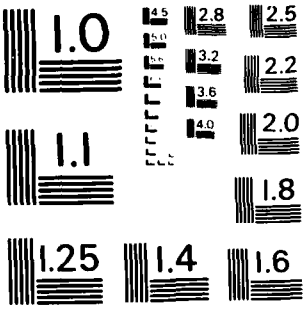
AD-A131 481 DEVELOPMENT OF A TEXT-EDITOR BASED RELATIONAL DATA BASE MANAGEMENT SYSTEM..(U) FLORIDA A AND M UNIV TALLAHASSEE DEPT OF DATA PROCESSING T W MASON AUG 81 1/1

UNCLASSIFIED AFOSR-TR-83-0665 AFOSR-81-0131 F/G 9/2 NL

END
DATE
FILMED
DTIC



MICROCOPY RESOLUTION TEST CHART
NATIONAL BUREAU OF STANDARDS - 1963 - A

AFOSR-TR- 83-0665

3

FINAL REPORT

AD A131481

Development of a Text-Editor based Relational Data base Management System

Grant No. AFOSR-81-0131

June 1, 1981 - August 31, 1981

Thomas W. Mason
Florida A&M University
Tallahassee, FL 32307

DTIC FILE COPY

DTIC
ELECTE
AUG 19 1983
S D
D

Approved for public release;
distribution unlimited.

83 08 08 198

UNCLASSIFIED

UNCLASSIFIED

SECURITY CLASSIFICATION OF THIS PAGE (When Data Entered)

REPORT DOCUMENTATION PAGE

READ INSTRUCTIONS BEFORE COMPLETING FORM

1. REPORT NUMBER: AFOSR-TR- 83-0665

2. GOVT ACCESSION NO.: AD A131481

3. RECIPIENT'S CATALOG NUMBER:

4. TITLE (and Subtitle): DEVELOPMENT OF A TEXT-EDITOR BASED RELATIONAL DATA BASE MANAGEMENT SYSTEM

5. TYPE OF REPORT & PERIOD COVERED: FINAL, 1 JUN 81-31 AUG 81

6. PERFORMING ORG. REPORT NUMBER:

7. AUTHOR(s): Thomas W. Mason

8. CONTRACT OR GRANT NUMBER(s): AFOSR-81-0131

9. PERFORMING ORGANIZATION NAME AND ADDRESS:
Data Processing Department
Florida A&M University
Tallahassee FL 32307

10. PROGRAM ELEMENT, PROJECT, TASK AREA & WORK UNIT NUMBERS: PE61102F; 2304/A2

11. CONTROLLING OFFICE NAME AND ADDRESS:
Mathematical & Information Sciences Directorate
Air Force Office of Scientific Research
Bolling AFB DC 20332

12. REPORT DATE: Aug 1981

13. NUMBER OF PAGES: 36

14. MONITORING AGENCY NAME & ADDRESS (if different from Controlling Office):

15. SECURITY CLASS. (of this report): UNCLASSIFIED

15a. DECLASSIFICATION DOWNGRADING SCHEDULE:

16. DISTRIBUTION STATEMENT (of this Report)

Approved for public release; distribution unlimited.

17. DISTRIBUTION STATEMENT (of the abstract entered in Block 20, if different from Report)

18. SUPPLEMENTARY NOTES

19. KEY WORDS (Continue on reverse side if necessary and identify by block number)

20. ABSTRACT (Continue on reverse side if necessary and identify by block number)

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)

DD FORM 1 JAN 73 1473

UNCLASSIFIED

SECURITY CLASSIFICATION OF THIS PAGE (When Data Entered)

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.

Accession For
NTIS GRA&I ☒
DTIC TAB ☐
Unannounced ☐
Justification
By
Dist
Dist Special
A

NOTICE OF TRANSMITTAL TO DTIC
This technical report has been reviewed and is
approved for public release IAW AFR 190-12.
Distribution is unlimited.
MATTHEW J. KERPER
Chief, Technical Information Division

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 a line-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 proceding 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 on 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

2. Steuert, J. and J. Goldman, "The relational data management system: A perspective," Proceeding of ACM SIGFIDET Workshop on Data Description, Access, and Control, Ann Arbor, Michigan, May 1-3, 1974.

3. Held, G.,M.R. Stonebraker and E. Wong, "INGRES: A relational data base system," Proceedings of National Computer Conference, Anaheim, California, May 19-22, 1975.

4. Zloof, M.M., "Query by Example," Proceedings of National Computer Conference, Anaheim, California, May 19-22, 1975.

5. Meldman, M.J., D.J. McLeod, R.J. Pellicore and M. Squire, RISS: A Relational Data Base Managment System for Minicomputers, Van Nostrand Reinhold Co., New York, New York, 1978.

```
$MS
     $MO BS=1000
     $MO NU
     $RR.SRN
      $AS 10 = OUTFILE
      $SR.N #NN = 0
      $SR.S #NMM = NONAME
      $SR.F #NMM=10
      $AS 20 = #NMM
      $JE !GEN
!LAB2  $$$      RELATION
      $SR.N #NMA=0
      $SR.N #EE=1
      $SR.F #NMA = 10
        $$$      RELATION
      $MO NREG=50
      $SR.N #DL=0
      $SR.N #END=0
      $SR.N #BEG=0
      $SR.S #VAL=
      $PR
      $PR                    HOW MANY RECORDS ARE YOU INSERTING ?
      $SR.N #TPIN=0
      $SR.I #TPIN
      $JE !KEEP
!KEEP    $$$     FLAG DEVICE
         $$$     THIS SECTION EDITS RELATION AND
         $$$     INSERTS BLANK RECORDS FOR INSERTION.
        $ED #NMM RE
        $PE 0
        AE 1000000
        $JE !OUT
!OUT    $BE,1
        $E,0
        $SR.N #LIN=.ERN()
        $ED #NMM AB
        $SR.N #IN=0
!LAB    $$$
        IN #LIN BLANK,1
        $SR.N #IN = #IN + 1
        IF ( #IN=#TPIN ) $JU !OUT2
        $JU !LAB
!OUT2   $$$
        $$$                        THE INSERTION OF ATTRIBUTES BEGINS
        $UP
        $ED #NMM AB
        $SR.N #LIN = #LIN + 1
        $E #LIN
!AGAIN $$$
       $SR.F #VAL=10
       $SR.F #BEG=10
       $SR.F #END=10
       $PR
       $PR ATTRIBUTE #EE
!REN   $$$
       $PR #VAL >
       $SR.S #IN=NULL
       $SR.I #IN
       $JE 178 !NPO
!NPO   $$$                INSERT BLANK CHAR        --->
       $IF ( #IN=$END ) $JU !OUT4
       $SR.N #CHE = #END - #BEG + 1
```

```
        $IF ( #CLE > #CHE ) $JU !KEN
        $C #BEG-#END,#IN
        $IF ( #EE = #NMA ) $JU !OUT3
        $SR.N #EE = #EE + 1
        $JU !AGAIN
!OUT3   $$$                 FIRST ATTRIBUTE INSERTED
        $SR.N #EE = 1
        $SR.N #NN = #NN + 1
        $SR.N #LIN = #LIN + 1
        $RW 10
        $SR.F #NMM=10
        $SR.F #NMA = 10
        $IF ( #NN = #TPIN ) $JU !OUT4
        $E #LIN
        $PR                 * INSERT SEND ON NEXT ENTRY TO TERMINATE
        $PR                         ---- DATA ENTRY..
        $JU !AGAIN
!OUT4 $$$
        $$$     * IF SEND WAS ENTRIED THIS SECTION WILL DELETE
        $$$     THE REMAINING LINES THAT WAS ORGINALLY REQIESTED.
        $IF ( #NN = #TPIN ) $JU !OUT5
        $IF ( #NN < #TPIN ) $DE #LIN
        $SR.N #DL = #DL + 1
        $SR.N #NN = #NN + 1
        $SR.N #LIN = #LIN + 1
        $JU !OUT4
!OUT5 $$$   THIS SECTION TERMINATES THE DATA ENTRY PROCESS.
        $UP
        $SR.N #LIN = #LIN - #DL
        $SR.N #LIN = #LIN - 2
        $PR
        $PR
        $PR  RELATION NAME = #NMM : NUMBER OF RECORDS = #LIN
$ME
!GEN $$$
        $PR                 RELATION #NMM NOW BEING GENERATED
        $GE #NMM
        CO BLANK #NMM
        $JU !LAB2
        $ME
!KEN      $PR -ERROR GENERATED STRING LENGTH OF #CHE EXCEEDED
          $JU !REN
```

```
IDENTIFICATION DIVISION.
PROGRAM-ID.
            GET-A-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 RES                     PIC X(3)  VALUE SPACES.
77 TREL-NAME               PIC X(8) VALUE SPACE.
77 COND1                   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 BEG-C          PIC X(3).
          04 FILLER         PIC X.
          04 END-C          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-DATA
              WRITE OUT-REC FROM REL-NAME
              WRITE OUT-REC FROM NO-OF-ATT
             PERFORM WRITE-ROUTINE
               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 TC 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..
```

```
            $MS
            BBLK
            $MO BS=999
!BEGIN      $RR.SNR
            $PR
            $PR INPUT THE NAME OF THE RELATION IN WHICH YOU WOULD LIKE COPI
            $SR.I #NME
            $PR
            $PR #NME IS THE RELATION IN WHICH YOU WOULD LIKE COPIED? (Y/N)
!RESTAR     $SR.I #ANS
            $IF,(#ANS=N) $JU !BEGIN
            $JE !ERR
            $IF,(#ANS=Y) $JU !CONT
!ERR        $PR
            $PR PLEASE TYPE IN 'Y' OR 'N'...
            $JU !RESTAR
!CONT       $GE #NME
            $JE.K,318,!ERR44
            $JE !NEXT
            $JU !ERR1
!NEXT       $PR
            $PR WHAT IS THE NAME OF THE NEW RELATION?
!NEXT1      $SR.I #NM
            $PR
            $PR #NM IS THE NEW RELATION? (Y/N)
!RESTA      $SR.I #ANS1
            $IF,(#ANS1=N) $JU !NEXT
            $JE !ERR2
            $IF,(#ANS1=Y) $JU !CONT2
!ERR2       $PR
            $PR PLEASE TYPE IN 'Y' OR 'N'...
            $JU !RESTA
!CONT2      $GE #NM
            $JE.K,2150,!ERR3
            $JE !ERR33
            $CO #NME #NM
            $PR
            $JU !INDEX
!ERR1       $PR
            $PR #NME IS AN NON-EXISTING RELATION
            $PR
            $PR WOULD YOU LIKE TO COPY ANOTHER RELATION? (Y/N)
            $JU !REST
!ERR3       $JE !NEXT3
!NEXT3      $PR
            $PR #NM IS AN EXISTING RELATION (OR HARRIS FILE)
            $PR
!CONT33     $PR CHOOSE ANOTHER NAME FOR THE NEW RELATION...
            $JU !NEXT1
!INDEX      $CO INDEXLOG W7
            $AS 12=W6
            $PR.F,12,#NME
            $AS 13=W9
            SOURCE1
            $SR.F #LNE=13
            $ED INDEXLOG
            $IN #LNE INDEXLOG #LNE
            $UP
            $SR.N #LNE=#LNE+1
            $TAB
            $TAB 10
            $ED INDEXLOG
```

```
            $UP
            $TAB
            $EL W6
            $EL W7
            $EL W9
            $FREE ALL
            $PR #NME HAS BEEN COPIED INTO #NM
            $PR
            $PR WOULD YOU LIKE TO COPY ANOTHER RELATION? (Y/N)
!REST       $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
!ERR44      $JE !NEXT44
!NEXT44     $PR
            $PR #NME 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 TRY AGAIN...
            $PR
            $JU !BEGIN
!END1       B2
EOF..
```

```
100 AS 10 = "W2"
200 AS 20 = "OUTFILE"
400 AS 40 = "W3"
500 DIM INDX$(50),RELS(50),LABL$(99),BEG$(99),END$(99)
600 DIM BEG(99),ENDD(99),FILE$(3)
700 OPEN #20
1000 GOSUB 8100
5400 FOR FI = 1 TO 3
5500 START = 1
5600 FOR P = 1 TO ATT
5700 GOSUB 8900
5800 SIZE = (ENDD(P) - BEG(P)) + 1
5900 PRINT #10 TAB(11);"05  ";TAB(15);FILE$(FI);TAB(18);LABL$(P);->
                TAB(50);"PIC X(";
5910 IF SIZE < 10 DO
5920     PRINT #10 USING "#";TAB(56);SIZE;
5930 ELSE
5940     PRINT #10 USING "##";TAB(56);SIZE;
5950 DOEND
5960 PRINT #10 ")."
6000 START = ENDD(P) + 1
6100 NEXT P
6200 IF FI = 1 DO
6300     PRINT #10
6400     PRINT #10 TAB(7);"FD  OUT-FILE"
6500     PRINT #10 TAB(11);"DATA RECORD IS OT-REC."
6600     PRINT #10
6700     PRINT #10 TAB(7);"01  OT-REC."
6800 ELSE
6900     IF FI = 2 DO
7000         PRINT #10
7100         PRINT #10 TAB(7);"SD SORT-FILE"
7200         PRINT #10 TAB(11);"DATA RECORD IS SORT-REC."
7300         PRINT #10
7400         PRINT #10 TAB(7);"01  SORT-REC."
7500     ELSE
7600     DOEND
7700 DOEND
7800 NEXT FI
7900 PRINT #40 TAB(19);FILE$(3);TAB(22);KEY$
8000 STOP
8100 INPUT #20 INRELS
8150 INPUT #20 ATT
8160 FOR P = 1 TO ATT
8170 INPUT #20 LABL$(P)
8180 INPUT #20 BEG(P)
8190 INPUT #20 ENDD(P)
8200 NEXT P
8300 FILE$(1) = "IN-"
8400 FILE$(2) = "OT-"
8500 FILE$(3) = "ST-"
8510 PRINT
8520 PRINT
8600 PRINT "WHICH KEY(S) IN ";INRELS;" DO YOU WANT THE FILE TO";->
          " BE SORTED ON"
8700 INPUT "TYPE IN THE FIELD NAME(S) ";KEY$
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
9106         PRINT #10 USING "#";TAB(56);FIL;
```

```
9110          PRINT #10 USING "##";TAB(56);FIL;
9115      DOEND
9120      PRINT #10 ")."
9200 ELSE
9300 DOEND
9400 RETURN
9800 END
```

C R E R E L

AUTHORS

KENNTH MCCRAY

ARTHUR ROBERTS

THE CREREL RELATION WAS CREATED TO SERVE AS A DATA ENTRY SYSTEM FOR FREDB. 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 ALSC 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 ******
****** FCR EVER 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 IN THE ATTRIBUTE LENGTH FIELD TYPE THE NUMBER OF SPACES THAT IS TO BE PLACED BETWEEN THE ATTRIBUTES.

THIS WILL NCT 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	CREATES A NEW RELATION
CREATE1	RETRIEVES THE ATTRIBUTES FOR A GIVEN RELATION
CREREL	JCL FOR CREREL SYSTEM
CRE2	JCL FOR INSERTING OF ATTRIBUTES

LINK MODULE	PURPOSE	FILENAME
NEWRL	CREATE A NEW RELATION	CREATE
GETRL	GET A RELATION	CREATE1

EOF..

```
          $MS
          $MO BS=999
          BBLK
          $JO H !HELP
          $JO D !DESCRP
          $PR
          $PR
          $PR                          WELCOME TO
          $PR
          $PR
          $PR FFFFFFFFFF   RRRRRRRRR                DDDDDDDDD
          $PR F            R        R               D        D
          $PR F            R        R               D         D   B
          $PR F            R        R    EEEEEE     D         D   B
          $PR FFFFFF       RRRRRRRRR    E      E   D         D   B
          $PR F            R      RR    EEEEEEE    D         D   BBBBBBB
          $PR F            R       RR   E          D         D   B       B
          $PR F            R        RR  E          D        D    B       B
          $PR F            R        RR   EEEEEE    DDDDDDDDD     BBBBBBB
          $PR
          $PR
          $PR                    FAMU RELATIONAL DATABASE
          $PR
          $PR
          $PR
          $PR NOTE:  TO GET A LIST OF THE AVAILABLE MODULES IN THE 'FREDB'
          $PR        SYSTEM, TYPE IN:
          $PR                          FREDB.H
          B2
          $ME
!HELP     $PR
          $PR   THESE ARE THE ONLY AVAILABLE MODULES IN THE 'FREDB' SYSTEM.
          $PR
          $PR          CREREL              EDREL
          $PR          COPREL              COLREL
          $PR          SORREL              MERREL
          $PR          DELREL              RETREL
          $PR
          $PR
          $PR   NOTE:  IF YOU WISH TO GET A DESCRIPTION OF EACH MODULE, TYPE
          $PR
          $PR                 FREDB.D
          B2
          $ME
!DESCRP   $PR
$PR CREREL (RELATION CREATOR):  THIS MODULE ENABLES CREATION OF A 'FREDB'
$PR       RELATION.  THE USER SPECIFIES THE NAME OF THE RELATION, THE NUMBER
$PR       OF ATTRIBUTES (COLUMNS), AND THE ATTRIBUTE NAMES.  THIS MODULE ALSO
$PR       ALLOWS THE USER TO INPUT TUPLES INTO THE RELATION.
$PR
$PR EDREL (RELATION EDITOR):  THE RELATION EDITOR ALLOWS THE USER TO CHANGE,
$PR       DELETE, AND INSERT TUPLES (ROWS) IN THE RELATION.
$PR
$PR COPREL (RELATION COPIER):  THE RELATION-COPIER MODULE ALLOWS A USER TO
$PR       MAKE AN EXACT COPY OF AN EXISTING RELATION INTO A NON-EXISTING RELA
$PR
$PR COLREL (COLUMN SPECIFICATION MODIFIER):  THIS MODULE ALLOWS A USER TO
$PR       DELETE ATTRIBUTES FROM A RELATION.
$PR
$PR SORREL (RELATION SORTER):  THIS MODULE ALLOWS A USER TO SORT THE TUPLES
$PR       A RELATION BASED ON THE CONTENTS OF SOME SPECIFIED ATTRIBUTE IN THE
$PR       RELATION.
```

```
$PR
$PR DELREL (RELATION DELETER):  THIS MODULE ALLOWS A USER TO SELECTIVELY
$PR       RETRIEVE DATA FROM A 'FREDB' RELATION.
$ME
```

```
            $MS
            $FREE ALL
            BBLK
!BEGIN      $RR.SNR
            $MO BS=999
            $PR
            $PR
            $PR INPUT THE NAME OF THE RELATION WHICH YOU WOULD LIKE DELETED?
            $SR.I #NME
            $PR
            $PR #NME IS THE RELATION IN WHICH YOU WOULD LIKE DELETED? (Y/N)
!RESTAR     $SR.I #ANS
            $IF,(#ANS=N) $JU !BEGIN
            $JE !ERR
            $IF,(#ANS=Y) $JU !CONT
!ERR        $PR
            $PR PLEASE TYPE IN 'Y' OR 'N'...
            $JU !RESTAR
!CONT       $EL #NME
            $JE.K,312,!ERR3
            $JE !ERR1
            $PR
            $JU !INDEX
!ERR1       $PR
            $PR #NME IS AN NON-EXISTING RELATION
            $PR
            $PR WOULD YOU LIKE TO DELETE ANOTHER RELATION? (Y/N)
            $JU !RESTA
!INDEX      $MO RE
            $ED INDEXLOG
            $AE,1-8,#NME
            $DE 0
            $UP
            $MO AB
            $PR
            $FREE ALL
            $PR #NME HAS BEEN DELETED
            $PR
            $PR WOULD YOU LIKE TO DELETE ANOTHER RELATION? (Y/N)
!RESTA      $SR.I #ANS1
            $IF,(#ANS1=N) $JU !END
            $JE !ERR2
            $IF,(#ANS1=Y) $JU !BEGIN
!ERR2       $PR
            $PR PLEASE TYPE IN 'Y' OR 'N'...
            $JU !RESTA
!ERR3       $JE !NEXT3
!NEXT3      $PR
            $PR #NME 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 TRY AGAIN...
            $PR
            $JU !BEGIN
!END        B2
EOF..
```

```
$MS
$ ***********************************************
$ ** THIS MACRO RETRIEVES A RELATION FROM THE **
$ ** FAMU RELATIONAL DATABASE < FREDB >       **
$ ***********************************************
!BEGIN $$$
BBLK
B2
B2
$PR        THIS IS THE RELATION RETREIVING MODULE
$PR
$PR
$FR ALL
$ ***********************************************
$ * GETRL:: IS THE LINK MODULE FROM A COBOL    *
$ *         PROGRAM THAT SEARCHES FOR A RELATION
$ * ASSIGNMENTS ARE MADE INTERACTIVE TO THE    *
$ * SAME FILE IN THE COBOL PROGRAM < TREL >    *
$ * GOMIT:: IS THE EXECUTABLE LINK MODULE OF A *
$ *         COBOL PROGRAM <TREL> THAT RETRIEVES*
$ *         A WANTED ATTRIBUTE.                *
$ ***********************************************
GETRL
$AS 20 = OUTFILE
$SR.F *RRR = 20
$JE  !NOREC
$FR ALL
$AS 20 = #RRR
GOMIT
$JU  !NOREC
!NOREC $$$
$ ***********************************************
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..
```

```
$MS
BBLK
!ERR $$$
$RR.SRN
$PR              ***************************************************
$PR              **                                               **
$PR              **    W E L C O M E   T O   C O L R E L          **
$PR              **                                               **
$PR              ***************************************************
$PR
$PR
$PR
$PR  THIS COLREL MODULE ALLOWS YOU THE USER
$PR  TO DELETE ATTRIBUTES (COLUMNS) FROM AN
$PR  EXISTING RELATION.
$PR
$PR
$PR  TO DELETE A COLUMN FROM A RELATION THE
$PR  RELATION NAME AND ATTRIBUTE NAME MUST
$PR  BE TYPED IN WHERE SPECIFIED IN THE MODULE.
$PR
$PR
$PR
$PR
$PR PLEASE INPUT THE RELATION NAME>
$SR.I #N
$PR
$PR
$PR
$PR IS THIS THE CORRECT RELATION NAME #N ?
$PR
$PR
$PR PLEASE ENTER (Y) FOR YES OR (N) FOR (NO).
$SR.I #R
$JE !ERR
$IF (#R=Y) JU !CON
JU !ERR
!CON AS 20=#N
$AS 10=INPUT
$PR.F,10,#N
COBOL I "COL2"
$VX
EL #N
RN OUTREL #N
CE OUTREL
$ME
```

```
$MS
$RR.SRN
BBLK
$MO NREG=50
$SR.S #RES=NULL
!LAB1 $$$
$PR                    WHICH FUNCTION WOULD YOU LIKE TO PERFORM
$PR                    1) CREATE A NEW RELATION
$PR                    2) WRITE INTO A EXISTING RELATION
$PR                    3) NO OPERATION
$PR
$PR ENTER
$SR.N #N=0
$SR.I #N
$JE !LAB1
$IF ( #N = 1 ) $JU !CRLN
$IF ( #N = 2 ) $JU !CRLO
$IF ( #N = 3 ) $JU !STP
$JU !LAB1
!CRLO $$$       WRITING INTO A EXISTING RELATION.
      GETRL
      AS 21=OUTFILE
      SR.F #RES=21
      JE !STP
      CRE2
     $JU !STP
!CRLN $$$       CREATING A NEW RELATION.
      NEWRL
      BBLK
!NOR   $PR      WOULD YOU LIKE TO ENTER DATA INTO THE NEW RELATION ?
      $PR     ENTER ( Y FOR YES OR N FOR NO )
      $SR.I #RES
      $JE !NOR
      $IF ( #RES =Y ) $JU !DO
      $IF ( #RES =YES ) $JU !DO
      $IF ( #RES =N  ) $JU !STP
      $IF ( #RES =NO  ) $JU !STP
      $JU !NOR
!DO    CRE2
!STP   B2
      $PR    *************   F  I  N  I  S  H  E  D   ***************
      $ME
EOF..
```

```
IDENTIFICATION DIVISION.

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

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 OUT-FILE.
     STOP  RUN.
EOF..
```

```
$MS
B2
EL LO
$JE.P !ERROR1
!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 $IF (#A = 13) $JU !RUN
        $SR.N #A = #A + 1
        $PR
        $JU !LOOPY
!RUN    $PR
$PR                         *** FILE SORTED ***
$EL LO
$JE.P !STOP
$PR
!STOP $  PR
$ME
EOF..
```

```
          $MS
BBLK
          $PR
          $PR
          GETRL
          FR ALL
          $AS 12 = OUTFILE
          $SR.F #FN = 12
          $JE !NREC
          $CO #FN W8
          BA.C I SORR-B
          VX
          BBLK
          $PR
          $PR
          $PR
          $PR                NOW COMPILING & EXECUTING THE RELATION
          $PR
          $PR                   THE TERMINAL WILL PAUSE TEMPORARILY
          $CO SORR-C W7
          ED W7
          $IN 28 W2
          $IN 35 W3
          UP
          CB2 W7
          CO W9 #FN
          $REND
!NREC     $ME
EOF..
```

```
            $MS
            BBLK
            $MO BS=999
            $MO TR
!BEGIN      $RR.SNR
            $PR
            $PR INPUT THE NAME OF THE RELATION WHICH YOU WOULD LIKE TO
            $PR EDIT?
            $SR.I #NME
            $PR #NME IS THE RELATION IN WHICH YOU WOULD LIKE TO EDIT? (Y/N)
!RESTAR     $SR.I #ANS
            $IF,(#ANS=N) $JU !BEGIN
            $JE !ERR
            $IF,(#ANS=Y) $JU !CONT
!ERR        $PR PLEASE TYPE 'Y' OR 'N'...
            $JU !RESTAR
!CONT       $ED #NME
            $JE !ERR1
            $PR
            $PR WOULD YOU LIKE TO CHANGE, DELETE, OR INSERT A TUPLE?
!RESTA      $SR.I #ANS
            $IF,(#ANS=CHANGE) $JU !CHLN
            $IF,(#ANS=DELETE) $JU !DELN
            $IF,(#ANS=INSERT) $JU !INLN
            $JU !ERR2
!ERR1       $PR
            $PR #NME IS A NON-EXISTING RELATION
            $PR
            $PR WOULD YOU LIKE TO EDIT ANOTHER RELATION? (Y/N)
!REST       $SR.I #ANS
            $IF,(#ANS=Y) $JU !BEGIN
            $JE !ER1
            $IF,(#ANS=N) $JU !END1
!ER1        $PR
            $PR PLEASE TYPE 'Y' OR 'N'...
            $JU !REST
            $PR
!ERR2       $PR
            $PR PLEASE TYPE IN CHANGE, DELETE, OR INSERT...
            $JU !RESTA
            $PR
!CHLN       $PR
            $PR
            $PR INDICATE WHICH TUPLE YOU WOULD LIKE TO CHANGE BY
            $PR GIVING CORRESPONDING LINE NUMBER?
            $SR.I #LNC
            $DI #NME #LNC 1
            $JE !CHLN
            $PR
            $PR
            $PR IS THIS THE TUPLE THAT YOU WOULD LIKE TO CHANGE? (Y/N)
!RESTA1     $SR.I #ANS
            $IF,(#ANS=N) $JU !CHLN
            $JE !ERR3
            $IF,(#ANS=Y) $JU !CONT1
!ERR3       $PR
            $PR PLEASE TYPE IN 'Y' OR 'N'...
            $JU !RESTA1
            $PR
!CONT1      $PR
            $PR THIS IS THE TUPLE THAT YOU WOULD LIKE TO CHANGE.
```

```
          $PR
          $PR WOULD YOU LIKE TO CHANGE ANOTHER TUPLE? (Y/N)
!RESTA2   $SR.I #ANS
          $IF,(#ANS=N) $JU !END
          $JE !ERR4
          $IF,(#ANS=Y) $JU !CHLN
!ERR4     $PR
          $PR PLEASE TYPE IN 'Y' OR 'N'...
          $JU !RESTA2
!DELN     $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 !DELN
          $PR
          $PR
          $PR IS THIS THE TUPLE THAT YOU WOULD LIKE TO DELETE? (Y/N)
!RESTA3   $SR.I #ANS
          $IF,(#ANS=N) $JU !DELN
          $JE !ERR5
          $IF,(#ANS=Y) $JU !CONT2
!ERR5     $PR
          $PR PLEASE TYPE IN 'Y' OR 'N'...
          $JU !RESTA3
!CONT2    $DE #LND
          $JE !ERR6
          $PR
          $PR TUPLE #LND HAS BEEN DELETED
          $PR
          $PR WOULD YOU LIKE TO DELETE ANOTHER TUPLE? (Y/N)
!RESTA4   $SR.I #ANS
          $IF,(#ANS=N) JU !END
          $JE !ERR7
          $IF,(#ANS=Y) JU !DELN
!ERR6     $PR
          $PR #LND IS A NON-EXISTING TUPLE
          $PR
          $PR WOULD YOU LIKE TO DELETE ANOTHER TUPLE? (Y/N)
          $JU !DELN
          $PR
!ERR7     $PR
          $PR PLEASE TYPE 'Y' OR 'N'...
          $JU !RESTA4
          $PR
!INLN     $EL W9
          $PR
          $PR INDICATE THE TUPLE THAT YOU WOULD LIKE TO INSERT A NEW
          $PR TUPLE AFTER BY GIVING THE CORRESPONDING LINE NUMBER.
          $SR.I #LNI
          $PR
          $DI #NME #LNI 1
          $JE !INLN
          $PR
          $PR
          $PR IS THIS THE TUPLE THAT YOU WANT TO INSERT AFTER? (Y/N)
!RESTA5   $SR.I #ANS
          $IF,(#ANS=N) JU !INLN
          $JE !ERR8
          $IF,(#ANS=Y) JU !CONT3
!ERR8     $PR
          $PR PLEASE TYPE 'Y' OR 'N'...
```

```
!CONT9    $AS 9=W9
          $PR INPUT THE TUPLE(S) IN WHICH YOU WOULD LIKE INSERTED...
          $PR    NOTE:
          $PR            FOLLOW THE FORMAT OF THE TUPLE DISPLAYED TO
          $PR            THE TERMINAL.
          $PR
          $PR            IF YOU DESIRE TO STOP INSERTING, TYPE '@' IN
          $PR            THE FIRST COLUMN.
          $PR
          $PR
          $PR
          $DI #NME #LNI 1
!READ     $SR,I #TUP
          $JE !CONTUP
!CONTUP   $IF,(#TUP=@) $JU !EXIT
          $JE !CONTP
!CONTP    $PR,F,9,#TUP
         $JE !COTUP
!COTUP    $JU !READ
!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 !END
          $JE !ERR8
          $IF,(#ANS=Y) JU !INLN
!ERR8     $PR
          $PR PLEASE TYPE 'Y' OR 'N'...
          $JU !RESTA6
          $PR
!END      $UP
          $PR
          $PR WOULD YOU LIKE TO EDIT ANOTHER RELATION? (Y/N)
!RESTT    $SR,I #ANS
          $IF,(#ANS=N) $JU !END1
          $JE !ERRR4
          $IF,(#ANS=Y) $JU !BEGIN
!ERRR4    $PR
          $PR PLEASE TYPE IN 'Y' OR 'N'...
          $JU RESTT
!END1     B2
          $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 COND1                PIC X(1) VALUE "N".

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 X(3).
       03 OCC-OF-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 NEXT-COLUMN          PIC 999   VALUE 1.
        02 RES                  PIC XXX   VALUE SPACES.
        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-O    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
        ELSE
           DISPLAY "*****  ERROR  *****"
           GO TO MAIN-ROUTINE.

    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
```

```
            ELSE
               IF NUM = 2
                  MOVE C-1 TO H-3
                  MOVE "0" TO H-2, H-1
                  MOVE HOLD TO NO-OF-ATT
               ELSE
                  MOVE COLUMN-L TO HOLD
                  MOVE HOLD TO NO-OF-ATT.

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

         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.

         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 INFOMAT.

      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 COND1
         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 ' '.
         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 MAXIUM LENGHT ***'
             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..
```

END

DATE
FILMED

9 — 83

DTIC