IMS User Exits for Data Tailoring

Outline

Intro

Assembler primer

General user exit conventions

Exit: Segment Edit (compression)

Exit: Segment Edit (unload/reload/reorg)

Exit: Index Maintenance

Exit: Partition Selection

User Exits Intro

Small programs written by the user and “plugged in” to IMS

Benefit: Provide extreme customizability…

Cost: …at the expense of manual coding (often assembler)

Benefit: compatibility between versions is usually very safe. (low maintenance once written)

Cost: Easy to forget about them!

Assembler Primer

CSECT – the program and its constants “literals” (read only)

DSECT – layout for an area of storage

D(B) address -- “Base-displacement” memory addresses. A “base” register and fixed value (0-4095) to add

Examples: 0(R1), 50(R2), 4095(R15)

Note: If you see a lot of these, you’re coding wrong (use DSECTs)

USING – statement to “map” a DSECT/CSECT to “base register”

Note: the 4095 limit means a single base register only addresses 4096 bytes.

Assembler Primer (continued)

In general, right-to-left

Common Instruction Types
RR instruction – register and register LR R1,R5
RX instruction – register and address L R1,0(R5)
SS instructions – Address(length) and address MVC 0(26,R1),0(R5)
SI instructions – address and immediate value (immediate values are part of the instruction itself, no need to address) MVI 0(R1),C'X'

Amode/Rmode – attributes of resulting load module
Amode – Indicates what addressing mode the program expects to be called in (assume 31 for exits)
Rmode – Indicates whether the load module requires to be loaded "below the line" (accessible by amode24)
Condition code – value set by MANY instructions, most notably "compare" instructions
Values are "equal", "low", "high", "Overflow"
Condition code set relative to the FIRST operand
CLC FIELD1,FIELD2
if FIELD1 > FIELD2 → CC="high"

Assembler Primer (continued)

Branch / Jump – based on condition code, goto somewhere else

Jump = “Branch Relative”, just like a branch, but no addressability needed

```
CLC  FIELD1,FIELD2
JH    PROG10          if FIELD1 > FIELD2, go to PROG10
CLC  FIELD1,FIELD2
JNE  PROG20           if FIELD1 not = FIELD2, go to PROG20
```

Golden Rules of Assembler Coding

-Follow familiar patterns… Example:

```
If (X > Y) THEN
    X := Y
ENDIF
```

```
          CLC   X,Y
          JNH   LABEL100
          MVC   X,Y
LABEL100  DS    0H
```

```
WHILE (ENTRY NOT= X) DO
    ENTRY = NEXTENTRY
ENDWHILE
```

```
          LA    R1,TABLE
LOOP100   DS    0H
          CLC   X,0(R1)
          JE    LOOP200
          LA    R1,4(R1)
          J     LOOP100
LOOP200   DS    0H
```

-COULD does not mean SHOULD. Example:

```
XC      FIELD1,FIELD2
XC      FIELD2,FIELD1
XC      FIELD1,FIELD2
```

```
MVC     TEMP,FIELD1
MVC     FIELD1,FIELD2
MVC     FIELD2,TEMP
```

```
ST      R1,TEMP
LR      R1,R2
L       R2,TEMP
```

User Exits (General)

Exits ALWAYS have:

1. Usage (what is it for?)
2. Communication Scheme
    a. Environment and information supplied to it
    b. Values expected from it
3. Restrictions
    a. What is allowed and forbidden
    b. Required attributes (Amode/Rmode/RENT)
4. Sample Exit (in <hlq>.ADFSSRC or <hlq>.ADFSSMPL)

Individual User Exits

1. Intro (overview and benefits)
2. Technical Details
3. Example walkthrough

Exit: Segment Edit / Compression

Overview

-Encode and Decode segments

-Specified in DBD at individual segment level

-Valid for Full-function and DEDB

-Should leave Key field unchanged (but doesn’t have to)

Benefits:

-Compression (reduced DASD size, smaller+faster logs)

Exit: Segment Edit / Compression

Technical info

-Specified on COMPRTN= on SEGM in DBD.

-SEGM …,COMPRTN=(*name*,DATA,INIT,*max*,PAD)

Exit Name

DATA fields only. (recommended)

INIT and TERM calls required

Max bytes the segment can increase by or

Min result size if PAD is present

Exit: Segment Edit / Compression

Technical Info (continued)

-Adding COMPRTN requires DBD change REORG

-Info passed in:

- -Partition Specification Table (PST)
- -Input segment
- -Buffer for output segment
- -Exit configuration info (DMBCPAC DSECT)
- -Physical Segment Description (DMBPSDB DSECT)
- -Function code …
  - …(Compress,Expand,Partial Expand,Open, or Close)

Exit: Segment Edit / Compression

-Result returned:

- -Output segment in supplied buffer (always VL)
- -never abend (true abend can abend CTL region)
  - -Return R0=abend code, R15=reason code instead
  - -Possible abend codes are 2989-2992
  - -IMS will Issue User abend to DEPENDENT region

Exit: Segment Edit / Compression

Input and result segments (DATA only)

	Input	Output
FIXED LENGTH	D1,KEY,D2	LL,D1,KEY,D2*
VARIABLE LENGTH	LL,D1,KEY,D2	LL*,D1,KEY,D2*

Exit: Segment Edit / Compression

```
COMPRESS DS    0H
         USING SEGM_IN,R2          MAP THE INPUT SEGMENT
         USING SEGM_OUT,R3         MAP THE OUTPUT SEGMENT
         MVC   OUT_PREKEY,IN_PREKEY  COPY . . .
         MVC   OUT_KEY,IN_KEY      . . . UNCOMPRESSABLE PART
         SR    R14,R14             TABLE INDEX
         LA    R15,FIELD1TABLE     FIRST ENTRY

LOOKUP_FIELD1 DS 0H
         CLC   IN_FIELD1,0(R15)    THIS ENTRY MATCH?
         JE    DONE_FIELD1
         AHI   R15,L'IN_FIELD1
         AHI   R14,1               NEXT INDEX
         J     LOOKUP_FIELD1
DONE_FIELD1 DS 0H
         STC   R14,OUT_FIELD1      STORE IN OUTPUT

         LA    R15,FIELD2TABLE
*     ...

FIELD1TABLE DC C'BELGIUM                    '
            DC C'CHINA                      '
            DC C'FRANCE                     '
            DC C'GERMANY                    '
            DC C'NETHERLANDS                '
*           ...
```

```
SEGM_IN   DSECT
IN_PREKEY DS CL64
IN_KEY    DS CL8
IN_FIELD1 DS CL30
IN_FIELD2 DS CL100
    . . .

SEGM_OUT  DSECT
OUT_LL     DS H
OUT_PREKEY DS CL64
OUT_KEY    DS CL8
OUT_FIELD1 DS CL1
OUT_FIELD2 DS CL1
   . . .
```

Exit: Reorg Exit

Overview:

Keep, Modify, or Delete segments during unload/reload/reorg

Deletes (segment+dependents) or (segment+remaining record)

Specified in USEREXIT(name) keyword on utility execution

Benefits:

Very fast and efficient way to manipulate segments en masse

Use to change field size/format, delete obsolete segments,etc

Note: changes are not given to Data Capture Exits

Exit: Reorg Exit

Technical Info:

Entry: R1 = Address of parameter list of addresses

offset 0 – Segment info block

offset 4 – Input segment

offset 8 – Output buffer

Return:

R15 = return code…

0 - Keep segment as-is

4 - Replace segment (Output buffer contains new segment)

12 - Delete segment and remaining record

24 - Delete segment and dependents

Exit: Reorg Exit Example

```
SEG_INFO DSECT
SEG_NAME DS    CL8
SEG_CODE DS    CL1

CHGSEG5  EQU   5
CHGSEG7  EQU   7

UX       CSECT
UX       AMODE 31
UX       RMODE ANY
         STM   R14,R12,12(R13)
         LR    R12,R15
         USING UX,R12
         L     R13,8(R13)
         LM    R2,R4,0(R1)     *SEG_INFO,INPUT,OUTPUT
         USING SEG_INFO,R2
         USING INPUT,R3
         USING OUTPUT,R4
         CLI   SEG_CODE,CHGSEG5
         JE    R_SEG5
         CLI   SEG_CODE,CHGSEG7
         JE    R_SEG7
         J     RETURN0
```

```
R_SEG5   DS    0H
         MVC   OUT_FIELD1,IN_FIELD1
         UNPK  OUT_FIELD2,IN_FIELD2
          . . .
         J     RETURN4

RETURN0  SR    R15,R15
         J     RETURN
RETURN4  LA    R15,4
         J     RETURN
RETURN   DS    0H
         L     R13,4(,R13)
         L     R14,12(,R13)
         LM    R0,R12,20(R13)
         BR    R14
```

Exit: Secondary Index Maintenance

Overview:

-Informally called a “Sparse” Routine

-Allows customized suppression of index segments

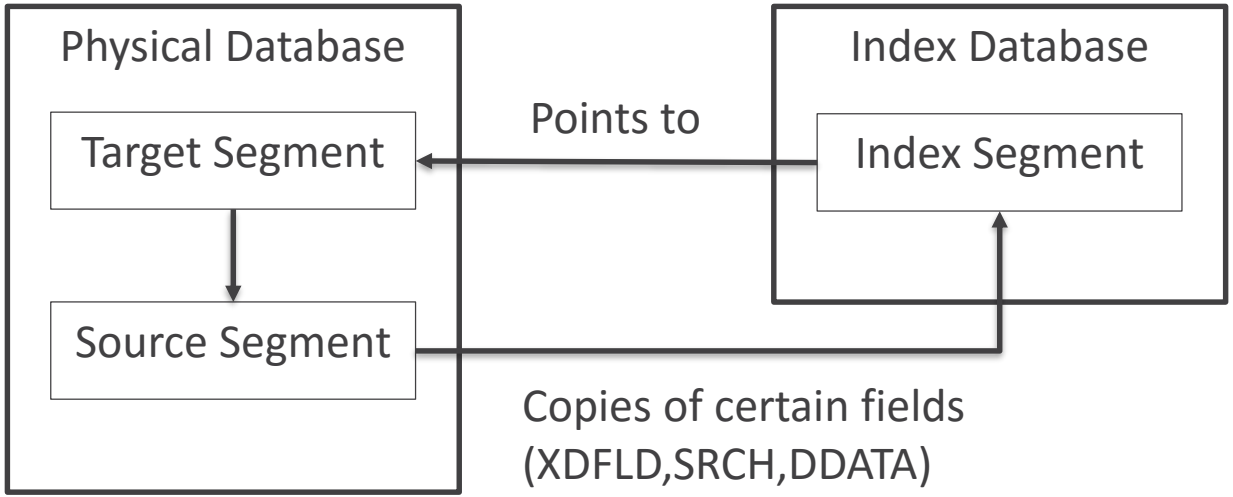
Instead or in addition to a NULLVAL parameter

Benefits:

-Optimizes Applications that process using secondary index but are interested in only a subset of the full database’s segments

-More flexibility than the NULLVAL option.

Secondary Index Review



Exit: Secondary Index Maintenance

Technical Info:

Called by IMS whenever update (ISRT/DLET/REPL) of index **source** segment

Function is the same regardless of update type

Entry:

-Index information (Indexed segment, XDFLD)

-Source segment

-Index segment (hypothetical)

Exit:

R15 = return code (0=keep, 4=suppress)

Exit: Index Sparse Example

```
CUSTOMER DSECT
CID      DS    CL8
CCOUNTRY DS    CL3
CBALANCE DS    F

PREFCUST` DSECT
CBCOUNTRY DS   CL3
CBID_LO   DS   CL8
CBID_HI   DS   CL8
CB_LEN    EQU  *-PREFCUST

SPARSE   CSECT
         STM   R14,R12,12(R13)
         LR    R12,R15
         USING SPARSE,R12
         L     R13,8(R13)
         USING CUSTOMER,R4     SUPPLIED ON ENTRY
         LA    R15,PREFCUST_TABLE
         USING PREFCUST,R15
LOOKUP   CLC   PCOUNTRY,CCOUNTRY    IN THIS COUNTRY?
         JE    FOUND
         AHI   R15,CB_LEN
         J     LOOKUP
FOUND    DS    0H
         CLC   CID,CBID_LO
         JL    RETURN4
         CLC   CID,CBID_HI
         JH    RETURN4
         J     RETURN0

PREFCUST_TABLE EQU *
         DC    CL3'USA'
         DC    CL8'12312312312'
         DC    CL8'24624624624'
         DC    CL3'GBA'
         DC    CL8'01010101010'
         DC    CL8'02020202020'
          . . .
```

Exit: Partition Selection

Overview

-Define the partitioning scheme for a HALDB

    -Instead of pure high-keys

    -Map root key to partition number

    -Order partition numbers for sequential processing

Benefits:

-More flexibility than high key schemes

Exit: Partition Selection

Technical info:

Call types:

Init – given single haldb info, build persistent areas

Term – clean up any built areas

Re-init – haldb info changed, re-build anything necessary (often unused)

Get-First – return first partition number

Get-next – given part number, return next part num (or end)

Get-Target – given root key, return partition number

Exit: Partition Selection

Info passed in: (DFSPSEIB macro)

-Communication area (DFSPECA dsect)

    -function code, entry parameter(s),return fields

-Overall Partitioning info (DFSPDA dsect)

    -number of partitions, longest key string length

    - 5 user fields persistent across calls (PDAUSR1 – PDAUSR5)

-Individual partition info (DFSPDAE dsect)

    -1 copy per defined partition

    -Partition name, keystring, number

Exit: Partition Selection

Return values

-PECRC : return code (all calls)

-PECFDB2 : partition number (first,next,target)

Exit: Partition Selection Example

```
WORKAREA DSECT
WS_PNUM  DS  H
WS_RC    DS  F
   . . .
WS_LEN   EQU *-WORKAREA

PSEINIT  DS    0H
         GETMAIN RC,LV=WS_LEN   ALLOCATE WORK AREA
         USING WORKAREA,R1
         XC    WORKAREA(WS_LEN),WORKAREA   CLEAR IT
         ST    R1,PDAUSR1
         DROP  R1
         J     RETURN0

PSETERM  DS    0H
         L     R1,PDAUSR1
         FREEMAIN RC,A=(R1),LV=WS_LEN
         J     RETURN0

PSERBLD  DS    0H
         J     RETURN0
```

```
RETURN0  XC    PECRC,PECRC
         J     RETURN
RETURN8  LA    R0,8
         ST    R0,PECRC
         J     RETURN
RETURN   DS    0H
         L     R13,4(,R13)
         LM    R14,R12,12(R13)
         BR    R14
```

Exit: Partition Selection Example

```
PENTRY   DSECT
P_NUM    DS    H
P_CTRY   DS    CL4
P_LEN    EQU   *-PENTRY

PNUMS    DC    H'1',CL4'ARG'
         DC    H'2',CL4'AUS'
         DC    H'4',CL4'BEL'
         DC    H'5',CL4'CAN'
         DC    H'8',CL4'DEU'
         DC    H'6',CL4'DNK'
         DC    H'7',CL4'GRL'
         DC    H'3',CL4'JPN'
         DC    XL2'FFFF'

PSEFIRST DS    0H
         MVC   PECFDB2,PNUMS
         J     RETURN0
```

```
PSENEXT  DS    0H
         LA    R15,PNUMS
         USING PENTRY,R15
PSEN100  CLC   PECCPID,P_NUM    FIND INPUT PART
         JE    PSEN200
         AHI   R15,P_LEN
         J     PSEN100
PSEN200  DS    0H               FOUND IT.
         AHI   R15,P_LEN        NEXT ENTRY
         CLI   P_NUM,X'FF'      NO MORE?
         JE    RETURN0          RETURN "END"
         MVC   PECFDB2,P_NUM
         DROP  R15
         J     RETURN12         RETURN "NEXT"
```

Exit: Partition Selection Example

```
ROOTKEY  DSECT
         DS    CL6
RK_CTRY  DS    CL4


PENTRY   DSECT
P_NUM    DS    H
P_CTRY   DS    CL4
P_LEN    EQU   *-PENTRY
```

```
PSETRGT  DS    0H
         LA    R15,PNUMS
         USING PENTRY,R15
         L     R14,PECKEY        INPUT KEY
         USING ROOTKEY,R14

PSET100  DS    0H
         CLI   P_NUM,X'FF'       END OF TABLE?
         JE    RETURN8           RETURN "NOT FOUND"
         CLC   P_CTRY,RK_CTRY    MATCH?
         JE    PSET200           YES
         AHI   R15,P_LEN
         J     PSET100
PSET200  DS    0H                FOUND IT.
         MVC   PECFDB2,P_NUM     SET RETURN PART
         J     RETURN0           RETURN "FOUND"
```

Recap

User exits:

-Fairly small

-Have a well-defined task and environment

-Powerful tools for customization

-low cost to maintain once built

-not THAT scary

Thank You

Bring IT to Life.™