DEVELOPMENT OF A TEXI-EDITOR BASED RELATIONAL DATA BASE MANAGEMENT SYSTEM (U) FLORIDA A AND M UNIV TALLAHASSEE DEPT OF DATA PROCESSING T W MASON AUG 81 UNCLASSIFIED AFOSR-TR-83-0665 AFOSR-81-0131 F/G 9/2 NL
Development of a Text-Editor based Relational Database Management System

Grant No. AFOSR-81-0131
June 1, 1981 - August 31, 1981

Thomas W. Mason
Florida A&M University
Tallahassee, FL 32307
Database management systems have historically been the domain of large mainframes. However, the popularity of mini and micro-computers has spurred the development of database systems appropriate for those devices. Concurrently, database systems design is turning away from traditional hierarchic and CODASYL models to embrace the conceptually simpler relational database approach (1,2,3,4).

The relational database approach views data as being in tables. The entries form the rows and are called tuples. The columns are called (CONTINUED)
ITEM #20, CONTINUED: Attributes. Simple selection commands are provided to search for entries with attributes of a given value. Other commands allow the extraction of a subset of entries (found by the selection commands) and the incorporation of that subset with others. This approach is not as efficient as the traditional model in its implementation but offers far greater flexibility in the incorporation of data and the ability to 'explore' the data base.

This report documents the attempt to develop a relational database management system for the Harris Minicomputer at Florida A&M University.
1.0. INTRODUCTION

Database management systems have historically been the domain of large mainframes. However, the popularity of mini and micro-computers has spurred the development of database systems appropriate for those devices. Concurrently, database systems design is turning away from traditional heirarchic and CODASYL models to embrace the conceptually simpler relational database approach (1,2,3,4).

The relational database approach views data as being in tables. The entries form the rows and are called tuples. The columns are called attributes. Simple selection commands are provided to search for entries with attributes of a given value. Other commands allow the extraction of a subset of entries (found by the selection commands) and the incorporation of that subset with others. This approach is not as efficient as the traditional model in its implementation but offers far greater flexibility in the incorporation of data and the ability to "explore" the data base.

This report documents the attempt to develop a relational database management system for the Harris Minicomputer at Florida A&M University.

2.0 SYSTEM DESIGN

It is unusual for the design of database management systems to be described in detail. Therefore, the description of RISS, a relational database management system for minicomputers (5), seemed a most fortuitous occurrence. The system development strategy was to translate RISS to the Harris computer and then augment its capabilities in accordance with the original project proposal.
2.1 RISS: RELATIONAL INQUIRY AND STORAGE SYSTEM

RISS was developed at the forest Hospital in Des Plaines, Illinois and implemented on a PDP 11/40 with the RSTS/E operating system. Data can be retrieved from RISS by human interaction from a terminal (called the naive-user interfaced level) or by requests from a computer program (called the applications program interface level).

The conversion effort focussed solely on the properties of the naive user interface level. Three subsystems provide the pathway for a human user to access a RISS database a relational editor, a retrieval package and a database manipulation and maintenance package. They are described below:

A. Relational editor

The editor is needed to create, examine and update entries in the database. The editor design is based on single-oriented text edition. Thus, there is a record pointer which identifies the entry to be entered, examined or changed. The editor commands implemented in RISS are:

1. move the record pointer forward in backward through the existing entries
2. search for a specified substring in the database and move the record pointer to the next occurrence of that substring.
3. delete one or more entries from the database.
4. enter a new entry in the database.
5. display or change the value of a field (attribute) of an existing entry.
6. provide descriptive information about a given set of entries (relation) in the database.

B. Retrieval Package

The retrieval commands allow the user to retrieve and analyze data in RISS relations:

1. selection of entries which satisfy a given attribute specification (i.e., Sex = "male", Age = 18) and the formation of a resulting relation.
2. formation of the union or intersection of relations, including relations formed by the proceeding command
3. extraction of a subset of columns (attributes) of a relation
4. Printing tabular reports based on a retrieved relation
5. Printing simple statistical information
6. grouping specific attribute values into user-specified ranges
7. producing a frequency distribution for all unique values of a given attribute.

c. Database manipulation and maintenance package

The database manipulation and maintenance package provide the usual utility functions associated with database management:

1. creating a relation (a set of related entries in database)
2. deleting a relation
3. copying a relation
4. sorting a relation
5. merging two relations
6. combining two relations
7. redefining the structure of a relation by adding or deleting a column (attribute)

3.0 SYSTEM IMPLEMENTATION

The initial implementation strategy was to copy all routines from the RISS text into the Harris computer. It was thought that the differences in the BASIC language between the two systems would be minor. This proved not to be the case.

The implementation of BASIC on the two computers differs significantly. Furthermore, closer investigation revealed that the RISS data structure design incorporated fundamental features of the PDP-11/40. In other words, to use the RISS code directly would entail the emulation of PDP-11/40 features on the Harris minicomputer. To compound the problem to the breaking point, at this time the air conditioning system of the Harris minicomputer broke down and remained inoperative for several weeks. This exclude computer use during the time although program development was possible.
In light of the problems discussed above, it was decided to re-design the FAMU Relational Database system (FREDB) using RISS as a guide but exploiting the features of the Harris minicomputer. The central core of the re-design was to abandon the RISS data structure.

The RISS data structure allowed for storage of four types of data - single ASCII characters, integers, floating-point numbers and alphanumeric character strings. Three files or tables were used for storage and descriptions - a tuple (entry) descriptor table, a tuple file and an alpha data file. Without going into a detailed description of the process, suffice it to say that the storage structure was based very closely on the actual storage format of the PDP-11/40.

Upon reconsideration of the design, questions began to arise as to the need for a relational editor. The reasoning was that an edition already existed in the Harris operating system. It could be used for all of the functions of the RISS relational editor. However, the use of the Harris line editor implied the use of the Harris storage strategy. Hence, one simplification led to another. FREDB has no provision for relational editing. The system recognizes the equivalence of flat files and relations. Therefore, the user enters, modifies and deletes all data using existing Harris editing procedures. The relations are then described in FREDB and those field (attribute) descriptions form a pathway for FREDB routine to access user-created files. The adoption of this approach led to the development of an initial system with many of the features of RISS.
4.0. The FREDB System

The actual implementation consists of a method to define and create relations. Additional procedures to join and select relations were attempted but not completed. The computer programs for relation definition and definition (called CREREL) are given in Appendix 1.
Bibliography

1. Codd, E.F., "Recent investigations in relational data base system," Information Processing '74, North Holland, Amsterdam, 1974


SMS
1MO BS=1000
$MO NU
$SRR,SRN
$AS 10 = OUTFILE
$SRN #NN = 0
$SRN #NAME = NAME
$SRF #NAME = 10
$AS 20 = #NAME
SJE IGEN
ILAB2 $$$$ RELATION
$SRN #NMA = 0
$SRN #EE = 1
$SRF #NMA = 10
$$$ RELATION
$MO NREG = 50
$SRN #DL = 0
$SRN #END = 0
$SRN #BEG = 0
$SR #VAL =
$PR
$PR HOW MANY RECORDS ARE YOU INSERTING?
$SRN #TPIN = 0
$SRN #TPIN
SJE IKEEP
IKEEP $$$$ FLAG DEVICE
$$ THIS SECTION EDITS RELATION AND
$$ INSERTS BLANK RECORDS FOR INSERTION.
$ED #NMM RE
$PE 0
AE 1000000
SJE IOUT
IOUT $BE = 1
$E = 0
$SRN #LIN = ERM()
$ED #NMM AB
$SRN #IN = 0
$$
ILAB IN #LIN BLANK, 1
$SRN #IN = #IN + 1
IF ( #IN = #TPIN ) SJU IOUT2
SJU ILAB
IOUT2 $$$
THE INSERTION OF ATTRIBUTES BEGINS
$$
$UP
$ED #NMM AB
$SRN #LIN = #LIN + 1
$E = #LIN
AGAIN $$$
$SRF #VAL = 10
$SRF #BEG = 10
$SRF #END = 10
$PR
$PR ATTRIBUTE #EE
IREN $$$
$PR #VAL
$SR #IN = NULL
$SRN #IN
SJE 178 INPO
INPO $$$ INSERT BLANK CHAR
SIF ( #IN = #END ) SJU IOUT4
$$
$SRN #CHE = #END - #BEG + 1
$IF ( #CHE > #CHE ) $JU IKEN
$C #BEG=#END,#IN
$IF ( #EE = #NNN ) $JU IOUT3
$SR,N #EE = #EE + 1
$JU IAGAIN

IOUT3 $$$ FIRST Attribute INSERTED
$SR,N #EE = 1
$SR,N #NN = #NN + 1
$SR,N #LIN = #LIN + 1
$RM 10
$SR,F #NMM=10
$SR,F #NMA = 10
$IF ( #NN = #TPIN ) $JU IOUT4
$E #LIN
$SPR * INSERT SEND ON NEXT ENTRY TO TERMINATE.
$SPR ---- DATA ENTRY..
$JU IAGAIN

IOUT4 $$$ * IF SEND WAS ENTERED THIS SECTION WILL DELETE
$$$ THE REMAINING LINES THAT WAS ORGINALLY REQUESTED.
$IF ( #NN = #TPIN ) $JU IOUT5
$IF ( #NN < #TPIN ) $DE #LIN
$SR,N #DL = #DL + 1
$SR,N #NN = #NN + 1
$SR,N #LIN = #LIN + 1
$JU IOUT4

IOUT5 $$$ THIS SECTION TERMINATES THE DATA ENTRY PROCESS.
$UP
$SR,N #LIN = #LIN = #DL
$SR,N #LIN = #LIN = 2
$SPR
$SPR
$SPR RELATION NAME = #NMM :
NUMBER OF RECORDS = #LIN

$ME IGEN $$$
$SPR RELATION #NMM NOW BEING GENERATED
$GE #NMM
$C0 BLANK #NMM
$JU ILAB2
$ME

IKEN $SPR -ERROR GENERATED STRING LENGTH OF #CHE EXCEEDED
$JU IREN
IDENTIFICATION DIVISION.
PROGRAM-ID. GET-A-RELATION.

AUTHOR. ARTHUR ROBERTS JR.
DATE-WRITTEN. MARCH 10, 1982.

ENVIRONMENT DIVISION.
CONFIGURATION SECTION.
SOURCE-COMPUTER. HARRIS-123.
OBJECT-COMPUTER. HARRIS-123.

INPUT-OUTPUT SECTION.
FILE-CONTROL.
SELECT UNIT-OUTPUT-FILE ASSIGN "OUTFILE".
SELECT UNIT-INDEX-FILE ASSIGN "INDEXLOG"
ORGANIZATION IS RELATIVE
ACCESS MODE IS SEQUENTIAL
RELATIVE KEY IS REC-POS.

DATA DIVISION.
FILE SECTION.
FD UNIT-OUTPUT-FILE
DATA RECORD IS OUT-REC.
  01 OUT-REC.
  02 FILLER PIC X(80).
FD UNIT-INDEX-FILE
DATA RECORD IS INDEX-REC.
  01 INDEX-REC.
  02 REL-NAME PIC X(8).
  02 INFOMAT PIC X(991).

WORKING-STORAGE SECTION.
  77 REC-POS PIC 9(5) VALUE 1.
  77 REC PIC X(3) VALUE SPACES.
  77 TREL-NAME PIC X(8) VALUE SPACE.
  77 COND PIC X(1) VALUE "N".
  77 SUB PIC 999 VALUE 0.
  01 OUTPUT-DATA.
  02 DATA-LINE.
    03 FILLER PIC X(991) VALUE SPACES.
  02 DATA-OUT REDEFINES DATA-LINE.
    03 FILLER PIC X.
    03 NO-OF-ATT PIC 9(3).
    03 OCC-OF-ATT OCCURS 34 TIMES.
    04 FILLER PIC X.
    04 ATT-NAME PIC X(20).
    04 FILLER PIC X.
    04 BEGIN PIC X(3).
    04 FILLER PIC X.
    04 END PIC X(3).
  03 FILLER PIC X.

PROCEDURE DIVISION.
PROCESS-CONTROL.
OPEN OUTPUT UNIT-OUTPUT-FILE
CLOSE
UNIT-OUTPUT-FILE
UNIT-INDEX-FILE.

STOP RUN.

MAIN-Routine.
MOVE "N" TC COND1.
DISPLAY "INPUT RELATION NAME? ",
DISPLAY " THIS NAME CAN ONLY BE 8 CHARACTERS LONG. ".
DISPLAY " NO SPECIAL CHARACTERS OR BLANKS. ".
ACCEPT TREL-NAME FROM TERMINAL.

DISPLAY " ",
DISPLAY " RELATION NAME =====> ", TREL-NAME.
DISPLAY " ",
DISPLAY " IS THIS INFORMATION CORRECT? (Y OR N) ".
DISPLAY " ",
ACCEPT RES FROM TERMINAL.
IF RES = "Y"
DISPLAY " OK ",
PERFORM CHECK-RELATION-NAME THRU CHECK-RELATION-NAME-EXIT
IF COND1 = "Y"
MOVE INFOMAT TO OUTPUT-DATE
WRITE OUT-REC FROM REL-NAME
WRITE OUT-REC FROM NO-OF-ATT
PERFORM WRITE-ROUTE
VARYING SUB FROM 1 BY 1 UNTIL SUB > NO-OF-ATT
ELSE
DISPLAY 'ERROR RELATION TREL-NAME 'DOES NOT EXIST '
DISPLAY '?
DISPLAY ' DO YOU WISH TO TRY AGAIN? (Y/N)? '
ACCEPT RES FROM TERMINAL
IF RES = 'Y'
PERFORM CLOSE-OPEN-FILE
GO TO MAIN-Routine
ELSE
NEXT-SENTENCE
ELSE
DISPLAY "***** ERROR *****"
GO TO MAIN-Routine.

MAIN-Routine-EXIT.
EXIT.

CHECK-RELATION-NAME.
READ UNIT-INDEX-FILE
AT END
GO TO CHECK-RELATION-NAME-EXIT.

IF REL-NAME = TREL-NAME
MOVE."Y" TO COND1
ELSE
GO TO CHECK-RELATION-NAME.

CHECK-RELATION-NAME-EXIT.
EXIT.

CLOSE-OPEN-FILE.
CLOSE UNIT-INDEX-FILE.
WRITE OUT-REC FROM ATT-NAME (SUB).
WRITE OUT-REC FROM BEG-C (SUB).
WRITE OUT-REC FROM END-C (SUB).

EOF.
BEGIN
$RR, $NR
$PR

INPUT THE NAME OF THE RELATION IN WHICH YOU WOULD LIKE COPIED?
$RR, I #NAME
$PR

#NAME IS THE RELATION IN WHICH YOU WOULD LIKE COPIED? (Y/N)
$RR, I #NAME
$PR

PLEASE TYPE IN 'Y' OR 'N'

WHAT IS THE NAME OF THE NEW RELATION?
$RR, I #NAME
$PR

#NAME IS THE NEW RELATION? (Y/N)
$RR, I #NAME
$PR

PLEASE TYPE IN 'Y' OR 'N'

WHAT IS THE NAME OF THE NEW RELATION?
$RR, I #NAME
$PR

#NAME IS THE NEW RELATION? (Y/N)
$RR, I #NAME
$PR

PLEASE TYPE IN 'Y' OR 'N'

WHAT IS THE NAME OF THE NEW RELATION?
$RR, I #NAME
$PR

#NAME IS THE NEW RELATION? (Y/N)
$RR, I #NAME
$PR

PLEASE TYPE IN 'Y' OR 'N'

WHAT IS THE NAME OF THE NEW RELATION?
$RR, I #NAME
$PR

#NAME IS THE NEW RELATION? (Y/N)
$RR, I #NAME
$PR

PLEASE TYPE IN 'Y' OR 'N'

WHAT IS THE NAME OF THE NEW RELATION?
$RR, I #NAME
$PR

#NAME IS THE NEW RELATION? (Y/N)
$RR, I #NAME
$PR

PLEASE TYPE IN 'Y' OR 'N'

WHAT IS THE NAME OF THE NEW RELATION?
$RR, I #NAME
$PR

#NAME IS THE NEW RELATION? (Y/N)
$RR, I #NAME
$PR

PLEASE TYPE IN 'Y' OR 'N'

WHAT IS THE NAME OF THE NEW RELATION?
$RR, I #NAME
$PR

#NAME IS THE NEW RELATION? (Y/N)
$RR, I #NAME
$PR

PLEASE TYPE IN 'Y' OR 'N'

WHAT IS THE NAME OF THE NEW RELATION?
$RR, I #NAME
$PR

#NAME IS THE NEW RELATION? (Y/N)
$RR, I #NAME
$PR

PLEASE TYPE IN 'Y' OR 'N'

WHAT IS THE NAME OF THE NEW RELATION?
$RR, I #NAME
$PR

#NAME IS THE NEW RELATION? (Y/N)
$RR, I #NAME
$PR

PLEASE TYPE IN 'Y' OR 'N'

WHAT IS THE NAME OF THE NEW RELATION?
$RR, I #NAME
$PR

#NAME IS THE NEW RELATION? (Y/N)
$RR, I #NAME
$PR

PLEASE TYPE IN 'Y' OR 'N'

WHAT IS THE NAME OF THE NEW RELATION?
$RR, I #NAME
$PR

#NAME IS THE NEW RELATION? (Y/N)
$RR, I #NAME
$PR

PLEASE TYPE IN 'Y' OR 'N'

WHAT IS THE NAME OF THE NEW RELATION?
$RR, I #NAME
$PR

#NAME IS THE NEW RELATION? (Y/N)
$RR, I #NAME
$PR

PLEASE TYPE IN 'Y' OR 'N'

WHAT IS THE NAME OF THE NEW RELATION?
$RR, I #NAME
$PR

#NAME IS THE NEW RELATION? (Y/N)
$RR, I #NAME
$PR

PLEASE TYPE IN 'Y' OR 'N'

WHAT IS THE NAME OF THE NEW RELATION?
Sup

Sel \#6
Sel \#7
Sel \#9

$free all
$pr \#nm has been copied into \#nm

!rest
$pr would you like to copy another relation? (y/n)
$sr, i \#ans

!if, (\#ans=n) $ju !end1
$je !err4
!if, (\#ans=y) $ju !begin
!err4
$pr
$pr please type in 'y' or 'n'...
$ju rest

!err33
$je !next33
!next33
$pr
$pr \#nm is an invalid relation name:
$pr
1) name has more than 8 characters
$pr
2) first character in the name is a number
$pr
3) name has an invalid character.
$pr

$pr
$ju !cont33
!next44
$je !next44
!next44
$pr
$pr \#nm is an invalid relation name:
$pr
1) name has more than 8 characters
$pr
2) first character in the name is a number
$pr
3) name has an invalid character.
$pr

$pr
$pr try again...
$pr
$ju !begin

!end1

!begin

eof..
100 AS 10 = "#2"
200 AS 20 = "OUTFILE"
400 AS 40 = "#3"
500 DIM INDX$(50), REL$(50), LABLS(99), BEG$(99), END$(99)
600 DIM BEG$(99), END$(99), FILES(3)
700 OPEN #20
1000 GOSUB 8100
5400 FOR FI = 1 TO 3
5500 START = 1
5600 FOR P = 1 TO ATT
5700 GOSUB 8900
5800 SIZE = (ENDD(P) - BEG(P)) + 1
5900 PRINT #10 TAB(11); "TAB(15); FILES$(FI); TAB(18); LABLS$(P);->
6000 TAB(50); "PIC X("
6910 IF SIZE < 10 DO
6920 PRINT #10 USING ";#"; TAB(56); SIZE;
6930 ELSE
6940 PRINT #10 USING ";#"; TAB(56); SIZE;
6950 DOEND
6960 PRINT #10 ");.";
6970 START = ENDD(P) + 1
6980 NEXT P
6990 IF FI = 1 DO
7000 PRINT #10
7100 PRINT #10 TAB(7); "FD OUT-FILE"
7200 PRINT #10 TAB(11); "DATA RECORD IS OT-REC."
7300 PRINT #10
7400 PRINT #10 TAB(7); "01 OT-REC."
7500 ELSE
7600 DOEND
7610 DOEND
7620 NEXT FI
7630 PRINT #40 TAB(19); FILES$(3); TAB(22); KEYS
8000 STOP
8100 INPUT #20 INRELS
8150 INPUT #20 ATT
8160 FOR P = 1 TO ATT
8170 INPUT #20 LABLS$(P)
8180 INPUT #20 BEG$(P)
8190 INPUT #20 END$(P)
8200 NEXT P
8300 FILES$(1) = "IN-
8400 FILES$(2) = "OT-
8500 FILES$(3) = "SI-
8510 PRINT
8520 PRINT "WHICH KEY(S) IN "; INRELS; " DO YOU WANT THE FILE IN? ->
8530 BE SORTED ON"
8700 INPUT "TYPE IN THE FIELD NAME(S) "; KEYS
8800 RETURN
8900 FIL = BEG$(P) - START
9000 IF FIL > 0 DO
9100 PRINT #10 TAB(11); "05 FIL"; TAB(50); "PIC X("
9105 IF FIL < 10 DO
9110 Print #10 USING ";#"; TAB(56); FIL
9115  DOEND
9120  PRINT #10 "."
9200  ELSE
9300  DOEND
9400  RETURN
9800  END
THE CREREL RELATION WAS CREATED TO SERVE AS A DATA ENTRY SYSTEM FOR FREDDB. THE VERSITILITY OF THIS SYSTEM WILL ALLOW THE USER TO HAVE A MAXIMUM RECORD SIZE OF 999 CHARACTERS, THE DATA FOR THIS SYSTEM MAYBE ENTERED IN ANY OF THE THREE (3) WAYS LISTED BELOW.

1). IF THE FILE ALREADY EXIST IT CAN BE ADDED TO THE INDEXLOG BY DEFINING THE RELATION NAME AND ATTRIBUTES USING CREREL.

2). DATA CAN BE ADDED AT THE SAME TIME THE RELATION NAME AND THE ATTRIBUTES ARE DEFINED.

3). DATA CAN ALSO BE ADDED TO THE FILE AT A LATER DATE.

THIS IS A SERIES OF ENTER ACTIVE COBOL PROGRAMS WHICH WILL PROMPT THE USER FOR THE FOLLOWING INFORMATION, TO CREATE A RELATION.

1). NAME OF RELATION TO BE CREATED.
   THIS NAME CAN NOT BE OVER 8 CHARACTERS LONG AND IT SHOULD NOT CONTAIN ANY SPECIAL CHARACTERS (%&*-?1234).

2). NUMBER OF ATTRIBUTES THE RELATION IS TO CONTAIN.
   THIS NUMBER CAN NOT EXCEED 34.

   ***** THE FOLLOWING INFORMATION WILL APPEAR *****
   ***** FOR EVERY ATTRIBUTE THAT IS REQUESTED *****

3). NAME OF ATTRIBUTE.
   THIS NAME CAN NOT EXCEED 20 CHARACTERS.

3.1). WHEN THE LETTER 'B' IS TYPED IN FOR THE ATTRIBUTE NAME IN THE ATTRIBUTE LENGTH FIELD TYPE THE NUMBER OF SPACES THAT IS TO BE PLACED BETWEEN THE ATTRIBUTES.
   THIS WILL NOT COUNT AS A ATTRIBUTE.

4). LENGTH OF ATTRIBUTE.
   THE MAXIMUM LENGTH OF THE ATTRIBUTE.

THE ENTIRE SYSTEM IS IED TOGETHER BY JCL.

FILENAME PURPOSE
CREATE
CREREL
CRE2
CREATE1
CREATE2

CREATE
CREATES A NEW RELATION

CREATE1
RETRIEVES THE ATTRIBUTES
FOR A GIVEN RELATION

CREREL
JCL FOR CREREL SYSTEM

CRE2
JCL FOR INSERTING OF ATTRIBUTES

---------- PURPOSE ---------- FILENAME
NEWRL CREATE A NEW RELATION CREATE
GETRL GET A RELATION CREATE1
EOF..
FAMU RELATIONAL DATABASE

NOTE: TO GET A LIST OF THE AVAILABLE MODULES IN THE 'FRED6' SYSTEM, TYPE IN:
FRED6.M

THESE ARE THE ONLY AVAILABLE MODULES IN THE 'FRED6' SYSTEM.

CREREL  EDREL
COPREL  COLREL
SORREL  MERREL
DELREL  RETREL

NOTE: IF YOU WISH TO GET A DESCRIPTION OF EACH MODULE, TYPE:
FRED6.O
$PR
$PR DELREL (RELATION DELETER): THIS MODULE ALLOWS A USER TO SELECTIVELY
$PR RETRIEVE DATA FROM A 'FREDB' RELATION.
Ims
FREE ALL
BBLK
BEGIN $RR, $NR
$MO BS=999
$PR
$PR INPUT THE NAME OF THE RELATION WHICH YOU WOULD LIKE DELETED?
$PR, I #NME
$PR
$PR #NME IS THE RELATION IN WHICH YOU WOULD LIKE DELETED? (Y/N)
BEGIN $RR, I #ANG
$IF, (#ANG=N) $JU BEGIN $JE IERR
$IF, (#ANG=Y) $JU ICONT
IERR $PR
$PR PLEASE TYPE IN 'Y' OR 'N'...
JU IRESTA
ICONT $SEL #NME
$JE, K, 312, IERR3
$JE IERR1
$PR
$JU IINDEX
IERR1 $PR
$PR #NME IS AN NON-EXISTING RELATION
$PR
$PR WOULD YOU LIKE TO DELETE ANOTHER RELATION? (Y/N)
JU IRESTA
INDEX $SMO RE
$ED INDEXLOG
$AE, 1-8, #NME
$DE 0
$UP
$SMO AB
$PR
FREE ALL
$PR #NME HAS BEEN DELETED
$PR
$PR WOULD YOU LIKE TO DELETE ANOTHER RELATION? (Y/N)
BEGIN $RR, I #ANG1
$IF, (#ANG1=N) $JU IEND
$JE IERR2
$IF, (#ANG1=Y) $JU BEGIN IERR2
$PR
$PR PLEASE TYPE IN 'Y' OR 'N'...
JU IRESTA
IERR3 $JE INEXT3
INEXT3 $PR
$PR #NME IS AN INVALID RELATION NAME:
$PR
$PR 1) NAME HAS MORE THAN 8 CHARACTERS
$PR 2) FIRST CHARACTER IN THE NAME IS A NUMBER
$PR 3) NAME HAS AN INVALID CHARACTER.
$PR
$PR TRY AGAIN...
$PR
$JU BEGIN IEND 92
EOF..
$MS
$  ************************************************************
$  ** THIS MACRO RETRIEVES A RELATION FROM THE **
$  ** FAMU RELATIONAL DATABASE < FREDB >  **
$  ************************************************************
!BEGIN $$
BBLK
B2
B2
$PR THIS IS THE RELATION RETREIVING MODULE
$PR
$PR
$PR
$PR ALL
$  ******************************************************
$  * GETRL## IS THE LINK MODULE FROM A COBOL -*
$  * PROGRAM THAT SEARCHES FOR A RELATION
$  * ASSIGNMENTS ARE MADE INTERACTIVE TO THE *
$  * SAME FILE IN THE COBOL PROGRAM < TREL > -*
$  * GOMIT## IS THE EXECUTABLE LINK MODULE OF A *
$  * COBOL PROGRAM < TREL > THAT RETRIEVES*
$  * A wanted attribute.*
$  ******************************************************
GETRL
$AS-20 = OUTFILE
$BRLF *RRR = 20
$JE INOREC
$FR ALL
$AS 20 = *RRR
GOMIT
$JU INOREC
INOREC $$$
$  ******************************************************
BBLK
$PR WOULD YOU LIKE TO TRY AGAIN
$SR, I #RES
$IF (#RES = Y-) JU !BEGIN
BBLK
B2
B2
!STOP $$$
$PR  **** THIS IS THE END OF THE RETREL RELATION ****
$PR
$PR
$PR
$PR  **** THANK YOU ****
$ME
EOF.
THIS COLREL MODULE ALLOWS YOU THE USER TO DELETE ATTRIBUTES (COLUMNS) FROM AN EXISTING RELATION.

TO DELETE A COLUMN FROM A RELATION THE RELATION NAME AND ATTRIBUTE NAME MUST BE TYPED IN WHERE SPECIFIED IN THE MODULE.

PLEASE INPUT THE RELATION NAME>

IS THIS THE CORRECT RELATION NAME? Y/N

PLEASE ENTER (Y) FOR YES OR (N) FOR NO.

IF (#R=Y) JU ICON JU IERR ICON AS 20=#N
SAS.10=INPUT
$PR,F,10,#N COBOL "COL2"
SVX EL #N RN OUTREL #N GE OUTREL $ME
WHICH FUNCTION WOULD YOU LIKE TO PERFORM

1) CREATE A NEW RELATION
2) WRITE INTO A EXISTING RELATION
3) NO OPERATION

$SR\_N \#N=0
$SR\_I \#I
$JE LAB1
$IF ( \#N = 1 ) $JU ICRLN
$IF ( \#N = 2 ) $JU ICRO
$IF ( \#N = 3 ) $JU ISTP
$JU LAB1
ICRLO $$$ WRITING INTO A EXISTING RELATION.

CRE2
$JU ISTP
ICRLN $$$ CREATING A NEW RELATION.

INOR $PR WOULD YOU LIKE TO ENTER DATA INTO THE NEW RELATION?
$PR ENTER ( Y FOR YES OR N FOR NO )
$SR\_I \#RES
$JE INOR
$IF ( \#RES =Y ) $JU IDO
$IF ( \#RES =YES ) $JU IDO
$IF ( \#RES =N ) $JU ISTP
$IF ( \#RES =NO ) $JU ISTP
$JU INOR
IDO CRE2
ISTP 62
$PR *************** FINISHED ***************
$ME
EOF
IDENTIFICATION DIVISION.

PROGRAM-ID. SORT-ALL.
AUTHOR. ROBERT SAWSER.
DATE-WRITTEN. 4-1-82.

ENVIRONMENT DIVISION.

CONFIGURATION SECTION.
SOURCE-COMPUTER. HARRIS-123.
OBJECT-COMPUTER. HARRIS-123.

INPUT-OUTPUT SECTION.

FILE-CONTROL.
SELECT IN-FILE ASSIGN TO "w8".
SELECT OUT-FILE ASSIGN TO "w9".
SELECT SORT-FILE ASSIGN TO "SORTING".

DATA DIVISION.
FILE SECTION.
FD IN-FILE
   DATA RECORDS IS IN-REC.
01 IN-REC.

WORKING-STORAGE SECTION.

PROCEDURE DIVISION.

PAR-SORT.
   SORT SORT-FILE ON ASCENDING KEY
       USING IN-FILE
       GIVING IN-FILE
   STOP RUN.

EOF...
$MS
B2
EL-LO
$JE.P IERROR1
!ERROR1 EL LR
$JE.P !ERROR2
!ERROR2 $PR ** COBCLING &1 ***
COBOL I &1
B2
$PR ** EXECUTING &1 **
VX
B2
$PR *** COMPILATION COMPLETE ***
$ME
EOF...
$MS

BBLK

$SR, N #A = 1

$LOOPY SIF (#A = 13) SJU IRUN

$SR, N #A = #A + 1

$PR

$SJU $LOOPY

IRUN

$PR

*** FILE SORTED ***

$SELO

$SJE, P ISTOP

$PR

ISTOP $ PR

$ME

EOF...
$MS

BBLK

$PR

$PR

GETRL

FR ALL

$AS 12 = OUTFILE

$SRF #FN = 12

$JE !INREC

$CO #FN W8

BA,C I SORR=8

VX

BBLK

$PR

$PR

$PR

$PR

$PR

$PR

$PR

$CO SORR=C W7

ED W7

SIN 28 W2

SIN 35 W3

UP

CB2 W7

CO W9 #FN

$SREND

!INREC $ME

EOF..
BEGIN SRR,SNR
SPR
SPR INPUT THE NAME OF THE RELATION WHICH YOU WOULD LIKE TO
SPR EDIT?
SRR, I #NAME
SPR #NAME IS THE RELATION IN WHICH YOU WOULD LIKE TO EDIT? (Y/N)
IRESTAR
SRR, I #ANS
SIF, (#ANS=Y) $JU IBEGIN
SJE IERR
SIF, (#ANS=N) $JU ICONT
IERR
SPR PLEASE TYPE 'Y' OR 'N'...
$JU IRESTAR
ICONT
$ED #NAME
SJE IERR1
SPR
SPR WOULD YOU LIKE TO CHANGE, DELETE, OR INSERT A TUPLE?
IRESTA
SRR, I #ANS
SIF, (#ANS=CHANGE) $JU ICHLN
SIF, (#ANS=DELETE) $JU IDELN
SIF, (#ANS=INSERT) $JU IINLN
$JU IERR2
IERR1
SPR
SPR #NAME IS A NON-EXISTING RELATION
SPR
SPR WOULD YOU LIKE TO EDIT ANOTHER RELATION? (Y/N)
IREST
SRR, I #ANS
SIF, (#ANS=Y) $JU IBEGIN
SJE IERI
SIF, (#ANS=N) $JU IENDI
IERI
SPR
SPR PLEASE TYPE 'Y' OR 'N'...
$JU IREST
SPR
IERR2
SPR
SPR PLEASE TYPE IN CHANGE, DELETE, OR INSERT...
$JU IRESTAR
SPR
ICH LN
SPR
SPR INDICATE WHICH TUPLE YOU WOULD LIKE TO CHANGE BY
SPR GIVING CORRESPONDING LINE NUMBER?
SRR, I #LNC
$DI #NAME #LNC I
SJE ICHLN
SPR
SPR
SPR IS THIS THE TUPLE THAT YOU WOULD LIKE TO CHANGE? (Y/N)
IRESTA1
SRR, I #ANS
SIF, (#ANS=N) $JU ICHLN
SJE IERR3
SIF, (#ANS=Y) $JU ICONT1
IERR3
SPR
SPR PLEASE TYPE IN 'Y' OR 'N'...
$JU IRESTA1
SPR
SPR
ICONT1
SPR
SPR THIS IS THE TUPLE THAT YOU WOULD LIKE TO CHANGE.
SPR WOULD YOU LIKE TO CHANGE ANOTHER TUPLE? (Y/N)

IESTA2 $SR,I #ANS
SIF,(#ANS=N) $SU 1END
$JE 1ERR4
SIF,(#ANS=Y) $SU ICLN

IERR4 $PR
$PR PLEASE TYPE IN 'Y' OR 'N'...
$SU IESTA2

IDELN $PR
$PR INDICATE WHICH TUPLE YOU WOULD LIKE TO DELETE
$PR BY GIVING CORRESPONDING LINE NUMBER.
$SR,I #LND
$PR
$DI #NME #LND 1
$JE IDELN
$PR

SPR PLEASE TYPE 'Y' OR 'N'...
$SU IESTA3

IESTA3 $SR,I #ANS
SIF,(#ANS=N) $SU IDELN
$JE 1ERR5
SIF,(#ANS=Y) $SU JCONT2

IERR5 $PR
$PR PLEASE TYPE 'Y' OR 'N'...
$SU IESTA3

ICONT2 $DE #LND
$JE 1ERR6
$PR
$PR TUPLE #LND HAS BEEN DELETED
$PR

IESTA4 $SR,I #ANS
SIF,(#ANS=N) JU IEND
$JE 1ERR7
SIF,(#ANS=Y) JU IDELN

IERR6 $PR
$PR #LND IS A NON-EXISTING TUPLE
$PR

IESTA4 $SR,I #ANS
SIF,(#ANS=N) JU IEND
$JE 1ERR7
SIF,(#ANS=Y) JU IDELN

IERR7 $PR
$PR PLEASE TYPE 'Y' OR 'N'...
$SU IESTA4
$PR

IINLN $EL 9
$PR
$PR INDICATE THE TUPLE THAT YOU WOULD LIKE TO INSERT A NEW TUPLE AFTER BY GIVING THE CORRESPONDING LINE NUMBER.
$SR,I #LNI
$PR
$DI #NME #LNI 1
$JE IINLN
$PR

IINLN $PR
$PR

IESTA5 $SR,I #ANS
SIF,(#ANS=N) JU IINLN
$JE 1ERR8
SIF,(#ANS=Y) JU JCONT3
$PR

IERR8 $PR
$PR PLEASE TYPE 'Y' OR 'N'...
ICONTUP $AS 9$W9
$PR$PR INPUT THE TUPLE(S) IN WHICH YOU WOULD LIKE INSERTED...
$PR$PR NOTE:
$PR$PR FOLLOW THE FORMAT OF THE TUPLE DISPLAYED TO
$PR$PR THE TERMINAL.
$PR$PR IF YOU DESIRE TO STOP INSERTING, TYPE 'O' IN
$PR$PR THE FIRST COLUMN.
$PR$PR
$PR$DI $NME $LNI 1
$READ $SR, I $TUP
$JE ICONTUP
$CONTUP $IF, (#TUP=$0) SJU IEXIT
$JE ICONTP
$CONTP $PR.F,9 $TUP
$JE ICONTUP
$CONTUP $SJU IREAD
$EXIT $IN $LNI $W9
$PR
$PR $PR WOULD YOU LIKE TO INSERT ANOTHER TUPLE INTO THE RELATION?
$PR (Y/N)
$RESTA6 $SR, I $ANS
$IF, (#ANS=N) JU IEND
$JE IERR8
$IF, (#ANS=Y) JU IINLN
$ERR8 $PR
$PR PLEASE TYPE 'Y' OR 'N'...
$SJU IRESTA6
$PR
$END $SUP
$PR
$PR $PR WOULD YOU LIKE TO EDIT ANOTHER RELATION? (Y/N)
$RESTT $SR, I $ANS
$IF, (#ANS=N) SJU IEND1
$JE IERRR4
$IF, (#ANS=Y) SJU IBEGIN
$ERRR4 $PR
$PR PLEASE TYPE IN 'Y' OR 'N'...
$SJU IRESTT
$END1 $2
$MO SI
$ME

EOF
IDENTIFICATION DIVISION.
PROGRAM-ID. CREATE-A-NEW-RELATION.
AUTHOR. ARTHUR ROBERTS JR.
DATE-WRITTEN. MARCH 10, 1982.
DATE-COMPILED.

ENVIRONMENT DIVISION.
CONFIGURATION SECTION.
SOURCE-COMPUTER. HARRIS-123.
OBJECT-COMPUTER. HARRIS-123.

INPUT-OUTPUT SECTION.
FILE-CONTROL.
SELECT UNIT=OUTPUT-FILE ASSIGN "OUTFILE".
SELECT UNIT=INDEX-FILE ASSIGN "INDEXLOG"
ORGANIZATION IS RELATIVE
ACCESS MODE IS SEQUENTIAL
RELATIVE KEY IS REC-POS.

DATA DIVISION.
FILE SECTION.
FD UNIT=OUTPUT-FILE
DATA RECORD IS OUT-REC.
01 OUT-REC.
  02 FILLER PIC X(80).
FD UNIT=INDEX-FILE
DATA RECORD IS INDEX-REC.
01 INDEX-REC.
  02 REL-NAME PIC X(8).
  02 INFOMAT PIC X(991).
WORKING-STORAGE SECTION.
77 REC-POS PIC 9(5) VALUE 1.
77 TREL-NAME PIC X(8) VALUE SPACE.
77 CONDI PIC X(1) VALUE "N".
01 OUTPUT-AREA.
  02 DATA-LINE.
    03 FILLER PIC X(991) VALUE SPACES.
  02 DATA-OUT REDEFINES DATA-LINE.
    03 FILLER PIC X.
    03 NO-CF-ATT PIC X(3).
    03 CC-CF-ATT OCCURS 34 TIMES.
      04 FILLER PIC X.
      04 ATT-NAME PIC X(20).
      04 FILLER PIC X.
      04 BEG-C PIC X(3).
      04 FILLER PIC X.
      04 END-C PIC X(3).
    03 FILLER PIC X.
01 WORK-AREA.
  02 BEG-COLUMN PIC 999 VALUE 0.
  02 END-COLUMN PIC 999 VALUE 0.
  02 COLUMN-L.
    04 C-1 PIC X.
    04 C-2 PIC X.
02 RES PIC XXX VALUE 1.
02 HOLD-C PIC 999 VALUE 0.
02 SUB PIC 999 VALUE 0.
02 NUM PIC 99 VALUE 0.
02 HOLD.
03 H-1 PIC X VALUE SPACES.
03 H-2 PIC X VALUE SPACES.
03 H-3 PIC X VALUE SPACES.

PROCEDURE DIVISION.
PROCESS-CONTROL.
OPEN
OUTPUT-UNIT-OUTPUT-FILE
I-0 UNIT-INDEX-FILE.

PERFORM MAIN-Routine thru MAIN-Routine-EXIT.

CLOSE
UNIT-OUTPUT-FILE
UNIT-INDEX-FILE.

STOP RUN.

MAIN-Routine.
MOVE "N" TO COND1.
DISPLAY "INPUT RELATION NAME ? ".
DISPLAY "THIS NAME CAN ONLY BE 8 CHARACTERS LONG ".
DISPLAY "NO SPECIAL CHARACTERS OR BLANKS ".
ACCEPT TREL-NAME FROM TERMINAL.

DISPLAY " ".
DISPLAY "RELATION NAME = = = = TREL-NAME ".
DISPLAY " ".
DISPLAY "IS THIS INFORMATION CORRECT ? (Y OR N)".
DISPLAY " ".
ACCEPT RES FROM TERMINAL.
IF RES = "Y"
    DISPLAY "OK"
    PERFORM CHECK-RELATION-NAME THRU CHECK-RELATION-NAME-EXIT
    PERFORM CLOSE-OPEN-FILE.
    IF COND1 = "Y"
        DISPLAY "-ERROR RELATION " TREL-NAME " ALREADY EXISTS"
    GO TO MAIN-Routine
    ELSE
        NEXT SENTENCE
    END-IF
ELSE
    DISPLAY "***** ERROR *****"
    GO TO MAIN-Routine.
END-IF

MAIN.
PERFORM BLANK-DISPLAY 5 TIMES.
DISPLAY "INPUT NUMBER OF ATTRIBUTES ? ".
DISPLAY "THERE CAN NOT BE MORE THAN 34 ATTRIBUTES"
ACCEPT COLUMN-L FROM TERMINAL.

MOVE 0 TO NUM.
INSPECT COLUMN-L TALLYING NUM FOR ALL ".
IF NUM = 0
    MOVE COLUMN-L TO NO-OF-ATT
ELSE
    IF NUM = 1
        MOVE C-2 TO H-3
    END-IF
ELSE
   IF NUM = 2
      MCVE C-1 TO H-3
      MCVE "O" TO H-2, M-1
      MCVE HOLD TO NO-OF-ATT
   ELSE
      MCVE COLUMN-L TO HOLD
      MCVE HOLD TO NO-OF-ATT.
   ENDIF

   DISPLAY "",
   DISPLAY "",
   DISPLAY "",
   IF NO-OF-ATT GREATER THAN '034'
      DISPLAY '** TO MANY ATTRIBUTES ===>' NO-OF-ATT
      DISPLAY '**** ERROR ****'
      GO TO MAIN.
   ENDIF

   IF NO-OF-ATT NOT NUMERIC
      DISPLAY "NUMBER OF ATTRIBUTES MUST BE NUMERIC ===>",
      NO-OF-ATT
      DISPLAY "**** ERROR ****".
      GO TO MAIN
   ELSE
      DISPLAY "NUMBER OF ATTRIBUTES ===>", NO-OF-ATT.
      DISPLAY "",
      DISPLAY "IS THIS INFORMATION CORRECT? (Y OR N)."
      ACCEPT RES FROM TERMINAL.
      IF RES = "Y"
         DISPLAY "OK"
      ELSE
         IF RES = "N"
            DISPLAY "**** ERROR ****"
            GO TO MAIN
      ELSE
         DISPLAY "EXPECTING (Y OR N)."
         DISPLAY "**** ERROR ****"
         GO TO MAIN.
   ENDIF

   PERFORM MAIN=1 THRU MAIN=EXIT
   VARYING SUB FROM 1 BY 1
   UNTIL SUB GREATER THAN 34 OR
   SUB GREATER THAN NO-OF-ATT.

   MOVE TREL-NAME TO REL-NAME.
   WRITE OUT-REC FROM REL-NAME.
   WRITE OUT-REC FROM NO-OF-ATT.
   PERFORM WRITE=ROUTINE
   VARYING SUB FROM 1 BY 1 UNTIL
   SUB GREATER THAN 34 OR
   SUB GREATER THAN NO-OF-ATT.
   MOVE DATA-LINE TO INFORM.

   WRITE=INDEX=RECORD.
   WRITE INDEX=REC INVALID KEY GO TO COMPUTE=RECORD=POSITION.
   GO TO MAIN=ROUTINE=EXIT.

   COMPUTE=RECORD=POSITION.
   COMPUTE REC-POS = REC-POS + 1.
   GO TO WRITE=INDEX=RECORD.

   MAIN=ROUTINE=EXIT.
AT END
GO TO CHECK-RELATION-NAME-EXIT.

IF REL-NAME = TREL-NAME
MOVE "Y" TO CONDI
ELSE
GO TO CHECK-RELATION-NAME.

CHECK-RELATION-NAME-EXIT.
EXIT.

CLOSE-OPEN-FILE.
CLOSE UNIT-INDEX-FILE.
OPEN I-O UNIT-INDEX-FILE.

MAIN-1.
PERFORM BLANK-DISPLAY 20 TIMES.
DISPLAY "ATTRIBUTE " SUB "- REQUESTED ", NO-OF-ATT.
DISPLAY ".
DISPLAY
"INPUT THE FOLLOWING FOR SPACES BETWEEN ATTRIBUTES !".
DISPLAY "FOR ATTRIBUTE NAME ==> B".
DISPLAY ".
DISPLAY "FOR ATTRIBUTE LENGTH ==> NUMBER-OF-SPACES".
DISPLAY ".
DISPLAY ".
DISPLAY "INPUT ATTRIBUTE NAME ? ".
DISPLAY "THIS NAME CAN BE A MAX OF 20 CHARACTERS".
ACCEPT ATT-NAME (SUB) FROM TERMINAL.

DISPLAY ".
DISPLAY "ATTRIBUTE NAME ===> ", ATT-NAME (SUB).
DISPLAY ".
DISPLAY "IS THIS INFORMATION CORRECT ? (Y OR N)".
DISPLAY ".
ACCEPT RES FROM TERMINAL.
IF RES = "Y"
DISPLAY "OK "
ELSE
IF RES = "N"
DISPLAY "***** ERROR *****
GO TO MAIN-1
ELSE
DISPLAY "EXPECTING (Y OR N)"
DISPLAY "***** ERROR *****
GO TO MAIN-1.

MAIN-2.
PERFORM BLANK-DISPLAY 5 TIMES.
IF END-COLUMN GREATER THAN 999
MOVE 99 TO SUB
DISPLAY
"*** RECORD HAS REACHED MAXIMUM-LENGTH ***
GO TO MAIN-EXIT.

DISPLAY "ATTRIBUTE " SUB "- REQUESTED ", NO-OF-ATT.
DISPLAY ".
DISPLAY "INPUT ATTRIBUTE LENGTH ? ".
ACCEPT COLUMN-L FROM TERMINAL.
ELSE
  IF NUM = 1
    MOVE C-2 TO H-3
    MOVE C-1 TO H-2
    MOVE "0" TO H-1
    MOVE HOLD TO COLUMN-L
  ELSE
    IF NUM = 2
      MOVE C-1 TO H-3
      MOVE "0" TO H-2, H-1
      MOVE HOLD TO COLUMN-L.
      DISPLAY " ".
      DISPLAY " ".
      DISPLAY ".
      IF COLUMN-L NOT NUMERIC
        DISPLAY "COLUMN LENGTH HAS TO BE 3 NUMERIC CHARACTERS ==> ".
        COLUMN-L
        DISPLAY "***** ERROR *****"
        GO TO MAIN-2
      ELSE
        MOVE NEXT-COLUMN TO BEG-COLUMN, HOLD-C
        MOVE COLUMN-L TO COLUMN-LEN
        ADD COLUMN-LEN, BEG-COLUMN GIVING NEXT-COLUMN
        SUBTRACT 1 FROM NEXT-COLUMN GIVING END-COLUMN.
        DISPLAY "ATTRIBUTE NAME ==> ", ATT-NAME (SUB),
        " COLUMN LENGTH ==> ", COLUMN-LEN.
        DISPLAY " ".
        DISPLAY "STARTING POSITION ==> ", BEG-COLUMN,
        " ENDING POSITION ==> ", END-COLUMN.
        DISPLAY ".
        DISPLAY "NEXT AVAILABLE POSITION ===> ", NEXT-COLUMN.
        DISPLAY " ".
        DISPLAY "IS THIS INFORMATION CORRECT? (Y OR N) ".
        ACCEPT RES FROM TERMINAL.
        IF RES = "Y"
          DISPLAY " OK "
        ELSE
          IF RES = "N"
            MOVE HOLD-C TO NEXT-COLUMN
            DISPLAY "***** ERROR *****"
            GO TO MAIN-2
          ELSE
            MOVE HOLD-C TO NEXT-COLUMN
            DISPLAY "EXPECTING (Y OR N) 
            DISPLAY "***** ERROR *****"
            GO TO MAIN-2.
          IF ATT-NAME (SUB) = "B"
            GO TO MAIN-1
          ELSE
            MOVE BEG-COLUMN TO BEG-C (SUB)
            MOVE END-COLUMN TO END-C (SUB).
            MAIN-EXIT.
            EXIT.
WRITE ROUTINE.
WRITE OUT-REC FROM ATT-NAME (SUB).
WRITE OUT-REC FROM BEG-C (SUB).
WRITE OUT-REC FROM END-C (SUB).
EOF..