



87

DB2

January 2000

In this issue

- 3 Monitoring space growth using traces
 - 13 Fuzzy SELECT
 - 20 Extracting catalog information – part 2
 - 37 Optimizing tablespace and indexspace – part 1
 - 48 DB2 news
-

© Xephon plc 2000

update

DB2 Update

Published by

Xephon
27-35 London Road
Newbury
Berkshire RG14 1JL
England
Telephone: 01635 38030
From USA: 01144 1635 38030
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

Disclaimer

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

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 1994 issue, are available separately to subscribers for £22.50 (\$33.50) each including postage.

© Xephon plc 2000. All rights reserved. None of the text in this publication may be reproduced, stored in a retrieval system, or transmitted in any form or by any means, without the prior permission of the copyright owner. Subscribers are free to copy any code reproduced in this publication for use in their own installations, but may not sell such code or incorporate it in any commercial product. No part of this publication may be used for any form of advertising, sales promotion, or publicity without the written permission of the publisher. Copying permits are available from Xephon in the form of pressure-sensitive labels, for application to individual copies. A pack of 240 labels costs \$36 (£24), giving a cost per copy of 15 cents (10 pence). To order, contact Xephon at any of the addresses above.

Printed in England.

Monitoring space growth using traces

INTRODUCTION

IBM has introduced the new IFCID 258 to trace dataset extents' activity and monitor database space growth. IFCID 258 is available via APAR PQ17544 or with DB2 Version 6.

Previously, many shops were dependent on home-grown utilities, using IDCAM LISTCAT etc, to track dataset growth.

This new IFCID is very useful for DB2 shops to track and monitor DB2 dataset growth and to find out when DB2 VSAM datasets reach their maximum limit in size, extents, or number of volumes. This IFCID requires class 3 statistical trace. Most of the DB2 shops keep class 3 statistical traces in SMF on all the time, because of its low overhead.

The following Assembler program extracts information from SMF dumps and generates a report on dataset extent activities in DB2 subsystems.

INVOCATION

In order to run this routine the following DD cards are needed.

SMFIN

The SMFIN DD card contains the SMF dump dataset as an input dataset. The sample JCL expects the SMF dump dataset to have BLKSIZE 27998, LRECL 32767, and RECFM as VBS. If your installation has SMF dumps with a different blocksize, you need to change the DCB parameter in the program appropriately.

SMFOUT

The SMFOUT DD card contains the dataset extent activity report. The dataset should have RECFM FB, LRECL 255, and BLKSIZE 25500.

SYSOUT

The SYSOUT DD card contains write error message, the number of SMF records processed, etc.

SAMPLE JCL

```
//SMF      EXEC PGM=I258,REGION=ØM
//STEPLIB DD DSN=<load library>,DISP=SHR
//SYSOUT   DD SYSOUT=*
//SYSUDUMP DD SYSOUT=*
//SMFIN    DD DISP=SHR,DSN=<smfdump dataset>
//SMFOUT   DD DISP=OLD,DSN=<output report dataset>
```

SAMPLE ASSEMBLE JCL

```
//ASM      EXEC PGM=ASMA9Ø,
//          PARM=(NODECK,OBJECT,NOALIGN),
//          REGION=ØM
//SYSLIB   DD DSN=SYS1.MACLIB,
//          DISP=SHR
//          DD DSN=<source dataset>,DISP=SHR
//          DD DSN=SYSDB2C.DSNMACS,DISP=SHR
//SYSUT1   DD DSN=&UT1,
//          SPACE=(17ØØ,(4ØØ,5Ø)),
//          UNIT=SYSDA
//SYSPRINT DD SYSOUT=*
//SYSLIN   DD DSN=&&LOADSET(I258),
//          UNIT=SYSDA,
//          SPACE=(4ØØ,(4Ø,1Ø,2)),
//          DISP=(MOD,PASS),
//          DCB=(LRECL=8Ø,BLKSIZE=Ø,RECFM=FB)
//SYSIN    DD DISP=SHR,DSN=<input data(I258) >
//*
//* * * * *
//*
//LKED     EXEC PGM=HEWLHØ96,
//          PARM=(XREF,LIST,AMODE(31),RMODE(24)),
//          REGION=ØM,COND=(4,LT,ASM)
//SYSPUNCH DD DSN=&&LOADSET(I258),
//          DISP=(OLD,DELETE)
//SYSLMOD  DD DSN=<load library>,DISP=SHR
//SYSUT1   DD DSN=&UT1,
//          UNIT=SYSDA,
//          SPACE=(1Ø24,(5Ø,2Ø))
//SYSPRINT DD SYSOUT=*
//SYSLIN   DD *
//          INCLUDE SYSPUNCH(I258)
```

ENTRY I258
NAME I258(R)

/*

PROGRAM CODE

```
TITLE 'DATASET EXTENT REPORT TRACES FROM SMF'
*****
***** DATASET EXTENT REPORT PROGRAM *****
*****
* AUTHOR: VENKAT R PILLAY *
* FOR MORE INFORMATION...: DB2 ADMIN GUIDE *
* APAR INFO : WWW.SOFTWARE.IBM.COM/DATA/DB2/OS390/V5APAR.HTML *
*****
***** REGISTERS *****
*****
* 3 - SMF DATASET ADDRESS *
* 4 - SELF DEFINING SECTION ADDRESS *
* 5 - PRODUCT / DATA SECTION ADDRESS *
* 12 - BASE ADDRESS *
*****
***** MACROS USED *****
*****
* CALL - RUN EXTERNAL ROUTINES *
* CLOSE - CLOSE A DATASET *
* DCB - DATA COMMUNICATION BLOCK *
* GET - READ FROM A DATASET QSAM *
* OPEN - OPEN A DATASET *
* PUT - WRITE A DATASET (QSAM) *
*****
***** PSEUDO CODE *****
*****
* READ SMF RECORDS *
* DO UNTIL ALL SMF RECORDS ARE READ *
* CHECK FOR RECORD TYPE 102 *
* CHECK FOR IFCID 258 *
* GET THE OFFSET OF THE DATASECTION *
* FORMAT AND PRINT DATA SECTION *
* END *
*****
* ENTRY AND SETUP
I258 CSECT
      YREGS
      SAVE (14,12)
      LR R12,R15 GET ENTRY POINT
      USING I258,R12 USE R4 AS BASE
      XC SAVEAREA(72),SAVEAREA INITIALIZE OUR SAVE AREA
      ST R13,SAVEAREA+4 SAVE CALLERS SAVE AREA ADDRESS
      LA R13,SAVEAREA POINT R13 AT OUR SAVE AREA
```

```

MVI PRNTLINE,C' '
MVC PRNTLINE+1(254),PRNTLINE ..INITIALIZE PRINT LINE
MVI MSGLINE,C' '
MVC MSGLINE+1(132),MSGLINE ..INITIALIZE PRINT LINE
SETUP DS 0H
OPEN (SYSOUT,(OUTPUT)) OPEN SYSOUT DATASET
LTR R15,R15 CHECK OPEN RETURN CODE
BNZ RETURN2
OPEN (INDATA,(INPUT),OUTDATA,(OUTPUT))
LTR R15,R15 CHECK OPEN RETURN CODE
BNZ ERR_RT
PUT OUTDATA,HDRLINE WRITE HEADER
PUT OUTDATA,HDRLIN1 WRITE HEADER
PUT OUTDATA,UNDERLIN WRITE UNDERLINES
* READ SMF
READSMF GET INDATA
LR R3,R1 SAVE ADDRESS OF SMF INTO R3
USING SM101,R3 POINT TO QWAS (SMF HEADER)
L R6,RECCOUNT LOAD THE COUNTER TO R6
A R6,ONE ADD 1 TO COUNTER
ST R6,RECCOUNT STORE THE COUNTER
CLI SM101RTY,X'66' CHECK SMF RECORD TYPE 102 OR NOT
BNE READSMF IF YES CHECK OTHER PARAMETER
SLEN LH R7,SM101LEN CHECK IF SMF RECORD IS OK
C R7,=F'0' IF DEFECTIVE THEN
BNH READSMF SKIP THIS AND READ NEXT
L R6,SCOUNT LOAD THE COUNTER TO R6
A R6,ONE ADD 1 TO COUNTER
ST R6,SCOUNT STORE THE COUNTER
GETDET LA R4,28(R3) GET THE SELF DEFINING SECTIONS
USING QWA0,R4 POINT TO QWA0
L R5,QWA01PS0 LOAD THE OFFSET OF PRODUCT SECT
AR R5,R3 LOAD THE OFFSET OF PRODUCT SECT
USING QWHS,R5 POINT TO STANDARD HEADER
CLC QWHSIID,IFCID CHECK IF IFCID IS 258 OR NOT
BNE READSMF DIFFERENT IFCID DONT PROCESS
L R6,ICOUNT LOAD THE COUNTER TO R6
A R6,ONE ADD 1 TO COUNTER
ST R6,ICOUNT STORE THE COUNTER
* DATE TIME FIELD
BAL R6,DT_TM FORMAT THE DATE FROM SMF HEADER
* DATE TIME FIELD ENDS
DROP R5
L R5,QWA01R10 LOAD THE OFFSET OF DATA SECTION
BAL R6,DETAIL FILL THE DETAILS
PUT OUTDATA,PRNTLINE WRITE THE FORMATTED OUTPUT
B READSMF
RETURN1 DS 0H
MVC MSG1,=C' SUMMARY '
PUT SYSOUT,MSGLINE

```

```

MVC MSG1,=C'          ===== '
PUT  SYSOUT,MSGLINE
MVC MSG1,=C'          '
PUT  SYSOUT,MSGLINE
*
*
MVC MSG1,=C' NUMBER OF SMF 102 RECORDS ='
L   R8,SCOUNT          CONVERT BINARY
BAL R9,CVT_COD1        TO
MVC MSG2(8),CODE       DISPLAY FORMAT
PUT  SYSOUT,MSGLINE
*
MVC MSG1,=C' NUMBER OF IFCID 258 RECORDS ='
L   R8,ICOUNT          CONVERT BINARY
BAL R9,CVT_COD1        TO
MVC MSG2(8),CODE       DISPLAY FORMAT
PUT  SYSOUT,MSGLINE
*
MVC MSG1,=C' THE NUMBER OF SMF RECORDS READ ='
L   R8,RECCOUNT        CONVERT BINARY
BAL R9,CVT_COD1        TO
MVC MSG2(8),CODE       DISPLAY FORMAT
PUT  SYSOUT,MSGLINE
*
MVI  MSGLINE,C' '
MVC  MSGLINE+1(132),MSGLINE  ..INITIALIZE MESSAGE LINE
PUT  SYSOUT,MSGLINE
MVI  MSGLINE,C'='
MVC  MSGLINE+1(132),MSGLINE  ..INITIALIZE MESSAGE LINE
PUT  SYSOUT,MSGLINE
*
CLOSE (INDATA)          CLOSE SMF DATASET
CLOSE (OUTDATA)        CLOSE OUTPUT FILE
RETURN2 DS  0H
CLOSE (SYSOUT)         CLOSE SYSOUT
L   R13,SAVEAREA+4     GET CALLERS SAVEAREA ADDRESS
RETURN (14,12)
ERR_RT MVC  MSGLINE,=C' ERROR IN OPENING DATASET '
PUT  SYSOUT,MSGLINE    WRITE THE MESSAGE
B   RETURN2
CVT_COD1 DS  0H
* CONVERT FULL WORD INTO DISPLAY FORMAT
CVD  R8,D
MVC  CODE,=X'402020202020212020' TWO SIGNIFICANT DIGITS
ED   CODE,D+4          CHOP OFF LAST 4 DIGITS
BR   R9                GO BACK TO THE CALLER
CVT_COD2 DS  0H
* CONVERT FULL WORD INTO DISPLAY FORMAT
CVD  R8,D
MVC  CODE,=X'4020202020202020' ALL CAN BE SPACES

```

	ED	CODE,D+4	CHOP OFF LAST 4 DIGITS
	BR	R9	GO BACK TO THE CALLER
DT_TM	DS	ØH	
	SR	R8,R8	CLEAR REGISTER 8
	SR	R9,R9	CLEAR REGISTER 9
	L	R9,SM1Ø1TME	LOAD TIME INTO REGISTER 9
	D	R8,HOUR1	DIVIDE BY 36ØØØØ
	ST	R8,WREM	KEEP REMAINDER FOR FUTURE USE
	LR	R8,R9	
	BAL	R9,CVT_COD1	CALL CONVERT ROUTINE
	MVC	WHR,CODE	MOVE THE CONVERTED CODE TO HOUR
	SR	R8,R8	CLEAR REGISTER 8
	SR	R9,R9	CLEAR REGISTER 9
	L	R9,WREM	
	D	R8,MINT1	DIVIDE BY 6ØØØ TO GET THE MINUT
	ST	R8,WREM	KEEP REMAINDER FOR FUTURE USE
	LR	R8,R9	
	BAL	R9,CVT_COD1	CALL CONVERSION ROUTINE
	MVC	WMN,CODE	MOVE CONVERTED MINUTES VALUE
	SR	R8,R8	CLEAR REGISTER 8
	SR	R9,R9	CLEAR REGISTER 9
	L	R9,WREM	
	D	R8,SECD1	DIVIDE BY 1ØØ TO GET THE SECON
	LR	R8,R9	
	BAL	R9,CVT_COD1	CALL THE CONVERSION ROUTINE
	MVC	WSC,CODE	MOVE CONVERTED SECONDS VALUE
	MVC	HR1,HR	
	MVC	MN1,MN	
	MVC	SC1,SC	
	MVC	WTIME,WTIMEC	MOVE THE FORMATTED FIELD TO OTP
	MVC	WDATEC,SM1Ø1DTE	GET THE DATE FROM SMF HEADER
	MVC	WDATE,PATTERN2	
	ED	WDATE,WDATEC+1	FORMAT THE DATE
	MVC	WSSI,SM1Ø1SSI	GET THE DB2 SYSTEM NAME
	BR	R6	GO BACK TO THE CALLER
*	GET THE DATA SECTION DETAILS START		
DETAIL	DS	ØH	
	AR	R5,R3	GET THE OFFSET
	USING	QWØ258,R5	MAP THE DSECT
	MVC	WØ258DS,QWØ258DS	MOVE DATASET TO OUTPUT FIELD
	MVC	WØ258DN,QWØ258DN	DATABASE NAME
	MVC	WØ258TN,QWØ258TN	SPACE NAME (INDEX OR TABLESPACE
	L	R8,QWØ258PQ	PRIMARY QUANTITY IN 4KB
	BAL	R9,CVT_COD2	
	MVC	WØ258PQ,CODE	
*			
	L	R8,QWØ258SQ	SECONDARY QUANTITY IN 4KB
	BAL	R9,CVT_COD2	
	MVC	WØ258SQ,CODE	
*			
	L	R8,QWØ258MS	MAX DATASET SIZE 4KB

```

      BAL R9,CVT_COD2
      MVC W0258MS, CODE
*
      L    R8,QW0258HB          HI ALLOCATED BEFORE EXTENTS
      BAL R9,CVT_COD2          (4KB)
      MVC W0258HB, CODE
*
      L    R8,QW0258HA          HI ALLOCATED AFTER EXTENTS
      BAL R9,CVT_COD2          (4KB)
      MVC W0258HA, CODE
*
      LH   R8,QW0258XM          MAXIMUM EXTENTS ALLOWED
      BAL R9,CVT_COD2
      MVC W0258XM, CODE
*
      LH   R8,QW0258XB          NO OF EXTENTS BEFORE
      BAL R9,CVT_COD2
      MVC W0258XB, CODE
*
      LH   R8,QW0258XA          NO OF EXTENTS AFTER
      BAL R9,CVT_COD2
      MVC W0258XA, CODE
*
      LH   R8,QW0258VM          MAX VOLUME ALLOWED
      BAL R9,CVT_COD2
      MVC W0258VM, CODE
*
      LH   R8,QW0258VB          NO OF VOLUME BEFORE
      BAL R9,CVT_COD2
      MVC W0258VB, CODE
*
      LH   R8,QW0258VA          NO OF VOLUMES AFTER
      BAL R9,CVT_COD2
      MVC W0258VA, CODE
*
      BR   R6
*
*  DETAIL DATA SECTION ENDS
*
      LTORG
SYSOUT  DCB  DSORG=PS,MACRF=PM,DDNAME=SYSOUT,          X
          RECFM=FBA,LRECL=133,BLKSIZE=133
INDATA  DCB  DSORG=PS,MACRF=GL,DDNAME=SMFIN,EODAD=RETURN1,  X
          RECFM=VBS,LRECL=32767,BLKSIZE=27998,BFTEK=A
OUTDATA DCB  DSORG=PS,MACRF=PM,DDNAME=SMFOUT,LRECL=255,    X
          RECFM=FB,BLKSIZE=25500
D       DS   0D,PL8          DOUBLE WORD
CODE    DS   CL8            CONVERTED FROM BINARY TO DISPLAY
ONE     DC   F'1'
RECCOUNT DC F'0'          TOTAL SMF RECORD COUNT
SCOUNT  DC   F'0'          TOTAL 102 TYPE COUNT

```


	DC	CL1' '
	DC	CL8' -----'
	DC	CL1' '
	DC	CL8' -----'
	DC	CL1' '
	DC	CL8' -----'
	DC	CL1' '
	DC	CL44'-----'
	DC	CL8' '
	DC	CL1' '
	DC	CL8' '
	DC	CL1' '
	DC	CL8' '
	DC	CL1' '
	DC	CL45' '
HDRLINE	DC	ØCL255' '
HTIME	DC	CL8' TIME'
	DC	CL1' '
HDATE	DC	CL7' DATE'
	DC	CL1' '
HSSI	DC	CL4' MEM'
	DC	CL1' '
HØ258DN	DC	CL8' DBNAME'
	DC	CL1' '
HØ258TN	DC	CL8' SPACENAM'
	DC	CL1' '
HØ258PQ	DC	CL8' PRIQTY'
	DC	CL1' '
HØ258SQ	DC	CL8' SECQTY'
	DC	CL1' '
HØ258MS	DC	CL8' MX SIZE'
	DC	CL1' '
HØ258HB	DC	CL8' SZ BEFR'
	DC	CL1' '
HØ258HA	DC	CL8' SZ AFTR'
	DC	CL1' '
HØ258XM	DC	CL8' XT MAX'
	DC	CL1' '
HØ258XB	DC	CL8' XT BEFR'
	DC	CL1' '
HØ258XA	DC	CL8' XT AFTR'
	DC	CL1' '
HØ258VM	DC	CL8' MX VOL'
	DC	CL1' '
HØ258VB	DC	CL8' VL BEFR'
	DC	CL1' '
HØ258VA	DC	CL8' VL AFTR'
	DC	CL1' '
HØ258DS	DC	CL44' DB2 VSAM NAME'
HØ258TS	DC	CL8' '
	DC	CL1' '

H0258DB	DC	CL8' '
	DC	CL1' '
H0258PS	DC	CL8' '
	DC	CL1' '
	DC	CL45' '
HDRLIN1	DC	ØCL255
	DC	CL40' '
	DC	CL9' (4K)'
	DC	CL9' (4K)'
	DC	CL9' (4K)'
	DC	CL9' (4K)'
	DC	CL9' (4K)'
	DC	CL170' '
MSGLINE	DC	ØCL80
MSG1	DC	CL35' '
MSG2	DC	CL45' '
PRNTLINE	DS	ØCL255
WTIME	DS	CL8
	DS	CL1
WDATE	DS	CL7
	DS	CL1
WSSI	DS	CL4
	DS	CL1
W0258DN	DS	CL8
	DS	CL1
W0258TN	DS	CL8
	DS	CL1
W0258PQ	DS	CL8
	DS	CL1
W0258SQ	DS	CL8
	DS	CL1
W0258MS	DS	CL8
	DS	CL1
W0258HB	DS	CL8
	DS	CL1
W0258HA	DS	CL8
	DS	CL1
W0258XM	DS	CL8
	DS	CL1
W0258XB	DS	CL8
	DS	CL1
W0258XA	DS	CL8
	DS	CL1
W0258VM	DS	CL8
	DS	CL1
W0258VB	DS	CL8
	DS	CL1
W0258VA	DS	CL8
	DS	CL1
W0258DS	DS	CL44
W0258TS	DS	CL8

```
      DS      CL1
WØ258DB DS      CL8
      DS      CL1
WØ258PS DS      CL8
      DS      CL1
      DS      CL45
*      COPY THE DSECTS FROM DSNMACS
      DSNDQWAS DSECT=YES, SUBTYPE=Ø
      DSNDQWØ4
      END      I258
```

Venkat Pillay
DB2 DBA (USA)

© Xephon 2000

Fuzzy SELECT

Editor's note: although this article is independent of RDBMS, it is appropriate for DB2, as well as for Oracle7, Sybase, Ingres, etc.

OVERVIEW

This article provides techniques for 'fuzzifying' the DB2 SELECT statement by using 'fuzzy logic'. Fuzzy logic uses Gaussian or median distribution on existing data. Fuzzy SELECT allows decision support and end users to ask questions in their own way, get answers on incomplete data, and ask simple business questions.

BIVALENT LOGIC

Bivalent logic is the logic to which most people are referring when using the term 'logic'. It was devised by Democritus and Plato, later being codified by Aristotle. Aristotle's Law states that every statement or sentence is either true or false (ie A or not-A). Bivalent logic, combined with John von Neumann's symbolic theory, forms the basis for all computing.

TRIVALENT LOGIC

SQL is the first major software to deviate from bivalent logic by

<i>A</i>	<i>and</i>	<i>B</i>	<i>Evaluation</i>	<i>A</i>	<i>or</i>	<i>B</i>	<i>Evaluation</i>
true		true	true	true		true	true
true		false	false	true		false	true
true		null	unknown	true		null	true
false		true	false	false		true	true
false		false	false	false		false	false
false		null	false	false		null	unknown
null		true	unknown	null		true	true
null		false	false	null		false	unknown
null		null	unknown	null		null	unknown

Figure 1: Boolean evaluation table

introducing the null concept, modifying Aristotle’s Law to ‘A or not-A, or unknown’ whenever null participates in a predicate. ANSI has Boolean operators of AND/OR/NOT where:

- AND evaluates to true if all objects are true.
- OR evaluates to true if any object is true.
- NOT reverses true/false evaluation, but leaves unknowns unchanged.

Figure 1 shows a Boolean evaluation table. Note that:

- True and null is unknown because the null value may be updated to a true value, in which case the evaluation would be true.
- False and null is false because it does not matter what the null value is updated to.
- Null and null is unknown because null is not equal to anything, including another null.

- True or null is true because if any object is true then the evaluation is true.
- False or null is unknown because the null value may be updated to a true value, in which case the evaluation would be true.
- Null or null is unknown because either object may be updated to true.
- NOT(true) is false; NOT(false) is true; NOT(unknown) is still unknown.

‘WHERE ... IS NULL’ is the SQL method for reducing trivalent to bivalent because it evaluates true if null and false if not.

FUZZY OR MULTIVALENT LOGIC

Fuzzy logic has two meanings. The first is multivalent ‘vague’ logic, developed in the early 1900s. Vague logic states that everything is a matter of degree, including truth and set membership. This was expanded by Zadeh to fuzzy sets and rules. Zadeh’s Law is ‘A and not-A’; each statement is true to some degree.

Aristotle’s bivalent logic is expressed as ‘A or not-A’. Zadeh fuzzy

<i>Aristotle</i>	<i>Zadeh</i>
exact	partial
all or none	some
0 or 1	continuum between 0 and 1
digital computer	human brain – that is, a neural network
COBOL	natural language
bits	fits (Fuzzy unITS)

Figure 2: Aristotle versus Zadeh

logic is expressed as ‘A and not-A’. Other contradictions include those shown in Figure 2.

You should note that:

- Bits must be either 0 or 1, whereas fits can be any value between 0 and 1.
- Bits can represent only yes or no, whereas fits can represent degree of truth or falsity.

Fuzzy logic is the basis of ‘smart’ machines such as camcorders that can eliminate shakiness.

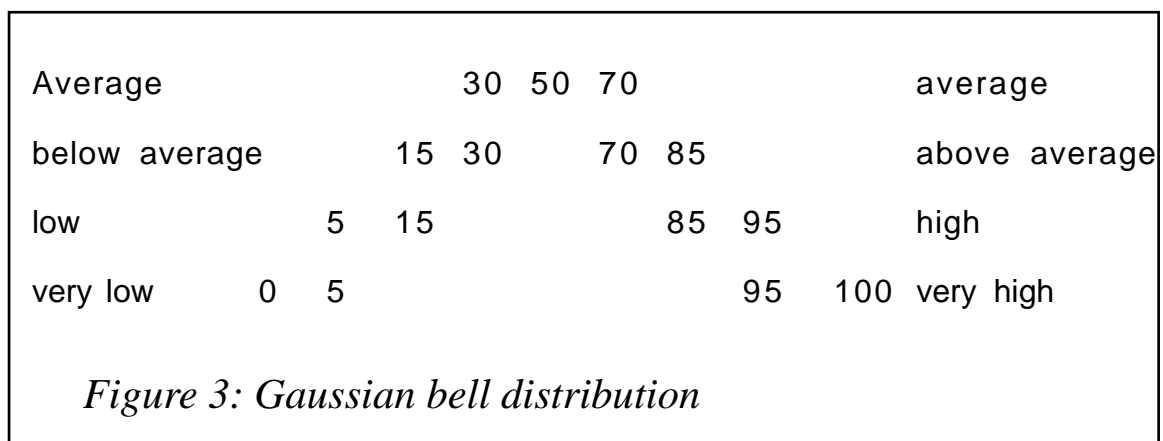
GAUSSIAN DISTRIBUTION

Karl Gauss developed the Gaussian distribution in the late 1700s, based on bell-shaped curves. This, and his other mathematical theorems such as least square curve fitting, allowed him to predict the return of Halley’s Comet. Gaussian distribution can be used to ‘fuzz’ existing data domains to provide SELECTs with fuzzy predicates.

A standard embedded SELECT provides a Gaussian distribution:

```
SELECT  AVG(column_tobe_fuzzed) * :guassian_value
INTO    :average
FROM    table_containing_column
```

Gaussian logic should be used only with bell curve data. Assuming an average (AVG) of 50, a bell curve dictates that half of the data points



<i>Range</i>	<i>User</i>	<i>Low</i>	<i>High</i>
very low	awful	0	5
low	not good	5	15
below avg	so so	15	30
average	alright	30	70
above avg	better	70	85
high	right on	85	95
very high	WOW	95	100

Figure 4: Gaussian fuzzy table

lie below AVG and the other half above. The Gaussian bell distribution is:

$$\begin{aligned}
 40\% \times \text{AVG}(50) &= 20 \\
 30\% \times \text{AVG}(50) &= 15 \\
 20\% \times \text{AVG}(50) &= 10 \\
 10\% \times \text{AVG}(50) &= 5
 \end{aligned}$$

The first 40% of data points lie within ± 20 of AVG (or 30 AVG 70). The next 30% of data points lie within the range 15 to 30 and 70 to 85, and so forth. This is shown in Figure 3.

It is a simple matter to convert Figure 3 into a relational table, as shown in Figure 4.

The RANGE column uses normal statistical terms, but could be anything. The USER column is the end user's words. There could be multiple USER columns selected by the USER option.

Business data is normally skewed, making Gaussian distribution unsuitable for generating fuzzy predicates. Median distribution accurately produces fuzzy predicates.

MEDIAN DISTRIBUTION

The median is the middle data point. Half of the data points fall below it while the other half lie above it. Median and mean (average) are the same in a bell curve distribution but can substantially differ when there is skewed data.

The median is based on the Pythagorean theorem of right-angled triangles, designating a line from the vertex to the middle of the opposite side. Mean accuracy is the difference between the 'true' value and the arithmetic mean (average). The formula is:

$$\text{Mean accuracy \%} = ((\text{median} - \text{average}) / \text{average}) * 100$$

For example, a 33% mean accuracy would show acutely skewed data (the skew is high when the average is greater than the median), denoting that Gaussian logic by itself should not be used. However, it is used to determine the statistical ranges (very low, low, below average, average, above average, high, very high) within the median distribution. Median requires a row COUNT within each range, as shown in Figure 5.

Gaussian logic has determined that the lowest 5% (very low) of the data points below the median have a range of 15 to 139, with 206 rows having those values. The next lowest 10% range is between 140 and 329, with 413 rows. The COUNT column allows 'around' and other such 'fuzzy' predicates. Assume that 'around' is defined as 5% of the requested value and that the end user wants 'around 40'. Then

```
SELECT      range, low, high, count
INTO        :range, :low, :high, :count
```

<i>range</i>	<i>user</i>	<i>low</i>	<i>range</i>	<i>count</i>
very low	awful	15	139	206
low	not good	140	329	413
...				

Figure 5: Median fuzzy table

```
FROM      median_fuzzy_table
WHERE     low >= 40
AND       high <= 40
```

finds the RANGE in which the desired value falls (very low). The host calculations are:

average incremental value per row = $(139 - 15) / 206$ or $.602$
around value = $(206 * 5%) * .602$ or 6.20

Q around 40 = 33.8 to 46.2

Using a straight 5% ($40 * 5\%$), would yield a range of 38 to 42 that would cause erroneous results (and probably some user concern because it does not feel right).

Other fuzzy predicates are 'above average', 'below average', 'high', 'very high', 'low', and 'very low'. SQL has AVG but not above/below; BETWEEN is not around; MAX is highest not high or very high; MIN is lowest not low or very low; etc.

Dynamic SQL programs can be coded to process fuzzy queries.

FUZZY BENEFITS

The benefits of fuzzy logic include:

- The use of standard statistical procedures to analyse actual data values to produce fuzzy tables to provide Gaussian or median distributions for end user queries. This is analogous to RUNSTATS providing statistics to the optimizer for better performance.
- It provides a foundation for using advanced mathematical theorems such as the Cauchy-Schwartz uncertainty algorithm or chaos theory.
- It allows users to use their own terminology, which is probably the greatest benefit.

The Japanese have made extensive use of fuzzy logic to gain competitive advantage. Perhaps others should take note and follow their lead.

Eric Garrigue Vesely
Principal Analyst
Workbench Consulting (Malaysia)

© Xephon 2000

Extracting catalog information – part 2

This month we continue the program that extracts catalog information from DB2 and generates DDL statements for DB2 objects.

```
/* db2_call */
db2_call:

hsts = 0
stat=msg('OFF')
free fi(sysprint)
free fi(sysin)
free fi(sysrec00)
free fi(syspunch)
stat=msg('ON')
alloc fi(sysrec00) space(100,100) block(4096)
alloc fi(sysprint) recfm(f b) lrecl(133) space(1,1) block(4096)
alloc fi(syspunch) recfm(f b) lrecl(80) space(1,1) block(4096)
alloc fi(sysin) recfm(f b) lrecl(80) blksize(80)
push ''
do i = s to 1 by -1
  push sel.i
end
execio * diskw sysin (finis
  if dbid = 'D'
    then do
      cmd = run program(dsntiaul) plan(DSNTIAUL)
      cmd = cmd || lib('ddsn.runlib.load')
      cmd = cmd || parms('SQL')
    end
  if dbid = 'T'
    then do
      cmd = run program(dsntiaul) plan(DSNTIAUL)
      cmd = cmd || lib('tdsn.runlib.load')
      cmd = cmd || parms('SQL')
    end
  if dbid = 'E'
    then do
      cmd = run program(dsntiaul) plan(DSNTIAUL)
      cmd = cmd || lib('edsn.runlib.load')
      cmd = cmd || parms('SQL')
    end
  if dbid = 'P'
    then do
```

```

                cmd = run program(dsntiaul) plan(DSNTIAUL)
                cmd = cmd || lib('pdsn.runlib.load')
                cmd = cmd || parms('SQL')
            end
queue ''
queue cmd
queue 'end '
if dbid = 'D' then
    'dsn system(dbd00)'
if dbid = 'T' then
    'dsn system(dbt00)'
if dbid = 'E' then
    'dsn system(dbe00)'
if dbid = 'P' then
    'dsn system(dbp00)'
if wflag = '1' then do
    execio * disk sysin      (stem errout.
    execio * disk sysin      (finis
    do i = 1 to errout.00
        say errout.i
    end
    execio * disk sysprint (stem errout.
    execio * disk sysprint (finis
    do i = 1 to errout.00
        say errout.i
    end
end
execio * disk sysrec00 (stem selout.
execio * disk sysrec00 (finis
do i = 1 to selout.00
    if wflag = '1' then say selout.i
    hsts = hsts + 1
    sel_out.hsts = selout.i
end
stat=msg('OFF')
free fi(sysprint)
free fi(sysin)
free fi(sysrec00)
free fi(syspunch)
stat=msg('ON')
return
/*****
/*   end_db2_call                               */
*****/
ddl_member_input:
/*****
/* create ddl member input values                */
*****/
/*****
/* this is tablespace section                    */
*****/

```

```

/*****
if ws_ddl_type = 'ts1'
then do
    wi = wi + 1
    ws_input.wi = ' '
    wi = wi + 1
    ws_input.wi = '- USER : ' || wuser || '   DATE: ' || wdate
    wi = wi + 1
    ws_input.wi = '   '
    wi = wi + 1
    ws_input.wi = '-DROP TABLESPACE ' || ts_dbname.ts || '.'
    ws_input.wi = ws_input.wi || ts_name.ts || ';'
    wi = wi + 1
    ws_input.wi = '-COMMIT;'
    wi = wi + 1
    ws_input.wi = '   '
    wi = wi + 1
    if ts_type.ts = 'L' then
    ws_input.wi = 'CREATE LARGE TABLESPACE '
    else
    ws_input.wi = 'CREATE TABLESPACE '
    ws_input.wi = ws_input.wi || ts_name.ts || ' IN ' || ts_dbname.ts
end
/*****
/* if tablespace not partitioned */
/*****
if ws_ddl_type = 'tsp'
then do
    if ts_part.ts = Ø then do
        wi = wi + 1
        ws_input.wi = '          USING STOGROUP ' || tsp_storname.tsp_num
        w_pqty = Ø
        w_sqty = Ø
        w_pqty = abs(tsp_pqty.tsp * 4 )
        w_sqty = abs(tsp_sqty.tsp * 4 )
        wi = wi + 1
        ws_input.wi = '          PRIQTY ' || w_pqty
        ws_input.wi = ws_input.wi || ' SECQTY ' || w_sqty
        ws_fpage = abs(tsp_freepage.tsp)
        ws_pfree = abs(tsp_pctfree.tsp)
        wi = wi + 1
        ws_input.wi = '          FREEPAGE ' || ws_fpage
        ws_input.wi = ws_input.wi || ' PCTFREE ' || ws_pfree
        if tsp_compress.tsp = 'Y' then do
            wi = wi + 1
            ws_input.wi = '          COMPRESS YES      '
        end
        wi = wi + 1
        if tsp_gbpcache.tsp = ' ' then
            ws_input.wi = '          GBPCACHE  CHANGED'

```

```

        else
            ws_input.wi = '          GBPCACHE ALL'
        end

/*****
/* if tablespace partitioned */
/*****
    if ts_part.ts > 0 then do
        wi = wi + 1
        ws_part = 0
        ws_part = abs(ts_part.ts)
        ws_input.wi = '          NUMPARTS ' || ws_part
        wi = wi + 1
        ws_input.wi = '          ('
        do tt = 1 to ws_part
            wi = wi + 1
            ws_input.wi = '          PART ' || tt
            wi = wi + 1
            ws_input.wi = '          USING STOGROUP '
            ws_input.wi = ws_input.wi || tsp_storname.tt
            w_pqty = 0
            w_sqty = 0
            w_pqty = abs(tsp_pqty.tt * 4 )
            w_sqty = abs(tsp_sqty.tt * 4 )
            wi = wi + 1
            ws_input.wi = '          PRIQTY ' || w_pqty
            ws_input.wi = ws_input.wi || ' SECQTY ' || w_sqty
            ws_fpage = abs(tsp_freepage.tt)
            ws_pfree = abs(tsp_pctfree.tt)
            wi = wi + 1
            ws_input.wi = '          FREEPAGE ' || ws_fpage
            ws_input.wi = ws_input.wi || ' PCTFREE ' || ws_pfree
            if tsp_compress.tt = 'Y' then do
                wi = wi + 1
                ws_input.wi = '          COMPRESS YES      '
            end
            wi = wi + 1
            if tsp_gbpcache.tt = ' ' then
                ws_input.wi = '          GBPCACHE CHANGED'
            else
                ws_input.wi = '          GBPCACHE ALL      '
            end
            wi = wi + 1
            if tt = ws_part then
                ws_input.wi = '          )'
            else
                ws_input.wi = '          ,'
            end
        end
    end
end
end
/*****/

```

```

/*                                                                 */
/*****
if ws_ddl_type = 'ts2'
then do
    if ts_segsize.ts > 0 then do
        wi = wi + 1
        ws_input.wi = '          SEGSIZE      ' || ts_segsize.ts
    end
    wi = wi + 1
    ws_input.wi = '          BUFFERPOOL ' || ts_bpool.ts
    wi = wi + 1
    if ts_lock.ts = 'A' then
        ws_input.wi = '          LOCKSIZE ANY  '
    if ts_lock.ts = 'P' then
        ws_input.wi = '          LOCKSIZE PAGE  '
    if ts_lock.ts = 'R' then
        ws_input.wi = '          LOCKSIZE ROW   '
    if ts_lock.ts = 'T' then
        ws_input.wi = '          LOCKSIZE TABLE'
    if ts_lock.ts = 'S' then
        ws_input.wi = '          LOCKSIZE TABLESPACE'
    wi = wi + 1
    if ts_close.ts = 'Y' then
        ws_input.wi = '          CLOSE YES      '
    else
        ws_input.wi = '          CLOSE NO      '
    wi = wi + 1
    if ts_lockpart.ts = 'Y' then
        ws_input.wi = '          LOCKPART YES;   '
    else
        ws_input.wi = '          ;               '
    wi = wi + 1
    ws_input.wi = '          '
    wi = wi + 1
    ws_input.wi = 'COMMIT;   '
    wi = wi + 1
    ws_input.wi = '          '
end
/*****
/* this is table section                                         */
/*****
if ws_ddl_type = 'tb'
then do
    wi = wi + 1
    ws_input.wi = '    CREATE TABLE ' || word(wtbcreator,1)
    ws_input.wi = ws_input.wi || '.' || wtbyname
    wi = wi + 1
    ws_input.wi = '          (          '
end

```

```

/*****
/* this is column section
/*****
if ws_ddl_type = 'cl'
  then do
    do i = 1 to cl_num
      if cl_keyseq.i > 0 then do
        ws_pri_flag = '1'
        ws_ks = ws_ks + 1
        ws_keys = abs(cl_keyseq.i)
        ws_pri.ws_keys = cl_name.i
      end
      wi = wi + 1
      ws_cl_name = '          ' || cl_name.i || ' '
      ws_input.wi = ws_cl_name || word(cl_coltype.i,1)
      if word(cl_coltype.i,1) = 'TIMESTMP' then
        ws_input.wi = ws_cl_name || 'TIMESTAMP'
      if word(cl_coltype.i,1) = 'LONGVAR' then
        ws_input.wi = ws_cl_name || 'LONG VARCHAR'
      if word(cl_coltype.i,1) = 'LONGVARG' then
        ws_input.wi = ws_cl_name || 'LONG VARGRAPHIC'
      ws_len = 0
      ws_len = abs(cl_length.i)
      ws_scale = 0
      ws_scale = abs(cl_scale.i)
      if cl_coltype.i = 'CHAR' then
        ws_input.wi = ws_input.wi || '(' || ws_len || ')'
      if cl_coltype.i = 'VARCHAR' then
        ws_input.wi = ws_input.wi || '(' || ws_len || ')'
      if cl_coltype.i = 'VARG' then
        ws_input.wi = ws_input.wi || '(' || ws_len || ')'
      if cl_coltype.i = 'DECIMAL ' then do
        ws_input.wi = ws_input.wi || '(' || ws_len || ','
        ws_input.wi = ws_input.wi || ws_scale || ')'
      end
      if cl_coltype.i = 'FLOAT' then
        ws_input.wi = ws_input.wi || '(' || ws_len || ')'
      if cl_coltype.i = 'LONGVAR ' then
        ws_input.wi = ws_input.wi || '(' || ws_len || ')'
      if cl_nulls.i = 'N' then
        ws_input.wi = ws_input.wi || ' NOT NULL '
      if cl_default.i = 'N' then
        ws_input.wi = ws_input.wi || ' WITH DEFAULT'
      if i < cl_num then
        ws_input.wi = ws_input.wi || ','
      else
        if ws_pri_flag = '1' then
          ws_input.wi = ws_input.wi || ','
        end
      *****/

```

```

/*  primary key section  */
/*****
  if ws_pri_flag = '1' then do
    wi = wi + 1
    ws_input.wi = '    PRIMARY KEY ('
    do i = 1 to ws_ks
      wi = wi + 1
      if i = ws_ks then do
        ws_input.wi = '                ' || ws_pri.i
        wi = wi + 1
        ws_input.wi = '                )'
      end
    else
      ws_input.wi = '                ' || ws_pri.i || ','
    end
  end
end
wi = wi + 1
ws_input.wi = '                )'
/*****
/*  foreign key section  */
/*****

if re_num > 0 then do
  wi = wi + 1
  ws_input.wi = ' '
end
do ref = 1 to re_num
  wfk_relname = re_relname.ref
  wpk_tbname   = translate(re_reftab.ref,'40'x,'00'x)
  wpk_creator  = translate(re_refcre.ref,'40'x,'00'x)
  call find_foreign_key
  wi = wi + 1
  ws_input.wi = ' FOREIGN KEY ' || re_relname.ref
  wi = wi + 1
  ws_input.wi = '                ('
  do i = 1 to fk_num
    wi = wi + 1
    if fk_num = i then
      ws_input.wi = '                ' || fk_colname.i || ','
    else
      ws_input.wi = '                ' || fk_colname.i
    end
  end
  wi = wi + 1
  ws_input.wi = '                )'
  wi = wi + 1
  ws_input.wi = ' REFERENCES ' || word(re_refcre.ref,1) || ',
                .' || word(re_reftab.ref,1)

  wi = wi + 1
  ws_input.wi = '                ('
  do i = 1 to pk_num

```

```

        wi = wi + 1
        if pk_num = i then
            ws_input.wi = '          ' || pk_colname.i || ','
        else
            ws_input.wi = '          ' || pk_colname.i
        end
        wi = wi + 1
        ws_input.wi = '          )'
        wi = wi + 1
        if re_delrule.ref = 'C' then
            ws_input.wi = ' ON DELETE CASCADE'
        if re_delrule.ref = 'N' then
            ws_input.wi = ' ON DELETE SET NULL'
        if re_delrule.ref = 'R' then
            ws_input.wi = ' ON DELETE RESTRICT'
        if re_delrule.ref = 'A' then
            ws_input.wi = ' ON DELETE NO ACTION'
        wi = wi + 1
        if ref = re_num then
            ws_input.wi = ', '
        else
            ws_input.wi = ' '
        end
        /*****
        /*
        /*****

wi = wi + 1
ws_input.wi = ' IN ' || ts_dbname.ts || '.' || ts_name.ts
if tb_edproc.tb > ' ' then do
    wi = wi + 1
    ws_input.wi = ' EDITPROC ' || tb_edproc.tb
end
if tb_valproc.tb > ' ' then do
    wi = wi + 1
    ws_input.wi = ' VALIDPROC ' || tb_valproc.tb
end
if tb_auditing.tb = 'A' then do
    wi = wi + 1
    ws_input.wi = ' AUDIT ALL '
end
if tb_auditing.tb = 'C' then do
    wi = wi + 1
    ws_input.wi = ' AUDIT CHANGE'
end
if tb_auditing.tb = ' ' then do
    wi = wi + 1
    ws_input.wi = ' AUDIT NONE '
end
if tb_clstype.tb = 'Y' then do

```

```

        wi = wi + 1
        ws_input.wi = '    WITH RESTRICT ON DROP'
    end
wi = wi + 1
if tb_datacap.tb = 'Y' then
    ws_input.wi = '    DATA CAPTURE CHANGES'
else
    ws_input.wi = '    DATA CAPTURE NONE '
wi = wi + 1
ws_input.wi = '    ;'
end

/*****
/* this is    index    section                                */
/*****/
if ws_ddl_type = 'ix1'
then do
    wi = wi + 1
    select
        when(ix_uniquerule.ix = 'P') then
            ws_input.wi = 'CREATE TYPE 2 UNIQUE INDEX '
        when(ix_uniquerule.ix = 'U') then
            ws_input.wi = 'CREATE TYPE 2 UNIQUE INDEX '
        when(ix_uniquerule.ix = 'C') then
            ws_input.wi = 'CREATE TYPE 2 UNIQUE INDEX '
        otherwise
            ws_input.wi = 'CREATE TYPE 2 INDEX '
    end
    wi = wi + 1
    ws_input.wi = '    ' || word(ix_creator.ix,1)
    ws_input.wi = ws_input.wi || '.' || word(ix_name.ix,1)
    wi = wi + 1
    ws_input.wi = '    ON ' || word(tb_creator.tb,1)
    ws_input.wi = ws_input.wi || '.' || word(tb_name.tb,1)
    wi = wi + 1
    ws_input.wi = '    ('
end
if ws_ddl_type = 'ixk'
then do
    do d = 1 to ixk_num
        wi = wi + 1
        if ixk_ordering.d = 'A' then ord = 'ASC'
            else ord='DESC'
        if d < ixk_num then
            ws_input.wi = '    ' || ixk_colname.d || ord || ','
        else
            ws_input.wi = '    ' || ixk_colname.d || ord
        end
        wi = wi + 1
        ws_input.wi = '    )'
    end
end

```

```

        if ix_clustering.ix = 'Y' then do
            wi = wi + 1
            ws_input.wi = '        CLUSTER '
        end
    end

end

/*****
/* if the index is not partition
/*****
if ws_ddl_type = 'ix2'
    then do
        if ixp_num = 1 then do
            wi = wi + 1
            ws_input.wi = '    USING STOGROUP ' || ixp_storname.ixp_num
            wi = wi + 1
            ws_input.wi = '        PRIQTY ' || abs(ixp_pqty.ixp_num * 4)
            ws_input.wi = ws_input.wi ||,
                ' SECQTY ' || abs(ixp_sqty.ixp_num * 4)

            wi = wi + 1
            ws_input.wi = '        FREEPAGE ' || abs(ixp_freepage.ixp_num)
            ws_input.wi = ws_input.wi ||,
                ' PCTFREE ' || abs(ixp_pctfree.ixp_num)

            wi = wi + 1
            ws_input.wi = '    GBPCACHE CHANGED '
        end
    end
/*****
/* if the index is partition
/*****
if ixp_num > 1 then do
    wi = wi + 1
    ws_input.wi = '    ('
    do tt = 1 to ixp_num
        wi = wi + 1
        ws_input.wi = '        PART ' || tt
        wi = wi + 1
        ws_limitkey = translate(tsp_limitkey.tt,'40'x,'FF'x)
        ws_input.wi = '        VALUES(' || word(ws_limitkey,1) || ')'
        wi = wi + 1
        ws_input.wi = '    USING STOGROUP ' || ixp_storname.tt
        wi = wi + 1
        ws_input.wi = '        PRIQTY ' || abs(ixp_pqty.tt * 4)
        ws_input.wi = ws_input.wi ||,
            ' SECQTY ' || abs(ixp_sqty.tt * 4)

        wi = wi + 1
        ws_input.wi = '        FREEPAGE ' || abs(ixp_freepage.tt)
        ws_input.wi = ws_input.wi ||,
            ' PCTFREE ' || abs(ixp_pctfree.tt)

        wi = wi + 1
        ws_input.wi = '        GBPCACHE CHANGED '
    end
    ws_input.wi = ')'
end

```

```

        if tt = ixp_num then
            ws_input.wi = '      )'
        else
            ws_input.wi = '      ,'
```

end

```

    end
end
end
/*****
/*
/*****
if ws_ddl_type = 'ix3'
    then do
        wi = wi + 1
        ws_input.wi = '    BUFFERPOOL ' || ix_bpool.ix
        wi = wi + 1
        ws_input.wi = '    CLOSE YES ;'
        wi = wi + 1
        ws_input.wi = '    '
```

end

```

/*****
/* this is    privilege    section    */
/*****
if ws_ddl_type = 'gr'
    then do
        wi = wi + 1
        ws_input.wi = ' COMMIT;    '
        wi = wi + 1
        ws_input.wi = '    '
```

do gg = 1 to tba_num

```

        wi = wi + 1
        ws_input.wi = 'GRANT '
        if tba_alter.gg = 'Y' then
            ws_input.wi = ws_input.wi || ' ALTER,'
        if tba_delete.gg = 'Y' then
            ws_input.wi = ws_input.wi || ' DELETE,'
        if tba_index.gg = 'Y' then
            ws_input.wi = ws_input.wi || ' INDEX,'
        if tba_insert.gg = 'Y' then
            ws_input.wi = ws_input.wi || ' INSERT,'
        if tba_select.gg = 'Y' then
            ws_input.wi = ws_input.wi || ' SELECT,'
        if tba_update.gg = 'Y' then
            ws_input.wi = ws_input.wi || ' UPDATE,'
        ws_length = length(ws_input.wi)
        ws_input.wi = substr(ws_input.wi,1,ws_length-1)
        wi = wi + 1
        ws_input.wi = 'ON TABLE ' || word(tb_creator.tb,1) || '.'
        ws_input.wi = ws_input.wi || word(tb_name.tb,1) || ' TO '
        ws_input.wi = ws_input.wi || tba_grantee.gg || ';'
    end
end

```

```

        wi = wi + 1
        ws_input.wi = ' '
    end
return
/*****
/*   cre_ddl_member
*****/
cre_ddl_member:
    w_mem = word(ts_name.ts,1)
    say w_mem '   member was created'
    alloc fi(ddlmem) da(syspdba.pd0.ddl(w_mem)) shr
    push ''
    do k = wi to 1 by -1
        ws_mem = translate(ws_input.k,'40'x,'00'x)
        push ws_mem
    end
    execio * diskw ddlmem (finis
    free fi(ddlmem)
return
/*****
/*   cre_index
*****/
cre_index:
    w_mem = 'A' || dbid || 'INDEX'
    say w_mem '   member is created '
    alloc fi(ddlmem) da(syspdba.pd0.ddl(w_mem)) shr
    push ''
    do k = rr to 1 by -1
        push ws_index.k
    end
    execio * diskw ddlmem (finis
    free fi(ddlmem)
return

```

PANEL

```

)PANEL
)ATTR
+ TYPE(TEXT)    INTENS(HIGH)  COLOR(WHITE)
? TYPE(TEXT)    INTENS(LOW)   COLOR(RED)    HILITE(REVERSE)
# TYPE(TEXT)    INTENS(HIGH)  COLOR(BLUE)   CAPS(OFF)
> TYPE(INPUT)   INTENS(LOW)   COLOR(YELLOW) HILITE(REVERSE)
Q TYPE(OUTPUT)  INTENS(HIGH)  COLOR(YELLOW)
)BODY
+          ?   CREATE DDL MEMBER FOR TABLESPACE          +User : Qwuser
+
+ #Database code.....: >d+          D:DBD0 T:DBT0 P:DBP0
+
+ #Tablespace name.....: >tsname1 +

```

+
+
+
+
+
+
+
+
+
+
+
+
+
+
+
+
+
+

+ PF3 : RETURN
)INIT
)END

ENTER : PROCESS

OUTPUT

```
- USER : XXXXXX   DATE: 20/12/99

-DROP TABLESPACE TSD0001.TSP0001;
-COMMIT;

CREATE TABLESPACE TSP0001 IN TSD0001
  NUMPARTS 10
  (
    PART 1
    USING STOGROUP SGP0001
    PRIQTY 36000 SECQTY 7200
    FREEPAGE 0 PCTFREE 5
    GBPCACHE CHANGED
    ,
    PART 2
    USING STOGROUP SGP0001
    PRIQTY 39600 SECQTY 7920
    FREEPAGE 0 PCTFREE 5
    GBPCACHE CHANGED
    ,
    PART 3
    USING STOGROUP SGP0001
    PRIQTY 37440 SECQTY 7488
    FREEPAGE 0 PCTFREE 5
    GBPCACHE CHANGED
    ,
    PART 4
    USING STOGROUP SGP0001
```

```

        PRIQTY 41760 SECQTY 8352
        FREEPAGE 0 PCTFREE 5
        GBPCACHE CHANGED
    ,
    PART 5
    USING STOGROUP SGP0001
    PRIQTY 40320 SECQTY 8064
    FREEPAGE 0 PCTFREE 5
    GBPCACHE CHANGED
    ,
    PART 6
    USING STOGROUP SGP0001
    PRIQTY 36720 SECQTY 7344
    FREEPAGE 0 PCTFREE 5
    GBPCACHE CHANGED
    ,
    PART 7
    USING STOGROUP SGP0001
    PRIQTY 37440 SECQTY 7488
    FREEPAGE 0 PCTFREE 5
    GBPCACHE CHANGED
    ,
    PART 8
    USING STOGROUP SGP0001
    ,
    PART 9
    USING STOGROUP SGP0001
    PRIQTY 33840 SECQTY 6768
    FREEPAGE 0 PCTFREE 5
    GBPCACHE CHANGED
    ,
    PART 10
    USING STOGROUP SGP0001
    PRIQTY 25920 SECQTY 5184
    FREEPAGE 0 PCTFREE 5
    GBPCACHE CHANGED
    )
    BUFFERPOOL BP2
    LOCKSIZE ANY
    CLOSE YES
;

COMMIT;

CREATE TABLE PTS1.CUST_ACC_UNIT
(

```

```

BRM_KOD          SMALLINT NOT NULL ,
MUDI_NO         CHAR(8) NOT NULL ,
MUST_NO        INTEGER NOT NULL WITH DEFAULT,
HT             CHAR(2) NOT NULL WITH DEFAULT,
MT             CHAR(2) NOT NULL WITH DEFAULT,
AK             CHAR(2) NOT NULL WITH DEFAULT,
DVZ_KOD        SMALLINT NOT NULL WITH DEFAULT,
BRM_KODM       SMALLINT NOT NULL WITH DEFAULT,
HSP_TIP_KOD    CHAR(2) NOT NULL WITH DEFAULT,
HSP_CZDN_NO    INTEGER NOT NULL WITH DEFAULT,
DEFK_NO        CHAR(8) NOT NULL WITH DEFAULT,
KUMULE_BORC    DECIMAL(17,2) NOT NULL WITH DEFAULT,
KUMULE_ALAC    DECIMAL(17,2) NOT NULL WITH DEFAULT,
ILERI_TAR_ISL_BKY DECIMAL(17,2) NOT NULL WITH DEFAULT,
REZERV_BKY     DECIMAL(17,2) NOT NULL WITH DEFAULT,
AH_LMT         DECIMAL(15,0) NOT NULL WITH DEFAULT,
HSP_DRM_KOD    CHAR(1) NOT NULL WITH DEFAULT,
PERS_MAAS_HSP_FLAG CHAR(1) NOT NULL WITH DEFAULT,
AH_KOD         CHAR(1) NOT NULL WITH DEFAULT,
SON_MHSB_TAR   DATE NOT NULL WITH DEFAULT,
MUS_MUNF_KOD   CHAR(1) NOT NULL WITH DEFAULT,
HSP_ACLS_TAR   DATE NOT NULL WITH DEFAULT,
GRNT_LMT_CEK_ADET SMALLINT NOT NULL WITH DEFAULT,
OZEL_HSP_FLAG  CHAR(1) NOT NULL WITH DEFAULT,
OZEL_FZ_KOD    CHAR(1) NOT NULL WITH DEFAULT,
EXTRE_FLAG     CHAR(1) NOT NULL WITH DEFAULT,
AH_TMNT_FLAG   CHAR(1) NOT NULL WITH DEFAULT,
AH_LMT_ART_FLAG CHAR(1) NOT NULL WITH DEFAULT,
EXTRE_ADRES_KOD CHAR(1) NOT NULL WITH DEFAULT,
AH_TNM_TAR     DATE NOT NULL WITH DEFAULT,
ACLS_OP_ID     CHAR(8) NOT NULL WITH DEFAULT,
GUNC_OP_ID     CHAR(8) NOT NULL WITH DEFAULT,
GUNC_TAR       DATE NOT NULL WITH DEFAULT,
ONAY_BEK_BKY   DECIMAL(17,2) NOT NULL WITH DEFAULT,
EXTRE_TUT      DECIMAL(15,0) NOT NULL WITH DEFAULT,
AH_LMT_GUNC_TAR DATE NOT NULL WITH DEFAULT,
PRIMARY KEY (
    BRM_KOD          ,
    MUDI_NO
)
)
IN TSD0001 .TSP0001
AUDIT NONE
DATA CAPTURE NONE
;
CREATE TYPE 2 UNIQUE INDEX
PIX1.IXP0001A
ON PTS1.CUST_ACC_UNIT
(
    BRM_KOD          ASC,

```

```

                MUDI_NO          ASC
            )
    CLUSTER
    (
        PART 1
        VALUES(117)
        USING STOGROUP SGP0001
        PRIQTY 4800 SECQTY 252
        FREEPAGE 0 PCTFREE 10
        GBPCACHE CHANGED

        ,

        PART 2
        VALUES(141)
        USING STOGROUP SGP0001
        PRIQTY 5232 SECQTY 252
        FREEPAGE 0 PCTFREE 10
        GBPCACHE CHANGED

        ,

        PART 3
        VALUES(190)
        USING STOGROUP SGP0001
        PRIQTY 4752 SECQTY 252
        FREEPAGE 0 PCTFREE 10
        GBPCACHE CHANGED

        ,

        PART 4
        VALUES(240)
        USING STOGROUP SGP0001
        PRIQTY 5568 SECQTY 252
        FREEPAGE 0 PCTFREE 10
        GBPCACHE CHANGED

        ,

        PART 5
        VALUES(345)
        USING STOGROUP SGP0001
        PRIQTY 5280 SECQTY 252
        FREEPAGE 0 PCTFREE 10
        GBPCACHE CHANGED

        ,

        PART 6
        VALUES(432)
        USING STOGROUP SGP0001
        PRIQTY 4848 SECQTY 252
        FREEPAGE 0 PCTFREE 10
        GBPCACHE CHANGED

        ,

        PART 7
        VALUES(523)
        USING STOGROUP SGP0001
        PRIQTY 4848 SECQTY 252

```

```
FREEPAGE 0 PCTFREE 10
GBPCACHE CHANGED
```

```
PART 8
VALUES(781)
USING STOGROUP SGP0001
PRIQTY 4848 SECQTY 252
FREEPAGE 0 PCTFREE 10
GBPCACHE CHANGED
```

```
PART 9
VALUES(899)
USING STOGROUP SGP0001
PRIQTY 4704 SECQTY 252
FREEPAGE 0 PCTFREE 10
GBPCACHE CHANGED
```

```
PART 10
VALUES(999)
USING STOGROUP SGP0001
PRIQTY 3312 SECQTY 252
FREEPAGE 0 PCTFREE 10
GBPCACHE CHANGED
```

```
)
BUFFERPOOL BP3
CLOSE YES ;
```

```
CREATE TYPE 2 INDEX
PIX1.IXP0001B
ON PTS1.CUST_ACC_UNIT
(
    DEFK_NO          ASC,
    HT               ASC,
    MT               ASC,
    AK               ASC,
    DVZ_KOD          ASC,
    BRM_KOD          ASC
)
USING STOGROUP SGP0001
    PRIQTY 20000 SECQTY 1000
    FREEPAGE 0 PCTFREE 10
GBPCACHE CHANGED
BUFFERPOOL BP3
CLOSE YES ;
```

```
CREATE TYPE 2 INDEX
PIX1.IXP0001C
ON PTS1.CUST_ACC_UNIT
(
    MUST_NO          ASC
```

```

        )
    USING STOGROUP SGP0001
        PRIQTY 28080 SECQTY 1000
        FREEPAGE 0 PCTFREE 10
    GBPCACHE CHANGED
    BUFFERPOOL BP3
    CLOSE YES ;

COMMIT;

GRANT ALTER,DELETE,INDEX,INSERT,SELECT,UPDATE
ON TABLE PTS1.CUST_ACC_UNIT          TO PRDUSRX      ;

```

Ali Ozturk

DBA

Pamukbank (Turkey)

© Xephon 2000

Optimizing tablespace and indexspace – part 1

The name of this REXX procedure is ATI. The ATI service helps you manage DB2 DASD, enabling you to generate a JCL stream that alters the space used by table and index spaces. This is beneficial because it will recapture space when spaces are over-allocated and it will expand spaces when extent limits have been reached. You can establish your own thresholds (via the ‘Option’ parameter shown in Figure 1) to determine when objects need resizing. The ATI service uses current catalog information and VSAM LISTCAT information to determine the current space requirements.

STARTING THE ATI PROCEDURE

Figure 1 shows the ATI Parameter entry panel. On the right of the panel, there is the Prompt column, which should help to explain which values to enter in the Parameter value column.

Further explanation of the input parameters follows:

- ‘SSID’ – the DB2 subsystem identifier.
- ‘Object’ – the object you want to calculate space for. Valid entries are TS (tablespace), IX (index space) or BOTH. The BOTH value is available if the Option field is empty.

```

Option ==> TS0 ATI
_____ ALTER Space - Parameter Entry _____
Command ==>

PARAMETER  PARAMETER VALUE                                PROMPT

SSID       => DSNN                                         DB2 Sub-System Identifier
Object     => BOTH                                         TS,IX or BOTH
Option     => __      _____                          E-Extent,0-Over Allocation
Creator    => NADI                                         Table Creator
Name       => _____                                    Table Name
Tspace     => _____                                    Tablespace Name
Dbname     => DBTEMP                                       Database Name
Icopy      => NONE                                         Before-After-Both-None
Copypref   => DSNN                                         Dataset PREFIX for ICs
Shrlevel   => NONE                                         None-Reference-Change
Log        => YES                                          Log option YES or NO
Sortdata   => YES                                          Sortdata YES or NO
Keepdict   => YES                                          Keepdictionary YES or NO
Runstats   => YES                                          Runstats tspace YES or NO
Volume     => MVSDB2 _____                          Volser
Tracks     => 15                                           Tracks per Cylinder - 15

Enter parameter values for the ALTER TS/IX space service !
PF3 Return

```

Figure 1: Parameter entry panel

- ‘Option’ – this has two fields. In the first field you can enter E (Extent) or O (Over allocation), or leave it blank. In the second field, you can enter numeric values.

For example:

```
Option => E      13      E-Extent,0-Over allocation
```

This option shows only those table/index spaces that have more than 13 extents.

```
Option => 0      50      E-Extent,0-Over allocation
```

This option shows only those table/index spaces that have more than 50% unused spaces.

- The next four fields are search criteria, which have to be applied to the DB2 catalog. One of them must be filled, and all fields support generic search (SQL-like format):
 - ‘Creator’ is the table creator
 - ‘Name’ is the table name
 - ‘Tspace’ is the tablespace name
 - ‘Dbname’ is the database name.
- ‘Icopy’ – this can be done before or after reorganization. Valid entries are also BOTH and NONE.
- ‘Copypref’ – in this field, you can enter the prefix dataset name of the image copy. If the value in the Icopy field is NONE, then the Copypref field should be empty.
- ‘Shrlevel’ – this field determines which kind of reorg operation you want to do. Accepted values are NONE, REFERENCE, or CHANGE.

If you use reorg with SHRLEVEL NONE, then applications will have read-only access during unloading and no access during reloading. The reorg with SHRLEVEL REFERENCE allows read-only access during most phases of organization, and SHRLEVEL CHANGE allows read-write access during most phases of reorg (on-line reorg). Before running reorg tablespace with SHRLEVEL CHANGE, you must create a mapping table and an index for it (see DDL statements).

- ‘Log’ – do you want to run the reorg with LOG YES or LOG NO? If you use SHRLEVEL REFERENCE or SHRLEVEL CHANGE, this must be LOG NO.
- ‘Sortdata’ – do you want to run the reorg with the SORTDATA attribute? This only applies if the tablespace has a clustered index. You can enter YES (generate SORTDATA) or NO (do not generate SORTDATA).
- ‘Keepdict’ – do you want to run the reorg with the KEEPDICTIONARY attribute? This applies only to tablespaces

that have been created with the COMPRESS YES option.

- ‘Runstats’ – do you want to run the Runstats after reorg is executed? Valid entries are YES and NO.
- ‘Volume’ – the disk volume name. Three volumes are available; if either one or all are empty, the procedure will use the default value (UNIT=SYSDA).
- ‘Tracks’ – the number of tracks per cylinder.

The Job Control skeleton uses SORTNUM 4 and SORTKEYS attributes. The first specifies the number of temporary datasets to be dynamically allocated by the sort program. The second specifies that index keys will be sorted in parallel with the reload and build phase to improve performance. This option is recommended if more than one index needs to be created.

The second panel of the ATI procedure shows the selection results (see Figure 2).

```
ALTER Space Service - Selection Result                               Row 31 to 40 of 83
Command ==>                                                         Scroll ==> PAGE
Press Enter to have this service continue.
Press End  to respecify your  PARAMETERS.

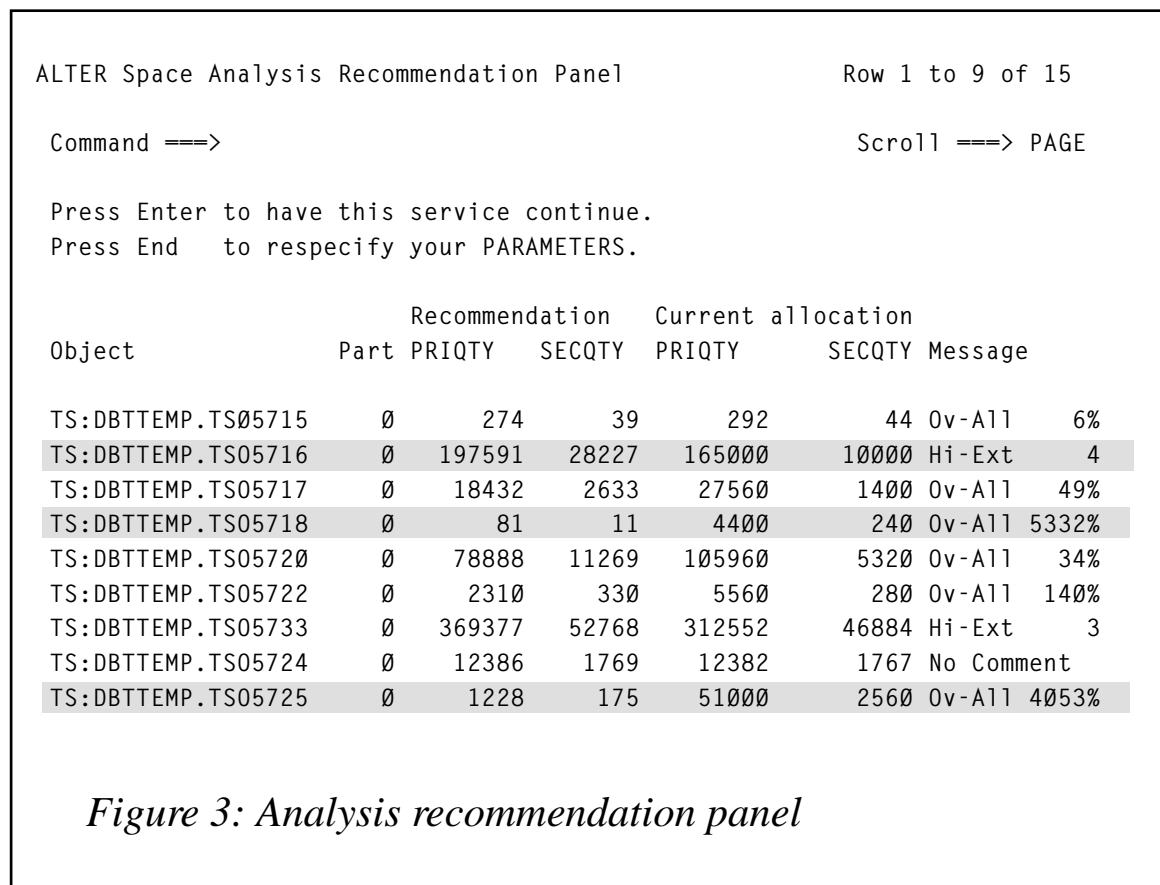
Dbname   Tsname   Table           Creator           Card  Message
DBTTEMP  TS05715  TL05715         NADI              6.870
DBTTEMP  TS05716  TL05716         NADI             1.924.967
DBTTEMP  TS05717  TL05717         NADI             411.252
DBTTEMP  TS05718  TL05718         NADI              -1  Runstat
DBTTEMP  TS05720  TL05720         NADI             1.164.425
DBTTEMP  TS05722  TL05722         NADI             54.474
DBTTEMP  TS05723  TL05723         NADI             4.953.808
DBTTEMP  TS05724  TL05724         NADI             174.492
DBTTEMP  TS05725  TL05725         NADI             161.808
DBTTEMP  TS05728  TL05728         NADI             833.000
```

Figure 2: The selection results panel

You can scroll the selection list in this panel using PF keys. If you want to confirm the selection, press Enter. If you want to re-specify your parameters, press the End key. When you have confirmed the selection results, the ATI procedure calculates the current space requirements and the necessary space requirements. When the calculation is finished, ATI shows the 'Alter Space Analysis Recommendation Panel' (see Figure 3).

Figure 3 shows only tablespace objects. In addition, you can see the Current allocation part and the Recommendation part. The first one shows the current allocation from the DB2 Catalog (PQTY and SQTY from SYSTABLEPART or SYSINDEXPART). The second one shows recommendation space from VSAMLISTCAT (data from HI-A-RBA and HI-U-RBA). The last part in Figure 3 is the Message column, which displays a short message. This column will display the following codes:

- 'No Comment' – the space is properly sized and does not need modification.



- ‘Ov-All n%’ – the space is over-allocated for ‘n’%.
- ‘Hi-Ext n’ – the space has ‘n’ extents.

The shaded rows in Figure 3 are candidates for reorganization. The first row has four extents and the remaining two shaded rows are over-allocated. The output JCL is in an ISPF editing session. You can edit, save, or submit this JCL.

An example of the output JCL (on-line reorg with CHANGE option) is shown below:

```
//SYSADMX JOB (#ACCT),' ',
//          NOTIFY=SYSADM,REGION=4M,
//          CLASS=A,MSGCLASS=X,MSGLEVEL=(1,1)
//RUNSQL EXEC PGM=IKJEFT01
//STEPLIB DD DISP=SHR,DSN=DSN510.SDSNLOAD
//          DD DISP=SHR,DSN=CEE.SCEERUN
//SYSTSPRT DD SYSOUT=*
//SYSTSIN DD *
DSN SYSTEM(DSNN)
RUN PROGRAM(DSNTEP2) PLAN(DSNTEP51) -
LIB('DSN510.RUNLIB.LOAD')
//SYSPRINT DD SYSOUT=*
//SYSUDUMP DD SYSOUT=*
//SYSIN DD *
SET CURRENT SQLID = 'SYSADM' ;
ALTER TABLESPACE DBTTEMP.TS05723
COMPRESS YES
PRIQTY 369377 SECQTY 52768 ;
- PRIQTY 312552 SECQTY 46884 ; ORIGINAL-VALUES
//*
/*-----
/*--- REORG -----
//REORG EXEC DSNUPROC,SYSTEM=DSNN,REGION=4096K,
//          UID='SYSADM.REORG',UTPROC=''
//STEPLIB DD DSN=DSN510.SDSNLOAD,DISP=SHR
//SYSREC DD UNIT=3390,VOL=SER=MVSDB2,
//          DSN=SYSADM.REORG.SYSREC.D0191285,
//          SPACE=(CYL,(50,50,)),RLSE,,ROUND),
//          DISP=(NEW,DELETE,CATLG)
//SORTOUT DD UNIT=SYSDA,
//          DSN=SYSADM.REORG.SORTOUT.D0191285,
//          SPACE=(CYL,(50,50,)),RLSE,,ROUND),
//          DISP=(NEW,DELETE,CATLG)
//SYSUT1 DD UNIT=SYSDA,
//          DSN=SYSADM.REORG.SYSUT1.D0191285,
//          SPACE=(CYL,(50,50,)),RLSE,,ROUND),
```

```
//      DISP=(NEW,DELETE,CATLG)
//COPY1 DD UNIT=3390,VOL=SER=MVSDB2,
//      DSN=DSNN.DBTTEMP.TS05723.D0180760,
//      DCB=(BUFNO=20,BLKSIZE=22528),
//      DISP=(,CATLG,DELETE),
//      SPACE=(TRK,(7516,375,),RLSE)
//SYSIN DD *
```

```
REORG  TABLESPACE DBTTEMP.TS05723
      LOG NO
      SORTKEYS
      SORTDEVT SYSDA SORTNUM 4
      SHRLEVEL CHANGE
      COPYDDN COPY1
      MAPPINGTABLE SYSADM.MAPPTB
      KEEPDICTIONARY
RUNSTATS TABLESPACE DBTTEMP.TS05723
/*
```

The ATI procedure does not support SHRLEVEL REFERENCE and CHANGE for a partition tablespace. In this case, you must work with SHRLEVEL NONE.

The DDL statements for mapping table and an index for it are:

```
– CREATE TABLESPACE FOR ON-LINE REORG
```

```
CREATE TABLESPACE MAPPTS IN database
      USING          STOGROUP stgname
                  PRIQTY 400
                  SECQTY 100
                  ERASE NO
                  FREEPAGE 0
                  PCTFREE 0
      BUFFERPOOL BP0
      LOCKSIZE TABLE
      CLOSE NO
      CCSID EBCDIC
      LOCKMAX 0
      SEGSIZE 4
      ;
COMMIT ;
```

```
– CREATE TABLE FOR ON-LINE REORG
```

```
CREATE TABLE SYSADM.MAPPTB
      (TYPE          CHAR(1)          NOT NULL
      ,SOURCE_RID    CHAR(5)          NOT NULL
```

```

, TARGET_XRID          CHAR(9)          NOT NULL WITH DEFAULT
, LRSN                 CHAR(6)          NOT NULL
) IN database.MAPPTS
CCSID EBCDIC
;
COMMIT ;

```

– CREATE INDEX FOR ON-LINE REORG

```

CREATE TYPE 2 UNIQUE INDEX SYSADM.MAPPIX
                ON SYSADM.MAPPTB
( SOURCE_RID ASC
, TYPE ASC
, TARGET_XRID ASC
, LRSN ASC
)
USING          STOGROUP stgname
                PRIQTY 400
                SECQTY 100
                ERASE NO
                FREEPAGE 0
                PCTFREE 0
BUFFERPOOL BP0
                CLOSE YES
                PIECESIZE 2097152 K
;
COMMIT ;

```

COMPONENTS OF ATI

ATI has the following components:

- ATI – the driver procedure
- ALTTSM – the main menu
- ALTTSL – the selection result panel
- ALTTSR – the space recommendation panel
- ALTMES – message display
- ALT00 – ATI message
- PALTTSI – PL/I source code
- ALTTSIX – JCL skeleton

This procedure works on MVS DB2 Version 5.

ATI

```
/* REXX */
/* Build an alter table - index space */
/* trace r */
zpfctl = 'OFF'
Y=MSG("OFF")
/*****/
/* Change to your convention standards */
program = 'PALTTSI'
plan     = 'PALTTSI'
llib    = 'SKUPNI.BATCH.LOADLIB'
/*****/
address ispexec 'vput (zpfctl) profile'
Call Alloc
cur='crec'
Call Create_messg
TOP:
address ispexec "display panel(ALTSM) cursor("CUR")"
if rc=8 then do
    Call Free_proc
    address ispexec "tbclose "messdb""
    exit
end
/* Check input parameters */
if op=' ' | op='E' | op='0' then nop
else do
    message='Invalid Option value. '||,
    'Valid values are: E-Extent, 0-Over allocation or blank.'
    Call Error 'op'
end
if (op='E' | op='0') & obj='BOTH'
then do
    message='Type TS or IX if option '||op||'.'
    Call Error 'obj'
end
if op='0' & pct<>' ' then do
    if datatype(pct,'N') <> 1
    then do
        message='Enter numeric data in %.'
        Call Error 'pct'
    end
end
if obj='TS' | obj='IX' | obj='BOTH' then nop
else do
    message='Invalid OBJECT value. '||,
    'Valid values are: TS-tablespace, IX-indexspace or BOTH.'
```

```

    Call Error 'obj'
end
if crec=' ' & tabc=' ' & tsnc=' ' & dbnc=' ' then do
    message='At least one Catalog search field must be entered.'
    Call Error 'crec'
end
if ico='BEFORE' | ico='AFTER' | ico='BOTH' | ico='NONE' then nop
else do
    message='Invalid Image Copy value. '||,
        'Valid values are: Before, After, Both or None.'
    Call Error 'ico'
end
if pref=' ' & ico ≠ 'NONE'
then do
    message='Invalid Copypref value. '||,
        'Dataset PREFIX for image copies required.'
    Call Error 'pref'
end
if rtype='NONE' | rtype='REFERENCE' | rtype='CHANGE' then nop
else do
    message='Invalid Reorg Shrlevel value. '||,
        'Valid values are: None, Reference, or Change.'
    Call Error 'rtype'
end
if rtype='REFERENCE' | rtype='CHANGE'
then do
    rshr='ON'
    ico = 'NONE'
    log = 'NO'
end
if log='YES' | log='NO' then nop
else do
    message='Invalid LOG parameter. Valid values are: YES, NO.'
    Call Error 'log'
end
if sor='YES' | sor='NO' then nop
else do
    message='Invalid SORTDATA parameter. Valid values are: YES, NO.'
    Call Error 'sor'
end
if dic='YES' | dic='NO' then nop
else do
    message='Invalid KEEPDICTIONARY parameter. '||,
        'Valid values are: YES, NO.'
    Call Error 'dic'
end
if rru='YES' | rru='NO' then nop
else do
    message='Invalid RUNSTATS parameter. '||,
        'Valid values are: YES, NO.'

```

```

    Call Error 'rru'
end
if trk=' '
then do
    message='Invalid Tracks value. '||,
            'Tracks per Cylinder required.'
    Call Error 'trk'
end
ind = verify(trk,'0123456789')
IF ind > 0 | trk < 1 then do
    message='Enter a valid NUMBER for Tracks parameter.'
    Call Error 'trk'
end
parm=substr(crec,1,8)||substr(tabc,1,18)||substr(tsnc,1,8)||,
      substr(dbnc,1,8)||substr(obj,1,4)
messg = "Accessing db2 system "db2""
messg = time() || " " || messg
Call Send_messg
messg = 'Select          systables          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 messsdb'
    Call Create_messg
    message = 'Error.  'db2||' ssid is not valid. |'
    Call Error 'db2'
end
"EXECIO * DISKR SYSPRINT (STEM ROW."
if substr(row.1,2) = 'DB2 CATALOG' then do
    Call Free_proc
    Call Aloc
    address ispexec 'tbend messsdb'
    Call Create_messg
    message='DB2 Catalog tables are excluded from ALTER processing.'
    Call Error 'crec'
end

```

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

*Bernard Zver
Database Administrator
Informatika Maribor (Slovenia)*

© Xephon 2000

DB2 news

Serena Software has announced plans to offer a DB2 option for its StarTool file and data management package, aimed at shortening development time and increasing productivity.

This move is expected by the company to extend the market for StarTool to developers of DB2-based applications and to large organizations needing a single product to manage the delivery of applications regardless of the underlying VSAM, DB2, or IMS database.

For further information contact:
Serena Software, 500 Airport Blvd, Second Floor, Burlingame, CA 94010-1904, USA.
Tel: (650) 696 1800.
URL: <http://www.serena.com>.

* * *

iE, formerly Intelligent Environments, has new releases of its iE Integrator (formerly Amazon Integrator) and iE ScreenSurfer (formerly ScreenSurfer) integration products, now combined into a single product suite.

It claims that over a hundred new features have been added, including integrated XML support, five times higher performance, better COM integration, support for Microsoft JScript and Visual BASIC, extended AS/400 support, mainframe 3270E terminal support, an extended security model, and improved database and communications support.

It's all aimed at providing enterprise applications with access to a wide range of legacy systems. The ScreenSurfer is for converting mainframe terminals into Web pages, while iE Integrator integrates

messaging, terminal, transaction, object, and database systems as well as publishing to multiple front ends.

Other new and updated features include better database integration, including DB2/ UDB static SQL, and SQL record-set caching, and claimed developer tools for SQL code generation.

For further information contact:
iE, 67 S Bedford St, Suite 401 E, Burlington, MA 01803-5152, USA.
Tel: (781) 272 9700.
URL: <http://www.ie.com>.

* * *

IBM has announced DB2 High Performance Unload, promising a range of unload functions for DB2 for OS/390 Version 3 or later for sites needing fast unload times and low CPU consumption.

It unloads from both active tablespaces and image copies and provides a set of management and control facilities for unload activities.

IBM also announced support for System/390 in Version 1.1 of its DB2 OLAP Server. The software gets the System/390 to perform analysis and carry out business intelligence functions.

The new version uses DB2 for OS/390 to maintain relational data, allowing the use of SQL programs and other relational tools to access and manage the data.

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



xephon