = testdb
Number of nodes have returned status   = 1

Node number                            = 0
Rollforward status                     = not pending
Next log file to be read               =
Log files processed                    = S0000000.LOG - S0000000.LOG
Last committed transaction             = 2001-04-01-16.52.50.000000
```

The file which contains the data from test_table is C:\BACKUPS\NODE0000\DATA.

Using notepad on this file shows:

```
"data1     "
"data2     "
"data3     "
```

Recreate the table from the DDL with the list history dropped table command:

```
>db2 "CONNECT TO testdb"

>db2 "CREATE TABLE test_table(name char(1Ø)) IN USERSPACE1"

>db2 "IMPORT FROM c:\backups\node0000\data OF DEL MODIFIED BY COLDEL,
INSERT INTO test_table"

SQL31Ø9N  The utility is beginning to load data from file
"c:\backups\node0000\data".

SQL311ØN  The utility has completed processing.  "3" rows were read from
the input file.

SQL3221W  ...Begin COMMIT WORK. Input Record Count = "3".

SQL3222W  ...COMMIT of any database changes was successful.

SQL3149N  "3" rows were processed from the input file.  "3" rows were
successfully inserted into the table.  "Ø" rows were rejected.


Number of rows read         = 3
Number of rows skipped      = Ø
Number of rows inserted     = 3
Number of rows updated      = Ø
Number of rows rejected     = Ø
Number of rows committed    = 3
```

Check that the previous data has been successfully loaded:

```
>db2 "SELECT * FROM test_table"

NAME
----------
data1
data2
data3

  3 record(s) selected.
```

The recovery is now complete. I would take another full back-up of the database, just to give a clean recovery point.

C Leonard
Freelance Consultant (UK)

DB2 UDB V7.1 SQL enhancements: UNION everywhere

My article on DB2 Warehouse Manager (*DB2 OLAP Server for OS/390 V7.1: DB2 Warehouse Manager*) appeared in Issue 101 of *DB2 Update*, July 2001. It discussed UDB V7.1 performance enhancements with increased parallelism for IN-list index access, more transform-correlated subqueries, partitioning data sets with parallel open, asynchronous INSERT preformatting, and improvements in ORDER BY and MIN/MAX. This article discusses SQL enhancements with UNION everywhere, expanded row expressions for subquery, and self-referencing UPDATE and DELETE.

UNION EVERYWHERE

UNION was restricted to fullselect clauses prior to V7, making it unavailable for table expressions, predicates, and INSERT and UPDATE statements, and creating VIEWs (keywords in text are not conjugated). V7 allows UNION to be used anywhere that a subselect clause is valid. This brings V7 into conformance with the SQL99 standard. It also increases user usability by letting them VIEW multiple smaller tables as a single table without knowing how to code a UNION. Subselect is DB2's basic SELECT containing SELECT, FROM, WHERE, GROUP BY, and HAVING clauses. It does not allow a UNION keyword. To use UNION or UNION ALL required two subselects joined by UNION:

```
SELECT      …      FROM        …      WHERE      …
GROUP BY    …      HAVING      …
UNION or UNION ALL followed by the second subselect.
```

UNION VIEW

IBM has replaced subselect with fullselect in the syntax diagram:

```
>>--CREATE VIEW--view-name--AS--subselect--fullselect-->
(rest of syntax diagram remains unchanged)
```

Assume three tables of alumni, instructor, and student with each table containing 'salary'. A VIEW could be coded using UNION:

```
CREATE VIEW      lu_salary      (salary)      AS
SELECT           salary
FROM             alumni
UNION ALL
SELECT           salary
FROM             instructor
UNION ALL
SELECT           salary
FROM             student
```

This creates for users a single logical table to which they can apply functions like COUNT, AVERAGE, SUM, etc:

```
SELECT           SUM(salary), COUNT(*)
FROM             lu_salary
```

UNION VIEW is read-only but has the advantages of avoiding temporary tables and complex SQL.

UNION in table-specs:

```
>>-----table-name---------------?-------------------------?--><
   ? ?-view-name                ?  ?--correlation-clause--?
   ? ?-table-locator-reference--?                         ?
   ?--(-subselect-)-(-fullselect-)---correlation-clause---?
(rest of syntax diagram remains unchanged)
```

This enhancement allows functions across similar data in multiple tables:

```
SELECT           salary,        SUM(salary)
FROM             lu_salary
      TABLE      (SELECT        salary      FROM         alumni
      UNION ALL   SELECT        salary      FROM         instructor
      UNION ALL   SELECT        salary      FROM         student)
WHERE     …    GROUP BY …
```

UNION in basic predicate:

```
>--expression---=--?----expression-----------------?-->
             ?-<>-? ?--(subselect)--(fullselect)--?
(rest of syntax diagram remains unchanged)
```

This allows data in separate tables to be merged for comparison with single column values within an SQL statement.

UNION in quantified predicate:

```
>--expression--=--SOME--(subselect)--(fullselect)-->
(rest of syntax diagram remains unchanged)
```

This allows values from multiple tables to be compared against a

result set.

UNION in EXISTS predicate:

```
>--EXISTS(subselect)--(fullselect)-->
```

UNION allows access to data across multiple tables in a single predicate. Prior to V7, it would require an OR of the EXISTS predicate with each table.

UNION in the IN predicate:

```
>--expression--IN--(subselect)--(fullselect)-->
(rest of syntax diagram remains unchanged)
```

UNION allows access to data across multiple tables in a single predicate. Prior to V7, it would require an AND of the IN clauses against each table individually.

UNION in INSERT and UPDATE:

```
INSERT INTO      …
(SELECT            …      FROM            …      WHERE      …)
UNION or UNION ALL followed by next SELECT.
```

UNION INSERT allows data from multiple tables to be SELECTed, merged, and INSERTed into a single table.

```
UPDATE      …
      SET   …      =      (SELECT      …       FROM      …
                            WHERE      …)
                        UNION or UNION ALL followed by next SELECT.
```

UNION UPDATE allows a column to be populated from multiple tables.

UNION EXPLAIN

V7 has added two columns to the PLAN_TABLE:

```
PARENT_QBLOCKNO      contains parent query block number
TABLE_TYPE           values are:
      F              function
      Q              temporary intermediate result table
                     (STAR JOIN only)
      T              table
      W              work file
```

The following new values have been added to QBLOCK_TYPE:

- TABLEX – table expression
- UNION – UNION
- UNIONA – UNION ALL.

EXPLAIN should be used extensively while becoming familiar with UNION everywhere.

ROW EXPRESSIONS FOR THE IN PREDICATE

V7 has an enhanced row expression to allow the subquery query predicate to use multiple columns instead of a single column (a restriction that existed with the previous version). V7 has added support for equal and not-equal operators:

```
= ANY (fullselect)     = SOME (fullselect)    <> ALL (fullselect)
(exp, exp…) = (exp, exp…)   (exp, exp…) <> (exp, exp…)
```

The IBM V7 example is to list all tables that do not have indexes defined:

```
SELECT      T1.CREATOR, T1.NAME
FROM        SYSIBM.SYSTABLES T1
WHERE       (T1.CREATOR, T1.NAME)      NOT IN
            (SELECT      T2.TBCREATOR, T2.TBNAME
FROM        SYSIBM.SYSINDEXES T2)
```

The IN predicate also allows multiple expressions on the left side when a fullselect is specified on the right:

```
>-----expression----------?--IN-----(fullselect)--------?--->
   ?              ?--NOT--?      ?----?--,---------?----?
   ?                             ?--(---expression-?-)--?
   ?---?-,------------?                                  ?
   ?-(----expression--?--)----------?---IN--(fullselect)?
                           ?--NOT--?
```

The IBM V7 example is to list all indexes that support a unique constraint:

```
SELECT      T1.CREATOR, T1.NAME
FROM        SYSIBM.SYSINDEXES T1
WHERE       (T1.CREATOR, T1.NAME)      IN
            (SELECT      T2.IXOWNER, T2.IXNAME
FROM        SYSIBM.SYSTABCONST T2
WHERE       T2.TYPE      IN      ('U','V'))
ORDER BY    1, 2
```

A usage restriction is that the fullselect must return the same number of columns in the result set as provided by the multiple expressions. The predicate is evaluated by matching the first expression value against the first result set column, second against second, and so forth. The values are AND together and the predicate is true only if all the columns match.

QUANTIFIED PREDICATES

The quantified predicate now allows multiple expressions so multiple columns can be compared. It has also added SOME, ANY, and ALL keywords.

```
>---expression1----=---?---SOME--?--------(fullselect)---------->
  ?              ?--<>--?-?-ANY---?
  ?              ?-->---?-?-ALL---?
  ?              ?--(syntax diagram shows <, >=, <=…)
  ?--?--,-------------?
  ?--?---expression2--?--…
```

Determining whether the predicate is true or false is now more complex so I suggest extensive testing to assure yourself that you are getting the correct evaluation.

SELF REFERENCING UPDATE/DELETE

Have you been frustrated with:

```
dsnt408i sqlcode = -118, error: the object table or view of the delete
or update statement is also identified in a from clause
```

as would be caused by the following SQL:

```
UPDATE      SALARY
SET         SALARY = SALARY * 1.2
WHERE       SALARY <      (SELECT      AVG(SALARY)
FROM        INSTRUCTOR
```

Be frustrated no longer because V7 allows self-referencing in UPDATE/DELETE! A non-correlated subquery is evaluated once, after which WHERE is tested with valid rows being updated or deleted. Correlated subqueries usually require two steps, with the first creating a work file containing the RID for a DELETE and RID plus column value for an UPDATE. The second step reads the work file, repositions on the base

table record, and then either UPDATEs or DELETEs the required rows.

SUMMARY

V7 SQL enhancements will make life easier for DBAs and for those people who code SQL. The performance enhancements should make V7 run more efficiently.

A future article will discuss the final V7 SQL enhancement of scrollable cursors where you can move forwards and backwards, and position on a specific row within the result set.

Eric Garrigue Vesely
Principal/Analyst
Workbench Consulting (Malaysia)

Simplifying occasional, regular, and periodic tasks of the DBA – part 2

This month we conclude the procedures designed to simplify some occasional, regular, and periodic tasks of a database administrator.

```
NODBPRM  EQU   *
* OBTAIN <SQLQUERY>
         LA    R5,IRX_SHVBLOCK
         USING SHVBLOCK,R5
         MVI   SHVCODE,SHVFETCH
         MVC   SHVBUFL,=A(L'SQLQUERY)
         MVC   SHVVALA,=A(SQLQUERY)
         MVC   VN,=C'SQLQUERY '
         BAL   R14,GETVNL            GET LENGTH OF <VN>
         ST    RØ,SHVNAML            L(<VN>)
         MVC   SHVNAMA,=A(VN)        A(<VN>)
         L     R15,AIRXEXCOM
         CALL  (15),(IRX_IRXEXCOM,Ø,Ø,IRX_SHVBLOCK),VL
         LTR   R15,R15               REXX RETURN CODE
         BNZ   BLOWREXX              PARAMETER ERROR
         L     R1,SHVVALL            L(PARAMETER)
         STH   R1,SELECT
*        CONNECT TO DB2
         CALL  DSNALI,(CONNECT,SSID,TECB,SECB,RIBPTR),VL,MF=(E,CAFCALL)
         ST    R15,RETCODE
```

```
         ST    RØ,REASCODE
         CLC   RETCODE,=F'Ø'
         BNE   BLOW                  DB2 CONNECT ERROR
         CALL  DSNALI,(OPEN,SSID,PLAN),VL,MF=(E,CAFCALL)
         ST    RØ,REASCODE
         ST    R15,RETCODE
         CLC   RETCODE,=F'Ø'
         BNE   BLOWCON               DB2 PLAN ERROR
* OBTAIN POSSIBLE HOST VARIABLE <HOSTVAR>
         LA    R5,IRX_SHVBLOCK
         USING SHVBLOCK,R5
         MVI   SHVCODE,SHVFETCH
         MVC   SHVBUFL,=A(L'HOSTVAL)
         MVC   SHVVALA,=A(HOSTVAL)
         MVC   VN,=C'HOSTVAL '
         BAL   R14,GETVNL            GET LENGTH OF <VN>
         ST    RØ,SHVNAML            L(<VN>)
         MVC   SHVNAMA,=A(VN)        A(<VN>)
         L     R15,AIRXEXCOM
         CALL  (15),(IRX_IRXEXCOM,Ø,Ø,IRX_SHVBLOCK),VL
         LTR   R15,R15               TEST REXX RETURN CODE
         BNZ   DB2ST                 <> Ø, NO HOSTVAR SUPPLIED
         MVC   HOSTSW,YES            YES, THERE IS
         L     R1,SHVVALL            L(PARAMETER)
         STH   R1,HOSTVAR            STORE LENGTH
* DETERMINE NO OF COLUMNS IN TABLE
DB2ST    DS    ØH
         LA    R9,SQL_CA
         USING SQLDSECT,R9
         LA    R8,SQL_DA
         USING SQLDA,R8
* DUMMY PREPARE
         EXEC  SQL PREPARE S1 FROM :SELECT
         BAL   R14,CHECK_SQL
         MVC   SQLN,=H'Ø'            RETURN COUNT ONLY
* DESCRIBE
         EXEC  SQL DESCRIBE S1 INTO :SQL_DA
         BAL   R14,CHECK_SQL
* ANALYSE DESCRIPTOR AND ACQUIRE STORAGE
* <SQLD>: NO. OF COLUMNS
         LH    R2,SQLD
         MVC   RC,=F'Ø'              RESET RETURN CODE
         LTR   R2,R2
         BZ    A5ØØ                   :NON-SELECT
         LH    R3,SQLD
         MH    R2,=AL2(SQLVARN_SIZE)
         LA    R2,(SQLVAR-SQLDA)(R2)            +L(FIXED HDR)
* R2: TOTAL COLUMN SIZE
         ST    R2,COLSIZE
         GETMAIN EU,LV=(2),A=A_SQLDA
         L     R8,A_SQLDA
```

```
* PERFORM DESCRIBE WITH CORRECT LENGTH
         STH   R3,SQLD
         STH   R3,SQLN
         ST    R2,SQLDABC
* DESCRIBE
         EXEC  SQL DESCRIBE S1 INTO :SQLDA
         BAL   R14,CHECK_SQL
* BUILD ROW BUFFER
         LH    R6,SQLD             NO OF OCCURRENCES (COLUMNS)
         LTR   R6,R6
         BZ    EOP                 NO ENTRIES
         LA    R7,SQLVAR
         USING SQLVARN,R7
         SR    R4,R4               ZEROIZE BUFFER SIZE ACCUMULATOR
* OUTPUT NO OF COLUMNS (<_VN.Ø>)
         CNOP  Ø,4
         CVD   R6,D
         MVC   WK,=X'FØ2Ø2Ø2Ø2Ø2Ø2Ø2Ø'
         ED    WK,D+4
         LA    R3,WK                 A(DATA)
         MVC   VL,=A(L'WK)           L(DATA)
         MVC   VNINDEX,=C'.Ø'
         MVC   VLINDEX,=A(2)
         MVC   VN,=C'_VN '
* SET DATA INTO RESS VARIABLE
         BAL   R14,SETVAR
A23Ø     DS    ØH
* COLUMN TYPE
         BAL   R14,GET_TYPE
* R2: DATA WIDTH
* R3: FIELD SIZE
* R1Ø: DSECT_TYPE ENTRY
         USING DSECT_TYPE,R1Ø
         LA    R4,2(R3,R4)           ACCUMULATE TOTAL BUFFER SIZE
         AP    INDEX,=P'1'           INCREMENT INDEX
* CONVERT INDEX TO FORM: .N
         MVC   VNINDEX,=X'4Ø2Ø2Ø2Ø2Ø2Ø212Ø'
         LA    R1,VNINDEX+7
         EDMK  VNINDEX,INDEX
         BCTR  R1,Ø
         MVI   Ø(R1),C'.'
         MVC   VNINDEX,Ø(R1)
         LA    RØ,VNINDEX+L'VNINDEX
         SR    RØ,R1
         ST    RØ,VLINDEX            L(INDEX)
         MVC   VN,=C'_VN '           NAME PREFIX
         LA    R3,SQLNAME+2          COLUMNS NAME
         LH    RØ,SQLNAME
* NEW CODE FOR UNNAMED COLUMNS (EG COUNT(*) )
* TEST WHETHER  NULL COLUMN NAME
         LH    RØ,SQLNAME            LOAD COLUMN NAME
```

```
         LTR   RØ,RØ
         BNZ   A231                NO COLUMN NAME
         MVC   SQLNAME+2(8),=CL8'_VN'
         AP    VNCT,=P'1'
         UNPKSQLNAME+5(1),VNCT
         OI    SQLNAME+5,C'Ø'
         LA    RØ,5
         STH   RØ,SQLNAME
         LH    RØ,SQLNAME
A231     EQU   *
         ST    RØ,VL
* SET DATA INTO REXX VARIABLE
         BAL   R14,SETVAR
         LA    R7,SQLVARN_SIZE(R7)
         BCT   R6,A23Ø
* END OF SCAN PHASE, ALLOCATE DATA BUFFER
* R5: TOTAL BUFFER SIZE
         ST    R4,BUFFSIZE
         GETMAIN EU,LV=(4),A=A_DBUF
* COMPLETE SQLDA
         L     R5,A_DBUF           PTR(DATA BUFFER)
         LH    R6,SQLD             # OF OCCURRENCES COLUMNS
         LA    R7,SQLVAR           REINIT POINTER
* COLUMN TYPE
A24Ø     BAL   R14,GET_TYPE
* R3: FIELD SIZE
* R1Ø: DSECT_TYPE ENTRY
         ST    R5,SQLIND           A(INDICATOR), IF NEEDED
         LA    R5,2(R5)            UPDATE PTR
         ST    R5,SQLDATA          A(HOST VARIABLE)
         AR    R5,R3               UPDATE PTR
         LA    R7,SQLVARN_SIZE(R7)
         BCT   R6,A24Ø
* RETRIEVE RECORDS
* DECLARE CURSOR
         EXEC  SQL DECLARE CSR CURSOR FOR S1
         BAL   R14,CHECK_SQL
* OPEN CURSOR
         EXEC  SQL OPEN CSR
         BAL   R14,CHECK_SQL
         ZAP   INDEX,=P'Ø'        INITIALIZE ROW INDEX
A4ØØ     DS ØH
* FETCH ROW
         EXEC  SQL FETCH CSR USING DESCRIPTOR :SQLDA
         CLC   SQLCODE,=F'1ØØ'
         BE    EOD                END OF DATA
         AP    INDEX,=P'1'        INCREMENT INDEX
*  CONVERT INDEX TO FORM: .N
         MVC   VNINDEX,=X'4Ø2Ø2Ø2Ø2Ø2Ø212Ø'
         LA    R1,VNINDEX+7
         EDMK  VNINDEX,INDEX
```

```
         BCTR  R1,Ø
         MVI   Ø(R1),C'.'
         MVC   VNINDEX,Ø(R1)
         LA    RØ,VNINDEX+L'VNINDEX
         SR    RØ,R1
         ST    RØ,VLINDEX
         LH    R6,SQLD              NO OF OCCURRENCES (COLUMNS)
         LA    R7,SQLVAR            REINITIALIZE POINTER
* WRITE TABLE ROW
A41Ø     DS    ØH
         USING SQLVARN,R7
* DEFAULT (NULL) VALUE
         LA    R2,1
         LA    R3,=C'-'
         L     R1,SQLIND            A(INDICATOR FIELD)
         LH    RØ,Ø(R1)
         CH    RØ,=H'-1'
         BE    A42Ø                 NO DATA
         BAL   R14,GET_TYPE
* R2: DATA WIDTH
* R3: FIELD SIZE
* R1Ø: DSECT_TYPE ENTRY
         MVC   A_DSADDR,DS_ADDR
         L     R15,A_DSADDR         A(ROUTINE)
         BALR  R14,R15
* R3: A(FORMATTED FIELD)
* R2: L(FORMATTED FIELD)
A42Ø     ST    R2,VL
         LH    R1,SQLNAME           L(NAME)
         LA    RØ,L'VN              MAX(L(NAME))
         CR    R1,RØ
         BNH   *+6
         LR    R1,RØ                TRUNCATE
         BCTR  R1,Ø                 LC(NAME)
         MVC   VN,VN-1
         MVC   VN,SQLNAME+2         COLUMN NAME
         EX    R1,*-6
* SET DATA INTO REXX VARIABLE
         BAL   R14,SETVAR
         LA    R7,SQLVARN_SIZE(R7)
         BCT   R6,A41Ø              GET NEXT COLUMN IN ROW
         B     A4ØØ                 GET NEXT ROW
A5ØØ     DS    ØH                   PROCESS NON-SELECT
         CLC   HOSTSW,YES
         BNE   NOVARS
         EXEC  SQL EXECUTE S1 USING :HOSTVAR
         BAL   R14,CHECK_SQL
         B     EOD
NOVARS   DS    ØH
         EXEC  SQL EXECUTE S1
         BAL   R14,CHECK_SQL
```

```
EOD      CALL  DSNALI,(CLOSE,SYNC),VL,MF=(E,CAFCALL)
* END OF PROCESSING
EOP      DS    ØH
* OUTPUT # OF ROWS PROCESSED
         MVC   WK,=X'FØ202020202020202Ø'
         ED    WK,INDEX
         LA    R3,WK             A(DATA)
         MVC   VL,=A(L'WK)       L(DATA)
         MVC   VLINDEX,=A(Ø)     NO INDEX
         MVC   VN,=C'_NROWS '
* SET DATA INTO REXX VARIABLE
         BAL   R14,SETVAR
* RELEASE ALLOCATED STORAGE
         L     R2,COLSIZE
         LTR   R2,R2
         BZ    NOCOLS
         L     R3,A_SQLDA
         FREEMAIN R,LV=(2),A=(3)
NOCOLS   L     R2,BUFFSIZE
         LTR   R2,R2
         BZ    NOBUF
         L     R3,A_DBUF
         FREEMAIN R,LV=(2),A=(3)
NOBUF    EQU   *
*        DISCONNECT FROM DB2
         CALL  DSNALI,(DISCON),VL,MF=(E,CAFCALL)
GETOUT   EQU   *
         L     R13,4(R13)            RESTORE A(OLD SAVE AREA)
         L     R15,RC                SET RETURN CODE
         EINDE SAVE=SA,RCR=R15
BLOWREXX EQU   *
         MVC   RC,=F'8'
         B     NOBUF
BLOWCON  EQU   *
         MVC   RC,RETCODE
         MVC   CS4BYTE,REASCODE
         BAS   R14,HEXCONV             CONVERT TO EBCDIC
         MVC   REASWORK,CS8BYTE
         LA    R3,REASWORK             A(DATA)
         MVC   VL,=A(L'REASWORK)       L(DATA)
         MVC   VLINDEX,=A(Ø)           NO INDEX
         MVC   VN,=C'_REASON '
* SET DATA INTO REXX VARIABLE
         BAL   R14,SETVAR
         B     NOBUF
BLOW     EQU   *
         MVC   RC,=F'99'
         B     GETOUT
RC       DS    F                       PROGRAM RETURN CODE
*  CAF VARIABLES
```

```
HOSTSW   DC    CL1' '
REASWORK DS    CL8
YES      DC    CL1'Y'
SYNC     DC    CL4'SYNC'
CLOSE    DC    CL12'CLOSE       '
OPEN     DC    CL12'OPEN        '
CONNECT  DC    CL12'CONNECT     '
DISCON   DC    CL12'DISCONNECT  '
RETCODE  DS    F
REASCODE DS    F
RIBPTR   DS    F
SECB     DS    F
TECB     DS    F
SSID     DS    CL4
PLAN     DC    CL8'SQLISPF '
LIALI    DS    F
CAFCALL  CALL ,(*,*,*,*,*),VL,MF=L
CAFLEN   EQU  *-CAFCALL
*  END CAF VARIABLES
         TITLE 'SUBROUTINES'
*-------------------------------------------------------------------*
*  SUBROUTINE HEXCONV - CONVERTS ADDRESSES TO PRINTABLE HEX EBCDIC   *
*  INPUT: CS4BYTE       OUTPUT: CS8BYTE                              *
*  CSECT MUST BE 240 BYTES OR LONGER                                 *
*-------------------------------------------------------------------*
HEXCONV  DS    ØH
         B     CODING
SAVEHC   DS    18F                    SAVE AREA HEXCONV
CS5BYTE  DS    ØCL5
CS4BYTE  DS    CL4
         DS    C
CS9BYTE  DS    ØCL9
CS8BYTE  DS    CL8
         DS    C
         DC    C'$'
TRTABLE  ORG   *-X'FØ'
         ORG   TRTABLE+X'FØ'
         DC    C'Ø123456789ABCDEF'
CODING   STM   RØ,R15,SAVEHC
         UNPK  CS9BYTE,CS5BYTE
         TR    CS8BYTE,TRTABLE
EXITHC   DS    ØF
         LM    RØ,R15,SAVEHC
         BR    R14
GETVNL   DS    ØH                  DETERMINE ACTUAL LENGTH OF NAME
* INPUT: <VN> - NAME
* OUTPUT: RØ - L(NAME)
* R15 - A(FIRST BLANK)
         LA    R1,L'VN
         SR    RØ,RØ               COUNTER
```

```
         LA    R15,VN
GETVNL1  CLI   Ø(R15),C' '
         BER   R14                    END FOUND
         AH    RØ,=H'1'
         LA    R15,1(R15)
         BCT   R1,GETVNL1
* RØ: L(NAME). WITHOUT TRAILING BLANKS
         BR    R14
         DS    A
SETVAR   ST    R14,SETVAR-4           SET REXX VARIABLE
* <VN>: VAR1ABLE NAME, PREFIX
* <VNINDEX>: VARIABLE NAME, SUFFIX
* <VLINDEX>: LENGTH (VARIABLE NAME, SUFFIX)
* <VL>: L(VARIABLE DATA)
* R3: A(VARIABLE DATA)
         BAL   R14,GETVNL         GET L(VN)
* RØ: L(VN), R15: A(FIRST BLANK IN <VN>)
         MVC   Ø(L'VNINDEX,R15),VNINDEX
         A     RØ,VLINDEX
         LA    R5,IRX_SHVBLOCK
         USING SHVBLOCK,R5
         ST    RØ,SHVNAML
         MVC   SHVNAMA,=A(VN)
         MVI   SHVCODE,SHVSTORE
         ST    R3,SHVVALA
         MVC   SHVVALL,VL
         L     R15,AIRXEXCOM           A(IRXEXCOM)
         CALL  (15),(IRX_IRXEXCOM,Ø,Ø,IRX_SHVBLOCK),VL
         L     R14,SETVAR-4
         BR    R14                     RETURN
GET_TYPE DS    ØH                      GET COLUMN TYPE AND SIZE(S)
* INPUT:
* DSECT_SQLVARN ENTRY
* OUTPUT:
* R2: DATA WIDTH
* R3: FIELD SIZE
* DSECT TYPE ENTRY
         LA    R1Ø,T_TYPE-DS_L
         USING DSECT_TYPE,R1Ø
GETTYPE1 LA    R1Ø,DS_L(R1Ø)
         CLC   SQLTYPE,DS_TYPE
         BNE   GETTYPE1
* ENTRY FOUND, GET COLUMN-LENGTH
         LH    R2,SQLLEN
         LR    R3,R2                   PRESET DATA FIELD SIZE
         CLC   DS_CODE,=CL2'P'        (PACKED) DECIMAL?
         BNE   GETTYPE2                :NO
         SR    R2,R2
         IC    R2,SQLPRCSN             PRECISION
         LR    R3,R2
```

```
         SRL   R3,1                  NO OF DIGIT PAIRS
         LA    R3,1(R3)              TRUE DATA FIELD SIZE
GETTYPE2 CLC   DS_CODE,=CL2'CV'      CHARACTER (VARIABLE)?
         BNE   GETTYPE3             :NO
         LA    R3,2(R3)              ALLOC ROOM FOR LENGTH
GETTYPE3 BR    R14                   RETURN
         TITLE 'CONVERSION ROUTINES'
T_TYPE   DS    ØH                    ALIGNMENT
         DC    H'384',AL1(@CHAR,Ø),CL2'D ',AL4(D_DATE)
         DC    H'385',AL1(@CHAR,Ø),CL2'D ',AL4(D_DATE)
         DC    H'388',AL1(@CHAR,Ø),CL2'T ',AL4(D_TIME)
         DC    H'389',AL1(@CHAR,Ø),CL2'T ',AL4(D_TIME)
         DC    H'392',AL1(@CHAR,Ø),CL2'TS',AL4(D_TIME)
         DC    H'393',AL1(@CHAR,Ø),CL2'TS',AL4(D_TIME)
         DC    H'448',AL1(@CHAR,Ø),CL2'CV',AL4(D_CHARV)
         DC    H'449',AL1(@CHAR,Ø),CL2'CV',AL4(D_CHARV)
         DC    H'452',AL1(@CHAR,Ø),CL2'C ',AL4(D_CHAR)
         DC    H'453',AL1(@CHAR,Ø),CL2'C ',AL4(D_CHAR)
         DC    H'456',AL1(@CHAR,Ø),CL2'CV',AL4(D_CHAR)
         DC    H'457',AL1(@CHAR,Ø),CL2'CV',AL4(D_CHAR)
         DC    H'484',AL1(@NUM,Ø),CL2'P ',AL4(D_DEC)
         DC    H'485',AL1(@NUM,Ø),CL2'P ',AL4(D_DEC)
         DC    H'496',AL1(@NUM,Ø),CL2'I ',AL4(D_INT)
         DC    H'497',AL1(@NUM,Ø),CL2'I ',AL4(D_INT)
         DC    H'5ØØ',AL1(@NUM,Ø),CL2'I ',AL4(D_SIN)
         DC    H'5Ø1',AL1(@NUM,Ø),CL2'I ',AL4(D_SIN)
         DC    H'Ø'                        EOT
D_DATE   EQU   D_CHAR
D_TIME   EQU   D_CHAR
D_CHARV  EQU   D_CHAR
@NULL    EQU   X'Ø1'
@CHAR    EQU   1
@NUM     EQU   2
D_CHAR   DS    ØH                    CHARACTER
         L     R3,SQLDATA            A(DATA)
         LH    R2,SQLLEN             L(DATA)
         CLC   DS_CODE,=C'CV'
         BNER  R14
         LH    R2,Ø(R3)
         LA    R3,2(R3)
         BR    R14
D_DEC    DS    ØH                    PACKED DECIMAL
         L     R3,SQLDATA            A(DATA)
         SR    R1,R1
         IC    R1,SQLLEN             L(DATA), PRECISION
* R1: NO. OF DECIMAL DIGITS
         SRL   R1,1
* R1: LENGTH CODE OF PACKED FIELD
         EX    R1,ZAP
         B     FMT_DEC               FORMAT DECIMAL FIELD
```

```
D_SIN    DS    ØH                      BINARY SMALL INTEGER (2 BYTES)
         L     R3,SQLDATA              A(DATA) IN R3
         MVC   H2(2),Ø(R3)             MAKE SURE ITS HALFWORD BOUNDARY
         LH    RØ,H2
         CNOP  Ø,4
         CVD   RØ,D                    CONVERT IT TO DECIMAL
         B     FMT_DEC                 FORMAT DECIMAL FIELD
D_INT    DS    ØH                      BINARY INTEGER
         L     R3,SQLDATA              A(DATA)
         LH    R1,SQLLEN               L(DATA)
* R1: FIELD SIZE
* CREATE MASK FOR ICM INSTRUCTION
         L     R2,=X'ØØØØØØØF'         LOAD MASK
         SLL   R2,Ø(R1)
         SRL   R2,4
         N     R2,=X'ØØØØØØØF'         R2: ICM MASK
         SR    RØ,RØ
         EX    R2,ICM                  LOAD BINARY VALUE INTO RØ
         CNOP  Ø,4
         CVD   RØ,D                    CONVERT IT TO DECIMAL
         B     FMT_DEC                 FORMAT DECIMAL FIELD
FMT_DEC  DS    ØH                      FORMAT DECIMAL FIELD
* INPUT:
* <D>: PACKED DECIMAL FIELD
* R14: RETURN ADDRESS
* OUTPUT:
* R2: L(FORMATTED FLD)
* R3: A(FORMATTED FLD)
         MVC   EDWK,EDMKINT            MOVE MASK
         LA    R1,EDWK+L'EDWK-2
         EDMK  EDWK,D
* R1: 1ST SIGNIFICANT CHARACTER
         LA    R2,EDWK_E             END ADDR
         SR    R2,R1                 L(FORMATTED FLD)
         LR    R3,R1                 A(FORMATTED FLD)
         BR    R14
* EX INSTRUCTIONS
ZAP      ZAP   D,Ø(Ø,R3)
ICM      ICM   RØ,Ø,Ø(R3)
* WORK FIELDS
SCALE    DS    F
H2       DS    H
FILLER2  DS    ØD
D        DS    PL8
WK       DS    CL8
EDWK     DS    CL17
EDWK_E   EQU   *
EDMKINT  DC    X'4Ø',13X'2Ø',X'212Ø',C'-'      EDIT MASK INTEGERS
         DS A
CHECK_SQL ST   R14,CHECK_SQL-4     CHECK SQLCODE
```

```
         L     R15,SQLCODE
         LTR   R15,R15
         BZR   R14                  :SOL OK
         MVC   RC,SQLCODE
         B     EOD
         TITLE 'DATA AREAS'
STARTWRK DS    ØF
A_DEBUG  DS    A                   VALENTINO
A_SQLDA  DS    A                   A(ALLOCATED SQLDA)
A_DBUF   DS    A                   A(DATA BUFFER)
A_DSADDR DS    AL4
COLSIZE  DC    F'Ø'                COLUMN SIZE
BUFFSIZE DC    F'Ø'                BUFFER SIZE
         DC    C' '                CLEAR BYTE
VN       DS    2CL18               VARIABLE NAME
VL       DS    A                   VARIABLE LENGTH
VNCT     DC    PL4'Ø'              GENERATED NAME COUNT (MAX 9)
INDEX    DC    PL4'Ø'              ROW INDEX
VNINDEX  DS    2CL8
VLINDEX  DS    F
SQL_CA   DS    CL(SQLDLEN)         BASIC SQLCA
SQL_DA   DS    CL16                BASIC SQLDA
         EXEC  SQL INCLUDE SQLCA
         EXEC  SQL INCLUDE SQLDA
SQLVARN_SIZE EQU 44
         LTORG
AIRXEXCOM DS A
IRX_IRXEXCOM DC CL8'IRXEXCOM'
         DS    ØA                  ALIGN
IRX_SHVBLOCK DC (SHVBLEN)X'Ø'
DB2VAR   DC    H'4',CL4' '
         ORG   DB2VAR+2
DB2V     DS    CL4
SELECT   DC    H'8Ø',CL8Ø' '
         ORG   SELECT+2
SQLQUERY DS    CL4Ø96
HOSTVAR  DS    H,CL32760
         ORG   HOSTVAR+2
HOSTVAL  DS    CL32760
         TITLE 'DSECTS'
DSECT_TYPE DSECT
DS_TYPE  DS    HL2
DS_GEN   DS    AL1            GENERIC TYPE (NUMERIC, CHARACTER)
         DS    AL1            FILLER
DS_CODE  DS    CL2
DS_ADDR  DS    AL4
DS_L     EQU   *-DS_TYPE
         IRXSHVB              DEFINITION OF REXX SHVB
         END
//*
```

```
//*            ASSEMBLE IF THE PRECOMPILE RETURN CODE
//*            IS 4 OR LESS
//*
//ASM     EXEC PGM=ASMA90,PARM='OBJECT,NODECK',COND=(4,LT,PC)
//SYSIN    DD  DSN=&&DSNHOUT,DISP=(OLD,DELETE)
//SYSLIB   DD  DSN=SYS1.MACLIB,DISP=SHR
//         DD  DSN=SYS1.MODGEN,DISP=SHR
//         DD  DSN=SYSADM.USER5.ASM,DISP=SHR
//SYSLIN   DD  DSN=&&LOADSET,DISP=(MOD,PASS),UNIT=SYSDA,
//             SPACE=(800,(10,10)),DCB=(BLKSIZE=800)
//SYSPRINT DD  SYSOUT=*
//SYSUDUMP DD  SYSOUT=*
//SYSUT1   DD  SPACE=(800,(10,10),,,ROUND),UNIT=SYSDA
//*
//*            LINKEDIT IF THE PRECOMPILER AND ASSEMBLER
//*            RETURN CODES ARE 4 OR LESS
//*
//LKED    EXEC PGM=IEWL,PARM='XREF,AMODE=31,RMODE=ANY',
//             COND=((4,LT,ASM),(4,LT,PC))
//SYSLIB   DD  DISP=SHR,DSN=DSN510.SDSNLOAD
//         DD  DSN=SYS1.DSN510.SDSNEXIT,DISP=SHR
//SYSLIN   DD  DSN=&&LOADSET,DISP=(OLD,DELETE)
//         DD  DDNAME=SYSIN
//*SYSLMOD  DD DSN=POSTP.BASEV5.LOAD(SQLISPF),DISP=SHR
//SYSLMOD  DD DSN=SYSADM.USER5.LOAD(SQLISPF),DISP=SHR
//SYSPRINT DD  SYSOUT=*
//SYSUDUMP DD  SYSOUT=*
//SYSUT1   DD  SPACE=(1024,(50,50)),UNIT=SYSDA
//SYSIN    DD  *
  INCLUDE SYSLIB(DSNALI)
  NAME SQLISPF(R)
/*
//*
//BINDPROD EXEC PGM=IKJEFT01,DYNAMNBR=20,COND=(4,LT)
//DBRMLIB  DD DSN=SYSADM.DBRMLIB5.DATA,DISP=SHR
//SYSTSPRT DD SYSOUT=*
//SYSPRINT DD SYSOUT=*
//SYSUDUMP DD SYSOUT=*
//SYSIN   DD *
   GRANT BIND, EXECUTE ON PLAN SQLISPF TO PUBLIC;
//SYSTSIN DD *
 DSN SYSTEM(DSN)
 BIND PLAN(SQLISPF) MEMBER(SQLISPF) ACT(REP) ISO(CS) EXPLAIN(NO)
 RUN  PROGRAM(DSNTIAD) PLAN(DSNTIA51) -
      LIB('DSN510.RUNLIB.LOAD')
 END
//*
//BINDTEST EXEC PGM=IKJEFT01,DYNAMNBR=20,COND=(4,LT)
//DBRMLIB  DD DSN=SYSADM.DBRMLIB5.DATA,DISP=SHR
//SYSTSPRT DD SYSOUT=*
```

```
//SYSPRINT DD SYSOUT=*
//SYSUDUMP DD SYSOUT=*
//SYSIN   DD *
   GRANT BIND, EXECUTE ON PLAN SQLISPF TO PUBLIC;
//SYSTSIN DD *
 DSN SYSTEM(DBT)
 BIND PLAN(SQLISPF) MEMBER(SQLISPF) ACT(REP) ISO(CS) EXPLAIN(NO)
 RUN  PROGRAM(DSNTIAD) PLAN(DSNTIA51) -
      LIB('DBT51Ø.RUNLIB.LOAD')
 END
//

BEGIN

*
*     REQUIRED KEYWORDS:  SAVE AND BASE
*     DEPENDENCIES : GLOBAL PARAMETERS ALSO USED IN MACRO "DATE"
*

         MACRO
&NAME    BEGIN &SAVE=,&BASE=,&PARM=,&DATE=,&TIME=,&PRINT=ON
         SPACE 2
*************************************************************************
*                                                                       *
*                                                                       *
*************************************************************************
         SPACE 2
         AIF   ('&PRINT' NE 'NOGEN').A1
         PRINT &PRINT
.A1      ANOP
         GBLC  &SYDNBØ1,&SYDNBØ2,&SYDNBØ3,&SYDNBØ4
         GBLC  &SYDNBØ5,&SYDNBØ6,&SYDNBØ7
         LCLC  &B,&C,&D,&E,&F,&N,&O,&P
&B       SETC  'B'.'&SYSNDX'.'AA'
&C       SETC  'B'.'&SYSNDX'.'HW'
&D       SETC  'B'.'&SYSNDX'.'AC'
&E       SETC  'B'.'&SYSNDX'.'AB'
&F       SETC  'B'.'&SYSNDX'.'AD'
&N       SETC  'B'.'&SYSNDX'.'AE'
&O       SETC  'B'.'&SYSNDX'.'AF'
&P       SETC  'B'.'&SYSNDX'.'ED'
         AIF   ('&SAVE' EQ '').FT1
         AIF   ('&BASE' EQ '').FT2
         AIF   ('&PRINT' EQ 'NOGEN').AØ
         AIF   ('&PRINT' EQ 'ON').AØ
         PRINT &PRINT
.AØ      ANOP
&NAME    DS    ØH
         B     6Ø(15)            BRANCH AROUND CONSTANTS
MY_CSECT DC    CL1Ø'&SYSECT'     CSECT NAME
```

```
         DC    C'&SYSDATE'       ASSEMBLY DATE
         DC    C' '
         DC    C'&SYSTIME'       ASSEMBLY TIME
         DC    C' '
         DC    C' ¸ DE NEDERLANDSCHE BANK N.V.  '
         STM   14,12,12(13)
         CNOP  2,4
         BALR  &BASE(1),Ø
         USING *,&BASE(1)        1ST BASE
.*
         AIF   (N'&BASE LT 5).A
         MNOTE 4,'*** MAXIMUM NUMBER OF BASE-REGISTERS IS FOUR. ***'
.*
.A       ANOP
         AIF   ('&BASE(2)' EQ '').B
&B       EQU   *
         L     &BASE(2),*+8
         USING &B+4Ø96,&BASE(2)    2ND BASE
         B     *+8
         DC    A(&B+4Ø96)
*
         AIF   ('&BASE(3)' EQ '').B
         L     &BASE(3),*+8
         USING &B+8192,&BASE(3)    3RD BASE
         B     *+8
         DC    A(&B+8192)
*
         AIF   ('&BASE(4)' EQ '').B
         L     &BASE(4),*+8
         USING &B+12288,&BASE(4)    4TH BASE
         B     *+8
         DC    A(&B+12288)
*
.B       ANOP
         LA    14,&SAVE
         ST    14,8(13)
         ST    13,&SAVE+4
         LA    13,&SAVE
*
         AIF   ('&PARM' EQ '').C
         L     14,Ø(1)
         LH    15,Ø(14)
         CH    15,&C
         BE    &D
         BCTR  15,Ø
         EX    15,&E
         B     &D
*
&E       MVC   &PARM.(1),2(14)
&C       DC    H'Ø'
```

```
*
&D       EQU   *
.C       ANOP
         AIF   ('&DATE' EQ '' AND '&TIME' EQ '').H
.*       AIF   ('&SYDNBØ6' NE '').CNT1Ø
&SYDNBØ6 SETC  'B'.'&SYSNDX'.'FW'
.CNT1Ø   ANOP
         ST    1,&SYDNBØ6                      SAVE REG 1
         LA    1,2(Ø,Ø)                        SPECIFY MAGNITUDE
         SVC   11                              GET TIMER
         B     &F                              BRANCH AROUND CONSTANTS
*
.*       AIF   ('&SYDNBØ1' NE '').CNT2Ø
&SYDNBØ1 SETC  'B'.'&SYSNDX'.'DB'
&SYDNBØ1 DC    D'Ø'
.CNT2Ø   ANOP
.*       AIF   ('&SYDNBØ6' NE 'B&SYSNDX.FW').CNT3Ø
&SYDNBØ6 DS    F
.CNT3Ø   AIF   ('&DATE' EQ '').D
.*       AIF   ('&SYDNBØ2' NE '').CNT4Ø
&SYDNBØ2 SETC  'B'.'&SYSNDX'.'P1'
&SYDNBØ2 DC    PL2'1'
.CNT4Ø   ANOP
.*       AIF   ('&SYDNBØ3' NE '').CNT5Ø
&SYDNBØ3 SETC  'B'.'&SYSNDX'.'PØ'
&SYDNBØ3 DC    PL2'Ø'
.CNT5Ø   ANOP
.*       AIF   ('&SYDNBØ4' NE '').D
&SYDNBØ4 SETC  'B'.'&SYSNDX'.'C19'
&SYDNBØ4 DC    C'19'
.*
.D       ANOP
         AIF   ('&TIME' EQ '').E
.*       AIF   ('&SYDNBØ7' NE '').E
&SYDNBØ7 SETC  'B'.'&SYSNDX'.'ED'
&SYDNBØ7 DC    CL14' '
.*
.E       ANOP
         AIF   ('&DATE' EQ '').F
.*       AIF   ('&SYDNBØ5' NE '').CNT6Ø
&SYDNBØ5 SETC  'B'.'&SYSNDX'.'TAB'
&SYDNBØ5 DC    PL2'31',C'JAN.'
         DC    PL2'28',C'FEB.'
         DC    PL2'31',C'MRT.'
         DC    PL2'3Ø',C'APR.'
         DC    PL2'31',C'MEI '
         DC    PL2'3Ø',C'JUNI'
         DC    PL2'31',C'JULI'
         DC    PL2'31',C'AUG.'
         DC    PL2'3Ø',C'SEPT'
```

```
         DC    PL2'31',C'OKT.'
         DC    PL2'3Ø',C'NOV.'
         DC    PL2'31',C'DEC.'
*
.CNT6Ø   ANOP
&F       EQU   *                    SUPPLY DATE
         ST    1,&SYDNBØ1+4
         MVO   &SYDNBØ1,&SYDNBØ1.(6)
         MVC   &DATE+8(2),&SYDNBØ4
         UNPK  &DATE+1Ø(2),&SYDNBØ1
         OI    &DATE+11,X'FØ'
         CVB   14,&SYDNBØ1
         ST    14,&SYDNBØ1
         TM    &SYDNBØ1+3,B'ØØØØØØ11'
         BNZ   *+1Ø
         AP    &SYDNBØ5+6(2),&SYDNBØ2
         ST    1,&SYDNBØ1
         XC    &SYDNBØ1.(2),&SYDNBØ1
         ZAP   &SYDNBØ1+4(4),&SYDNBØ3
         LA    1,&SYDNBØ5
         LA    14,12
*
&N       EQU   *
         AP    &SYDNBØ1+4(4),Ø(2,1)
         CP    &SYDNBØ1.(4),&SYDNBØ1+4(4)
         BNH   &O
         LA    1,6(1)
         BCT   14,&N
         DC    X'FFFF'
*
&O       EQU   *
         SP    &SYDNBØ1+4(4),Ø(2,1)
         SP    &SYDNBØ1.(4),&SYDNBØ1+4(4)
         UNPK  &DATE.(2),&SYDNBØ1+2(2)
         OI    &DATE+1,X'FØ'
         MVC   &DATE+3(4),2(1)
*
.F       ANOP
         AIF   ('&DATE' NE '').G
&F       EQU   *                  SUPPLY TIME
.G       ANOP
         AIF   ('&TIME' EQ '').I
         XC    &SYDNBØ1,&SYDNBØ1
         ST    Ø,&SYDNBØ1+4
         MVC   &SYDNBØ1+3(4),&SYDNBØ1+4
         MVI   &SYDNBØ1+7,X'ØC'
         MVO   &SYDNBØ1.(8),&SYDNBØ1.(7)
         MVC   &SYDNBØ7.(13),=X'4Ø2Ø21214B21214B21216B2121'
         ED    &SYDNBØ7.(13),&SYDNBØ1+3
         MVC   &TIME.(11),&SYDNBØ7+2
.I       ANOP
```

```
         L     1,&SYDNBØ6
.H       ANOP
         AIF   ('&PRINT' EQ 'NOGEN').K
         AIF   ('&PRINT' EQ 'ON').K
         PRINT ON
.K       ANOP
*                                    END "BEGIN"
         MEXIT
.FT1     MNOTE 8,'*** KEYWORD SAVE MISSING. ***'
         AGO   .H
.FT2     MNOTE 8,'*** KEYWORD BASE MISSING. ***'
         AGO   .H
         MEND
```

EINDE

```
         MACRO
&NAME    EINDE &SAVE=,&RC=,&RCR=,&PARMLST=
         AIF   ('&SAVE' EQ '').FT
         AIF   ('&RCR' NE '').B
&NAME    L     13,&SAVE+4
         LM    14,12,12(13)
         AIF   ('&PARMLST' EQ '').AØ
         LA    1,&PARMLST
.AØ      AIF   ('&RC' EQ '').A1
         LA    15,&RC.(Ø,Ø)
.A1      ANOP
         BR    14
*
         MEXIT
.B       ANOP
&NAME    DS    ØH
         LR    15,&RCR                          LOAD RETURNCODE
         L     13,&SAVE+4
         L     14,12(13)
         LM    Ø,12,2Ø(13)
         AIF   ('&PARMLST' EQ '').BØ
         LA    1,&PARMLST
.BØ      ANOP
         BR    14
*
         MEXIT
.FT      MNOTE 8,'KEYWORD SAVE MISSING'
         MEND
```

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

Imagecopy generator procedure

INTRODUCTION

I work in a big software house where there is a considerable amount of copying and recovering of data. To carry out frequent activities involving data manipulation in development and test environments, users often call on the support group to recover tablespaces to a specific time for consistency. In order to manage back-ups, recovery, and the SYSCOPY managment, I have created some batch REXX procedures and ISPF tools.

In this article I will write about the implementation of our imagecopy generator procedure, which performs the back-up of the data.

PROCEDURE DESCRIPTION

The procedure has been designed for centralized back-up management. It allows databases, single tablespaces, or a list of tablespaces to be inserted or removed from the list of objects to be COPYed using a View customization.

The procedure performs the following step:

- Displays and terminates the utility in a ‘stopped’ state.
- Displays the object to be copied in a ‘restricted’ state, removes them from the list of the object to be copied if they are in the ‘wrong’ state, and displays them in the batch procedure
- Takes the imagecopy.
- Modifies a tablespace to release a copy older than a specified value.
- Archive the sysout on dataset.

PARAMETER DESCRIPTION

The following parameters describe how you can customize the REXX EXEC batch procedure:

- Subsys – DB2 subsystem name.
- GDGid – GDG type S00 (for Weekly COPY list) G00 (for Daily COPY list).
- DELAGE – to delete a COPY older than this value from SYSCOPY.
- JobNum – this parameter specifies the number of the job to be generated.
- Hold – specifies whether the jobs generated must be submitted with or without the HOLD parameter.
- NextJob – this parameter allows the user to submit another job that he wants at the end of the imagecopy jobs.

CHECKLIST FOR INSTALLATION

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

- Allocate a USER.LIBRARY
- Copy all REXX, macro, proc, JCL, and Views into the USER.LIBRARY:
  - REXX – $DB2COP0, $DB2COP1, $DB2ACC0, DB2PAR0, $DB2ALL0, and $DB2OUBK.
  - Macro – $MDB2018,$MDB2042, and $MDB2043.
  - Proc – DB2REXX1.
  - Sample View – VDSNZG00, and VDSNZS00.
  - Sample job – $DB2IMG0, and CREAGDG.
- Customize $DB2PAR0 REXX for the global environment setting highlighted variables.
- Customize proc DB2REXX1 according to your environment.
- Customize job $DB2IMG0 according to your environment.
- Customize the sample job CREAGDG to create one or more GDGs for the COPY procedure.

- Create a member, IEBDDIN, in the user.library with the string:

  ```
  COPY OUTDD=OUT1,INDD=INP1
  ```

- Set the parameter 'ExitDD' in $DB2COP0 REXX. This parameter sets the maximum number of DDs that a job can allocate in your installation. This parameter currently is set to 2100

The sample jobs and parameter are customized for a DB2 subsystem name equal to DSNZ and the DB2 subsystem command prefix equal to '+'.

The test environment is DB2 V5 in an OS/390 V8 environment.

$DB2COP0 REXX EXEC

```
/* REXX */
   /**********  Full Image Copy generator  **********/
arg parmin ; parm = translate(parmin,' ',',') ; nparm = words(parm)
Subsys = word(parm,1);GDGid = word(parm,2) ; DelAGE  = word(parm,3)
JobNum = word(parm,4);Hold = word(parm,5) ; Nextjob = word(parm,6)
   /*---  Test input parameters                ---*/
 if nparm < 6 then do ; say '' ; say '' ; say '>>>>>>>>'
    say '>>>>>>>>  Parameter string is incomplete !!!!'
    say '>>>>>>>>           'parmin
    say '>>>>>>>>' ; say '' ; say '' ; exit ; end
 if JobNum = '*' then JobNum = 1
 if Hold = '*' then Hold = NO
 if Hold ¬= NO & hold ¬= YES then do
    say '' ; say '' ; say '>>>>>>>>'
    say '>>>>>>>>  Wrong Hold 'Hold' variable !!!!!'
    say '>>>>>>>>  Specify : */YES/NO              '
    say '>>>>>>>>' ; say '' ; say '' ; exit ; end
 if JobNum > 6 then do ; say '' ; say '' ; say '>>>>>>>>'
    say '>>>>>>>>  Wrong JobNum 'JobNum' variable !!!!!'
    say '>>>>>>>>  Specify a number between 1 and 6    '
    say '>>>>>>>>' ; say '' ; say '' ; exit ; end
 if DelAGE < 1 | DelAGE > 32767 then do ; say '' ; say '' ; say
'>>>>>>>>'
    say '>>>>>>>>  Wrong DelAGE 'DelAGE' variable !!!!! '
    say '>>>>>>>>  Specify a number between 1 and 32767 '
    say '>>>>>>>>' ; say '' ; say '' ; exit ; end
 if Nextjob = '*' then Nextjob =  NO
   /*---  Parameters assignment                ---*/
 call @db2parØ Subsys ; 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           ---*/
 ExitDD   = 2100 ; NumDD = 0 ; stp = 0 ; ctr0 = 0 ; ctst0 = 0 ; testata
= yes
 labe = 0 ; labepr = 0 ; #vet0 = 0 ; lfill0 = 0 ; wrk = 0 ; blk = ;
NexJid = 0
 Jobname  = SAWE || substr(Subsys,4,1) || substr(GDGid,1,1)
 outds1   = $hiwork'.'subsys'.@DB2COP1.SYSTSINP'
 outds2   = $hiwork'.'subsys'.@DB2COP1.SYSTSPRT'
 outdsrec = $hiwork'.'subsys'.@DB2COP0.SYSREC00'
 outdsrec1= $hiwork'.'subsys'.@DB2COP1.SYSREC00'
 outdstsin= $hiwork'.'subsys'.'$user'.SYSTSIN'
 outdsprt = $hiwork'.'subsys'.'$user'.SYSPRINT'
   /*---  SYSIN file allocation               ---*/
 call Free0 ; dsn = sysdsn(''''outdsin'''')
 if dsn ¬= OK then do ; say '' ; outdsin=
$hiwork'.'subsys'.'$user'.SYSIN'
    prmalloc = subsys' 'outdsin' 0 5,1 f,b 80 2720 sysin yes'
    call @db2all0 prmalloc ; if word(result,1) = 99 then exit ; end
 else "alloc da('"outdsin"') f("sysin") shr reuse"
 sk.1='  SELECT * FROM V'subsys||GDGid' ORDER BY 1,2 ;       '
 sk.0=1 ; jobw = sysin ; call WriteRec
 call @db2acc0 subsys','@DB2COP0
 if word(result,1) = 99 then do ; call Free ; exit ; end
 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 = ØØ then do
   /*---  Verify Max DD for job                ---*/
    NumDD = NRrec % JobNum
    if Numdd > ExitDD then do ; say '' ; say '' ; say '>>>>>>>>'
       say '>>>>>>>>                    A  T  T  E  N  T  I  O  N    '
       say '>>>>>>>> Exceeded  maximum number of DD allowed for an
ImageCopy job'
       say '>>>>>>>> Increase ExitDD value or define a new GDG for
backup and increase'
       say '>>>>>>>> JobNum value                                    '
       say '>>>>>>>>             ---  Max DD allowed    : 'ExitDD
       say '>>>>>>>>             ---  Max DD generated  : 'NumDD
       say '>>>>>>>>' ; say '' ; say '' ; call Free ; exit ; end
   /*---  Display Utility Stopped              ---*/
    call @DB2COP1 Subsys||',*,DISP' ; if word(result,1) = 99 then exit
   /*---  Display DB spacename Restrict        ---*/
    call @DB2COP1 Subsys',V'Subsys||GDGid',STAR'
    if word(result,1) = 99 then exit ; call AllocØØ ; call ElaboØØ
   /*---  Submit ImageCopy jobs                ---*/
    xx=outtrap(trpsubØ.) ; address tso "submit '"outdsimg"'"
    xx=outtrap(off)
    if rc > Ø then do
       do #a = 1 to trpsubØ.Ø ; say trpsubØ.#a ; end ; exit ; end
    mess  = substr(trpsubØ.1,1,22) ; say '' ; say '' ; say time()
    say time() ' 'ctstØ' Job of ImageCopy has been submitted ...'
    say time() ; say '' ; say '' ; end
 else do ; say '' ; say '>>>>>>>>' ; say '>>>>>>>>'
    say '>>>>>>>>    Unpredictable value of RCdb2. End elaboration '
    say '>>>>>>>>' ; say '>>>>>>>>' ; say '' ; end ; call Free ; exit
   /*---  Elaboration routine                  ---*/
 ElaboØØ :
   /*---  Read record selected                 ---*/
    jobw = sysrecØØ
    "alloc da('"outdsrec"') f("jobw") shr reuse"
    xx=outtrap(trpreadØ1.) ; "execio * diskr sysrecØØ (stem sysrecØØ.
finis"
    xx=outtrap(off)
    if rc > Ø then do ; do #a = 1 to trpreadØ1.Ø ; say trpreadØ1.#a ;
end
       say '' ; say '' ; say '>>>>>>>>'
       say '>>>>>>>>  Error reading file "'outdsrec'"    '
       say '>>>>>>>>  RC='rc'. Verify.                   '
       say '>>>>>>>>' ; say '' ; say '' ; exit ; end
    say ''
    say '  ____________________________________________________________ '
    say '*                                          'time()'     *'
    say '*        ImageCopy jobs building in progress               *'
    say '*                                                          *'
    say '*                        Wait please ........              *'
    say '*                                                          *'
```

```
    say '  ____________________________________________________________  '
    say ''
    welab25 = (sysrec00.0 * 25)%100 ; welab50 = (sysrec00.0 * 50)%100
    welab75 = (sysrec00.0 * 75)%100 ; welab00 = sysrec00.0
    DO #d = 1 to sysrec00.0
       #vet0   = #vet0 + 1 ; ctr0 = ctr0 + 1 ; stp = stp + 1
       stp     = right(stp,4,'0') ; labe = labe + 1
       $DBname = strip(substr(sysrec00.#d,1,8))
       $TSname = strip(substr(sysrec00.#d,9,16))
       lfill0 = length($DBname'.'$TSname) ; wrk = 17 - lfill0
   /*---  Write Header jobs                    ---*/
       if testata = yes then do
          testata = no  ; ctst0 = ctst0 + 1 ; numsav  = ctst0
          ctst0   = right(ctst0,2,'0') ; wgdgid = substr(GDGid,1,1)
          NexJid  = ctst0 + 1  ; NexJid  = right(NexJid,2,'0')
          if wGDGid = S then
             GDGname =
'SYSG.'subsys'.SAVEDATI.'wGDGid||ctst0'.COPYW(+1)'
          else
             GDGname =
'SYSG.'subsys'.SAVEDATI.'wGDGid||ctst0'.COPYG(+1)'
          data =,
          substr(date(s),7,2)'/'substr(date(s),5,2)'/
'substr(date(s),1,4)
          ora = time()
sk.1='//'Jobname||ctst0' JOB ('$accn'),''Image Copy'',CLASS='$class', '
sk.2='//      MSGCLASS='$msgcla',USER='$user',REGION='$region',       '
          if Hold = YES | ctst0 > 1 then do
sk.3='//      MSGLEVEL=('$msglvl'),NOTIFY='$notif',                   '
sk.4='//      TYPRUN=HOLD                                             '
             end
          else do
sk.3='//      MSGLEVEL=('$msglvl'),NOTIFY='$notif
sk.4='//*                                                             '
             end
sk.5='/*JOBPARM BYTES=999999,LINES=9999                               '
sk.6='//* ------------------------------------------------------ *    '
sk.7='//* ---                  Image Copy                      --- *    '
sk.8='//* ---          of : 'data' time : 'ora '                --- *    '
sk.9='//* ------------------------------------------------------ *    '
sk.10='//JOBLIB    DD  DISP=SHR,DSN='$dsnload
sk.11='//DB2PROC   JCLLIB ORDER=('$proclib')                          '
sk.12='//* ------------------------------------------------------ *   '
sk.13='//STEP00  EXEC
PGM=DSNUTILB,PARM='''Subsys',COPY'Jobname||ctst0''''
sk.14='//SYSPRINT DD  SYSOUT=*                                        '
sk.15='//SYSC0001 DD  DSN='GDGname',                                  '
sk.16='//         DISP=(,CATLG),LABEL=('labe',SL),VOL=(,RETAIN,,99),  '
sk.17='//         DCB=(SYSG.COPYG.PATTERN),UNIT='$unitta
          sk.0=17; jobw = fiimg
```

```
          "alloc da('"outdsimg"') f("jobw") mod reuse" ; call WriteRec
skcop.#vet0='COPY TABLESPACE '$DBname'.'$TSname copies(' ',wrk)'FULL YES
COPYDDN
 SYSC000'
skmod.#vet0='MODIFY RECOVERY TABLESPACE '$DBname'.'$TSname copies('
',wrk)'DELET
E AGE('DelAGE')'
          end
       else do
          labepr = labe - 1
sk.1='//SYSC'stp' DD
VOL=(,RETAIN,,99,REF=*.SYSC'right(labepr,4,'0')'),'
sk.2='//         DSN=*.SYSC0001,LABEL=('labe',SL),DCB=*.SYSC0001,    '
sk.3='//         DISP=(,KEEP,KEEP)                                   '
          sk.0=3; jobw = fiimg
          "alloc da('"outdsimg"') f("jobw") mod reuse" ; call WriteRec;
end
skcop.#vet0='COPY TABLESPACE '$DBname'.'$TSname copies(' ',wrk)'FULL YES
COPYDDN
 SYSC'stp
skmod.#vet0='MODIFY RECOVERY TABLESPACE '$DBname'.'$TSname copies('
',wrk)'DELET
E AGE('DelAGE')'
   /*---  Cut job                                ---*/
       if ctr0 > NumDD | #d = sysrec00.0 then do
sk.1='//SYSIN    DD  *                                              '
          sk.0=1; jobw = fiimg
          "alloc da('"outdsimg"') f("jobw") mod reuse" ; call WriteRec
   /*---  Write sysin Copy statments           ---*/
          skcop.0=#vet0
          jobw = fiimg
          "alloc da('"outdsimg"') f("jobw") mod reuse" ; call WriteCop
sk.1='//* ------------------------------------------------------ *    '
sk.2='//* ---            Modify DELETE AGE('DelAGE')         --- *    '
sk.3='//* ------------------------------------------------------ *    '
sk.4='//LAB0  IF (STEP00.RC EQ 0) THEN                              '
sk.5='//STEP01  EXEC
PGM=DSNUTILB,PARM='''subsys',MODI'Jobname||ctst0''''
sk.6='//SYSPRINT DD  SYSOUT=*                                       '
sk.7='//SYSIN    DD  *                                              '
          sk.0=7; jobw = fiimg
          "alloc da('"outdsimg"') f("jobw") mod reuse" ; call WriteRec
   /*---  Write sysin Copy statments           ---*/
          skmod.0=#vet0
          jobw = fiimg
          "alloc da('"outdsimg"') f("jobw") mod reuse" ; call Writemod
          ssn = substr(subsys,4,1) ; idgior = date(j)
          outmem =  S || ssn || numsav || idgior
          if JobNum = 1 | #d = sysrec00.0 then do
sk.1='//LAB0END  ENDIF                                              '
```

```
sk.2='//* ------------------------------------------------------ *    '
sk.3='//* ---       Save output on DataSet                     --- *    '
sk.4='//* ------------------------------------------------------ *    '
sk.5='//REXX00   EXEC   DB2REXX1,COND=EVEN                            '
sk.6='//REXX00.SYSTSIN  DD  *                                        '
sk.7='  ISPSTART CMD(@DB2OUBK +                                       '
sk.8=Subsys','Jobname||ctst0','$report','outmem',WORK)               '
sk.9='//* ------------------------------------------------------ *    '
            sk.0=9
            if Nextjob ¬= NO then do
sk.10='//* ------          Start next job                  ------ *   '
sk.11='//* ------------------------------------------------------ *   '
sk.12='//NEXTSTRT EXEC PGM=IEBGENER,COND=EVEN                        '
sk.13='//SYSUT1   DD  DISP=SHR,DSN='$jcllib'('nextjob')              '
sk.14='//SYSUT2   DD  SYSOUT=(,INTRDR)                               '
sk.15='//SYSIN    DD  DUMMY                                          '
sk.16='//SYSPRINT DD  SYSOUT=X                                       '
               sk.0=16 ; end ; end
          if Jobnum > 1 & #d < sysrec00.0 then do
sk.1='//LAB0END  ENDIF                                               '
sk.2='//* ------------------------------------------------------ *    '
sk.3='//* ---          Save output on DataSet                  --- *    '
sk.4='//* ------------------------------------------------------ *    '
sk.5='//REXX00   EXEC   DB2REXX1,COND=EVEN                            '
sk.6='//REXX00.SYSTSIN  DD  *                                        '
sk.7='  ISPSTART CMD(@DB2OUBK +                                       '
sk.8=Subsys','Jobname||ctst0','$report','outmem',WORK)               '
sk.9='//* ------------------------------------------------------ *    '
sk.10='//RELEASE EXEC PGM=IEBGENER                                   '
sk.11='//SYSPRINT  DD  SYSOUT=*                                      '
sk.12='//SYSUT2    DD  SYSOUT=(,INTRDR)                              '
sk.13='//SYSUT1    DD  *,DLM=EE                                      '
sk.14='/*$A '''Jobname||NexJid'''                                    '
sk.15='EE                                                            '
sk.16='//SYSPRINT  DD  SYSOUT=*                                      '
sk.17='//SYSIN     DD  DUMMY                                         '
sk.18='//* ------------------------------------------------------ *   '
            sk.0=18 ; end
          jobw = fiimg ; "alloc da('"outdsimg"') f("jobw") mod reuse"
          call WriteRec ;testata = yes ;#vet0 = 0 ;ctr0 = 0 ;stp = 0
;labe = 0
          end ; call VerElab ; end  /* DO #d End */
    return
   /*---   Dataset allocation routine          ---*/
 Alloc00 :
   /*---  ImageCopy job DataSet allocation     ---*/
    say  '' ; outdsimg = $hiwork'.'subsys'.@DB2COP0.JOBIMAG'
    prmalloc = subsys' 'outdsimg' 0 45,15 f,b 80 27920 fiimg yes'
    call @db2all0 prmalloc ; if word(result,1) = 99 then exit ; return
   /*---  Routine Display % elab.              ---*/
```

```
 VerElab :
    select
       when #d = welab25 then
          say '  25%  --$---------  Record processed n>
'right(welab25,4,'Ø')
       when #d = welab5Ø then
          say '  5Ø%  -----$------  Record processed n>
'right(welab5Ø,4,'Ø')
       when #d = welab75 then
          say '  75%  --------$---  Record processed n>
'right(welab75,4,'Ø')
       when #d = welabØØ then do
          say ' 1ØØ%  -----------$  Record processed n>
'right(welabØØ,4,'Ø')
          say ''
          end
       otherwise
          nop ; end ; return
   /*---  ImageCopy record writer              ---*/
 WriteRec :
    "EXECIO * DISKW "jobw" (STEM sk. FINIS"
 ClearRec:
    DO #f = 1 to sk.Ø ; sk.#f = blk ; end ; return
   /*---  INPCopy record writer                ---*/
 WriteCop :
    "EXECIO * DISKW "jobw" (STEM skcop. FINIS"
 ClearCop:
    DO #f = 1 to skcop.Ø ; skcop.#f = blk ; end ; return
   /*---  INPMody record writer                ---*/
 WriteMod :
    "EXECIO * DISKW "jobw" (STEM skmod. FINIS"
 ClearMod:
    DO #f = 1 to skmod.Ø ; skmod.#f = blk ; end ; return
   /*---  Free work DataSet                    ---*/
 FreeØ :
    xx=outtrap(trpdummy.)
       "free fi(sysprint)" ; "free fi(sysrecØØ)" ; "free fi(systsinp)"
       "free fi(sysin)" ; "free fi(syspunch)" ; "free fi(fiimg)"
    xx=outtrap(off) ; return
   /*---  Free & delete work DataSet           ---*/
 Free  :
    xx=outtrap(trpdummy.)
       address tso "delete '"outdsprt"'"  ; "free fi(sysprint)"
       address tso "delete '"outdsrec"'"  ; "free fi(sysrecØØ)"
       address tso "delete '"outdsrec1"'" ; address tso "delete
'"outdstsin"'"
       "free fi(systsinp)" ; "free fi(sysin)" ;address tso "delete
'"outdsin"'"
       "free fi(syspunch)" ; address tso "delete '"outdsimg"'"
       "free fi(fiimg)" ; address tso "delete '"outds1"'"
```

```
       address tso "delete '"outds2"'"
    xx=outtrap(off) ; return
```

$DB2COP1 REXX EXEC

```
/* REXX */
   /**********  DB2 Command Utility Tools  *********/
arg parmin ; parm  = translate(parmin,' ',',')
nparm
=words(parm);subsys=word(parm,1);ViewDB=word(parm,2);TipCom=word(parm,3)
   /*---  Test input parameters                ---*/
 if nparm < 3 then do ; say '' ; say '' ; say '>>>>>>>>'
    say '>>>>>>>>  Parameter string is incomplete !!!! ' parmin
    say '>>>>>>>>' ; say '' ; say '' ; $exitc = 99 ; return $exitc ; end
 if TipCom ¬= DISP & TipCom ¬= STAR then do
    say '' ; say '' ; say '>>>>>>>>'
    say '>>>>>>>>  Wrong TipCom 'TipCom' variable !!!!!             '
    say '>>>>>>>>  Specify :                                        '
    say '>>>>>>>>             - DISP(for of display Utility Stopped)'
    say '>>>>>>>>             - STAR(for display DataBase Restrct)  '
    say '>>>>>>>>' ; say '' ; say '' ; $exitc = 99 ; return $exitc ; end
   /*---  Parameters assignment                ---*/
 call @db2parØ 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)
 $unitta  =word(result,1Ø);$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,2Ø);$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  =    ; ExitWhile = no ; UtilStop = no ; DispRestr = no ; TspaRestr
= no
 call free
 if TipCom = STAR then do
    call ElabSTAR ; call AllocCOM ; call WriteSta ; end
 else do ; call AllocCOM ; call WriteDIS ; end
   /*---  SYSTSPRT file allocation            ---*/
 outds2= $hiwork'.'subsys'.@DB2COP1.SYSTSPRT'
 dsn = sysdsn(''''outds2'''')
 if dsn ¬= OK then do
    prmalloc = subsys' 'outds2' Ø 45,15 f,b 8Ø 2792Ø systspri yes'
    call @db2allØ prmalloc
    if word(result,1) = 99 then do ; $exitc = 99 ; return $exitc ; end ;
end
 else "alloc da('"outds2"') f("systspri") shr reuse"
 if TipCom = STAR then tipcom = DISREST
 else tipcom = DISUTIL ; Call Rundb2
 do while ExitWhile = no
    do #a = 1 to trpØ3.Ø ; analisi = substr(trpØ3.#a,1,8)
       select
       when analisi = 'DSNE1ØØI' then do
   /*---  DB2 subsystem not active            ---*/
          subs = center(subsys,1Ø) ; say '>>>>>>' ; say '>>>>>>'
          say '>>>>>> DB2 subsystem --->'subs'<--- is not active'
          say '>>>>>>' ; say '>>>>>>'
          say '' ; say '' ; call free ; $exitc = 99 ; return $exitc ;
end
       when analisi = 'DSNT5ØØI' then do
   /*---  Unavailable DB2 resource            ---*/
          say '>>>>>>' ; say '>>>>>>'
          say '>>>>>>  Display KO. DB2 Unavailable resource !!!!!  '
          say '>>>>>>  Verify problem.                              '
          say '>>>>>>' ; say '' ; say ''
          do #a = 1 to trpØ3.Ø ; say trpØ3.#a ; end
          call free ; $exitc = 99 ; return $exitc ; end
       when analisi = 'DSNU112I' then do
   /*---  No Utility found                    ---*/
          ExitWhile = yes ; UtilStop  = no
          say '>>>>>>'
          say '>>>>>>  No DB2 Utility found for UTILID = *   '
          say '>>>>>>  Display OK                            '
          say '>>>>>>' ; say '' ; say '' ; call free ; iterate ; end
       when analisi = 'DSNU1Ø5I' then do
   /*---  Utility in progress .....           ---*/
          iterate ; end
       when analisi = 'DSN9Ø22I' then do
   /*---  End Normal complete                 ---*/
          leave ; end
       when analisi = 'DSNU1ØØI' then do
```

```
   /*---  Display ok Utility Stopped           ---*/
          UtilStop = yes
          do #b = 1 to 7 ; sk.#b = trp03.#a ; #a = #a + 1 ; end
          #a = #a - 1
          "alloc da('"outds2"') f("systspri") mod reuse"
          sk.0= 7 ; jobw = systspri ; Call WriteRec
          if #a > trp03.0 then leave
          iterate ; end
   /*---  Display DataBase Restrict            ---*/
       when analisi = 'DSNT360I' then do
          ExitWhile = yes ; DispRestr = yes
          do #b = 1 to trp03.0 ; sk.#b = trp03.#b ; end
          "alloc da('"outds2"') f("systspri") mod reuse"
          sk.0= trp03.0
          jobw = systspri ; Call WriteRec ; #a = trp03.0 ; end
       otherwise
          say ''; say '>>>>>> Unpredictable error. End elaboration
'
          say ''
          do #a = 1 to trp03.0 ; say trp03.#a ; end
          $exitc = 99 ; return $exitc ; end ; end
   /*---  Process Disp Util                    ---*/
    if UtilStop = yes then do ; Call ReadSyspr
       Do #b = 1 to systspri.0 ; say substr(systspri.#b,1,78) ; end
       xx=OUTTRAP(trp07.) ; "ispexec edit dataset('"outds2"')
macro(@mdb2042)"
       xx=OUTTRAP(OFF)
       if rc > 0 then do ; do #a = 1 to trp07.0 ; say trp07.#a ; end
          $exitc = 99 ; return $exitc ; end
sk.1='DSN SYSTEM('subsys')                                    '
       Call ReadSyspr ; #x = 2
       Do #c = 1 to systspri.0
sk.#x='-TER UTIL('word(systspri.#c,1)')                        '
          #x = #x + 1 ; end
sk.#x='END                                                     '
       sk.0=#x ; jobw = systsinp ;Call WriteRec ;tipcom = TERUTIL ; Call
Rundb2
       call WriteDis ; tipcom = DISUTIL ; Call Rundb2 ; end ; end
   /*---  Process Disp DB Restrict             ---*/
    if DispRestr = yes then do ; Call ReadSyspr
       xx=OUTTRAP(trp07.) ; "ispexec edit dataset('"outds2"')
macro(@mdb2043)"
       xx=OUTTRAP(OFF)
       if rc > 0 then do ; do #a = 1 to trp07.0 ; say trp07.#a ; end
          $exitc = 99 ; return $exitc ; end ; Call ReadSyspr
sk.1='/* REXX */                                               '
sk.2='trace ?o                                                 '
sk.3='isredit macro                                            '
       #x = 3
       Do #b = 1 to systspri.0
```

```
          $DSNT362I = word(systspri.#b,1)
          if $DSNT362I = DSNT362I then do
             $ddbname = word(systspri.#b,5) ; wddbname = $ddbname
             #b = #b + 1 ; $dtsname = word(systspri.#b,1)
             if $dtsname ¬= '********' then do
                TspaRestr = yes ; #x = #x + 1
                say systspri.#b
           say 'The tablespace '$ddbname'.'$dtsname' not sar{ saved !!!'
sk.#x='isredit exclude ''''''$ddbname'  '$dtsname'''''' all '
                end ; end
          else do
             $dtsname = word(systspri.#b,1)
             #x = #x + 1 ; say systspri.#b
           say 'The tablespace 'wddbname'.'$dtsname' not sar{ saved !!!'
sk.#x='isredit exclude ''''''$ddbname'  '$dtsname'''''' all '
             end ; end
           if TspaRestr = yes then do ; #x = #x + 1
sk.#x='isredit delete x all         '
          #x = #x + 1
sk.#x='isredit save                 '
             #x = #x + 1
sk.#x='isredit end                  '
          sk.Ø=#x ; jobw = sysrecØØ
        "alloc da('"outdsrec"') f("jobw") shr reuse" ; Call WriteRec
        "ispexec lminit dataid("inpØ")  dataset('"outdsrec"') enq(shr)"
        "ispexec lminit dataid("out1")  dataset('"$librexx"') enq(shr)"
        "ispexec lmcopy fromid("inpØ") todataid("out1"),tomem(@MDBTMPØ)
           replace"
          outdsrec = $hiwork'.'subsys'.@DB2COPØ.SYSRECØØ'
          xx=OUTTRAP(trpØ7.)
             "ispexec edit dataset('"outdsrec"') macro(@mdbTMPØ)"
          xx=OUTTRAP(OFF)
          if rc > Ø then do ; do #a = 1 to trpØ7.Ø ; say trpØ7.#a ; end
             $exitc = 99 ; return $exitc ; end ; end
       else do
          say '>>>>>>' ;
                say '>>>>>>  No Tablespace found in Restricted State '
          say '>>>>>>' ; say '' ;
                                say '' ; end  ; end ; call Free ; exit
   /*---  DB2 command routine                    ---*/
 Rundb2 :
    wrk = center(tipcom,8)
    say '  ____________________________________________________________ '
    say '*                                          'time()'     *'
    say '*       Command  'wrk  'is running                          *'
    say '*                                                           *'
    say '*                        Please Wait  .........             *'
    say '  ____________________________________________________________ '
    say ''
```

```
    xx=outtrap(trpØ3.) ; address tso "ex '"outds1"'" ; xx=outtrap(off)
    return
   /*---  Write routine                          ---*/
 WriteRec :
    "EXECIO * DISKW "jobw" (STEM sk. FINIS"
 ClearRec:
    DO #f = 1 to sk.Ø ; sk.#f = blk ; end ; return
   /*---  Write SYSTSINP                         ---*/
 WriteDis :
sk.1='DSN SYSTEM('subsys')                                  '
sk.2='-DIS UTIL(*)                                          '
sk.3='END                                                   '
    sk.Ø=3; jobw = systsinp ; Call WriteRec ; return
   /*---  Read SYSTSPRI file                     ---*/
 ReadSyspr:
    xx=outtrap(trpØ6.) ; "execio * diskr systspri (stem systspri. finis"
    xx=outtrap(off)
    if rc > Ø then do ; do #a = 1 to trpØ6.Ø ; say trpØ6.#a ; end
       say '' ; say '' ; say '>>>>>>>>'
       say '>>>>>>>>   Error reading file "'outds2'"    '
       say '>>>>>>>>   RC='rc'. Verify.                 '
       say '>>>>>>>>' ; say '' ; say '' ; $exitc = 99 ; return $exitc ;
end
    return
   /*---  DataSet Command allocation             ---*/
 AllocCOM :
   /*---  SYSIN file allocation                  ---*/
    "alloc dummy f(sysin)"
   /*---  SYSTSINP file allocation               ---*/
    outds1= $hiwork'.'subsys'.@DB2COP1.SYSTSINP'
    dsn = sysdsn(''''outds1'''')
    if dsn ¬= OK then do
       prmalloc = subsys' 'outds1' Ø 5,1 f,b 8Ø 2792Ø systsinp yes'
       call @db2allØ prmalloc
       if word(result,1) = 99 then do ; $exitc = 99 ; return $exitc ;
end ; end
    else  "alloc da('"outds1"') f("systsinp") shr reuse" ; return
   /*---  Diaplay TS Restrict                    ---*/
 ElabSTAR :
   /*---  File bridge name                       ---*/
    outdstsin = $hiwork'.'subsys'.'$user'.SYSTSIN'
    outdsprt  = $hiwork'.'subsys'.'$user'.SYSPRINT'
    outdsrec  = $hiwork'.'subsys'.@DB2COP1.SYSRECØØ'
   /*---  Allocation file SYSIN                  ---*/
    outdsin= $hiwork'.'subsys'.'$user'.SYSIN'
    dsn = sysdsn(''''outdsin'''')
    if dsn ¬= OK then do
       prmalloc = subsys' 'outdsin' Ø 5,1 f,b 8Ø 2792Ø sysin yes'
       call @db2allØ prmalloc ; if word(result,1) = 99 then exit ; end
```

```
    else "alloc da('"outdsin"') f("sysin") shr reuse"
sk.1='SELECT DISTINCT DBNAME,''
'''
sk.2='       FROM 'ViewDB';
'
    sk.Ø=2 ; jobw = sysin ; call WriteRec ; call @db2accØ
subsys','@DB2COP1
    if word(result,1) = 99 then do ; $exitc = 99 ; return $exitc ; end
    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 = ØØ then do
       jobw = sysrecØØ ; "alloc da('"outdsrec"') f("jobw") shr reuse"
       xx=outtrap(trpreadØ1.) ; "execio * diskr sysrecØØ (stem sysrecØØ.
finis"
       xx=outtrap(off)
       if rc > Ø then do ; do #a = 1 to trpreadØ1.Ø ; say trpreadØ1.#a ;
end
          say '' ; say '' ; say '>>>>>>>>'
          say '>>>>>>>>  Error reading file "'outdsrec'"    '
          say '>>>>>>>>  RC='rc'. Verify.                   '
          say '>>>>>>>>' ; say '' ; say '' ; $exitc = 99 ; return
$exitc; end
       end
    else do ; say '' ; say '' ; say '>>>>>>>>' ; say '>>>>>>>>'
    say '>>>>>>>>  Unpredictable RCdb2 'RCdb2' value.End elaboration   '
       say '>>>>>>>>' ; say '>>>>>>>>'
       say '' ; call Free ; $exitc = 99 ; return $exitc ; end  ; return
   /*---  Write SYSTSINP START RESTRICT        ---*/
 WriteSTA :
    #y = 2
sk.1='DSN SYSTEM('subsys')                                  '
sk.2='-DIS DB (            -                                '
    DO #d = 1 to sysrecØØ.Ø
       #y = #y + 1 ; $DBname = strip(substr(sysrecØØ.#d,1,8))
       lfillØ = length($DBname) ; wrk = 8 - lfillØ
       if #d = sysrecØØ.Ø then
sk.#y='         '$DBname ''copies(' ',wrk)'   -'
       else
sk.#y='         '$DBname ''copies(' ',wrk)',  -'
       end ; #y = #y + 1
sk.#y='                      -                              '
       #y = #y + 1
sk.#y='        ) SPACENAM(*) RESTRICT LIMIT(999)            '
       #y = #y + 1
sk.#y='END                                                  '
    sk.Ø=#y ; jobw = systsinp ; Call WriteRec ; return
   /*---  Free work dataset                    ---*/
```

```
 Free :
    xx=outtrap(trpdummy.)
       "free fi(sysin)"  ; "free fi(systsinp)" ; "free fi(systspri)"
    xx=outtrap(off) ; return
```

$DB2PAR0 REXX EXEC

```
/* REXX */
   /**********  Global Parameter member   **********/
   /*--- DB2 version & subsystem id assignment ---*/
 RunEnv = 'BATCH' ; arg subsys runenv
 select
    when subsys = DSNT then do ; db2ver  = 51 ; $ctsubs = "*"     ; end
    when subsys = DSNZ then do ; db2ver  = 51 ; $ctsubs = "+"     ; end
    when subsys = DSNY then do ; db2ver  = 51 ; $ctsubs = "-DSNY" ; end
    otherwise
    if RunEnv ¬= 'ONLINE' then do ; say '>>>>>>>>'
  say '>>>>>>>>    !!!!!! A T T E N T I O N    P L E A S E !!!!!!      '
  say '>>>>>>>>'
  say '>>>>>>>>              Wrong DB2 subsystem                        '
  say '>>>>>>>>              >>>>> 'subsys' <<<<<                       '
  say '>>>>>>>>' ; end ; $exitc = 99 ; return $exitc ; exit ; end
   /*---       job Class & LPAR assignment     ---*/
 call valsysid
    select
       when result = SYSS then $class = P
       when result = SYST then $class = T
       when result = SYSZ then $class = S
       otherwise
   say '>>>>>>>>'
   say '>>>>>>>>    !!!!!! A T T E N T I O N    P L E A S E !!!!!!!  '
   say '>>>>>>>>'
   say '>>>>>>>>                    Wrong MVS LPAR                    '
   say '>>>>>>>>          Define MVS LPAR 'result' in $DB2PARØ        '
   say '>>>>>>>>' ; $exitc = 99 ; return $exitc ; exit ; end
 address ispexec 'vget (zacctnum)  shared'
   /*---   General variables                    ---*/
 $lpar    = result                        /*  LPAR                */
 $accn    = zacctnum                      /*  Account name        */
 $class   = $class                        /*  Class               */
 $msgcla  = 'X'                           /*  Message class       */
 $region  = '4M'                          /*  Region              */
 $msglvl  = '1,1'                         /*  Message level       */
 $notif   = userid()                      /*  Notify              */
 $user    = userid()                      /*  User                */
 $unitda  = '339Ø'                        /*  Type of unit dasd   */
 $unitta  = 'ROBOT'                       /*  Type of unit tape   */
 $esunit  = 'WORKA'                       /*  Esoteric work name  */
```

```
$prt     = 'NØZ5'                          /*  Printer name         */
$hiwork  = 'DB2WORK'                       /*  Hi-level work areas  */
$db2ver  = db2ver                          /*  Version of DB2       */
$ctsubs  = $ctsubs                         /*  Carattere subsystem  */
  /*---       DB2 V5.1 variables "OS/39Ø" LE  ---*/
select
   when $db2ver = '51' then do
      $librexx = 'user.library'            /*  REXX   library       */
      $parmlib = 'DSN51Ø.DSNPARM'          /*  Parmlib   "          */
      $proclib = 'user.library'            /*  Proclib   "          */
      $jcllib  = 'user.library'            /*  JCL       "          */
      $report  = 'user.report.library'     /*  Report    "          */
      $libexec = 'DSN51Ø.DSNEXEC'          /*  Sysexec Library      */
      $isptenu = 'ISP.SISPTENU'            /*  ISPF   library       */
      $isppenu = 'ISP.SISPPENU'            /*   "        "          */
      $ispmenu = 'ISP.SISPMENU'            /*   "        "          */
      $ispslib = 'ISP.SISPSLIB'            /*   "        "          */
      $plilink = 'CEE.SCEELKED'            /*  PLI       "          */
      $sibmlnk = 'CEE.SCEERUN'             /*   "        "          */
      $sortlib = 'SYS1.SYNCSORT.LVL36F.LINKLIB' /* Sort  "          */
      $runlib  = 'DSN51Ø.RUNLIB.LOAD'      /*  DB2 Runlib           */
      $dsnload = 'SYS1.DSN51Ø.SDSNLOAD'    /*  DB2 system library   */
      $tep2pgm = 'DSNTEP2'                 /*  DSNTEP2  (program)   */
      $tep2pln = 'DSNTEP51'                /*     "     (plan)      */
      $unlopgm = 'DSNTIAØ1'                /*  DSNTIAUL User.(pgm.) */
      $unlopln = 'DSNTIBØ1'                /*     "     (plan)      */
      $dunlopg = 'DSNTIAUL'                /*  DSNTIAUL IBM  (pgm.) */
      $dunlopl = 'DSNTIB51'                /*     "     (plan)      */
      $dsnproc = 'DSNUPROD'                /*  Procname DB2         */
      end
  /*---       DB2 V4.1 variables "OS/39Ø" LE  ---*/
   when $db2ver = '41' then do
      $librexx = 'DSN41Ø.DSNREXX'          /*  REXX   library       */
      $parmlib = 'DSN41Ø.DSNPARM'          /*  Parmlib   "          */
      $proclib = 'DSN41Ø.PROCLIB'          /*  Proclib   "          */
      $jcllib  = 'DSN41Ø.JCLLIB'           /*  JCL       "          */
      $report  = 'DSN41Ø.REPORT.LOGDB2'    /*  Report    "          */
      $libexec = 'DSN41Ø.DSNEXEC'          /*  Sysexec Library      */
      $isptenu = 'ISP.SISPTENU'            /*  ISPF   library       */
      $isppenu = 'ISP.SISPPENU'            /*   "        "          */
      $ispmenu = 'ISP.SISPMENU'            /*   "        "          */
      $ispslib = 'ISP.SISPSLIB'            /*   "        "          */
      $plilink = 'CEE.SCEELKED'            /*  PLI       "          */
      $sibmlnk = 'CEE.SCEERUN'             /*   "        "          */
      $sortlib = 'SYS1.SYNCSORT.LVL36F.LINKLIB' /* Sort  "          */
      $runlib  = 'DSN41Ø.RUNLIB.LOAD'      /*  DB2 Runlib           */
      $dsnload = 'SYS1.DSN41Ø.SDSNLOAD'    /*  DB2 system library   */
      $tep2pgm = 'DSNTEP2'                 /*  DSNTEP2  (program)   */
      $tep2pln = 'DSNTEP41'                /*     "     (plan)      */
```

```
      $unlopgm = 'DSNTIAØ1'              /*  DSNTIAUL User.(pgm.) */
      $unlopln = 'DSNTIBØ1'              /*     "     (plan)      */
      $dunlopg = 'DSNTIAUL'              /*  DSNTIAUL IBM  (pgm.) */
      $dunlopl = 'DSNTIB41'              /*     "     (plan)      */
      $dsnproc = 'DSNUPROD'              /*  Procname DB2         */
      end
   otherwise
      say '>>>>>>>>';
          say '>>>>>>>> Wrong DB2 version. Stop elaboration  !!!!! '
      say '>>>>>>>>' ; $exitc = 99 ; return $exitc ; exit ; end
 return $lpar    $accn    $class   $msgcla  $region $msglvl $notif ,
        $user                                                    ,
        $unitda  $unitta  $esunit  $prt     $hiwork              ,
        $db2ver  $ctsubs                                         ,
        $librexx $parmlib $proclib $jcllib  $report $libexec     ,
        $isptenu $isppenu $ispmenu $ispslib                      ,
        $plilink $sibmlnk $sortlib                               ,
        $runlib  $dsnload                                        ,
        $tep2pgm $tep2pln $unlopgm $unlopln $dunlopg $dunlopl    ,
        $dsnproc
 valsysid:
   numeric digits 1Ø ; cvt = addc(16,'ØØ') ; system = addc(cvt,Ø154)
   return storage(d2x(ad1+x2d(ad2)),4)
   addc:arg ad1,ad2
   return   c2d(storage(d2x(ad1+x2d(ad2)),4))
```

$DB2ALL0 REXX EXEC

```
/* REXX */
   /***********  File Allocation Routine    ***********/
arg parm ; nparm = words(parm) ; subsys = word(parm,1) ; vdsfile =
word(parm,2)
vdir     = word(parm,3); vspace   = word(parm,4) ; vrecfm  =
word(parm,5)
vlrecl   = word(parm,6); vblksize = word(parm,7) ; vddfile =
word(parm,8)
vokdispl = word(parm,9)
   /*---  Test input parameter                ---*/
 if nparm < 9 then do
    say '>>>>>>>>' ;
    say '>>>>>>>>  Parameter list is incomplete !!!!!' parm
    say '>>>>>>>>' ; exit ; end
   /*---  Parameters assignment               ---*/
 call @db2parØ subsys ; 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)
 $unitta  =word(result,1Ø);$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,2Ø);$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)
 $dsnproc =word(result,37)
    /*---  File allocation                      ---*/
 xx=outtrap(trpalloc0.) ; address tso "delete '"vdsfile"'"
    "alloc da('"vdsfile"') dir("vdir") space("vspace") dsorg(ps)" ,
    "recfm("vrecfm") lrecl("vlrecl") blksize("vblksize") tracks ",
    "unit("$esunit") new catalog f("vddfile") " ; xx=outtrap(off)
 if rc > Ø then do ;
    do #a = 1 to trpalloc0.Ø ; say trpalloc0.#a ; end
    say '>>>>>>>>' ; say '>>>>>>>>  Error reading file "'vdsfile'"    '
    say '>>>>>>>>  RC='rc'. Verify.                    '
    say '>>>>>>>>' ; $exitc = 99 ; return $exitc ; exit ; end
 else do
    if vokdispl = 'YES' then do ; vdsfilec = left(vdsfile,46)
       say '>>>>>>>>  'vdsfilec'  Allocation OK  ' ; end ; end ; return
```

$DB2ACC0 REXX EXEC

```
/* REXX */
   /***********  DB2 Batch 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,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     =   ; call free
   /*---  SYSTSIN file allocation              ---*/
 outdstsin= $hiwork'.'subsys'.'$user'.SYSTSIN'
 dsn = sysdsn(''''outdstsin'''')
 if dsn ¬= OK then do
    prmalloc = subsys' 'outdstsin' 0 5,1 f,b 80 2792 systsinp yes'
    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.0=4; jobw = systsinp ; call WriteRec
   /*---  SYSPRINT file allocation             ---*/
 outdsprt= $hiwork'.'subsys'.'$user'.SYSPRINT'
 dsn = sysdsn(''''outdsprt'''')
 if dsn ¬= OK then do
    prmalloc = subsys' 'outdsprt' 0 1,15 f,b,a 121 1210 sysprint yes'
    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' 0 15,45 f,b 0 0 sysrec00 yes'
    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
       subs = center(subsys,6)
     say '>>>>>>>>' ; say '>>>>>>>>'
     say '>>>>>>>>   The DB2 subsystem --->'subs'<--- is not active !!!'
     say '>>>>>>>>' ; say '>>>>>>>>'
     say '' ; say '' ; call Free ; $exitc = 99 ; return $exitc ; end
   /*---  DB2 subsystem not defined on LPAR    ---*/
    when analisi = 'DSNE110E' then do ; subs = center(subsys,6)
       say '>>>>>>>>' ; say '>>>>>>>>'
       say '>>>>>>>>  The DB2 subsystem --> 'subs' <-- is not defined
on' $lpar
       say '>>>>>>>>' ; say '>>>>>>>>'
       say '' ; say '' ; call Free ; $exitc = 99 ; return $exitc ; end
    otherwise
       say '>>>>>>>>' ; say '>>>>>>>>'
       say '>>>>>>>>  Unpredictable error !!!!!                '
       say '>>>>>>>>            Contact your Support staff      '
       say '>>>>>>>>' ; say '>>>>>>>>'
       do #a = 1 to trpdb00.0 ; say trpdb00.#a ; end
       $exitc = 99 ; return $exitc ; 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 ; say '' ; say '>>>>>>>>' ; say '>>>>>>>>'
          say '>>>>>>>>    Ø Record found. !!!!!!  '
          say '>>>>>>>>' ; say '>>>>>>>>' ; say ''
          xx=outtrap(trpdummy.)
             address tso "printoff ('"outdsin"')  class(R)"
             address tso "printoff ('"outdsprt"')  class(R)"
          xx=outtrap(off)
          $exitc = 99 ; return $exitc ; end ; end
    else do ; say '' ; say '' ; say '>>>>>>>>' ; say '>>>>>>>>'
       say ' DB2 access KO. Verify the sysout '$user' for analyse the
failure reason !!'
       say '>>>>>>>>' ; say '>>>>>>>>' ; say ''
       xx=outtrap(trpdummy.) ; address tso "printoff ('"outdsprt"')
class(R)"
       xx=outtrap(off)
       do #b = 1 to sysprint.Ø ; say sysprint.#b ; end
       $exitc = 99 ; return $exitc ; 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
```

$DB2OUBK REXX EXEC

```
/* REXX */
   /***********       Output Archive          **********/
arg parmin ; parm = translate(parmin,' ',',') ; nparm = words(parm)
subsys = word(parm,1) ; jobSave = word(parm,2) ;DSsave = word(parm,3)
MEMsave = word(parm,4) ; DBname = word(parm,5)
```

```
   /*---  test input parameter                 ---*/
 if nparm < 5 then do ; say '' ; say '' ; say '>>>>>>>>'
    say '>>>>>>>>  Parameter string is incomplete !!! ' parmin
    say '>>>>>>>>' ; say '' ;  say '' ; exit ; end
   /*---  Parameters assignment                ---*/
 call @db2parØ subsys ; 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)
 $unitta  =word(result,1Ø);$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,2Ø);$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  =
   /*---  IsfOut file allocation               ---*/
 outds1= $hiwork'.'subsys'.'DBname'.'jobSave'.@DB2OUBK.ISFOUT'
 prmalloc = subsys' 'outds1' Ø 6Ø,3Ø f,b,a 133 Ø isfout no'
 call @db2allØ prmalloc ; if word(result,1) = 99 then exit
   /*---  ISFIN file allocation                ---*/
 outds2 = $hiwork'.'subsys'.'DBname'.'jobSave'.@DB2OUBK.ISFIN'
 prmalloc = subsys' 'outds2' Ø 1,1 f,b 8Ø 2792Ø isfin no'
 call @db2allØ prmalloc ; if word(result,1) = 99 then exit
sk.1=' PRE  'jobSave
sk.2=' DA   '
sk.3=' FIND 'jobSave
sk.4=' ++S  '
sk.5=' ++ALL'
sk.Ø=5;jobw = isfin ; call WriteRec
 xx=outtrap(trpsdsfØ.)
    "ispexec select pgm(SDSF) "
```

```
    if rc > Ø then do
       say '' ; say '>>>>>>' ; say '>>>>>> Call SDSF Failed RC = 'rc
       say '>>>>>> Verify output. End elaboration'
    say '>>>>>>' ; say '' ; address tso "printoff ('"outds1"') class(X)"
       "free fi(isfin)" ; "free fi(isfout)" ; exit ; end
 xx=outtrap(off)
 xx=OUTTRAP(trpmacØ.) ; "ispexec edit dataset('"outds1"')
macro(@MDB2Ø18)"
 xx=OUTTRAP(OFF)
 if rc > Ø then do
    do #a = 1 to trpmacØ.Ø ; say trpmacØ.#a ; end ; exit ; end
 "ispexec lminit dataid("inp1")  dataset('"outds1"') enq(shr)"
 "ispexec lminit dataid("out1")  dataset('"DSsave"') enq(shr)"
 "ispexec lmcopy fromid("inp1") todataid("out1"),tomem("MEMsave")
replace"
 if rc = Ø then do ; say ''; say '' ; say '>>>>>>>>'
    say '>>>>>>>>  Output 'jobSave' successfully archived in DataSet '
    say '>>>>>>>>  'DSsave'('MEMsave')'
    say '>>>>>>>>' ; say '' ; end
 else do ; say '' ; say '' ; say '>>>>>>>>'
    say '>>>>>>>>  Error during the archive of output 'jobSave
    say '>>>>>>>>  Verify the problem !!!!!!!!!!!!'
    say '>>>>>>>>' ; say '' ; end ; call Free ;exit
   /*---  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(isfin)" ; "free fi(isfout)"
       address tso "delete '"outds1"'" ;address tso "delete '"outds2"'"
    xx=outtrap(off) ; return
```

$MDB2018 MACRO

```
/* REXX */
/*-----   Used in @DB2OUBK REXX            -----*/
isredit macro
isredit find '''TOP OF DATA'''
isredit exclude all .zfirst .zcsr
isredit exclude all '''1 SDSF OUTPUT DISPLAY''' 1
isredit exclude all '''COMMAND INPUT ===>   ''' 3
isredit find '''BOTTOM OF DATA'''
isredit exclude all .zcsr .zlast
isredit delete x all
```

```
isredit save
isredit end
```

$MDB2042 MACRO

```
/* REXX */
/*-----   Used in @DB2COP1 REXX            -----*/
isredit macro
isredit exclude all
isredit find '''UTILID ='''  17 all
isredit change '''UTILID ='''  ''' '''  all
isredit delete x all
isredit save
isredit end
```

$MDB2043 MACRO

```
/* REXX */
/*-----   Used in @DB2COP1 REXX            -----*/
isredit macro
isredit exclude all
isredit find '''DSNT362I''' all
isredit find '''NO SPACES FOUND'''  all
isredit find '''TS''' 1Ø   all
isredit delete x all
isredit exclude '''COPY'''  all
isredit exclude ''' RO '''  all
isredit delete x all
isredit save
isredit end
```

DB2REXX1 PROC

```
//DB2REXX1 PROC
//* -----       PROC for OS/39Ø V5.1              ----- *
//COPYTMP  EXEC PGM=IEBCOPY
//SYSPRINT  DD  SYSOUT=Z
//SYSUT3    DD  UNIT=VIO,SPACE=(CYL,(5,1))
//SYSUT4    DD  UNIT=VIO,SPACE=(CYL,(5,1))
//INP1      DD  DISP=SHR,DSN=ISP.SISPTENU
//OUT1      DD  DSN=&&TENU,DISP=(,PASS),SPACE=(CYL,(1,1,1Ø)),
//          UNIT=WORKA
//SYSIN     DD  DISP=SHR,DSN=user.library(IEBDDIN)
//LAB69     IF (COPYTMP.RC EQ Ø) THEN
//REXXØØ   EXEC PGM=IKJEFTØ1,DYNAMNBR=15Ø
```

```
//STEPLIB   DD  DISP=SHR,DSN=SYS1.DSN510.SDSNLOAD
//SYSTSPRT  DD  SYSOUT=X
//SYSTERM   DD  SYSOUT=*
//SYSPROC   DD  DISP=SHR,DSN=user.library
//ISPLOG    DD  DUMMY
//ISPPROF   DD  DISP=(,DELETE,DELETE),UNIT=WORKA,SPACE=(CYL,(1,1,10)),
//          DCB=(LRECL=80,BLKSIZE=3120,RECFM=FB,DSORG=PO),DSN=&&PROF
//ISPPLIB   DD  DISP=SHR,DSN=ISP.SISPPENU
//ISPMLIB   DD  DISP=SHR,DSN=ISP.SISPMENU
//ISPSLIB   DD  DISP=SHR,DSN=ISP.SISPSLIB
//ISPTLIB   DD  DISP=(OLD,DELETE),DSN=&&TENU
//SYSTSIN   DD  DUMMY
//LAB69END  ENDIF
```

IEBDDIN DD for SYSIN

```
COPY OUTDD=OUT1,INDD=INP1
```

$DB2COP0 SAMPLE JOB

```
//&SYSUID.$ JOB (00000000),'DB2-GESTIONE',CLASS=S,MSGCLASS=X,
//       USER=ZZDBA00,REGION=3M,MSGLEVEL=(1,1),NOTIFY=&SYSUID
/*JOBPARM BYTES=999999,LINES=999999
//* ------------------------------------------------------------ *
//DB2PROC  JCLLIB ORDER=(user.library)
//REXX00   EXEC   DB2REXX1
//REXX00.SYSTSIN  DD  *
  ISPSTART CMD(@DB2COP0 dsnz,g00,14,2,no,MYJOB00)
                                        /* Parameter for Daily COPY */
  ISPSTART CMD(@DB2COP0 dsnz,s00,60,6,yes,*)
                                       /* Parameter for Weekly COPY */
                           |   |   | |  |  |
Subsystem name ------------+   |   | |  |  |
                               |   | |  |  |
S00 for Weekly G00 for Daily -+   | |  |  |
                                  | |  |  |
Number of days for DELETE AGE ----+ |  |  |
                                    |  |  |
Number of job to be generated ------+  |  |
                                       |  |
Submit job with Hold parameter --------+  |
                                          |
Submit your job at the end of ImageCopy --+
```

Giuseppe Rendano
DB2 Systems Programmer (Italy)

Contributing to *DB2 Update*

In addition to *DB2 Update*, the Xephon family of *Update* publications now includes *CICS Update*, *MVS Update*, *TCP/SNA Update*, *VSAM Update*, *RACF Update*, *AIX Update*, *Domino Update*, *MQ Update*, *NT Update*, *Oracle Update*, and *TSO/ISPF Update*. Although the articles published are of a very high standard, the vast majority are not written by professional writers, and we rely heavily on our readers themselves taking the time and trouble to share their experiences with others. Many have discovered that writing an article is not the daunting task that it might appear to be at first glance.

They have found that the effort needed to pass on valuable information to others is more than offset by our generous terms and conditions and the recognition they gain from their fellow professionals. Often, just a few hundred words are sufficient to describe a problem and the steps taken to solve it.

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. A copy of our *Notes for Contributors*, which explains the terms and conditions under which we publish articles, is available from our Web site at www.xephon.com/contnote.html. Articles can be sent to the editor, Trevor Eddolls, at any of the addresses shown on page 2, or e-mailed to trevore@xephon.com.

Tivoli has announced Version 1.5 of its Business Systems Manager (TBSM) enterprise system management product for monitoring and controlling system management events across sites with multiple data centres and multiple platforms. It integrates with other Tivoli products to provide the ability to view systems management events and helps manage the OS/390 and distributed systems from any NT or Windows 2000 system.

Its system management application components integrate all OS/390 and distributed platform management disciplines into a single object model and provide a single console for managing and viewing all resources.

There's also a new ability to launch products from the TBSM GUI within context, plus updated and new third-party monitoring support for MVS, CICS, DB2, and IMS.

Requirements include OS/390 Version 2.7 or later, z/OS Version 1.1, NetView for OS/390 Version 1.2 or later, for managing CICS, DB2, IMS, or MVS environments, NT 4.0 or later, or Windows 2000.

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

* * *

Computer Associates has released CA-ACF2 6.4 and CA-Top Secret 5.2 security applications, which integrate application-level control across both mainframe and distributed platforms.

The new releases include the eTrust LDAP Server, letting users of both products use LDAP to query and update mainframe security information for use on distributed systems.

Also, users can access information stored in DB2 UDB Server, commercial and in-house LDAP directories, popular third-party mainframe security solutions, and other back-end data stores to help create a single source of security information.

In addition, users can use eTrust Admin to integrate the management of mainframe security information with distributed applications and platforms, enabling LDAP-based queries and updates for all systems.

For further information contact:
Computer Associates, 1 CA Plaza, Hauppauge, NY 11749, USA.
Tel: (516) 342 5224.
URL: http://www.ca.com.

* * *

IBM has announced WebSphere Version 4, claiming to handle twice the number of JDBC and EJB transactions as BEA for the same cost, and integrating open application development tools for integrating developers' tools portfolios.

The software connects and interoperates with 35 software platforms and applications, including SAP, PeopleSoft, CICS, IMS and host integration. It's also more tightly integrated with databases including DB2 and SQLServer, and with middleware such as MQSeries, Tivoli, and Lotus Domino.

For further information contact your local IBM representative.
URL: http://www.ibm.com/websphere