89

DB2 update

March 2000

In this issue

DB2 Update

Published by

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

North American office

Xephon/QNA
1301 West Highway 407, Suite 201-405
Lewisville, TX 75077-2150
USA
Telephone: 940 455 7050

Contributions

Articles published in *DB2 Update* are paid for at the rate of £170 ($250) per 1000 words and £90 ($140) per 100 lines of code for original material. To find out more about contributing an article, without any obligation, please contact us at any of the addresses above and we will send you a copy of our *Notes for Contributors*.

DB2 Update on-line

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

Editor

Trevor Eddolls

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.

Dynamic SQL for Fuzzy SELECT

My article *Fuzzy SELECT* (*DB2 Update*, Issue 87, January 2000) specified methods for fuzzifying the standard SELECT so users could use predicates like ‘awful’, ‘so so’, ‘better’, ‘WOW’, etc. An example for a bank could be:

```
SELECT     customers, deposits, location
FROM       customer_table
WHERE      customers = 'well to do'
AND        deposits = 'above average'
AND        location = 'around Port Dickson'
```

The data values in quotes do not exist, but are user terms based on median ranges calculated by a fuzzy algorithm on existing table data. Standard SQL would return ‘not found’. Another example:

```
SELECT     borrowers
FROM       borrower_table
WHERE      borrower = 'very overdue'
```

This SELECT uses a different column/table and only a single predicate compared to the first SELECT. Allowing users to enter fuzzy predicates on different columns/tables with variable objects requires dynamic SQL for varying-list SELECT statements.

The techniques discussed are suitable for all current versions of DB2. It requires compatible CLI, CLP, and REXX.

DYNAMIC SQL

Dynamic SQL has four options, which are:

- Execute immediate.
- Non-SELECT statements.
- Fixed-list SELECT statements.
- Varying-list SELECT statements.

Execute immediate implicitly processes and executes a subset of complete SQL statements coded in host variables. It does *not* process SELECT.

Non-SELECT explicitly processes and executes a subset of complete SQL statements using PREPARE and EXECUTE. As the name states, it does *not* process SELECT.

Fixed-list SELECT explicitly processes and executes SELECT whose columns are known and unchanging.

Varying-list SELECT explicitly processes and executes SELECT whose columns are *unknown*. It is mandatory for processing interactive SQL.

PERFORMANCE MYTHS

Dynamic SQL has been available almost from the beginning. DB2 veterans know that many shops then, as some still do, banned dynamic SQL because of perceived poor performance. Poor performance is usually caused by using dynamic SQL for the wrong function or by poor coding. An example is using dynamic SQL with a changeable predicate for a SELECT requiring multiple but unchanging predicates. Better performance will result from coding multiple static SELECT statements.

Dynamic SQL is prepared at run time and may incur additional network traffic. Note: network traffic can be avoided by using CLI deferred PREPARE. Static SQL is prepared at precompile time. IBM suggests (*Application Development Guide*, IBM DB2 Universal Database, Version 6, SC09-2845-00) using dynamic SQL for:

- Time to run SQL statement >10 seconds
- Frequent range predicates (< >, BETWEEN , LIKE)
- Random query
- Highly non-uniform data distribution
- Frequent RUNSTATS.

Fuzzy SELECT transformed to an executable SELECT will:

- Often require > 10 seconds because of the nature of the query.
- Almost always use range predicates.
- Always be random.

- Work best with highly skewed data.
- Work best with frequent RUNSTATS.

Properly designed and coded dynamic SELECT will outperform embedded static SELECT because the optimizer uses distribution statistics to choose the best access plan. Optimizer cannot use many RUNSTATS statistics on host variables because it does not know the complete SELECT statement.

A dynamic SQL trap is to use parameter markers which are equivalent to host variables.

Random query interfaces such as CLP do *not* use parameter markers thereby providing better execution time.

TRANSFORMATION TASKS

Transformation tasks include:

- Translating fuzzy input to SQL predicates
- Executing transformed SQL
- Returning result set to the user.

Reprising the second fuzzy SELECT:

```
SELECT     borrowers
FROM       borrower_table
WHERE      borrower = 'very overdue'
```

The users need to know what columns and fuzzy predicates are available to them. This can be customized to each user by utilizing the USER authorization-name and password. The users need not and should not be required to enter the SELECT keyword and the FROM/ WHERE clauses. Their input should be:

```
borrowers 'very overdue'
```

or:

```
borrowers who are 'very overdue'
```

‘Who are’ are ‘courtesy’ words allowing the users to make their query more English-like. Quotes enclose the ‘fuzzy predicate’.

CLI or REXX stored procedures can be used to associate each user with the available columns, tables, and fuzzy predicates, and to transform user input to a SELECT statement. Note: IBM has stabilized REXX in DB2 Version 5, meaning there will no future enhancements such as processing SQL object identifier names >18 bytes.

Borrowers 'very overdue' is transformed to:

```
SELECT     borrowers, payment_status
FROM       borrower_table
WHERE      payment status BETWEEN 83 AND 105
```

The stored procedure processing derives payment_status, borrower_table, and BETWEEN values for 'very overdue' of 83 to 105 days.

CLI or REXX can execute the SELECT returning the output to the user. Note: REXX requires EXECUTE IMMEDIATE or PREPARE and EXECUTE statements in the SQLEXEC routine or a SQLDB2 routine to CALL DB2 API with syntax of CALL SQLDB2 'command string'. 'Command string' must be executable by CLP.

OTHER CONSIDERATIONS

CLI can save common user input and the resultant SELECT source statement within a table to avoid the transformation process.

CLP can be used in conjunction with CLI using db2cli or REXX (see above), or in its own right to:

- EXPLAIN dynamic SQL statements producing an access plan graph using db2exp.
- Maintain common CLP requests in an imbedded shell script command file.
- Execute INVOKE STORED PROCEDURE (DART [Database Application Remote Interface]) for applications designed to run in client/server mode.
- Execute RUNSTATS; can be used dynamically to provide up-to-date statistics for columns or tables having volatile activity to yield best access plan.

CONCLUSIONS

Processing user fuzzy input, transforming it to SELECT, and coding dynamic SQL to provide fuzzy values is hard work. I hope I have shown that it is worth the effort because:

- The users get more relevant answers allowing them to make better decisions that can *dramatically* and *positively* affect the bottom line.
- It allows DBAs to be heroes since they are satisfying important user needs.
- Experience with CLI, CLP, and REXX can lead to other dynamic SQL applications that can provide many additional benefits to any organization including using advanced mathematical theories such as Cauchy-Schwartz uncertainty or chaos algorithms.

Eric Garrigue Vesely
Principal Analyst
Workbench Consulting (Malaysia)

New DB2 back-up procedure – revisited

An error crept into the code published in the December 1999 issue of *DB2 Update*. The final eight lines of code on page 7 read:

```
  tstart.k=tsart.k||tinp.i||') ACCESS(RW)'
end
k=k+1
tstart.k=' END'
tstart.Ø=k
/* Write the output files to disk                               */
   'execio * diskw startrep (stem tstart. '
   'execio * diskw copyrep (stem tcopy. '
```

That first line should have read:

```
  tstart.k=tstart.k||tinp.i||') ACCESS(RW)'
```

We apologize for any inconvenience caused to our readers.

A real-time Coupling Facility monitor – part 2

This month we conclude the code for a real-time Coupling Facility monitor.

```
STCACHED EQU   *
         USING IXLYAMDSTRC,R4
CLC   STRNAME,IXLYAMDSTRC_STRNAME
         BE    GOTSTD
GETSTDE  EQU   *
         L     R4,IXLYAMDSTRC_STRNEXT
         LTR   R4,R4                        LAST ENTRY ?
         BNZ   LOOPSTRD
GOTSTD   EQU   *
         BAL   R6,GETSTR
         CALL  ISPLINK,(DISPLAY,STPANEL),VL
         C     R15,=F'8'               HAS PF3   BEEN HIT (R15 = 8)?
         BE    REDISPP2
         B     REDISPP3
NOSELECT EQU   *
         B     DISPP1
ERROR1   EQU   *
         CALL  ISPLINK,(SETMSG,MSG1),VL
         B     RETURN
ERROR2   EQU   *
         CALL  ISPLINK,(SETMSG,MSG2),VL
         B     RETURN
*
* RETURN
* ======
*
RETURN   EQU   *
*        CALL  ISPLINK,(TBCLOSE,FSTABLE),VL  CLOSE TABLE
         L     R13,4(R13)
         L     R1,8(R13)
         FREEMAIN R,LV=WORKL,A=(R1)
         L     R14,12(R13)
         LM    RØ,R12,2Ø(R13)
         BR    R14
IXLMG    EQU   *
         AUTHON                                 AUTH SVC
         MODESET KEY=ZERO
         IXLMG DATAAREA=ANSAREA,                                       X
               DATALEN=ANSLEN,                                         X
               RETCODE=RETCODE,                                        X
               RSNCODE=RSNCODE
         LTR   R15,R15
         BNZ   ERROR2
         MODESET KEY=NZERO
         AUTHOFF                                RESET AUTH
         BR    R6
GETSTR   EQU   *
         MVC   STLENT,=CL8' "
         MVC   STLENTU,=CL8' "
         MVC   STDELM,=CL8' "
         MVC   STDELMU,=CL8' "
         MVC   STLSIZE,=CL8' "
         MVC   STLHD,=CL8' "
         MVC   STDENT,=CL8' "
         MVC   STDENTU,=CL8' "
         USING IXLYAMDSTRL,R4
         CLI   IXLYAMDSTRL_TYPE,X'21'
         BNE   STCACHE                          * CACHE STRUCTURE ?
         MVC   STRNAME,IXLYAMDSTRL_STRNAME
*                                               RESPONSE TIME
*                                               SYNC
*
         MVC   NSTIMEC,IXLYAMDSTRL_SYNCTIMECOUNT
         MVC   NSTIMES,IXLYAMDSTRL_SYNCSUMTIME
         MVC   NATIMEC,IXLYAMDSTRL_ASYNCTIMECOUNT
         MVC   NATIMES,IXLYAMDSTRL_ASYNCSUMTIME
         CLC   OSTIMEC,=F'Ø'             FIRST TIME ?
         BE    ACCESSL
         L     R3,NSTIMES+4
         L     R2,OSTIMES+4
         SR    R3,R2
         BNH   BADS
         XR    R2,R2
         L     RØ,NSTIMEC
         S     RØ,OSTIMEC
         BNH   BADS
         DR    R2,RØ
         CVD   R3,DOUBLE
         MVC   STSTIME,MASK
         ED    STSTIME(8),DOUBLE+4
BADS     EQU   *
         L     R3,NATIMES+4
         L     R2,OATIMES+4
         SR    R3,R2
         BNH   BADA
         XR    R2,R2
         L     RØ,NATIMEC
         S     RØ,OATIMEC
         BNH   BADA
         DR    R2,RØ
         CVD   R3,DOUBLE
         MVC   STATIME,MASK
         ED    STATIME(8),DOUBLE+4
BADA     EQU   *
ACCESSL  EQU   *
*                                            ACCESS RATE
```

```
         STCKCONV STCKVAL=OTIME,CONVVAL=TWORK,TIMETYPE=BIN
         L     R2,TWORK
         STCKCONV STCKVAL=NTIME,CONVVAL=TWORK,TIMETYPE=BIN
         L     RØ,TWORK
         SR    RØ,R2
         BNH   BADDELTA
         CLC   OSTIMEC,=F'Ø'             FIRST TIME ?
         BE    BADDELTA
         L     R3,NSTIMEC
         S     R3,OSTIMEC
         L     R1,=F'1ØØØ'     XXX.X REQ/SEC
         SR    R2,R2
         MR    R2,R1
         DR    R2,RØ
         CVD   R3,DOUBLE
         MVC   STSCNT,MASKRATE
         ED    STSCNT(8),DOUBLE+5
         L     R3,NATIMEC
         S     R3,OATIMEC
         L     R1,=F'1ØØØ'     XXX.X REQ/SEC
         SR    R2,R2
         MR    R2,R1
         DR    R2,RØ
         CVD   R3,DOUBLE
         MVC   STACNT,MASKRATE
         ED    STACNT(8),DOUBLE+5
BADDELTA EQU   *
         MVC   OTIME,NTIME
         MVC   OSTIMEC,NSTIMEC
         MVC   OSTIMES,NSTIMES
         MVC   OATIMEC,NATIMEC
         MVC   OATIMES,NATIMES
         CLI   IXLYAMDSTRL_TTY,IXLYAMDA_LIST    * LIST STRUCTURE ?
         BE    STLIST
         CLI   IXLYAMDSTRL_TTY,IXLYAMDA_LOCK    * LOCK STRUCTURE ?
         BE    STLOCK
STLIST   EQU   *
         MVC   STTYPE,=CL8"LIST'
*                                           DIRECTORY ENTRIES
         L     R8,IXLYAMDSTRL_MLSEC
         CVD   R8,DOUBLE
         MVC   STDENT,MASK
         ED    STDENT(8),DOUBLE+4
         L     R8,IXLYAMDSTRL_LSEC
         CVD   R8,DOUBLE
         MVC   STDENTU,MASK
         ED    STDENTU(8),DOUBLE+4
*                                           LIST ENTRIES
         L     R8,IXLYAMDSTRL_MLSELC
         CVD   R8,DOUBLE
         MVC   STDELM,MASK
         ED    STDELM(8),DOUBLE+4
         L      R8,IXLYAMDSTRL_LSELC
         CVD    R8,DOUBLE
         MVC    STDELMU,MASK
         ED     STDELMU(8),DOUBLE+4
*                                               LIST HEADERS
         L      R8,IXLYAMDSTRL_LC
         CVD    R8,DOUBLE
         MVC    STLHD,MASK
         ED     STLHD(8),DOUBLE+4
*                                               LIST ELM SIZE
*                                               256*(2**LELX)
*
         SR     R8,R8
         IC     R8,IXLYAMDSTRL_LELX
         LA     R3,1
         LTR    R8,R8
         BZ     MUT1
EXP1     EQU    *
         SLL    R3,1
         BCT    R8,EXP1
MUT1     EQU    *
         MH     R3,=H'256'
         CVD    R3,DOUBLE
         MVC    STLSIZE,MASK
         ED     STLSIZE(8),DOUBLE+4
         B      FLAGØ1
STLOCK   EQU    *
         MVC    STTYPE,=CL8"LOCK'
         L      R8,IXLYAMDSTRL_NLE              LOCK ENTRIES
         CVD    R8,DOUBLE
         MVC    STLENT,MASK
         ED     STLENT(8),DOUBLE+4
         L      R8,IXLYAMDSTRL_NLTEC
         CVD    R8,DOUBLE
         MVC    STLENTU,MASK
         ED     STLENTU(8),DOUBLE+4
*                                               LOCK SIZE
*                                                  (2**LTECH)
         SR     R8,R8
         IC     R8,IXLYAMDSTRL_LTECH
         LA     R3,1
         LTR    R8,R8
         BZ     MUT2
EXP2     EQU    *
         SLL    R3,1
         BCT    R8,EXP2
MUT2     EQU    *
         CVD    R3,DOUBLE
         MVC    STLSIZE,MASK
         ED     STLSIZE(8),DOUBLE+4
*
FLAGØ1   EQU    *
```

```
*===
*        MVC   WTO(WTOL),WTOC
*        MVC   WTO+Ø4(16),STRNAME
*        MVC   WTO+25(4),IXLYAMDSTRL_STRNEXT
*        WTO   MF=(E,WTO)
*===
*
         L     R8,IXLYAMDSTRL_SS              STRUCTURE SIZE
         MH    R8,=H'4'
         AR    R5,R8
         CVD   R8,DOUBLE
         MVC   STSIZE,MASK
         ED    STSIZE(8),DOUBLE+4
         B     GETSTEND
STCACHE  EQU   *
         USING IXLYAMDSTRC,R4
         MVC   STRNAME,IXLYAMDSTRC_STRNAME
         MVC   STTYPE,=CL8"CACHE'
         MVC   NSTIMEC,IXLYAMDSTRC_SYNCTIMECOUNT
         MVC   NSTIMES,IXLYAMDSTRC_SYNCSUMTIME
         MVC   NATIMEC,IXLYAMDSTRC_ASYNCTIMECOUNT
         MVC   NATIMES,IXLYAMDSTRC_ASYNCSUMTIME
         CLC   OSTIMEC,=F'Ø'               FIRST TIME ?
         BE    ACCESSC
         L     R3,NSTIMES+4
         L     R2,OSTIMES+4
         SR    R3,R2
         BNH   BADSC
         XR    R2,R2
         L     RØ,NSTIMEC
         S     RØ,OSTIMEC
         BNH   BADSC
         DR    R2,RØ
         CVD   R3,DOUBLE
         MVC   STSTIME,MASK
         ED    STSTIME(8),DOUBLE+4
BADSC    EQU   *
         L     R3,NATIMES+4
         L     R2,OATIMES+4
         SR    R3,R2
         BNH   BADAC
         XR    R2,R2
         L     RØ,NATIMEC
         S     RØ,OATIMEC
         BNH   BADAC
         DR    R2,RØ
         CVD   R3,DOUBLE
         MVC   STATIME,MASK
         ED    STATIME(8),DOUBLE+4
BADAC    EQU   *
ACCESSC  EQU   *
*                                            ACCESS RATE
```

```
*
         STCKCONV STCKVAL=OTIME,CONVVAL=TWORK,TIMETYPE=BIN
         L     R2,TWORK
         STCKCONV STCKVAL=NTIME,CONVVAL=TWORK,TIMETYPE=BIN
         L     RØ,TWORK
         SR    RØ,R2
         BNH   BADDELT
         CLC   OSTIMEC,=F'Ø'             FIRST TIME ?
         BE    BADDELT
         L     R3,NSTIMEC
         S     R3,OSTIMEC
         L     R1,=F'1ØØØ'     XXX.X REQ/SEC
         SR    R2,R2
         MR    R2,R1
         DR    R2,RØ
         CVD   R3,DOUBLE
         MVC   STSCNT,MASKRATE
         ED    STSCNT(8),DOUBLE+5
         L     R3,NATIMEC
         S     R3,OATIMEC
         L     R1,=F'1ØØØ'     XXX.X REQ/SEC
         SR    R2,R2
         MR    R2,R1
         DR    R2,RØ
         CVD   R3,DOUBLE
         MVC   STACNT,MASKRATE
         ED    STACNT(8),DOUBLE+5
BADDELT  EQU   *
         MVC   OTIME,NTIME
         MVC   OSTIMEC,NSTIMEC
         MVC   OSTIMES,NSTIMES
         MVC   OATIMEC,NATIMEC
         MVC   OATIMES,NATIMES
*                                            STRUCTURE SIZE
         L     R8,IXLYAMDSTRC_SS
         MH    R8,=H'4'
         AR    R5,R8
         CVD   R8,DOUBLE
         MVC   STSIZE,MASK
         ED    STSIZE(8),DOUBLE+4
*                                            DIRECTORY ENTRIES
         L     R8,IXLYAMDSTRC_TDEC
         CVD   R8,DOUBLE
         MVC   STDENT,MASK
         ED    STDENT(8),DOUBLE+4
         L     R8,IXLYAMDSTRC_TSCC
         CVD   R8,DOUBLE
         MVC   STDENTU,MASK
         ED    STDENTU(8),DOUBLE+4
*                                            DATA ELEMENT ENTRIES
         L     R8,IXLYAMDSTRC_TDAEC
         CVD   R8,DOUBLE
         MVC   STDELM,MASK
         ED    STDELM(8),DOUBLE+4
         L     R8,IXLYAMDSTRC_TCDEC
         CVD   R8,DOUBLE
         MVC   STDELMU,MASK
         ED    STDELMU(8),DOUBLE+4
*                                             DATA ELM SIZE
*                                             256*(2**LTECH)
         SR    R8,R8
         IC    R8,IXLYAMDSTRC_DAEX
         LA    R3,1
         LTR   R8,R8
         BZ    MUT3
EXP3     EQU   *
         SLL   R3,1
         BCT   R8,EXP3
MUT3     EQU   *
         MH    R3,=H'256'
         CVD   R3,DOUBLE
         MVC   STLSIZE,MASK
         ED    STLSIZE(8),DOUBLE+4
GETSTEND EQU   *
         BR    R6
*
* ISPF MESSAGES
* =============
*
MSG1     DC    CL8"IXCØØ1E'
MSG2     DC    CL8"IXCØØ2E'
* ISPF OBJECTS (PANELS, SKELETONS...)
* ===================================
*
FSPANEL  DC    CL8"IXCCF'                     ISPF PANEL NAME
FSPANELS DC    CL8"IXCCFST'                   ISPF PANEL NAME
FSSKEL   DC    CL8"IXCCFISS'                  ISPF SKELETON
FSMOUT   DC    CL8"IXCCFISO'                  FT OUTPUT MEMBER
FSTABLE  DC    CL8"FSTABLE'                   TABLE NAME
FSTABLES DC    CL8"FSTABLES'                  TABLE NAME
STPANEL  DC    CL8"IXCCFSTD'                  ISPF PANEL NAME
*
* ISPF VARIABLES
* ==============
*
FZTDSELS DC    CL8"ZTDSELS'
ZTDSELS  DS    CL4
FSELECT  DC    CL8"SELECT'
SELECT   DS    CL1
FCFNAME  DC    CL8"CFNAME'
CFNAME   DS    CL8
FCFNODE  DC    CL8"CFNODE'
CFNODE   DS    CL54
FTSPACE  DC    CL8"TSPACE'
```

```
TSPACE   DS    CL8
FFSPACE  DC    CL8"FSPACE'
FSPACE   DS    CL8
FDSPACE  DC    CL8"DSPACE'
DSPACE   DS    CL8
FCFLEVEL DC    CL8"CFLEVEL'
CFLEVEL  DS    CL8
FSTINC   DC    CL8"STINC'
STINC    DS    CL8
FVOL     DC    CL8"VOL'
VOL      DS    CL1
FSTRNAME DC    CL8"STRNAME'
STRNAME  DS    CL16
FSTSIZE  DC    CL8"STSIZE'
STSIZE   DS    CL8
FSTTOT   DC    CL8"STTOT'
STTOT    DS    CL8
FSTTYPE  DC    CL8"STTYPE'
STTYPE   DS    CL8
FSTLENT  DC    CL8"STLENT'
STLENT   DS    CL8
FSTLENTU DC    CL8"STLENTU'
STLENTU  DS    CL8
FSTDELM  DC    CL8"STDELM'
STDELM   DS    CL8
FSTDELMU DC    CL8"STDELMU'
STDELMU  DS    CL8
FSTLSIZE DC    CL8"STLSIZE'
STLSIZE  DS    CL8
FSTLHD   DC    CL8"STLHD'
STLHD    DS    CL8
FSTDENT  DC    CL8"STDENT'
STDENT   DS    CL8
FSTDENTU DC    CL8"STDENTU'
STDENTU  DS    CL8
FSTSTIME DC    CL8"STSTIME'
STSTIME  DS    CL8
FSTATIME DC    CL8"STATIME'
STATIME  DS    CL8
FSTSCNT  DC    CL8"STSCNT'
STSCNT   DS    CL8
FSTACNT  DC    CL8"STACNT'
STACNT   DS    CL8
*
*                                                 VARIABLES LIST
*
NAMELIST DC    CLØ7Ø'(CFNAME CFNODE TSPACE DSPACE DSPACE CFLEVEL STINC X
               FSPACE VOL)'
NAMELISS DC    CL2ØØ'(STRNAME STSIZE STTYPE STLENT STDELM STLSIZE STLHDX
                STDENT STDELM STDENTU STDELMU STLENTU STSCNT STACNT)'
*
*                                                 SORT PARMS
```

```
*
SORTPARM DS    ØCL12
         DC    CL1'("
SORTKEY  DS    CL8
         DC    CL1','
SORTTYPE DS    CL3
         DC    CL1')'
SORTPARS DS    ØCL12
         DC    CL1'("
SORTKEYS DS    CL8
         DC    CL1','
SORTTYPS DS    CL3
         DC    CL1')'
*
* ISPF CONSTANTS
* ==============
*
DISPLAY  DC    CL7"DISPLAY'
VDEFINE  DC    CL7"VDEFINE'
TBADD    DC    CL5"TBADD'
TBCLOSE  DC    CL7"TBCLOSE'
TBCREATE DC    CL8"TBCREATE'
TBDISPL  DC    CL7"TBDISPL'
TBSORT   DC    CL6"TBSORT'
TBTOP    DC    CL5"TBTOP'
ORDER    DC    CL5"ORDER'
NOWRITE  DC    CL7"NOWRITE'
REPLACE  DC    CL7"REPLACE'
SETMSG   DC    CL6"SETMSG'
FTOPEN   DC    CL6"FTOPEN'
FTINCL   DC    CL6"FTINCL'
FTCLOSE  DC    CL7"FTCLOSE'
SAVE     DC    CL4"SAVE'
RESTORE  DC    CL7"RESTORE'
CHARASND DC    CL3"C,A'
NUMRDSND DC    CL3"N,D'
*                                            TYPES
*                                            -----
CHAR     DC    CL4"CHAR'
*
*                                            LENGTH
*                                            ------
L1       DC    F'1'
L4       DC    F'4'
L8       DC    F'8'
L16      DC    F'16'
L54      DC    F'54'
*                                            FIELDS
* WTO TO DEBUG
*
WTOC     WTO   "                                                    X
                                                ",MF=L,ROUTCDE=(11)
```

```
WTOL     EQU   *-WTOC            LENGTH OF MACRO EXPANSION
WTO      DS    CL(WTOL)
         LTORG
*
* PROGRAM DATAAREAS
* ================
*
MASK     DC    X'4020202020202120'
MASKRATE DC    X'404020202021204B20'
NTIME    DC    D'0'
OTIME    DC    D'0'
TWORK    DC    2D'0'
NSTIMEC  DC    F'0'
NSTIMES  DC    D'0'
OSTIMEC  DC    F'0'
OSTIMES  DC    D'0'
NATIMEC  DC    F'0'
NATIMES  DC    D'0'
OATIMEC  DC    F'0'
OATIMES  DC    D'0'
ANSLEN   DC    F'400960'                    10*4096
ANSAREA  DS    10CL4096
DSECT    DSECT
SAVEAREA DS    18F                          SAVEAREA
RETCODE  DS    F
RSNCODE  DS    F
*
DOUBLE   DS    D
LGDSECT  EQU   *-DSECT
WORKL    EQU   LGDSECT                      LENGTH OF WORAREA
         IXLYAMDA
         IXLYNDE
         REGISTER
         END
```

IXCSTIS

IXCSTIS uses the IXCQUERY macro to retrieve information about structures allocated in Coupling Facilities. The IXCQUERY macro allows any authorized caller to request information about the resources the Cross-System Coupling Facility (XCF) manages. The REQINFO parameter determines whether the information is about XCF groups, systems in the sysplex, the sysplex itself, Coupling Facility resources, or information related to the automatic restart manager.

When using the REQINFO=STR_ALLDATA parameter, IXCQUERY returns information about all Coupling Facility structures.

You need to use the ANSAREA parameter to tell XCF where to return the information, and ANSLEN to tell XCF the length of the answer area. Sections in the IXCYQUAA mapping macro provide the format for the returned data.

QUAHDR maps the offset and length of the other record types.

QUACFSTR maps information about Coupling Facility structures allocated in a coupling facility.

QUASTR maps the Coupling Facility structure record.

```
IXCSTIS  CSECTIXCSTIS
         AMODE 31IXCSTIS
         RMODE ANY*
         SAVE  (14,12)
         BALR  R12,Ø
         USING *,R12
         GETMAIN R,LV=WORKL
         ST    R1,8(R13)
         ST    R13,4(R1)
         LR    R13,R1
         USING DSECT,R13
*
*        CREATE ISPF VARIABLES
*
         CALL  ISPLINK,(VDEFINE,FSELECT,SELECT,CHAR,L1),VL
         CALL  ISPLINK,(VDEFINE,FSKEY,SKEY,CHAR,L8),VL
         CALL  ISPLINK,(VDEFINE,FSTRNAME,STRNAME,CHAR,L16),VL
         CALL  ISPLINK,(VDEFINE,FALLOC,ALLOC,CHAR,L13),VL
         CALL  ISPLINK,(VDEFINE,FPENDING,PENDING,CHAR,L21),VL
         CALL  ISPLINK,(VDEFINE,FCFNAME,CFNAME,CHAR,L8),VL
         CALL  ISPLINK,(VDEFINE,FUM,UM,CHAR,L1),VL
         CALL  ISPLINK,(VDEFINE,FCFNODE,CFNODE,CHAR,L54),VL
         CALL  ISPLINK,(VDEFINE,FPLCF,PLCF,CHAR,L71),VL
         CALL  ISPLINK,(VDEFINE,FXLCF,XLCF,CHAR,L67),VL
         CALL  ISPLINK,(VDEFINE,FINITSIZ,INITSIZE,CHAR,L8),VL
         CALL  ISPLINK,(VDEFINE,FSIZE,SIZE,CHAR,L8),VL
         CALL  ISPLINK,(VDEFINE,FREBUILD,REBUILDP,CHAR,L8),VL
         CALL  ISPLINK,(VDEFINE,FSYSNAME,SYSNAME,CHAR,L8),VL
         CALL  ISPLINK,(VDEFINE,FJOBNAME,JOBNAME,CHAR,L8),VL
         CALL  ISPLINK,(VDEFINE,FCONNAME,CONNAME,CHAR,L16),VL
         CALL  ISPLINK,(VDEFINE,FCSTATUS,CSTATUS,CHAR,L16),VL
         CALL  ISPLINK,(VDEFINE,FALLOWA,ALLOWA,CHAR,L1),VL
         CALL  ISPLINK,(VDEFINE,FALLOWR,ALLOWR,CHAR,L1),VL
*
REDISPP1 EQU  *
*
*        CREATE AND SORT ISPF TABLE
         CALL ISPLINK,(TBCREATE,FSTABLE,,NAMELIST,NOWRITE,REPLACE),VL
         MVC  SORTKEY,FSTRNAME        SPECIFY DEFAULT SORT FIELD
         MVC  SORTTYPE,CHARASND       SPECIFY SORT DIRECTION
         LA    R2,ANSAREA
         USING QUAHDR,R2
         BAL   R6,IXCQUERY
         L     R3,QUAHSGOF
         LR    R4,R2
         AR    R4,R3
         USING QUASTR,R4
         USING QUASTRCF,R5
LOOPSTR  EQU   *
*                                               SET DEFAULT VALUES
         MVC   CFNAME,=CL8"N/A'
         MVC   ALLOC,=CL13"NOT ALLOCATED'
         MVC   PENDING,=CL21' "
         MVC   PLCF,=CL71' "
         MVC   XLCF,=CL67"LIST IS EMPTY'
         MVC   CFNODE,=CL54' "
         MVC   REBUILDP,=CL84"N/A'
         MVC   UM,=CL1' "
         MVC   STRNAME,QUASTRNAME
         L     R7,QUASTRINITSIZE              INIT SIZE
         MH    R7,=H'4'
         CVD   R7,DOUBLE
         MVC   INITSIZE,MASK
         ED    INITSIZE(8),DOUBLE+4
         L     R7,QUASTRSIZE                  SIZE
         MH    R7,=H'4'
         CVD   R7,DOUBLE
         MVC   SIZE,MASK
         ED    SIZE(8),DOUBLE+4
         SR    R7,R7
         IC    R7,QUASTRREBUILDPERCENT        REBUILD PERCENT
         LTR   R7,R7
         BZ    FLAGØ6
         CVD   R7,DOUBLE
         MVC   REBUILDP,MASK
         ED    REBUILDP(8),DOUBLE+4
FLAGØ6   EQU   *
         TM    QUASTRINHDW,QUASTRINHDWON      STRUCTURE ALLOCATED ?
         BNO   FLAGØ2
FLAGØ1   EQU   *
         MVC   ALLOC,=CL13"ALLOCATED'
         LR    R5,R4
         L     R6,QUASTRCFO
         AR    R5,R6
         MVC   CFNAME,QUASTRCFNAME
         LA    R7,QUASTRCFND
         USING NDE,R7
         MVC   CFNODE(6),NDETYPE
         MVI   CFNODE+6,C'.'
         MVC   CFNODE+7(3),NDEMFG
         MVI   CFNODE+1Ø,C'.'
         MVC   CFNODE+11(2),NDEPLANT
         MVI   CFNODE+13,C'.'
         MVC   CFNODE+14(12),NDESEQUENCE
         MVC   CFNODE+27(1Ø),=C'PARTITION:'
         XR    R8,R8
         IC    R8,NDEPARTITION                * PARTITION
         CVD   R8,DOUBLE
         UNPK  DOUBLE(3),DOUBLE+6(2)
         OI    DOUBLE+2,X'FØ'
         MVC   CFNODE+38(2),DOUBLE+1
         MVC   CFNODE+41(6),=C'CPCID:'
         XR    R8,R8
         IC    R8,NDECPCID                    * CPCID
         CVD   R8,DOUBLE
         UNPK  DOUBLE(3),DOUBLE+6(2)
         OI    DOUBLE+2,X'FØ'
         MVC   CFNODE+48(2),DOUBLE+1
         DROP  R7
         TM    QUASTRSTATE1,QUASTRSTDPEND       POLICY CHANGE PENDING ?
         BNO   FLAGØ2
FLAGØ3   EQU   *
         MVC   PENDING,=CL21"POLICY CHANGE PENDING'
FLAGØ2   EQU   *
*                                               PREFERENCE LIST
*
         L     R7,QUASTRPL#
         LTR   R7,R7                            ENTRIES ?
         BZ    FLAGØ4                           NO
         LR    R7,R4
         L     R6,QUASTRPLO
         AR    R7,R6
         USING QUASTRPL,R7
         LA    R11,PLCF
LOOPPL   EQU   *
         MVC   Ø(8,R11),QUASTRPLNAME
         LA    R1Ø,9
         AR    R11,R1Ø
         TM    QUASTRPLTYP,QUATYPSTRPL_LAST     LAST PL ENTRY ?
         BO    FLAGØ4
         AH    R7,QUASTRPLLEN
         B     LOOPPL
FLAGØ4   EQU   *
         CLC   ALLOC,=CL13"ALLOCATED'          STRUCTURE ALLOCATED ?
         BNE   FLAGØ7
         CLC   PLCF(8),CFNAME
         BE    FLAGØ7
         MVC   UM,=CL1'*'
*                                               EXCLUSION  LIST
FLAGØ7   EQU   *
*
         L     R7,QUASTRXL#
         LTR   R7,R7                           ENTRIES ?
         BZ    FLAGØ5                          NO
         LR    R7,R4
         L     R6,QUASTRXLO
         AR    R7,R6
         USING QUASTRXL,R7
         LA    R11,XLCF
LOOPXL   EQU   *
         MVC   Ø(16,R11),QUASTRXLNAME
*        MVC   Ø(5Ø,R11),QUASTRXLTYP
         LA    R1Ø,17
         AR    R11,R1Ø
         TM    QUASTRXLTYP,QUATYPSTRXL_LAST    LAST PL ENTRY ?
         BO    FLAGØ5
         AH    R7,QUASTRXLLEN
         B     LOOPXL
*
FLAGØ5   EQU   *
*        ADD A NEW ROW
         CALL  ISPLINK,(TBADD,FSTABLE,,ORDER),VL  ADD DATA INTO TABLE
         TM    QUASTRTYP,QUATYPSTR_LAST        LAST STRUCTURE ?
         BO    STLAST
         AH    R4,QUASTRLEN
         B     LOOPSTR
STLAST   EQU   *
         CALL  ISPLINK,(TBSORT,FSTABLE,SORTPARM),VL  SORT TABLE
DISPLAY  CALL  ISPLINK,(TBTOP,FSTABLE),VL  POINT TO TOP OF TABLE
REDISP   CALL  ISPLINK,(TBDISPL,FSTABLE,FSPANEL),VL  DISPLAY TABLE
         C     R15,=F'8'                 PF3 ?
         BE    RETURN
         CLC   SELECT,=CL1"S'
         BE    REDISPP2
         CALL  ISPLINK,(TBCLOSE,FSTABLE),VL  CLOSE TABLE
         B     REDISPP1
REDISPP2 EQU   *
         CALL ISPLINK,(TBCREATE,FSTABLES,,NAMELISS,NOWRITE,REPLACE),VL
         MVC  SORTKES,FCONNAME         SPECIFY DEFAULT SORT FIELD
         MVC  SORTTYPS,CHARASND        SPECIFY SORT DIRECTION
         LA    R2,ANSAREA
         USING QUAHDR,R2
         BAL   R6,IXCQUERY
         L     R3,QUAHSGOF
         LR    R4,R2
         AR    R4,R3
         USING QUASTR,R4
         CLC   ALLOC,=CL13"ALLOCATED'          STRUCTURE ALLOCATED ?
         BE    SSTR
DISPP3   EQU   *
         CALL  ISPLINK,(TBSORT,FSTABLES,SORTPARS),VL  SORT TABLE
         CALL  ISPLINK,(TBTOP,FSTABLES),VL  POINT TO TOP OF TABLE
         CALL  ISPLINK,(TBDISPL,FSTABLES,FSPANELS),VL  DISPLAY TABLE
         C     R15,=F'8'                 PF3 ?
         BE    REDISP
         CALL  ISPLINK,(TBCLOSE,FSTABLES),VL  CLOSE TABLE
         B     REDISPP2
SSTR     EQU   *
         CLC   STRNAME,QUASTRNAME
         BNE   SSTRN
         L     R7,QUASTRUSERO
         AR    R7,R4
         USING QUASTRUSER,R7
LCONN    EQU   *
         MVC   CONNAME,QUASTRUSERCNAME
         MVC   SYSNAME,QUASTRUSERSYS
         MVC   JOBNAME,QUASTRUSERJOB
*                                                     ALLOW REBUILD
         MVC   ALLOWR,=C'Y'
         TM    QUASTRUSERFLG1,QUASTRUSERALLOWREBLD
         BO    FLAGØ9
         MVC   ALLOWR,=C'N'
*FLAGØ9   EQU   *
*                                                     ALLOW ALTER
         MVC   ALLOWA,=C'Y'
         TM    QUASTRUSERALTERFLG,QUASTRUSERALTERALLOWED
         BO    FLAGØ8
         MVC   ALLOWA,=C'N'
FLAGØ8   EQU   *
         TM    QUASTRUSERFLG1,QUASTRUSERACT
         BO    FLAG1ØA
         TM    QUASTRUSERFLG1,QUASTRUSERFAIL
         BO    FLAG1ØB
         MVC   CSTATUS,=CL16"UNKNOWN'
         B     FLAG1Ø
FLAG1ØA  EQU   *
         MVC   CSTATUS,=CL16"ACTIVE'
         B     FLAG1Ø
FLAG1ØB  EQU   *
         MVC   CSTATUS,=CL16"FAILED PERSISTENT'
         B     FLAG1Ø
FLAG1Ø   EQU   *
         CALL  ISPLINK,(TBADD,FSTABLES,,ORDER),VL  ADD DATA INTO TABLE
         CLI   QUASTRUSERTYP,X'A4'       LAST CONNECTIONS ?
         BE    DISPP3
         SR    R4,R4
         LH    R4,QUASTRUSERLEN
         AR    R7,R4
         B     LCONN
SSTRN    EQU   *
         AH    R4,QUASTRLEN
         B     SSTR
ERROR1   EQU   *
         CALL  ISPLINK,(SETMSG,MSG1),VL
         B     RETURN
RETURN   EQU   *
```

```
         L     R13,4(R13)
         L     R1,8(R13)
         FREEMAIN R,LV=WORKL,A=(R1)
         L     R14,12(R13)
         LM    RØ,R12,2Ø(R13)
         BR    R14
IXCQUERY EQU   *
         AUTHON                                 AUTH SVC
         MODESET KEY=ZERO
         IXCQUERY REQINFO=STR_ALLDATA,                                 X
               ANSAREA=ANSAREA,                                        X
               ANSLEN=ANSLEN,                                          X
               RETCODE=RETCODE,                                        X
               RSNCODE=RSNCODE
         LTR   R15,R15
         BNZ   ERROR1
         MODESET KEY=NZERO
         AUTHOFF                                RESET AUTH
         BR    R6
ANSLEN   DC    F'4Ø96Ø'
MASK     DC    X'4Ø2Ø2Ø2Ø2Ø2Ø2Ø2Ø'
*                                             MESSAGE
*                                             -------
MSG1     DC    CL8"IXCØØ1E'
FSPANEL  DC    CL8"IXCST'           <===     ISPF PANEL NAME
FSPANELS DC    CL8"IXCSTST'         <===     ISPF PANEL NAME
*
*                                             FIELDS
FSKEY    DC    CL8"SKEY'
SKEY     DS    CL8
FSELECT  DC    CL8"SELECT'
SELECT   DS    CL1
FSTRNAME DC    CL8"STRNAME'
STRNAME  DS    CL16
FALLOC   DC    CL8"ALLOC'
ALLOC    DS    CL13
FCFNAME  DC    CL8"CFNAME'
CFNAME   DS    CL8
FPENDING DC    CL8"PENDING'
PENDING  DS    CL21
FPLCF    DC    CL8"PLCF'
PLCF     DS    CL71
FXLCF    DC    CL8"XLCF'
XLCF     DS    CL67
FINITSIZ DC    CL8"INITSIZE'
INITSIZE DS    CL8
FSIZE    DC    CL8"SIZE'
SIZE     DS    CL8
FREBUILD DC    CL8"REBUILDP'
REBUILDP DS    CL8
FCFNODE  DC    CL8"CFNODE'
CFNODE   DS    CL54
```

```
FUM      DC    CL8"UM'
UM       DS    CL1
FSYSNAME DC    CL8"SYSNAME'
SYSNAME  DS    CL8
FCONNAME DC    CL8"CONNAME'
CONNAME  DS    CL16
FJOBNAME DC    CL8"JOBNAME'
JOBNAME  DS    CL8
FCSTATUS DC    CL8"CSTATUS'
CSTATUS  DS    CL16
FALLOWA  DC    CL8"ALLOWA'
ALLOWA   DS    CL1
FALLOWR  DC    CL8"ALLOWR'
ALLOWR   DS    CL1
NAMELIST DC    CL15Ø'(STRNAME ALLOC PENDING CFNAME CFNODE PLCF XLCF    X
               INITSIZE SIZE REBUILDP UM)'
NAMELISS DC    CL15Ø'(CONNAME SYSNAME JOBNAME CSTATUS ALLOWA ALLOWR)'
FSTABLE  DC    CL8"FSTABLE'                   TABLE NAME
FSTABLES DC    CL8"FSTABLES'                  TABLE NAME
*
*                                             SORT PARMS
*
CHARASND DC    CL3"C,A'
NUMRDSND DC    CL3"N,D'
SORTPARM DS    ØCL12
         DC    CL1'("
SORTKEY  DS    CL8
         DC    CL1','
SORTTYPE DS    CL3
         DC    CL1')'
SORTPARS DS    ØCL12
         DC    CL1'("
SORTKES  DS    CL8
         DC    CL1','
SORTTYPS DS    CL3
         DC    CL1')'
*                                             ISPF FUNCTIONS
*
VDEFINE  DC    CL7"VDEFINE'
TBADD    DC    CL5"TBADD'
TBCLOSE  DC    CL7"TBCLOSE'
TBCREATE DC    CL8"TBCREATE'
TBDISPL  DC    CL7"TBDISPL'
TBSORT   DC    CL6"TBSORT'
TBTOP    DC    CL5"TBTOP'
TRKS     DC    CL8"TRKS'
ORDER    DC    CL5"ORDER'
NOWRITE  DC    CL7"NOWRITE'
REPLACE  DC    CL7"REPLACE'
SETMSG   DC    CL6"SETMSG'
```

```
*
*                                              TYPES
*                                              -----
CHAR     DC    CL4"CHAR'
*
*                                              LENGTH
*                                              ------
L1       DC    F'1'
L4       DC    F'4'
L8       DC    F'8'
L13      DC    F'13'
L16      DC    F'16'
L21      DC    F'21'
L54      DC    F'54'
L67      DC    F'67'
L71      DC    F'71'
*                                              FIELDS
* WTO TO DEBUG
WTOC     WTO   "                                                       X
                                                   ",MF=L,ROUTCDE=(11)
WTOL     EQU   *-WTOC             LENGTH OF MACRO EXPANSION
WTO      DS    CL(WTOL)
DSECT    DSECT
SAVEAREA DS    18F                                   SAVEAREA
*
RETCODE  DS    F
RSNCODE  DS    F
DOUBLE   DS    D
ANSAREA  DS    1ØCL4Ø96
LGDSECT  EQU   *-DSECT
WORKL    EQU   LGDSECT                               LENGTH OF WORAREA
         IXCYQUAA
         IXLYNDE
         REGISTER
         END
```

ISPF PANELS

You need to install the following ISPF panels in a library included in your ISPPLIB concatenation.

Panel IXC

```
%----------- Coupling Facility Monitor --------------
%OPTION  ===>_ZCMD                                                  +
%                                                +USERID   - &ZUSER
%                                                +TIME     - &ZTIME
%
%   C +Coupling Facility  - Coupling Facilities Display
%   S +Structure          - CF Structures Display
%
)INIT
)PROC
&ZQ = &Z
  IF (&ZCMD ¬= " ")
    &ZQ = TRUNC(&ZCMD,'.')
    IF (&ZQ = " ")
      .MSG = ISRUØØØ
  &ZSEL = TRANS( &ZQ
                C,'PGM(IXCCFIS)'
                S,'PGM(IXCSTIS)'
              " "," "
                X,'EXIT'
                *,'?' )
  &ZTRAIL = .TRAIL
)END
```

Panel IXCCF

```
)ATTR
   _ TYPE(INPUT)  INTENS(HIGH) CAPS(ON) COLOR(RED)
   — TYPE(OUTPUT) INTENS(LOW) JUST(RIGHT)
   ¬ TYPE(OUTPUT) INTENS(LOW) JUST(LEFT)
   ! TYPE(OUTPUT) INTENS(HIGH) JUST(LEFT)
   } TYPE(OUTPUT) INTENS(HIGH) JUST(LEFT) COLOR(RED)
   # TYPE(TEXT) COLOR(RED) INTENS(HIGH)
   ‡ TYPE(TEXT) COLOR(YELLOW) INTENS(HIGH)
   % TYPE(TEXT) COLOR(GREEN) INTENS(HIGH)
   $ TYPE(TEXT) SKIP(ON) INTENS(LOW)
)BODY EXPAND(……)
+…-…#Coupling Facilities info+…-…+
$‡COMMAND%===>_ZCMD                                 %SCROOL ===>_SAMT+
$
)MODEL CLEAR(SELECT)
+————————————————————————————————————————
_z% CFNAME:!z      %
     %Node:¬z                                                   %
     %CFLEVEL:¬z      %   %Storage Increment:—z      %k   Volatile:—z%
      Storage Usage:
       %Total:—z      % k
       %Dump :—z      % k
       %Free :—z      % k
)INIT
 .ZVARS = "(SELECT CFNAME CFNODE CFLEVEL STINC +
           vol +
           TSPACE DSPACE FSPACE)'
 &ZTDMARK = "****************************** BOTTOM OF DATA +
*******************************'
)PROC
```

```
  IF (.RESP = ENTER)
   VER (&SKEY,LIST,CFNAME)

)END
```

Panel IXCCFST

```
)ATTR
   _ TYPE(INPUT)  INTENS(HIGH) CAPS(ON) COLOR(RED)
   — TYPE(OUTPUT) INTENS(LOW) JUST(RIGHT)
   ¬ TYPE(OUTPUT) INTENS(LOW) JUST(LEFT)
   ! TYPE(OUTPUT) INTENS(HIGH) JUST(LEFT)
   } TYPE(OUTPUT) INTENS(HIGH) JUST(LEFT) COLOR(RED)
   # TYPE(TEXT) COLOR(RED) INTENS(HIGH)
   ‡ TYPE(TEXT) COLOR(YELLOW) INTENS(HIGH)
   % TYPE(TEXT) COLOR(GREEN) INTENS(HIGH)
   $ TYPE(TEXT) SKIP(ON) INTENS(LOW)
)BODY EXPAND(……)
+…-…#Coupling Facility Detail+…-…+
$‡COMMAND%===>_ZCMD                                 %SCROOL ===>_SAMT+
$
 %CFNAME:!CFNAME %
 %Dump:—dspace %k Structures:—sttot  %k Free:—fspace %k Total:—tspace %k

 STRNAME        Storage Type    Lst/Dir Lst     Data    Lock    Lock/Elm
                  (k)           Entries Headers Element Entries Size (b)
                                     Tot/Use         Tot/Use
)MODEL CLEAR(SELECT)
 _Z¬Z              —Z      ¬Z      —Z      —Z      —Z       —Z      —Z
%
                                   —Z      %       —Z       —Z      %
)INIT
 .ZVARS = "(SELECT STRNAME STSIZE STTYPE STDENT STLHD STDELM STLENT
STLSIZE +
            STDENTU STDELMU STLENTU)'
 &ZTDMARK = "****************************** BOTTOM OF DATA +
*******************************'
)PROC
  IF (.RESP = ENTER)
   VER (&SKEY,LIST,CFNAME)

)END
```

Panel IXCCFSTD

```
)ATTR
   _ TYPE(INPUT)  INTENS(HIGH) CAPS(ON) COLOR(RED)
   — TYPE(OUTPUT) INTENS(LOW) JUST(RIGHT)
   ¬ TYPE(OUTPUT) INTENS(LOW) JUST(LEFT)
   ! TYPE(OUTPUT) INTENS(HIGH) JUST(LEFT)
   } TYPE(OUTPUT) INTENS(HIGH) JUST(LEFT) COLOR(RED)
   # TYPE(TEXT) COLOR(RED) INTENS(HIGH)
   ‡ TYPE(TEXT) COLOR(YELLOW) INTENS(HIGH)
   % TYPE(TEXT) COLOR(GREEN) INTENS(HIGH)
   $ TYPE(TEXT) SKIP(ON) INTENS(LOW)
)BODY EXPAND(……)
+…-…#Structure Detail+…-…+
$‡COMMAND%===>_ZCMD                                    %SCROOL ===>_SAMT+
$
 %CFNAME:¬CFNAME %

    STRNAME:      !Z              %     Storage:           —z       %k

    Type:                   —z      %   Response Time:
                                             SYNC:         —Z       %æs
    List/Directory Entries:                  ASYNC:        —Z       %æs
         Max:               —z      %
         Used:              —z      %   Access Rate:
    List Headers:           —z      %        SYNC:         —Z     %/sec
    Data Elements:                           ASYNC:        —Z     %/sec
         Max:               —z      %
         Used:              —z      %
    Lock Entries:
         Max:               —z      %
         Used:              —z      %
    Lock/Element Size:      —z      %

)INIT
 .ZVARS = "(STRNAME +
           STSIZE +
           STTYPE +
           STSTIME +
           STATIME +
           STDENT STDENTU +
           STLHD +
           STscnt +
           stacnt +
           STDELM +
           STDELMU +
           STLENT STLENTU +
           STLSIZE)'

 &ZTDMARK = "****************************** BOTTOM OF DATA +
*******************************'
)PROC
  IF (.RESP = ENTER)
   VER (&SKEY,LIST,CFNAME)

)END
```

Panel IXCST

```
)ATTR
   _ TYPE(INPUT)  INTENS(HIGH) CAPS(ON) COLOR(RED)
   — TYPE(OUTPUT) INTENS(LOW) JUST(RIGHT)
   ¬ TYPE(OUTPUT) INTENS(LOW) JUST(LEFT)
   ! TYPE(OUTPUT) INTENS(HIGH) JUST(LEFT)
   } TYPE(OUTPUT) INTENS(HIGH) JUST(LEFT) COLOR(RED)
   # TYPE(TEXT) COLOR(RED) INTENS(HIGH)
   ‡ TYPE(TEXT) COLOR(YELLOW) INTENS(HIGH)
   % TYPE(TEXT) COLOR(GREEN) INTENS(HIGH)
   $ TYPE(TEXT) SKIP(ON) INTENS(LOW)
)BODY EXPAND(……)
+…-…#Structures Info+…-…+
$‡COMMAND%===>_ZCMD                                  %SCROOL ===>_SAMT+
$
)MODEL clear(select)
+————————————————————————————————————————
_Z% STRNAME:!Z              % Status:¬Z            % ¬Z
+
        %CF:¬Z      %}Z% ¬Z
        %Prefrence List:¬z
+
        %Exclusion List:¬z
+
        %Initsize:—Z      %K - Size:—Z      %K - Rebuild Pct:—Z      %
)INIT
 .ZVARS = "(SELECT STRNAME ALLOC +
           PENDING CFNAME UM CFNODE PLCF XLCF INITSIZE +
           SIZE REBUILDP)'
 &ZTDMARK = "****************************** BOTTOM OF DATA +
*******************************'
)PROC
  IF (.RESP = ENTER)
)END
```

Panel IXCSTST

```
)ATTR
   _ TYPE(INPUT)  INTENS(HIGH) CAPS(ON) COLOR(RED)
   — TYPE(OUTPUT) INTENS(LOW) JUST(RIGHT)
   ¬ TYPE(OUTPUT) INTENS(LOW) JUST(LEFT)
   ! TYPE(OUTPUT) INTENS(HIGH) JUST(LEFT)
   } TYPE(OUTPUT) INTENS(HIGH) JUST(LEFT) COLOR(RED)
   # TYPE(TEXT) COLOR(RED) INTENS(HIGH)
   ‡ TYPE(TEXT) COLOR(YELLOW) INTENS(HIGH)
   % TYPE(TEXT) COLOR(GREEN) INTENS(HIGH)
   $ TYPE(TEXT) SKIP(ON) INTENS(LOW)
```

```
)BODY EXPAND(……)
+…-…#Structures Info+…-…+
$‡COMMAND%===>_ZCMD                                  %SCROOL ===>_SAMT+
$
    STRNAME:!Z               % STATUS:¬Z            % ¬Z
+
        %CF:¬Z      %}Z% ¬Z
        %Preference List:¬z
+
        %Exclusion List:¬z
+
        %Initsize:—z      %k - Size:—z      %k - Rebuild pct:—z      %

                                                               Allow
 Connection Name  Sysname  Jobname  Status           Alter    Rebuild
+————————————————————————————————————————
)MODEL
¬Z               ¬Z       ¬Z       ¬Z                ¬Z%      ¬z%
)INIT
 .ZVARS = "(STRNAME ALLOC +
            PENDING CFNAME UM CFNODE PLCF XLCF INITSIZE +
            SIZE REBUILDP +
            CONNAme +
            SYSNAME +
            jobname +
            cstatus +
            allowa  +
            allowr)'
 &ZTDMARK = "******************************* BOTTOM OF DATA +
*******************************'
)PROC
  IF (.RESP = ENTER)

)END
```

ISPF MESSAGES

You will need to install the following ISPF messages member in a library included in your ISPMLIB concatenation.

Message IXC

```
IXCØØ1E "ERROR..." .ALARM=YES
"ERROR DURING IXCQUERY MACRO..."
IXCØØ2E "ERROR..." .ALARM=YES
"ERROR DURING IXLMG MACRO..."
```

Patrick Renard
CTRNE (France)

DSN1COPY generator utility

The REXX procedure DCU generates several DSN1COPY JCL streams. The DSN1COPY is executed as an MVS job, and could be executed when the DB2 subsystem is either active or not active. My procedure allows the following:

- The creation of a back-up copy of a DB2 dataset on DASD (3390) or tape. The procedure includes all dependent indexes, if the WITHINDX field has a value of YES.
  - SYSUT1 is DB2/VSAM
  - SYSUT2 is a sequential dataset (3390, tape).
- The restoration of a back-up copy of a DB2 dataset from DASD (3390) or tape. The indexes are included.
  - SYSUT1 is DSN1COPY sequential dataset (3390, tape)
  - SYSUT2 is DB2/VSAM.
- The movement of a DB2 dataset to another DB2 dataset. You can specify, via a parameter field, what table spaces you want to copy with DSN1COPY. The procedure determines the set of table spaces matching your search criteria.

  The generated job is a DSN1COPY job with DBID, OBID, and PSID (ISOBIDs for indexes) translation cards also being generated.
  - SYSUT1 is DB2/VSAM, the source subsystem
  - SYSUT2 id DB2/VSAM, the target subsystem.
- The performance of validity checking on a DB2 dataset:
  - SYSUT1 is DB2/VSAM
  - SYSUT2is DUMMY
  - The parameter is CHECK.
- The performance of validity checking on print in a DB2 dataset:

- SYSUT1 is DB2/VSAM
- SYSUT2 is DUMMY
- The parameters are CHECK and PRINT.

- The restoration of a table space from an image copy dataset. In this case, be sure that image copies are produced using the COPY utility with the SHRLEVEL REFERENCE parameter. Using this parameter ensures that the data contained in your image copies is consistent. After using the FULLCOPY parameter to restore a table space, you must recover any indexes associated with that table space. You can do this by using the RECOVER INDEX utility.
  - SYSUT1 is a DB2 full image copy
  - SYSUT2 is DB2/VSAM
  - The parameter is FULLCOPY.

The Main menu is shown below:

```
                                DSN1COPY Utility
Command ===>
              _  Create a back-up copy of a DB2 dataset
              _  Restore a back-up copy of a DB2 dataset

              _  Move a DB2 dataset to another DB2 dataset

              _  Perform validity checking on a DB2 dataset
              _  Perform validity checking on and print a DB2 dataset

              _  Restore a table space from an Image copy


                 Place cursor on choice and press <Enter>
PF3 - End                                                    Avg 1999,"ZB"
```

If you place a cursor on the line 'Move a DB2 dataset to another DB2 dataset' and press Enter, the Entry panel shown below appears:

```
----------  MOVE A DB2 DATASET TO ANOTHER DB2 DATASET          ---------
Command ===>

PARAMETER   PARAMETER VALUE                           PROMPT

SSID     => DSNN                                      DB2 Sub-System
```

```
Identifier
Tosystem => DSNT                                     To Sub-System Id
Location => DB2MB                                    Location name for
Tosystem
Creator  => NADI                                     Table Creator
Name     => TLØ57                                    Table Name
Tsname   => ________                                 Tablespace Name
Dbname   => ________                                 Database   Name
Stopts   => YES                                      Stop tablespace YES
or NO
Withindx => YES                                      Include Index YES
or NO
Runstats => YES                                      Runstats  YES or NO

            Enter values for the DSN1COPY service |

                                PF3 Return
```

Note:

- There must be DDF communication between DB2 subsystems (in my case DSNN and DSNT).
- The location field is DB2 location name on target site (DSNT).
- The DB2 objects (table space, tables, and indexes) must be the same on both sites.

The components of the DCU procedure are:

- DCU – REXX driver procedure:

```
/* REXX */
/* trace r */
 zpfctl = 'OFF'
 address ispexec 'vput (zpfctl) profile'
 CUR='F1'
 address ispexec "display panel(dsn1cmØ) cursor("CUR")"
 do while rc=Ø
    if kurs='F1' | kurs='FIELD1' then do
         Call dcu1 'F1' field1
         CUR='F1'
    end
    if kurs='F2' | kurs='FIELD2' then do
         Call dcu1 'F2' field2
         CUR='F2'
    end
    if kurs='F3' | kurs='FIELD3' then do
         Call dcu2 'F3' field3
         CUR='F3'
```

```
   end
   if kurs='F4' | kurs='FIELD4' then do
        Call dcu1 'F4' field4
        CUR='F4'
   end
   if kurs='F5' | kurs='FIELD5' then do
        Call dcu1 'F5' field5
        CUR='F5'
   end
   if kurs='F6' | kurs='FIELD6' then do
        Call dcu3 'F6' field6
        CUR='F6'
   end
   address ispexec "display panel(dsn1cmØ) cursor("CUR")"
end
exit
```

- DCU1 – REXX procedure:

```
/* REXX */
/* DSN1COPY Utility                                        */
ARG poz text
/* trace r */
 zpfctl = 'OFF'
 Y=MSG("OFF")
 /*********************************************************/
 /*Change to your convention standards                     */
 program = 'DSN1CP1'
 plan    = 'DSN1CP1'
 llib    = 'SKUPNI.BATCH.LOADLIB'
 /*********************************************************/
 address ispexec 'vput (zpfctl) profile'
 Call Aloc
 head=text
 cur='crec'
 Call Create_messg
 TOP:
 field=text
 address ispexec "display panel(dsn1cm1) cursor("CUR")"
 if rc=8 then do
    Call Free_proc
    address ispexec "tbclose "tbname""
    exit
 end
 /* Check input parameters                            */
 if crec=' ' & tabc=' ' & tsnc=' ' & dbnc=' ' then do
    message='At least one Catalog search field must be entered.'
    Call Error 'crec'
 end
 if sts='YES' | sts='NO' then nop
```

```
else do
   message='Valid values for Stop tablespace: YES, NO.'
   Call Error 'sts'
end
if devt='339Ø' | devt='TAPE' then nop
else do
   message='Valid values are 339Ø or TAPE.'
   Call Error 'devt'
end
if devt='TAPE' then do
   if rpd='' | rpd=Ø then rpd=14
   rpdi = verify(rpd,'Ø123456789')
   if rpdi > Ø then do
      message='Enter Numeric value.'
      Call Error 'rpd'
   end
end
if wix='YES' | wix='NO' then nop
else do
   message='Valid values for Include index: YES, NO.'
   Call Error 'wix'
end
parm=substr(crec,1,8)||substr(tabc,1,18)||substr(tsnc,1,8)||,
 substr(dbnc,1,8)||poz||substr(wix,1,3)
messg = "Accessing  db2 system "db2""
messg = time() || " " || messg
Call Send_messg
messg = 'Select   systablespace    information'
messg = time() || " " || messg
Call Send_messg
ADDRESS TSO
QUEUE "RUN PROGRAM("program") PLAN("plan"),
       LIBRARY ('"llib"'),
       PARMS ('/"parm"')"
QUEUE "END "
"DSN SYSTEM("db2")"
if rc=12 then do
   "delstack"
   Call Free_proc
   Call Aloc
   address ispexec 'tbend' tbname
   Call Create_messg
   message = 'Error.   'db2||' ssid is not valid  |'
   Call Error 'db2'
END
"EXECIO * DISKR SYSPRINT (STEM ROW."
if word(row.1,3) ¬= 'Ø' then do
   Call Free_proc
   Call Aloc
   address ispexec 'tbend' tbname
   Call Create_messg
   if word(row.1,3) = 100
   then message= 'No catalog entries found, check Search Fields'
   else message='Error. Sqlcode='||word(row.1,3)
   Call Error 'crec'
end
else do
   address ispexec 'addpop row(1) column(5)'
   address ispexec 'tbcreate "blist" names(ob v1 v2 v3 v4 v5 )'
   count=0
   num=row.0
   do i=2 to row.0
      ob= substr(row.i,2,2)
      v1= word(row.i,2)
      v2= word(row.i,3)
      v3= right(word(row.i,4),3)
      v4= right(word(row.i,7),13)
      v5= right(word(row.i,6),3)
      address ispexec 'tbadd "blist"'
   end
   address ispexec 'tbtop "blist"'
   address ispexec 'tbdispl "blist" panel(dsn1cm3)'
   if rc=8 then do
      Call Free_proc
      address ispexec 'tbend "blist"'
      Call Aloc
      address ispexec rempop all
      address ispexec 'tbend' tbname
      Call Create_messg
      signal top
   end
   address ispexec rempop all
end
ctime=time('s')
messg = 'Calculating Tablespace Dataset Sizes'
messg = time() || " " || messg
Call Send_messg
Call Free_proc
address ispexec 'tbcreate "alist" ,
names(ob db ts pts pr pr1 pri sec detail scu catn)'
asterisks= '***************************************************'
do i=2 to row.0
   count=count+1
end
procent=100/(2*count)
tot=0
cyl=0
scu=0
trk=15
do i=2 to row.0
```

```
      ob = substr(row.i,2,2)
      db = word(row.i,2)
      ts = word(row.i,3)
      if i=2 & poz='F2' then dsn1=userid()||'.DCU.DSN1CØØ1.'||DB||'.'||TS
      pr  = word(row.i,4)
      pr1 = word(row.i,4)
      if pr=Ø then pr1=1
      pr1 = right(pr1,3,'Ø')
      catn= word(row.i,5)
      pri=Ø
      sec=Ø
      part='.IØØØ1.A'||pr1
      file=catn||'.DSNDBD.'||strip(db)||'.'||strip(ts)||part
      dsn = "('"file"')"
      X=OUTTRAP('var.')
      address tso "listc" entries dsn allocation
      X=OUTTRAP('OFF')
      Call Check_dsn
      if rc=Ø then do
         hurba = word(translate(var.9,' ','-'),7)
         if hurba < trunc(73728Ø/trk,Ø) then do
            prip=1
            secp=1
         end
         else do
            prip=trunc((hurba/(73728Ø/trk)+1),Ø)
            secp=max(trunc(prip*Ø.Ø5,Ø),1)
         end
      end
      pri=pri+prip
      sec=sec+secp
      db=space(db,Ø)
      ts=space(ts,Ø)
      tot=tot+pri
      scu=scu+1
      if pr=Ø
      then suf='---'
      else suf=right(pr,3,'Ø')
      detail=right(scu,4)||'  '||left(db,1Ø)||,
             left(ts,12)||right(pri,7)||'    '||suf
      address ispexec 'tbadd "alist"'
      messg = substr(asterisks,1,trunc(procent*(i-1),Ø))
      Call Send_messg
   end
   tot=right(tot,7)
   cyl=trunc(tot/trk,Ø)
   if tot//trk > Ø then cyl=cyl+1
   cyl=right(cyl,7)
   address ispexec 'tbtop "alist"';
   messg = 'Building a Dsn1copy Job Control.'
```

```
messg = time() || " " || messg
Call Send_messg
if poz='F2' & devt='339Ø' then do
   dsn1c = sysdsn("'"dsn1"'")
   Call Check_dsn1
end
/* JCL Dsn1copy Skeleton       */
title = 'DSN1COPY UTILITY'
date=date()
time=time(c)
user=userid()
tempfile=userid()||'.DSN1.DSN1COPY'
address tso
"delete '"tempfile"'"
"free dsname('"tempfile"')"
"free ddname(ispfile)"
"free attrlist(formfile)"
"attrib formfile blksize(8ØØ) lrecl(8Ø) recfm(f b) dsorg(ps)"
"alloc ddname(ispfile) dsname('"tempfile"')",
       "new using (formfile) unit(339Ø) space(1 1) cylinders"
ctime=(time('s')-ctime)%6Ø min (time('s')-ctime)//6Ø
address ispexec
"ftopen"
if poz='F1' | poz='F4' | poz='F5' then "ftincl DSN1COP1"
if poz='F2' then "ftincl DSN1COP2"
"ftclose"
zedsmsg = "JCL shown"
zedlmsg = "DSN1COPY Job Control shown"
"setmsg msg(isrzØØ1)"
"edit dataset('"tempfile"')"
address ispexec 'tbend "alist"'
address ispexec 'tbend "blist"'
address ispexec  "tbclose "tbname""
Exit
Aloc:
  ADDRESS TSO "DELETE '"SYSVAR(SYSUID)".UTIL.DSN1COPY'"
  "ALLOC DD(SYSPRINT) DSN('"SYSVAR(SYSUID)".UTIL.DSN1COPY'),
  SPACE(24 8) TRACK MOD UNIT(339Ø) RECFM(F,B) LRECL(133),
  BLKSIZE(133Ø) F(SYSPRINT) CATALOG REUSE "
Return
Error:
  ARG cur_par
  cur=cur_par
  address ispexec "setmsg msg(dsncØØ1)"
  signal top
Return
Free_proc:
  "execio Ø diskr sysprint (finis"
  address tso "free f(sysprint)"
Return
```

```
 Check_dsn:
  if rc>0 then do
     message=file||' not found.'
     cur='crec'
     address ispexec "setmsg msg(dsnc001)"
     Call Free_proc
     Call Aloc
     address ispexec 'tbend "alist"'
     address ispexec 'tbend "blist"'
     address ispexec  "tbclose "tbname""
     Call Create_messg
     signal top
  end
 Return
 Check_dsn1:
  if dsn1c ¬= 'OK'
  then do
   say 'Dataset '||dsn1||' not found. Define first DSN1COPY Dataset'
  end
 Return
 Create_messg:
  messg = "S"||userid()
  tbname = 'TB'||time(s)
  address ispexec "tbcreate "tbname" names(messg) write replace"
 Return
 Send_messg:
  address ispexec "tbadd " tbname
  address ispexec "control display lock "
  address ispexec "addpop row(13) column(6)"
  address ispexec "tbdispl "tbname" panel(DSN1UT)"
  address ispexec rempop
 Return
```

- DCU2 – REXX procedure:

```
/* REXX */
/* DSN1COPY Utility                                            */
ARG poz text
/* trace r */
 zpfctl = 'OFF'
 Y=MSG("OFF")
 /***********************************************************/
 /*Change to your convention standards                         */
 program = 'DSN1CP2'
 plan    = 'DSN1CP2'
 llib    = 'SKUPNI.BATCH.LOADLIB'
 /***********************************************************/
 address ispexec 'vput (zpfctl) profile'
 Call Aloc
 head=text
```

```
cur='crec'
Call Create_messg
TOP:
field=text
address ispexec "display panel(dsn1cm2) cursor("CUR")"
if rc=8 then do
   Call Free_proc
   address ispexec "tbclose "tbname""
   exit
end
/* Check input parameters                              */
if crec=' ' & tabc=' ' & tsnc=' ' & dbnc=' ' then do
   message='At least one Catalog search field must be entered.'
   Call Error 'crec'
end
if sysi=' '  then do
   message='Enter target Sub System Id.'
   Call Error 'sysi'
end
if loc=' '  then do
   message='Enter DDF Location name for target Sub System Id.'
   Call Error 'loc'
end
if sts='YES' | sts='NO' then nop
else do
   message='Valid values for Stop tablespace: YES, NO.'
   Call Error 'sts'
end
if wix='YES' | wix='NO' then nop
else do
   message='Valid values for Indexes: YES, NO.'
   Call Error 'wix'
end
if rus='YES' | wix='NO' then nop
else do
   message='Valid values for Runstats: YES, NO.'
   Call Error 'rus'
end
parm=substr(crec,1,8)||substr(tabc,1,18)||substr(tsnc,1,8)||,
 substr(dbnc,1,8)||substr(wix,1,3)||substr(loc,1,8)
messg = "Accessing  db2 system "db2""
messg = time() || " " || messg
Call Send_messg
messg = 'Select   systablespace    information'
messg = time() || " " || messg
Call Send_messg
ADDRESS TSO
QUEUE "RUN PROGRAM("program") PLAN("plan"),
       LIBRARY ('"llib"'),
       PARMS ('/"parm"')"
```

```
QUEUE "END "
"DSN SYSTEM("db2")"
if rc=12 then do
   "delstack"
   Call Free_proc
   Call Aloc
   address ispexec 'tbend' tbname
   Call Create_messg
   message = 'Error.   'db2||' ssid is not valid  |'
   Call Error 'db2'
END
"EXECIO * DISKR SYSPRINT (STEM ROW."
if word(row.1,3) ¬= 'Ø' then do
   Call Free_proc
   Call Aloc
   address ispexec 'tbend' tbname
   Call Create_messg
   if word(row.1,3) = 1ØØ
   then message= 'No catalog entries found, check Search Fields'
   else message='Error. Sqlcode='||word(row.1,3)
   Call Error 'crec'
end
else do
   address ispexec 'tbcreate "blist" names(v1 v2 v3 v4 V5 V6 V7 V8)'
   count=Ø
   num=row.Ø
   job=1
   v1='';v2='';v3='';v4='';v5='';v6='';v7='';v8=''
   do i=2 to row.Ø
      if substr(row.i,2,2)='TS' then do
         v1= word(row.i,2)
         v2= word(row.i,3)
         v3= right(word(row.i,4),2)
         v4= word(row.i,8)
         if db2=sysi then do
            v5=v1
            v6=v2
            v7=v3
            v8=v4
         end
         else do
            v5=word(row.i,12)
            v6=word(row.i,13)
            v7=v3
            if v5='-'
            then v8='Table not found'
            else v8=v4
            if v5='-' then job=Ø
         end
         address ispexec 'tbadd "blist"'
      end
   end
   address ispexec 'tbtop "blist"'
   address ispexec 'tbdispl "blist" panel(dsn1cm4)'
   if rc=8 | job=Ø then do
      Call Free_proc
      address ispexec 'tbend "blist"'
      Call Aloc
      address ispexec 'tbend' tbname
      Call Create_messg
      signal top
   end
end
Call Free_proc
address ispexec,
'tbcreate "alist" names(ca1 db1 ts1 tab prts pr1 xid1,
                        xid2 xid3 ca2 db3 ts2 scu detail)'
scu=Ø
do i=2 to row.Ø
   if substr(row.i,2,2)='TS' then do
      ca1 = word(row.i,5)
      db1 = word(row.i,2)
      ts1 = word(row.i,3)
      pr1 = right(word(row.i,4),3,'Ø')
      prts= word(row.i,6)
      tab = word(row.i,8)
      dbid1 = word(row.i,9)
      psid1 = word(row.i,1Ø)
      obid1 = word(row.i,11)
      ca2 = word(row.i,14)
      db3 = word(row.i,12)
      ts2 = word(row.i,13)
      dbid2 = word(row.i,15)
      psid2 = word(row.i,16)
      obid2 = word(row.i,17)
      xid1 = right(dbid1,3)||','||left(dbid2,3)
      xid2 = right(psid1,3)||','||left(psid2,3)
      xid3 = right(obid1,3)||','||left(obid2,3)
      scu=scu+1
      detail=right(scu,3)||right(db1,1Ø)||right(ts1,11)||,
             right(pr1,5)||right(db3,12)||right(ts2,11)||right(pr1,5)
      address ispexec 'tbadd "alist"'
   end
end
address ispexec 'tbtop "alist"';
if wix='YES' then do
   address ispexec,
   'tbcreate "ilist" names(idb1 isp1 ipr1 ica1 ipr ixid1,
                           ixid2 ixid3 idb2 isp2 ica2 icu line)'
   icu=Ø
   do i=3 to row.Ø
      if substr(row.i,2,2)='IX' then do
         idb1 = word(row.i,2)
         isp1 = word(row.i,3)
         ipr1 = right(word(row.i,4),3,'Ø')
         ica1 = word(row.i,5)
         ipr  = word(row.i,6)
         idbid1 = word(row.i,7)
         isobid1= word(row.i,8)
         iobid1 = word(row.i,9)
         idb2 = word(row.i,1Ø)
         isp2 = word(row.i,11)
         ica2 = word(row.i,12)
         idbid2 = word(row.i,13)
         isobid2= word(row.i,14)
         iobid2 = word(row.i,15)
         ixid1 = right(idbid1,3)||','||left(idbid2,3)
         ixid2 = right(isobid1,3)||','||left(isobid2,3)
         ixid3 = right(iobid1,3)||','||left(iobid2,3)
         icu=icu+1
         line=right(icu,3)||right(idb1,1Ø)||right(isp1,11)||,
          right(ipr1,5)||right(idb2,12)||right(isp2,11)||right(pr1,5)
         address ispexec 'tbadd "ilist"'
      end
   end
   address ispexec 'tbtop "ilist"';
end
messg = 'Building a Dsn1copy Job Control.'
messg = time() || " " || messg
Call Send_messg
/* JCL Dsn1copy Skeleton       */
title = 'DSN1COPY UTILITY'
date=date()
time=time(c)
user=userid()
tempfile=userid()||'.DSN1.DSN1COPY'
address tso
"delete '"tempfile"'"
"free dsname('"tempfile"')"
"free ddname(ispfile)"
"free attrlist(formfile)"
"attrib formfile blksize(8ØØ) lrecl(8Ø) recfm(f b) dsorg(ps)"
"alloc ddname(ispfile) dsname('"tempfile"')",
       "new using (formfile) unit(339Ø) space(1 1) cylinders"
address ispexec
"ftopen"
"ftincl DSN1COP3"
"ftclose"
zedsmsg = "JCL shown"
zedlmsg = "DSN1COPY Job Control shown"
```

```
 "setmsg msg(isrzØØ1)"
 "edit dataset('"tempfile"')"
 address ispexec 'tbend "alist"'
 address ispexec 'tbend "blist"'
 if wix='YES' then address ispexec 'tbend "ilist"'
 address ispexec  "tbclose "tbname""
 Exit
 Aloc:
   ADDRESS TSO "DELETE '"SYSVAR(SYSUID)".UTIL.DSN1COPY'"
   "ALLOC DD(SYSPRINT) DSN('"SYSVAR(SYSUID)".UTIL.DSN1COPY'),
   SPACE(24 8) TRACK MOD UNIT(339Ø) RECFM(F,B) LRECL(13Ø),
   BLKSIZE(13ØØ) F(SYSPRINT) CATALOG REUSE "
 Return
 Error:
   ARG cur_par
   cur=cur_par
   address ispexec "setmsg msg(dsncØØ1)"
   signal top
 Return
 Free_proc:
   "execio Ø diskr sysprint (finis"
   address tso "free f(sysprint)"
 Return
 Create_messg:
  messg = "S"||userid()
  tbname = 'TB'||time(s)
  address ispexec "tbcreate "tbname" names(messg) write replace"
 Return
 Send_messg:
  address ispexec "tbadd " tbname
  address ispexec "control display lock "
  address ispexec "addpop row(13) column(6)"
  address ispexec "tbdispl "tbname" panel(DSN1UT)"
  address ispexec rempop
 Return
```

- DCU3 – REXX procedure:

```
/* REXX */
/* DSN1COPY Utility                                           */
ARG poz text
/* trace r */
 zpfctl = 'OFF'
 Y=MSG("OFF")
 /**********************************************************/
 /*Change to your convention standards                       */
 program = 'DSN1CP3'
 plan    = 'DSN1CP3'
 llib    = 'SKUPNI.BATCH.LOADLIB'
 /**********************************************************/
```

```
address ispexec 'vput (zpfctl) profile'
Call Aloc
head=text
cur='crec'
TOP:
field=text
address ispexec "display panel(dsn1cm5) cursor("CUR")"
if rc=8 then do
   Call Free_proc
   exit
end
/* Check input parameters                            */
if crec=' ' then do
   message='Enter creator name.'
   Call Error 'crec'
end
if tabc=' ' then do
   message='Enter table name.'
   Call Error 'tabc'
end
if sts='YES' | sts='NO' then nop
else do
   message='Valid values for Stop tablespace: YES, NO.'
   Call Error 'sts'
end
parm=substr(crec,1,8)||substr(tabc,1,18)
ADDRESS TSO
QUEUE "RUN PROGRAM("program") PLAN("plan"),
       LIBRARY ('"llib"'),
       PARMS ('/"parm"')"
QUEUE "END "
"DSN SYSTEM("db2")"
if rc=12 then do
   "delstack"
   Call Free_proc
   Call Aloc
   message = 'Error.   'db2||' ssid is not valid  |'
   Call Error 'db2'
END
"EXECIO * DISKR SYSPRINT (STEM ROW."
if word(row.1,3) = 'Ø' & word(row.2,2) = 1ØØ then do
   Call Free_proc
   Call Aloc
   message= 'Image copy not found.'
   Call Error 'crec'
end
if word(row.1,3) ¬= 'Ø' then do
   Call Free_proc
   Call Aloc
   if word(row.1,3) = 1ØØ
   then message= 'No catalog entries found, check Search Fields'
   else message='Error. Sqlcode='||word(row.1,3)
   if word(row.1,3) = 9999
   then message= 'Partition tablespace not supported.'
   Call Error 'crec'
end
else do
   address ispexec,
   'tbcreate "ilist" names(icd ict ity ipar disk dsn db ts vc)'
   num=row.Ø
   do i=2 to row.Ø
      icd= word(row.i,2)
      ict= word(row.i,3)
      ity= 'F'
      ipar='Ø'
      disk=word(row.i,4)
      dsn= word(row.i,5)
      db = word(row.i,6)
      ts = word(row.i,7)
      vc = word(row.i,8)
      address ispexec 'tbadd "ilist"'
   end
   address ispexec 'tbtop "ilist"'
   address ispexec 'tbdispl "ilist" panel(dsn1cm6)'
   if rc=8 then do
      Call Free_proc
      address ispexec 'tbend "ilist"'
      Call Aloc
      signal top
   end
end
ctime=time('s')
Call Free_proc
/* JCL Dsn1copy Skeleton       */
title = 'DSN1COPY UTILITY'
date=date()
time=time(c)
user=userid()
tempfile=userid()||'.DSN1.DSN1COPY'
address tso
"delete '"tempfile"'"
"free dsname('"tempfile"')"
"free ddname(ispfile)"
"free attrlist(formfile)"
"attrib formfile blksize(8ØØ) lrecl(8Ø) recfm(f b) dsorg(ps)"
"alloc ddname(ispfile) dsname('"tempfile"')",
       "new using (formfile) unit(339Ø) space(1 1) cylinders"
ctime=(time('s')-ctime)%6Ø min (time('s')-ctime)//6Ø
address ispexec
"ftopen"
```

```
"ftincl DSN1COP4"
"ftclose"
zedsmsg = "JCL shown"
zedlmsg = "DSN1COPY Job Control shown"
"setmsg msg(isrzØØ1)"
"edit dataset('"tempfile"')"
address ispexec 'tbend "ilist"'
Exit
Aloc:
  ADDRESS TSO "DELETE '"SYSVAR(SYSUID)".UTIL.DSN1COPY'"
  "ALLOC DD(SYSPRINT) DSN('"SYSVAR(SYSUID)".UTIL.DSN1COPY'),
  SPACE(24 8) TRACK MOD UNIT(339Ø) RECFM(F,B) LRECL(133),
  BLKSIZE(133Ø) F(SYSPRINT) CATALOG REUSE "
Return
Error:
  ARG cur_par
  cur=cur_par
  address ispexec "setmsg msg(dsncØØ1)"
  signal top
Return
Free_proc:
  "execio Ø diskr sysprint (finis"
  address tso "free f(sysprint)"
Return
```

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

Bernard Zver
Database Administrator
Informatika Maribor (Slovenia)

Point-in-time DB2 buffer pool reporting – revisited

Now available from our Web site is an updated version of the DB2BPRPT edit macro that was published in DB2 Update, in Issue 54, April 1997, in an article entitled *Point-in-time DB2 buffer pool reporting*. This new version has been updated for Y2K compatibility, as well as to handle 10 digits on some of the large computations. Our thanks for the update go to Antonio Salcedo.

DB2 news

Dynasty Technologies has announced Version 4 of its DDE DYNASTY Development Environment.

Among the new features is improved OO functionality with added support for interceptors, enabling developers to intercept messages and add rules and conditions.

There's said to be better facilities for component development, with the addition of the concept of an interface class, enabling developers to break their application into components more easily than before.

What's more, the bridge between DYNASTY and Rational's Rose product has been improved to allow greater interaction and changes have been made to speed up the generation process.

I18N support allows the application to have multiple language support and the server processes will act accordingly. This makes it possible for one application to support Japanese, Swedish, and English clients.

There's also native support for OS/390, CICS, and DB2 plus Microsoft Repository support during the development process.

For further information contact:
Dynasty Technologies, 101 Redwood Shores Parkway, #200 Redwood Shores, CA 94065, USA.
Tel: (650) 631 5430.
http://www.dynasty.com.

* * *

Ardent Software has announced that it has joined with IBM to develop DataStage/DB2 for Adabas. The new product integrates DB2 UDB and OLAP Server with DataStage, Ardent's extraction, transformation, and loading (ETL) tool.

DataStage/DB2 for Adabas will enable Adabas customers running mainframes to implement and populate Unix-based IBM data marts or data warehouses.

The companies estimate that Adabas users can speed deployment and reduce costs by more than 70% by using DataStage/DB2 to automate the data extraction and movement process.

For further information contact:
Ardent Software, 50 Washington Street, Westborough, MA 01581-1021, USA.
Tel: (508) 366 3888.
http://www.ardentsoftware.com.

* * *

Comshare has announced that Version 3.2 of its Comshare BudgetPLUS solution for management planning and control now supports DB2 Universal Database (UDB) Version 6.1.

BudgetPLUS is fully Web-architected and already runs on IBM DB2 OLAP Server in an NT environment.

BudgetPLUS comes with built-in best practices for planning, budgeting, management reporting, and analysis.

For further information contact:
Comshare, PO Box 1588, Ann Arbor, MI 48108, USA.
Tel: (734) 994 4800.
http://www.comshare.com.