F02-08-86

GARBAGE

- PHYSICALLY REMOVES LOGICALLY DELETED RECORDS
- CAN BE USED ON MASTER OR KSAM RD'S
- ALLOWS DATA BASE MANAGER TO CHANGE SIZE OF PHYSICAL FILES
- WRITES RECORDS IN ISN (OR KEY) SEQUENCE TO A WORK FILE CALLED TMASTER

*** NOTE: TMASTER DEFAULTS TO TAPE

- 3 BASIC FUNCTIONS:

1) STORE DB --> TMASTER
2) RESTORE TMASTER --> DB
3) REORG DB --> TMASTER --> DB

GARBAGE - MASTER

- RECOVERS WASTE SPACE IN MASTER FILE
- REMOVES LOGICALLY DELETED RECORDS
- OPTIONALLY EXTENDS LIMIT OF MASTER FILE AND/OR XREF FILE

- PRINTS LIST OF UNALLOCATED ISNS ON OUTPUT FILE GARBG

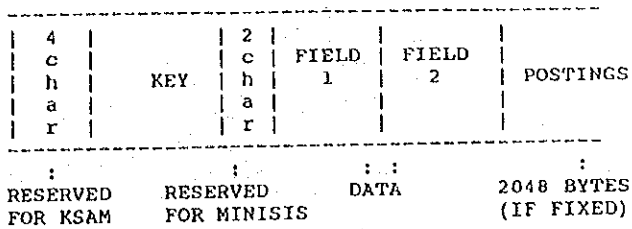
1.

F02-08-86

GARBAGE - KSAM

- REMOVES LOGICALLY DELETED RECORDS
- OPTIONALLY EXTENDS LIMIT OF KSAM FILE

- IF INVERTED FILE:
- OPTIONALLY RESETS POSTINGS TO ZERO
- OPTIONALLY MAKES POSTINGS AREA FIXED OR VARIABLE LENGTH
- IF POSTINGS ARE VARIABLE, WILL RECOVER WASTE SPACE IN POSTINGS AREA



2.

F02-08-86

GARBAGE - COMMANDS

ENTER DATA BASE NAME OR EXIT
- ENTER THE NAME OF THE RD

ENTER OPTION (STORE/RESTORE/REORG)

STORE
- WRITES DATA FILE TO TMASTER
- DISPLAYS NO. OF RECORDS STORED
  NO. OF RECORDS DELETED
  MAX, MIN AND AVG RECORD LENGTHS
  MAX, MIN AND AVG NO. OF FIELDS/RECORD
- TMASTER IS SAVED
- DATA FILES REMAIN UNCHANGED

RESTORE
- LOADS DATA FROM TMASTER
- DISPLAYS NO. OF RECORDS RESTORED
  MAX, MIN AND AVG RECORD LENGTHS
  MAX, MIN AND AVG NO. OF FIELDS/RECORD

F02-08-86

RESTORE (continued)

IF MASTER
ENTER SIZE OF MASTER FILE
ENTER SIZE OF XREF FILE

IF KSAM
INITIALIZE BITSTRING (Y/N)
ENTER SIZE OF KSAM DATA FILE
RECORD FORMAT (V/F)

CR LEAVES ORIGINAL VALUES UNCHANGED.

IF FILE SIZE IS CHANGED, FILES WILL BE PURGED AND REBUILT.

REORG
- COMBINES STORE AND RESTORE
- IF TMASTER IS A DISC FILE IT IS NOT SAVED UNLESS GARBAGE ENCOUNTERS AN ERROR

BOTH RESTORE AND REORG LIST UNUSED ISNS.

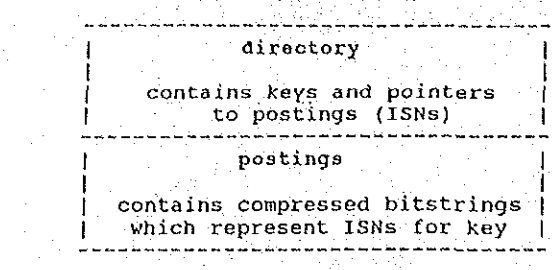
** DATA BASE CAN'T BE ACCESSED DURING RESTORE OR REORG **

5.

TREEMANT (B-Tree maintenance)

- TREEMANT IS A FEATURE OF INVERT
- USED TO CHANGE THE SIZE OF A B-TREE FILE
- USED TO RECOVER WASTE SPACE IN B-TREE FILE
- FOR REG, LEFT AND DOT B-TREE FILES
- NOT FOR ADJ OR LEFTADJ B-TREE FILES

B-TREE FILE STRUCTURE:



- EACH RECORD IS 2048 BYTES LONG
- CONTAINS 'N' UNIQUE KEYS
- N=2042-(L+P)/(L+P)*C
  - L=LENGTH OF KEY
  - P=6 OR 10-BYTE POINTER TO POSTINGS
  - C=NO. OF DIRECTORY RECORDS

RUNNING TREEMANT

TREEMANT ASKS FOR THE BTREE ROOT NAME

THE FIRST PROMPT IS
RECOVER WASTE SPACE
IF YOU REPLY NO, THE FOLLOWING
INFORMATION IS DISPLAYED

MAXIMUM KEY LENGTH=L
POTENTIAL NO. OF KEYS IN DIRECTORY=A
CURRENT NO. OF KEYS IN DIRECTORY=B
TOTAL DIRECTORY RECORDS=C
TOTAL USED DIRECTORY RECORDS=D
AVAILABLE POSTINGS RECORDS=E
USED POSTINGS RECORDS=F

C-D=NUMBER OF FREE DIRECTORY RECORDS AVAILABLE
A/C=NO. OF KEYS PER PHYSICAL DIRECTORY RECORD
F+E=TOTAL NUMBER OF PHYSICAL RECORDS ALLOCATED TO POSTINGS
C+E+F=TOTAL NUMBER OF PHYSICAL RECORDS IN THE BTREE FILE

IF D APPROACHES C OR E APPROACHES 0,
IT IS TIME TO RECOVER WASTE SPACE AND
POSSIBLY CHANGE THE SIZE OF THE FILE

RUNNING TREEMANT (continued)

IF YOU REPLY Y TO RECOVER WASTE SPACE

THE NEXT PROMPT WILL BE
INCREASE SIZE OF B-TREE FILE
IF Y, THE NEXT PROMPT WILL BE
ENTER MAXIMUM SIZE OF B-TREE FILE
THE NEW MAXIMUM NUMBER OF PHYSICAL
RECORDS SHOULD BE ENTERED

THE B-TREE STRUCTURE INFORMATION
WILL BE DISPLAYED

THE NEXT PROMPTS WILL BE
CHANGE SIZE OF DIRECTORY
IF Y, THE NEXT PROMPT WILL BE
NO. OF FREE DIRECTORY RECORDS AFTER REORG
ENTER THE NUMBER OF UNUSED PHYSICAL
DIRECTORY RECORDS REQUIRED

CHANGE SIZE OF POSTINGS AREA
IF Y, THE NEXT PROMPT WILL BE
NO. OF FREE POSTINGS RECORDS AFTER REORG
ENTER THE NUMBER OF UNUSED PHYSICAL RECORDS
REQUIRED

RUNNING TREEMANT (continued)

THE GARBAGE COLLECTION NOW TAKES PLACE.

- IF YOU REPLY Y TO RECOVER WASTE SPACE, GARBAGE COLLECTION IS DONE ON POSTINGS AREA
- WHEN COMPLETED, THE FOLLOWING MESSAGE APPEARS

  TOTAL USED POSTINGS RECORDS AFTER REORG=no.

- IF YOU REPLY "Y,DELETE=Y" TO RECOVER WASTE SPACE GARBAGE COLLECTION IS DONE ON DIRECTORY
- KEYS WITH ZERO POSTINGS ARE DELETED
- WHEN COMPLETED, SORT STATISTICS ARE DISPLAYED AND THE FOLLOWING MESSAGE APPEARS:

  NUMBER OF KEYS DELETED=no.

- DIRECTORY SIZE REMAINS FIXED UNLESS YOU REPLY Y TO CHANGE DIRECTORY SIZE

TREEMANT IS EXECUTED ON A COPY OF THE BTREE
NAMED NEWname (4-CHAR. ROOT).
WHEN TREEMANT SUCCESSFULLY EXECUTES, THE
ORIGINAL B-TREE FILE IS PURGED AND THE
NEW FILE IS RENAMED WITH THE ORIGINAL NAME.

F02-08-86

LISTDDT and LISTFORMAT

LISTDDT

- DISPLAYS A DATA BASE DEFINITION (FIELD ATTRIBUTES ONLY)
- REQUESTS
  - DATA BASE NAME
  - SORTING ?
    - IF YES - BY NAME
      TAG
      MNEMONIC
  - NUMBER OF COPIES
- SENDS OUTPUT TO FILE LPFILE (DEFAULTS TO LINE PRINTER)

LISTFORMAT

- LISTS FIELD ENTRIES IN A PRINT FORMAT, AND THEIR ATTRIBUTES
- SENDS OUTPUT TO FILE LFRMT (DEFAULTS TO LINE PRINTER)

F02-08-86

LISTDDT and LISTFORMAT (continued)

LISTDDT ASSUMES THAT A DATA BASE DEFINITION IS A USER-DEFINED STRUCTURE WITH ITS OWN DATA BASE DEFINITION "LISTDDT" (NO PHYSICAL FILES)

- INFORMATION FROM DD FILE IS DISPLAYED BY A PRINT FORMAT FILE
- DEFAULT FORMAT IS PLDDT00 - ENGLISH
  PLDDT01 - FRENCH
  PLDDT02 - SPANISH

LISTFORMAT ASSUMES THAT PRINT FORMAT FILE IS A USER-DEFINED STRUCTURE WITH DATA BASE "LISTFRMT"

- INFORMATION FROM PRINT FORMAT IS DISPLAYED BY A PRINT FORMAT FILE
- DEFAULT FORMAT IS PFRMT00 - ENGLISH
  PFRMT01 - FRENCH
  PFRMT02 - SPANISH

F02-08-86

LISTDDT and LISTFORMAT (continued)

MODIFYING PLDDTnn AND PFRMTnn FILES

- PLDDTnn AND PFRMTnn FILES ARE DISTRIBUTED WITH MINISIS
- THESE PRINT FORMAT FILES CAN BE MODIFIED USING THE PRINT PROCESSOR
- LISTDDT AND LISTFRMT DATA BASES HAVE SPECIAL SYSCHEMA AND DD FILES DISTRIBUTED WITH MINISIS
- SYSCHEMA FILE IS USERSCHM
- DD FILE IS DDUSER
- TO MODIFY PRINT FORMAT FILES FOR THESE SPECIAL DATA BASES:

```
:FILE SYSCHEMA.PUB=USERSCHM.group.account
:FILE DDUSER.PUB=DDUSER.group.account
```

F02-08-86

FIXXREF

- CHANGES CONTENTS OF AN XREF RECORD
- OPTIONALLY DUMPS MASTER RECORD ON PRINTER

*** SHOULD BE USED ONLY AS A LAST RESORT BY EXPERIENCED MINISIS DATA BASE MANAGER

```
PROMPTS
    XREF FILE NAME
    DISPLAY RECORD ON LP (Y/N)
IF Y THEN MASTER FILE NAME
    ENTER ISN OR END
    CHANGE XREF (Y/N)
IF Y THEN
    RECORD NUMBER OF MASTER FILE
    OFFSET
```

*** CR WILL DELETE THE XREF RECORD

RECOVERY

- USED WHEN ENTRY OR MODIFY TRANSACTIONS ARE LOGGED TO AN MPE LOG FILE
- WHEN DATA IS LOST, RESTORE DATA FILES FROM LAST BACK-UP
- RUN RECOVERY TO WRITE CONTENTS OF LOG FILE TO DATA BASE

F02-08-86

ERROR MESSAGES

3 BASIC TYPES OF ERROR MESSAGES

1) PROGRAM ABORT MESSAGE

```
ABORT.PROCESSOR NAME % LOCATION
EX:  PRINT.PUB.MINISIS.%1633.%17:PUSL.%1
```

2) HP FILE ERROR

- USUALLY PRODUCES A TOMBSTONE

```
+---------------------+
| file name is XXXX   |
|        .            |
|        .            |
| error no. XXX       |
+---------------------+
```

- COMMON ERRORS ARE
  - 0 - END OF FILE
  - 46 - OUT OF DISC SPACE
  - 50 - NON-EXISTENT ACCOUNT
  - 93 - SECURITY VIOLATION
  - 100 - DUPLICATE FILENAME

3) MINISIS ERROR MESSAGES

- MESSAGES COME FROM MESSnn and ERRnn
- SPECIFY NAME OF PROCESSOR
- SOME ERRORS ALLOW CONTINUATION OF FUNCTION
- OTHERS DO NOT

13.

F02-08-86

MINISIS MESSAGE FILES

- MINISIS-USER DIALOGUE CONTROLLED BY

```
MESSnn.PUB.MINISIS   (system messages)
ERRnn.PUB.MINISIS    (error messages)
SYNnn.PUB.MINISIS    (user syntax)
```

  WHERE nn IS A 2 DIGIT NUMBER (00 - 99)

- FILES MAINTAINED IN EDITOR FORMAT

- EACH LANGUAGE VERSION MAY HAVE A DIFFERENT NUMBER SUFFIX
  EG. ENGLISH MESS00, ERR00, SYN00
  FRENCH MESS01, ERR01, SYN01
  SPANISH MESS02, ERR02, SYN02
- DEFAULT IS ERR00, MESS00, SYN00

- FOR DIFFERENT SET
  : RUN MINISIS.PUB.MINISIS;LIB=P;PARM=nn

14.

F02-08-86

MESSnn.PUB.MINISIS

- EDITOR FORMAT
- MESSAGES GROUPED BY PROCESSOR
- EACH MESSAGE TERMINATED BY "\"
- CONTAINS SOME DEFAULT PARAMETERS
- MAY BE MODIFIED BY USER

ERRnn.PUB.MINISIS

- EDITOR FORMAT
- EACH MESSAGE TERMINATED BY "\"
- MESSAGES GROUPED BY PROCESSOR
- 6 CHARACTERS AT BEGINNING OF MESSAGE CAUSE CERTAIN ACTIONS
  POSITION 0 - abort program
  1 - invoke debug if program online
  2 - invoke stack dump
  3 - write to log file
  4 - display file tombstone
  5 - display MPE file error message

- 2 SYMBOLS BEFORE MESSAGE INDICATE:
  ** - USER-RECOVERABLE ERROR OR WARNING
  ?? - NOT IMMEDIATELY RECOVERABLE - CALL DATA BASE MANAGER
  $$ - CATASTROPHIC ERROR (MINISIS OR MPE)

15.

F02-08-86

SYNnn.PUB.MINISIS

- LEGAL USER COMMANDS FOR EACH PROCESSOR

- USER COMMANDS CHECKED AGAINST FILE
- IF NOT FOUND, **syntax error IS RETURNED

- NOT IN EDITOR FORMAT
- GENERATED BY SYNCOMP UTILITY FROM SYNnnT.UTILITY.MINISIS

  SYNnnT --> SYNCOMP --> SYNnn

MINEDIT and SYNCOMP

- OFFER EDITING CAPABILITIES FOR MESSAGE FILES IN NON-ROMAN CHARACTER SETS

16.

F02-08-86

SOME MESSnn PARAMETERS WHICH MAY BE MODIFIED

1350/1388 - USED TO GENERATE MINISIS MENU AND USER SECURITY PROFILES

343 - HEADER FOR QUERY OUTPUT

230, 364, 509, 717 - JOB CARDS FOR BATCH JOBS FROM QUERY, COMPUTE, MODIFY, INDEX

232, 367, 418, 719 - LOCATION OF MINISIS PROGRAM FOR BATCH JOBS

166 AND 169/170 - DEFAULT VALUES FOR CHAR./LINE, LINES/PAGE, MAX. NO. OF CHAR. IN PRINT

171 - DEFAULT OUTPUT DEVICE FOR PRINT

418 AND 452 - QUERY DEFAULT VALUES FOR RECLIMIT, DSPLIMIT

870/885 - ACCESS PARAMETER NAMES FOR MENU

831/834 - MAXIMUM EDS SIZES

2350/2372 - MASKS FOR VERSION AND PROCESSOR DATESTAMP

17.

F02-08-86

THE MINISIS PROGRAM

- MINISIS PRESENTS A MENU OF PROCESSORS AVAILABLE TO THE USER

```
:RUN MINISIS.PUB.MINISIS;LIB=P

1.  COMPUTE     2. ENTRY    ....
```

- MENU IS SELECTED FROM MESSnn FILE
  - UNDER SECURITY
  - USER PROFILE DEFINED BY DATA BASE MGR
  - NOT UNDER SECURITY
  - ACCESS PARAMETERS GENERATE MENU

ACCESS PARAMETERS

- LINES 1350/1388 IN MESSnn
- USED BY *MINISIS PROGRAM* TO PRESENT MENU OF PROCESSORS TO USER
- USED ONLY IF SECURITY FEATURE OF MINISIS NOT ACTIVATED FOR LOG-ON ACCOUNT

18.

F02-08-86

ACCESS PARAMETERS (continued)

```
INDEX,INDEX.PUB.MINISIS;ACCESS=MGR,INDEXER
  :      :   :        :            :
 MENU PROG.  GROUP     :          ACCESS
"NAME NAME         ACCOUNT        PARAMETERS
```

- MENU PRESENTED TO USER ACCORDING TO WHETHER ACCESS PARAMETERS OF PROCESSORS MATCH USER'S LOCAL ATTRIBUTE

```
LOCATTR  0=optr
         1=mgr
         2=searcher
         3=indexer
         4=supvr
         5=sys
         -=any
```

- PROCESSOR PRESENTED IN MENU IF USER'S LOCATTR MATCHES *ACCESS PARAMETER* OR IF ACCESS PARAMETER IS ANY

- EG: NEWUSER TECHSERV;LOCATTR=3
  TECHSERV WILL HAVE ALL PROCESSORS WITH PARAMETERS INDEXER OR ANY ON MENU

F02-08-86

MINISIS SECURITY

- REQUIRES AL CAPABILITY TO INVOKE
- ACTIVATED BY SECURE COMMAND IN DATADEF
- ACCOUNT-WIDE
- ASSUMES EXISTENCE OF GROUP SECU *IN ACCOUNT*
- WILL AUTOMATICALLY BUILD A KSAM FILE (SECUKEYD/SECUDIR) IN SECU GROUP
- IF SECUKEYD ABSENT - USER MENU COMES FROM ACCESS PARAMETERS
- IF SECUKEYD IS PRESENT - USER MAY NOT RUN ANY PROCESSOR OR ACCESS ANY DATA BASE UNLESS IT IS IN THE USER'S SECURITY PROFILE

2 CLASSES OF FUNCTIONS

UTILITIES
- PROGRAMS WHICH DON'T OPEN A DATA BASE
  - minisis
  - listformat
  - invert
  - syncomp
  - fixxref
  - minedit
  - datadef
- USER MAY OR MAY NOT BE GRANTED ACCESS DEPENDING UPON PROFILE

BREKR

20

F02-08-86

MINISIS SECURITY (continued)

PROCESSORS
- PROGRAMS SUCH AS ENTRY, PRINT, ...
- ACCESS TO PROCESSOR LINKED TO DATA BASE
- ACCESS TO DATA BASE DEFINED FOR USER
- PROCESSORS WHICH CAN BE USED TO ACCESS EACH DATA BASE ARE DEFINED
- EG: DB1 - PRINT, QUERY
  DB2 - ENTRY, MODIFY, PRINT, QUERY
- USER WOULD SEE
  ENTRY, MODIFY, PRINT, QUERY IN MENU
  BUT CAN ONLY USE PRINT AND QUERY IN DB1

UTILITIES CALLED BY OTHER PROGRAMS

- TWO UTILITIES ARE CALLED BY OTHER PROGRAMS RATHER THAN BY THE USER

INVRTSRT
- CALLED BY INVERT TO SORT ADJ B-TREE FILES

SDIREFMT
- CALLED BY QUERY IN SDI MODE TO CONVERT RECORDS IN A PROFILE DATA BASE TO SEARCH STATEMENTS

- THESE UTILITIES DO NOT APPEAR ON THE MENU

21.

F02-08-86

INSTALLATION PROCEDURES

ORGANISATION OF MINISIS TAPE

- STANDARD MPE STORE TAPE
- ALL FILES IN ACCOUNT MINISIS
- ALL FILES CREATED BY USER MGR
- STANDARD GROUPS
  - PUB
  - UTILITY
  - DOC
  - SOURCE
  - FIXES
- OPTIONAL GROUPS
  - IDRC
  - SAMPLE
  - DEMO

22.

F02-08-86

INTERESTING FILES

1) PUB
   - MINISIS APPLICATIONS PROGRAMS
   - message files SYNnn, MESSnn, ERRnn
   - SL - SEGMENTED LIBRARY

2) UTILITY
   - MINISIS UTILITY PROGRAMS
   - SYNnnT - EDITOR FORMAT OF SYNnn

3) DOC
   - MINISIS DOCUMENTATION FILES
   - GALLEY PROGRAM
   - JOBS TO PRINT DOCUMENTATION

4) SOURCE
   - PROGRAMMING INCLUDE FILES

5) FIXES
   - DATA BASE OF MINISIS SOFTWARE STATUS REPORTS

F02-08-86

PROCEDURES - INITIAL INSTALLATION

1. HAVE SYSTEM MANAGER CREATE MINISIS ACCOUNT
   :NEWACCT MINISIS,MGR;
   :CAP=IA,BA,AM,AL,PH,DS,SF,ND,MR;
   :ACCESS=(R,W,X,A,L:ANY);LOCATTR=%7

2. LOG-ON TO MINISIS ACCOUNT
   :HELLO MGR.MINISIS

   :ALTGROUP PUB;ACCESS=(R,X:ANY;W,A,L,S:AC)

3. RESTORE 2 FILES FROM TAPE
   :FILE MINTAPE;DEV=TAPE
   :RESTORE *MINTAPE;JOBGROUP,JOBLOAD

4. CREATE MINISIS GROUPS
   :STREAM JOBGROUP

5. RESTORE MINISIS FILES
   :STREAM JOBLOAD

PROCEDURES - UPDATES
(WHEN NEW RELEASE TAPE ARRIVES)

1. RESTORE FILES FROM TAPE
   :HELLO MGR.MINISIS
   :FILE MINTAPE;DEV=TAPE
   :RESTORE *MINTAPE;@.@.MINISIS

24.

F02-08-86

SETTING UP APPLICATION ACCOUNTS

- IDEALLY ONE APPLICATION PER ACCOUNT
- ONE COMPONENT OF APPLICATION PER GROUP
  EG. ACCOUNT - LIBRARY
  GROUPS - ACQUIS
  CATAL
  SEARCH
  :

- HAVE SYSTEM MANAGER CREATE ACCOUNT
  :NEWACCT xxxxxxx,MGR;
  ;CAP=AM,AL,IA,BA,PH,DS,ND,SF,MR;
  :ACCESS=(R,W,A,L,X:AC),LOCATTR=%7

- LOG-ON AS ACCOUNT MANAGER AND CREATE REQUIRED GROUPS INCLUDING SECU IF NECESSARY
  :NEWGROUP xxxx;CAP=IA,BA,PH,DS,MR;
  :ACCESS=(R,W,A,L,X,S:AC)

- CREATE USERS WITH APPROPRIATE LOCATTR
  :ALTUSER MGR;LOCATTR=%5
  :NEWUSER xx;LOCATTR=xx;HOME=xxx

F02-08-86

MINISIS UDC

- MINISIS PROGRAMS ASSUME MESSnn, ERRnn, SYNnn FILES ARE IN PUB GROUP OF ACCOUNT IN WHICH PROGRAM IS RUN

- IF YOU ONLY WANT TO HAVE ONE COPY OF THESE FILES ON YOUR SYSTEM, USE FILE EQUATIONS

- USE EDITOR TO BUILD UDC FILE:

```
MINFILE, OPTION LOGON
FILE ERR00.PUB=ERR00.PUB.MINISIS
FILE MES600.PUB=MES900.PUB.MINISIS
FILE SYN00.PUB=SYN00.PUB.MINISIS
***
MINISIS
RUN MINISIS.PUB.MINISIS;LIB=P
***
```

- MINISIS EXPECTS DDUSER AND USERSCHM FILES TO RESIDE IN PUB.MINISIS
- IF NOT, USE FILE EQUATIONS IN UDC:
  :FILE DDUSER.PUB.MINISIS=DDUSER.group.acct
  :FILE USERSCHM.PUB.MINISIS=USERSCHM.group.acct
- IF THESE FILES REMAIN IN MINISIS ACCOUNT, THEY MUST BE RELEASED

F02-08-86

MINISIS USER-CONTRIBUTED LIBRARY (UCL)

- PROGRAMS AND EXITS WRITTEN BY MINISIS USERS

- NOT SUPPORTED BY EITHER IDRC OR CONTRIBUTOR

- SOURCE CODE IS RELEASED
- USERS ARE FREE TO MAKE CHANGES

- DISTRIBUTED WITH MINISIS SYSTEM

- PROGRAMS ARE IN MINLIB ACCOUNT

- BUILD ACCOUNT MINLIB
  :NEWACCT MINLIB,MGR;
  :CAP=IA,BA,PH,DS,MR,SF,ND,AM,AL
  :ACCESS=(R,W,A,L,S,X:ANY);LOCATTR=%7

  :ALTGROUP PUB;ACCESS=(R,X:ANY;W,A,L,S:AC)

- BUILD GROUPS:
  - SOURCE (source code files)
  - DOC (documentation files)
  - OBJECT (programs & USL file)
  - QUERY (data base of information)
  - MINFILES (jobs to compile programs)
  - EXAMPLE (sample jobs etc.)

  - OPTIONALLY LANGARAB AND LANGCHIN

F02-08-86

INVERSION

- EXTRACTS KEYS FROM FIELDS IN MASTER DB
- WRITES POSTINGS TO FAST ACCESS FILE
- MAY ALSO WRITE KEYS TO FAST ACCESS FILE IF IT IS A B-TREE FILE

KEY
- DATA IN A FIELD
- MAY BE A WORD
  - DELIMITED BY BLANK SPACES
  - EG. NEW METHODS IN HARVESTING
    1) 2) 3) 4)
- MAY BE A TERM
  - DELIMITED BY NON-ALPHANUMERIC CHARACTERS, USUALLY //
  - EG. /NEW METHODS/ IN /HARVESTING/
    1) 2)
- MAY BE THE ENTIRE FIELD

POSTING
- ISN OF RECORD IN WHICH KEY WAS FOUND
- STORED IN COMPRESSED BITSTRINGS

1.

F02-08-86

TYPES OF FAST-ACCESS FILES

B-TREE

- KEYS ARE WRITTEN WITH THEIR POSTINGS
- IF KEY IS ALREADY PRESENT IN FILE, ONLY POSTING IS WRITTEN

KSAM

- KEYS ALREADY EXIST IN FILE
- ONLY POSTINGS ARE WRITTEN
- KEYS EXTRACTED FROM MASTER DB COMPARED TO KEYS IN FAST ACCESS FILE
- WHERE MATCH IS FOUND, POSTING STORED IN BITSTRINGS STORAGE AREA OF KSAM RECORD WHICH CONTAINS THAT KEY

MASTER DB	
ISN	TITLE
1	COMPUTER SYSTEM
.	
21	HARDWARE
.	
50	LIBRARY SYSTEM
.	
.	

FAST ACCESS FILE	
KEY	POSTINGS
COMPUTER	1...
.	
HARDWARE	21 ..
.	
LIBRARY	50 ..
.	
SYSTEM	1,50

2

F02-08-86

TYPES OF INVERSION

1) ONLINE - THROUGH ENTRY, MODIFY, ETC.
2) RELEASE - THROUGH RELEASE
3) OFFLINE - THROUGH INDEX AND INVERT

ONLINE INVERSION

- IF FIELD FLAGGED FOR ONLINE INVERSION, INVERSION TAKES PLACE DURING ENTRY, MODIFY, BATCHIN OR ISOCONV
- FAST ACCESS FILES ARE UPDATED WHEN:
  - A NEW RECORD IS ENTERED
  - INVERTED FIELD IS MODIFIED/DELETED
  - RECORD IS DELETED

INVERSION ATTRIBUTES ARE DEFINED IN FIELD DEFINITION:
EG:
- TYPE OF KEY
- LENGTH OF KEY
- TYPE AND NAME OF FAST ACCESS FILE
- IF KEY IS STRIPPED

3

F02-08-86

INVERSION ON RELEASE

- IF FIELD FLAGGED FOR INVERSION ON RELEASE, FAST ACCESS FILE IS NOT UPDATED UNTIL RECORD IS RELEASED FOR RETRIEVAL
- MAY BE SOME TIME AFTER RECORD ENTRY

RELEASING FOR RETRIEVAL

- LOCKS RECORD AGAINST FURTHER MODIFICATION
- IF FIELDS FLAGGED FOR INVERSION ON RELEASE POSTINGS ARE WRITTEN TO FAST ACCESS FILES
- IF RECORD IS TO BE UPDATED, IT MUST BE RELEASED FOR MODIFICATION
- POSTINGS ARE ERASED FROM FAST ACCESS FILE
- POSTINGS WILL BE RE-WRITTEN IF RECORD IS RELEASED FOR RETRIEVAL AGAIN
- RELEASE PROCESSOR PERMITS RECORD DELETION
- WHEN RECORD IS DELETED, ITS POSTINGS ARE ERASED FROM ALL FAST ACCESS FILES

4

F02-08-86

RELEASE COMMANDS

FUNCTION MOD/RET/DEL
ISN=nn/mm
KEYNAME=starting key value/ending key value
HITFILE=filename
QUERY FORMULATION
DELIMITER "delimiters"
VERIFY yes/no
FORMAT filename
EXIT

RECORD SELECTION
ISN=nn OR ISN=nn/mm

KEYNAME=starting/ending key value
EX: W010=ABC
W010=ABC/XYZ

QUERY FORMULATION
EX: = FISHERIES
OR FORESTS
$

HITFILE=filename

5

F02-08-86

THE FUNCTION COMMAND

FUNCTION MOD/RET/DEL

FUNCTION DEL (DELETE)
- LOGICALLY DELETES A RECORD
- REMOVES POSTINGS FOR ALL KEYS IN RECORD FROM ALL FAST ACCESS FILES

FUNCTION RET (RELEASE FOR RETRIEVAL)
- LOCKS RECORD AGAINST MODIFICATION
- UPDATES FAST ACCESS FILES IF FIELDS FLAGGED FOR INVERSION ON RELEASE

FUNCTION MOD (RELEASE FOR MODIFICATION)
- UNLOCKS RECORD FOR MODIFICATION
- REMOVES POSTINGS IF FIELDS FLAGGED FOR INVERSION ON RELEASE

- BY DEFAULT, ALL RECORDS ARE IN MOD STATE WHEN THEY ARE CREATED

6

F02-08-86

OTHER RELEASE COMMANDS

DELIMITER "delimiters"
- ALLOWS YOU TO SPECIFY DELIMITERS TERMS (OTHER THAN "//")
- EG. IF DELIMITER "* *"
FORESTS RECOGNISED AS A TERM
- DEFAULT IS "//"
- EG. /FORESTS/

VERIFY yes/no
- DISPLAYS EACH RECORD AND ASKS IF YOU WANT RECORD PROCESSED
- DEFAULTS TO VERIFY NO

FORMAT filename
- ALLOWS YOU TO SPECIFY ALTERNATE FORMAT FOR VERIFY

EXIT
- ALLOWS YOU TO SELECT ANOTHER DB OR RETURN TO MENU

7

F02-08-86

RELEASE ON A KSAM DB

- ONLY DELETE FUNCTION IS SUPPORTED
- IF KSAM FILE IS INVERTED, ANY POSTINGS IN DELETED RECORD ARE ALSO DELETED

EXAMPLE OF RELEASE

ENTER DB NAME OR EXIT - projects
- function mod
- ISN=5/10
- ISN=30
- function del
- ISN=5/10
- ISN=30
- function ret
- ISN=100/200
- exit
TOTAL RECORDS RELEASED FOR MODIFICATION=7
TOTAL RECORDS RELEASED FOR RETRIEVAL=101
TOTAL RECORDS DELETED=7

RELEASE MAY RUN INTERACTIVELY OR IN BATCH

8

OFFLINE INVERSION

- MAY TAKE PLACE IN BATCH OR INTERACTIVELY

- EXECUTED ON A RANGE OF RECORDS, USUALLY SOME TIME AFTER RECORDS ARE ENTERED

- EXECUTED THROUGH 2 PROCESSORS
  - INDEX
  - INVERT

* * * * * * * * *

1) INDEX

- EXTRACTS AND SORTS KEYS FROM MASTER DB
- OUTPUT RECORD HAS 1 (PRIMARY) KEY
- INPUT - ENTIRE DB OR SUBSET
- LENGTH OF KEY=KEYLENGTH OF FAST ACCESS FILE
- KEY IN UPPER CASE - UPSHIFT=YES
- CHARACTERS MAY BE STRIPPED FROM KEY
- TYPE OF EXTRACTION - WORD (TYPE=W)
  - TERM (TYPE=T)
  - ENTIRE FIELD
  - FUNC=NN

2) INVERT

- INVERT HAS 4 FEATURES:

INVERT	- WRITES POSTINGS FROM INDEX OUTPUT FILE TO FAST ACCESS FILE (MAY ALSO WRITE KEYS)
TREEMANT	- MAINTAINS B-TREE FILES
LIST	- LISTS KEYS AND NO. OF POSTINGS IN FAST ACCESS FILE
UNINVERT	- OPPOSITE OF INVERT - ERASES POSTINGS FOR KEYS IN AN INDEX OUTPUT FILE

- IF EXISTING FAST ACCESS FILE UPDATED, EXISTING KEYS WILL HAVE THEIR POSTINGS UPDATED

- AFTER MODIFICATION OF RECORDS AND RE-INVERSION, WHEN A KEY NO LONGER OCCURS IN A DB, IT MAY STILL APPEAR IN FAST ACCESS FILE WITH ZERO POSTING

SAMPLE INVERSION

1) RUN INDEX TO PRODUCE OUTPUT FILE CONTAINING KEYS AND ISNS

KEYS	ISNS
BOOKS	57
BOOKS	150
BOOKS	301
.	.
.	.
.	.
COMPUTERS	17
COMPUTERS	103
COMPUTERS	301

2) RUN INVERT TO WRITE ISNS TO FILE (AND KEYS IF FILE IS B-TREE AND KEYS AREN'T ALREADY THERE)

KEYS	POSTINGS
BOOKS	57, 150, 301
.	
.	
.	
COMPUTERS	17, 103, 301


```
SYNTAX OF INVERT
INVERT,TREEMANT,LIST,UNINVERT OR EXIT - i
INDEX FILE NAME - index output filename
KEY FILE NAME (4 CHAR) - fast access file
```

OUTPUT FROM INVERT

- INVERT WILL PRODUCE THESE LISTINGS:

OUTFL - DISPLAYS KEYS, NO. OF POSTINGS

KEYS	POSTINGS	ISNS
APPLE	17	4
BREAD	22	4
CAKE	9	4

ERRFL - FOR KSAM FAST ACCESS FILE
- DISPLAYS REJECTED KEYS

- BOTH FILES ARE SENT TO LINE PRINTER

- LIST FEATURE WILL ALSO PRODUCE OUTFL
- DISPLAYS KEYS AND NO. OF POSTINGS

F02-08-86

SAMPLE INDEX / INVERT BATCH JOB

```
!JOB INVTITL,MGR.MINISIS,IDRC
!PURGE TWORD
!PURGE TWORDDD
!RUN INDEX.PUB;LIB=P
LIBRARY
ISN=1/400
OUTPUT=TWORD
KEY=TITLE,LENGTH=40,UPSHIFT=YES,KEEP=NO
FIELD=TITLE,TYPE=W,STRIP=N
END
***
!RUN INVERT.UTILITY;LIB=P
INVERT
TWORD
TITL
EXIT
!PURGE TWORD
!PURGE TWORDDD
!EOJ
```

/3

F02-08-86

SAMPLE BATCH JOB FOR ADJ AND LEFTADJ FILES

```
!JOB INVTITL,MGR.MINISIS,IDRC
!RUN DATADEF.UTILITY;LIB=P
ERASE BTREE=TADJ                    (1)
EXIT
!RUN INDEX.PUB;LIB=P
LIB
ISN=1/500
OUTPUT=TWORD                              (2)
DATAFIELD=TITLE,LENGTH=40,UPSHIFT=YES,& (3)
KEEP=NO,SHOWOCC=YES,SHOWPOS=YES,SHOWTAG=YES
FIELD=TITLE,TYPE=W,STRIP=D
END
***
!RUN INVERT.UTILITY;LIB=P
INVERT
TWORD
TADJ
YES                    (4)
EXIT
!EOJ
```

(1) - FILE ERASED BEFORE RE-INVERSION
(2) - DATAFIELD, SHOWOCC, SHOWPOS, SHOWTAG
(3) - USE AMPERSAND IF LINE FOLDS
(4) - REPLY YES TO SORT INDEX OUTPUT FILE

/4

F02-08-86

SAMPLE BATCH JOB FOR DOT INVERTED FILES

```
!JOB INVDESC,MGR.MINISIS,IDRC
!RUN INDEX.PUB;LIB=P
LIB
ISN=1/500
OUTPUT=DESCX          (1)                 (2)
KEY=DESC,LENGTH=21,UPSHIFT=YES,BRKSEQ="/"
FIELD=DES,TYPE=T,BEFORE=YES,SUFFIX="P"   (3)
FIELD=DES,TYPE=T,BEFORE=NO,SUFFIX="S"    (4)
END          (5)
***
!RUN INVERT.UTILITY;LIB=P
INVERT
DESCX,SUFFIX=P,S              (6)
DESC
EXIT
!EOJ
```

(1) - LENGTH OF KEY + 1 FOR SUFFIX
(2) - IF BREAK SEQUENCE IS NOT DEFAULT
(3) - PRIMARY DESCRIPTORS
(4) - SECONDARY DESCRIPTORS
(5) - FOR DESCRIPTORS (TERMS) ONLY
(6) - NOTE SUFFIX COMMAND

F01-08-85

THE THLOADER PROCESSOR

- LOADS TERMS FROM MASTER DB TO KSAM FILE
- KSAM FILE BECOMES FAST ACCESS FILE FOR ANOTHER MASTER DB
- THLOADER ALSO:
  - ESTABLISHES RELATIONSHIPS BETWEEN TERMS AND OTHER-LANGUAGE EQUIVALENTS IN UP TO 9 LANGUAGES
  - ESTABLISHES RELATIONSHIPS BETWEEN TERMS AND THEIR BROADER, NARROWER, RELATED, ANY, AND USE FOR (FORBIDDEN) TERMS
- SO THAT THESAURUS QUERY MAY BE SUPPORTED IN THE QUERY PROCESSOR

F01-08-85

THE THLOADER PROCESSOR

- LOADS TERMS FROM MASTER DB TO KSAM FILE
- KSAM FILE BECOMES FAST ACCESS FILE FOR ANOTHER MASTER DB
- THLOADER ALSO:
  - ESTABLISHES RELATIONSHIPS BETWEEN TERMS AND OTHER-LANGUAGE EQUIVALENTS IN UP TO 9 LANGUAGES
  - ESTABLISHES RELATIONSHIPS BETWEEN TERMS AND THEIR BROADER, NARROWER, RELATED, ANY, AND USE FOR (FORBIDDEN) TERMS
- SO THAT THESAURUS QUERY MAY BE SUPPORTED IN THE QUERY PROCESSOR

F01-08-85

BUILDING A THESAURUS

- A THESAURUS IS BUILT BY CREATING A MASTER DB TO HOLD ALL TERMS IN THE SEARCH VOCABULARY IN UP TO 9 LANGUAGES
- IN THIS DB, EVERY RECORD CONTAINS 1 TERM
- IT ALSO CONTAINS TRANSLATIONS OF THE TERMS
- AND OTHER TERMS RELATED TO THE TERM

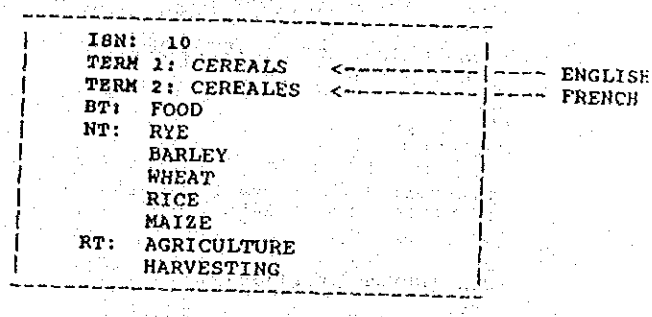
THEREFORE YOU MUST DEFINE THESE FIELDS:

- FIELD FOR THE TERM (ONE FOR EVERY LANGUAGE OF THESAURUS)
- REPEATABLE FIELD FOR BROADER TERMS
- REPEATABLE FIELD FOR NARROWER TERMS
- REPEATABLE FIELD FOR RELATED TERMS

OPTIONALLY:

- REPEATABLE FIELD FOR ANY TERMS
- FIELD FOR FORBIDDEN TERMS (1 FOR EVERY LANGUAGE)
- FIELD FOR USE TERMS (1 FOR EVERY LANGUAGE)

F01-08-85



- BT, NT, RT MAY REPEAT
- ONLY TERM MUST BE PRESENT IN ALL LANGUAGES
- BT, NT, RT MUST EXIST AS TERMS IN OTHER RECORDS
- LENGTH OF FIELDS SHOULD BE THE SAME AS LENGTH OF KEY USED IN SEARCH

F01-08-85

ANY TERMS

- WHEN TERMS BELONG IN AN ANY TABLE, THE ANY TERM IS STORED IN THE TERM RECORD

EG: IN ORDER TO SEARCH AS FOLLOWS:

```
>= NT CEREALS
     RYE      P=10
```

YOU WOULD DEFINE RYE AS THE NT OF THE TERM CEREALS

```
+-------------------------+
|  ISN:  10               |
|  TERM1:  CEREALS        |
|  TERM2:  CEREALES       |
|  NT:  RYE               |
+-------------------------+
```

F01-08-85

ANY TERMS (continued)

- BUT TO SEARCH AS FOLLOWS:

```
>= ANY AGRICULTURAL PRODUCTION
     CEREALS      P=90
```

YOU WOULD DEFINE AGRICULTURAL PRODUCTION AS THE ANY TERM OF CEREALS

```
+-------------------------+
|  ISN:  10               |
|  TERM1:  CEREALS        |
|  TERM2:  CEREALES       |
|  ANY:  AGRICULTURAL     |
|          PRODUCTION     |
+-------------------------+
```

- A TERM MAY BELONG TO MORE THAN ONE ANY TABLE

F01-08-85

FORBIDDEN TERMS

- FT'S ARE STORED IN THEIR OWN RECORDS
- THEY ARE ACCOMPANIED BY A *USE TERM* WHICH IS A VALID TERM IN ANOTHER RECORD IN THE DB
- FT'S AND USE TERMS MUST BE ENTERED IN ALL LANGUAGES OF THE THESAURUS

EX:

```
+-------------------------+
|  ISN:  10               |
|  TERM1:  CEREALS        |
|  TERM2:  CEREALES       |
|  BT:  FOOD              |
|  NT:  RYE               |
+-------------------------+

+-------------------------+
|  ISN:  10               |
|  FT1:  CEREAL           |
|  FT2:  CEREALE          |
|  USE1:  CEREALS         |
|  USE2:  CEREALES        |
+-------------------------+
```

IN QUERY

```
> FT on
>= CEREAL
   CEREALS       P=90
```

F01-08-85

LOADING THE THESAURUS

- WHEN THE THESAURUS HAS BEEN BUILT IT IS READY TO BE LOADED INTO THE KSAM FILE WHICH WILL SERVE AS A FAST ACCESS FILE FOR THE DATA BASE OF DOCUMENTS

- LOADING IS ACCOMPLISHED IN 3 STEPS:

1. RUN INDEX TO OBTAIN SORTED LISTS OF TERMS
2. BUILD THE KSAM FILE BY DEFINING A KSAM DB IN DATADEF (FILE TYPE THES)
3. RUN THLOADER TO LOAD TERMS INTO THE KSAM FILE

F01-08-85

1. CREATING INDEX OUTPUT FILES

- INDEX OUTPUT FILES WILL BE INPUT TO THLOADER
- *ONE OUTPUT FILE IS NEEDED FOR:*

  TERMS
  BROADER TERMS
  NARROWER TERMS
  RELATED TERMS
  ANY TERMS

- ONE OUTPUT FILE IN EACH LANGUAGE IS NEEDED FOR:

  FORBIDDEN TERMS

F01-08-85

THE TERMS OUTPUT FILE

- 1 PRIMARY KEY
- KEYLENGTH=KEYLENGTH OF KSAM FILE + 1
- UPSHIFT SHOULD BE YES UNLESS TERMS WERE WRITTEN IN UPPER CASE
- *FIELDS* - *TERMS IN ALL LANGUAGES*
  - SUFFIX=" " (language)
- EG. IF KEY LENGTH OF KSAM FILE IS 40

```
        KEY=X,LENGTH=41  <---------------
------> FIELD=TERM1,SUFFIX="E"          |
| ---->  FIELD=TERM2,SUFFIX="F"          |
| |                                      |
| |                  KEYLENGTH + 1 FOR SUFFIX
| |
--|--- E=ENGLISH TERM
  ---  F=FRENCH TERM
```


F01-08-85

THE BT, NT, RT OUTPUT FILES

- 2 KEYS - 1ST KEY IS THE TERM
  - 2ND KEY IS BT, NT OR RT

EG:

```
KEY=TERM,LENGTH=40   (1)
FIELD=TERM1   (2)
END
KEY=BT,LENGTH=40   (3)
FIELD=BT,BLANK=NO,OCC=100
END          (4)       (5)
```

1) KEYLENGTH OF KSAM FILE
2) TERM (IN ONE LANGUAGE ONLY)
3) BROADER TERM
4) ONLY GENERATE A KEY IF BT OCCURS
5) GENERATE A KEY FOR ALL OCCURRENCES OF BT

* * *REPEAT FOR NT AND RT* * *

F01-08-85

THE ANY OUTPUT FILES

- LIKE BT, NT AND RT OUTPUT FILES BUT ANY TERM IS THE *1ST KEY AND TERM IS THE 2ND KEY*

EX:

```
KEY=ANY TERM,LENGTH=40
FIELD=ANY
END
KEY=TERM,LENGTH=40
FIELD=TERM1
END
```

F01-08-85

THE FT OUTPUT FILES

- *MUST BE 1 FOR EACH LANGUAGE*
- 2 KEYS: 1ST KEY IS FT
  2ND KEY IS USE TERM

```
EX:  KEY=FT1,LENGTH=40
     FIELD=FT1
     END
     KEY=USE1,LENGTH=40
     FIELD=USE1
     END
```

REPEAT FOR ALL LANGUAGES

NAMING CONVENTION: OUTPUT FILE
FOR 1ST LANGUAGE IS name
FOR 2ND LANGUAGE IS name1
EG. FTX, FTX1

F01-08-85

2. BUILDING THE THESAURUS KSAM FILE

- IN DATADEF, CREATE AN RD
- *SUPPLY FILE TYPE THES*
- THIS IS A FORM OF KSAM FILE

- YOU WILL BE ASKED FOR:
  - KEYLENGTH
  - NO. OF LANGUAGES

  IN OUR EXAMPLE KEYLENGTH IS 40
  AND NO. OF LANGUAGES IS 2

- DO NOT INSERT FIELDS
- YOU WILL FIND THAT THESE FIELDS HAVE BEEN DEFINED FOR YOU:

```
MAIN TERM                --- PRIMARY KEY
CONCEPT NO. OF MAIN TERM --- SEC. KEY
CONCEPT NO. OF LANGUAGE1 --- 1 FOR EACH
CONCEPT NO. OF LANGUAGE2     LANGUAGE
ISN OF TERM
```

- THE CONCEPT NO. IS A UNIQUE NUMBER ASSIGNED TO EACH TERM WHEN IT IS LOADED BY THLOADER

F01-08-85

3. RUNNING THLOADER

SYNTAX IS:

```
NAME OF THESAURUS KSAM DB - db name
FILE THESAUR=terms output file
FILE ANY=any output file
FILE BT=bt output file
FILE RT=rt output file
FILE NT=nt output file
FILE FT=ft output file of 1st language
LANGUAGE EN="E",FR="F"
END
```

LANGUAGE
- EN, FR AS SUPPLIED IN LANG COMMAND IN QUERY
- ESTABLISHES LANGUAGE IDENTITY

TRIAL yes/no
- IF YES, SPECIFIES A TRIAL RUN ONLY
- KSAM FILE IS NOT LOADED

LIST yes/no
- LIST ALL TERMS PLUS CONCEPT NUMBERS

F01-08-85

WHEN THLOADER COMMANDS ARE SUPPLIED

1. - TERMS FROM TERMS OUTPUT FILE ARE ASSIGNED UNIQUE CONCEPT NUMBERS *AND LOADED INTO KEY* FIELDS OF KSAM FILE - 1 TERM TO 1 RECORD
   - CONCEPT NUMBERS ARE LOADED INTO CN OF MAIN TERM FIELD
   - CNS OF 1ST LANGUAGE TERM ARE LOADED INTO CN1 FIELD
   - CNS OF 2ND LANGUAGE TERM ARE LOADED INTO CN2 FIELD

   ```
   EX:  CEREALS=CN 30
        CEREALES=CN 31
   ```

   - *SUFFIX* E OR F INDICATES WHETHER TERM IS 1ST OR 2ND LANGUAGE

MAIN TERM=CEREALS CN OF MAIN TERM=30 CN1=30 CN2=31	MAIN TERM=CEREALES CN OF MAIN TERM=31 CN1=30 CN2=31

F01-08-85

2. - FTS ARE LOADED WITH THE CONCEPT NO. OF THEIR USE TERMS

   - FTS ARE LOADED WITH A VERTICAL BAR (|) AS THE FIRST CHARACTER IN THE KEY

3. - A KSAM FILE CALLED STRCKEYD IS BUILT BY THLOADER
   - EACH RECORD HAS 2 FIELDS:

     1. CN OF MAIN TERM AND STRUCTURE CODE
     2. CN OF STRUCTURE TERM

STRUCTURE CODES
BT=B
NT=N
RT=R
ANY=ANY

EG. IF TERM FOOD HAS CN=66
AND TERM CEREALS HAS CN=30
FOLLOWING RECORD IS LOADED INTO STRCKEYD:

B30	CN OF MAIN TERM PLUS CODE
66	CN OF STRUCTURE TERM

F01-08-85

4) AN OUTPUT LISTING IS PRODUCED IF LIST=YES

- LISTING WILL SHOW TERMS AND CNS:

	CN	CN 1	CN 2
:			
BARLEY	12	12	89
:			
CEREALS	30	30	31
CEREALES	31	30	31
:			
ORAGE	89	12	89

- IT WILL ALSO SHOW BT, FT, NT, RT AND ANY RELATIONSHIPS

- IF A BT, RT, NT, ANY OR USE TERM IS NOT A VALID TERM:
  - THE KSAM FILE WILL NOT BE LOADED
  - STRCKEYD WON'T BE BUILT
  - INVALID TERM AND ISN OF THE RECORD IN WHICH IT WAS FOUND WILL BE DISPLAYED ON THE THESLIST LISTING

/7

F01-08-85

WHEN THE THESAURUS IS LOADED

- THE DB MANAGER MUST INVERT THE KSAM FILE SO THAT USERS CAN SEARCH ON THE DB OF DOCUMENTS

- ANY KEY IN THE DOCUMENTS DB NOT FOUND FAST ACCESS FILE WILL BE REJECTED AND LISTED IN THE ERRFL OUTPUT

- THE INVERTED FIELD IN THE DOCUMENTS DB MUST HAVE THESE CHARACTERISTICS:
  - DEFAULT QUERY FIELD
  - OFFSET=4
  - TYPE OF INVERTED FILE=K
  - LENGTH OF STRING TO EXTRACT= KEYLENGTH OF KSAM FILE
  - THESAURUS STRUCT. ON FAST ACCESS FILE=YES
  - TYPE OF EXTRACTION=T OR ENTIRE FIELD
  - STRIPPING OF DIACRITICALS
  - NAME OF FAST ACCESS FILE=KSAM FILE

NEVER UPDATE THE KSAM FILE THROUGH MODIFY
IF YOU WANT TO CHANGE, ADD OR DELETE TERMS
UPDATE THE THESAURUS MASTER DB
YOU MUST RE-LOAD THE ENTIRE THESAURUS

F01-08-85

IN QUERY

- DISPLAYING A THESAURUS WILL DISPLAY THE TERM, ITS TRANSLATIONS, AND STRUCTURE TERMS

```
DISPLAY CEREAL@
    CEREALES          P=13
    CEREALS           P=30
      BT NOURRITURE   P=100
      NT SEIGLE       P=9
         ORAGE        P=15
```

TO DISPLAY FTS:

```
DISPLAY @|
    CEREAL
    CEREALS      P=30
```

F01-08-85

THE NON-THESAURUS ANY FILE

- OFFERS ANY TABLE SEARCHING CAPABILITIES WITHOUT A THESAURUS

- ANY FILE IS AUXILIARY FILE
- NOT FAST ACCESS FILE

- INVERTED FIELD IS LINKED WITH B-TREE FILE OR KSAM FAST ACCESS FILE
- ANY TABLES IN ANY FILE CONTAIN TERMS FROM FAST ACCESS FILE

TO BUILD AN ANY FILE:

IN DATADEF

- BUILD A DATA BASE
- FILE TYPE IS ANY
- DO NOT DEFINE FIELDS
- THESE FIELDS WILL BE BUILT BY DATADEF

X010 - ANY TERM
X020 - TERM

F01-08-85

IN ENTRY

- ENTER X010 (ANY TABLE HEADER)
- ENTER X020 (TERM)

- TERM IN X020 MUST BE TERM IN INVERTED FIELD
- ANY TABLE HEADER MAY BE ANY WORD OR PHRASE

- LENGTH OF FIELD = LENGTH OF KEY

- DATA IN FIELD MUST BE UPPER CASE
- MAY ALSO BE STRIPPED

- TERMS MAY BE IN SEVERAL DIFFERENT ANY TABLES
- BUT ANY TABLE HEADERS MUST BE UNIQUE

EX: X010=AFRICA
X020=ANGOLA

X010=AFRICA
X020=BOTSWANA

X010=AFRICA
X020=CAMEROON

.
.

F01-08-85

IN QUERY

- SYNTAX OF ANY SEARCH IS

>= <field id> ANY <ANY table header>

EX:

>= AREA ANY AFRICA

ANGOLA
BOTSWANA
CAMEROON
.
.

- TERMS FOUND ARE OR'd TOGETHER
- TERMS WITH ZERO POSTINGS ARE NOT DISPLAYED

F02-08-86

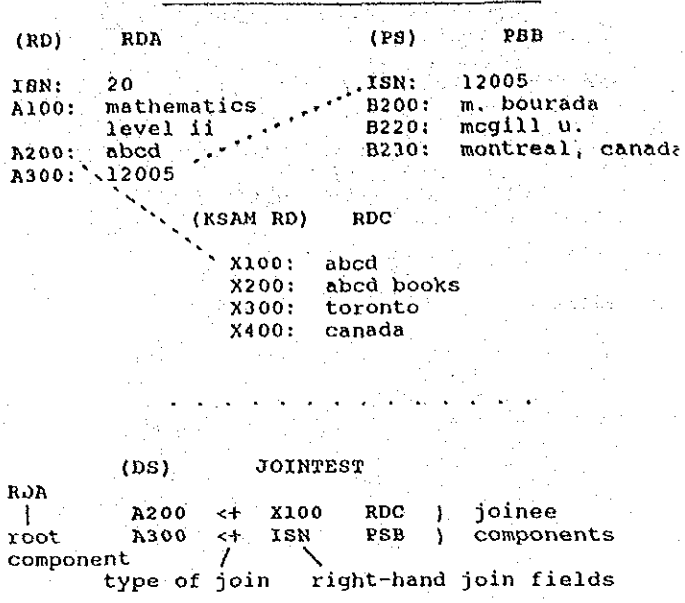
DATA SUBMODELS

- MOST COMPLEX TYPE OF USER VIEW
- RESULT OF COMBINING 2 OR MORE DATA BASES
- DS SHARES CHARACTERISTICS OF OTHER USER VIEWS
  - AUDIT TRAIL FILES
  - DEFAULT QUERY FIELD
  - DEFAULT PRINT FORMAT
  - INITIAL RESTRICTIONS
  - PROJECT LIST
- DS MAY CONTAIN MAXIMUM NUMBER OF 256 FIELDS FROM ITS COMPONENTS
- COMPONENTS RETAIN THEIR OWN PHYSICAL FILES WHEN RECORDS ARE READ FROM THE DS
- FIELDS ARE LINKED THROUGH JOIN RULES TO CREATE ONE OR MORE JOINED RECORDS
- IN SOME CASES, RECORDS MAY BE WRITTEN TO A DS
- RECORD IS WRITTEN THEN FIELDS ARE SPLIT UP AND WRITTEN TO RESPECTIVE COMPONENT DATABASES

/

F02-08-86

EXAMPLE 1: HOW A JOIN WORKS



2

F02-08-86

WHEN YOU OPEN JOINTEST AND SELECT ISN 20:

- ROOT DATABASE (RDA) IS SEARCHED FOR ISN 20
- THIS RECORD IS RETRIEVED
- CONTENTS OF FIELD A200 ARE USED AS A KEY TO SEARCH RDC FOR X100 WITH THE SAME VALUE (IN THIS CASE, abcd)
- THIS SEARCH IS A FAST ACCESS SEARCH
- JOIN FIELD FROM RIGHT-HAND COMPONENT MUST BE

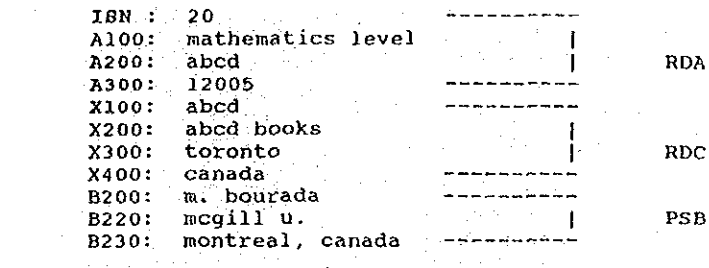
- ISN (IN A MASTER DATA BASE)
OR - INVERTED FIELD (IN A MASTER DATA BASE)
OR - PRIMARY OR SECONDARY KEY (IN A KSAM DATA BASE)

- FIELD A300 IN RESULT OF 2ND JOIN RULE IS USED AS A KEY TO RETRIEVE A RECORD WITH ISN 12005 FROM PSB

3

F02-08-86

FINAL RECORD AFTER ALL JOIN RULES ARE PROCESSED



- AFTER EXECUTION OF ALL JOIN RULES ALL FIELDS ARE PRESENT IN THE DS USER VIEW, UNLESS A PROJECT LIST IS DEFINED
- IF SO, ONLY FIELDS IN THE PROJECT LIST MAY BE READ

JOINEE FIELD ELIMINATION
WHEN THE JOIN FIELDS HAVE THE SAME TAG, ONE OF THE OCCURRENCES WILL BE ELIMINATED FROM THE DS RECORD

4

F02-06-86

JOIN RULES

- RULES TO BE APPLIED WHEN CREATING A DS

- EACH JOIN RULE NAMES A PS OR RD TO BE JOINED TO RESULT OF OF PREVIOUS JOINS

EX: DS1
DB1
A110 + B960 DB2
B300 + B770 DB3

WEIGHTING FORMULA LIMITS	
NO. OF DATA BASES WHICH CAN BE JOINED IN A DS	
DATA BASE	WEIGHT
DS ITSELF	1
RD	1
PS	2
*FLATTEN	1
MAXIMUM WEIGHT ALLOWED=17	

- RESULTING RECORD IS ALWAYS IN MASTER FORMAT

F02-08-86

DS ATTRIBUTES

HOW TO DEFINE A DS:

HEADER LEVEL

	DEFAULT VALUES
TYPE OF DATA DEFINITION (RD/PS/DS/CD)	RD
GROUP NAME	PUB
WRITE ACCESS TO DS (Y/N)	NO
IF YES	
AUTO-NUMBERING (Y/N)	NO
DYNAMICALLY CREATE DS COMPONENT (Y/N) ..	NO
BIBLIOGRAPHIC LEVEL INDICATOR	NO
LOGGING ACCESS (Y/N)	NO
USER EXIT TO VALIDATE ENTIRE RECORD ..	NO
DEFAULT QUERY FIELD TAG	NONE
DEFAULT PRINT FORMAT FILENAME	NONE
REGISTER ACCESS IN AUDIT TRAIL FILE	NO
INITIAL RESTRICTION	NONE

F02-08-86

IF WRITE ACCESS FOR DS IS YES:

AS YOU DEFINE EACH COMPONENT YOU WILL BE ASKED:
UPDATE ACCESS FOR COMPONENT

- IF THE REPLY IS NO
- NO DATA MAY BE WRITTEN THROUGH COMPONENT EVEN THOUGH WRITE ACCESS FOR THE DS IS YES

- IF THE REPLY IS YES AND THE COMPONENT IS AN RD
- DATA MAY BE WRITTEN THROUGH THIS COMPONENT

- IF THE REPLY IS YES AND THE COMPONENT IS A PS
- DATA MAY BE WRITTEN ONLY IF WRITE ACCESS IS ON FOR PS

- THIS PERMITS/PROHIBITS UPDATE ACCESS ON SELECTED COMPONENTS IN THE DS

F02-08-86

DEFINING JOIN RULES IN DS SUBCOMMAND HANDLER

TO CREATE A JOIN RULE:

INPUT COMMAND - i
FIRST COMPONENT - <dbname>
UPDATE ACCESS TO COMPONENT - yes OR no

TO CREATE SUBSEQUENT JOIN RULE(S):

INPUT COMMAND - i
NEXT COMPONENT - <dbname>
TYPE OF JOIN - LEFTINNER, LEFTOUTER, UNION OR INTERSECTION
SUBJOIN - yes OR no
(IF yes, TYPE OF JOIN IS IGNORED)
FIELD TAG IN PRECEDING RESULT - field tag
SOURCE OF THIS FIELD - <dbname>
FIELD TAG IN THIS COMPONENT - right-hand field

TO FLATTEN FIELD IN PRECEDING COMPONENT:

INPUT COMMAND - i
NEXT COMPONENT - *FLATTEN
FIELD TO BE FLATTENED IN PREVIOUS RESULT - tag
SOURCE OF THIS FIELD - <dbname>

F02-08-86

EXAMPLE 2

```
                        DS1
   RD1
          A100    <+    ISN    RD2

RD1                              RD2
  ISN :  1                       ISN :  2
  A100:  2 ------------------    X400:  smith
  A300:  mary
```

RESULT

```
DS1
  ISN :  1
  A300:  mary
  X400:  smith
  A100:  2
```

9

F02-08-86

EXAMPLE 3

```
                        DS2
   RD1
          A100    <+    A200   RD3

RD1                              RD3
  ISN :  1                       A200:  ab
  A100:  ab ------------------   A800:  smith
  A300:  mary
```

RESULT

```
DS2
ISN :  1
A300:  mary
A800:  smith
A100:  ab
A200:  ab
```

10

F02-08-86

TYPES OF JOINS

ROOTED JOINS

<+ - LEFT OUTER JOIN
- MOST COMMON TYPE
- IF JOIN IS NOT PERFORMED, ROOT RECORD IS RESULT

<* - LEFT INNER JOIN
- JOINS INTO A SUBFIELDED FIELD IN PRECEDING RESULT

* - NATURAL (OR INTERSECTION) JOIN
- BOTH COMPONENTS MUST OCCUR
- RESULTING RECORD IS AN AND OF BOTH COMPONENTS

NON-ROOTED JOINS

\+ - UNION JOIN
- ACCESS TO RECORD MUST BE KEY
- SUPPORTED ONLY WHEN JOIN FIELD IS ISN
- RESULTING RECORD IS AN OR
- EG. EITHER COMPONENT MAY OCCUR

11

F02-08-86

PROJECT LIST

WHEN ALL JOIN RULES ARE DEFINED, YOU WILL BE ASKED TO DEFINE A PROJECT LIST

CREATE OR MODIFY A PROJECT LIST
- DEFAULT IS NO
  I.E. ALL FIELDS IN ALL COMPONENTS ARE PRESENT IN DS

TO CREATE OR MODIFY A PROJECT LIST REPLY YES TO PROJECT LIST PROMPTS:

FIELD TAG
- FIELD FROM ONE OF THE COMPONENTS

SOURCE OF THIS FIELD
- COMPONENT NAME

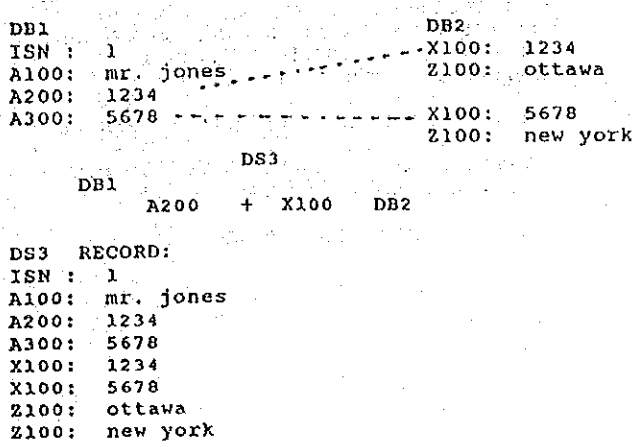
ANY MORE FIELD TAGS
- DEFAULT IS NO

12

PROJECT LIST (continued)

BECOMES REPEATING (Y/N)
- IF, BECAUSE MORE THAN ONE RECORD FROM 2ND COMPONENT MATCHES ROOT RECORD, A NON-REPEATABLE FIELD HAS SEVERAL OCCURRENCES IN THE JOIN RECORD, THE FIELD MUST BE REDEFINED AS REPEATABLE.

EXAMPLE 4



PROJECT LIST (continued)

OTHER FIELD ATTRIBUTES THAT MAY BE REDEFINED ARE:

PROMPT (Y/N)

REDEFINE INVERSION CRITERIA (Y/N)
- FIELDS MAY BE INVERTED IN DS PROJECT LIST WHERE THEY ARE NOT INVERTED IN RD OR PS, OR VICE VERSA
- INVERSION ATTRIBUTES MAY DIFFER FROM THOSE IN RD OR PS

EACH FIELD TAG SHOULD OCCUR ONLY ONCE

EACH FIELD TAG SHOULD BE A VALID FIELD IN THE SOURCE DATABASE

WHEN EDITING AN ALREADY DEFINED PROJECT LIST, YOU MAY INSERT OR DELETE FIELDS

TYPES OF JOINS

1) LEFT OUTER +
2) LEFT INNER *
3) NATURAL OR INTERSECTION *
4) UNION +
5) SUBJOIN

1) LEFT OUTER JOIN +

- MOST COMMON TYPE OF JOIN

- ROOTED JOIN
  I.E. A RECORD MUST BE OBTAINED FROM ROOT COMPONENT BEFORE JOIN IS EXECUTED

- IF JOIN FIELD IN ROOT RECORD DOES NOT OCCUR, NO JOIN IS EXECUTED

- RESULT IS SIMPLY ROOT RECORD

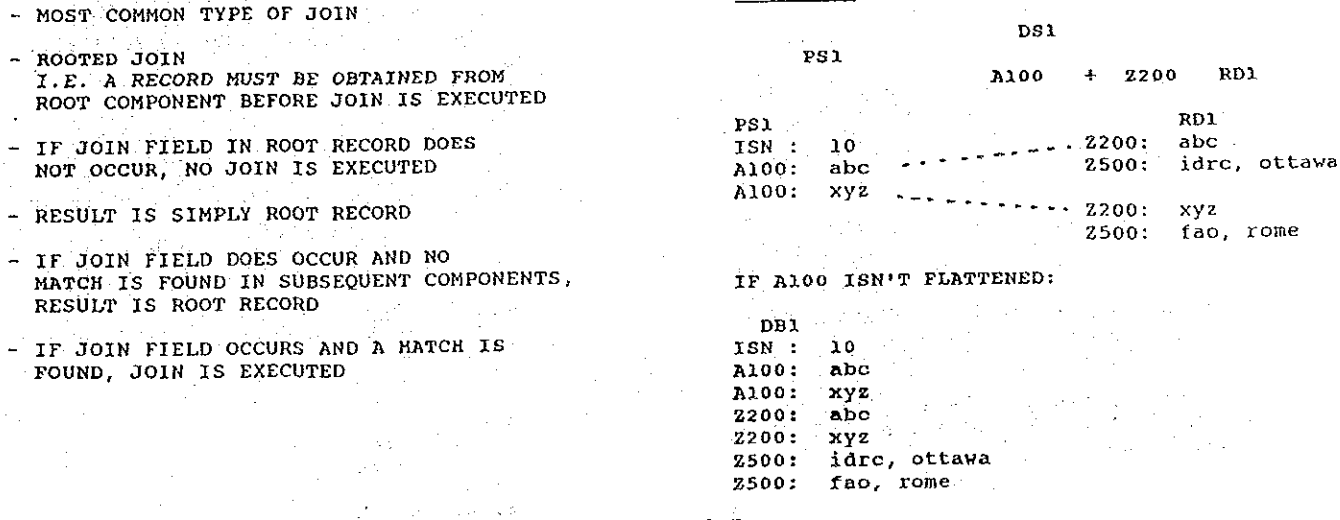
- IF JOIN FIELD DOES OCCUR AND NO MATCH IS FOUND IN SUBSEQUENT COMPONENTS, RESULT IS ROOT RECORD

- IF JOIN FIELD OCCURS AND A MATCH IS FOUND, JOIN IS EXECUTED

LEFT OUTER JOIN (continued)

IF THE LEFT-HAND JOIN FIELD IS REPEATABLE AND FLATTENED, THE RESULT IS A SET OF FLATTENED RECORDS. EACH OCCURRENCE OF THE FIELD WILL RESULT IN A RECORD - ALL FLATTENED RECORDS WILL HAVE THE SAME ISN.

EXAMPLE 5



F02-08-86

EXAMPLE 5 (continued)

IF A100 IS FLATTENED:

```
DB1
ISN :  10
A100:  abc
Z200:  abc
Z500:  idrc, ottawa

ISN :  10
A100:  xyz
Z200:  xyz
Z500:  fao, rome
```

17

F02-08-86

THE RESULT OF A ROOTED JOIN CAN BE USED AS THE ROOT FOR ADDITIONAL JOINS:

```
                DB1
   PB1
        A200    +   B300    RD2
        A300    +   X100    RD3
        X990    +   N700    RD4
```

RECORDS MAY BE WRITTEN/UPDATED THROUGH A LEFT OUTER JOIN.
ALL FIELDS ARE ENTERED THROUGH DS AND ARE WRITTEN TO THEIR RESPECTIVE COMPONENTS. THE RULES ARE:

- LEFT-HAND JOIN FIELD MUST BE SINGLE
- WRITE ACCESS MUST BE ON FOR DS AND AT LEAST ONE COMPONENT
- NO FLATTENING IS PERFORMED

18

F02-08-86

JOINING ON FAST ACCESS FILES

- THIS IS A SPECIAL CASE

```
                DS1
  RDA
        X110    +   Y120   RDB   (master)
```

WHERE Y120 IS AN INVERTED FIELD

- FOR THE JOIN TO EXECUTE, THE CONTENTS OF X110 MUST MATCH A KEY IN FAST ACCESS FILE FOR Y120
- MORE THAN ONE POSTING MAY BE FOUND FOR THIS KEY
- IF X110 IS SINGLE, AND MORE THAN 1 RECORD FROM RDB IS FOUND, RESULT IS A SET OF VIRTUALLY FLATTENED RECORDS
- IF X110 REPEATS AND IS FLATTENED, RESULT IS FLATTENED TWICE REAL AND VIRTUAL
- IF X110 REPEATS AND ISN'T FLATTENED, RESULT IS JOIN OF ROOT RECORD AND FIRST JOINEE RECORD FOUND

F02-08-86

VIRTUAL FLATTENING

- NO FLATTENING FLAG IS SET, BUT RESULT OF JOIN PRODUCES A SET OF FLATTENED RECORDS
- VIRTUAL FLATTENING IS RESULT OF A JOIN ON FAST ACCESS FIELD, WHERE MORE THAN ONE RECORD IS FOUND IN SECOND COMPONENT, AND ROOT FIELD IS SINGLE, OR REPEATS BUT IS FLATTENED

EXAMPLE 6

```
RDA                          RDB
ISN :  1                     ISN :  10
X100:  idrc                  X190:  mary
X110:  smith ..........      X120:  smith
                      \
                       \     ISN :  12
   DS 1                 \    X190:  john
ISN :  1                     X120:  smith
X100:  idrc
X110:  smith
X190:  mary
X120:  smith

ISN :  1
X100:  idrc
X110:  smith
X190:  john
X120:  smith
```

20

F02-08-86

RULES FOR JOINING ON FAST ACCESS

DS JOIN1

PS1

Y120 + X220 RD2

1) DATA IN ROOT JOIN FIELD MUST MATCH KEY IN FAST ACCESS FILE LINKED TO RIGHT-HAND JOIN FIELD
   - EG. IF KEY IS STRIPPED OF DIACRITICAL ACCENTS, SO MUST BE DATA IN ROOT JOIN FIELD
   - EXCEPTION IS UPSHIFTING
   - DATA IN LEFT-HAND FIELD MAY BE IN LOWER CASE

2) RESULTS WILL DIFFER DEPENDING ON WHETHER ROOT FIELD IS SINGLE OR REPEATABLE, FLATTENED OR UNFLATTENED.

21

F02-08-86

LEFT INNER JOIN *

- ANOTHER ROOTED JOIN
- WORKS ONLY ON SUBFIELDED FIELDS
- NO UPDATE ACCESS
- SUBFIELDED FIELD HEADER MUST EXIST IN ROOT COMPONENT
- ROOT COMPONENT IS MASTER
- SUBSEQUENT COMPONENT MAY BE KSAM OR MASTER

EXAMPLE 7

```
RD1                          RD2  (KSAM)
  ISN:  1                  X100:  idrc
  A220                     X200:  ottawa
    A221:  idrc            X300:  canada
  A220
    A221:  fao
                           X100:  fao
                           X200:  rome
                           X300:  italy
```

22

F02-08-86

LEFT INNER JOIN (continued)

TO JOIN 2 COMPONENTS, OPTIONS ARE:

1) LEFT OUTER JOIN OF A221 TO X100

- X200 AND X300 BECOME REPEATABLE FIELDS:

EXAMPLE 7.1

```
ISN:  1                  JOINEE FIELDS
A220                        ARE NOT
A221:  idrc               GROUPED
A220                        CORRECTLY
A221:  fao                    |
X200:    ottawa          X200:  rome
X300:    canada          X300:  italy
```

F02-08-86

LEFT INNER JOIN (continued)

2) LEFT OUTER JOIN OF A221 TO X100 WHERE A220 IS FLATTENED

- ROOT COMPONENT MUST BE A PS

- RESULTS IN A SET OF FLATTENED RECORDS:

EXAMPLE 7.2

```
ISN:  1
A220:
A221:  idrc
X200:   ottawa
X300:   canada

ISN:  1
A220:
A221:  fao
X200:  rome
X300:  italy
```


24

F02-08-86

LEFT INNER JOIN (continued)

3) LEFT INNER JOIN WHERE KSAM FIELDS HAVE BEEN REDEFINED AS SUBFIELDS OF A220

EXAMPLE 7.3

IF RD2 CONTAINS:

```
A221:  idrc          A221:  fao
A222:  ottawa        A222:  rome
A223:  canada        A223:  italy
```

AND JOIN IS:

```
            DS1
RD1
      A221    *   A221   RD2
```

RESULT IS:

```
ISN:  1
A220
 A221: idrc    A222: ottawa    A223: canada
A220
 A221: fao     A222: rome      A223: italy
```


F02-08-86

INTERSECTION OR NATURAL JOIN *

- ROOTED JOIN
- PERFORMS AN AND OF 2 COMPONENTS
- A DS RECORD IS PRODUCED ONLY IF EXECUTED
- IF NO MATCH IS FOUND IN RIGHT HAND COMPONENT FOR DATA IN LEFT-HAND JOIN FIELD, NO RESULTS (NOT EVEN A ROOT RECORD)

EXAMPLE 8

```
            DS1
RD1
    A100    *    D100    RD2
```

IF:

```
RD1                    RD2
ISN:  1                D100:  abcd
A100:  abcd            D200:  abcd books
```

A RECORD IS PRODUCED

IF:

```
RD1                    (NO RD2 RECORD
ISN:  2                WITH wxyz IN KEY)
A100:  wxyz
```

NO RECORD IS PRODUCED

F02-08-86

UNION JOIN +

- ONLY SUPPORTED WHEN RIGHT-HAND FIELD IS ISN
- NON-ROOTED JOIN
- A RECORD WILL BE PRODUCED IF KEY IS FOUND IN EITHER COMPONENT
- IF ROOT COMPONENT IS KSAM, VALUE IN PRIMARY KEY MUST BE NUMERIC
- IT BECOMES ISN OF RESULTING RECORD

EXAMPLE 9

```
KSAMRD                          MASTERRD
 W010:  3                            ISN = 1
 W020:  rome, italy                  A100:  idrc

 W010:  4                            ISN = 2
 W020:  new york, usa                A100:  aiba

 W010:  5                            ISN = 3
 W020:  geneva, switzerland          A100:  fao
```

```
                    DS1
KSAMRD
            W010   +   ISN    MASTERRD
```

F02-08-86

EXAMPLE 9 (continued)

- ALL RECORDS ARE FOUND IN DS1:

```
ISN = 1
A100:idrc

ISN = 2
A100:aiba

ISN = 3
A100:fao
W020:rome, italy

ISN = 4
W020:new york, usa

ISN = 5
W020:geneva, switzerland
```

F02-08-86

SUBJOIN

- USED TO LIMIT NO. OF RECORDS FOUND BY PRECEDING JOIN RULE
  JOIN RULE
- UPDATE ACCESS IS NOT PERMITTED

```
                    DS1
  RD1
          B550    +   B700    RD2   (1)
          B600    +   B900    RD2   (2)
```

- AFTER (1) IS EXECUTED, RECORDS FOUND ARE SCANNED TO SEE IF (2) IS TRUE
- ONLY RECORDS KEPT ARE THOSE WHERE B600 MATCHES B900
- IF NO MATCH IS FOUND, THERE ARE NO RESULTS (NOT EVEN A ROOT RECORD)
- IF LEFT-HAND SUBJOIN FIELD REPEATS, USE FLATTENING - OTHERWISE ONLY FIRST OCCURRENCE IS SCANNED

F02-08-86

CHAINED JOINS

- POSSIBLE WITH ROOTED JOINS
- JOINS A DATA BASE TO ITSELF

```
                DS1
  RD1
        A100    +   ISN   RD1
```

- USEFUL IF A RECORD IS GREATER THAN 4096 CHARACTERS
- ALLOWS YOU TO BREAK RECORD INTO 2 OR MORE SECTIONS

30

F02-08-86

EXAMPLE 10 (chain join)

```
RD1
ISN :  1          ISN :  2          ISN :  3
A100:  2          A100:  3          A100:  -
A200:  mary       A400:  idrc       A500:  ottawa
A300:  smith                        A600:  canada

DS1 RECORD IS:

ISN :  1
A200:  mary
A300:  smith
A400:  idrc
A500:  ottawa
A600:  canada
```