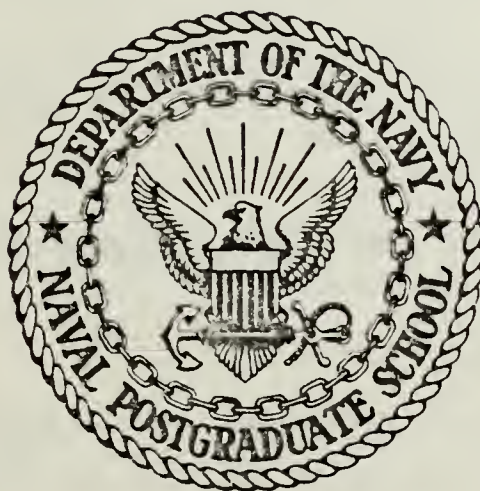


DUDLEY KNOX
NAVAL FC
MONTEREY CALIFORNIA 93943

NAVAL POSTGRADUATE SCHOOL

Monterey, California



THESIS

THE CREATION OF A CENTRAL DATABASE
ON A MICROCOMPUTER NETWORK

by

John G. Boynton
and
Ronald G. Nichols

March, 1984

Thesis Advisor:

N. R. Lyons

Approved for public release; distribution unlimited

T214354

REPORT DOCUMENTATION PAGE

READ INSTRUCTIONS
BEFORE COMPLETING FORM

1. REPORT NUMBER		2. GOVT ACCESSION NO.	3. RECIPIENT'S CATALOG NUMBER
4. TITLE (and Subtitle) The Creation of a Central Database on a Microcomputer Network		5. TYPE OF REPORT & PERIOD COVERED Master's Thesis March, 1984	
		6. PERFORMING ORG. REPORT NUMBER	
7. AUTHOR(s) John G. Boynton Ronald G. Nichols		8. CONTRACT OR GRANT NUMBER(s)	
9. PERFORMING ORGANIZATION NAME AND ADDRESS Naval Postgraduate School Monterey, California 93943		10. PROGRAM ELEMENT, PROJECT, TASK AREA & WORK UNIT NUMBERS	
11. CONTROLLING OFFICE NAME AND ADDRESS Naval Postgraduate School Monterey, California 93943		12. REPORT DATE March, 1984	
		13. NUMBER OF PAGES 218	
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) network, database, microcomputer, software design and system development			
20. ABSTRACT (Continue on reverse side if necessary and identify by block number) This thesis discusses the design and development of a central database on a network of microcomputers. It provides an overview of the methodology utilized in creating the system, along with the problems associated with a central database. The thesis includes the source listings for the creation of the system and a discussion of the difficulties of controlling contention within the networked database environment.			

Approved for public release; distribution unlimited.

The Creation of a
Central Database
on a Microcomputer Network

by

John G. Ecynton
Major, United States Army
B.S., United States Military Academy, 1972

and

Fonald G. Nichols
Lieutenant Commander, SC, United States Navy
B.S., Ohio State University, 1974

Submitted in partial fulfillment of the
requirements for the degree of

MASTER OF SCIENCE IN INFORMATION SYSTEMS

from the

NAVAL POSTGRADUATE SCHOOL
March 1984

1

ABSTRACT

This thesis discusses the design and development of a central database on a network of microcomputers. It provides an overview of the methodology utilized in creating the system, along with the problems associated with a central database. The thesis includes the source listings for the creation of the system and a discussion of the difficulties of controlling contention within the networked database environment.

TABLE OF CONTENTS

I. INTRODUCTION 7

II. METHODOLOGY 9

III. SYSTEM DEVELOPMENT 16

 A. SOFTWARE 16

 1. User Interface With QDR System 17

 2. Multi-User Environment 26

 3. System Security 28

 4. Flexibility And Maintenance 29

 B. HARDWARE 33

 1. Selected Hardware 34

 2. Hardware Integration 35

 3. Hardware Limitations 35

 C. TESTING 36

 1. Unit Testing 36

 2. Integration and System Testing 37

 3. Response Times 38

IV. CONCLUSIONS AND RECOMMENDATIONS 40

 A. CONCLUSIONS 40

 B. RECOMMENDATIONS 41

APPENDIX A: GLOBAL MEMORY DEFINITIONS 43

APPENDIX B: QDR DATABASE FILE STRUCTURES 51

APPENDIX C: DATA ELEMENT DEFINITIONS 55

APPENDIX D: QDR PROGRAMS 57

LIST OF REFERENCES 216

BIBLIGGRAPHY 217
INITIAL DISTRIBUTION LIST 218

LIST OF FIGURES

2.1	Basic QDR Flw and Initial QDR Receipt	10
2.2	Qdr Update and Close	11
2.3	User's System Hierarchy	13
2.4	Supervisor's System Hierarchy	14
3.1	Logon Prompt	18
3.2	Invalid I.D. Message	19
3.3	Currently Logged Message	19
3.4	Login Prompt	19
3.5	Main Menu	20
3.6	Verification Message	21
3.7	Initial Entry Screen	22
3.8	Bailout /Change Option	22
3.9	First Screen of data	23
3.10	Second Screen of Data	24
3.11	Response to FOST Choice	25
3.12	Feedback to Analyst	25
3.13	Database Handler Access Codes	27
3.14	Data Base Handler Return Codes	28
3.15	Response Times	38

I. INTRODUCTION

Current economic trends have brought about an increased awareness of the need for productivity gains in the workplace. Like all facets of business, government is finding increased pressures to reduce expenditures and still provide service to the people. To meet the challenges of reducing costs and maintaining service levels, government managers are looking toward office automation and computerization to increase individual productivity. The Navy Fleet Material Support Office (FMSC), like most Government Agencies, is doing its part to improve productivity.

The Defective Material Section of FMSO (Code 91423) is designated as the overall monitor for the quality deficiency management information reporting system. A Thesis entitled A System Analysis and Design For Updating the Internal Tracking of the Quality Deficiency Reporting System at the Navy's Fleet Material Support Office by Michael D. Carriger recommended the development of a prototype network of inexpensive microcomputers and the creation of a Central Database System. This prototype system will demonstrate the feasibility of automating the QDR Processing Procedures, and will allow the evaluation of processing with automated techniques. Additionally, the prototype will provide the basic design for future QDR Systems and its interaction with users. [Ref. 1]

Current microcomputer technology has allowed very powerful systems to be created at relatively low costs. Microprocessors with over 512,000 characters of memory can process over 500,000 instructions per second. Secondary storage units can access over 35 million characters of data at the rate of 5 million bits per second. Relational

Database systems allow microcomputers to create, update, and manage large databases of information at relatively low costs.

The purpose of this Thesis is to develop a Prototype Database Management Information System for use at the Defective Material Section of FMSO (Code 91423). This system will utilize current microcomputer technology and off-the-shelf hardware and software. Application programs will be generated with a high level database manipulation language. For this application, dBASE II (by Ashton-Tate), IBM-PC microcomputers, PCnet (by Orchid Technology), and 20 MB hard disk storage devices (by Tallgrass Technology) are utilized to create the Management Information System. This hardware and software was selected because: 1) it had already been evaluated and was in use at other sections of FMSO; 2) it appeared that it could meet the processing requirements for the QDR System; 3) it could be easily obtained with minimal cost to the project; and 4) it could be incorporated into both the short term and long term processing goals for FMSO [Ref. 1].

The major areas of concern for the project center around: 1) contention caused by multiple users accessing the same Data Base Files over a microcomputer network; 2) Security logon protection for the system; 3) Flexibility to respond to ad hoc information requests; and 4) providing meaningful system dialog for untrained computer users.

II. METHODOLOGY

The development of the Quality Deficiency Reporting (QDR) System was based on modern software engineering and design principles. Data flow diagrams, structure charts, and a high level programming language aided in the creation of the system. Using top-down design to provide a logical basis for development, the software creation involved: 1) studying and understanding the QDR process, 2) identifying at least one method of solving the problem, 3) creating data flow diagrams to show the gross data transformations, 4) using the data flow diagrams to construct a structure chart, and 5) describing each abstraction used in the solution in a manner that lends itself to eventual coding in a high level language. [Ref. 2]

The initial study of the QDR System was based on Michael D. Carriger's thesis work. This provided much of the background information that was necessary to formulate a possible solution. Based on the operational environment and the users' level of computer familiarity, it was decided that a menu driven system be created. This would provide an easy to understand interface for the unfamiliar user. Data flow diagrams were generated to identify the transformation of data from input to output. This provided a pictorial representation of the data used by the QDR System and established a means of identifying the changes that took place during the life of a QDR Case (See figure 2.1 and 2.2).

The data flow diagrams provided the basis for creating the system's hierarchical structure. By reviewing the basic transformations performed by the system, it became apparent that there are three main activities necessary for maintaining the Central Database. These basic activities

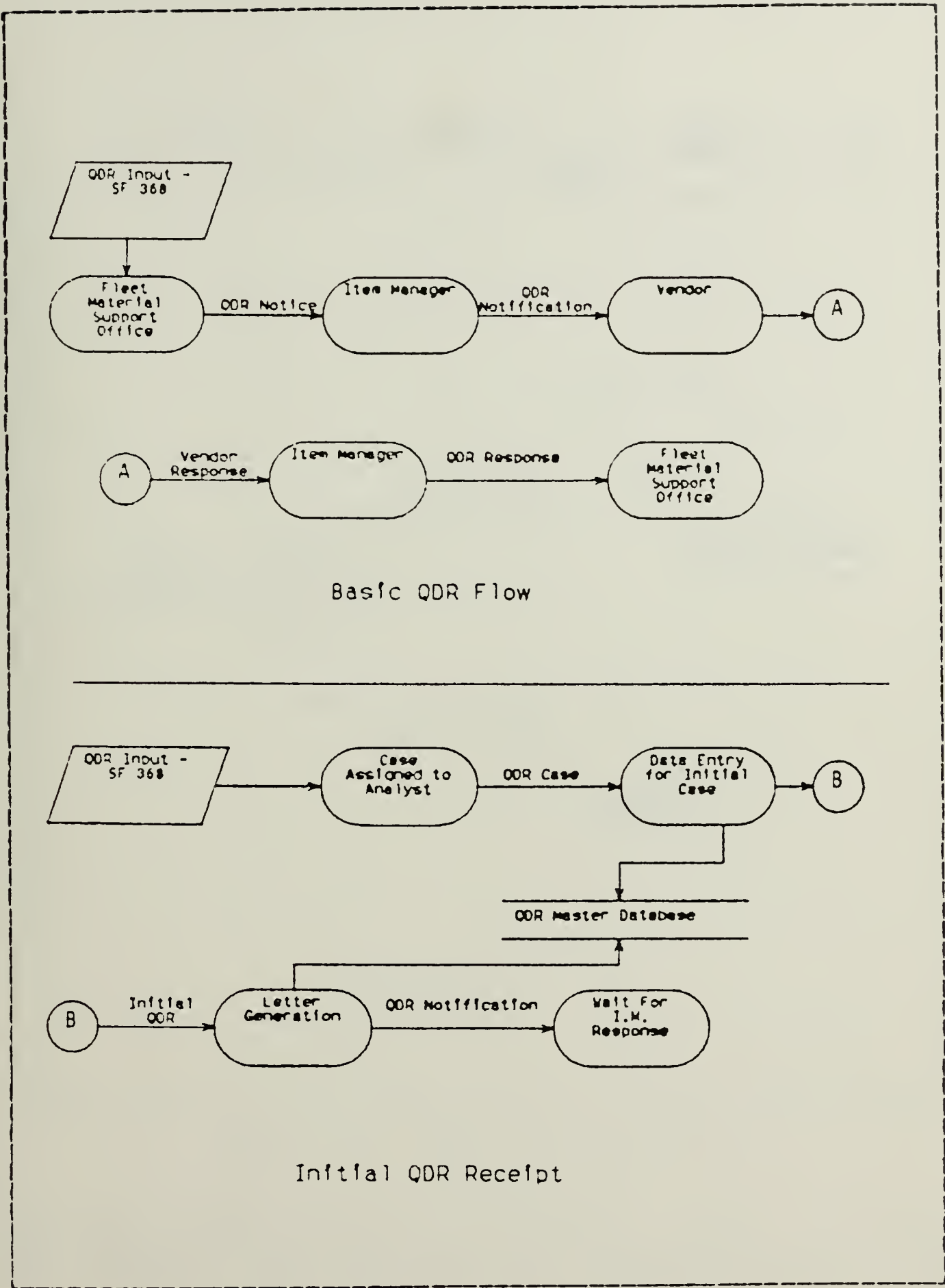


Figure 2.1 Basic QDR Flow and Initial QDR Receipt.

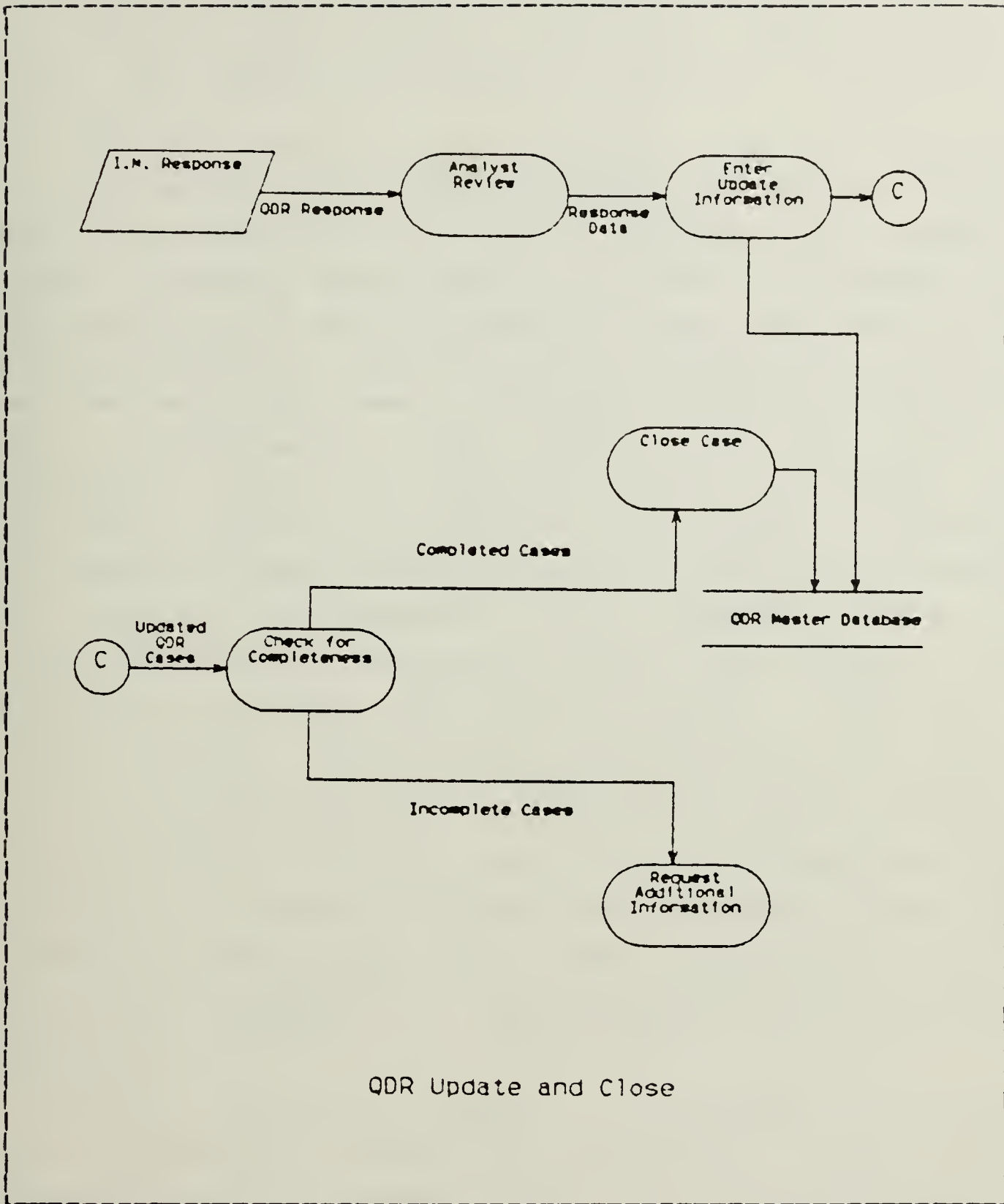


Figure 2.2 Qdr Update and Close.

prompted the creation of the Open Case, Update, and Close Case modules. All other functions associated with the QDR

System are support modules for presenting the Database information to the end user, maintaining support Database Files, and generating management information (See figures 2.3 and 2.4).

The QDR System was implemented using the command language for dBASE II, a Relational Database Product. This command language is a high level language that supports block structured development. It is an interpretive language that must be re-evaluated through each pass of the program execution. To ensure program clarity, the programs utilized meaningful names to identify variables and make the flow of information more apparent to the reader. Information hiding was utilized to reduce the amount of unnecessary information handled by each program. Required information is passed between programs as the data is needed for processing (See Appendix A for a complete listing of Passed Variables). Information hiding also conceals the processing algorithms used within a program. When interfacing programs, the programmer only needs to know what information passes between programs and not how the information is treated internally. The use of information hiding techniques reduces the complexity of systems development by allowing the programs to be developed independently based on interfacing requirements only. [Ref. 3] Another mechanism to simplify the programs is to include comment lines to make them more readable and understandable to maintenance personnel.

As each program was created, it was tested to ensure that it accurately performed the function for which it was designed and did not contain errors. As individual programs were tested, they were combined with other tested units to ensure compatibility between the various system segments. This integration testing was performed to ensure that the individual programs worked in conjunction with other

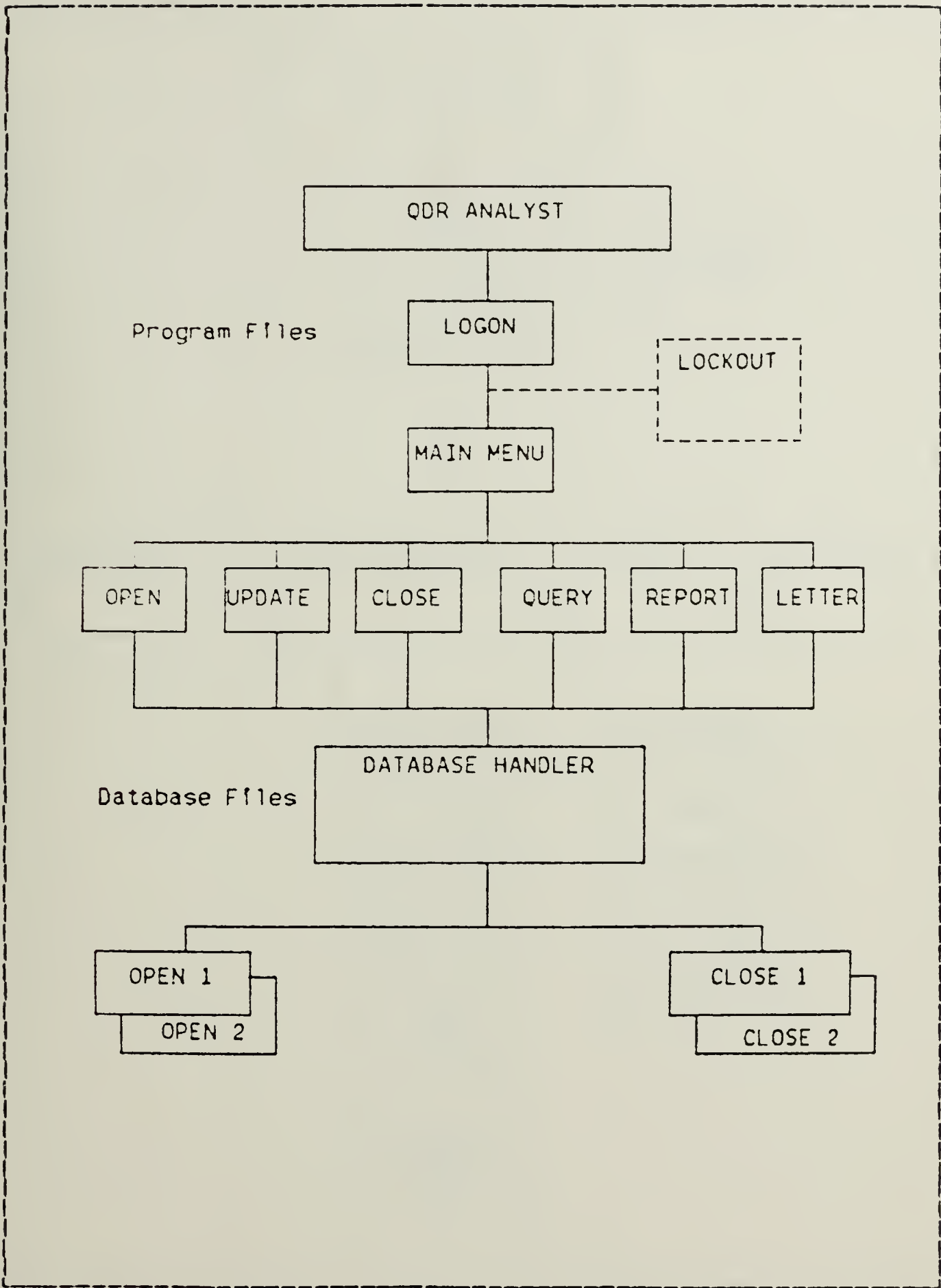


Figure 2.3 User's System Hierarchy.

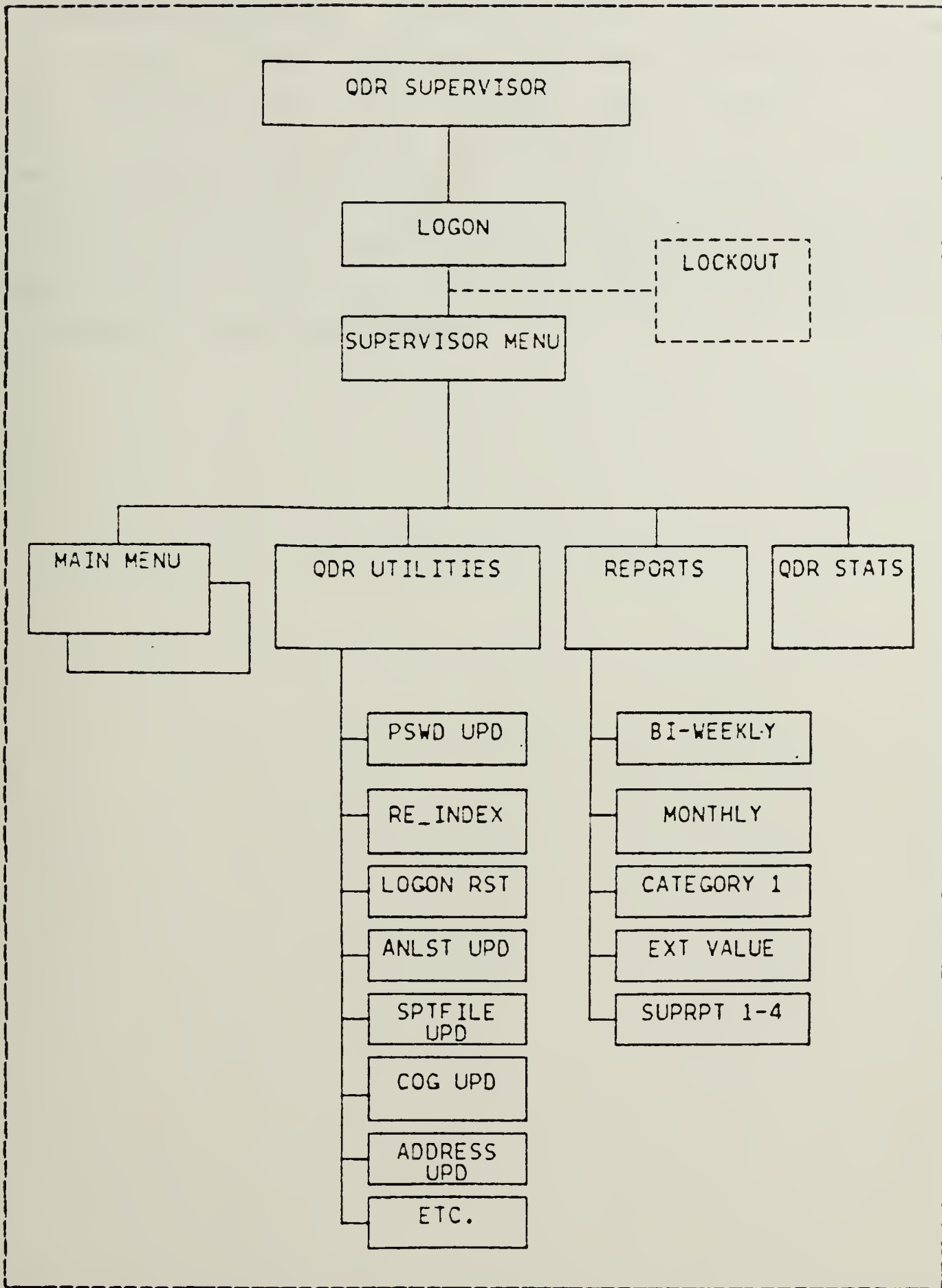


Figure 2.4 Supervisor's System Hierarchy.

programs and modules. As errors were detected in programs and modules, these bugs were corrected and then retesting was performed for both individual and integration tests. All of the initial tests were performed in a single user environment to reduce some of the system complexity. Once a program or module had completed both individual and integration testing, these units were then tested in the multi-user environment. Other aspects of the software development are discussed in more depth later.

III. SYSTEM DEVELOPMENT

The development of the QDR System considered many aspects of computer utilization. The system was designed as an integrated package of hardware and software that was to be utilized as a management information tool. To produce the desired results, both hardware selection and software development requirements were made considering the utilization of the system and the target group of system operators. Such things as user interface, the multi-user environment, security, system cost, and the availability of system components were integrated in the methodology of developing the QDR System.

A. SOFTWARE

The original QDR software design centered around a data base consisting of 8,000 - 10,000 records in the Open File and 16,000 - 20,000 records in the Closed File. Each of these records contained thirty data fields and required 275 characters of data. Headquarters level changes to QDR processing procedures expanded the scope of the data files considerably. New data requirements in support of the Product Deficiency Reporting System and Evaluation Program (PDRFP) increased the files to fifty four data elements requiring over 600 characters of data per QDR Case. (See Appendix B for a complete list of Database structures and Appendix C for the Data Element Definitions.) To accommodate these additional data items, the case records had to be split into two parts. This was necessary because of a restriction in the Database Management Software used for the system. The current DBMS allowed a maximum of thirty two data elements per database file.

The scope of the changes mentioned above required a total redesign of the QDR System. Up to that point, a substantial amount of design work and actual programming and testing had been completed. Although many of the "lessons learned" during the initial design could be applied to the redesign, and many of the initial algorithms could be modified and reused, the redevelopment effort required a significant amount of time and effort. All user interface programs had to be redesigned and, in many cases, reprogrammed to accommodate the new data elements and provide a meaningful interface. The change in scope drastically reduced the time available for complete testing, documentation and implementation thus resulting in the prototype system being more capable, but requiring additional effort in the above areas.

Much of the software development was aimed at providing a system that had an easy to understand user interface, could be used in a multi-user environment, provided a degree of security, and was maintainable. The following sections discuss each of these areas and provide some insight into how these were achieved. Throughout this chapter, the terms module and program are used interchangeably.

1. User Interface With QDR System

The interface between the user and the QDR System was a major point of concentration to ensure the usefulness of the entire system. In order to allow for the lack of experience of the analysts with automated tools and to avoid training costs for newly assigned personnel, the perception that the personal computers were really "QDR Machines" was utilized. The entire dialog of a user with the machine was centered around the functional aspects of the current working environment. A menu based system was utilized to guide the analysts through their case processing. At each

point where a choice could be made, the user was presented with an explicit message. If an invalid choice was made, the system would then provide a message indicating an error, and show the valid range of choices available to the user at that particular point. If the user was familiar with the range of valid inputs and did not make an entry error, then the messages would not appear on the screen. This allowed for the more experienced user to avoid some of the screen prompting. The System was written such that each of the user "QDR Machines" would automatically initialize itself and be in a state waiting for the user to logon (See figure 3.1).

```
ENTER YOUR USER I.D.  
XXXX
```

Figure 3.1 Logon Prompt.

This isolated the user from any of the underlying machine operating system and preparatory steps that are normally associated with putting today's microcomputers into operation. The only function that the computer was available for was the processing of QDR's. When an analyst terminated operations for the day the terminal would again go into a state ready for another user to logon or for the analyst to re-enter the system.

The procedure for an analyst to enter the QDR System is demonstrated by the following selection of screen displays and choices that are available. The process begins with entry into the system at the beginning of the day. Each analyst has been assigned a user I.D. and a unique

password by the system supervisor. The system has earlier been brought up by the supervisor. The first screen presented prompts for the analyst to enter his unique access identification (See figure 3.1). If the access I.D. is not on file, or was entered incorrectly, a message comes on the

```
ACCESS I.D. NOT ON FILE
PLEASE REENTER
```

Figure 3.2 Invalid I.D. Message.

screen (See figure 3.2). A check is also made to determine if the user is currently logged onto another terminal in the system. If currently logged on, then access is denied and the following message will then be displayed (See figure

```
USER CURRENTLY LOGGED ON
LOGON TERMINATED
```

Figure 3.3 Currently Logged Message.

3.3). If the correct Access I.D. is entered then the system will prompt to enter the password. The password requires

```
ENTER PASSWORD FOLLOWED BY <CR>
```

Figure 3.4 Login Prompt.

exact upper/lower case entry (Figure opn4). Three chances are given to successfully enter the password and if unsuccessful, the console is locked out and may only be put back into operation by the system supervisor. A successful logon will be followed by a greeting to the QDR System, and the user will be presented with the main menu which contains all of his processing options (See figure 3.5).

```
WELCOME TO THE QDR AUTOMATED TRACKING SYSTEM

1 - Open New Record
2 - Close Record
3 - Update Record
4 - Originate Letter
5 - Report Generation
6 - Query
7 - Exit from the System

Enter Your Choice
```

Figure 3.5 Main Menu.

From the main menu (figure 3.5), the analyst has the option to open a new case, update an existing case, close a case, originate a letter to an item manager, get a listing of all of his open cases in the QDR System, query the databases for information or to leave the QDR System.

As an example of the process required to open a new QDR case, the following sequence shows the screens as presented to the analyst. In order to open a case, the analyst chooses a "1" from the main menu and then is presented with a screen where verification of desire is required. This allows the analyst to change his mind before beginning the process and to return to the main menu. A choice of "1" puts the user in the case opening process (See figure 3.6).

***** OPEN NEW CASE *****

THIS PROGRAM ENABLES YOU TO OPEN A
NEW QDR CASE

1 - CONTINUE

2 - RETURN TO MENU

Figure 3.6 Verification Message.

The input screens presented to the user are designed with the source input document Standard Form 368 (SF 368) as the basis. Each of the items of information are captured from the numbered blocks of the SF 368. Where information is not identified on the form, yet is needed for the QDR case, input is requested at the location where most often written in by the analyst or by the originating office. The purpose of this was to maximize the ease and fluidity of data entry by the analyst by considering the physical location of the data as well as the logical relationship of the elements.

The first entry required was the date the case was received by FMSO. A standard (MMDDYY) format for dates was utilized throughout the QDR programs, based upon user specifications (See figure 3.7).

After entry of the national stock number, a prompt to verify the initial data is put on the screen. This enables the analyst to ensure that the correct case will be created and will alleviate a later need to delete an invalid case from the database (See figure 3.8). A choice of "2" allows the changing of any initial data item before continuing to the second screen.


```
***** ENTER DATA FOR THE NEW CASE *****
***** FROM SF 368 *****
```

```
DATE RECEIVED EY FMSO MMDDYY 'XXXXXX'
CAI 'X'
COG 'XX'
NSN 'XXX-XX-XXXX-XXXX'
```

Figure 3.7 Initial Entry Screen.

```
VERIFY ABOVE INFORMATION
YOU MAY NOT CHANGE IT AFTER THIS
WITHOUT STARTING OVER AGAIN
1 - CCNTINUE 2 - CHANGE 3 - EXIT
```

Figure 3.8 Bailout /Change Option.

The analyst is next presented with a full screen of data elements. The layout is such that the left side of the item labels contains the numbers relating to the SF 368 blocks. This portion is blank where the element is not on the SF 368. Following each element label is a reverse video representation showing the correct length of the input item. The inputs which are optional are marked by a <O> (See figure 3.9). The cursor moves from one data element to the next one as the analyst completes item entry. Any incorrect or out of bounds entry will result in a specific error message to the user showing the valid range of inputs. These error messages, when activated, appear on the last line of the screen. The user can thus consistently lock to a single location for status or error messages from the QDR System.


```

SF368
5.      NSN
        CATEGORY
        SMIC
1A.    UIC
3.      REPORT CONTROL
4.      DATE DISCOVERED      MMDDYY
6.      NOMENCLATURE
7.      FSCM                  <0>
8.      MFG. PART NUMBER     <0>
9.      SERIAL/LOT/BATCH     <0>
10.     CONTRACT/PO          <0>
        DOCUMENT NUMBER     <0>
11.     ITEM                  N OR O <0>
12.     DATE MFG/REP/OVHL    <0>
13.     CFN TIME AT FAILURE  <0>
14.     GOV FURNISHED MATL   <0>
15.     QTY: REC/INSP/DEF/STK
16a1.  TYPE/MODEL/SERIES    <0>
        a2. SERIAL NUMBER    <0>
        b.  NEXT HIGHER ASSY  <0>
        SUB-ASSEMBLY        <0>

```

```

***** CHECK PREVIOUS ENTRIES *****
CHOCSE 1- CONTINUE ENTRY  2- MAKE CORRECTIONS

```

Figure 3.9 First Screen of data.

Consistent with the previous choices of leaving a particular screen, the analyst has the ability to make changes before proceeding. The next screen of data presents the same basic format to the user, and allows the input of the second half of the data elements. The NSN and category of the case being input are echoed at the top of the screen so the analyst may keep track of them for later reference. Data items are calculated by the program where possible and then inserted into the screen at the appropriate point. The extended price is one such item which was previously hand computed (See figure 3.10).

Upon completion of data element entry, the analyst may elect to change an item, post the case or exit the opening program and go back to the main menu without posting the case. This is critical at this time in the entry


```

      UI
      UNIT PRICE          EXTENDED PRICE
18.  EST. CORRECTION COST          <O>
19.  WARRANTY - Y/N/U
20.  WORK UNIT CODE          <O>
21.  ACTICN/DISPOSITION -H/I/D/R/O  <O>
22.  DETAILS OF DISCREPANCY - FIRST 2
      LETTERS MUST BE DISCOVERY CODE
23A. ACTICN POINT
      DEFECT VERIFICATION CODE - N/O/U/Y <O>
      DEFECT RESPONSIBILITY - C/N/S/U/X <C>
      9Q
      ORIGIN CODE
30.  TYPE LOC
      TYPE DEFICIENCY

```

```

      1 - POST CASE
      2 - CHANGE DATA
      3 - EXIT WITHOUT POSTING

```

Figure 3.10 Second Screen of Data.

process. If a major mistake had been detected, it would be best to re-initiate the entry of a particular case instead of using the update program to change each item. This gives the analyst a final point where the process can start over without any interaction with the current cases. From the users point of view it is comforting to know that an earlier mistake could be eliminated prior to posting. On a busy network, the posting process may take a few minutes, thus the analyst is re-assured that "all is well" by a screen giving a status report on the process (See figure 3.11).

After successful assignment of a case number and posting to the database, the case number is displayed on the screen (See figure 3.12). After noting the case number on the SF 368 for any future reference as needed, the analyst can clear the screen by pressing any key, and then will be back at the point where he may input another new QDR case or return to the main menu.


```
CASE BEING POSTED TO DATA BASE
PLEASE STANDBY
```

```
*** DO NOT INTERRUPT ***
```

Figure 3.11 Response to POST Choice.

```
CASE NUMBER OF THE NEW CASE
'400192A'
```

```
PRESS ANY KEY TO CONTINUE
```

Figure 3.12 Feedback to Analyst.

The above sequence gives a flavor of the screens and messages that are present in the QDR System. Foremost consideration in design of screens and menus was the ease of use by the analysts. The screen design in a pattern which matched the source document as much as possible while considering additional input requirements led to a clean, easy to understand representation. The error messages were directed at identifying a specific range of acceptable input values and presenting these to the analysts for their review. Consistency of input parameters was maintained to enable the user to react to prompts and choice points thus requiring a minimum of additional thought and attention to the process of data entry and interaction with the "QDR Machine" itself.

2. Multi-User Environment

The QDR System uses dBASE II to handle all aspects of the system's operations. This database management product is designed for a single user and does not provide the locking mechanisms necessary for a multi-user environment. To overcome this deficiency, a Database Handler was created to control access to the various database files. Access to the Database is achieved by calling the Database Handler routine and providing it with a two character alpha-numeric type code which represents the type of the desired access (See figure 3.13). The Database Handler will either expect additional parameters or will provide information depending on the selected access type.

In order to perform record locking or file write functions, the Database Handler must first establish write access to the database file that is being written to. A special "File Status" data file provides the mechanism to determine write access. As each user process calls the Database Handler for file write transactions, the file status is checked to see if the file is currently locked by another process. If the file is locked, the Database Handler enters into a test and wait loop until the file is made available. When the file is unlocked, the Database Handler will then lock the required file by placing the User I.D. of the operator into the file status file. There is a point of contention at the moment the file is released by a process. Each terminal on the network has its own copy of the Database Handler and as such, when a file is unlocked, other processes will perform the same locking action. To ensure that a process has obtained write access, a verification check is made just prior to actually performing the write operation. If write access has been obtained, the Database Handler will perform the write and release the data

The first character of the Access Code represents the file being accessed and the second character represents type of access desired.

First Character	Database File Used
1	OPEN1
2	OPEN2
3	CLOSE1
4	CLOSE2

Second Character	Database Action
A	Read - NSN Access (See note 1)
B	Read - Case Number Access
C	Write - Unlock Record
D	Read/Lock - NSN Access
E	Read/Lock - Case Access
F	New Record Creation (See note 2)
G	Record Unlock
H	Read - Record Number Access
I	Skip/Read - Record Number Access

Note 1: Open2 and Close2 do not have NSN Access and default to Case Number Access.

Note 2: Close1 and Close2 create new records from the Open File records being closed.

Figure 3.13 Database Handler Access Codes.

file for others. If write access is not obtained, the test and wait loop is entered again.

To perform a write operation, the entire database file is locked so that no one else can write to it. When an individual record must be updated, it is undesirable to leave the Database file locked while the operator is making updates to the record. To prevent this, a record locking

capability was added to the Database Handler. This is done by including a timestamp data element in each data record. Prior to retrieving a record for update, the Database Handler checks to see if the record has previously been locked. Any attempt to update a locked record will result in a code being returned to the calling program/module (See

The Data Base Handler will return a one digit Code indicating the Success/Failure of a Data Base Access.

Return Code	Definition
0	Data Base Access Successful
1	Record Currently Locked
2-8	Unassigned (Available for future growth)
9	Record Not Found

Figure 3.14 Data Base Handler Return Codes.

Figure 3.14). As with general write operations, the database must be temporarily locked to allow the timestamp to be written out to the file. This record locking mechanism allows multiple users to function without unintentionally overwriting information.

3. System Security

Because of the amount of data held by the QDR System and the value of the information to FMSO Code 91423, the QDR System required some degree of security. There are basically two levels of security available for the system. The first level of security is the protection of the System Disks. The Master Network Station is the gateway to the QDR

Programs and Data Files. By keeping the master station operating disks under lock and key, the system is not generally accessible to unauthorized personnel. Access to these disks should be controlled by the system supervisor or his assistant supervisor.

The second level of security involves a logon and password system incorporated into the QDR Software. Each authorized user is provided a unique user identification code which will allow him access to the system. In addition to the required I.D. Code, a password is required to complete the logon procedure. The passwords may and should be changed periodically by the system supervisor to reduce the likelihood of unauthorized personnel becoming familiar with the passwords. To utilize the QDR System, the user accesses the system as described in the section on user interface. The user is given three attempts to access the System. If all three access attempts fail, the system will display an "Illegal Access Attempt" message and will lockout the terminal. The only way to return a locked terminal to an operational mode is to "reboot" the affected terminal.

Although this method of security is simplistic, it is the method most suitable for a system of this nature. As the value of the data held by the system increases, the security procedures should be reviewed to ensure they are adequate.

4. Flexibility And Maintenance

The fact that the QDR System could expect to undergo changes was considered in not only program development, but also in the database organization. Flexibility and maintainability of the entire system represented development objectives in order to support the earlier discussed design goals of modularity and information hiding.

IBASE II with its command language and relational database provided a powerful vehicle to construct the programs and databases for the system. The English-like quality of the command language provides the programmers a sense of code function over and above relying on algorithm inspection alone. Comments were spread throughout the program listing where they would assist understanding of specific portions of the processing, especially in QUERY and XCBHNDLR (See Appendix D for a complete set of QDR System Program Listings). Additionally, comments were provided in the program headers to identify critical information. Variables passed between the module of interest and all other modules as well as a list of subordinate and superordinate modules were provided. Maintenance of the programs could then be conducted with a knowledge of the current interface between the modules.

The structured programming technique of indentation was used to enhance readability and understandability of the code. This provided anyone reviewing the source listings with an easy to understand view of the control structures. Each level of control was indented to identify and clarify the hierarchy of control. Each hierarchical level can thus be traced from level to level by following the indentation pattern.

In addition to the general aids to maintainability described above, some specific areas were identified for likely future changes. Internally generated change was expected from assignment of different analysts, additional or modified internal reports and standard queries. External changes were likely in the areas of Cognizance Group (COG) assignments, changing addresses of Item Managers and report modifications.

The supervisor was provided with the means to update currently authorized users and their passwords via

the supervisor's main menu. The updating of COGs and Item Manager's information were also included. These were seen as routine housekeeping modifications which did not demand a programmer's attention.

If demand for specific, repetitive queries arise, the addition of this capability by maintenance programmers is very easy. Currently each analyst is able to receive a listing of each of his currently open cases as a standard query from his "report menu". To add any additional query would require modification of only one program module.

As an example, suppose that a commonly occurring query by all analysts was to receive a list of their open cases from a particular COG. The programmer would be able to provide this capability by adding only a few lines of code. The addition of a menu selection item would be accomplished by the following:

```
***** MENU ADDITICN
```

```
' 2 - List of assigned open cases for specific COG '
```

```
***** VALIDATING ENTRY
```

```
STORE T TO BADCCG
DO WHILE BADCCG
  @ LINE,COLUMN SAY 'Enter COG' GET ANSWER
  READ
  USE D:COGS
  FIND ANSWER
  IF # = 0
    @ 23,20 SAY 'COG NOT FOUND, TRY AGAIN'
  ELSE
    STORE F TC BADCCG
  ENDIF
ENDDO <BADCCG>
```

```
***** NOW FIND THE OPEN CASES FOR THIS ANALYST
***** CHCICE OF MEDIUM WAS MADE IN ORIGINAL MENU
```

```
USE D:OPEN1
DISPLAY CASE, NSN, NOMEN, $(DATES, 11, 5) FOR COG = ANSWER
. AND. WHO = C:WHO
```

This is but one of the many methods that could be used to provide the additional capability to the analysts.

The isolation of the functions within a single module combined with the power of a relational database are a definite asset to the programmer in extending the use of the system to its users.

While the capability to extend the functions provided has been built in, the decision to do so should not be taken without consideration of the impact on the system as a whole. An extension such as the one just described could be helpful and not be detrimental to the system operation if properly implemented. A choice would have to be made;

1. Restrict the use of this option to low use periods.
2. Implement it as a standard internal report, once a week for example.
3. Create an index file based on either COG or analyst and keep these updated during normal processing.

The supervisor must be aware of the impact of these alternatives. What in the first view looks like a very easy and useful method of producing the listings, may potentially cause system-wide problems. The two most likely drawbacks would be slowing processing response time to an unacceptable level or causing additional index files to steal precious space on the system hard disk. The first option would allow analysts to retain greatest flexibility, however it would be difficult to implement and control. The internal indexing of the databases for normal processing includes neither COG nor Analyst. For this reason a request as outlined above would require a sequential search of the OPEN1 database. A process that could take up to 30 minutes, not likely a satisfactory solution!

The second option has the advantage of being easy to control, has no impact on day to day processing

or storage and meets the requirement to provide a list to each analyst. Analyst flexibility would be compromised and the required periodicity would have to be determined.

Choice number three would allow the lists to be generated upon demand. The major drawback would be the addition of an additional index that would have to be updated at each case creation, update and closing. This would add overall processing requirements and thus slow down the complete system. Additionally, the index would require space on the hard disk, a critical resource.

The proper choice for the supervisor and programmer combination would balance the users needs and the system realities. Although the above example shows a simple, easy extension it points out the necessary considerations which must be included in all decisions. The micro computer system, as well as the mainframe computer does have application limits. In the QDR System all current requirements have been met, and while designed for ease of maintenance and extendability, the latter should be implemented with discretion and caution.

B. HARDWARE

The QDR System is a combined software and hardware suite which performs management information and database management functions. The hardware selected for the system was comprised of multiple microcomputers, secondary storage devices, printers, monitors, keyboards, and network hardware with controlling software. The selection of the supporting hardware is vital to the operation of the central database system because it provides the mechanism for sharing the database files and operating programs. To meet the demands of the network operation, the hardware must be compatible

and allow the equipment to be integrated into a complete system.

1. Selected Hardware

The microcomputer selected for the QDR System was the IBM-Personal Computer (PC). This provided expandability and supported both networking and relatively large hard disk storage devices. At the Naval Postgraduate School prototype site, the network was composed of four PC's. Each PC was equipped with a keyboard, a color monitor and color controller board which allowed the experimentation with color interfaces for users. All of the PC's contained two double-sided double-density 320 KByte floppy disk units with controllers and network controller boards for Orchid Technology's PCnet. Two of the four PC's had 128 KBytes of random access memory (RAM) while the other two had 320 KBytes of RAM. The 320 KB systems were also equipped with AST Research's MegaPlus board which provided 64 KB of the 320 KB RAM, a clock/calendar, a serial input/output (I/O) port, and a parallel I/O port. These I/O ports allowed the connection of either printers or modems. Two printers were connected to the network (one to each of the PCs with I/O ports). One printer provided letter quality output through its daisy wheel print, while the other provided the capability of printing text and drawing graphs through its dot matrix print.

The personal computers with the additional memory and I/O capabilities were also outfitted with interface cards for Tallgrass Technology's 20 MByte Hard Disk Storage Units. These disk units provided up to four logical disk drives and contained built-in tape backup units which allowed saving archival information.

2. Hardware Integration

The integration of the hardware was largely completed by the equipment manufacturers. The controller boards for both the network and the hard disk interfaces were specifically designed to become an integral part of the IBM-PC. The software that controlled both the hard disk and the network were created to work in conjunction with the PC-DOS operating system and with each other. The importance of this interface between the manufacturers became very apparent as system integration testing began. The initial versions of the network and hard disk software were not completely compatible. As a result, the system was prone to locking up during operations that involved large amounts of disk accessing. The respective companies worked together to solve the lockup problems and made available the corrected versions. Once the corrected versions were installed, the lockup problem appeared to be alleviated and cleared the path for the creation of the Central Database System.

3. Hardware Limitations

A limitation of the selected hardware suite is the inability for a shared PC to access files located on another shared PC. This means that the shared PCs are limited in their ability to access the total database. During the early design phases, this was not considered a problem because each record only required 275 bytes of information. Assuming a combined load of 30,000 records in the Open and Closed Database Files, the system required less than 9 Mega Bytes of storage. Under the revised QDR processing requirements, the same load of 30,000 records required over 18 MB of storage. The 18 MB requirement does not include overhead for programs, support files, or indices required by the QDR System. This meant that the Open and Closed Database Files

needed to be split across two hard disk units. Since the user PCs can have access to multiple shared devices, this limitation only restricts the use of the shared PCs. By utilizing the shared PCs as network controllers only, the databases can be split across the network and accessed by all users.

C. TESTING

Testing was conducted throughout the development period of the QDR System. The testing approach used was to progress from unit testing of one module to integration of tested modules. Validation of these modules against design criteria was followed by system testing using the complete software and hardware package. [Ref. 4]

1. Unit Testing

Unit testing of modules represented the first level of testing. Once the program modules were coded and had been cleansed of any syntactical ailments, they were individually tested. Both testing harnesses and program stubs were used at different stages of program development. The top down design had identified those key modules needed to support the function of the system. The first modules coded and tested were thus the Database Handler (XDBHNDLR) and the Open Case Module (XOPEN2). Testing XDBHNDLR required development of a harness in order to input expected parameters and make available specific data to the module. The emphasis of this module was two-fold. First it was expected to be the program's interface with the case databases, and as such had to properly read and write specific files according to the "type code" presented to it. The second concern was contention. Stepwise testing of the module was conducted. A testing harness

with a sample data set was used to exercise the Database Handler until it performed properly. The second testing stage was to use multiple inputs over the network to isolate any contention problems and eliminate them. Concurrently the Open Case module was undergoing parallel testing using the database directly.

2. Integration and System Testing

Integration testing was then conducted to bring together the XDBHNDIF and XOPEN2 modules. The purpose of this stage was to ensure the interface between the modules was in accordance with design. Once these programs were in this stage, the same sequence was utilized to test the other main modules, and bring them up to the integrated level.

By late October the main processing modules were integrated and the limited system was operating satisfactorily. The system re-design and development discussed earlier caused testing to begin anew. At that time the operating system version was changed from PC DOS 1.1 to PC DOS 2.0. The network and hard disk software were also upgraded.

Limited time for completion of system development and coding resulted in only partial system testing by mid January on the NPS prototype system. During demonstration at the FMSO site, the QDR system operated properly as a single user system but not with multiple simultaneous transactions over the network. The cause was not determined at that time. Orchid Technology and Tallgrass were contacted to discuss the difficulties, resulting in an updated release of both software packages being sent to NPS.

Limited system testing on the NPS network indicated that the problem had been corrected. Subsequent operation of the QDR system by personnel at FMSO (with the

upgraded software) was not successful. Multiuser system failure could not be duplicated at NPS. Reasons for the network failure at FMSO have not been identified.

3. Response Times

Multiple users and large databases affect response time on micro computer systems to a large degree. Figure 3.15 shows the time required for specific operations with different system loads. Where depicted, multiple users are performing the exact same operations simultaneously.

OPERATION	SIZE	100 rcds			6000 rcds		
	Users	1	2	3	1	2	3
1. Pcst Case:		:26	1:15	2:25	:59	2:29	3:36
2. Case Update							
1st screen		:38	1:27	1:58	:52	1:59	2:37
2d screen		:17	:37	1:12	:24	:50	1:08
3. Case Closing		1:28	3:25	4:24	1:48	4:07	6:10

Times shown as minutes:seconds. Where time represents multiple users, the time shown is completion time for all users

Figure 3.15 Response Times.

These times give an idea of the different response the user can expect with the loadings as indicated. The difference in times used by adding a second user is not significant. However, if numerous analysts were conducting operations at the same time then the times could increase to a level where input operations would be significantly

delayed. The differential in response times for added users reflects two items. The first is the contention induced for packet access to the network, which represents the main portion of the delay time. The second is a delay due to internal checking in XDBHNDLR to allow only one of the asynchronous processes access to the database. Time differences between the operations reflect the amount of data which has to be stored into the database as well as the number of different databases which must be accessed to complete the operation (2 for posting a case, 4 for closing a case).

IV. CONCLUSIONS AND RECOMMENDATIONS

A prototype system is designed to provide an interface for users and acquaint them with the potential value of an automated operation. It allows the user to see how the final system will present and accept data and provides the opportunity to modify the interface before final system implementation. Prototyping allows the rapid development of a system but generally does not contain all of the capabilities of an operational system.

The QDR System, although a prototype, was designed with capabilities beyond normal prototyping. It provides not only user interface capabilities, but also full database management capabilities. The additional features were added to allow the QDR System to be placed in an operational environment to be tested and to acquaint the user with automated systems and their operations. To provide functionality, the system was designed for multiple users to access the database files. This meant that the system allowed for record locking, multiple read/write operations, and security access to the system.

A. CONCLUSIONS

The design and implementation of an automated Quality Deficiency Reporting System prototype has been described in this study. A working prototype has been established and is available for future evaluation. Conclusions drawn from this development include:

1. The prototype software design meets current QDR processing requirements and includes PDREP derived data elements identified for future inclusion in the QDR System.

2. There are two main operating limitations with the current system; speed of processing with multiple users and limited system capacity to meet increased QDR requirements.

3. The NPS prototype system demonstrated the feasibility of accomplishing QDR processing on a microcomputer based system.

4. If new software and hardware become available to alleviate network and database limitations, the prototype could be modified to provide an operational system.

B. RECOMMENDATIONS

1. Continue development of a microcomputer based system to provide automation of the QDR workflow.

2. Consider migration to a minicomputer or mainframe computer using the basic prototype design, in order to allow for faster response time and growth potential.

The recommendations above provide for the user to become familiar with the automation capabilities that can be implemented in the QDR processing environment. Additionally, it opens up the channels for user feedback to system designers that are working on future versions of QDR support systems. The initial design considerations that went into creating the prototype system are valid for mini/mainframe computer implementation. The data dictionary used, the menus and interface screens, and the security considerations will remain valid in both the microcomputer and mini/mainframe computer environments. The use of a higher level language in the prototype system provides the potential to directly convert the algorithms to a new system.

If it is more practical to continue utilizing the microcomputer network, the size of the database could be expanded

by the use of 35 MF hard disk units, thus approximately doubling the system capacity. Additional speed for the microcomputer network could possibly be achieved by converting the file and record locking/unlocking operations into assembly or machine language routines (although this is not recommended because of future maintenance headaches). Another avenue to explore is the acquisition of a multi-user version of dBase II that was recently announced. This would allow the elimination of file locking and control currently accomplished by the QDR System software, and with a small amount of reprogramming the XDBHNDLR program could be eliminated.

The prototype QDR System provided a starting point for future growth in the QDR processing environment. The groundwork that has been laid can be utilized for either a continuation in the microcomputer realm or can be utilized with larger computers.

APPENDIX A
 GLOBAL MEMORY DEFINITIONS

There are various types of memory variables utilized by the QDR System. These variables are divided into Global and Local Variables. Global Variables are used to transfer data between programs and modules and are designated by either M: or C:. Local Variables are utilized for internal control within programs. These variables are identified by U: for XUPDATE, H: for XDBHNDLR, O: for XOPEN2, etc.

The variables listed below are the Global Variables utilized by the QDR System. They are presented as:

Variable Name	Using Modules
Description Of Variable	

C:JULIAN	(CLOSREC, LOGON, MENU1, XDBHNDLR, XOPEN2, XUPDAT, XXBISTAT, XXMNSTAT)
MEMORY VARIABLE WHICH HOLDS TODAY'S JULIAN DATE. THIS DATE IS GENERATED BY ACCESSING THE SYSTEM CALENDAR AND CONVERTING TO A JULIAN DATE.	
C:WHO	(CLOSREC, COGCNT, C-REASGN, DEPACK, LOGON, MENU1, OCASERPT, QUERY, RPTMENU, STATGEN, SUPMENU1, SUPRPTS, SUPRPT2, UTILMENU, UTILNDX, XDBHNDLR, XOPEN2, XUPDAT, XXBISTAT, XXMNSTAT)
MEMORY VARIABLE THAT HOLD THE LOGON IDENTIFICATION OF THE ANALYST. THIS IS CAPTURED DURING THE LOGON PROCESS.	
M:ACTDISF	(XDBHNDLR, XOPEN2, XUPDAT)
MEMORY VARIABLE WHICH IDENTIFIES THE ACTION/DISPOSITION INSTRUCTIONS. ORIGINALLY CAPTURED FROM BLOCK 21 OF THE SF 368.	
M:ACTPT	(XDBHNDLR, XOPEN2, XUPDAT)
MEMORY VARIABLE WHICH IDENTIFIES THE ACTION POINT.	

M:ACTTKN (XDBHNDLR, XUPDAT)
MEMORY VARIABLE WHICH IDENTIFIES THE ACTION CODE. ORIGINAL-
LY CAPTURED FROM BLOCK 21 OF THE SF 368.

M:CASE (CLOS REC, C-REASGN, XDBHNDLR, XOPEN2,
XUPDAT)
MEMORY VARIABLE USED TO CAPTURE THE NUMBER OF THE QDR CASE.

M:CAT (C-REASGN, XDBHNDLR, XOPEN2, XUPDAT)
MEMORY VARIABLE WHICH HOLDS THE CLASS OF THE QDR CASE.
CASES MAY BE EITHER CATEGORY 1 (HIGH PRIORITY) OR CATEGORY 2
(NORMAL PRIORITY). ORIGINALLY CAPTURED FROM THE SF 368 OR
QDR MESSAGE.

M:CAUSEC (XDBHNDLR, XOPEN2, XUPDAT)
MEMORY VARIABLE WHICH IDENTIFIES THE CAUSE CODE.

M:CCOST (XDBHNDLR, XOPEN2, XUPDAT)
MEMORY VARIABLE WHICH IDENTIFIES THE ESTIMATED CORRECTION
COST. ORIGINALLY CAPTURED FROM BLOCK 18 OF SF 368.

M:CLOSE (CLOS REC, XOPEN2, XUPDAT)
MEMORY VARIABLE WHICH HOLDS THE DATE THE CASE WAS CLCSED.
ORIGINALLIY ENTERED UFCN CLOSING THE CASE.

M:COG (C-REASGN, XDBHNDLR, XOPEN2, XUPDAT,
XXBI STAT)
MEMORY VARIABLE WHICH HOLDS THE COGNIZANCE SYMBOL FOR THE
ASSOCIATED NSN. ORIGINALLY CAPTURED FROM BLOCK 5 OF THE
SF 368.

M:COSTC (XDBHNDLR, XUPDAT)
MEMORY VARIABLE WHICH IDENTIFIES THE COST CODE.

M:CR (CLOS REC, XDBHNDLR, XUPDAT)
MEMORY VARIABLE WHICH IDENTIFIES THE CREDIT CODE.

M:DATES (CLOS REC, C-REASGN, XDBHNDLR, XOPEN2,
XUPDAT, XXBI STAT, XXMNSTAT)
MEMORY VARIABLE WHICH HOLDS THE CONCATINATION OF THE MAJOR
DATES ASSOCIATED WITH THE QDR SYSTEM.

M:DEF (XDBHNDLR, XOPEN2, XUPDAT)
MEMORY VARIABLE WHICH IDENTIFIES THE TYPE DEFECT CODE.

M:DEFR (XDBHNDLR, XOPEN2, XUPDAT)
MEMORY VARIABLE WHICH HOLDS THE DEFECT RESPONSIBILITY CCDE.

M:DEFV (XDBHNDLR, XOPEN2, XUPDAT)
MEMORY VARIABLE WHICH HOLDS THE DEFECT VERIFICATION CODE.

M:DETAILS (XDBHNDLR, XOPEN2, XUPDAT)
MEMORY VARIABLE WHICH IDENTIFIES THE DETAILS OF THE QDR.
ORIGINALLY CAPTURED FROM BLOCK 22 OF THE SF 368.

M:DIS (XDBHNDLR, XOPEN2, XUPDAT)
MEMORY VARIABLE WHICH IDENTIFIES THE DISCOVERY CODE.
ORIGINALLY CAPTURED FROM BLOCK 22 OF THE SF 368.

M:DITEM (XDBHNDLR, XOPEN2, XUPDAT)
MEMORY VARIABLE WHICH CONCATENATES MODEL, SERIAL NUMBER OF
DEFICIENT PART, NEXT HIGHER ASSEMBLY, AND SUB ASSEMBLY.
ORIGINALLY CAPTURED FROM BLOCK 16 OF SF 368.

M:DOC (XDBHNDLR, XOPEN2, XUPDAT)
MEMORY VARIABLE WHICH HOLDS THE TYPE DOCUMENT. ORIGINALLY
CAPTURED FROM BLOCK 30 OF THE SF 368.

M:DOCNO (XDBHNDLR, XOPEN2, XUPDAT)
MEMORY VARIABLE WHICH HOLDS THE DOCUMENT NUMBER. ORIGINALLY
CAPTURED FROM BLOCK 10 OF THE SF 368.

M:EPRC (XDBHNDLR, XOPEN2, XUPDAT)
MEMORY VARIABLE WHICH HOLDS THE EXTENDED PRICE OF THE
DEFICIENT MATERIAL. THE EXTENDED PRICE IS CALCULATED BY
MULTIPLYING THE QUANTITY DEFICIENT BY THE UNIT PRICE.

M:FSCM (XDBHNDLR, XOPEN2, XUPDAT)
 MEMORY VARIABLE WHICH HOLDS THE FEDERAL SUPPLY CODE OF MANUFACTURER. ORIGINALLY CAPTURED FROM BLOCK 7 OF THE SF 368.

M:GOV (XDBHNDLR, XOPEN2, XUPDAT)
 MEMORY VARIABLE WHICH IDENTIFIES GOVERNMENT FURNISHED MATERIAL. ORIGINALLY CAPTURED FROM BLOCK 14 OF THE SF 368.

M:ITEM (XDBHNDLR, XOPEN2, XUPDAT)
 MEMORY VARIABLE WHICH IDENTIFIES THAT THE ITEM IS NEW OR A REPAIR/OVERHAUL ITEM. ORIGINALLY CAPTURED FROM BLOCK 11 OF THE SF 368.

M:KEY (CLOSREC, C-REASGN, XDBHNDLR, XOPEN2)
 MEMORY VARIABLE WHICH CONTAINS THE DATABASE ACCESS KEY.

M:LDATE (CLOSREC, XOPEN2)
 MEMORY VARIABLE WHICH HOLDS THE DATE THE CASE WAS TRANSMITTED TO THE ITEM MANAGER. ORIGINALLY ENTERED UPON TRANSMISSION OF THE CASE.

M:LOT (XDBHNDLR, XOPEN2, XUPDAT)
 MEMORY VARIABLE WHICH IDENTIFIES THE MANUFACTURERS LOT NUMBER. ORIGINALLY CAPTURED FROM BLOCK 16B(3) OF SF 368.

M:MFG (XDBHNDLR, XOPEN2, XUPDAT)
 MEMORY VARIABLE WHICH IDENTIFIES THE MANUFACTURERS PART NUMBER. ORIGINALLY CAPTURED FROM BLOCK 16B(3) OF SF 368.

M:NOMEN (XDBHNDLR, XOPEN2, XUPDAT)
 MEMORY VARIABLE WHICH HOLDS THE NOMENCLATURE OF THE MATERIAL BEING REPORTED IN THE QDR. ORIGINALLY CAPTURED FROM BLOCK 6 OF THE SF 368.

M:NSN (C-REASGN, XDBHNDLR, XOPEN2, XUPDAT)
 MEMORY VARIABLE WHICH IDENTIFIES THE NATIONAL STOCK NUMBER. ORIGINALLY CAPTURED FROM BLOCCK 5 OF SF 368.

M:NUM (XOPEN2, XUPDAT)
MEMORY VARIABLE WHICH IDENTIFIES THE CONTRACT NUMBER UNDER WHICH THE REPORTED MATERIAL WAS RECEIVED. ORIGINALLY CAPTURED FROM BLOCK 10 OF THE SF 368.

M:09Q (XDBHNDLR, XOPEN2, XUPDAT)
MEMORY VARIABLE WHICH IDENTIFIES THE GSA REGION CODE FOR 9Q ITEMS.

M:OPEN (CLOSREC, XOPEN2, XUPDAT)
MEMORY VARIABLE WHICH HOLDS THE DATE THE CASE WAS OPENED. ORIGINALLY ENTERED BY THE SYSTEM UPON NEW CASE INPUT.

M:ORG (XDBHNDLR, XOPEN2, XUPDAT)
MEMORY VARIABLE WHICH HOLDS THE ORIGIN CODE.

M:OTF (XDBHNDLR, XOPEN2, XUPDAT)
MEMORY VARIABLE WHICH IDENTIFIES THE OPERATING TIME AT FAILURE. ORIGINALLY CAPTURED FROM BLOCK 13 OF THE SF 368.

M:OVER (XDBHNDLR, XOPEN2, XUPDAT)
MEMORY VARIABLE WHICH IDENTIFIES THE DATE OF MANUFACTURE/OVERHAUL. ORIGINALLY CAPTURED FROM BLOCK 12 OF THE SF 368.

M:QTYDEF (XDBHNDLR, XOPEN2, XUPDAT)
MEMORY VARIABLE WHICH HOLDS THE QUANTITY OF MATERIAL REPORTED AS DEFICIENT. ORIGINALLY CAPTURED FROM BLOCK 15C OF THE SF 368.

M:QTYINS (XDBHNDLR, XOPEN2, XUPDAT)
MEMORY VARIABLE WHICH HOLDS THE QUANTITY OF MATERIAL INSPECTED BY THE REPORTING ACTIVITY. ORIGINALLY CAPTURED FROM ELCK 15B OF THE SF 368.

M:QTYREC (XDBHNDLR, XOPEN2, XUPDAT)
MEMORY VARIABLE WHICH HOLDS THE QUANTITY OF MATERIAL RECEIVED BY THE REPORTING ACTIVITY. ORIGINALLY CAPTURED FROM BLOCK 15A OF THE SF 368.

M:QTYSTK (XDBHNDLR, XOPEN2, XUPDAT)
 MEMORY VARIABLE WHICH HOLDS THE QUANTITY OF MATERIAL IN STOCK AT THE REPORTING ACTIVITY WHEN THE QDR WAS SUBMITTED. ORIGINALLY CAPTURED FROM BLOCK 15D OF THE SF 368.

M:RDATE (CLOSREC, XOPEN2, XUPDAT)
 MEMORY VARIABLE WHICH HOLDS THE DATE RECEIVED FROM ORIGIN. ORIGINALLY CAPTURED FROM THE MAILROOM TIMESTAMP ON RECEIPT DATE.

M:REC1 (CLOSREC, XDBHNDLR, XOPEN2)
 MEMORY VARIABLE WHICH IDENTIFIES THE RECORD NUMBER OF THE RECORD BEING PROCESSED. THIS IS A SYSTEM GENERATED VARIABLE.

M:REOPEN (CLOSREC, XOPEN2, XUPDAT)
 MEMORY VARIABLE WHICH HOLDS THE DATE A CLOSED CASE IS REOPENED. ORIGINALLY ENTERED UPON REOPENING A CLOSED CASE.

M:REPCON (XDBHNDLR, XOPEN2, XUPDAT)
 MEMORY VARIABLE WHICH HOLDS THE REPORT CONTROL NUMBER. ORIGINALLY CAPTURED FROM BLOCK 3 OF THE SF 368.

M:REPLY (XDBHNDLR, XUPDAT)
 MEMORY VARIABLE WHICH IDENTIFIES THE REPLY RECEIVED FROM THE ITEM MANAGER. ORIGINALLY CAPTURED FROM BLOCK 32 OF THE SF 368.

M:RETC (XDBHNDLR, XUPDAT)
 MEMORY VARIABLE WHICH IDENTIFIES THE RETURN CODE OF THE QDR.

M:RIMDATE (CLOSREC, XUPDAT)
 MEMORY VARIABLE WHICH HOLDS THE DATE RETURNED FROM THE ITEM MANAGER. ORIGINALLY ENTERED UPON RECEIPT OF A RESPONSE FROM THE ITEM MANAGER.

M:SCR (XDBHNDLR, XUPDAT)
 MEMORY VARIABLE WHICH IDENTIFIES THE SCREENING CODE.

M:SCFQTY (XDBHNDLB, XUPDAT)
MEMORY VARIABLE WHICH IDENTIFIES THE SCREENING QUANTITY.

M:SM (XDBHNDLB, XOPEN2, XUPDAT)
MEMORY VARIABLE WHICH IDENTIFIES THE SUPPLY MANAGEMENT INFORMATION CODE. ORIGINALLY CAPTURED FROM BLOCK 5 OF THE SF 368.

M:STATUSC (XDBHNDLB, XUPDAT)
MEMORY VARIABLE WHICH IDENTIFIES THE STATUS CODE.

M:TIME (LOGON, XDBHNDLR, XOPEN2)
MEMORY VARIABLE WHICH HOLDS THE TIMESTAMP. THIS IS A SYSTEM VARIABLE USED TO LOCK INDIVIDUAL RECORDS.

M:TYPE (CLOSREC, C-REASGN, XDBHNDLB, XOPEN2, XUPDAT)
MEMORY VARIABLE WHICH HOLDS THE CODE SPECIFYING THE DATABASE HANDLER ACCESS CODE.

M:UI (XDBHNDLB, XOPEN2, XUPDAT)
MEMORY VARIABLE WHICH HOLDS THE UNIT OF ISSUE FOR THE REPORTED MATERIAL. ORIGINALLY CAPTURED FROM THE ML-N BASED ON THE NSN BEING REPORTED.

M:UIC (XDBHNDLR, XOPEN2, XUPDAT)
MEMORY VARIABLE WHICH HOLDS THE UNIT IDENTIFICATION CODE OF THE ACTIVITY SUBMITTING THE QDR. ORIGINALLY CAPTURED FROM BLOCK 1A OF THE SF 368.

M:UPFC (XDBHNDLB, XOPEN2, XUPDAT)
MEMORY VARIABLE WHICH HOLDS THE UNIT PRICE FOR THE REPORTED MATERIAL. ORIGINALLY CAPTURED FROM THE ML-N BASED ON THE REPORTED NSN.

M:VLC (CLOSREC, XDBHNDLR, XUPDAT)
MEMORY VARIABLE WHICH IDENTIFIES THE VENDOR LIABILITY CODE.

M:WHO

(C-REASGN, XDBHNDLR)

MEMORY VARIABLE WHICH IDENTIFIES THE INDIVIDUAL CREATING THE RECORD. THIS IS A SYSTEM VARIABLE WHICH IS CAPTURED FROM THE SYSTEM LOGON.

M:WNTY

(XOPEN2, XUPDAT, XDBHNDLR)

MEMORY VARIABLE WHICH IDENTIFIES THE WARRANTY STATUS OF THE ITEM. ORIGINALLY CAPTURED FROM BLOCK 19 OF SF 368.

M:WUC

(XDBHNDLR, XOPEN2, XUPDAT)

MEMORY VARIABLE WHICH IDENTIFIES THE WORK UNIT CODE. ORIGINALLY CAPTURED FROM BLOCK 20 OF THE SF 368.

APPENDIX B
QDR DATABASE FILE STRUCTURES

STRUCTURE FOR FILE: D:OPEN1 .DBF

FLD	NAME	TYPE	WIDTH	DEC
001	CASE	C	007	
002	CCG	C	002	
003	NSN	C	013	
004	CAT	C	001	
005	NCMEN	C	019	
006	UIC	C	006	
007	UI	C	002	
008	QTYDEF	N	006	
009	UPRC	N	009	002
010	EPRC	N	012	002
011	ORG	C	003	
012	DCC	C	001	
013	DOCNO	C	014	
014	DATES	C	046	
015	REPCON	C	012	
016	FSCM	C	006	
017	TIME	C	011	
018	WHO	C	004	
019	NUM	C	017	
020	CR	C	001	
021	SCR	C	003	
022	SM	C	002	
023	O9Q	C	001	
024	DEF	C	002	
025	VLC	C	001	
026	ACTPT	C	011	
027	SCRQTY	N	006	

** TOTAL ** 00219 BYTES

STRUCTURE FOR FILE: D:OPEN2 .DBF

FLD	NAME	TYPE	WIDTH	DEC
001	CASE	C	007	
002	QTYINS	N	006	
003	QTYREC	N	006	
004	QTYSTK	N	006	
005	DEFV	C	001	
006	DEFR	C	001	
007	ITEM	C	001	
008	OVER	C	005	
009	OIF	C	005	
010	GCV	C	001	
011	TIME	C	011	
012	WHO	C	004	
013	DITEM	C	035	
014	CCOST	N	012	002
015	WNTY	C	001	
016	WUC	C	007	
017	DIS	C	002	

018	DETAILS	C	120
019	REPLY	C	120
020	ACTTKN	C	003
021	COSTC	C	001
022	STATUSC	C	002
023	CAUSEC	C	001
024	RETC	C	001
025	ACTDISP	C	001
026	MFG	C	016
027	LOT	C	009

** TOTAL ** 00386 BYTES

STRUCTURE FOR FILE: D:CLOSE1 .DBF

FLD	NAME	TYPE	WIDTH	DEC
001	CASE	C	007	
002	COG	C	002	
003	NSN	C	013	
004	CAT	C	001	
005	NCMEN	C	019	
006	UIC	C	006	
007	UI	C	002	
008	QTYDEF	N	006	
009	UURFC	N	009	002
010	HURFC	N	012	002
011	ORG	C	003	
012	DOC	C	001	
013	DCCNO	C	014	
014	DATES	C	046	
015	WHPCON	C	012	
016	WSCM	C	006	
017	TIME	C	011	
018	WHO	C	004	
019	NUM	C	017	
020	CR	C	001	
021	SCR	C	003	
022	SM	C	002	
023	OOO	C	001	
024	DEF	C	002	
025	VIC	C	001	
026	ACTP	C	011	
027	SCRQTY	N	006	

** TOTAL ** 00219 BYTES

STRUCTURE FOR FILE: I:CLOSE2 .DBF

FLD	NAME	TYPE	WIDTH	DEC
001	CASE	C	007	
002	CITYINS	N	006	
003	QTYREC	N	006	
004	QTYSTK	N	006	
005	DEPV	C	001	
006	DEFR	C	001	
007	ITEM	C	001	
008	OVER	C	005	
009	CTE	C	005	
010	GOV	C	001	
011	TIME	C	011	

012	WHO	C	004	
013	DITEM	C	035	
014	CCOST	N	012	002
015	WNTY	C	001	
016	WUC	C	007	
017	DIS	C	002	
018	DETAILS	C	120	
019	REPLY	C	120	
020	ACTTKN	C	003	
021	CCSTC	C	001	
022	STATUSC	C	002	
023	CAUSEC	C	001	
024	RETC	C	001	
025	ACTDISP	C	001	
026	MFG	C	016	
027	LCT	C	009	

** TOTAL ** 00386 BYTES

STRUCTURE FCR FILE: D:COG .DBF

FLD	NAME	TYPE	WIDTH	DEC
001	COG	C	002	
002	IM	C	007	
003	CCUN	N	004	

** TOTAL ** 00014 BYTES

STRUCTURE FCR FILE: D:TECHCODE.DBF

FLD	NAME	TYPE	WIDTH	DEC
001	NAME	C	020	
002	TECHCODE	C	004	
003	PSWD	C	008	
004	LCGGED	C	001	
005	ASSIGNED	N	004	
006	ACTIVE	N	004	
007	TRANSMIT	N	004	
008	RESPOND	N	004	
009	CLOSED	N	004	

** TOTAL ** 00054 BYTES

STRUCTURE FCR FILE: D:ADDRESS .DBF

FLD	NAME	TYPE	WIDTH	DEC
001	IM	C	007	
002	TITLE	C	018	
003	CCMMAND	C	040	
004	CCMMAND2	C	040	
005	ATTN	C	015	
006	STREET	C	020	
007	CITY	C	020	
008	STATE	C	002	
009	ZIP	C	005	
010	CCUNT	N	004	

** TOTAL ** 00172 BYTES

STRUCTURE FCR FILE: D:WHERDIS .DBF

FLD	NAME	TYPE	WIDTH	DEC
001	CCDE	C	002	
002	TEXT	C	020	

** TOTAL ** 00023 BYTES

STRUCTURE FCR FILE: D:FILESTAT.DBF

FLD	NAME	TYPE	WIDTH	DEC
001	OPEN1	C	004	
002	OPEN2	C	004	
003	CLOSE1	C	004	
004	CLOSE2	C	004	

** TOTAL ** 00017 BYTES

STRUCTURE FCR FILE: D:BIWKSTAT.DBF

FLD	NAME	TYPE	WIDTH	DEC
001	YEAR	C	004	
002	TOTALS	N	005	
003	LAST	C	005	

** TOTAL ** 00015 BYTES

APPENDIX C

DATA ELEMENT DEFINITIONS

This provides a list of the Data Element pictures that are used in various QDR programs. The column labeled "Data Element" contains a short description of the actual Element that the Variable represents. The column "ID" contains the variable name associated with the Data Element. The ID is further defined in Appendix A. The "Type" is either character "C" or numeric "N", with a length as shown. The "Picture" shows the size and character type of each Data Element. Standard representations are used:

- 'A' - Alphabetic
- '9' - Numeric, 0-9
- 'X' - Either numeric or alphabetic

<u>Data Element</u>	<u>ID</u>	<u>Picture</u>	<u>Type</u>
Case Number	M:CASE	'999999A'	A7
Cognizance	M:CCG	'XX'	A2
Category	M:CAT	'9'	A1
Nomenclature	M:NCMEN	'XXX...XX'	A19
UIC	M:UIC	'AXXXXX'	A6
Unit of issue	M:UI	'AA'	A2
Unit Price	M:UFPC	'999999.99'	N9
Quantity Def.	M:QTYDEF	'9999999'	N6
Quantity Insp.	M:QTYINS	'9999999'	N6
Quantity Recvd	M:QTYREC	'9999999'	N6
Qty in Stock	M:QTYSTK	'9999999'	N6
Extended Price	M:EFPC	'999999999.99'	N12
Origin	M:ORIG	'XXX'	A3
Deficiency Ver	M:DEFV	'A'	A1
Deficiency Resp	M:DEFR	'A'	A1
Type Document	M:DCC	'9'	A1
Discovery Date	M:DDATE	'999999'	A5
Date Rcvd fm Org	M:RDATE	'999999'	A5
Open Date	M:OPEN	'999999'	A5
Date Ltr Typed	M:LCATE	'999999'	A5
Screen Rpt Date	M:SCFDAT	'999999'	A5
Interim Resp Dat	M:IRDATE	'999999'	A5
Date rtn fm IM	M:RIMDAT	'999999'	A5
Close Date	M:CLCSE	'999999'	A5
Reopen Date	M:RECPEN	'999999'	A5
Dates concat'ned	M:DATES	See note 1	A46
Date change ID	M:DATECI	'X' See note 2	A1
Report Control #	M:REFCON	See note 3	A12
Document Number	M:DCCNO	See note 4	A14
FSCM	M:FSCM	'XXXXXX'	A6
Time Stamp	M:TIME	'999999999999'	A11
Analyst Code	M:WHC	'XXXX'	A4
New-Repair/Ovhl	M:ITEM	'A'	A1
Date Mfg/Ovhl	M:OVER	'999999'	A5
Opn Time-Failure	M:OTF	'A99999'	A5
Govnment Furnish	M:GCV	'X'	A1
Work Unit Code	M:WUC	'XXXXXXXX'	A7

Discovery Code	M:DIS	'AA'	A2
Details Section	M:DETAILS	'XX...198..X'	A198
Return Code	M:RETC	'9'	A1
Record Variable	M:REC1	'99999A'	A6
Contract Number	M:NUM	See note 5	A17
Credit Code	M:CR	'A'	A1
Screening Code	M:SCF	'XXX'	A3
Reply Section	M:REPLY	'XX..198..X'	A198
Action Code	M:ACITKN	'AAA'	A3
Cost Code	M:CCSTC	'A'	A1
Status Code	M:STATUSC	'AA'	A2
Cause Code	M:CAUSEC	'A'	A1
Action Disp'n	M:ACIDISP	'A'	A1
SMIC	M:SM	'AX'	A2
9Q Region Code	M:O9C	'X'	A1
Type Defect	M:DEF	'99'	A2
Vendor Liab Code	M:VIC	'A'	A1
Action Point	M:ACTPT	'AXXXX99999'	A11
Part Number	M:MFG	'XX..16..XX'	A16
Lot/ser/batch	M:LCT	'XXXXXXXXXX'	A9
NSN	M:NSN	See note 6	A13
Type/Model/Ser	O:MCDEL	'XXXXXXXX'	A7
Def Item Ser #	O:DEFSER	'XXXXXX'	A6
Higher Assy	O:HASSY	'XXXXXXXXXXX'	A10
Sub assembly	O:SASSY	'XXXXXXXXXXXXX'	A12
Def Item	M:DITEM	'XXXX 35 XXXX'	A35
Est Crrr Cost	M:CCCST	'999999999.99'	N12
Warranty	M:WNTY	'A'	A1
Screen Quantity	M:SCFQTY	'999999'	N6

NOTES:

1. All dates, followed by M:DATECI, are concatenated into the variable M:DATES for storage into the databases. This is necessary to minimize the number of variables active in the CDR programs, and due to the 32 field limit per database.
2. This variable is the last field in M:DATES. Values are either a blank, "N", or "*". "N" depicts a newly formed case that has not been accounted for in the statistics. A "*" shows that a date was changed during a case update. These are blanked after statistics are calculated.
3. Report Control Number (RCN) 'XXXXXX-99-9999'
4. M:DOCNO PICTURE 'XXXXXX-9999-9999'
5. Contract number picture 'XXXXXX-99-A-9999-9999'
6. NSN (FSC+NATO+FIIN) '9999-XX-XXX-9999'

APPENDIX D
QDR PROGRAMS

I.	LOGCN MODULE	59
II.	LOCKOUT	62
III.	MAIN PROCESSING MODULE	63
IV.	NEW CASE INPUT MODULE	66
V.	CASE UPDATE MODULE	82
VI.	CASE CLOSING MCDULE	101
VII.	DATA BASE HANDLER MODULE	106
VIII.	SUPERVISOR MENU	127
IX.	SUPERVISOR UTILITY MENU	129
X.	USER REPORT MENU	132
XI.	SUPERVISOR REPCRT MENU	134
XII.	QUERY MODULE	138
XIII.	STATISTICS GENERATION MCDULE	158
XIV.	JULIAN DATE CCNVERSION MODULE	163
XV.	CCG COUNT MODULE	164
XVI.	BI-WEEKLY STATIISTICS REPORT MODULE	166
XVII.	MCNTHLY STATISIICS REPORT MODULE	172
XVIII.	SORTED LISTING REPORT MENU	182

XIX.	CASE REASSIGNMENT MODULE	190
XX.	ITEM MANAGER FILE UPDATE	192
XXI.	CCG FILE UPDATE MODULE	197
XXII.	DATA BASE PACK MODULE	203
XXIII.	ANALYST FILE UPDATE MODULE	205
XXIV.	PASSWORD FILE UPDATE MODULE	210
XXV.	DATA BASE RE-INDEX MODULE	212
XXVI.	OPEN CASE REPCRT	214

I. LOGON MODULE

```
*****  
**  
** Date: 23 Nov 1983  
** Version: 1.0  
** Module Name: LOGCN  
** Module Purpose: Provide Password Logon Facilities  
**                   for the QDR System  
**  
** Module Interface Definition  
**   Inputs: None  
**   Outputs: C:JULIAN, C:WHO  
**  
** Module Processing Narrative Description:  
**   Accepts The Password From The Operator, Val-  
**   idates The Password, and Calls The Necessary  
**   Modules  
**  
** Superordinate Modules: None  
** Subordinate Modules: SUPMENU1, MENU1, LOCKOUT  
** Author: R. G. NICHOLS  
**  
*****
```

```
SET TALK OFF  
SET BELL CFF  
SET COLOR TO 112,3  
SET EXACT ON  
SET COLON OFF  
STORE T TO V:CONTINUE
```

***** Accept ID of Person Logging On To The System

```
DO WHILE V:CONTINUE  
  ERASE  
  STORE ' ' TO C:WHO  
  @ 10,29 SAY 'ENTER YOUR ACCESS I.D.'  
  @ 11,37 GET C:WHO  
  READ  
  STORE !(C:WHO) TO C:WHO  
  IF C:WHO = 'QUIT'  
    QUIT  
  ENDF
```

***** Validate ID To See If A Valid User Is Logging On

```
USE D:TECHCODE INDEX D:TECH  
FIND &C:WHO  
DO WHILE # = 0  
  @ 13,28 SAY 'ACCESS I.D. NOT ON FILE'  
  @ 14,33 SAY 'PLEASE REENTER'  
  STORE ' ' TC C:WHO  
  @ 11,37 GET C:WHO  
  READ  
  STORE !(C:WHO) TO C:WHO  
  IF C:WHO = 'QUIT'  
    QUIT  
  ENDF  
FIND &C:WHO  
ENDDO  
STORE F TO V:LOGGED
```


***** Check To See If Previously Logged On

```
IF LOGGED <> ' '  
  @ 16,28 SAY 'USER CURRENTLY LOGGED ON'  
  @ 17,32 SAY 'LOGON TERMINATED'  
  @ 23,0 SAY ' '  
  STCRE T TO V:LOGGED  
ENDIF
```

***** Allow Three Attempts to Enter The Correct Password

```
IF .NCT. V:LOGGED  
  STORE 2 TO V:ATTEMPTS  
  @ 16,30 SAY 'ENTER YOUR PASSWORD'  
  @ 17,30 SAY ' FOLLOWED BY <CR>'  
  SET CCNSOLE OFF  
  @ 19,35 SAY ' '  
  STCRE T TO V:TRUE  
  DO WHILE V:TRUE  
    STCRE ' ' TC V:PSWD  
    ACCEPT TO V:PSWD  
    IF V:ATTEMPTS = 0 .AND. PSWD <> V:PSWD  
      SET CONSCIE ON
```

***** If Three Unsuccessful Passwords Are Entered, Call
***** For System Lockup Program

```
    DO C:LOCKCUT  
  ELSE  
    IF PSWD <> V:PSWD  
      STORE V:ATTEMPTS-1 TO V:ATTEMPTS  
      @ 21,27 SAY 'INCORRECT PASSWORD ENTERED'  
      @ 19,35 SAY ' '+ CHR(7)  
    ELSE  
      STORE F TO V:TRUE  
    ENDIF  
  ENDIF  
ENDIF  
ENDDO  
SET CCNSOLE ON  
REPL LOGGED WITH '*'  
USE  
SET EXACT OFF  
RELEASE ALL LIKE V:*
```

***** If Either Supervisor is Logging On The System Call
***** For Supervisor Menu To Be Displayed Otherwise
***** Display General User Menu

```
IF C:WHO ='0000' .OR. C:WHO='0001'  
  DO C:SUPMENU1  
ELSE  
  DO C:MENU1  
ENDIF
```

```
STORE T TO V:CCONTINUE
```

***** Allow The Logged User To Logoff

```
USE D:TECHCODE INDEX D:TECH  
FIND &C:WHC  
IF # = 0  
  ERASE  
  @ 10,32 SAY 'LOG OFF FAILURE'+CHR(7)  
  @ 11,27 SAY 'CONTACT SYSTEM SUPERVISOR'  
  @ 23,26 SAY 'STRIKE ANY KEY TO CONTINUE '  
  WAIT  
ELSE  
  REPL LOGGED WITH ' '  
  USE
```


***** Clear The Screen, Read The Clock And Display The
***** Logged Off Message and The Time

```
ERASE
STORE " " TC V:DUMMY
PCKE 61440, 180, 44, 205, 33, 137, 22, 13, 240,;
      137, 14, 15, 240, 195
SET CALL TO 61440
CALL V:DUMMY
STORE STR(PEEK(61456),2) TO V:HOURL
STORE STR(PEEK(61455),2) TO V:MIN
STORE STR(PEEK(61454),2) TO V:SEC
IF $(V:HOURL,1,1)=" "
  STORE "0" +$(V:HOURL,2,1) TO V:HOURL
ENDIF
IF $(V:MIN,1,1)=" "
  STORE "0" +$(V:MIN,2,1) TO V:MIN
ENDIF
IF $(V:SEC,1,1)=" "
  STORE "0" +$(V:SEC,2,1) TO V:SEC
ENDIF
STORE V:HOURL +":" +V:MIN +":" +V:SEC TO M:TIME
@ 8,30 SAY 'LOG OFF COMPLETED AT'
@ 10,36 SAY M:TIME
RELEASE ALL
STORE T TO V:CONTINUE
ENDIF
ENDIF
@ 22,28 SAY 'PRESS ANY KEY TO CONTINUE '
WAIT
USE
ENDDC
***** NOTE: This Program Will Continue To Accept Logons
***** Until The Quit Command Is Entered
***** END OF PROGRAM
```


II. LOCKOUT

```
*****  
**  
** Date: 23 Nov 1983  
** Version: 1.0  
** Module Name: LOCKCUT  
** Module Purpose: To Lock The System After An Illegal  
** Logon Attempt  
**  
** Module Interface Definition  
** Inputs: None  
** Outputs: None  
**  
** Module Processing Narrative Description:  
** This Program Will Display An Illegal Logon  
** Message and Will Sound The Buzzer  
**  
** Superordinate Modules: LOGON  
** Subordinate Modules: None  
** Author: R. G. NICHOLS  
**  
*****
```

```
SET TALK OFF  
SET CCICR TC 4,4  
STORE T TO V:CONTINUE  
STORE T TO V:TOGGLE
```

```
***** Clear The Screen and Display The Illegal Access  
***** Message
```

```
ERASE  
DO WHILE V:CONTINUE  
STORE 5 TO V:INNER  
DO WHILE V:INNER > 0  
IF V:TOGGLE  
@ 10,29 SAY 'ILLEGAL ACCESS ATTEMPT'  
@ 21, 0 SAY ' ' + CHR(7)  
ELSE  
@ 10,29 SAY ' '  
@ 21, 0 SAY ' '  
ENDIF  
STORE V:INNER-1 TO V:INNER  
ENDDO  
IF V:TOGGLE  
STORE F TO V:TOGGLE  
ELSE  
STORE T TO V:TOGGLE  
ENDIF  
ENDDO
```

```
***** NOTE: The System Must Be Rebooted To Exit From  
***** An Illegal Access Attempt
```

```
***** END OF PROGRAM
```


III. MAIN PROCESSING MODULE

```
*****
**
** DATE: 15 NOVEMBER 1983
** VERSION: 1.0
** MODULE NAME: MENU1
** MODULE PURPOSE: PROVIDE THE USER A MENU OF ALL
** PROCESSING OPTIONS AVAILABLE TO HIM/HER IN
** THE QDR SYSTEM.
** MODULE INTERFACE DEFINITION
** INPUTS: C:WHO
** OUTPUTS: C:JULIAN
** MODULE PROCESSING NARRATIVE DESCRIPTION:
**
** DISPLAYS ALL PROCESSING OPTIONS AVAILABLE TO
** THE USER. UPON USER SELECTION, CALLS THE
** APPROPRIATE MODULE FOR CONTINUED PROCESSING
** ALLOWS USER TO EXIT FROM THE QDR SYSTEM.
**
** SUPERORDINATE MODULES: LOGCN
** SUBORDINATE MODULES: OPEN,CLOSE,UPDAT,LTR,RPTMENU,
** QUERY
** AUTHOR: J.G. BOYNTON & R.G. NICHOLS
**
*****
```

***** THIS SECTION ACCESSES THE SYSTEM DATE

```
STORE TO V:DUMMY
POKE 61440, 180, 42, 205, 33, 137, 22, 13, 240, 137, ;
14, 15, 240, 195
SET CALL TO 61440
CALL V:DUMMY
STORE PEEK(61454) TO V:MM
STORE PEEK(61453) TO V:DD
STORE PEEK(61456)*256+PEEK(61455)-1900 TO V:YY
```

***** THIS SECTION CONVERTS THE SYSTEM DATE TO A JULIAN DATE

```
DO CASE
CASE V:MM = 01
STCRE V:DD TO V:DAY
CASE V:MM = 02
STCRE V:DD + 31 TO V:DAY
CASE V:MM = 03
STCRE V:DD + 59 TO V:DAY
CASE V:MM = 04
STCRE V:DD + 90 TO V:DAY
CASE V:MM = 05
STCRE V:DD + 120 TO V:DAY
CASE V:MM = 06
STCRE V:DD + 151 TO V:DAY
CASE V:MM = 07
STCRE V:DD + 181 TO V:DAY
CASE V:MM = 08
STCRE V:DD + 212 TO V:DAY
CASE V:MM = 09
STCRE V:DD + 243 TO V:DAY
CASE V:MM = 10
STCRE V:DD + 273 TO V:DAY
CASE V:MM = 11
STCRE V:DD + 304 TO V:DAY
```



```

CASE V:MM = 12
  STCRE V:DD + 334 TO V:DAY
ENDCASE
IF INT(V:YY/4)*4 = V:YY .AND. V:DAY >= 60
  IF V:MM= 02 .AND. V:DD= 29
    STOF V:DAY TO V:DAY
  ELSE
    STOF V:DAY + 1 TO V:DAY
  ENDIF
ENDIF
STORE V:YY * 1000 + V:DAY TO V:JULIAN
STORE STR(V:JULIAN,5) TO C:JULIAN
RELEASE ALL EXCEPT C:*

STORE T TO V:CONTINUE
DO WHILE V:CONTINUE
ERASE
SET TALK OFF
STORE ' ' TO V:CHOICE
TEXT

```

WEELCOME TO THE QDR AUTOMATED TRACKING SYSTEM

- 1 - OPEN NEW RECORD
- 2 - CLOSE RECORD
- 3 - UPDATE RECORD
- 4 - CRIGINATE LETTER
- 5 - REPORT GENERATION
- 6 - QUERY
- 7 - EXIT FROM THE SYSTEM

ENTER YOUR CHOICE

```

ENDTEXT
@ 19,30 GET V:CHOICE
READ
?
IF V:CHOICE >= 1 .AND. V:CHOICE <= 7
?
DO CASE
  CASE V:CHOICE= 1
    RELEASE ALL EXCEPT C:*
    DO C:XCPEN2.PRG
  CASE V:CHOICE= 2
    RELEASE ALL EXCEPT C:*
    DO C:CICSREC.PRG
  CASE V:CHOICE= 3
    RELEASE ALL EXCEPT C:*
    DO C:XUPDAT.PRG
  CASE V:CHOICE= 4
    RELEASE ALL EXCEPT C:*
    DO C:LTR.PRG
  CASE V:CHOICE= 5
    RELEASE ALL EXCEPT C:*
    DO C:RFIMENU
  CASE V:CHOICE=6
    RELEASE ALL EXCEPT C:*
    DO C:QUERY.PRG
  CASE V:CHOICE=7
    RELEASE ALL EXCEPT C:*
    RETURN
ENDCASE

```



```
        STORE T TO V:CONTINUE
        STORE ' ' TO V:CHOICE
ELSE
  @ 21,20 SAY ' < PLEASE ANSWER WITH 1 - 7 ONLY >'
  @ 23,20 SAY '<PRESS ANY KEY TO CONTINUE>'
  WAIT
ENDIF <V:CHCICE>
ENDDO <V:CONTINUE>
***** END OF PROGRAM
```


IV. NEW CASE INPUT MODULE

```
*****
**
** DATE: 18 NOV 1983
** VERSICN: 1.0
** MODULE NAME: OPEN
** MODULE PURPOSE: NEW QDR CASE CREATION
** MODULE INTERFACE DEFINITION
** INPUTS: C:WHO, C:JULIAN, V:JULDATE
** OUTPUTS: M:CASE, V:MM, V:DD, V:YY, AND ALL OF THE
** DATA ELEMENTS IN OPEN1 AND OPEN2.
** MODULE PROCESSING NARRATIVE DESCRIPTION:
**
** PROMPTS THE USER FOR INPUT OF ALL DATA FROM
** SF 368 IN CRDR TO CREATE A NEW QDR CASE.
** VALIDATION OF DATA ITEMS OCCURS UPON INPUT AND
** IS BASED UPON CURRENT GE TIMESHARE VALIDATION,
** AS MODIFIED BY FMSO TECHNICAL BRANCH. DATES
** ARE CAPTURED FOR MANAGEMENT STATISTICS.
**
** SUPERORDINATE MODULES: MENU1
** SUBORDINATE MODULES: OJULIAN, XDBHNDLR
** AUTHCR: J.G. BOYNTON
**
*****
```

```
STORE T TO C:TRUE
DO WHILE O:TRUE
ERASE
STORE ' ' TO O:CHOICE
TEXT
```

***** OPEN NEW CASE *****

THIS PROGRAM ENABLES YOU TO OPEN A
NEW QDR CASE

1 - CONTINUE

2 - RETURN TO MENU

```
ENDTEXT
@ 20,30 SAY ' ' GET O:CHOICE
READ
DO WHILE C:CHOICE <> '1' .AND. O:CHOICE <> '2'
@ 23,20 SAY 'ANSWER WITH A 1 OR 2 ONLY'
@ 20,30 SAY ' ' GET O:CHOICE
READ
ENDDO
ERASE
IF C:CHOICE = '2'
RELEASE ALL EXCEPT C:*
RETURN
ENDIF
```

***** INITIALIZE MEMORY VARIABLES

```
STORE ' ' TO O:KEY
```



```

STORE ' ' TO M:CASE
STORE ' ' TO M:COG
STORE ' ' TO M:SM
STORE ' ' TO M:CAT
STORE ' ' TO M:NOMEN
STORE ' ' TC M:UIC
STORE 0 TO M:QTYREC
STORE 0 TO M:QTYINS
STORE 0 TO M:QTYDEF
STORE 0 TO M:QTYSTK
STORE ' ' TO M:NUM
STORE ' ' TC M:OPEN
STORE ' ' TC M:OVER
STORE ' ' TO M:REPCON
STORE ' ' TC M:FSCM
STORE ' ' TO M:TIME
STORE ' ' TO M:MFG
STORE ' ' TO M:LOT
STORE ' ' TO M:ITEM
STORE ' ' TC M:OTF
STORE ' ' TC M:GCV
STORE ' ' TO M:DOCNO
STORE ' ' TO M:DEF
STORE ' ' TO M:WNTY
STORE ' ' TC O:DDATE
STORE ' ' TO O:MODEL
STORE ' ' TC O:DEFSER
STORE ' ' TO O:HASSY
STORE ' ' TO O:SASSY

```

```

***** THIS SEQUENCE CALCULATES THE UPPER AND LOWER YEARS
***** FOR INPUT AND IS BASED ON THE CURRENT JULIAN DATE
***** C:JULIAN. O:LLIMIT= YEAR MINUS TWO YEARS
***** O:ULIMIT = YEAR PLUS ONE YEAR

```

```

STORE $(C:JULIAN,1,2) TO TEMP1
STORE VAI(TEMP1) TO TEMP1A
STORE VAI('2') TO LOW
STORE VAI('1') TO HIGH
STORE TEMP1A-LOW TO LLMT
STORE TEMP1A+HIGH TO ULMT
STORE STR(LLMT,2) TO C:LLIMIT
STORE STR(ULMT,2) TO C:ULIMIT
RELEASE TEMP1,TEMP1A,LOW,HIGH,LLMT,ULMT

```

```

***** START OF THE INPUT FOR THE NSN

```

```

STORE ' ' TC O:FLATE
STORE T TO O:ANSWER
DO WHILE O:ANSWER
@ 5,20 SAY '***** ENTER DATA FOR THE NEW CASE *****'
@ 6,20 SAY '***** FROM SF 368 *****'

STORE T TO O:RDATE
LO WHILE O:RDATE
@ 8,20 SAY 'DATE RECEIVED BY FMSO MMDDYY' ;
GET O:RDATE PICTURE '999999'

READ
IF $(O:RDATE,1,2) <'01' .OR. $(O:RDATE,1,2) >'12';
.OR. $(O:RDATE,3,2) <'01' .OR. $(O:RDATE,3,2) >'31';
.OR. $(O:RDATE,5,2) < O:LLIMIT ;
.OR. $(O:RDATE,5,2) > C:ULIMIT
@ 23,30 SAY 'DATE OUT OF RANGE'
ELSE
STORE F TO O:RDATE
ENDIF
ENDDO <O:RDATE>

```


@ 23,30 SAY ' '

***** ENTER THE CALL TC C:OJULIAN TO CHANGE MMDDYY TO
***** JULIAN FORMAT, STORE TO M:RDATE THEN RELEASE O:RDATE

```
STORE VAL($ (O:RDATE,1,2)) TC V:MM
STORE VAL($ (O:RDATE,3,2)) TO V:DD
STORE VAL($ (O:RDATE,5,2)) TO V:YY
DO C:CJULIAN
STORE V:JULDATE TO M:RDATE
RELEASE ALL LIKE V:*
RELEASE C:RDATE
STORE T TO C:CAT
DO WHILE O:CAT
  @ 10,20 SAY 'CAT'
  @ 10,50 SAY ' ' GET M:CAT PICTURE '9'
  READ
  IF M:CAT = '1' .OR. M:CAT = '2'
    STORE F TO O:CAT
  ELSE
    @ 23,20 SAY ' 1 OR 2 ONLY'
  ENDIF
ENDDO O:CAT
@ 23,20 SAY ' '
RELEASE C:CAT
```

```
STORE T TO C:COG1
STORE T TO C:COG2
DO WHILE O:COG1 .OR. C:COG2
  DC WHILE O:COG1
    @ 12,20 SAY 'COG'
    @ 12,50 SAY ' ' GET M:COG PICTURE '9X'
    READ
    IF $(M:COG,2,1) = ' '
      @ 23,20 SAY ' NO BLANKS IN 2D POSITION'
    ELSE
      STORE F TO C:COG1
      STORE ! (M:CCG) TO M:COG
    ENDIF
  ENDDO <O:COG1>
  @ 23,20 SAY ' '
```

***** CHECKS THAT COG IS VALID IN CURRENT COG TABLE... MUST
***** BE VALID TO CONTINUE

```
USE D:COG INDEX D:COGS
FIND &M:COG
IF # = 0
  @ 23,10 SAY ' COG INVALID - ENTER CORRECTED ENTRY'
  STORE T TO C:COG1
ELSE
  STORE F TO C:COG2
ENDIF
ENDDO <O:COG1 & O:CCG2>
RELEASE O:COG1, O:CCG2
@ 23,10 SAY ' '
```

```
@ 14,20 SAY 'NSN'
@ 14,50 SAY ' ' GET O:KEY PICTURE '9999-XX-XXX-9999'
READ
```

```
STORE T TO O:NATOT
DO WHILE C:NATOT
  IF $(O:KEY,5,1) = ' ' .OR. $(O:KEY,6,1) = ' '
    @ 23,20 SAY ' NATO CODE MAY NOT HAVE BLANKS'
    @ 14,50 SAY ' ' GET O:KEY PICTURE '9999-XX-XXX-9999'
  READ
  ELSE
```



```

        @ 23,20 SAY '
        STORE F TO O:NATOT
    ENDDC <O:NATOT>
    RELEASE C:NATOT
    @ 23,20 SAY '

STORE T TO C:FIINT
DO WHILE O:FIINT
    IF $(O:KEY,7,1) = ' ' .OR. $(O:KEY,8,1) = ' ';
        .CR. $(O:KEY,9,1) = ' '
        @ 23,40 SAY 'NO BLANKS IN THE FIRST 3 POSITIONS'
        @ 14,50 SAY ' GET O:KEY PICTURE '9999-XX-XXX-9999'
        READ
    ELSE
        @ 23,40 SAY '
        STORE F TO O:FIINT
    ENDDC <C:FIINT>
    RELEASE O:FIINT
    @ 23,20 SAY '

STORE ' ' TO O:REPLY
@ 18,20 SAY ' VERIFY ABOVE INFORMATION '
@ 19,20 SAY ' YOU MAY NOT CHANGE IT AFTER THIS'
@ 20,20 SAY ' WITHOUT STARTING OVER AGAIN'
@ 22,25 SAY ' 1 - CONTINUE 2 - CHANGE 3 - EXIT'
@ 23,40 SAY ' GET C:REPLY
READ
IF O:REPLY = '1'
    STORE F TO O:ANSWER
    ERASE
    @ 23,20 SAY 'SEARCHING FOR ANOTHER CASE WITH THIS NSN'
    RELEASE O:RDATE
ELSE
    IF O:REPLY = '3'
        RELEASE ALL EXCEPT C:*
        RETURN
    ELSE
        CLEAR GETS
        @ 22,25 SAY '
        @ 23,25 SAY '
    ENDDC O:ANSWER
ENDIF
ENDIF
ENDDC O:ANSWER

STORE $(O:KEY,1,4) + $(O:KEY,6,2) + $(O:KEY,9,3);
+ $(O:KEY,13,4) TO M:KEY
STORE M:KEY TO M:NSN

***** M:TYPE CODES TELL THE DBHANDLER WHAT TO DO WITH
***** THE PARAMETERS

STORE '1A' TO M:TYPE
DO C:XLBHNDIR.PRG

***** CONTROL RETURNS TO THIS PROGRAM NOW
***** IF M:TYPE = 9 THEN THERE IS NOT A CURRENTLY
***** OPEN FILE

IF M:TYPE='9'
    STORE T TO O:ONONE
ELSE
    STORE F TO O:ONONE
    STORE M:REC1 TO O:PREVREC

    STORE T TO O:WHICH
    DO WHILE O:WHICH

```



```
STCRE '1I' TO M:TYPE
DO C:XDBHNDLR.FRG
```

```
IF M:NSN <> M:KEY .OR. ECF
STCRE O:PREVREC TO M:REC1
STORE '1H' TC M:TYPE
DO C:XDBHNDLR.FRG
STORE F TO C:WHICH
```

```
***** THESE MUST BE RELEASED OR ELSE TOO MANY VARIABLES
***** WILL BE ASSIGNED (IE >64)
```

```
RELEASE M:UI,M:UPRC,M:WUC,M:ACTDISP,M:ACTPT,;
M:DETAILS,M:DEFV,M:DEPR,;
M:O9C,M:DOC,M:ORG
```

```
STORE ' ' TO M:DATES
STORE ' ' TO M:CLOSE
STCRE ' ' TO M:UIC
STORE ' ' TO M:FSCM
STCRE ' ' TO M:NUM
STORE ' ' TO M:MFG
STCRE ' ' TO M:REOPEN
STORE ' ' TO M:DOCNO
STCRE ' ' TO M:SCRDAT
STORE ' ' TO M:REPCON
STCRE ' ' TO M:TIME
STORE 0 TO M:QTYINS
STCRE 0 TO M:QTYREC
STORE 0 TO M:QTYSTK
STCRE 0 TO M:QTYDEF
```

```
ELSE
STCRE M:REC1 TO O:PREVREC
ENDIF
```

```
ENDDO <O:WHICH>
ENDIF <OPENFILE>
STCRE M:CASE TO O:CCASE
```

```
***** SAVE THE CASE FROM THE OPENFILE FOR FUTURE COMPARISON
***** GO TO THE CLOSED DATA BASE AND CHECK FOR CASE WITH
***** THAT NSN
```

```
STORE '3A' TO M:TYPE
DO C:XLBENDLR.FRG
```

```
***** CCNTROL RETURNS TO THIS PROGRAM
***** IF M:TYPE = 9 THEN THERE IS NOT A CASE IN THE CLOSED
***** FILE
```

```
IF M:TYPE= '9'
STORE T TO O:CNONE
ELSE
STORE F TO O:CNONE
STORE M:REC1 TO O:PREVREC
```

```
STCRE T TO O:WHICH
DO WHILE O:WHICH
STCRE '3I' TO M:TYPE
DO C:XDBHNDLR.FRG
STCRE M:REC1 TC O:PREVREC
```

```
IF M:NSN <> M:KEY .OR. ECF
STCRE O:PREVREC TO M:REC1
STORE '3H' TC M:TYPE
DO C:XDBHNDLR.FRG
STORE F TO C:WHICH
```


***** THESE MUST BE RELEASED OR ELSE TOO MANY VARIABLES
***** WILL BE ASSIGNED (IE >64)

```
RELEASE M:UI,M:UPRC,M:WUC,M:ACTDISP,M:ACTPT,;  
M:DETAILS,M:DEFV,M:DEFR,M:O9Q,M:DOC,;  
M:ORG,M:REPLY,M:ACTTKN,M:STATUSC,;  
M:CAUSEC,R ETC
```

```
STORE ' ' ;  
STORE ' ' TO M:DATES  
STORE ' ' TO M:CLCSE  
STORE ' ' TO M:UIC  
STORE ' ' TO M:FSCM  
STORE ' ' TO M:NUM  
STORE ' ' TO M:MFG  
STORE ' ' TO M:REOPEN  
STORE ' ' TO M:DOCNO  
STORE ' ' TO M:SCRDAT  
STORE ' ' TO M:REPCON  
STORE ' ' TO M:TIME  
STORE 0 TO M:QTYINS  
STORE 0 TO M:QTYREC  
STORE 0 TO M:QTYSTK  
STORE 0 TO M:QTYDEF
```

```
ELSE  
STORE M:REC1 TO O:PREVREC  
ENDIF
```

```
ENDDO <O:WHICH>  
STORE M:CASE TO O:CCASE  
ENDIF <CIOS EFILE>
```

***** COMPARE THE VALUES OF CASE NUMBER FROM OPEN AND
***** CLCSE, AND USE THE LARGEST ONE FOR SUFFIX
***** CALCULATION

```
IF C:OCASE > O:CCASE  
STORE O:OCASE TO M:CASE  
RELEASE O:CCASE,O:OCASE  
ELSE  
STORE O:CCASE TO M:CASE  
RELEASE O:OCASE,O:CCASE  
ENDIF
```

***** ONLY GO INTO THE NEXT IF-ENDIF WHERE THE NSN WAS NOT
***** FOUND IN EITHER THE OPEN OR THE CLOSED FILE

```
IF O:CNCNE .AND. O:CNCNE
```

```
ELSE
```

***** CALCULATE SUFFIX FOR THE ADDITIONAL CASE FOR THE NSN

```
STORE $(M:CASE,7,1) TO O:LAST  
IF O:LAST = '  
STORE $(M:CASE,1,6) + 'A' TO M:CASE  
ELSE  
STORE RANK(C:LAST) + 1 TO O:SUFFIX  
STORE CHR(C:SUFFIX) TO O:LETTER  
STORE $(M:CASE,1,6) + O:LETTER TO M:CASE  
ENDIF
```

```
ENDIF
```

```
ENDIF  
RELEASE C:LAST,O:LETTER,O:KEY,O:PREVREC,O:ONONE,O:CNONE,O:WHICH  
ERASE
```


***** START OF NEW CASE DATA ENTRY

```
@ 0,1 SAY 'SF368'
@ 1,2 SAY '5. NSN
@ 2,2 SAY ' CATEGORY
@ 3,2 SAY ' SMIC ' GET M:SM ;
PICTURE 'AX'
@ 4,2 SAY '1A. UIC ' GET M:UIC
@ 5,2 SAY '3. REFCRT CONTROL
:
GET M:REPCON PICTURE 'XXXXXX-99-9999'
@ 6,2 SAY '4. DATE DISCOVERED MMDDYY ' ;
GET C:DDATE PICTURE 'XXXXXX'
@ 7,2 SAY '6. NOMENCLATURE ' ;
GET M:NOMEN PICTURE 'XXXXXXXXXXXXXXXXXXXXX'
@ 8,2 SAY '7. FSCM <O> ' ;
GET M:FSCM PICTURE 'XXXXXX'
@ 9,2 SAY '8. MFG. PART NUMBER <O> ' ;
GET M:MFG PICTURE 'XXXXXXXXXXXXXXXXXXXXX'
@ 10,2 SAY '9. SERIAL/LOT/BATCH <O> ' ;
GET M:LOT PICTURE 'XXXXXXXXXX'
@ 11,2 SAY '10. CONTRACT/PO <O> ' ;
GET M:NUM PICTURE 'XXXXXX-99-A-XXXX-XXXX'
@ 12,2 SAY ' DOCUMENT NUMBER <O> ' ;
GET M:DOCNO PICTURE 'XXXXXX-9999-9999'
@ 13,2 SAY '11. ITEM N OR O <O> ' ;
GET M:ITEM PICTURE 'A'
@ 14,2 SAY '12. DATE MFG/REP/OVHL <O> ' ;
GET M:OVER PICTURE '99999'
@ 15,2 SAY '13. OPN TIME AT FAILURE <O> ' ;
GET M:OTF PICTURE 'AXXX'
@ 16,2 SAY '14. GOV FURNISHED MATL <O> ' ;
GET M:GOV PICTURE 'X'
@ 17,2 SAY '15. QTY: REC/INSP/DEF/STK ' ;
GET M:QTYREC PICTURE '999999'
@ 18,2 SAY '16A1. TYPE/MODEL/SERIES <O> ' GET O:MODEL
@ 19,2 SAY ' A2. SERIAL NUMBER <O> ' GET O:DEFSER
@ 20,2 SAY ' B. NEXT HIGHER ASSY <O> ' GET O:HASSY
@ 21,2 SAY ' SUE-ASSEMBLY <O> ' GET C:SASSY
```

```
@ 1,38 SAY $(M:KEY,1,4)+'-'+'$(M:KEY,5,2)+'-'+'$(M:KEY,7,3);
+'-'+'$(M:KEY,10,4)
@ 2,38 SAY M:CAT
CLEAR GETS
```

```
STORE T TO C:CORRECT
DO WHILE O:CORRECT
```

```
IF M:COG = '1H' .OR. M:COG = '2H' .OR. M:COG = '7H'
  STORE T TO O:SMIC1
  DC WHILE O:SMIC1
    @ 3,2 SAY ' SMIC ' GET;
    M:SM PICTURE 'AX'
    READ
    STCRE ! (M:SM) TO M:SM
    IF $(M:SM,1,1) = 'X' .OR. $(M:SM,1,1) = 'L'
      STORE F TO O:SMIC1
    ELSE
      @ 23,30 SAY 'X OR L ONLY'
    ENDIF
  ENDDC <C:SMIC1>
ENDIF
RELEASE O:SMIC1
@ 23,30 SAY '
```

```
STORE T TO C:UIC1
DO WHILE O:UIC1
  @ 4,35 SAY ' ' GET M:UIC PICTURE 'AXXXXX'
```



```

STCRE !(M:UIC) TC M:UIC
READ
IF $(M:UIC,1,1) = ' ' .OR. $(M:UIC,2,1) = ' ' .OR. ;
  $(M:UIC,3,1) = ' ' .OR. $(M:UIC,4,1) = ' ' .OR. ;
  $(M:UIC,5,1) = ' ' .OR. $(M:UIC,6,1) = ' ' ;
  @ 23,20 SAY ' NO BLANKS ALLOWED IN UIC'
ELSE
  STORE F TO O:UIC1
ENDIF
ENDDO <O:UIC1>
@ 23,20 SAY '
RELEASE C:UIC1

***** REPORT CONTROL

***** REPORT CONTROL NUMBER (RCN) FORMAT CHANGED DUE TO
***** MSG FROM FMSO NCV83
***** OLD: 'XXXXXX-XXXX-XXXX' NEW: 'XXXXXX-99-9999'

@ 5,35 SAY ' ' GET M:REPCON PICTURE 'XXXXXX-99-9999'
READ

***** 1.TAKE DATE TO JULIAN FORMAT 2. NUMERIC DATA

STORE T TO C:DDATET
DO WHILE O:DDATET
  @ 6,35 SAY ' ' GET O:DDATE PICTURE '999999'
  READ
  IF C:DDATE = ' '
    @ 23,30 SAY 'MAY NOT BE BLANK'
  ELSE
    IF $(O:DDATE,1,2) < '01' .OR. $(O:DDATE,1,2) > '12';
      .OR. $(O:DDATE,3,2) < '01';
      .OR. $(O:DDATE,3,2) > '31';
      .OR. $(O:DDATE,5,2) < O:LLIMIT;
      .OR. $(O:DDATE,5,2) > O:ULIMIT;
      @ 23,30 SAY ' DATE OUT OF RANGE'
    ELSE
      STORE F TO O:DDATET
    ENDIF
  ENDIF
ENDIF
ENDDC <O:DDATET>
@ 23,30 SAY '
RELEASE C:DDATET

***** CALL C:OJULIAN TO CONVERT TO JULIAN DATE

STORE VAI($ (O:DDATE,1,2)) TO V:MM
STORE VAI($ (O:DDATE,3,2)) TO V:DD
STORE VAI($ (O:DDATE,5,2)) TO V:YY
DO C:OJULIAN
STORE V:JULDATE TO M:DDATE
RELEASE ALL LIKE V:*

STORE T TC C:NOMEN
DO WHILE O:NOMEN
  @ 7,35 SAY ' ' GET M:NOMEN PICTURE 'XXXXXXXXXXXXXXXXXXXXX'
  READ
  IF $(M:NOMEN,1,1) = ' ' .OR. $(M:NOMEN,2,1) = ' ' ;
    .OR. $(M:NOMEN,3,1) = ' ' ;
    @ 23,30 SAY ' NO BLANKS IN FIRST 3 POSITIONS'
  ELSE
    STORE F TO O:NOMEN
  ENDIF
ENDIF
ENDDO <O:NOMEN>
@ 23,30 SAY '
RELEASE C:NCMEN

***** INPUT FSCM

```



```

@ 8,35 SAY ' ' GET M:FSCM PICTURE 'XXXXXX'
***** INPUT MANUFACTURERS PART NUMBER
@ 9,35 SAY ' ' GET M:MFG PICTURE 'XXXXXXXXXXXXXXXXXX'
@ 10,35 SAY ' ' GET M:LOT PICTURE 'XXXXXXXXXX'
***** INPUT CONTRACT NUMBER
@ 11,35 SAY ' ' GET M:NUM PICTURE 'XXXXXX-99-A-XXXX-XXXX'
***** INPUT DOCUMENT NUMBER
STORE T TO O:UICT
STORE T TO O:PREPT
STORE T TO O:DOCT
DO WHILE C:DOCT .OR. C:UICT
  @ 12,35 SAY ' ' GET M:DOCNO PICTURE 'AXXXXX-9999-9999'
  READ
  IF M:DOCNO = ' - - '
    STORE F TO O:DOCT
    STORE F TO O:UICT
  ELSE
    IF $(M:DOCNO,1,1) = ' ' .OR. $(M:DOCNO,2,1) = ' ' ;
      .OR. $(M:ECCNO,3,1) = ' ' ;
      .OR. $(M:ECCNO,4,1) = ' ' ;
      .OR. $(M:ECCNO,5,1) = ' ' ;
      @ 23,20 SAY ' NO BLANKS ALLOWED IN UIC '
    ELSE
      STORE F TO O:DOCT
    ENDIF
    IF $(M:DCCNO,12,3) > '366' ;
      .OR. $(M:ECCNO,12,3) = ' ' ;
      .OR. $(M:ECCNO,11,4) = ' ' ;
      @ 23,50 SAY 'PREP DATE OUT OF RANGE '
    ELSE
      STORE F TO O:UICT
    ENDIF
  ENDIF <ALL BLANKS>
ENDDO <O:DOCT .AND. C:UICT>
RELEASE C:UICT,O:DOCT
@ 23,20 SAY '
***** DOCUMENT NUMBER END
STORE T TO O:ITEM
DO WHILE O:ITEM
  @ 13,35 SAY ' ' GET M:ITEM PICTURE 'A'
  READ
  IF M:ITEM = 'N' .OR. M:ITEM = 'O' .OR. M:ITEM = ' '
    STORE F TO C:ITEM
  ELSE
    @ 23,30 SAY ' USE N OR O '
  ENDIF
ENDDC <C:ITEM>
@ 23,30 SAY '
RELEASE C:ITEM
IF M:ITEM <> ' '
***** THE NEXT FIVE LINES CALCULATE EARLIEST YEAR TO ALLOW
***** FOR OVERHAUL ENTRY
STORE $(C:JULIAN,1,2) TO TEMP1
STORE VAL(TEMP1) TO TEMP1A
STORE VAL('10') TO LCW
STORE TEMP1A-LOW TO TEMP2

```



```

STORE STR(TEMP2,2) TC O:TENYRS
RELEASE TEMP1,TEMP1A,TEMP2,LOW
STORE T TO O:OVER
DC WHILE O:OVER
  @ 14,35 SAY ' ' GET M:OVER PICTURE '99999'
  READ
  IF M:OVER=' '
    STORE F TO O:OVER
  ELSE
    IF $(M:OVER,3,3) > '365':
      .OR. $(M:OVER,1,2) < O:TENYRS ;
      .OR. $(M:OVER,1,2) > O:ULIMIT ;
      @ 23,30 SAY 'DATE OUT OF RANGE'
    ELSE
      STORE F TO O:OVER
    ENDIF
  ENDIF
ENDDC <O:OVER>
@ 23,30 SAY ' '
ENDIF
RELEASE C:OVER,O:TENYRS

```

```

STORE T TO O:OTF
DO WHILE O:CTF
  @ 15,35 SAY ' ' GET M:OTF PICTURE 'A9999'
  READ
  IF M:OTF = ' '
    STORE F TO O:CTF
  ELSE
    IF $(M:OTF,1,1) = 'N' .OR. $(M:OTF,1,1) = 'O';
      .AND. $(M:OTF,2,4) > '0000'
      STORE F TO O:CTF
    ELSE
      @ 23,30 SAY 'USE N OR O AND THEN TIME (A9999)'
    ENDIF
  ENDIF
ENDDO <O:OTF>
@ 23,30 SAY ' '
RELEASE C:OTF
STORE T TO C:GOV
DO WHILE O:GOV
  @ 16,35 SAY ' ' GET M:GOV PICTURE 'X'
  READ
  IF M:GOV = ' ' .OR. M:GOV = 'Y' .OR. M:GOV = 'N'
    STORE F TO C:GOV
  ELSE
    @ 23,30 SAY 'USE EITHER Y OR N OR LEAVE BLANK'
  ENDIF
ENDDO <O:GOV>
@ 23,30 SAY ' '
RELEASE C:GOV
STORE T TO C:QTYRECT
DO WHILE O:QTYRECT

```

```

  @ 17,35 SAY ' ' GET M:QTYREC PICTURE '999999'
  READ
  IF M:QTYREC < 0 .OR. M:QTYREC > 999999
    @ 23,30 SAY 'CUT OF RANGE'
  ELSE
    STORE F TO O:QTYRECT
  ENDIF
ENDDO
@ 23,30 SAY ' '
RELEASE C:QTYRECT
STORE T TO C:QTYINS
DO WHILE O:QTYINS
  @ 17,43 SAY '/' GET M:QTYINS PICTURE '999999'
  READ

```

```

  @ 17,35 SAY ' ' GET M:QTYREC PICTURE '999999'
  READ
  IF M:QTYREC < 0 .OR. M:QTYREC > 999999
    @ 23,30 SAY 'CUT OF RANGE'
  ELSE
    STORE F TO O:QTYRECT
  ENDIF
ENDDO
@ 23,30 SAY ' '
RELEASE C:QTYRECT
STORE T TO C:QTYINS
DO WHILE O:QTYINS
  @ 17,43 SAY '/' GET M:QTYINS PICTURE '999999'
  READ

```



```

IF M:QTYINS < 0 .CR. M:QTYINS > 999999
  @ 23,30 SAY ' CUT OF RANGE'
ELSE
  STORE F TO O:QTYINS
ENDIF
ENDDO <O:QTYINS>
@ 23,30 SAY '
RELEASE C:QTYINS
STORE T TO C:QTYDEF
DO WHILE O:QTYDEF
  @ 17,50 SAY '/' GET M:QTYDEF PICTURE '999999'
  READ
  IF M:QTYDEF < 1 .CR. M:QTYDEF > 999999
    @ 23,30 SAY ' DEFICIENT'
  OUT CF FANGÉ'
  ELSE
    STORE F TO O:QTYDEF
  ENDIF
ENDDO O:QTYDEF
RELEASE C:QTYDEF
@ 23,30 SAY '
STORE T TO O:QTYSTK
DO WHILE O:QTYSTK
  @ 17,57 SAY '/' GET M:QTYSTK PICTURE '999999'
  READ
  IF M:QTYSTK < 0 .CR. M:QTYSTK > 999999
    @ 23,30 SAY ' IN STOCK'
  OUT OF RANGE'
  ELSE
    STORE F TO O:QTYSTK
  ENDIF
ENDDO O:QTYSTK
@ 23,30 SAY '
RELEASE C:QTYSTK

@ 18,35 SAY ' ' GET C:MODEL PICTURE 'XXXXXXX'
READ
@ 19,35 SAY ' ' GET C:DEFSER PICTURE 'XXXXXX'
READ
@ 20,35 SAY ' ' GET C:HASSY PICTURE 'XXXXXXXXXXXX'
READ
@ 21,35 SAY ' ' GET C:SASSY PICTURE 'XXXXXXXXXXXXXXX'
READ
STORE O:MODEL+O:DEFSER+O:HASSY+ O:SASSY TO M:DITEM

***** PROMPT USER FOR RESPONSE

STORE T TO C:END
DO WHILE O:END

  STORE ' ' TC O:REPLY
  @ 22,10 SAY ' ***** CHECK PREVIOUS ENTRIES ' ;
  + '***** '
  @ 23,10 SAY ' CHOOSE 1- CONTINUE ENTRY 2- MAKE ' ;
  + 'CORRECTIONS ' GET O:REPLY
  READ
  IF C:REPLY <> '1' .AND. O:REPLY <> '2'
    @ 23,10 SAY ' ANSWER WITH A 1 OR 2 ONLY
  ELSE
    STCRE F TO C:END
  ENDIF
ENDDO <O:END>

IF O:REPLY = '2'
  STCRE T TO O:CORRECT
  @ 22,10 SAY '
  @ 23,10 SAY '
ELSE

```



```

STORE F TO O:CORRECT
ENDIF
ENDDC <O:CORRECT>
ERASE
RELEASE C:MODEL,O:DEFSER,O:HASSY,O:SASSY,O:END,O:COUNT

***** HERE IS THE COMPRESSION OF M:REPCON,M:NUM,M:DOCNO

STORE $(M:REPCON,1,6)+$(M:REPCON,8,2)+$(M:REPCON,11,4) TO ;
O:REPCON
STORE C:FEPCON TC M:REPCON
STORE $(M:NUM,1,6)+$(M:NUM,8,2)+$(M:NUM,11,1)+$(M:NUM,13,4) ;
+$(M:NUM,18,4) TO O:NUM
STORE O:NUM TO M:NUM
STORE $(M:DOCNO,1,6)+$(M:DOCNO,8,4)+$(M:DOCNO,13,4) TO ;
O:DOCNO
STORE O:DCCNO TO M:DCCNO
RELEASE C:REPCON,O:NUM,O:DOCNO

```

```

***** CAPTURE THE JULIAN DATE AND PUT INTO OPENING DATE

STORE C:JULIAN TO M:OPEN
STORE M:DDATE+M:RDATE+M:OPEN+;
+' N' TO M:DATES
RELEASE M:DATE,M:RDATE,M:OPEN,O:ACTPPT,O:PREPT,;
O:LLIMIT,O:ULIMIT

```

```

***** THIS IS THE START OF THE SECOND SCREEN OF DATA ENTRY

STORE ' ' TO M:UI
STORE 0 TO M:UPRC
STORE ' ' TO M:WUC
STORE ' ' TC M:ACTDISP
STORE ' ' TO M:ACTPT
STORE ' ' TO M:DETAILS
STORE ' ' TO M:DEFV
STORE ' ' TC M:DEFR
STORE ' ' TO M:DEF
STORE ' ' TC M:O9Q
STORE ' ' TO M:DOC
STORE ' ' TO M:ORG
STORE 0 TO M:CCOST
STORE T TO C:PAGE2
DO WHILE O:PAGE2

```

```

@ 0,10 SAY ' ' UI ' ;
GET M:UI
@ 1,10 SAY ' ' UNIT PRICE ' GET M:UPRC PICTURE ;
'999999.99'
@ 2,10 SAY '18. EST. CORRECTION COST <O> 'GET;
M:CCOST PICTURE '999999999.99'
@ 3,10 SAY '19. WARRANTY - Y/N/U 'GET;
M:WNTY PICTURE 'A'
@ 4,10 SAY '20. WORK UNIT CODE <O> 'GET;
M:WUC
@ 5,10 SAY '21. ACTION/DISPOSITION -H/I/D/R/O <O> 'GET;
M:ACTDISP PICTURE 'X'
@ 6,10 SAY '22. DETAILS OF DISCREPANCY - FIRST 2 '
@ 7,10 SAY ' LETTERS MUST BE DISCOVERY CODE'
@ 8,10 SAY ' ' GET M:DETAILS
@ 12,10 SAY '23A. ACTION POINT ' ;
GET M:ACTPT PICTURE 'AXXXX99999'
@ 13,10 SAY ' DEFECT VERIFICATION CODE - N/O/U/Y <O>' ;
GET M:DEFV PICTURE 'A'
@ 14,10 SAY ' DEFECT RESPONSIBILITY - C/N/S/U/X <O>' ;
GET M:DEFR PICTURE 'A'
@ 15,10 SAY ' 9Q ' ;

```



```

@ 16,10 GET M:O9Q PICTURE 'X'
        SAY ' ORIGIN CODE ' ;
@ 17,10 GET M:ORG PICTURE 'AAX'
        SAY '30. TYPE DOC ' ;
@ 18,10 GET M:DOC PICTURE '9'
        SAY ' TYPE DEFICIENCY ' ;
        GET M:DEF PICTURE '99'

```

CLEAR GETS

```

        STORE T TO O:UI
        DO WHILE O:UI
            @ 0,10 SAY ' UI ' ;
            GET M:UI PICTURE 'AA'
            READ
            IF $(M:UI,1,1) = ' ' .OR. $(M:UI,2,1) = ' '
                @ 23,30 SAY ' NO BLANKS '
            ELSE
                STORE F TO O:UI
            ENDIF
        ENDDO O:UI
        @ 23,30 SAY '
        RELEASE O:UI

```

```

        STORE T TO C:EPRC
        STORE T TO O:UPRC
        DO WHILE O:UPRC .OR. O:EPRC

```

```

            DO WHILE O:UPRC
                @ 1,10 SAY ' UNIT PRICE ' GET;
                M:UPRC PICTURE '999999.99'

```

```

                READ
                IF M:UPRC < .01 .OR. M:UPRC > 999999.99
                    @ 23,30 SAY ' AMOUNT OUT OF RANGE '
                ELSE

```

```

                    STORE F TO O:UPRC
                ENDIF

```

```

            ENDDO <C:UPRC>

```

```

            @ 23,30 SAY '
            STORE (M:UPRC * M:QTYDEF) TO M:EPRC

```

```

            @ 1,43 SAY 'EXT PRICE $'
            @ 1,54 SAY M:EPRC PICTURE '999999999.99'

```

```

            IF M:EPRC >= 100000000
                @ 23,30 SAY ' EXTENDED PRICE OUT OF RANGE '
            ELSE

```

```

                STORE F TO O:EPRC
            ENDIF

```

```

        ENDDO <O:UPRC & O:EPRC>

```

```

        RELEASE O:UPRC,O:EPRC

```

```

        @ 23,30 SAY '
        @ 2,10 SAY '18. EST. CORRECTION COST ' ;
        + '<O>' GET M:CCOST PICTURE '999999999.99'
        READ

```

```

        STORE T TO C:WNTY

```

```

        DO WHILE O:WNTY

```

```

            @ 3,10 SAY '19. WARRANTY - Y/N/U ' ;
            GET M:WNTY PICTURE 'A'

```

```

            STORE !(M:WNTY) TO M:WNTY

```

```

            READ

```

```

            IF M:WNTY <> 'Y' .AND. M:WNTY <> 'N' ;
                .AND. M:WNTY <> 'U'

```

```

                @ 23,30 SAY 'USE Y,N OR U '
            ELSE

```

```

                STORE F TO O:WNTY
            ENDIF

```

```

        ENDDO <O:WNTY>

```

```

        @ 23,30 SAY '

```

```

        RELEASE O:WNTY

```



```

@ 4,10 SAY '20. WORK UNIT CODE ;
      <O> ' GET M:WUC PICTURE 'XXXXXXX'
STORE T TO C:ACTDISP
DO WHILE O:ACTDISP
  @ 5,10 SAY '21. ACTION/DISPOSITION ' ;
      + '-H/I/D/R/O <O> ' ;
      GET M:ACTDISP PICTURE 'X'
  READ
  IF M:ACTDISP = 'H' OR. M:ACTDISP = 'I' ;
      .OR. M:ACTDISP = 'D' .OR. M:ACTDISP = 'R' ;
      .OR. M:ACTDISP = 'O' .OR. M:ACTDISP = ' ' ;
      STORE F TO O:ACTDISP
  ELSE
    @ 23,30 SAY ' ERROR IN CODE'
  ENDIF
ENDDO <C:ACTDISP>
@ 23,30 SAY '
RELEASE O:ACTDISP

STORE T TO O:DISCODE
DO WHILE O:DISCODE
  @ 6,10 SAY '22. DETAILS OF DISCREPANCY - FIRST 2 '
  @ 7,10 SAY ' LETTERS MUST BE DISCOVERY CODE'
  @ 8,10 SAY ' ' GET M:DETAILS
  READ
  STORE $(M:DETAILS,1,2) TO M:DIS
  STORE !(M:DIS) TO M:DIS
  USE D:WHEREDIS INDEX D:DISCODE
  FIND &M:DIS
  IF # = 0
    @ 23,30 SAY 'WHERE DISCOVERED CODE INCORRECT'
  ELSE
    STORE F TO O:DISCODE
  ENDIF
ENDDO <O:DISCODE>
@ 23,30 SAY '
RELEASE O:DISCODE

STORE T TO C:ACTPTT
DO WHILE O:ACTPTT
  @ 12,10 SAY '23A. ACTION POINT ;
      ' GET M:ACTPT PICTURE 'AXXXXX99999'
  READ
  IF M:ACTPT = ' '
    @ 23,30 SAY 'MAY NOT BE BLANK'
  ELSE
    STORE F TO O:ACTPTT
  ENDIF
ENDDO <O:ACTPTT>
@ 23,30 SAY '

STORE T TO O:DEFV
DO WHILE C:DEFV
  @ 13,10 SAY ' DEFECT VERIFICATION CODE' ;
      + '-N/O/U/Y <O>' GET M:DEFV PICTURE 'A'
  READ
  IF M:DEFV = 'N' .OR. M:DEFV = 'O' ;
      .OR. M:DEFV = 'U' ;
      .OR. M:DEFV = 'Y' .OR. M:DEFV = ' '
    STORE F TO O:DEFV
  ELSE
    @ 23,30 SAY 'CORRECT CODE MUST BE ENTERED'
  ENDIF
ENDDO <O:DEFV>
@ 23,30 SAY '
RELEASE O:DEFV

```



```

STORE T TO C:DEFR
DO WHILE O:DEFR
  @ 14,10 SAY '          DEFECT RESPONSIBILITY -;
  +'C/N/S/U/X <O>' GET M:DEFR PICTURE 'A'
  READ
  IF M:DEFR = 'C' .OR. M:DEFR = 'N' ;
  .OR. M:DEFR = 'S' .OR. M:DEFR = 'U' ;
  .OR. M:DEFR = 'X' .OR. M:DEFR = ' ' ;
  STORE F TO O:DEFR
  ELSE
    @ 23,30 SAY 'CORRECT CODE MUST BE ENTERED'
  ENDIF
ENDDO <O:DEFR>
@ 23,30 SAY '
RELEASE O:DEFR

IF M:COG = '9Q'
  STORE T TO O:9Q
  DO WHILE O:9Q
    @ 15,10 SAY '          9Q
    +'          ' GET M:09Q PICTURE 'X';
    READ
    IF M:09Q = '2' .OR. M:09Q = '4' ;
    .OR. M:09Q = '5' .OR. M:09Q = '7' ;
    .OR. M:09Q = '9' .OR. M:09Q = ' ' ;
    STORE F TO O:9Q
    ELSE
      @ 23,30 SAY ' OUT OF RANGE'
    ENDIF
  ENDDO <O:9Q>
  @ 23,30 SAY '
  RELEASE O:9Q
ENDIF <M:COG = 9Q>

STORE T TO C:ORG
DO WHILE O:ORG
  @ 16,10 SAY '          ORIGIN CODE
  +'          ' GET M:ORG PICTURE 'AAX' ';
  READ
  IF $(M:ORG,1,1) = ' ' .OR. $(M:ORG,2,1) = ' '
    @ 23,20 SAY ' FIRST 2 POSITIONS MAY NOT';
    +' CONTAIN BLANKS'
  ELSE
    STORE F TO O:ORG
  ENDIF
ENDDO <O:ORG>
@ 23,20 SAY '
RELEASE O:ORG

STORE T TO C:TYPE
DO WHILE O:TYPE
  @ 17,10 SAY '30. TYPE DOC
  +'          ' GET M:DOC PICTURE '9' ';
  READ
  IF M:DOC < '1' .OR. M:DOC > '7'
    @ 23,30 SAY 'OUT OF RANGE'
  ELSE
    STORE F TO O:TYPE
  ENDIF
ENDDO <O:TYPE>
@ 23,30 SAY '
RELEASE O:UIC,O:UIC2,O:PREP,O:DOC,O:SERNO,O:TYPE

STORE T TO C:DEF
DO WHILE O:DEF
  @ 18,10 SAY '          TYPE DEFICIENCY
  +'          ' GET M:DEF PICTURE '99' ';
  READ
  IF M:DEF < '01' .OR. M:DEF > '19'

```



```

        @ 23,30 SAY 'USE 01 - 19 ONLY'
    ELSE
        STORE F TO O:DEF
    ENDIF
ENDDO <O:DEF>
RELEASE O:DEF
@ 23,30 SAY '

```

***** EFCMPT USER FOR RESPONSE

```

STORE T TO C:END
DC WHILE O:ENC

```

```

    STORE ' ' TO O:REPLY
    @ 20,20 SAY ' 1 - POST CASE'
    @ 21,20 SAY ' 2 - CHANGE DATA'
    @ 22,20 SAY ' 3 - EXIT WITHOUT POSTING '

    @ $+1,34 SAY ' ' GET O:REPLY
    READ
    IF O:REPLY <> '1' .AND. O:REPLY <> '2' ;
        .AND. O:REPLY <> '3'
        @ 23,5 SAY ' ANSWER WITH 1 - 2 - 3 ONLY '
    ELSE
        STORE F TO O:END
    ENDIF
ENDDO <C:END>
@ 23,10 SAY '
RELEASE O:ENI

```

```

ERASE
IF O:REPLY = '1'
    @ 10,20 SAY 'CASE BEING POSTED TO DATA BASE '
    @ 13,20 SAY ' PLEASE STANDBY '
    @ 20,20 SAY '*** DO NOT INTERRUPT *** '

```

```

    STORE F TO O:PAGE2
    STORE '1F' TO M:TYPE
    DO C:XDEHNDLR.PRG
    STORE '2F' TO M:TYPE
    DO C:XDEHNDLR.PRG

```

```

ERASE
@ 10,20 SAY ' CASE NUMBER OF THE NEW CASE '
@ 12,33 SAY ' M:CASE '
@ 23,20 SAY ' PRESS ANY KEY TO CONTINUE '
WAIT

```

```

ENDIF
IF O:REPLY = '2'
    STORE T TO O:PAGE2
ELSE
    IF C:REPLY = '3'
        STORE F TO O:PAGE2
    ENDIF
ENDIF

```

```

ENDDC <C:PAGE2>
RELEASE ALL EXCEPT C:*
STORE T TO O:TRUE
ENDDO <O:TRUE>
RETURN

```

***** END OF PROGRAM

V. CASE UPDATE MODULE

```

*****
**
** DATE: 8 DECEMBER 1983
** VERSICN: 1.0
** MCDUIE NAME: UPDATE
** MCDUIE PURPOSE: ALLOW ADDITION AND/OR CORRECTION OF
**                  DATA IN QDR CASE CURRENTLY IN QDR
**                  SYSTEM.
** MCDUIE INTERFACE DEFINITION
**   INPUTS: CASE, C:WHO, C:JULIAN
**   CUTPUTS: ALL DATA ELEMENTS IN OPEN1 & OPEN2,
**            M:TYPE
** MCDUIE PROCESSING NARRATIVE DESCRIPTION:
**
**   USER ENTERS CASE NUMBER OF CASE TO BE CHANGED.
**   MCDULE SEARCHES DATA BASE FOR CASE AND
**   DISEPLAYS INFORMATION CURRENTLY ON FILE THROUGH
**   A SERIES OF THREE MENUS. DATA IS WRITTEN TO
**   FIRST DATA BASE MIDWAY IN PROCESS DUE TO
**   LIMIT OF 64 MEMORY VARIABLES AT ANY ONE TIME.
**   CHANGE OF DATES IS NOTED FOR STATISTIC
**   MCDULE UTILIZATION.
**
** SUPERORDINATE MODULES: MENU1
** SUPORDINATE MODULES: XDEHNLRL
** AUTHOR: J.G. BOYNTON
**
*****

```

```

STORE T TO U:UPDATE
DO WHILE U:UPDATE
  STORE T TC U:TRUE
  DO WHILE U:TRUE
    ERASE
    STORE ' ' TO U:CHOICE
  TEXT

```

***** UPDATE *****

THIS PROGRAM ALLOWS YOU TO
UPDATE A QDR CASE

- 1 - CONTINUE
- 2 - RETURN TO MENU

```

ENDTEXT
@ 20,40 SAY ' ' GET U:CHOICE
READ
DO WHILE U:CHOICE <> '1' .AND. U:CHOICE <> '2'
  @ 23,20 SAY 'ENTER 1 OR 2 FOR YOUR RESPONSE'
  @ 20,40 SAY ' ' GET U:CHOICE
  READ
ENDDO <U:CHOICE>
ERASE
IF U:CHOICE = '2'
  RELEASE ALL EXCEPT C:*

```



```

        RETURN
    ENDIF

ERASE STORE ' ' TO M:CASE
TEXT ***** SELECT RECORD FOR UPDATE *****

                ENTER THE CASE NUMBER
                OF THE RECORD TO BE UPDATED

ENDTEXT
STORE ' ' TC U:REPLY
@ 10, 29 SAY 'CASE ' GET M:CASE PICTURE '999999X'
READ
STORE M:CASE TO M:KEY
STORE '1E' TO M:TYPE
DO C:XIBHNDLR
IF M:TYPE = '9'
    @ 12,25 SAY 'RECORD NOT FOUND IN OPEN FILE '
    @ 13,21 SAY 'DO YCU WISH TO CHECK THE CLOSED FILE ?'
    @ 14,40 GET U:REPLY PICTURE 'A'
    READ
    DO WHILE !(U:REPLY) <> 'Y' .AND. !(U:REPLY) <> 'N'
        @ 14,45 SAY 'ENTER Y OR N'
        @ 14,40 GET U:REPLY PICTURE 'A'
        READ
    ENDDO
    @ 14,45 SAY ' '
    IF !(U:REPLY) = 'Y'
        STORE '3E' TO M:TYPE
        DO C:XDBHNDLR
        IF M:TYPE = '9'
            @ 16,23 SAY 'RECORD NOT FOUND IN THE QDR SYSTEM'
            @ 17,27 SAY 'STRIKE ANY KEY TO CONTINUE'
            WAIT TO U:REPLY
        ELSE
            IF M:TYPE = '1'
                @ 18,28 SAY 'RECORD CURRENTLY IN USE'
                @ 19,27 SAY 'STRIKE ANY KEY TO CONTINUE'
                WAIT TO U:REPLY
            ELSE
                ERASE
                STORE F TC U:TRUE
                STORE 'CLCSE' TO U:FILE
            ENDIF
        ENDIF
    ENDIF
ELSE
    IF M:TYPE = '1'
        @ 12,28 SAY 'RECORD CURRENTLY IN USE'
        @ 13,27 SAY 'STRIKE ANY KEY TO CONTINUE'
        @ 14,40 GET U:REPLY
        READ
    ELSE
        ERASE
        STORE F TO U:TRUE
        STORE 'OPEN' TC U:FILE
    ENDIF
ENDIF
ENDIF
ENDDO <U:TRUE>

***** THIS SECTION FOR CURRENT DATES VALUE CAPTURE
ERASE
STORE $(M:DATES,1,5) TO M:DDATE

```



```

STORE $(M: DATES, 6, 5) TO M: RDATE
STORE $(M: DATES, 11, 5) TO M: OPEN
STORE $(M: DATES, 16, 5) TO M: LDATE
STORE $(M: DATES, 21, 5) TO M: SCRDATE
STORE $(M: DATES, 26, 5) TO M: IRDATE
STORE $(M: DATES, 31, 5) TO M: RIMDAT
STORE $(M: DATES, 36, 5) TO M: CLCSE
STORE $(M: DATES, 41, 5) TO M: REOPEN

```

```

STORE M: DDATE TO T: DDATE
STORE M: RDATE TO T: RDATE
STORE M: OPEN TO T: CPEN
STORE M: LDATE TO T: LDATE
STORE M: SCRDATE TO T: SCRDATE
STORE M: IRDATE TO T: IRDATE
STORE M: RIMDAT TO T: RIMDAT
STORE M: CLCSE TO T: CIOSE
STORE M: REOPEN TO T: REOPEN

```

```

***** THIS SEQUENCE CALCULATES THE UPPER AND LOWER YEARS FOR
***** INPUT AND IS BASED ON THE CURRENT JULIAN DATE
***** U:LLIMIT= YEAR MINUS TWO YEARS
***** U:ULIMIT = YEAR PLUS ONE YEAR

```

```

STORE $(C: JULIAN, 1, 2) TO TEMP1
STORE VAL(TEMP1) TO TEMP1A
STORE VAL('2') TO LOW
STORE VAL('1') TO HIGH
STORE TEMP1A-LOW TO LLMT
STORE TEMP1A+HIGH TO ULMT
STORE STR(LLMT, 2) TO U:LLIMIT
STORE STR(ULMT, 2) TO U:ULIMIT
RELEASE TEMP1, TEMP1A, LOW, HIGH, LLMT, ULMT
ERASE

```

```

@ 3, 2 SAY ' DATES CURRENTLY IN FILE FOR CASE
@ 3, 45 SAY M: CASE
@ 8, 2 SAY ' DISCOVERY DATE 'GET M: DDATE
@ 9, 2 SAY ' RECEIVED FROM ORIGIN 'GET M: RDATE
@ 10, 2 SAY ' OPENING DATE * '
@ 10, 36 SAY M: OPEN
@ 11, 2 SAY ' TRANSMITTAL DATE 'GET M: LDATE
@ 12, 2 SAY ' SCREEN REPORT DATE 'GET M: SCRDATE
@ 13, 2 SAY ' INTERIM RESPONSE DATE 'GET M: IRDATE
@ 14, 2 SAY ' RETURN FROM ITEM MGR 'GET M: RIMDAT
@ 15, 2 SAY ' CLCSE * '
@ 15, 36 SAY M: CLOSE
@ 16, 2 SAY ' RECPEN 'GET M: RECPEN
@ 18, 2 SAY ' <* MAY NOT CHANGE THESE DATES> '
CLEAR GETS

```

```

STORE ' ' TO U: REPLY
STORE T TO U: DATET
DO WHILE U: DATET

```

```

STORE T TO U: DDATE
DO WHILE U: DDATE

```

```

@ 8, 35 SAY ' ' GET M: DDATE
READ
IF M: DDATE <> ' '
IF $(M: DDATE, 1, 2) < U: LLIMIT ;
.CR. $(M: DDATE, 1, 2) > U: ULIMIT;
.CR. $(M: DDATE, 3, 3) < '001';
.CR. $(M: DDATE, 3, 3) > '365';
.CR. M: DDATE > C: JULIAN

```



```

                @ 23,30 SAY 'DATE OUT OF RANGE'
            ELSE
                STORE F TO U:DDATET
            ENDIF
        ELSE
            STORE F TO U:DDATET
        ENDIF <ELANK>
    ENDDO <U:DDATET>
    @ 23,30 SAY '

STORE T TO U:RDATE
DO WHILE U:RDATE

    @ 9,35 SAY ' ' GET M:RDATE
    READ
    IF $(M:RDATE,1,2) < U:LLIMIT ;
        .OR. $(M:RDATE,1,2) > U:ULIMIT ;
        .OR. $(M:RDATE,3,3) < '001' ;
        .OR. $(M:RDATE,3,3) > '365' ;
        .OR. M:RDATE > C:JULIAN ;
        .OR. M:RDATE < M:DDATE ;
        .OR. M:RDATE > M:OPEN
            @ 23,30 SAY 'DATE OUT OF RANGE'
        ELSE
            STORE F TO U:RDATE
        ENDIF
    ENDDO <U:RDATE>
    @ 23,30 SAY '
    RELEASE U:DDATET,U:RDATE

STORE T TO U:LDATE
DO WHILE U:LDATE

    @ 11,35 SAY ' ' GET M:LDATE
    READ
    IF M:LDATE <> ' '
        IF $(M:LDATE,1,2) < U:LLIMIT ;
            .CR. $(M:LDATE,1,2) > U:ULIMIT ;
            .CR. $(M:LDATE,3,3) < '001' ;
            .CR. $(M:LDATE,3,3) > '365' ;
            .CR. M:LDATE > C:JULIAN ;
            .CR. M:LDATE < M:OPEN
                @ 23,30 SAY 'DATE OUT OF RANGE'
            ELSE
                STORE F TO U:LDATE
            ENDIF
        ELSE
            STORE F TO U:LDATE
        ENDIF
    ENDDO <U:LDATE>
    @ 23,30 SAY '

STORE T TO U:SCDATE
DO WHILE U:SCDATE

    @ 12,35 SAY ' ' GET M:SCDATE
    READ
    IF M:SCDATE <> ' '
        IF $(M:SCDATE,1,2) < U:LLIMIT ;
            .CR. $(M:SCDATE,1,2) > U:ULIMIT ;
            .OR. $(M:SCDATE,3,3) < '001' ;
            .OR. $(M:SCDATE,3,2) > '365' ;
            .CR. M:SCDATE < M:LDATE
                @ 23,30 SAY 'DATE OUT OF RANGE'
            ELSE
                STORE F TO U:SCDATE
            ENDIF
        ENDIF
    ENDIF

```



```

ELSE
  STORE F TO U:SCDATET
ENDIF
ENDDO <U:SCIATET>
@ 23,30 SAY '
RELEASE U:LDATET,U:SCDATET

STCRE T TO U:IRDATET
DO WHILE U:IRDATET

  @ 13,35 SAY ' ' GET M:IRDATE
  READ
  IF M:IRDATE <> ' '

    IF $(M:IRDATE,1,2) <U:LLIMIT ;
      .CR. $(M:IRDATE,1,2) > U:ULIMIT;
      .CR. $(M:IRDATE,3,3) <'001' ;
      .CR. $(M:IRDATE,3,3) > '365' ;
      .CR. M:IRDATE < M:OPEN
      @ 23,30 SAY 'DATE OUT OF RANGE'
    ELSE
      STORE F TO U:IRDATET
    ENDIF
  ELSE
    STORE F TO U:IRDATET
  ENDIF
ENDIF
ENDDO <U:IRDATET>
@ 23,30 SAY '

STCRE T TO U:RIMDAT
DO WHILE U:RIMDAT

  @ 14,35 SAY ' ' GET M:RIMDAT
  READ
  IF M:RIMDAT <> ' '

    IF $(M:RIMDAT,1,2) <U:LLIMIT ;
      .CR. $(M:RIMDAT,1,2) > U:ULIMIT;
      .CR. $(M:RIMDAT,3,3) <'001' ;
      .CR. $(M:RIMDAT,3,3) > '365' ;
      @ 23,30 SAY 'DATE OUT OF RANGE'
    ELSE
      IF M:RIMDAT < M:LDATE
        @ 23,30 SAY 'RTN DATE NOT ' ;
          + 'BEFORE TRANSMITTAL DATE'
      ELSE
        STORE F TO U:RIMDAT
      ENDIF
    ENDIF
  ELSE
    STORE F TO U:RIMDAT
  ENDIF
ENDIF
ENDDO <U:RIMDAT>
@ 23,30 SAY '
+ '
RELEASE U:IRDATET,U:RIMDAT

STCRE T TO U:REOPEN
DO WHILE U:REOPEN

  @ 16,35 SAY ' ' GET M:REOPEN
  READ
  IF M:REOPEN <> ' '

    IF $(M:REOPEN,1,2) <U:LLIMIT ;
      .CR. $(M:REOPEN,1,2) > U:ULIMIT;
      .CR. $(M:REOPEN,3,3) <'001' ;
      .CR. $(M:REOPEN,3,3) > '365' ;
      @ 23,30 SAY 'DATE OUT OF RANGE'

```



```

ELSE
  IF M:RECPEN < M:OPEN
    @ 23,30 SAY 'REOPEN DATE MAY NOT';
    + ' BE LESS THAN OPEN DATE !'
  ELSE
    STORE F TO U:REOPENT
  ENDIF
ENDIF
ELSE
  STORE F TO U:REOPENT
ENDIF
ENDDO <U:RECPENT>
@ 23,30 SAY ' ';
STORE T TO U:END
DO WHILE U:END
@ 21,10 SAY ' ***** CHECK DATES ABOVE ***** '
@ 22,10 SAY ' <CHOCSE> 1- CONTINUE 2- CHANGE 3-EXIT '
@ 23,10 SAY ' GET U:REPLY PICTURE '9'
READ
IF U:REPLY <> '1'.AND. U:REPLY <> '2'.AND. U:REPLY <> '3'
@ 22,10 SAY ' ANSWER WITH A 1 - 2 - 3 ONLY'
ELSE
  STORE F TO U:END
ENDIF
ENDDO <U:END>
@ 21,10 SAY '
@ 22,10 SAY '
@ 23,10 SAY '
RELEASE U:REOPENT,U:END

IF U:REPLY = '1'
  STORE F TO U:DATEI
  STORE T TO U:CCNT1
  IF M:DDATE <> T:DDATE .OR. M:RDATE <> T:RDATE ;
    .CR. M:OPEN <> T:OPEN .OR. M:LDATE <> T:LDATE ;
    .CR. M:SCRDATE <> T:SCRDATE ;
    .CR. M:IRDATE <> T:IRDATE .OR. M:RIMDAT <> T:RIMDAT ;
    .OR. M:CLOSE <> T:CLOSE .OR. M:REOPEN <> T:REOPEN
    STORE '*' TO M:DATECI
  ELSE
    STORE ' ' TO M:DATECI
  ENDIF
  RELEASE ALL LIKE T:*
  STORE M:DDATE+M:RDATE+M:OPEN+M:LDATE+M:SCRDATE;
  +M:IRDATE+M:RIMDAT+M:CLOSE+M:REOPEN+M:DATECI;
  TO M:DATES
  RELEASE M:DDATE,M:RDATE,M:OPEN,M:LDATE,M:SCRDATE;
  M:IRDATE,M:RIMDAT,M:CLOSE,M:REOPEN,M:DATECI

ELSE
  IF U:REPLY = '3'
    STORE F TC U:DATEI
    STORE F TC U:CONT1
    STORE F TC U:CONT2
    STORE F TC U:CONT3
    IF U:FILE = 'OPEN'
      STORE '1G' TO M:TYPE
    ELSE
      STORE '3G' TO M:TYPE
    ENDIF
    DO C:XDBHNLR
    RELEASE ALL EXCEPT C:*
    RETURN
  ENDIF
ENDIF

```


ENDDO <U:DATET>
RELEASE U:DATET, U:END

ERASE

DO WHILE U:CONT1

***** DISPLAY OF CASE DATA FROM FIRST DATABASE

```
@ 1,2 SAY '          NSN:'  
@ 1,17 SAY '$(M:NSN,1,4)+'-'+'$(M:NSN,5,2)+'-'+'  
          +$(M:NSN,7,3)+'-'+'$(M:NSN,10,4)'  
@ 1,42 SAY 'CAT' GET M:CAT  
@ 1,55 SAY 'CASE NUMEER:'  
@ 1,67 SAY 'M:CASE'  
@ 2,2 SAY '          COG' ' GET M:COG ;  
          PICTURE 'XX'  
@ 2,41 SAY 'SMIC' GET M:SM PICTURE 'AX'  
@ 3,2 SAY '          UIC' ' GET M:UIC  
@ 4,2 SAY '          REPCRT CONTROL'  
          ;  
          GET M:REPCON PICTURE 'XXXXXX999999'  
@ 5,2 SAY '          ACTICN POINT' ' GET M:ACTPT ;  
          PICTURE 'AXXXXX99999'  
@ 6,2 SAY '          NOMENCLATURE' ;  
          GET M:NOMEN PICTURE 'XXXXXXXXXXXXXXXXXXXXX'  
@ 7,2 SAY '          FSCM' ;  
          GET M:FSCM PICTURE 'XXXXXX'  
@ 8,2 SAY '          CONTRACT'  
          ;  
          GET M:NUM PICTURE 'XXXXXX99AXXXXXXXXXX'  
@ 9,2 SAY '          DOCUMENT'  
          ;  
          GET M:DOCNO PICTURE 'XXXXXX99999999'  
@ 10,2 SAY '          QUANTITY DEFICIENT' ;  
          GET M:QTYDEF PICTURE '999999'  
@ 11,2 SAY '          UNIT OF ISSUE' ;  
          GET M:UI PICTURE 'XX'  
@ 12,2 SAY '          UNIT PRICE' ;  
          GET M:UPRC PICTURE '999999.99'  
@ 13,2 SAY '          ORIGIN' ;  
          GET M:ORG PICTURE 'XXX'  
@ 14,2 SAY '          9Q REGION CODE' ;  
          GET M:O9Q PICTURE 'X'  
@ 15,2 SAY '          SCREEN QUANTITY' ;  
          GET M:SCRQTY PICTURE '999999'  
@ 16,2 SAY '          SCREEN CODE' ;  
          GET M:SCR PICTURE 'XXX'  
@ 17,2 SAY '          TYPE DOCUMENT' ;  
          GET M:DOC PICTURE '9'  
@ 18,2 SAY '          VENDOR LIABILITY CODE' ;  
          GET M:VLC PICTURE 'A'  
@ 19,2 SAY '          CREDIT CODE' ;  
          GET M:CR PICTURE 'A'  
@ 20,2 SAY '          TYPE DEFECT' ;  
          GET M:DEF PICTURE '99'
```

CLEAR GETS

STORE ' ' TO U:REPLY

@ 22,10 SAY ' ENTER <N> TO SKIP '

@ 23,30 SAY ' GET U:REPLY

READ

IF !(U:REPLY) = 'N'

STORE F TO U:FIRSTPG

STORE F TO U:CONT1

STORE T TO U:CONT2

IF U:FILE = 'CPEN'


```

        STORE '1C' TO M:TYPE
    ELSE
        STORE '3C' TO M:TYPE
    ENDIF
ELSE
    STORE T TO U:FIRSTPG
ENDIF
@ 22,10 SAY '
@ 23,10 SAY '

```

***** SKIP FIRST PAGE OF UPDATE IF REPLY WAS <N>

```

DO WHILE U:FIRSTPG
    STORE T TO U:CAT
    DO WHILE U:CAT
        @ 1,42 SAY 'CAT ' GET M:CAT PICTURE '9'
        READ
        IF M:CAT = '1' .OR. M:CAT = '2'
            STORE F TO U:CAT
        ELSE
            @ 23,20 SAY ' 1 OR 2 ONLY'
        ENDIF
    ENDDO U:CAT
    @ 23,20 SAY '
    RELEASE U:CAT

```

```

STORE T TO U:COG1
STORE T TO U:COG2
DC WHILE U:CCG1 .OR. U:COG2
    DO WHILE U:COG1
        @ 2,35 SAY ' ' GET M:COG PICTURE '9X'
        READ
        STORE ! (M:COG) TO M:COG
        IF $(M:COG,2,1) = ' '
            @ 23,20 SAY ' NO BLANKS IN 2D ' ;
                +' POSITION '
        ELSE
            STORE F TO U:COG1
        ENDIF
    ENDDO <U:COG1>
    @ 23,20 SAY '
        +'

```

***** CHECKS THAT COG IS VALID IN CURRENT COG TABLE... MUST
 ***** BE VALID TO CONTINUE

```

        USE D:COG INDEX D:COGS
        FIND &M:COG
        IF # = 0
            @ 23,20 SAY ' COG INVALID - ENTER ' ;
                +' CORRECTED ENTRY '
        ELSE
            STORE F TO U:COG2
        ENDIF
    ENDDO <U:COG1 & U:COG2>
    RELEASE U:CCG1, U:COG2
    @ 23,20 SAY '
        +'

```

```

IF M:COG = '1H' .OR. M:COG = '2H' .OR. M:COG = '7H'
    STORE T TO U:SMIC1
    DO WHILE U:SMIC1
        @ 2,45 SAY ' ' GET M:SM PICTURE 'AX'
        READ
        STORE ! (M:SM) TO M:SM
        IF $(M:SM,1,1) = 'X' .OR. $(M:SM,1,1) = 'L'
            STORE F TO U:SMIC1
        ENDIF
    ENDDO U:SMIC1

```



```

ELSE
    @ 23,30 SAY 'X OR L ONLY'
ENDIF
ENDDO <U:SMIC1>
ENDIF
RELEASE U:SMIC1
@ 23,30 SAY '

STORE T TO U:UIC1
DO WHILE U:UIC1
    @ 3,35 SAY ' ' GET M:UIC PICTURE 'AXXXXX'
    READ
    IF $(M:UIC,1,1) = ' ' .OR. $(M:UIC,2,1) = ' ' .OR. ;
        $(M:UIC,3,1) = ' ' .OR. $(M:UIC,4,1) = ' ' .OR. ;
        $(M:UIC,5,1) = ' ' .OR. $(M:UIC,6,1) = ' ' ;
            @ 23,20 SAY ' NO BLANKS ALLOWED IN UIC'
        ELSE
            STORE F TO U:UIC1
        ENDIF
    ENDDO <U:UIC1>
    @ 23,20 SAY '
RELEASE U:UIC1

@ 4,35 SAY ' ' GET M:REPCON PICTURE 'XXXXXX99999'
READ

STORE T TO U:ACTPTT
DO WHILE U:ACTPTT
    @ 5,35 SAY ' ' GET M:ACTPT PICTURE 'AXXXXX99999'
    READ
    IF M:ACTPT = '
        @ 23,30 SAY 'MAY NOT BE BLANK'
    ELSE
        STORE F TO U:ACTPTT
    ENDIF
    ENDDO <U:ACTPTT>
    @ 23,30 SAY '

STORE T TO U:NOMEN
DO WHILE U:NCMEN
    @ 6,35 SAY ' ' GET M:NOMEN PICTURE ;
        'XXXXXXXXXXXXXXXXXXXXX'
    READ
    IF $(M:NCMEN,1,1) = ' ' ;
        .OR. $(M:NOMEN,2,1) = ' ' ;
        .OR. $(M:NOMEN,3,1) = ' ' ;
            @ 23,30 SAY ' NO BLANKS IN FIRST 3' ;
                + ' POSITIONS'
        ELSE
            STORE F TO U:NOMEN
        ENDIF
    ENDDO <U:NOMEN>
    @ 23,30 SAY '
RELEASE U:NCMEN

```

***** INFUT FSCM

```

@ 7,35 SAY ' ' GET M:FSCM PICTURE 'XXXXXX'

```

***** INFUT CONTRACT NUMBER

```

@ 8,35 SAY ' ' GET M:NUM PICTURE 'XXXXXX99AXXXXXXXXX'

```

***** INFUT DOCUMENT NUMBER

```

STORE T TO U:UIC1
STORE T TO U:PREPT
STORE T TO U:DCCT
DO WHILE U:DOCT .OR. U:UIC1

```



```

@ 9,35 SAY ' 'GET M:DOCNO PICTURE 'XXXXXX99999999'
READ
IF M:DOCNO = ' '
    STORE F TO U:DOCT
    STORE F TO U:UICT
ELSE
    IF $(M:DCCNO,1,1) = ' ' .OR. $(M:DOCNO,2,1) = ' ' ;
        .OR. $(M:DOCNO,3,1) = ' ' ;
        .OR. $(M:DOCNO,4,1) = ' ' ;
        .OR. $(M:DOCNO,5,1) = ' ' ;
        @ 23,20 SAY ' NO BLANKS ALLOWED IN UIC'
    ELSE
        STORE F TO U:DOCT
    ENDIF
    IF $(M:DCCNO,12,3) > '366' ;
        .OR. $(M:DOCNO,12,3) = ' ' ;
        .OR. $(M:DOCNO,11,4) = ' ' ;
        @ 23,50 SAY 'PREP DATE OUT OF RANGE '
    ELSE
        STORE F TO U:UICT
    ENDIF
ENDIF <ALL BLANKS>
ENDDO <U:DOCT .AND. U:UICT>
RELEASE U:UICT,U:DOCT
@ 23,20 SAY '

```

***** DOCUMENT NUMBER END

```

STORE T TO U:QTYDEF
DO WHILE U:QTYDEF
    @ 10,35 SAY ' ' GET M:QTYDEF PICTURE '999999'
    READ
    IF M:QTYDEF < 1 .OR. M:QTYDEF > 999999
        @ 23,20 SAY ' DEFICIENT # OUT OF RANGE'
    ELSE
        STORE F TO U:QTYDEF
    ENDIF
ENDDO U:QTYDEF
RELEASE U:QTYDEF
@ 23,25 SAY '

```

```

STORE T TO U:UI
DO WHILE U:UI
    @ 11,35 SAY ' ' GET M:UI PICTURE 'AA'
    READ
    IF $(M:UI,1,1) = ' ' .OR. $(M:UI,2,1) = ' '
        @ 23,30 SAY ' NO BLANKS'
    ELSE
        STORE F TO U:UI
    ENDIF
ENDDO U:UI
@ 23,30 SAY '
RELEASE U:UI

```

```

STORE T TO U:EPRC
STORE T TO U:UPRC
DO WHILE U:UPRC .OR. U:EPRC
    DO WHILE U:UPRC
        @ 12,35 SAY ' 'GET M:UPRC PICTURE '999999.99'
        READ
        IF M:UPRC < .01 .OR. M:UPRC > 999999.99
            @ 23,30 SAY ' AMOUNT OUT OF RANGE '
        ELSE
            STORE F TO U:UPRC
        ENDIF
    ENDDO U:UPRC
    @ 23,30 SAY '
    STORE (M:UPRC * M:QTYDEF) TO M:EPRC
    STORE T TO U:EPRC

```



```

DO WHILE U:EPRC
  @ 12,50 SAY 'EXT PRICE $'
  @ 12,61 SAY M:EPRC PICTURE '999999999.99'
  IF M:EPRC >= 100000000
    @ 23,30 SAY ' EXTENDED PRICE OUT OF RANGE'
  ELSE
    STORE F TO U:EPRC
  ENDIF
ENDDO <U:EPRC>
ENDEO <U:UPRC & U:EPRC>
RELEASE U:UPRC,U:EPRC

```

```

STORE T TO U:ORG
DO WHILE U:CRG
  @ 13,35 SAY ' ' GET M:ORG PICTURE 'AAX'
  READ
  IF $(M:CRG,1,1) = ' ' .OR. $(M:ORG,2,1) = ' '
    @ 23,20 SAY ' FIRST 2 POSITIONS MAY NOT';
    + ' CONTAIN BLANKS'
  ELSE
    STORE F TO U:ORG
  ENDIF
ENDDO <U:ORG>
@ 23,20 SAY '
RELEASE U:ORG

```

```

IF M:COG = '9C'
  STORE T TO U:9Q
  DO WHILE U:9Q
    @ 14,35 SAY ' ' GET M:09Q PICTURE 'X'
    READ
    IF M:09Q = '2' .OR. M:09Q = '4' ;
      .OR. M:09Q = '5' .OR. M:09Q = '7';
      .OR. M:09Q = '9' .OR. M:09Q = ' '
      STORE F TO U:9Q
    ELSE
      @ 23,30 SAY ' OUT OF RANGE'
    ENDIF
  ENDDO <U:9Q>
  @ 23,30 SAY '
  RELEASE U:9Q
ENDIF <M:COG = 9Q>

```

```

@ 15,35 SAY ' ' GET M:SCRQTY PICTURE '999999'
READ

```

```

@ 16,35 SAY ' ' GET M:SCR PICTURE 'XXX'
READ

```

```

STORE T TO U:DOC
DO WHILE U:LCC

```

```

  @ 17,35 SAY ' ' GET M:DOC PICTURE '9'
  READ
  IF M:DOC < '1' .OR. M:DOC > '7'
    @ 23,30 SAY ' 1 THROUGH 7 ONLY'
  ELSE
    STORE F TO U:DOC
  ENDIF
ENDDO <U:DOC>
@ 23,30 SAY '
RELEASE U:DCC

```

```

@ 18,35 SAY ' ' GET M:VLC PICTURE 'A'

```

```

@ 19,35 SAY ' ' GET M:CR PICTURE 'A'

```

```

@ 20,35 SAY ' ' GET M:DEF PICTURE '99'

```



```

READ
STORE T TO U:END
DO WHILE U:END

  STORE ' ' TO U:REPLY
  @ 22,10 SAY '<CHOOSE> 1- CONTINUE 2- CHANGE';
  @ 23,30 SAY '3- EXIT'
  @ 23,30 SAY 'GET U:REPLY PICTURE '9'
  READ
  IF U:REPLY <> '1' .AND. U:REPLY <> '2' .AND.;
    U:REPLY <> '3'
    @ 23,10 SAY 'ANSWER WITH A 1 - 2 - 3 .ONLY'
  ELSE
    STORE F TO U:END
  ENDIF
ENDDO <U:END>
@ 23,10 SAY '

IF U:REPLY = '2'
  STORE T TO U:FIRSTPG
  @ 22,10 SAY '
  @ 23,10 SAY '
ELSE
  IF U:REPLY = '3'

    STORE F TO U:FIRSTPG
    STORE F TO U:CONT1
    STORE F TO U:CONT2
    STORE F TO U:CONT3
    IF U:FILE = 'OPEN'
      STORE '1G' TO M:TYPE
    ELSE
      STORE '3G' TO M:TYPE
    ENDIF
    DO C:XDBHNDLR
    RELEASE ALL EXCEPT C:*
    RETURN
  ELSE

    STORE F TO U:FIRSTPG
    STORE T TO U:CONT2
    STORE F TO U:CONT1
    IF U:FILE = 'OPEN'
      STORE '1C' TO M:TYPE
    ELSE
      STORE '3C' TO M:TYPE
    ENDIF

  ENDIF
ENDIF
ENDIF
ENDDO <U:FIRSTPG>
ERASE
RELEASE U:END,U:COUNT,U:FIRSTPG

***** CHOICE ABOVE ALLOWS ANALYST TO ABANDON OR TO PCST
***** CHANGES MADE THUS FAR

ENDDO <U:CONT1>
RELEASE U:CCNT1

IF U:REPLY <> '3'
  @ 10,20 SAY 'RECCRD BEING PARTIALLY UPDATED'
  @ 13,20 SAY 'PLEASE STANDBY'

***** WRITE DATA TO CPEN1/CLOSE1 AND RELEASE UNNECESSARY
***** VARIAELES BEFORE READING OPEN2/CLOSE2 FOR FURTHER
***** UPDATE INFORMATION

```



```

DC C:XDEHNDLR
RELEASE M:COG,M:CAT,M:NOMEN,M:UIC,M:UI,M:QTYDEF,M:UPRC,;
      M:EPFC,M:CRG,M:DOC,M:DOCN0,M:DATES,M:REPCON,;
      M:FSCM,M:NUM,M:CF,M:SCR,M:SM
RELEASE M:O9Q,M:DEF,M:VLC,M:ACTPT,SCRQTY,M:DDATE

```

```

IF U:FILE = 'OPEN'
  STORE '2E' TC M:TYPE
ELSE
  STORE '4E' TC M:TYPE
ENDIF
DC C:XDEHNDLR

```

ENDIF

DO WHILE U:CONT2

```

STORE $(M:DITEM,1,7) TO U:TYPE
STORE $(M:DITEM,8,6) TO U:SERNO
STORE $(M:DITEM,14,10) TO U:HASSY
STORE $(M:DITEM,24,12) TO U:SASSY
STORE ' ' TO M:OVER

```

***** DISPLAY FOR SECCND SCREEN OF UPDATE PROGRAM

ERASE

```

@ 1,2 SAY ' NSN:'
@ 1,16 SAY $(M:NSN,1,4)+'-'+'$(M:NSN,5,2)+'-'+'$(M:NSN,7,3);
      +'-'+'$(M:NSN,10,4)
@ 1,55 SAY 'CASE NUMBER:'
@ 1,67 SAY M:CASE
@ 2,2 SAY ' MFG. PART NUMBER ' GET M:MFG ;
      PICTURE 'XXXXXXXXXXXXXXXXX'
@ 3,2 SAY ' SERIAL/LOT/BATCH ' GET M:LOT ;
      PICTURE 'XXXXXXXXXX'
@ 4,2 SAY ' ITEM N OR O ' GET M:ITEM;
      PICTURE 'A'
@ 5,2 SAY ' DATE MFG/REP/OVHL ' GET M:OVER;
      PICTURE '99999'
@ 6,2 SAY ' OPN TIME AT FAILURE ' GET M:OTF ;
      PICTURE 'AXXXX'
@ 7,2 SAY ' GOV FURNISHED MATL ' GET M:GOV ;
      PICTURE 'A'
@ 8,2 SAY ' QUANTITY: RECV/INSP/STK 'GET M:QTYREC;
      PICTURE '999999'
@ 8,43 SAY ' GET M:CTYINS PICTURE '9999999'
@ 8,50 SAY ' GET M:CTYSTK PICTURE '9999999'
@ 9,2 SAY ' TYPE/MODEL/SERIES ' GET U:TYPE ;
      PICTURE 'XXXXXXXXX'
@ 10,2 SAY ' SERIAL NUMBER ' GET U:SERNO;
      PICTURE 'XXXXXXXXX'
@ 11,2 SAY ' NEXT HIGHER ASSY ' GET U:HASSY;
      PICTURE 'XXXXXXXXXXXXX'
@ 12,2 SAY ' SUE-ASSEMBLY ' GET U:SASSY;
      PICTURE 'XXXXXXXXXXXXX'
@ 13,2 SAY ' ESTIMATED CORRECTION COST ' GET M:CCOST;
      PICTURE '99999999.99'
@ 14,2 SAY ' WORK UNIT CODE ' GET M:WUC ;
      PICTURE 'XXXXXXXXX'
@ 15,2 SAY ' DEFECT VERIF - N/O/U/Y ' GET M:DEFV;
      PICTURE 'A'
@ 16,2 SAY ' DEFECT RESP - C/N/S/U/X ' GET M:DEFR ;
      PICTURE 'A'
@ 17,2 SAY ' STAIUS CODE ' GET ;
      M:STATUSC PICTURE 'AA'
@ 18,2 SAY ' CAUSE CODE ' GET ;
      M:CAUSEC PICTURE 'A'
@ 19,2 SAY ' ACTION/DISP -H/I/D/R/O ' GET ;

```



```

M:ACTDISP PICTURE 'X'
@ 20,2 SAY ' WAFRANTY ' GET M:WNTY PICTURE 'A'
@ 20,19 SAY 'COST CODE ' GET M:COSTC PICTURE 'A'
@ 20,30 SAY ' ACTION CODE' GET M:ACTTKN PICTURE 'AAA'
@ 21,2 SAY ' RETURN CODE ' GET M:RETC ;
PICTURE '9'
CLEAR GETS

```

```

@ 22,10 SAY ' ENTER <N> TO SKIP '
@ 23,30 SAY ' ' GET U:REPLY
READ
IF ! (U:REPLY) = 'N'
    STORE F TO U:PAGE2
    STCRE F TO U:CONT2
    STORE T TO U:CONT3
ELSE
    STORE T TO U:PAGE2
ENDIF
@ 22,10 SAY '
@ 23,10 SAY '
DO WHILE U:PAGE2

```

***** INPUT MANUFACTURERS PART NUMBER

```

@ 2,35 SAY ' ' GET M:MFG PICTURE 'XXXXXXXXXXXXXXXXXXXXX'
@ 3,35 SAY ' ' GET M:LOT PICTURE 'XXXXXXXXXX'

STCRE T TO U:ITEM
DO WHILE U:ITEM

    @ 4,35 SAY ' ' GET M:ITEM PICTURE 'A'
    READ
    IF M:ITEM = 'N' .OR. M:ITEM = 'O' ;
        .OR. M:ITEM = ' ' ;
        STCRE F TO U:ITEM
    ELSE
        @ 23,30 SAY ' USE N OR O '
    ENDIF
ENDDO <U:ITEM>
@ 23,30 SAY '
RELEASE U:ITEM
IF M:ITEM <> ' '

```

***** THE NEXT FIVE LINES CALCULATE EARLIEST YEAR TO ALLOW
***** FOR OVERHAUL ENTRY

```

STORE $(C:JULIAN,1,2) TO TEMP1
STCRE VAL(TEMP1) TO TEMP1A
STORE VAL('10') TO LOW
STCRE TEMP1A-LOW TO TEMP2
STORE STR(TEMP2,2) TO U:TENYRS
RELEASE TEMP1,TEMP1A,TEMP2,LOW

STORE T TO U:OVER
DO WHILE U:CVER
    @ 5,35 SAY ' ' GET M:OVER PICTURE '99999'
    READ
    IF $(M:CVER,3,3) > '365' ;
        .OR. $(M:OVER,1,2) < U:TENYRS ;
        .OR. $(M:OVER,1,2) > U:ULIMIT
        @ 23,30 SAY 'DATE OUT OF RANGE'
    ELSE
        STORE F TO U:OVER
    ENDIF
ENDDO <U:OVER>
ENDIF
RELEASE U:OVER,U:TENYRS

```



```

STORE T TO U:OTF
DO WHILE U:OTF
  @ 6,35 SAY ' ' GET M:OTF PICTURE 'A9999'
  READ
  IF M:OTF = ' '
    STORE F TO U:OTF
  ELSE
    IF $(M:OTF,1,1) = 'N'.OR.$(M:OTF,1,1) = 'C';
      .AND. $(M:OTF,2,4) > '0000'
      STORE F TO U:OTF
    ELSE
      @ 23,30 SAY ' USE N OR O AND THEN
        TIME (A9999) '
    ENDIF
  ENDIF
ENDDO <U:OTF>
@ 23,30 SAY '
RELEASE U:OTF

STORE T TO U:GOV
DO WHILE U:GOV
  @ 7,35 SAY ' ' GET M:GOV PICTURE 'A'
  READ
  IF M:GOV = ' ' .OR. M:GOV = 'Y'.OR. M:GOV = 'N'
    STORE F TO U:GOV
  ELSE
    @ 23,30 SAY 'USE EITHER Y OR N'
  ENDIF
ENDDO <U:GOV>
@ 23,30 SAY '
RELEASE U:GOV

STORE T TO U:QTYRECT
DO WHILE U:QTYRECT
  @ 8,35 SAY ' ' GET M:QTYREC PICTURE '999999'
  READ
  IF M:QTYREC < 0 .OR. M:QTYREC > 999999
    @ 23,30 SAY 'OUT OF RANGE'
  ELSE
    STORE F TO U:QTYRECT
  ENDIF
ENDDO
@ 23,30 SAY '
RELEASE U:QTYRECT
STORE T TO U:QTYINS
DO WHILE U:QTYINS
  @ 8,43 SAY '/' GET M:QTYINS PICTURE '999999'
  READ
  IF M:QTYINS < 0 .OR. M:QTYINS > 999999
    @ 23,30 SAY ' OUT OF RANGE'
  ELSE
    STORE F TO U:QTYINS
  ENDIF
ENDDO <U:QTYINS>
@ 23,30 SAY '
RELEASE U:QTYINS

STORE T TO U:QTYSTK
DO WHILE U:QTYSTK
  @ 8,50 SAY '/' GET M:QTYSTK PICTURE '999999'
  READ
  IF M:QTYSTK < 0 .OR. M:QTYSTK > 999999
    @ 23,30 SAY 'IN STOCK # OUT OF RANGE'
  ELSE
    STORE F TO U:QTYSTK
  ENDIF

```



```

ENDDO U:<QTYSTK>
@ 23,30 SAY '
RELEASE U:QTYSTK

@ 9,35 SAY ' ' GET U:TYPE PICTURE 'XXXXXXX'

@ 10,35 SAY ' ' GET U:SERNO PICTURE 'XXXXXX'

@ 11,35 SAY ' ' GET U:HASSY PICTURE 'XXXXXXXXXXXX'

@ 12,35 SAY ' ' GET U:SASSY PICTURE 'XXXXXXXXXXXXXX'
READ

STCRE U:TYPE+U:SERNO+U:HASSY+U:SASSY TO M:DITEM

@ 13,35 SAY ' ' GET M:CCOST PICTURE '999999999.99'
READ

@ 14,35 SAY ' ' GET M:WUC PICTURE 'XXXXXXX'
READ

@ 15,35 SAY ' ' GET M:DEFV PICTURE 'A'

STORE T TO U:DEFV
DO WHILE U:DEFV
  @ 15,35 SAY ' ' GET M:DEFV PICTURE 'A'
  READ
  IF M:DEFV = 'N' .OR. M:DEFV = 'O' ;
    .OR. M:DEFV = 'U' .OR. M:DEFV = 'Y' ;
    .OR. M:DEFV = ' ' ;
    STORE F TO U:DEFV
  ELSE
    @ 23,30 SAY ' CORRECT CODE MUST BE ENTERED'
  ENDIF
ENDDO <U:DEFV>
@ 23,30 SAY '
RELEASE U:DEFV

STORE T TO U:DEFR
DO WHILE U:DEFR
  @ 16,35 SAY ' ' GET M:DEFR PICTURE 'A'
  READ
  IF M:DEFR = 'C' .OR. M:DEFR = 'N' ;
    .OR. M:DEFR = 'S' .OR. M:DEFR = 'U' ;
    .OR. M:DEFR = 'X' .OR. M:DEFR = ' ' ;
    STORE F TO U:DEFR
  ELSE
    @ 23,30 SAY 'CORRECT CODE MUST BE ENTERED'
  ENDIF
ENDDO <U:DEFR>
@ 23,30 SAY '
RELEASE U:DEFR

@ 17,35 SAY ' ' GET M:STATUSC PICTURE 'AA'
READ

@ 18,35 SAY ' ' GET M:CAUSEC PICTURE 'A'
READ

@ 19,35 SAY ' ' GET M:ACTDISP PICTURE 'X'
READ

STORE T TO U:WNTY
DO WHILE U:WNTY
  @ 20,2 SAY ' WARRANTY 'GET M:WNTY
  READ
  IF M:WNTY = 'Y' .OR. M:WNTY = 'N' ;
    OR. M:WNTY = 'U'
    STORE F TO U:WNTY

```



```

ELSE @ 23,30 SAY 'Y, U OR N ONLY'
ENDIF
ENDDO
@ 23,30 SAY '
RELEASE U:WNIY
@ 20,19 SAY 'COST CODE ' GET M:COSTC PICTURE 'A'
@ 20,30 SAY 'ACTION CODE' GET M:ACTTKN PICTURE 'AAA'
@ 21,35 SAY ' ' GET M:RETC PICTURE '9'
READ
STORE T TO U:END
DO WHILE U:END

```

```

STORE ' ' TO U:REPLY
@ 22,10 SAY ' <CHOOSE> 1- CONTINUE 2- CHANGE'
@ 23,35 SAY ' ' GET U:REPLY PICTURE '9'
READ
IF U:REPLY <> '1' .AND. U:REPLY <> '2'
@ 23,05 SAY 'ANSWER WITH A 1 OR 2 ONLY '
ELSE
STORE F TO U:END
ENDIF
ENDDO <U:END>
@ 23,05 SAY '

```

```

IF U:REPLY = '2'
STORE T TO U:PAGE2
@ 22,10 SAY '
@ 23,10 SAY '
ELSE
STORE F TO U:CONT2
STORE T TO U:CONT3
STORE F TO U:PAGE2
ENDIF

```

```

ENDDO <U:PAGE2>
ERASE
RELEASE U:TYPE, U:SERNO, U:SASSY, U:HASSY, U:END, U:PAGE2
ENDDO <U:CONT2>

```

***** START OF THE THIRD SCREEN FOR THE UPDATE PROGRAM

```

ERASE
DO WHILE U:CONT3

```

***** DISPLAY OF CASE DATA FROM OPEN2 DETAILS & REPLY

```

@ 1,2 SAY '
@ 1,17 SAY '$(M:NSN,1,4)+'-'+$(M:NSN,5,2)+'-';
+$(M:NSN,7,3)+'-'+$(M:NSN,10,4)
@ 1,55 SAY 'CASE NUMBER:
@ 1,68 SAY 'M:CASE
@ 5,2 SAY 'DETAILS OF DISCREPANCY - FIRST 2 LETTERS'
@ 6,10 SAY ' MUST BE WHERE DISCOVERED CODE'
@ 8,10 SAY ' GET M:DETAILS
@ 13,2 SAY 'REPLY FROM ITEM MANAGER'
@ 14,10 SAY ' GET M:REPLY
CLEAR GETS

```

```

STORE ' ' TO U:REPLY
@ 22,10 SAY ' ENTER <N> TO SKIP & UPDATE RECORD'
@ 23,30 SAY ' GET U:REPLY

```



```

READ
IF !(U:REPLY) = 'N'
    STORE F TO U:PAGE3
    STORE F TO U:CONT3
ELSE
    STORE T TO U:PAGE3
ENDIF
@ 22,10 SAY '
      + '

```

***** SKIP THIRD PAGE OF UPDATE IF REPLY WAS <N>

```

DC WHILE U:PAGE3

```

```

    STORE T TO U:DISCODE
    DO WHILE U:DISCODE

```

```

        @ 8,10 SAY ' ' GET M:DETAILS
        READ
        STORE $(M:DETAILS,1,2) TO M:DIS
        USE D:WHERE DIS INDEX'D:DISCODE
        FIND &M:DIS
        IF # = 0
            @ 23,30 SAY 'WHERE DISCOVERED CODE INCORRECT'
        ELSE
            STORE F TO U:DISCODE
        ENDIF
    ENDDO <U:DISCODE>
    @ 23,30 SAY '
    RELEASE U:DISCODE

```

```

@ 14,10 SAY ' ' GET M:REPLY

```

```

READ
STORE T TO U:END
DO WHILE U:END
    STORE ' ' TO U:REPLY
    @ 21,10 SAY ' ***** CHECK PREVIOUS';
    + ' ENTRIES ***** '
    @ 22,10 SAY ' <CHOOSE> 1- CONTINUE ';
    + ' 2- CHANGE '
    @ 23,35 SAY ' ' GET U:REPLY
    READ
    IF U:REPLY <> '1' .AND. U:REPLY <> '2'
        @ 23,10 SAY ' ANSWER WITH A 1 OR 2 ONLY'
    ELSE
        STORE F TO U:END
    ENDIF
ENDDO <U:END>

```

```

@ 23,10 SAY '
IF U:REPLY = '2'
    STORE T TO U:PAGE3
    @ 22,10 SAY '
    @ 23,10 SAY '
ELSE
    STORE F TO U:PAGE3
    STORE F TO U:CONT3
ENDIF

```

```

ENDDO <U:PAGE3>
RELEASE U:PAGE3,U:COUNT
ERASE

```

```

ENDDO <U:CONT3>
IF U:REPLY <> '3'
    @ 10,20 SAY 'YOUR CASE IS BEING UPDATED NOW'
    @ $+2,20 SAY ' PLEASE STANDBY'
ENDIF
IF U:REPLY <> '3'
    IF U:FILE = 'OPEN'
        STORE '2C' TO M:TYPE
    ENDIF

```



```
        ELSE
          STORE '4C' TO M:TYPE
        ENDIF
      DC C:XDBHNDIF
    ENDIF

    RELEASE U:CONT3,U:REPLY,U:END
    RELEASE ALL EXCEPT C:*
    STORE T TO U:UPDATE
  ENDDO <U:UPDATE>
RETURN

*****  END CF PROGRAM
```


VI. CASE CLOSING MODULE

```
*****
**
** Date: 18 December 1984
** Version: 1.0
** Module Name: CLOSEEC
** Module Purpose: Close Current Case
** Module Interface Definition
** Inputs: C:WHO, C:JULIAN
** Outputs: None
** Module Processing Narrative Description:
**
** Prompts the Analyst for the desired closing date
** to assign to the case and then for the case
** number. The database is searched and
** reads current values. Insures that there are
** transmittal and return dates assigned. If not
** then the case must be updated before closing.
** If dates are present, the credit code and vendor
** liability codes must be entered in response to
** the prompts. The case is then written to the
** CLOSE1 and CLOCSE2 Databases and is marked for
** deletion in the OPEN1 and OPEN2.
**
** Superordinate Modules: MENU1
** Subordinate Modules: XDBHNDLR
** Author: J.G. BOYNTON
**
*****
```

```
ERASE
STORE T TO CL:CLOSE
DO WHILE CL:CLOSE
TEXT
```

***** CLOSE CASE *****

This program enables you to

CLOSE A QDR CASE

1 - Continue

2 - Return to Menu

ENDTEXT

```
STORE ' ' TO CL:REPLY
@ 20,38 SAY ' ' GET CL:REPLY
READ
DO WHILE CL:REPLY <> '1' .AND. CL:REPLY <> '2'
    @ 23,20 SAY ' ANSWER WITH A 1 OR 2 ONLY'
    @ 20,38 SAY ' ' GET CL:REPLY
    READ
ENDDC <CL:REPLY>

ERASE
IF CL:REPLY = '2'
    RELEASE ALL EXCEPT C:*
```



```

RETURN
ENDIF

STORE ' ' TO LC:CLDATE
STORE ' ' TO M:CASE
STORE ' ' TO CL:VIC
STORE ' ' TO CL:CR

@ 10,25 SAY '***** CLOSE CASE *****'
STORE T TO CI:DATET
DO WHILE CL:DATET

@ 14,26 SAY 'CLOSING DATE MMDDYY ';
GET LC:CLDATE
READ
IF LC:CLDATE = ' '
RELEASE ALL EXCEPT C:*
RETURN
ENDIF

STORE $(LC:CLDATE,5,2) TO CL:TEMP1
STORE VAL(CL:TEMP1)-1 TO CL:LOWDATE
STORE STR(CL:LOWDATE,2) TO CL:LDATE

IF $(LC:CLDATE,1,2) <'01':
.OR. $(LC:CLDATE,1,2) >'12':
.OR. $(LC:CLDATE,3,2) <'01':
.OR. $(LC:CLDATE,3,2) >'31':
.OR. $(LC:CLDATE,5,2) >$(C:JULIAN,1,2)

ELSE @ 23,30 SAY ' DATE OUT OF RANGE'
STORE F TO CL:DATET
ENDIF
ENDDO <CL:DATET>
@ 23,30 SAY '
RELEASE CL:DATET,CL:LDATE,CL:LOWDATE,CL:TEMP1

STORE T TO CL:MCFE
DO WHILE CL:MORE

@ 10,25 SAY ' ***** CLOSE CASE ***** '
@ 14,26 SAY 'CLOSING DATE MMDDYY ';
GET LC:CLDATE
CLEAR GETS

STORE T TO CI:REPLY
DO WHILE CL:REPLY
@ 15,26 SAY 'CASE NUMBER '
GET M:CASE PICTURE '999999X!';

READ
USE D:OPEN1 INDEX D:OCASE1
FIND &M:CASE
IF # = 0
STORE ' ' TO CL:AGAIN
@ 20,22 SAY ' That Case Not In Open File'
@ 22,18 SAY ' 1-To Try Again 2-To Return';
+' To Menu'
@ 23,33 SAY ' ' GET CL:AGAIN
READ
IF CI:AGAIN <> '1'
RELEASE ALL EXCEPT C:*
RETURN
ENDIF
ELSE
IF .NOT. *
STORE F TO CL:REPLY

```



```

        STORE T TO CL:FILLED
    ENDIF
    IF *
        STORE ' ' TO CL:AGAIN
        @ 20,22 SAY '        That CASE Already';
        +' CLOSED'
        @ 22,22 SAY ' 1-To Try Again 2-To ';
        +'Return To Menu'
        @ 23,33 SAY ' ' GET CL:AGAIN
        READ
        IF CL:AGAIN<>'1'
            RELEASE ALL EXCEPT C:*
            RETURN
        ENDIF
    ENDIF

```

```

    ENDIF
ENDDO <CL:REPLY>
@ 20,22 SAY ' ;
+' ;
@ 22,18 SAY ' ;
+' ;
@ 23,33 SAY ' ;
+' ;

```

```

***** GO TO THE OPEN CASE FILE AND READ THE CURRENT VLC
***** AND CREDIT CODE

```

```

STORE M:CASE TO M:KEY
STORE '1E' TO M:TYPE
DO C:XDBHNDLR

```

```

STORE $(M:DATES,1,5) TO M:DDATE
STORE $(M:DATES,6,5) TO M:RDATE
STORE $(M:DATES,11,5) TO M:OPEN
STORE $(M:DATES,16,5) TO M:LDATE
STORE $(M:DATES,21,5) TO M:SCRDATE
STORE $(M:DATES,26,5) TO M:IRDATE
STORE $(M:DATES,31,5) TO M:RIMDAT
STORE $(M:DATES,36,5) TO M:CLOSE
STORE $(M:DATES,41,5) TO M:REOPEN

```

```

IF M:RDATE = ' ' .OR. M:OPEN = ' ' .OR. ;
M:LDATE = ' ' .OR. M:RIMDAT = ' ' ;
@ 20,20 SAY 'Key Date/Dates Are Missing.';
+' CASE may NOT'
@ 21,20 SAY 'Be Closed Until Update Is ';
+'Accmplished'
@ 23,20 SAY ' Press Any Key To Continue'
WAIT
ERASE
@ 10,25 SAY '***** Please Standby *****'

```

```

STORE '1G' TO M:TYPE
DO C:XEBHNDLR
STORE F TO CL:FILLED

```

```

ENDIF

```

```

***** IF THE CASE IS COMPLETE AND READY TO BE CLOSED

```

```

IF CL:FILLED

```

```

STORE T TO CL:VLCT
DO WHILE CL:VLCT

```

```

@ 16,26 SAY 'VENDOR LIABILITY CODE ' ;
GET CL:VLC PICTURE 'A'

```

```

READ
IF CL:VLC = ' '

```



```

                @ 23,30 SAY 'VENDOR CODE MAY NOT ' ;
                    + 'BE BLANK'
            ELSE
                STORE F TO CL:VLCT
            ENDIF
        ENDDO <CL:VLCT>
        @ 23,30 SAY '
            + '
        RELEASE CL:VLCT

        STORE T TO CL:CRT
        DO WHILE CL:CRT
            @ 17,26 SAY 'CREDIT CODE
                GET CL:CR
            READ
            IF CL:CR = '
                @ 23,30 SAY ' CREDIT CODE MAY NCT ' ;
                    + 'BE BLANK'
            ELSE
                STORE F TO CL:CRT
            ENDIF
        ENDDO <CL:CRT>
        @ 23,30 SAY '
            + '
        RELEASE CL:CRT

        STORE ' ' TO CL:REPLY

        @ 20,22 SAY '1 - CLOSE CASE 2 - CHANGE ' ;
            + '3 - EXIT'
        @ 22,40 GET CL:REPLY
        READ
        IF CL:REPLY = '3'
            RELEASE ALL EXCEPT C:*
            RETURN
        ENDIF
        IF CL:REPLY = '1'
            ERASE
            @ 12,30 SAY 'CASE NUMBER'
            @ 12,44 SAY M:CASE
            @ 14,31 SAY 'IS BEING CLOSED'
            @ 16,30 SAY ' PLEASE STANDBY'

```

***** TRANSLATE IC:CLDATE FROM MMDDYY TO JULIAN FORM

```

        STCFE VAL ($ (LC:CLDATE, 1, 2) ) TO V:MM
        STORE VAL ($ (IC:CLDATE, 3, 2) ) TO V:DD
        STCFE VAL ($ (LC:CLDATE, 5, 2) ) TO V:YY
        DO C:OJULIAN
        STCFE V:JULDATE TO M:CLOSE
        RELEASE ALL LIKE V:*
        STCFE M:CASE TO M:KEY

```

***** PUT CLOSING DATE INTO PROPER FORMAT FOR STORAGE

```

        STCFE $(M:DATES, 41, 5) TO M:REOPEN
        STCFE $(M:DATES, 1, 35) + M:CLOSE + ;
            M:REOPEN TO CL:DATES
        STCFE CL:DATES TO M:DATES
        STCFE CL:VLC TO M:VLC
        STCFE CL:CR TO M:CR
        STORE '1C' TO M:TYPE
        STCFE M:REC1 TO T:REC1

```

```

        DO C:XDBHNDLR

```

***** CREATE RECORD IN CLOSE1

```

        STORE '3F' TO M:TYPE

```



```

DO C:XDBHNDLR
STORE M:CASE TO T:CASE
RELEASE ALL LIKE M:*
STORE T:CASE TO M:CASE
USE D:OPEN1 INDEX D:OCASE1,D:ONSM
GOTC T:REC1
DELETE
STORE '2B' TO M:TYPE

STORE T:CASE TO M:KEY
DO C:XDBHNDLR

STORE # TC T:REC2
STORE '4F' TO M:TYPE

DO C:XDBHNDLR

STORE M:CASE TO T:CASE
RELEASE ALL LIKE M:*
STORE T:CASE TO M:CASE
USE D:OPEN2 INDEX D:OCASE2
GOTC T:REC2
DELETE

STORE F TO CL:ENTER
ERASE
ENDIF
@ 20,22 SAY ' ' ;
+ ' ' ;
@ 22,22 SAY ' ' ;
+ ' ' ;
ERASE
ENDIF <CL:FILLED>
ENDDC <CL:MOVE>
RELEASE ALL LIKE CL:*
RELEASE ALL LIKE M:*
RELEASE ALL LIKE T:*
STORE T TO CL:CLCSE
ENDDO <CL:CLOSE>

***** END OF PROGRAM

```


VII. DATA BASE HANDLER MODULE

```
*****  
**  
** DATE: 29 NOV 1983  
** VERSION: 1.0  
** MODULE NAME: XDBHNDLR  
** MODULE PURPOSE: TO PROVIDE ACCESS TO THE DATA BASE  
** RECORDS FOR READ AND UPDATE  
**
```

MODULE INTERFACE DEFINITION

```
**  
** INPUTS: M:CASE, M:COG, M:NSN, M:CAT, M:NOMEN,  
** M:UIC, M:UI, M:QTYDEF, M:UPRC, M:EPRC,  
** M:ORG, M:DOC, M:DOCNO, M:DATES, M:REPCON,  
** M:FSCM, M:TIME, M:WHO, M:NUM, M:CR, M:SCR,  
** M:SM, M:O9Q, M:DEF, M:VLC, M:ACTPT,  
** M:SCRQTY, M:REC1, M:QTYINS, M:QTYREC,  
** M:QTYSTK, M:DEFV, M:DEFR, M:ITEM, M:OVER,  
** M:OTF, M:GOV, M:DITEM, M:CCOST, M:WNTY,  
** M:WUC, M:DIS, M:DETAILS, M:REPLY,  
** M:ACTTKN, M:COSTC, M:STATUSC, M:CAUSEC,  
** M:RETC, M:ACTDISP, M:MFG, M:LOT, M:TYPE  
**
```

```
**  
** OUTPUTS: M:CASE, M:COG, M:NSN, M:CAT, M:NOMEN,  
** M:UIC, M:UI, M:QTYDEF, M:UPRC, M:EPRC,  
** M:ORG, M:DOC, M:DOCNO, M:DATES,  
** M:REPCON, M:FSCM, M:TIME, M:WHO, M:NUM,  
** M:CR, M:SCR, M:SM, M:O9Q, M:DEF, M:VLC,  
** M:ACTPT, M:SCRQTY, M:REC1, M:QTYINS,  
** M:QTYREC, M:QTYSTK, M:DEFV, M:DEFR,  
** M:ITEM, M:OVER, M:OTF, M:GOV, M:DITEM,  
** M:CCOST, M:WNTY, M:WUC, M:DIS, M:DETAILS,  
** M:REPLY, M:ACTTKN, M:COSTC, M:STATUSC,  
** M:CAUSEC, M:RETC, M:ACTDISP, M:MFG,  
** M:LOT, M:TYPE  
**
```

MODULE PROCESSING NARRATIVE DESCRIPTION:

```
**  
** ACCEPTS THE TRANSACTION TYPE CODE AND ACCESSES  
** THE DATA BASE (I.E. OPEN1, OPEN2, CLOSE1,  
** OR CLOSE2) WITH THE DESIRED OPERATION (I.E.  
** READ, READ/LCCK, WRITE/UNLOCK, UNLOCK). THE  
** OPERATION PERFORMED DEPENDS ON THE TYPE CODE  
** RECEIVED. THE MODULE WILL RETURN A TYPE CODE  
** TO INDICATE THE SUCCESS OR FAILURE OF THE  
** OPERATION.  
**
```

```
** SUPERORDINATE MODULES: XOPEN2, XUPDAT, CLOSREC  
** SUBORDINATE MODULES: NONE  
** AUTHCR: R. G. NICHOLS  
**
```

```
***** THE GENERAL OPERATION OF THE DATA BASE HANDLER  
***** IS BASED ON A CASE CONSTRUCT
```

```
***** M:TYPE IS THE SELECTION KEY THAT DETERMINES THE  
***** TRANSACTION TO PERFORM - THE FIRST DIGIT REPRESENTS  
***** THE FILE THAT IS TO BE USED AND THE SECOND DIGIT  
***** REPRESENTS THE TYPE OF ACTIVITY (I.E. READ ACCESS  
***** WITH NSN KEY, READ ACCESS WITH CASE KEY, READ/LCCK  
***** WRITE NEW RECCRD, WRITE UPDATE UNLCOK, RECORD UNLOCK
```



```

***** ETC.)
DO CASE
***** USE OPEN1 DATA BASE FILE
CASE $(M:TYPE,1,1) = '1'
IF $(M:TYPE,2,1) = 'A' .OR. $(M:TYPE,2,1) = 'B'
***** IF 'A' THEN ACCESS BY NSN
IF $(M:TYPE,2,1) = 'A'
STORE 'USE D:OPEN1 INDEX D:ONSN' TO H:USEFILE
ELSE
***** IF 'B' THEN ACCESS BY CASE
STORE 'USE D:OPEN1 INDEX D:OCASE1' TO H:USEFILE
ENDIF
***** USE INDIRECT FILE IDENTIFICATION TO SELECT USE FILE
&H:USEFILE
FIND &M:KEY
***** SEARCH FOR DESIRED RECORD. IF FOUND RETURN DATA
***** ELEMENTS AND SET M:TYPE TO 0 OTHERWISE SET M:TYPE
***** TC 9
IF # = 0
STORE '9' TO M:TYPE
RELEASE ALL LIKE H:*
RETURN
ELSE
STORE # TC M:REC 1
STORE CASE TO M:CASE
STORE COG TO M:COG
STORE NSN TO M:NSN
STORE CAT TO M:CAT
STORE NOMEN TO M:NOMEN
STORE UIC TO M:UIC
STORE UI TC M:UI
STORE QTYDEF TO M:QTYDEF
STORE UPRC TO M:UPRC
STORE EPRC TO M:EPRC
STORE ORG TC M:ORG
STORE DOC TO M:DOC
STORE DOCNC TO M:DOCNO
STORE DATES TO M:DATES
STORE REPCCN TO M:REPCCN
STORE FSCM TO M:FSCM
STORE TIME TO M:TIME
STORE WHO TO M:WHO
STORE NUM TO M:NUM
STORE CR TC M:CR
STORE SCR TO M:SCR
STORE SM TC M:SM
STORE O9Q TC M:O9Q
STORE DEF TO M:DEF
STORE VLC TO M:VLC
STORE ACTPT TO M:ACTPT
STORE SCRQTY TO M:SCRQTY
STORE '0' TO M:TYPE
RELEASE ALL LIKE H:*
RETURN
ENDIF
ELSE
IF $(M:TYPE,2,1) = 'H' .OR. $(M:TYPE,2,1) = 'I'

```



```

*****
***** READ RECORD SPECIFIED BY M:REC1 (RECORD NUMBER)
***** IF RECORD NOT FOUND RETURN M:TYPE = 9 OTHERWISE
***** RETURN THE RECCRD ELEMENTS

```

```

USE D:OPEN1 INDEX D:ONSN
GOTO M:REC1
IF # <> M:REC1
  STORE '9' TO M:TYPE
  RELEASE ALL LIKE H:*
  RETURN
ELSE

```

```

***** IF M:TYPE = I THEN SKIP TO NEXT RECORD AND READ

```

```

  IF $(M:TYPE,2,1) = 'I'
    SKIP
  ENDIF
  STORE # TO M:REC1
  STORE CASE TO M:CASE
  STORE CCG TO M:COG
  STORE NSN TO M:NSN
  STORE CAT TO M:CAT
  STORE NCMEN TC M:NOMEN
  STORE UIC TO M:UIC
  STORE UI TO M:UI
  STORE CIYDEF TO M:QTYDEF
  STORE UPRC TO M:UPRC
  STORE EPRC TO M:EPRC
  STORE CFG TO M:ORG
  STORE ECC TO M:DOC
  STORE ECCNO TO M:DOCNO
  STORE LATES TC M:DATES
  STORE REPCON TO M:REPCON
  STORE FSCM TO M:FSCM
  STORE TIME TO M:TIME
  STORE WHO TO M:WHO
  STORE NUM TO M:NUM
  STORE CR TO M:CR
  STORE SCR TO M:SCR
  STORE SM TO M:SM
  STORE C9Q TO M:O9Q
  STORE DEF TO M:DEF
  STORE VIC TO M:VLC
  STORE ACTPT TC M:ACTPT
  STORE SCRQTY TO M:SCRQTY
  STORE '0' TO M:TYPE
  RELEASE ALL LIKE H:*
  RETURN

```

```

  ENDIF
ELSE

```

```

***** THE FOLLOWING SECTION OF CODE UTILIZES A DELAY LOOP
***** AND A LOCKING MECHANISM TO ENSURE THAT ONLY ONE USER
***** IS WRITING TO A FILE AT ANY GIVEN TIME

```

```

  STORE T TC H:FAIL
  DO WHILE H:FAIL
    STORE 0 TO H:LOOPCNTR
    STORE 2 TO H:CNTR
    USE D:FILESTAT

```

```

***** WHILE OPEN1 IS BEING USED, ENTER DELAY LOOP

```

```

  DO WHILE OPEN1<>' '
    STORE H:CNTR-1 TO H:CNTR
    IF H:CNTR=0
      STORE 2 TO H:CNTR
      STORE H:LOOPCNTR+1 TO H:LOOPCNTR

```


ENDIF

***** IF IN DELAY LCCP A SHORT PERIOD OF TIME DISPLAY THE
***** FACT THAT THE FILE IS CURRENTLY IN USE

IF H:LOOPCNT=2
@ 23,16 SAY 'OPEN CASE FILE CURRENTLY IN USE';
+ ' - PLEASE STANDBY'
ENDIF

***** CLOSE OUT THE USE FILE THEN REOPEN IT TO CHECK LATEST
***** STATUS - THE FILE MUST FIRST BE CLOSED AND THEN
***** REOPENED TO CHECK LATEST STATUS

USE
USE D:FILESTAT

***** DELAY BEFORE TRYING AGAIN

STORE 1 TO H:DELAY
DO WHILE H:DELAY < 5
SICRE H:DELAY + 1 TO H:DELAY
ENDDC
ENDDO

***** IF FILE NOT IN USE, WRITE OUT YOUR LOCK INFORMATION

@ 23,16 SAY ' ' ;
+ ' ' ;
@ 23,16 SAY ' FILE LOCKED ' ;
+ ' ' ;
REPL OPEN1 WITH C:WHO
USE

***** IF TYPE C TRANSACTION - PERFORM A WRITE/UNLOCK

IF \$(M:TYPE,2,1) = 'C'
USE D:FILESTAT

***** VERIFY THAT YOU HAVE WRITE ACCESS TO THE DATA BASE
***** RESET THE LOCK ON THE RECORD AND WRITE OUT THE
***** UPDATED INFORMATION

IF CPEN1 = C:WHO
STORE ' ' TO M:TIME
USE D:OPEN1 INDEX D:ONSN, D:OCASE1
GCTO M:REC1
REPL CASE WITH !(M:CASE), COG WITH !(M:COG), NSN WITH ;
!(M:NSN), CAT WITH !(M:CAT), NOMEN WITH !(M:NOMEN), UIC ;
WITH !(M:UIC), UI WITH !(M:UI), QTYDEF WITH M:QTYDEF, UPRC ;
WITH M:UPRC, EPRC WITH M:EPRC, ORG WITH !(M:ORG), DOC WITH ;
!(M:DCC)
REPL DOCNO WITH !(M:DCCNO), DATES WITH !(M:DATES), REPCON ;
WITH !(M:REPCON), FSCM WITH !(M:FSCM), TIME WITH !(M:TIME), ;
WHO WITH !(M:WHO), NUM WITH !(M:NUM), CR WITH !(M:CR), SCR ;
WITH !(M:SCR), SM WITH !(M:SM), O9Q WITH !(M:O9Q)
REPL DEF WITH M:DEF, VLC WITH !(M:VLC), ACTPT WITH ;
!(M:ACTPT), SCRQTY WITH M:SCRQTY
STORE '0' TO M:TYPE
USE

***** UNLOCK THE DATA FILE FOR OTHERS TO WRITE

USE D:FILESTAT
REPL OPEN1 WITH ' '
USE
RELEASE ALL LIKE H:*
RETURN
ENDIF

ELSE

```
***** IF TYPE D THEN PERFORM READ/LOCK WITH NSN ACCESS KEY
      IF $(M:TYPE,2,1)='D' .OR. $(M:TYPE,2,1)='E'
      IF $(M:TYPE,2,1)='D'
      STORE 'USE D:OPEN1 INDEX D:CNSN, '
      + 'D:OCASE1' TO H:USEFILE
```

```
***** IF TYPE C THEN PERFORM READ/LOCK WITH CASE ACCESS KEY
      ELSE
      STORE 'USE D:OPEN1 INDEX D:OCASE1, '
      + ' D:ONSN' TO H:USEFILE
      ENDIF
      USE D:FILESTAT
```

```
***** CHECK TO SEE IF THE USER HAS THE FILE LOCKED FOR
***** WRITING
```

```
      IF OPEN1 = C:WHO
      &H:USEFILE
      FIND &M:KEY
```

```
***** CHECK TO SEE IF DESIRED RECORD EXISTS. IF SO LOCK
***** THE RECORD BY FILLING THE TIME STAMP AND RETURN THE
***** RECORDS CONTENTS
```

```
      IF # = 0
      STORE '9' TO M:TYPE
      USE D:FILESTAT
      REPLACE OPEN1 WITH ' '
      USE
      RELEASE ALL LIKE H:*
      RETURN
      ELSE
```

```
***** CHECK TO SEE IF THE RECORD HAS PREVIOUSLY BEEN
***** LOCKED FOR UPDATE - RETURN TYPE = 1 IF PREVIOUSLY
***** LOCKED OTHERWISE LOCK THE RECORD BY FILLING IN THE
***** TIMESTAMP AND READ THE RECORD
```

```
      IF TIME <> ' '
      STORE '1' TO M:TYPE
      RELEASE ALL LIKE H:*
      USE D:FILESTAT
      REPLACE OPEN1 WITH ' '
      USE
      RETURN
      ELSE
```

```
***** READ DATE/TIME FOR TIMESTAMP
```

```
      STORE TO H:DUMMY
      POKE 61440, 180, 44, 205, 33, ;
      137, 22, 13, 240, 137, 14, ;
      15, 240, 195
      SET CALL TO 61440
      CALL H:DUMMY
      STORE STR(PEEK(61456),2) TC ;
      H:HOURL
      STORE STR(PEEK(61455),2) TC ;
      H:MIN
      STORE STR(PEEK(61454),2) TC ;
      H:SEC
      IF $(H:HOURL,1,1) =
      STORE 0 +$(H:HOURL,2,1) ;
      TO H:HOURL
      ENDIF
```



```

IF $(H:MIN,1,1) =
  STORE 0+$(H:MIN,2,1) TO ;
  H:MIN
ENDIF
IF $(H:SEC,1,1) =
  STORE 0+$(H:SEC,2,1) TO ;
  H:SEC
ENDIF
STORE C:JULIAN+H:HOURL +H:MIN+;
H:SEC TO M:TIME
REPL TIME WITH M:TIME
STORE # TO M:REC1
STORE CASE TO M:CASE
STORE COG TO M:COG
STORE NSN TO M:NSN
STORE CAT TO M:CAT
STORE NOMEN TO M:NOMEN
STORE UIC TO M:UIC
STORE UI TO M:UI
STORE QTYDEF TO M:QTYDEF
STORE UPRC TO M:UPRC
STORE EPRC TO M:EPRC
STORE ORG TO M:ORG
STORE DOC TO M:DOC
STORE DOCNO TO M:DOCNO
STORE DATES TO M:DATES
STORE REPCON TO M:REPCON
STORE FSCM TO M:FSCM
STORE TIME TO M:TIME
STORE WHO TO M:WHO
STORE NUM TO M:NUM
STORE CR TO M:CR
STORE SCR TO M:SCR
STORE SM TO M:SM
STORE O9Q TO M:O9Q
STORE DEF TO M:DEF
STORE VLC TO M:VLC
STORE ACTPT TO M:ACTPT
STORE SCRQTY TO M:SCRQTY
STORE '0' TO M:TYPE
USE D:FILESTAT
REPL OPEN1 WITH ' '
USE
RELEASE ALL LIKE H:*
RETURN
ENDIF
ENDIF
ENDIF
ELSE

```

***** TYPE F WILL BE USED TO CREATE NEW RECORDS

```

IF $(M:TYPE,2,1) = 'F'
  USE D:FILESTAT

```

***** CHECK TO SEE IF THE USER HAS THE FILE LOCKED FOR
 ***** WRITING

```

* IF OPEN1 = C:WHO
  @ 23,25 SAY ' UPDATING CASE ';
  + 'FILE '
  USE D:OPEN1 INDEX D:OCASE1,;
  D:ONSN

```

***** IF NC CASE NUMEER HAS BEEN ASSIGNED BECAUSE OF A
 ***** PREVIOUS CASE, ASSIGN A NEW CASE NUMBER

```

IF M:CASE = ' '

```


***** CHECK FOR LAST CASE IN THE DATA BASE AND ASSIGN
***** NEXT AVAILABLE NUMBER

```
GOTO BOTTOM
STORE $(CASE,1,1) TO H:YR
STORE VAL$(CASE,2,5)+1 TO ;
H:SERIAL
IF H:SERIAL > 9999
STORE H:YR + ;
STR(H:SERIAL,5,0) TO M:CASE
ELSE
IF H:SERIAL > 999
STORE H:YR + '0' + ;
STR(H:SERIAL,4,0) TO ;
M:CASE
ELSE
IF H:SERIAL > 99
STORE H:YR + '00' + ;
STR(H:SERIAL,3,0) TO;
M:CASE
ELSE
IF H:SERIAL > 9
STORE H:YR+'000'+;
STR(H:SERIAL,2,0);
TO M:CASE
ELSE
STORE H:YR+'0000';
+STR(H:SERIAL,1,0);
TO M:CASE
ENDIF
ENDIF
ENDIF
ENDIF
ENDIF
@ 23,26 SAY 'CREATING NEW RECCRD';
+ ' - OPEN1'
```

***** CREATE NEW RECCRD AND FILL WITH DATA

```
APPEND BLANK
REPL CASE WITH !(M:CASE), COG WITH !(M:COG), NSN WITH ;
!(M:NSN), CAT WITH !(M:CAT), NOMEN WITH !(M:NOMEN), UIC WITH ;
!(M:UIC), UI WITH !(M:UI), QTYDEF WITH M:QTYDEF, UPRC WITH ;
M:UPRC, EPRC WITH M:EPRC, ORG WITH !(M:ORG), DOC WITH !(M:DOC)
REPL DOCNO WITH !(M:DCCNO), DATES WITH !(M:DATES), REPCON ;
WITH !(M:REPCON), FSCM WITH !(M:FSCM), TIME WITH !(M:TIME), ;
WHO WITH !(C:WHO), NUM WITH !(M:NUM), SM WITH !(M:SM), O9Q ;
WITH !(M:O9Q), DEF WITH M:DEF, ACTPT WITH !(M:ACTPT)
@ 23,26 SAY '
a 23,25 SAY '
'
```

```
STORE '0' TO M:TYPE
USE D:FILESTAT
REPL OPEN1 WITH ' '
USE
RELEASE ALL LIKE H:*
RETURN
ENDIF
ELSE
```

***** IF TYPE G THEN UNLOCK A PREVIOUSLY LOCKED RECORD
***** (NO UPDATE WILL TAKE PLACE)

```
IF $(M:TYPE,2,1) = 'G'
USE D:FILESTAT
```

***** CHECK TO SEE IF THE USER HAS THE FILE LOCKED FOR
***** WRITING

```
IF OPEN1 = C:WHO
```


USE D:OPEN1
GOTO M:REC1

***** CLEAR THE TIMESTAMP TO UNLOCK

```
                IF TIME = M:TIME  
                  REPL TIME WITH '  
                ENDIF  
                USE D:FILESTAT  
                REPL OPEN1 WITH '  
                USE  
                ENDIF  
                RELEASE ALL LIKE H:*\n                RETURN  
            ENDIF  
        ENDIF  
    ENDIF  
ENDDO  
ENDIF  
ENDIF
```

***** USE OPEN2 DATA BASE FILE

CASE \$(M:TYPE,1,1) = '2'

***** SINCE OPEN2 HAS A SINGLE KEY, BOTH TYPE A AND B
***** MAY BE USED FOR ACCESS

```
IF $(M:TYPE,2,1) = 'A' .OR. $(M:TYPE,2,1) = 'B'  
  USE D:OPEN2 INDEX D:OCASE2  
  FIND &M:KEY
```

***** FIND REQUESTED RECORD IF FOUND RETURN THE DATA
***** ELEMENTS AND TYPE = 0, OTHERWISE RETURN TYPE = 9

```
IF # = 0  
  STORE '9' TO M:TYPE  
  RELEASE ALL LIKE H:*\n  RETURN  
ELSE  
  STORE # TC M:REC1  
  STORE CASE TO M:CASE  
  STORE QTYINS TO M:QTYINS  
  STORE QTYREC TO M:QTYREC  
  STORE QTYSTK TO M:QTYSTK  
  STORE DEFV TO M:DEFV  
  STORE DEFR TO M:DEFR  
  STORE ITEM TO M:ITEM  
  STORE OVER TO M:OVER  
  STORE OTF TO M:OTF  
  STORE GOV TO M:GOV  
  STORE TIME TO M:TIME  
  STORE WHO TO M:WHO  
  STORE DITEM TO M:DITEM  
  STORE CCOST TO M:CCOST  
  STORE WNTY TO M:WNTY  
  STORE WUC TO M:WUC  
  STORE DIS TO M:DIS  
  STORE DETAILS TO M:DETAILS  
  STORE REPLY TO M:REPLY  
  STORE ACTTKN TO M:ACTTKN  
  STORE COSIC TO M:COSTC  
  STORE STATUSC TO M:STATUSC  
  STORE CAUSEC TO M:CAUSEC  
  STORE RETC TO M:RETC  
  STORE ACTDISP TO M:ACTDISP
```



```

DO WHILE OPEN2<>'
STORE H:CNTR-1 TO H:CNTR
IF H:CNTR=0
STORE 2 TO H:CNTR
STORE H:LOOPCNT+1 TO H:LOOPCNT
ENDIF
IF H:LOOPCNT=2
@ 23,16 SAY 'OPEN CASE FILE CURRENTLY IN';
+ ' USE - PLEASE STANDBY'
ENDIF

```

```

***** CLOSE AND REOPEN THE FILE STATS TO DETERMINE ANY
***** CHANGE IN FILE LOCKING STATUS

```

```

USE
USE D:FILESTAT

```

```

***** DELAY BEFORE NEXT ATTEMPT TO ACCESS THE DATA BASE

```

```

STORE 1 TO H:DELAY
DO WHILE H:DELAY < 5
STORE H:DELAY + 1 TO H:DELAY
ENDDC

```

```

ENDDO
@ 23,16 SAY ' ';
* @ 23,16 SAY ' FILE LOCKED ';
+ ' '

```

```

***** WRITE LOCK TO FILESTAT

```

```

REPL OPEN2 WITH C:WHO
USE

```

```

***** IF TYPE C PERFORM WRITE/UNLOCK

```

```

IF $(M:TYPE,2,1) = 'C'
USE D:FILESTAT

```

```

***** CHECK TO SEE IF USER HAS WRITE ACCESS TO THE DATA
***** BASE

```

```

IF OPEN2 = C:WHO
STORE ' ' TO M:TIME
USE D:OPEN2 INDEX D:OCASE2

```

```

***** WRITE UPDATE INFORMATION TO THE FILE

```

```

GOTO M:REC1
REPL CASE WITH !(M:CASE), QTYINS WITH M:QTYINS, QTYREC WITH ;
M:QTYREC, QTYSTK WITH M:QTYSTK, DEFR WITH !(M:DEFR), DEFR ;
WITH !(M:DEFR), ITEM WITH !(M:ITEM), OVER WITH !(M:OVER), ;
OTF WITH !(OTF), GOV WITH !(M:GOV), TIME WITH !(M:TIME)
REPL WHO WITH !(M:WHC), DITEM WITH !(M:DITEM), CCOST WITH ;
M:CCOST, WNTY WITH !(M:WNTY), WUC WITH !(M:WUC), DIS WITH ;
!(M:DIS), DETAILS WITH !(M:DETAILS), REPLY WITH !(M:REPLY), ;
ACTTKN WITH !(M:ACTTKN), COSTC WITH !(M:COSTC)
REPL STATUSC WITH !(M:STATUSC), CAUSEC WITH !(M:CAUSEC), ;
RETC WITH !(M:RETC), ACTDISP WITH !(M:ACTDISP), MFG WITH ;
!(M:MFG), LOT WITH !(M:LOT)
STORE '0' TO M:TYPE
USE

```

```

***** UNLOCK FILE FOR OTHERS USE

```

```

USE D:FILESTAT
REPL OPEN2 WITH ' '
USE

```



```

                RELEASE ALL LIKE H:*
                RETURN
            ENDIF
        ELSE

```

```

***** IF TYPE D OR E PERFORM READ/LOCK
                IF $(M:TYPE,2,1) = 'D' .OR. $(M:TYPE,2,1) = 'E'
                    USE D:FILESTAT

```

```

***** CHECK TO SEE IF USER HAS WRITE ACCESS TO THE DATA
***** BASE

```

```

                IF OPEN2 = C:WHO
                    USE D:OPEN2 INDEX D:OCASE2
                    FIND &M:KEY

```

```

***** IF DESIRED RECCRD FOUND VERIFY THAT RECORD IS NOT
***** CURRENTLY IN USE - IF NOT FOUND RETURN TYPE = 9

```

```

                IF # = 0
                    STORE '9' TO M:TYPE
                    USE D:FILESTAT
                    REPLACE OPEN2 WITH ' '
                    USE
                    RELEASE ALL LIKE H:*
                    RETURN
                ELSE

```

```

***** IF TIMESTAMP FILLED, RECORD IN USE - RETURN
***** TYPE = 1

```

```

                IF TIME <> ' '
                    STORE '1' TO M:TYPE
                    RELEASE ALL LIKE H:*
                    USE D:FILESTAT
                    REPLACE OPEN2 WITH ' '
                    USE
                    RETURN
                ELSE

```

```

***** LOAD TIME/DATE INTO TIMESTAMP AND READ THE RECORD

```

```

                    STORE          TO H:DUMMY
                    POKE 61440, 180, 44, 205, 33,;
                        137, 22, 13, 240, 137, 14,;
                        15, 240, 195
                    SET CALL TO 61440
                    CALL H:DUMMY
                    STORE STR(PEEK(61456),2) TO ;
                        H:HOURL
                    STORE STR(PEEK(61455),2) TO ;
                        H:MIN
                    STORE STR(PEEK(61454),2) TO ;
                        H:SEC
                    IF $(H:HOURL,1,1) =
                        STORE 0 +$(H:HOURL,2,1) ;
                            TO H:HOURL
                    ENDIF
                    IF $(H:MIN,1,1) =
                        STORE 0 +$(H:MIN,2,1) TO ;
                            H:MIN
                    ENDIF
                    IF $(H:SEC,1,1) =
                        STORE 0 +$(H:SEC,2,1) TO ;
                            H:SEC
                    ENDIF
                    STORE C:JULIAN+H:HOURL +H:MIN+;
                        H:SEC TO M:TIME

```



```

REPL TIME WITH M:TIME
STORE # TO M:REC1
STORE CASE TO M:CASE
STORE QTYINS TO M:QTYINS
STORE QTYREC TO M:QTYREC
STORE QTYSTK TO M:QTYSTK
STORE DEFV TO M:DEFV
STORE DEFR TO M:DEFR
STORE ITEM TO M:ITEM
STORE OVER TO M:OVER
STORE OTF TO M:OTF
STORE GOV TO M:GOV
STORE TIME TO M:TIME
STORE WHO TO M:WHO
STORE DITEM TO M:DITEM
STORE CCOST TO M:CCOST
STORE WNTY TO M:WNTY
STORE WUC TO M:WUC
STORE DIS TO M:DIS
STORE DETAILS TO M:DETAILS
STORE REPLY TO M:REPLY
STORE ACTTKN TO M:ACTTKN
STORE COSTC TO M:COSTC
STORE STATUSC TO M:STATUSC
STORE CAUSEC TO M:CAUSEC
STORE RETC TO M:RETC
STORE ACTDISP TO M:ACTDISP
STORE MFG TO M:MFG
STORE LOT TO M:LOT
STORE '0' TO M:TYPE

```

***** UNLOCK DATA BASE FOR OTHER USERS

```

USE D:FILESTAT
REPL OPEN2 WITH ' '
USE
RELEASE ALL LIKE H:*
RETURN

```

ENDIF

ENDIF

ENDIF

ELSE

***** TYPE F CREATES NEW RECORDS

```

IF $(M:TYPE,2,1) = 'F'
USE D:FILESTAT

```

***** CHECK TO SEE IF USER HAS WRITE ACCESS TO THE DATA
***** BASE

```

IF OPEN2 = C:WHO
@ 23,25 SAY ' UPDATING CASE';
+ ' FILE
USE D:OPEN2 INDEX D:OCASE2

```

***** ADD THE NEW RECORD AND ENTER DATA

```

APPEND BLANK
REPL CASE WITH !(M:CASE), QTYINS WITH M:QTYINS, QTYREC WITH;
M:QTYREC, QTYSTK WITH M:QTYSTK, DEFV WITH !(M:DEFV), DEFR ;
WITH !(M:DEFR), ITEM WITH !(M:ITEM), OVER WITH !(M:OVER), ;
OTF WITH !(OTF), GOV WITH !(M:GOV), TIME WITH !(M:TIME)
REPL WHO WITH !(C:WHO), DITEM WITH !(M:DITEM), WNTY WITH ;
!(M:WNTY), WUC WITH !(M:WUC), DIS WITH !(M:DIS), DETAILS ;
WITH !(M:DETAILS), ACTDISP WITH !(M:ACTDISP), MFG WITH ;
!(M:MFG), LOT WITH !(M:LOT)

```

STORE '0' TO M:TYPE

***** UNLOCK DATA BASE FOR CTHERS

```
USE D:FILESTAT
REPL OPEN2 WITH ' '
USE
@ 23,25 SAY ' '
RELEASE ALL LIKE H:*
RETURN
ENDIF
ELSE
```

***** IF TYPE G PERFCRM UNLOCK (NO UPDATE)

```
IF $(M:TYPE,2,1) = 'G'
USE D:FILESTAT
IF OPEN2 = C:WHO
USE D:OPEN2
GOTO M:REC1
IF TIME = M:TIME
REPL TIME WITH ' '
ENDIF
```

***** UNLOCK DATA BASE FOR OTHERS

```
USE D:FILESTAT
REPL OPEN2 WITH ' '
USE
ENDIF
RELEASE ALL LIKE H:*
RETURN
ENDIF
ENDIF
ENDIF
ENDIF
ENDDO
ENDIF
ENDIF
```

***** TYPES 3 AND 4 DEAL WITH CLOSE1 AND CLOSE2
***** THE METHODOLOGY USED FOR THESE TYPES IS THE SAME
***** AS FOR THE OPEN1 AND OPEN2 DATA BASE FILES
***** ONLY DIFFERENCES WILL BE NOTED BELOW SINCE THE
***** BASIC COMMENTS ARE THE SAME AS ABOVE

***** TYPE 3 USES CLCSE1

```
CASE $(M:TYPE,1,1) = '3'
IF $(M:TYPE,2,1) = 'A' .OR. $(M:TYPE,2,1) = 'B'
IF $(M:TYPE,2,1) = 'A'
STORE 'USE D:CLOSE1 INDEX D:CNSN' TO H:USEFILE
ELSE
STORE 'USE D:CLOSE1 INDEX D:CCASE1' TO H:USEFILE
ENDIF
&H:USEFILE
FIND &M:KEY
IF # = 0
STORE '9' TO M:TYPE
RELEASE ALL LIKE H:*
RETURN
ELSE
STORE # TC M:REC1
STORE CASE TO M:CASE
STORE COG TO M:COG
STORE NSN TO M:NSN
STORE CAT TO M:CAT
STORE NOMEN TO M:NOMEN
```



```

STORE UIC TO M:UIC
STORE UI TC M:UI
STORE QTYDEF TO M:QTYDEF
STORE UPRC TO M:UPRC
STORE EPRC TO M:EPRC
STORE ORG TC M:ORG
STORE DOC TO M:DOC
STORE DOCNO TO M:DOCNO
STORE DATES TO M:DATES
STORE REPCCN TO M:REPCON
STORE FSCM TO M:FSCM
STORE TIME TO M:TIME
STORE WHO TO M:WHO
STORE NUM TO M:NUM
STORE CR TC M:CR
STORE SCR TO M:SCR
STORE SM TC M:SM
STORE O9Q TO M:O9Q
STORE DEF TO M:DEF
STORE VLC TO M:VIC
STORE ACTPT TO M:ACTPT
STORE SCRQTY TO M:SCRQTY
STORE '0' TO M:TYPE
RELEASE ALL LIKE H:*
RETURN
ENDIF
ELSE
IF $(M:TYPE,2,1) = 'H' .OR. $(M:TYPE,2,1) = 'I'
USE D:CLCSE1
GOTO M:REC1
IF # <> M:REC1
STORE '9' TO M:TYPE
RELEASE ALL LIKE H:*
RETURN
ELSE
IF $(M:TYPE,2,1) = 'I'
SKIIF
ENDIF
STORE # TO M:REC1
STORE CASE TO M:CASE
STORE CCG TO M:COG
STORE NSN TO M:NSN
STORE CAT TO M:CAT
STORE NCMEN TC M:NOMEN
STORE UIC TO M:UIC
STORE UI TO M:UI
STORE QTYDEF TO M:QTYDEF
STORE UPRC TO M:UPRC
STORE EPRC TO M:EPRC
STORE CRG TO M:ORG
STORE DCC TO M:DOC
STORE DCCNO TC M:DOCNO
STORE DATES TO M:DATES
STORE REPCON TO M:REPCON
STORE FSCM TO M:FSCM
STORE TIME TO M:TIME
STORE WHO TO M:WHO
STORE NUM TO M:NUM
STORE CR TO M:CR
STORE SCR TO M:SCR
STORE SM TO M:SM
STORE CSQ TO M:O9Q
STORE DEF TO M:DEF
STORE VIC TO M:VIC
STORE ACTPT TO M:ACTPT
STORE SCRQTY TO M:SCRQTY
STORE '0' TO M:TYPE
RELEASE ALL LIKE H:*
RETURN

```



```

ENDIF
ELSE
STORE T TC H:FAIL
DO WHILE H:FAIL
STORE 0 TO H:LOOPCNT
STORE 2 TO H:CNTR
USE D:FILESTAT
DO WHILE CLOSE1<>' '
STORE H:CNTR-1 TO H:CNTR
IF H:CNTR=0
STORE 2 TO H:CNTR
STORE H:LOOPCNT+1 TO H:LOOPCNT
ENDIF
IF H:LOOPCNT=2
@ 23,16 SAY 'CLOSE CASE FILE CURRENTLY IN';
+ ' USE - PLEASE STANDBY'

ENDIF
USE
USE D:FILESTAT
STORE 1 TO H:DELAY
DO WHILE H:DELAY < 5
SICRE H:DELAY + 1 TO H:DELAY
ENDDO
ENDDO
@ 23,16 SAY ' ';
* @ 23,16 SAY ' FILE LOCKED BY ' ;
+ 'CLOSE1
REPL C:CLOSE1 WITH C:WHO
USE
IF $(M:TYPE,2,1) = 'C'
USE D:FILESTAT
IF CLOSE1 = C:WHO
STORE ' ' TO M:TIME
USE D:CLOSE1 INDEX D:CNSN, D:CCASE1
GOTO M:REC1
REPL CASE WITH !(M:CASE), COG WITH !(M:COG), NSN WITH ;
!(M:NSN), CAT WITH !(M:CAT), NOMEN WITH !(M:NOMEN), UIC WITH ;
!(M:UIC), UI WITH !(M:UI), QTYDEF WITH M:QTYDEF, UPRC WITH ;
M:UPRC, EFRC WITH M:EFRC, ORG WITH !(M:ORG), DOC WITH !(M:DOC)
REPL DOCNO WITH !(M:DCCNO), DATES WITH !(M:DATES), REPCON ;
WITH !(M:RECON), FSCM WITH !(M:FSCM), TIME WITH !(M:TIME);;
WHO WITH !(M:WHO), NUM WITH !(M:NUM), CR WITH !(M:CR), SCR ;
WITH !(M:SCR), SM WITH !(M:SM), O9Q WITH !(M:O9Q)
REPL DEF WITH M:DEF, VLC WITH !(M:VLC), ACTPT WITH ;
!(M:ACTPT), SCROTQTY WITH M:SCROTQTY
STORE '0' TO M:TYPE
USE
USE D:FILESTAT
REPL CLOSE1 WITH ' '
USE
RELEASE ALL LIKE H:*
RETURN
ENDIF
ELSE
IF $(M:TYPE,2,1)='D' .OR. $(M:TYPE,2,1)='E'
IF $(M:TYPE,2,1) = 'D'
STORE 'USE D:CLOSE1 INDEX D:CNSN, ' ;
+ 'D:CCASE1' TO H:USEFILE
ELSE
STORE 'USE D:CLOSE1 INDEX D:CCASE1, ' ;
+ ' D:CNSN' TO H:USEFILE
ENDIF
USE D:FILESTAT
IF CLOSE1 = C:WHO
&H:USEFILE
FIND &M:KEY
IF # = 0
STORE '9' TO M:TYPE

```



```

USE D:FILESTAT
REPLACE CLOSE1 WITH ' '
USE
RELEASE ALL LIKE H:*
RETURN
ELSE
IF TIME <> ' '
STORE '1' TO M:TYPE
RELEASE ALL LIKE H:*
USE D:FILESTAT
REPLACE CLOSE1 WITH ' '
USE
RETURN
ELSE
STORE TO H:DUMMY
POKE 61440, 180, 44, 205, 33,;
    137, 22, 13, 240, 137, 14,;
    15, 240, 195
SET CALL TO 61440
CALL H:DUMMY
STORE STR(PEEK(61456), 2) TC ;
    H:HOURL
STORE STR(PEEK(61455), 2) TC ;
    H:MIN
STORE STR(PEEK(61454), 2) TC ;
    H:SEC
IF $(H:HOURL, 1, 1) =
    STORE 0 + $(H:HOURL, 2, 1) ;
    TO H:HOURL
ENDIF
IF $(H:MIN, 1, 1) =
    STORE 0 + $(H:MIN, 2, 1) TO ;
    H:MIN
ENDIF
IF $(H:SEC, 1, 1) =
    STORE 0 + $(H:SEC, 2, 1) TO ;
    H:SEC
ENDIF
STORE C:JULIAN+H:HOURL +H:MIN+;
    H:SEC TO M:TIME
REPL TIME WITH M:TIME
STORE # TO M:REC1
STORE CASE TO M:CASE
STORE COG TO M:COG
STORE NSN TO M:NSN
STORE CAT TO M:CAT
STORE NOMEN TO M:NOMEN
STORE UIC TO M:UIC
STORE UI TO M:UI
STORE QTYDEF TO M:QTYDEF
STORE UPRC TO M:UPRC
STORE EPRC TO M:EPRC
STORE ORG TO M:ORG
STORE DOC TO M:DOC
STORE DOCNO TO M:DOCNO
STORE DATES TO M:DATES
STORE REPCON TO M:REPCON
STORE FSCM TO M:FSCM
STORE TIME TO M:TIME
STORE WHO TO M:WHO
STORE NUM TO M:NUM
STORE CR TO M:CR
STORE SCR TO M:SCR
STORE SM TO M:SM
STORE O9Q TO M:O9Q
STORE DEF TO M:DEF
STORE VLC TO M:VLC
STORE ACTPT TO M:ACTPT
STORE SCRQTY TO M:SCRQTY

```



```

        STORE '0' TO M:TYPE
        USE D:FILESTAT
        REPL CLOSE1 WITH '
        USE
        RELEASE ALL LIKE H:*
        RETURN
    ENDF
  ENDF
ELSE
***** FOR TYPE F, A NEW RECORD IS CREATED BY TRANSFERRING
***** DATA FROM THE OPEN FILE TO THE CLOSE FILE

        IF $(M:TYPE,2,1) = 'F'
            USE D:FILESTAT
            IF CLOSE1 = C:WHO
                @ 23,25 SAY '          UPDATING CASE';
                + ' FILE
            USE D:CLOSE1 INDEX D:CCASE1, ;
                D:CNSN
            APPEND BLANK

REPL CASE WITH !(M:CASE), COG WITH !(M:COG), NSN WITH ;
!(M:NSN), CAT WITH !(M:CAT), NOMEN WITH !(M:NOMEN), UIC WITH ;
!(M:UIC), UI WITH !(M:UI), QTYDEF WITH M:QTYDEF, UPRC WITH ;
M:UPRC, EPRC WITH M:EEFC, ORG WITH !(M:ORG), DOC WITH !(M:DOC)
REPL DOCNO WITH !(M:DCCNO), DATES WITH !(M:DATES), REPCON ;
WITH !(M:REPCON), FSCM WITH !(M:FSCM), TIME WITH !(M:TIME), ;
WHO WITH !(C:WHO), NUM WITH !(M:NUM), SM WITH !(M:SM), O9Q ;
WITH !(M:O9Q), DEF WITH M:DEF, ACTPT WITH !(M:ACTPT)

        STORE '0' TO M:TYPE
        USE D:FILESTAT
        REPL CLOSE1 WITH '
        USE
        @ 23,25 SAY '
        + '
        RELEASE ALL LIKE H:*
        RETURN
    ENDF
  ELSE
        IF $(M:TYPE,2,1) = 'G'
            USE D:FILESTAT
            IF CLOSE1 = C:WHO
                USE D:CLOSE1
                GOTO M:REC1
                IF TIME = M:TIME
                    REPL TIME WITH '
                ENDF
            USE D:FILESTAT
            REPL CLOSE1 WITH '
            USE
            ENDF
            RELEASE ALL LIKE H:*
            RETURN
        ENDF
    ENDF
  ENDF
  ENDDO
ENDIF
ENDIF

```

***** TYPE 4 USES CICSE2 DATA BASE FILE

```

CASE $(M:TYPE,1,1) = '4'
  IF $(M:TYPE,2,1) = 'A' .OR. $(M:TYPE,2,1) = 'B'
    USE D:CLOSE2 INDEX D:CCASE2
  
```



```

FIND &M:KEY
IF # = 0
  STORE '9' TO M:TYPE
  RELEASE ALL LIKE H:*
  RETURN
ELSE
  STORE # TC M:REC 1
  STORE CASE TO M:CASE
  STORE QTYINS TO M:QTYINS
  STORE QTYREC TO M:QTYREC
  STORE QTYSTK TO M:QTYSTK
  STORE DEFV TO M:DEFV
  STORE DEFR TO M:DEFR
  STORE ITEM TO M:ITEM
  STORE OVER TO M:OVER
  STORE OTF TO M:OTF
  STORE GOV TO M:GOV
  STORE TIME TO M:TIME
  STORE WHO TO M:WHO
  STORE DITEM TO M:DITEM
  STORE CCOST TO M:CCOST
  STORE WNTY TO M:WNTY
  STORE WUC TO M:WUC
  STORE DIS TO M:DIS
  STORE DETAILS TO M:DETAILS
  STORE REPLY TO M:REPLY
  STORE ACTTKN TO M:ACTTKN
  STORE COSTC TO M:COSTC
  STORE STATUSC TO M:STATUSC
  STORE CAUSEC TO M:CAUSEC
  STORE RETC TO M:RETC
  STORE ACTDISP TO M:ACTDISP
  STORE MFG TO M:MFG
  STORE LOT TO M:LOT
  STORE '0' TO M:TYPE
  RELEASE ALL LIKE H:*
  RETURN
ENDIF
ELSE
IF $(M:TYPE,2,1) = 'H' .OR. $(M:TYPE,2,1) = 'I'
  USE D:CLCSE2
  GOTO M:REC1
  IF # <> M:REC1
    STORE '9' TO M:TYPE
    RELEASE ALL LIKE H:*
    RETURN
  ELSE
    IF $(M:TYPE,2,1) = 'I'
      SKIP
    ENDIF
    STORE # TO M:REC1
    STORE CASE TO M:CASE
    STORE QTYINS TO M:QTYINS
    STORE QTYREC TO M:QTYREC
    STORE QTYSTK TO M:QTYSTK
    STORE DEFV TO M:DEFV
    STORE DEFR TO M:DEFR
    STORE ITEM TO M:ITEM
    STORE OVER TO M:OVER
    STORE OTF TO M:OTF
    STORE GOV TO M:GOV
    STORE TIME TO M:TIME
    STORE WHO TO M:WHO
    STORE DITEM TO M:DITEM
    STORE CCOST TO M:CCOST
    STORE WNTY TO M:WNTY
    STORE WUC TO M:WUC
    STORE DIS TO M:DIS
    STORE DETAILS TO M:DETAILS

```



```

STORE REPLY TO M:REPLY
STORE ACTTKN TO M:ACTTKN
STORE CCSTC TO M:COSTC
STORE STATUSC TO M:STATUSC
STORE CAUSEC TO M:CAUSEC
STORE RETC TO M:RETC
STORE ACTDISP TO M:ACTDISP
STORE MFG TO M:MFG
STORE LOT TO M:LOT
STORE '0' TO M:TYPE
RELEASE ALL LIKE H:*
RETURN
ENDIF
ELSE
STORE T TO H:FAIL
DO WHILE H:FAIL
STORE 0 TO H:LOOPCNT
STORE 2 TO H:CNTR
USE D:FILESTAT
DO WHILE CLOSE2<>' '
STORE H:CNTR-1 TO H:CNTR
IF H:CNTR=0
STORE 2 TO H:CNTR
STORE H:LOOPCNT+1 TO H:LOOPCNT
ENDIF
IF H:LOOPCNT=2
@ 23,16 SAY 'CLOSE CASE FILE CURRENTLY IN USE ';
+ '- PLEASE STANDBY'
ENDIF
USE
USE D:FILESTAT
STORE 1 TO H:DELAY
DO WHILE H:DELAY < 5
SICRE H:DELAY + 1 TO H:DELAY
ENDDO
ENDDO
@ 23,16 SAY '
REPL CICSE2 WITH C:WHO
USE
IF $(M:TYPE,2,1) = 'C'
USE D:FILESTAT
IF CLOSE2 = C:WHO
STORE ' ' TO M:TIME
USE D:CLOSE2 INDEX D:CCASE2
GOTO M:REC1
REPL CASE WITH !(M:CASE), QTYINS WITH M:QTYINS, QTYREC ;
WITH M:QTYREC, QTYSTK WITH M:QTYSTK, DEFR WITH !(M:DEFR), ;
DEFR WITH !(M:DEFR), ITEM WITH !(M:ITEM), OVER WITH ;
!(M:OVER), OTF WITH !(CTF), GOV WITH !(M:GOV), TIME WITH ;
!(M:TIME)
REPL WHO WITH !(M:WHC), DITEM WITH !(M:DITEM), CCOST WITH ;
M:CCOST, WNTY WITH !(M:WNTY), WUC WITH !(M:WUC), DIS WITH ;
!(M:DIS), DETAILS WITH !(M:DETAILS), REPLY WITH !(M:REPLY), ;
ACTTKN WITH !(M:ACTTKN), COSTC WITH !(M:COSTC)
REPL STATUSC WITH !(M:STATUSC), CAUSEC WITH !(M:CAUSEC), ;
RETC WITH !(M:RETC), ACTDISP WITH !(M:ACTDISP), MFG WITH ;
!(M:MFG), LOT WITH !(M:LOT)
STORE '0' TO M:TYPE
USE
USE D:FILESTAT
REPL CLOSE2 WITH ' '
USE
RELEASE ALL LIKE H:*
RETURN
ENDIF
ELSE
IF $(M:TYPE,2,1) = 'D' .OR. $(M:TYPE,2,1) = 'E'
USE D:FILESTAT
IF CLOSE2 = C:WHO

```



```

USE D:CLOSE2 INDEX D:CCASE2
FIND &M:KEY
IF # = 0
  STORE '9' TO M:TYPE
  USE D:FILESTAT
  REPLACE CLOSE2 WITH ' '
  USE
  RELEASE ALL LIKE H:*
  RETURN
ELSE
  IF TIME <> ' '
    STORE '1' TO M:TYPE
    RELEASE ALL LIKE H:*
    USE D:FILESTAT
    REPLACE CLOSE2 WITH ' '
    USE
    RETURN
  ELSE
    STORE      TO H:DUMMY
    PCKE 61440, 180, 44, 205, 33, 137, ;
        22, 13, 240, 137, 14, 15, 240, 195
    SET CALL TO 61440
    CALL H:DUMMY
    STORE STR(PEEK(61456), 2) TO H:HOURL
    STORE STR(PEEK(61455), 2) TO H:MIN
    STORE STR(PEEK(61454), 2) TO H:SEC
    IF $(H:HOURL, 1, 1) =
      STORE 0 + $(H:HOURL, 2, 1) TO H:HOURL
    ENDIF
    IF $(H:MIN, 1, 1) =
      STORE 0 + $(H:MIN, 2, 1) TO H:MIN
    ENDIF
    IF $(H:SEC, 1, 1) =
      STORE 0 + $(H:SEC, 2, 1) TO H:SEC
    ENDIF
    STORE C:JULIAN+H:HOURL+H:MIN+H:SEC;
      TO M:TIME
    REPL TIME WITH M:TIME
    STORE # TO M:REC1
    STORE CASE TO M:CASE
    STORE QTYINS TO M:QTYINS
    STORE QTYREC TO M:QTYREC
    STORE QTYSTK TO M:QTYSTK
    STORE DEFR TO M:DEFR
    STORE DEFV TO M:DEFV
    STORE ITEM TO M:ITEM
    STORE OVER TO M:OVER
    STORE OTF TO M:OTF
    STORE GOV TO M:GOV
    STORE TIME TO M:TIME
    STORE WHO TO M:WHO
    STORE DITEM TO M:DITEM
    STORE CCOST TO M:CCOST
    STORE WNTY TO M:WNTY
    STORE WUC TO M:WUC
    STORE DIS TO M:DIS
    STORE DETAILS TO M:DETAILS
    STORE REPLY TO M:REPLY
    STORE ACTTKN TO M:ACTTKN
    STORE COSTC TO M:COSTC
    STORE STATUSC TO M:STATUSC
    STORE CAUSEC TO M:CAUSEC
    STORE RETC TO M:RETC
    STORE ACTDISP TO M:ACTDISP
    STORE MFG TO M:MFG
    STORE LOT TO M:LOT
    STORE '0' TO M:TYPE
    USE D:FILESTAT
    REPL CLOSE2 WITH ' '

```



```

USE
RELEASE ALL LIKE H:*
RETURN
ENDIF
ENDIF
ENDIF
ELSE

```

```

***** FOR TYPE F, A NEW RECORD IS CREATED BY TRANSFERRING
***** DATA FROM THE CPEN FILE TO THE CLOSE FILE

```

```

IF $(M:TYPE,2,1) = 'F'
USE D:FILESTAT
IF CLOSE2 = C:WHO
USE D:OPEN2 INDEX D:OCASE2
FIND &M:CASE
IF # <> 0
+ ' FILE
USE D:CLOSE2 INDEX D:CCASE2
APPEND BLANK

```

```

REPL CASE WITH !(M:CASE), OTYINS WITH M:OTYINS, OTYREC WITH;
M:OTYREC, OTYSTK WITH M:OTYSTK, DEFR WITH !(M:DEFR), DEFR;
WITH !(M:DEFR), ITEM WITH !(M:ITEM), OVER WITH !(M:OVER),;
OTF WITH !(OTF), GOV WITH !(M:GOV), TIME WITH !(M:TIME)
REPL WHO WITH !(C:WHO), DITEM WITH !(M:DITEM), WNTY WITH;
!(M:WNTY), WUC WITH !(M:WUC), DIS WITH !(M:DIS), DETAILS;
WITH !(M:DETAILS), ACTDISP WITH !(M:ACTDISP), MFG WITH;
!(M:MFG), LOT WITH !(M:LOT)

```

```

STORE '0' TO M:TYPE
USE D:FILESTAT
REPL CLOSE2 WITH ' '
USE
@ 23,25 SAY ' '
+ '
RELEASE ALL LIKE H:*
RETURN

```

```

ENDIF
ENDIF
ELSE
IF $(M:TYPE,2,1) = 'G'
USE D:FILESTAT
IF CLOSE2 = C:WHO
USE D:CLOSE2
GOTO M:REC1
IF TIME = M:TIME
REPL TIME WITH ' '
ENDIF
USE D:FILESTAT
REPL CLOSE2 WITH ' '
USE
ENDIF
RELEASE ALL LIKE H:*
RETURN

```

```

ENDIF
ENDIF
ENDIF
ENDDO
ENDIF
ENDIF
ENDCASE
RETURN

```

```

***** END OF PROGRAM

```


VIII. SUPERVISOR MENU

```
*****
**
** DATE: 11 JANUARY 1984
** VERSICN: 1.0
** MODULE NAME: SUPMENU1
** MODULE PURPOSE: PROVIDE MENU FOR SUPERVISOR TO
** ACCESS QDR SYSTEM PROGRAMS
**
** MODULE INTERFACE DEFINITION
** INPUTS: C:WHO. C:JULIAN
** OUTPUTS: NONE
** MODULE PROCESSING NARRATIVE DESCRIPTION:
**
** THE SUPERVISOR IS PRESENTED WITH A MENU OF
** ALL PROSESSING CAPABILITIES AVAILABLE. AFTER
** ONE IS CHCSEN, THE MODULE THEN CALLS THE
** DESIGNATED PROGRAM INTO ACTION OR LOGS THE
** SUPERVISOR OUT OF THE QDR SYSTEM
**
** SUPERORDINATE MODULES: LOGON
** SUBORDINATE MODUIES: MENU1,C-REASGN,UTILMENU,SUPRPTS,
** SUPRPT2
** AUTHCR: J.G. BCYNTON
**
*****
```

```
ERASE
STORE T TO V:CONTINUE
DO WHILE V:CONTINUE
STORE ' ' TC V:CHOICE
TEXT
```

WELCOME TO THE QDR SUPERVISOR MENU

- 1 - MAIN MENU PROCESSING
- 2 - CASE REASSIGNMENT
- 3 - ANALYST WORKLOAD STATISTICS
- 4 - UTILITY PROGRAMS
- 5 - REPORT GENERATION
- 6 - YEAR END PROCESSING
- 7 - SORTED LISTINGS
- 8 - EXIT FROM THE SYSTEM

ENTER YOUR CHOICE

```
ENDTEXT
@ 21,30 GET V:CHOICE
READ
?
IF V:CHOICE >= 1 .AND. V:CHOICE <= 8
?
DO CASE
CASE V:CHOICE= 1
RELEASE ALL LIKE V:*
DO C:MENU1.PRG
```



```

CASE V:CHOICE= 2
  RELEASE ALL LIKE V:*
  DO C:C-REASGN.PRG
CASE V:CHOICE= 3
  RELEASE ALL LIKE V:*
  DO C:STATGEN.PRG
CASE V:CHOICE= 4
  RELEASE ALL LIKE V:*
  DO C:UTILMENU.PRG
CASE V:CHOICE= 5
  RELEASE ALL LIKE V:*
  DO C:SUERPTS.PRG
CASE V:CHOICE= 6
  RELEASE ALL LIKE V:*
  DO C:YEAREND.PRG
CASE V:CHOICE=7
  RELEASE ALL LIKE V:*
  DO C:SUFPRPT2.PRG
CASE V:CHOICE= 8
  RELEASE ALL EXCEPT C:*
  ERASE
  RETURN
ENDCASE
ERASE
STORE T TO V:CONTINUE
STORE ' ' TC V:CHOICE
ELSE
  '?' < PLEASE ANSWER WITH 1 - 8 ONLY >'
ENDIF <V:CHOICE>
ENDDO <V:CONTINUE>
***** END OF PROGRAM

```


IX. SUPERVISOR UTILITY MENU

```
*****
**
** Date: 16 January 1984
** Version: 1.0
** Module Name: UTIIMENU
** Module Purpose: Provide Supervisor with menu of
**                  utility programs available to him.
**
** Module Interface Definition
**   Inputs: C:WHO, C:JULIAN
**   Outputs: None
** Module Processing Narrative Description:
**
**   Displays menu of all utility programs avail-
**   able to the supervisor. Calls the appropriate
**   program after user selection. Additional level
**   of security required for packing Data Base.
**
** Superordinate Modules: SUPMENU1
** Subordinate Modules: ANALYST,PASS,COGUPDT,ADDRUPDT,
**                      STATGEN,COGCNT,UTILNDX,DBPACK
** Author: J.G. BOYNTON & R.G. NICHOLS
**
*****
```

STORE T TO U:CONTINUE

***** Display Options Available To The Operator

```
DO WHILE U:CONTINUE
ERASE
@ 6,25 SAY '***** Utility Processing *****'
@ 9,29 SAY '1 - Analyst Update'
@ $+1,29 SAY '2 - Password Processing'
@ $+1,29 SAY '3 - COG Update'
@ $+1,29 SAY '4 - Address File Update'
@ $+1,29 SAY '5 - Internal Statistics Update'
@ $+1,29 SAY '6 - Cog Count'
@ $+1,29 SAY '7 - Re-Index Index Files For The System'
@ $+1,29 SAY '8 - Clean Up The Database (Pack)'
@ $+1,29 SAY '9 - Exit To Supervisor Menu'
STORE ' ' TO U:REPLY
@ 19,40 GET U:REPLY PICTURE '9'
READ
```

***** Accept Menu Selection

```
DO WHILE U:REPLY < '1' .OR. U:REPLY > '9'
@ 23,32 SAY 'Enter 1 - 9 Only' + CHR(7)
@ 19,40 GET U:REPLY PICTURE '9'
READ
ENDDO
```

***** Call Routine Necessary To Perform Desired Function

```
DO CASE
CASE U:REPLY = '9'
```



```
                WAIT
            ENDIF
        ENDIF
    ENDIF
CASE U:REPLY = '8'
DO C:DBPACK
ENDCASE
ENDDO

***** END OF PROGRAM
```


X. USER REPORT MENU

```
*****  
**  
** Date: 11 January 1984 **  
** Version: 1.0 **  
** Module Name: RPTMENU **  
** Module Purpose: Allow analyst to receive a listing **  
**                  of his current open cases. **  
** Module Interface Definition **  
**   Inputs: C:WHO, C:JULIAN **  
**   Outputs: None **  
** Module Processing Narrative Description: **  
**  
**   Menu is provided in order to select a listing **  
**   of open cases that belong to the Analyst **  
**   making the request. If report listing is **  
**   chosen, then module OCASERPT is called. Exit **  
**   is to return to MENU1. **  
** Superordinate Modules: MENU1 **  
** Subordinate Modules: OCASERPT **  
** Author: J.G. BOYNTON **  
*****  
STORE T TO C:TRUE  
DO WHILE C:TRUE  
ERASE  
*  
STORE ' ' TC V:CHOICE  
TEXT
```

***** QUERY REPORT AVAILABLE *****

- 1 - Openfile by Case
- 2 - Exit

Enter Your Choice

```
ENDTEXT  
@ 19,38 GET V:CHOICE  
READ  
*  
IF V:CHOICE >= '1' .AND. V:CHOICE <= '2'  
*  
  DC CASE  
  CASE V:CHOICE = '1'  
    DO C:OCASERPT  
  CASE V:CHOICE = '2'  
    STORE F TC C:TRUE  
  ENDCASE  
  
ELSE  
  ?  
  ?  
  ' < Please Answer With a 1 - 2 ONLY >'
```


?
?

PRESS ANY KEY TO CONTINUE'

WAIT

ENDIF <V:CHOICE>
ENDDO <C:TRUE>
RELEASE ALL LIKE V: *
RELEASE C:TRUE

***** END OF PROGRAM

XI. SUPERVISOR REPORT MENU

```
*****  
**  
** Date: 15 January 1984  
** Version: 1.0  
** Module Name: SUPRPTS  
** Module Purpose: Provide Supervisor a menu of  
** available reports.  
**  
** Module Interface Definition  
** Inputs: C:WHO, C:JULIAN  
** Outputs: None  
** Module Processing Narrative Description:  
**  
** Displays a menu of available reports and prompts  
** Supervisor to chose one or return to SUPMENU1.  
** Weekly and Monthly reports are directed to the  
** printer. Category I and Extended value reports  
** are created in text files on D: drive and may be  
** printed by 'typing' the file using standard  
** operating system functions. All reports should  
** be run only during 'off' hours due to their  
** large amount of resource utilization.  
**  
** Superordinate Modules: SUPMENU1  
** Subordinate Modules: XXBWSTAT,XXMNSTAT,CATIRPT,EXTVAL  
** Author: J.G. BOYNTON  
**  
*****
```

```
ERASE  
STORE T TO V:CONTINUE  
DO WHILE V:CONTINUE  
SET TALK OFF  
STORE ' ' TC V:CHOICE  
TEXT
```

WELCOME TO THE QDR SPECIAL REPORT MENU

- 1 - Biweekly Statistics Report
- 2 - Monthly Statistics Report
- 3 - Category I Report
- 4 - Extended Value Report
- 5 - Exit to Supervisor Menu

Enter your choice

```
ENDTEXT  
@ 19,35 GET V:CHOICE  
READ  
?  
IF V:CHOICE >= "1" .AND. V:CHOICE <= "5"  
IF V:CHOICE = "1" .OR. V:CHOICE = "2"  
ERASE  
@ 3,15 SAY '** YOUR PRINTER MUST BE TURNED ON AND';  
+ 'AVAILABLE **'
```



```

@ 12,20 SAY '          PRESS ANY KEY TO START'
WAIT
@ 22,10 SAY '
ENDIF

```

?

```

DO CASE
CASE V:CHOICE= "1"
RELEASE ALL LIKE V:*
DO C:XXEISTAT.PRG
CASE V:CHOICE= "2"
RELEASE ALL LIKE V:*
DO C:XXMNSTAT.PRG
CASE V:CHOICE= "3"
RELEASE ALL LIKE V:*
USE D:CFEN1 INDEX D:OCASE1
SET TALK OFF
STORE 0 TO P:COUNT
STORE C TO P:TOTAL
SET FOFMAT TO SCREEN
ERASE
SET ALTERNATE TO D:CATIRPT
SET ALTERNATE ON
? 'Date: ',DATE()
?
?
? '          *****          QDR';
+ ' CATEGORY I REPORT          *****
?
? '          CASE #          EXTENDED PRICE';
+ '          OPEN DATE          COG '
?
STORE 0 TO P:PAGE
STORE 5 TO ROW
DO WHILE .NOT. EOF
STORE P:TOTAL+1 TO P:TOTAL
IF CAT='1'
? '          ',CASE,'          ',EPRC,;
? '          ',$(DATES,11,5),;
? '          ',COG
STORE ROW+1 TO ROW
SKIP
STORE P:COUNT+1 TO P:COUNT
IF ROW > 60
ERASE
? CHR(12)
STORE 0 TO ROW
STORE P:PAGE+1 TO P:PAGE
?
?
? '          PAGE ',P:PAGE
?
? '          CASE #          EXT';
+ ' ENDED PRICE          OPEN DATE';
+ '          COG '
?
STORE ROW+4 TO ROW
ENDIF <PAGE IS FULL>
ELSE
SKIP
ENDIF <NOT CAT I>
ENDDO
?
?
?
? '          CAT 1 CASES:',P:COUNT
? '          TOTAL CASES:',P:TOTAL
?
? '          *****          END OF CAT';

```



```

+ 'EGORY I REPORT          *****'
? CHR (12)
SET ALTERNATE OFF
SET ALTERNATE TO
? CHR (7)
? CHR (7)
ERASE
@ 12,20 SAY ' You May Receive Your Cat I ';
+ ' Report On '
@ 13,20 SAY ' D:CATIRPT.TXT '
@ 20,20 SAY ' Press Any Key To Continue'
WAIT

```

```

CASE V:CHOICE= "4"
RELEASE ALL LIKE V:*
USE D:CFEN! INDEX D:EXTVAL
REINDEX
GOTO TCF
SKIP
SET TALK OFF
STORE 0 TO P:COUNT
STCRE 0 TO P:TOTAL
SET FORMAT TO SCREEN
ERASE
SET ALTERNATE TO D:EXTVALUE
SET ALTERNATE ON
? 'Date: ',DATE ()
?
?
? ' ***** QDR';
+ ' EXTENDED VALUE REPORT *****
?
?
+ '
+ '
? ' CASE # COG SM SCREENING'
+ ' NSN CAT NOMEN UIC '
+ ' EXT PRICE OPEN DATE CODE/DATE'
?
STORE 0 TO P:PAGE
STCRE 6 TO ROW
DO WHILE .NOT. EOF
STORE P:TOTAL+1 TO P:TOTAL
? ' CASE ' COG ' SM ' ;
$(NSN,1,4) ' ' $(NSN,5,2) ' ' ;
$(NSN,7,7) ' ' CAT ' ;
$(NOMEN,1,10) ' ' UIC ' ' EPRC ;
' ' $(DATES,11,5) ' ' SCR, / ' ;
$(DATES,21,5)
STORE ROW+1 TO ROW
STORE P:COUNT+1 TO P:COUNT
IF ROW > 60
ERASE
? CHR (12)
STORE 0 TO ROW
STORE P:PAGE+1 TO P:PAGE
?
?
? ' PAGE ',P:PAGE
?
?
+ '
+ '
+ '
? ' SCREENING'
+ ' NSN CASE # COG SM
+ ' UIC CAT NOMEN
+ ' DATE CODE/DATE' OPEN
?
STORE ROW+4 TO ROW

```


XII. QUERY MODULE

```
*****
**
** Date: 23 Nov 1983
** Version: 1.0
** Module Name: QUERY
** Module Purpose: Free Format Query Against the OPEN
**                  and CLOSED Data Files
**
** Module Interface Definition
**   Inputs: C:WHO, C:JULIAN
**   Outputs: None
**
** Module Processing Narrative Description:
**
**   Accepts Selection and Display Parameters from
**   the user and generates the necessary Data Base
**   Commands to extract the desired information.
**   Temporary files are created as the QUERY is
**   being processed. These files are deleted upon
**   exiting. The user may either print or display
**   the information extracted.
**
** Superordinate Modules: MENU1
** Subordinate Modules: None
** Author: R. G. NICHOLS
**
*****

***** Display Menu Selection Options and Accept Response

*SET COLCR TO 112, 6
STORE ' ' TC Q:REPLY
ERASE
@ 6,26 SAY '***** Query Processing *****'
@ 10,27 SAY 'THIS PROGRAM ALLOWS YOU TO'
@ 12,28 SAY 'QUERY THE QDR DATA BASE'
@ 15,32 SAY '1 - Continue'
@ 17,32 SAY '2 - Return to Menu'
@ 20,40 SAY ' ' GET Q:REPLY
READ
DO WHILE Q:REPLY <> '1' .AND. Q:REPLY <> '2'
  @ 23,20 SAY 'Enter 1 or 2 for Your Response'+chr(7)
  @ 20,40 SAY ' ' GET Q:REPLY
  READ
ENDDO

***** If Response is to Exit Release all Memory Variables
***** and Return to MENU1

IF Q:REPLY = '2'
  RELEASE ALL LIKE Q:*
  RETURN
ENDIF

***** Allow User to Select Files to Run The Query Against

STORE ' ' TC Q:REPLY
ERASE
@ 6,24 SAY '*** Query Processing Module ***'
@ 10,20 SAY 'Select File(s) to be Used for this Query'
@ 12,25 SAY '1 - OPEN FILE'
```



```

@ 14,25 SAY '2 - CLOSED FILE'
@ 16,25 SAY '3 - Merged OPEN and CLOSED File'
@ 20,40 GET Q:REPLY PICTURE '9'
READ
DO WHILE Q:REPLY<>'1' .AND. Q:REPLY<>'2' .AND.Q:REPLY<>'3'
  @ 23,28 SAY 'Entry MUST Be 1, 2, or 3' + CHR(7)
  @ 20,40 GET Q:REPLY PICTURE '9'
  READ
ENDDO

@ 23,28 SAY '
STORE 10 TO Q:CNTR
STORE 'Q:I' + STR(Q:CNTR,2) TO Q:LINE

DO CASE
***** If OPEN File Is Selected Indicate O File Selection
  CASE Q:REPLY = '1'
    STORE 1 TO Q:NFPASSES
    STORE 'O' TO Q:FILE
***** If CLOSE File Is Selected Indicate C File Selection
  CASE Q:REPLY = '2'
    STORE 1 TO Q:NFPASSES
    STORE 'C' TO Q:FILE
***** If OPEN File Is Selected Indicate O File Selection
***** and Indicate Two Passes Required For Execution of
***** Generated Code
  CASE C:REPLY = '3'
    STORE 2 TO Q:NFPASSES
    STORE 'O' TO Q:FILE
ENDCASE
STORE ' ' TO Q:SELCMD1
STORE ' ' TO Q:SELCMD2
STORE '00' TO Q:SELECT
STORE 0 TO Q:ITEM

***** Start Loop To Accept Selection Criteria
DO WHILE Q:ITEM <= 4 .AND. Q:SELECT <> '58'
***** Display First Screen of Menu
  IF Q:SELECT = '00'
* - SCREEN1 MENU -
  ERASE
  @ 2,20 SAY 'Enter Selection Criteria For This Query'
  @ 3,20 SAY '(A Maximum of 5 Items May Be Selected)'
  @ 4,0 SAY '-----!';
  + '-----!';
  @ 5,0 SAY 'Data Elements'
  @ 5,25 SAY '!'
* SET COLOR TO 112,2
  @ 5,27 SAY ' 58 End Element Select'
  @ 5,53 SAY ' 59 Abandon Query'
* SET COLOR TO 112, 6
  @ 5,51 SAY '!'
  @ 6,25 SAY '!'
  @ 6,51 SAY '!'
  @ 7,1 SAY '01 Case Number'
  @ 7,25 SAY '!! 11 Origin Code'
  @ 7,51 SAY '!! 21 Interim Repor'
  @ 7,71 SAY 't Date'
  @ 8,1 SAY '02 Cog'
  @ 8,25 SAY '!! 12 Type Document'

```



```

@ 8,51 SAY '!' 22 Origin Prep Date'
@ 9,1 SAY '03 NSM'
@ 9,25 SAY '!' 13 Discovery Date '!';
+ ' 23 Document Number'
@ 10,1 SAY '04 Category'
@ 10,25 SAY '!' 14 Date Received'
@ 10,51 SAY '!' 24 Report Contro'
@ 10,71 SAY '1 Number'
@ 11,1 SAY '05 Nomenclature'
@ 11,25 SAY '!' 15 Open Date'
@ 11,51 SAY '!' 25 FSCM'
@ 12,1 SAY '06 UIC of Origin'
@ 12,25 SAY '!' 16 Transmittal Date '!';
+ ' 26 Contract Number'
@ 13,1 SAY '07 Unit of Issue'
@ 13,25 SAY '!' 17 IM Response Date '!';
+ ' 27 Credit Code'
@ 14,1 SAY '08 Unit Price'
@ 14,25 SAY '!' 18 Close Date'
@ 14,51 SAY '!' 28 Screening Code'
@ 15,1 SAY '09 Quantity Deficient '!';
+ ' 19 Reopen Date '!' 29 SMIC'
@ 16,1 SAY '10 Extended Price'
@ 16,25 SAY '!' 20 Screen Report'
@ 16,46 SAY 'Date ! 30 Next Page of Elements'
@ 17,0 SAY '-----';
+ '-----';
@ 18,0 SAY 'Relations a - Include b - Exclude'
@ 18,42 SAY 'c - Range d - Equal'
@ 19,10 SAY 'e - Not Equal f - Less Than ';
+ ' g - Greater Than'
@ 21,27 SAY 'Enter Data Element Number ' GET ;
Q:SELECT PICTURE '99'

READ
DO WHILE Q:SELECT < '00' .OR. Q:SELECT > '59'
@ 23,26 SAY 'Select From Above (00 - 59)' + CHR(7)
@ 21,27 SAY 'Enter Data Element Number ' GET ;
Q:SELECT PICTURE '99'

READ
ENCDO
@ 23,26 SAY '
ELSE

```

***** Display Second Screen of Menu

```

IF Q:SELECT = '30'
ERASE
@ 2,20 SAY 'Enter Selection Criteria For This';
+ ' Query'
@ 3,20 SAY '(A Maximum of 5 Items May Be Selected)'
@ 4,0 SAY '-----';
+ '-----';
@ 5,0 SAY 'Data Elements'
@ 5,25 SAY '!'
* SET COLCR TC 112,2
@ 5,27 SAY ' 58 End Element Select'
@ 5,53 SAY ' 59 Abandon Query'
* SET COLCR TC 112, 6
@ 5,51 SAY '!'
@ 6,25 SAY '!'
@ 6,51 SAY '!'
@ 7,1 SAY '31 90 Region'
@ 7,25 SAY '!' 40 Deficiency Ver '!';
+ ' 49 Action Code'
@ 8,1 SAY '32 Type Defect'
@ 8,25 SAY '!' 41 Deficiency Resp '!';
+ ' 50 Cost Code'
@ 9,1 SAY '33 Vendor Liab Code'
@ 9,25 SAY '!' 42 New-Repair/Ovhl '!';

```



```

+ ' 51 Status Code'
@ 10,1 SAY '34 Action Point'
@ 10,25 SAY '!! 43 Date Mfg/Ovhl'
@ 10,51 SAY '!! 52 Cause Code'
@ 11,1 SAY '35 Screen Quantity'
@ 11,25 SAY '!! 44 Opn Time at Failure!";
+ " 53 Action Dis'n"
@ 12,1 SAY '36 Analyst Code'
@ 12,25 SAY '!! 45 GFM'
@ 12,51 SAY '!! 54 Part Number'
@ 13,1 SAY '37 Quantity Inspected !';
+ ' 46 Work Unit Code !';55'
@ 13,58 SAY 'Lot/Ser/Batch'
@ 14,1 SAY '38 Quantity Received !';
+ ' 47 Discovery Code !';56'
@ 14,58 SAY 'Def Item'
@ 15,1 SAY '39 Quantity in Stock !';
+ ' 48 Return Code !';57'
@ 15,58 SAY 'Warranty'
@ 16,1 SAY '58 End Element Select !';
+ ' 59 Abandon Query'
@ 16,51 SAY '!! 00 Prev Page of Elements'
@ 17,0 SAY '-----';
+ '-----';
@ 18,0 SAY 'Relaticns a - Include b - Exclude'
@ 18,42 SAY 'c - Range d - Equal'
@ 19,10 SAY 'e - Nct Equal f - Less Than ';
+ ' g - Greater Than'
@ 21,27 SAY 'Enter Data Element Number ' GET ;
Q:SELECT PICTURE '99'
READ
DO WHILE Q:SELECT < '00' .OR. Q:SELECT > '59'
@ 23,26 SAY 'Select From Above (00 - 59)'+CHR(7)
@ 21,27 SAY 'Enter Data Element Number ' ;
GET Q:SELECT PICTURE '99'
READ
ENDDO
@ 23,26 SAY '
ELSE

```

```

***** Begin Case To Generate Formats For Entering Initial
***** Values - Each Selected Item Has Its Name and Picture
***** Stored in an Indirect Variable

```

```
DO CASE
```

```

***** If Termination Requested Release Local Memory and
***** Return to Calling Routine

```

```

CASE Q:SELECT = '59'
RELEASE ALL LIKE Q:*
RETURN

```

```

***** Begin Generating Selection Code
***** If a Character Field Set Character Flag
***** Load The Picture for the Data Field
***** Initialize the Data Field and Then Continue
***** To Generate Code

```

```

CASE Q:SELECT = '01'
STORE 'CASE' TO Q:SELITEM
STORE "'9999999A'" TO Q:SELPIC
STORE ' ' TO Q:INIT1
STORE T TO Q:CHAR
STORE 'Q:SELCMD1' TO Q:SELCMD
CASE Q:SELECT = '02'
STORE 'COG' TO Q:SELITEM
STORE "'9A'" TO Q:SELPIC
STORE T TO Q:CHAR

```



```

STORE ' ' TO Q:INIT1
STORE 'C:SELCMD1' TO Q:SELCMD
CASE Q:SELECT = '03'
STORE 'NSN' TO Q:SELITEM
STORE "'99999XXXXX9999'" TO Q:SELPIC
STORE T TO Q:CHAR
STORE ' ' TO Q:INIT1
STORE 'C:SELCMD1' TO Q:SELCMD
CASE Q:SELECT = '04'
STORE 'CAT' TO Q:SELITEM
STORE "'9'" TO Q:SELPIC
STORE ' ' TO Q:INIT1
STORE T TO Q:CHAR
STORE 'C:SELCMD1' TO Q:SELCMD
CASE Q:SELECT = '05'
STORE 'NOMEN' TO Q:SELITEM
STORE "'XXXXX XXXXXXXXXXXXXXX'" TO Q:SELPIC
STORE ' ' TO Q:INIT1
STORE T TO Q:CHAR
STORE 'C:SELCMD1' TO Q:SELCMD
CASE Q:SELECT = '06'
STORE 'UIC' TO Q:SELITEM
STORE "'AXXXX'" TO Q:SELPIC
STORE ' ' TO Q:INIT1
STORE T TO Q:CHAR
STORE 'C:SELCMD1' TO Q:SELCMD
CASE Q:SELECT = '07'
STORE 'UI' TO Q:SELITEM
STORE "'AA'" TO Q:SELPIC
STORE ' ' TO Q:INIT1
STORE T TO Q:CHAR
STORE 'C:SELCMD1' TO Q:SELCMD
CASE Q:SELECT = '08'
STORE 'UPRC' TO Q:SELITEM
STORE "'999999.99'" TO Q:SELPIC
STORE 9 TO Q:NR
STORE 2 TO Q:DEC
STORE 0 TO Q:INIT1
STORE F TO Q:CHAR
STORE 'C:SELCMD1' TO Q:SELCMD
CASE Q:SELECT = '09'
STORE 'QTYDEF' TO Q:SELITEM
STORE "'999999'" TO Q:SELPIC
STORE 6 TO Q:NR
STORE 0 TO Q:DEC
STORE 0 TO Q:INIT1
STORE F TO Q:CHAR
STORE 'C:SELCMD1' TO Q:SELCMD
CASE Q:SELECT = '10'
STORE 'EPRC' TO Q:SELITEM
STORE "'999999999.99'" TO Q:SELPIC
STORE 12 TO Q:NR
STORE 2 TO Q:DEC
STORE 0 TO Q:INIT1
STORE F TO Q:CHAR
STORE 'C:SELCMD1' TO Q:SELCMD
CASE Q:SELECT = '11'
STORE 'ORG' TO Q:SELITEM
STORE "'XXX'" TO Q:SELPIC
STORE ' ' TO Q:INIT1
STORE T TO Q:CHAR
STORE 'C:SELCMD1' TO Q:SELCMD
CASE Q:SELECT = '12'
STORE 'DOC' TO Q:SELITEM
STORE "'9'" TO Q:SELPIC
STORE ' ' TO Q:INIT1
STORE T TO Q:CHAR
STORE 'C:SELCMD1' TO Q:SELCMD
CASE Q:SELECT = '13'

```



```

STORE '$(DATES,1,5)' TO Q:SELITEM
STORE "'99999'" TO Q:SELPIC
STORE ' ' TO Q:INIT1
STORE I TO Q:CHAR
STORE 'C:SELCMD1' TO Q:SELCMD
CASE Q:SELECT = '14'
STORE '$(DATES,6,5)' TO Q:SELITEM
STORE "'99999'" TO Q:SELPIC
STORE ' ' TO Q:INIT1
STORE T TO Q:CHAR
STORE 'C:SELCMD1' TO Q:SELCMD
CASE Q:SELECT = '15'
STORE '$(DATES,11,5)' TO Q:SELITEM
STORE "'99999'" TO Q:SELPIC
STORE ' ' TO Q:INIT1
STORE T TO Q:CHAR
STORE 'C:SELCMD1' TO Q:SELCMD
CASE Q:SELECT = '16'
STORE '$(DATES,16,5)' TO Q:SELITEM
STORE "'99999'" TO Q:SELPIC
STORE ' ' TO Q:INIT1
STORE T TO Q:CHAR
STORE 'C:SELCMD1' TO Q:SELCMD
CASE Q:SELECT = '17'
STORE '$(DATES,26,5)' TO Q:SELITEM
STORE "'99999'" TO Q:SELPIC
STORE ' ' TO Q:INIT1
STORE I TO Q:CHAR
STORE 'C:SELCMD1' TO Q:SELCMD
CASE Q:SELECT = '18'
STORE '$(DATES,36,5)' TO Q:SELITEM
STORE "'99999'" TO Q:SELPIC
STORE ' ' TO Q:INIT1
STORE T TO Q:CHAR
STORE 'C:SELCMD1' TO Q:SELCMD
CASE Q:SELECT = '19'
STORE '$(DATES,41,5)' TO Q:SELITEM
STORE "'99999'" TO Q:SELPIC
STORE ' ' TO Q:INIT1
STORE T TO Q:CHAR
STORE 'C:SELCMD1' TO Q:SELCMD
CASE Q:SELECT = '20'
STORE '$(DATES,21,5)' TO Q:SELITEM
STORE "'99999'" TO Q:SELPIC
STORE ' ' TO Q:INIT1
STORE T TO Q:CHAR
STORE 'C:SELCMD1' TO Q:SELCMD
CASE Q:SELECT = '21'
STORE '$(DATES,31,5)' TO Q:SELITEM
STORE "'99999'" TO Q:SELPIC
STORE ' ' TO Q:INIT1
STORE T TO Q:CHAR
STORE 'C:SELCMD1' TO Q:SELCMD
CASE Q:SELECT = '22'
STORE '$(DATES,46,5)' TO Q:SELITEM
STORE "'99999'" TO Q:SELPIC
STORE ' ' TO Q:INIT1
STORE T TO Q:CHAR
STORE 'C:SELCMD1' TO Q:SELCMD
CASE Q:SELECT = '23'
STORE 'DOCNO' TO Q:SELITEM
STORE "'XXXXX X999999999'" TO Q:SELPIC
STORE ' ' TO Q:INIT1
STORE T TO Q:CHAR
STORE 'C:SELCMD1' TO Q:SELCMD
CASE Q:SELECT = '24'
STORE 'REPCON' TO Q:SELITEM
STORE "'XXXXX X9999999'" TO Q:SELPIC
STORE ' ' TO Q:INIT1

```



```

STORE T TO Q:CHAR
STORE 'Q:SELCMD1' TO Q:SELCMD
CASE Q:SELECT = '25'
STORE 'FSCM' TO Q:SELITEM
STORE 'XXXXXX' TO Q:SELPIC
STORE ' ' TO Q:INIT1
STORE T TO Q:CHAR
STORE 'Q:SELCMD1' TO Q:SELCMD
CASE Q:SELECT = '26'
STORE 'NUM' TO Q:SELITEM
STORE "XXXXX X99AXXXXXXXXX" TO Q:SELPIC
STORE ' ' TO Q:INIT1
STORE T TO Q:CHAR
STORE 'Q:SELCMD1' TO Q:SELCMD
CASE Q:SELECT = '27'
STORE 'CR' TO Q:SELITEM
STORE "A" TO Q:SELPIC
STORE ' ' TO Q:INIT1
STORE T TO Q:CHAR
STORE 'Q:SELCMD1' TO Q:SELCMD
CASE Q:SELECT = '28'
STORE 'SCR' TO Q:SELITEM
STORE "XXX" TO Q:SELPIC
STORE ' ' TO Q:INIT1
STORE T TO Q:CHAR
STORE 'Q:SELCMD1' TO Q:SELCMD
CASE Q:SELECT = '29'
STORE 'SM' TO Q:SELITEM
STORE "AX" TO Q:SELPIC
STORE ' ' TO Q:INIT1
STORE T TO Q:CHAR
STORE 'Q:SELCMD1' TO Q:SELCMD
CASE Q:SELECT = '31'
STORE 'C9Q' TO Q:SELITEM
STORE "X" TO Q:SELPIC
STORE ' ' TO Q:INIT1
STORE T TO Q:CHAR
STORE 'Q:SELCMD1' TO Q:SELCMD
CASE Q:SELECT = '32'
STORE 'DEF' TO Q:SELITEM
STORE "99" TO Q:SELPIC
STORE ' ' TO Q:INIT1
STORE T TO Q:CHAR
STORE 'Q:SELCMD1' TO Q:SELCMD
CASE Q:SELECT = '33'
STORE 'VLC' TO Q:SELITEM
STORE "A" TO Q:SELPIC
STORE ' ' TO Q:INIT1
STORE T TO Q:CHAR
STORE 'Q:SELCMD1' TO Q:SELCMD
CASE Q:SELECT = '34'
STORE 'ACTPT' TO Q:SELITEM
STORE "AXXXX X99999" TO Q:SELPIC
STORE ' ' TO Q:INIT1
STORE T TO Q:CHAR
STORE 'Q:SELCMD1' TO Q:SELCMD
CASE Q:SELECT = '35'
STORE 'SCROT' TO Q:SELITEM
STORE "999999" TO Q:SELPIC
STORE 6 TO Q:NR
STORE 0 TO Q:EEC
STORE 0 TO Q:INIT1
STORE F TO Q:CHAR
STORE 'Q:SELCMD1' TO Q:SELCMD
CASE Q:SELECT = '36'
STORE 'WHO' TO Q:SELITEM
STORE "XXXX" TO Q:SELPIC
STORE ' ' TO Q:INIT1
STORE T TO Q:CHAR

```



```

STORE 'Q:SELCMD1' TO Q:SELCMD
CASE Q:SELECT = '37'
STORE 'QTYINS' TO Q:SELITEM
STORE "'999999'" TO Q:SELPIC
STORE 6 TO Q:NR
STORE 0 TO Q:DEC
STORE 0 TO Q:INIT1
STORE F TO Q:CHAR
STORE 6 TO Q:NR
STORE 'Q:SELCMD2' TO Q:SELCMD
CASE Q:SELECT = '38'
STORE 'QTYREC' TO Q:SELITEM
STORE "'999999'" TO Q:SELPIC
STORE 6 TO Q:NR
STORE 0 TO Q:DEC
STORE 0 TO Q:INIT1
STORE F TO Q:CHAR
STORE 6 TO Q:NR
STORE 'Q:SELCMD2' TO Q:SELCMD
CASE Q:SELECT = '39'
STORE 'QTYSTK' TO Q:SELITEM
STORE "'999999'" TO Q:SELPIC
STORE 6 TO Q:NR
STORE 0 TO Q:DEC
STORE 0 TO Q:INIT1
STORE F TO Q:CHAR
STORE 6 TO Q:NR
STORE 'Q:SELCMD2' TO Q:SELCMD
CASE Q:SELECT = '40'
STORE 'DEFV' TO Q:SELITEM
STORE "'A'" TO Q:SELPIC
STORE " " TO Q:INIT1
STORE I TO Q:CHAR
STORE 'Q:SELCMD2' TO Q:SELCMD
CASE Q:SELECT = '41'
STORE 'DEFR' TO Q:SELITEM
STORE "'A'" TO Q:SELPIC
STORE " " TO Q:INIT1
STORE I TO Q:CHAR
STORE 'Q:SELCMD2' TO Q:SELCMD
CASE Q:SELECT = '42'
STORE 'ITEM' TO Q:SELITEM
STORE "'A'" TO Q:SELPIC
STORE " " TO Q:INIT1
STORE I TO Q:CHAR
STORE 'Q:SELCMD2' TO Q:SELCMD
CASE Q:SELECT = '43'
STORE 'CVER' TO Q:SELITEM
STORE "'99999'" TO Q:SELPIC
STORE " " TO Q:INIT1
STORE I TO Q:CHAR
STORE 'Q:SELCMD2' TO Q:SELCMD
CASE Q:SELECT = '44'
STORE 'CTF' TO Q:SELITEM
STORE "'A9999'" TO Q:SELPIC
STORE " " TO Q:INIT1
STORE I TO Q:CHAR
STORE 'Q:SELCMD2' TO Q:SELCMD
CASE Q:SELECT = '45'
STORE 'GOV' TO Q:SELITEM
STORE "'X'" TO Q:SELPIC
STORE " " TO Q:INIT1
STORE I TO Q:CHAR
STORE 'Q:SELCMD2' TO Q:SELCMD
CASE Q:SELECT = '46'
STORE 'WUC' TO Q:SELITEM
STORE "'XXXXXX'" TO Q:SELPIC
STORE " " TO Q:INIT1
STORE T TO Q:CHAR

```



```

Q:SELECT<>'58'
STCRE Q:ITEM + 1 TO Q:ITEM
ERASE
STCRE ' ' TO Q:SELECT
@ 06,0 SAY 'Relations a - Include b - Exclude'
@ 06,42 SAY 'c - Range d - Equal'
@ 07,10 SAY 'e - Not Equal f - Less Than';
+ 'g - Greater Than'
@ 10,10 SAY 'Enter Relationship for Selection ' GET ;
Q:SELECT PICTURE 'A'
READ

```

***** Validate Entered Value

```

DO WHILE !(Q:SELECT) < 'A' .OR. !(Q:SELECT) > 'G'
@ 23,18 SAY 'Enter Relationship as Displayed';
+ 'Above (A - G)' + CHR(7)
@ 10,10 SAY 'Enter Relationship for Selection';
GET Q:SELECT PICT 'A'
READ
ENDDC
@ 23,18 SAY ' ';
+ ' ';

```

***** If Range Selected, Accept Two Initial Values

```

IF !(Q:SELECT) = 'C'
STCRE Q:INIT1 TO Q:INIT2
@ 14,10 SAY 'Enter MINIMUM Value Allowed';
GET Q:INIT1 PICT &Q:SELPIC
READ
@ 16,10 SAY 'Enter MAXIMUM Value Allowed';
GET Q:INIT2 PICT &Q:SELPIC
READ

```

***** Ensure That Values are Properly Ordered

```

IF Q:INIT1 > Q:INIT2
STORE Q:INIT1 TO Q:TEMP
STORE Q:INIT2 TO Q:INIT1
STORE Q:TEMP TO Q:INIT2
RELEASE Q:TEMP
ENDIF

```

***** If Character Field, Place Quotes Around Initial Value(s)

```

IF Q:CHAR
STORE '"' + ! (Q:INIT1) + '"' TO Q:INIT1
STORE '"' + ! (Q:INIT2) + '"' TO Q:INIT2
ELSE

```

***** Format Numerics to be Characters For Code Generation

```

STORE STR(Q:INIT1,Q:NR,Q:DEC) TO Q:INIT1T
STORE Q:INIT1T TO Q:INIT1
STORE STR(Q:INIT2,Q:NR,Q:DEC) TO Q:INIT2T
STORE Q:INIT2T TO Q:INIT2
RELEASE Q:INIT1T, Q:INIT2T
ENDIF

```

***** Form Partial Command Line

```

IF &Q:SELCMD = ' '
STORE Q:SELITEM+'>'+Q:INIT1+'.AND.'+Q:SELITEM+'<'+Q:INIT2 TO &Q:SELCMD
ELSE
STORE &Q:SELCMD+'.AND.'+Q:SELITEM+'>'+Q:INIT1;
+'.AND.'+Q:SELITEM+'<'+Q:INIT2 TO &Q:SELCMD

```



```

        ENDIF
ELSE
***** Accept Selection Values For Query (Single Value)
        @ 12,10 SAY 'Enter Value for Selection ' GET;
          Q:INIT1 PICT &Q:SELPIC
        READ
***** If Character, Place Quotes Around Initial Value
        IF Q:CHAR
          STORE '"' + (Q:INIT1) + '"' TO Q:INIT1
        ELSE
***** Format Numerics to be Characters For Code Generation
          STORE STR(Q:INIT1,Q:NR,Q:DEC) TO Q:INIT1T
          STORE Q:INIT1T TO Q:INIT1
          RELEASE Q:INIT1T
        ENDIF
***** Form Partial Command Line
***** Command Line Formation Uses Indirect Addressing to
***** Point to the Location Of the Command Line
***** If A Previous Line Has Been Created, Join Together
***** With an AND
          IF &Q:SELCMD <> ' '
            STORE &Q:SELCMD+'.AND.' TO &Q:SELCMD
          ENDIF
          DO CASE
            CASE !(Q:SELECT) = 'A'
              STORE &Q:SELCMD-Q:SELITEM+'=' + Q:INIT1 TO &Q:SELCMD
            CASE !(Q:SELECT) = 'B'
              STORE &Q:SELCMD-Q:SELITEM+'<>' + Q:INIT1 TO &Q:SELCMD
            CASE !(Q:SELECT) = 'D'
              STORE &Q:SELCMD-Q:SELITEM+'=' + Q:INIT1 TO &Q:SELCMD
            CASE !(Q:SELECT) = 'E'
              STORE &Q:SELCMD-Q:SELITEM+'<>' + Q:INIT1 TO &Q:SELCMD
            CASE !(Q:SELECT) = 'F'
              STORE &Q:SELCMD-Q:SELITEM+'<' + Q:INIT1 TO &Q:SELCMD
            CASE !(Q:SELECT) = 'G'
              STORE &Q:SELCMD-Q:SELITEM+'>' + Q:INIT1 TO &Q:SELCMD
          ENDCASE
        ENDIF
        STORE '00' TO Q:SELECT
      ENDIF
    ENDIF
  ENDIF
ENDDO
*SET COLCR TO 112, 3
RELEASE Q:SELITEM, Q:SELPIC, Q:CHAR, Q:INIT1, Q:INIT2, Q:NR
STORE F TC Q:DATESEL
STORE 'CASE ' TO Q:FIDCMD1
STORE 'CASE ' TO Q:FIDCMD2
STORE ' ' TO Q:FIELD
STORE 0 TO Q:ITEM
STORE '00' TO Q:SELECT
STORE ' ' TC Q:CHOSEN
***** Display Selection Menu to Allow Selection of Items
***** To Be Displayed - Displays First Screen
DO WHILE Q:ITEM <= 9 .AND. Q:SELECT <> '58'
  IF Q:SELECT = '00'
    ERASE
    @ 2,21 SAY 'Enter Field Selection For This Query'

```



```

@ 3,19 SAY '(A Maximum of 10 Fields May Be Selected)';
@ 4,0 SAY '-----!';
+ '-----!';
@ 5,0 SAY 'Data Elements'
@ 5,25 SAY '!'
* SET CCLOR TO 112,2
@ 5,27 SAY ' 58 End Element Select'
@ 5,53 SAY ' 59 Abandon Query'
* SET CCLOR TO 112,3
@ 5,51 SAY '!'
@ 6,25 SAY '!'
@ 6,51 SAY '!'
@ 7,1 SAY '01 Case Number'
@ 7,25 SAY '!' 11 Origin Code'
@ 7,51 SAY '!' 21 Interim Report'
@ 7,71 SAY 't Date'
@ 8,1 SAY '02 Cog'
@ 8,25 SAY '!' 12 Type Document'
@ 8,51 SAY '!' 22 Origin Prep Date'
@ 9,1 SAY '03 NSN'
@ 9,25 SAY '!' 13 Discovery Date !';
+ '!' 23 Document Number'
@ 10,1 SAY '04 Category'
@ 10,25 SAY '!' 14 Date Received'
@ 10,51 SAY '!' 24 Report Contro'
@ 10,71 SAY '1 Number'
@ 11,1 SAY '05 Nomenclature'
@ 11,25 SAY '!' 15 Open Date'
@ 11,51 SAY '!' 25 FSCM'
@ 12,1 SAY '06 UIC of Origin'
@ 12,25 SAY '!' 16 Transmittal Date !';
+ '!' 26 Contract Number'
@ 13,1 SAY '07 Unit of Issue'
@ 13,25 SAY '!' 17 IM Response Date !';
+ '!' 27 Credit Code'
@ 14,1 SAY '08 Unit Price'
@ 14,25 SAY '!' 18 Close Date'
@ 14,51 SAY '!' 28 Screening Code'
@ 15,1 SAY '09 Quantity Deficient !';
+ '!' 19 Reopen Date !' 29 SMIC'
@ 16,1 SAY '10 Extended Price'
@ 16,25 SAY '!' 20 Screen Report'
@ 16,46 SAY 'Date ! 30 Next Page of Elements'
@ 17,0 SAY '-----!';
+ '-----!';

```

***** Display Previously Selected Fields

```

@ 19,0 SAY 'Fields Currently Selected '+ Q:CHOSEN
@ 21,27 SAY 'Enter Field Number ' GET Q:SELECT PICTURE '99'
READ

```

***** Validate Field Selection

```

DO WHILE Q:SELECT < '00' .OR. Q:SELECT > '59'
@ 23,26 SAY 'Select From Above (00 - 59)'+CHR(7)
@ 21,27 SAY 'Enter Field Number ' GET ;
Q:SELECT PICTURE '99'

```

```

READ
ENDDO

```

```

@ 23,26 SAY '
ELSE

```

***** Display Second Screen For Selection

```

IF Q:SELECT = '30'
ERASE
@ 2,21 SAY 'Enter Field Selection For This Query'
@ 3,19 SAY '(A Maximum of 10 Fields May Be Selected) '

```



```

@ 4,0 SAY '-----';
+ '-----';
@ 5,0 SAY 'Data Elements'
@ 5,25 SAY '!!'
* SET COLOR TO 112,2
@ 5,27 SAY '58 End Element Select'
* @ 5,53 SAY '59 Abandon Query'
SET COLOR TO 112,3
@ 5,51 SAY '!!'
@ 6,25 SAY '!!'
@ 6,51 SAY '!!'
@ 7,1 SAY '31 90 Region'
@ 7,25 SAY '40 Deficiency Ver !';
+ '49 Action Code'
@ 8,1 SAY '32 Type Defect'
@ 8,25 SAY '41 Deficiency Resp !';
+ '50 Cost Code'
@ 9,1 SAY '33 Vendor Liab Code'
@ 9,25 SAY '42 New-Repair/Ovhl !';
+ '51 Status Code'
@ 10,1 SAY '34 Action Point'
@ 10,25 SAY '43 Date Mfg/Ovhl'
@ 10,51 SAY '52 Cause Code'
@ 11,1 SAY '35 Screen Quantity'
@ 11,25 SAY '44 Opn Time at Failure!";
+ "53 Action Dis'n"
@ 12,1 SAY '36 Analyst Code'
@ 12,25 SAY '45 GFM'
@ 12,51 SAY '54 Part Number'
@ 13,1 SAY '37 Quantity Inspected !';
+ '46 Work Unit Code ! 55'
@ 13,58 SAY 'Lot/Ser/Batch'
@ 14,1 SAY '38 Quantity Received !';
+ '47 Discovery Code ! 56'
@ 14,58 SAY 'Def Item'
@ 15,1 SAY '39 Quantity in Stock !';
+ '48 Return Code ! 57'
@ 15,58 SAY 'Warranty'
@ 16,1 SAY '58 End Element Select !';
+ '59 Abandon Query'
@ 16,51 SAY '00 Prev Page of Elements'
@ 17,0 SAY '-----';
+ '-----';
@ 19,0 SAY 'Fields Currently Selected '+ Q:CHOSEN
@ 21,27 SAY 'Enter Field Number ' GET ;
Q:SELECT PICTURE '99'

```

READ

***** Validate Field Selection

```

DO WHILE Q:SELECT < '00' .OR. Q:SELECT > '59'
@ 23,26 SAY 'Select From Above (00 - 59)'+CHR(7)
@ 21,27 SAY 'Enter Field Number ' GET ;
Q:SELECT PICTURE '99'

```

READ

ENDDO

```

@ 23,26 SAY '
ELSE

```

***** Begin Creating Code For Fields Selected

DO CASE

***** If 59 Entered, Release All Local Memory and Return

```

CASE Q:SELECT = '59'
RELEASE ALL LIKE Q:*
RETURN

```


***** Store Selection Name to Q:SELITEM
***** Store Field Selection Pointer To Q:FLDCMD

```
CASE Q:SELECT = '01'  
  STORE 'CASE' TO Q:SELITEM  
  STORE 'Q:FLDCMD1' TO Q:FLDCMD  
CASE Q:SELECT = '02'  
  STORE 'COG' TO Q:SELITEM  
  STORE 'Q:FLDCMD1' TO Q:FLDCMD  
CASE Q:SELECT = '03'  
  STORE 'NSN' TO Q:SELITEM  
  STORE 'Q:FLDCMD1' TO Q:FLDCMD  
CASE Q:SELECT = '04'  
  STORE 'CAT' TO Q:SELITEM  
  STORE 'Q:FLDCMD1' TO Q:FLDCMD  
CASE Q:SELECT = '05'  
  STORE 'NOMEN' TO Q:SELITEM  
  STORE 'Q:FLDCMD1' TO Q:FLDCMD  
CASE Q:SELECT = '06'  
  STORE 'UIC' TO Q:SELITEM  
  STORE 'Q:FLDCMD1' TO Q:FLDCMD  
CASE Q:SELECT = '07'  
  STORE 'UI' TO Q:SELITEM  
  STORE 'Q:FLDCMD1' TO Q:FLDCMD  
CASE Q:SELECT = '08'  
  STORE 'UPRC' TO Q:SELITEM  
  STORE 'Q:FLDCMD1' TO Q:FLDCMD  
CASE Q:SELECT = '09'  
  STORE 'QTYDEF' TO Q:SELITEM  
  STORE 'Q:FLDCMD1' TO Q:FLDCMD  
CASE Q:SELECT = '10'  
  STORE 'EPRC' TO Q:SELITEM  
  STORE 'Q:FLDCMD1' TO Q:FLDCMD  
CASE Q:SELECT = '11'  
  STORE 'CRG' TO Q:SELITEM  
  STORE 'Q:FLDCMD1' TO Q:FLDCMD  
CASE Q:SELECT = '12'  
  STORE 'DOC' TO Q:SELITEM  
  STORE 'Q:FLDCMD1' TO Q:FLDCMD  
CASE Q:SELECT = '13'  
  STORE '$(DATES, 1, 5)' TO Q:SELITEM  
  STORE 'Q:FLDCMD1' TO Q:FLDCMD  
CASE Q:SELECT = '14'  
  STORE '$(DATES, 6, 5)' TO Q:SELITEM  
  STORE 'Q:FLDCMD1' TO Q:FLDCMD  
CASE Q:SELECT = '15'  
  STORE '$(DATES, 11, 5)' TO Q:SELITEM  
  STORE 'Q:FLDCMD1' TO Q:FLDCMD  
CASE Q:SELECT = '16'  
  STORE '$(DATES, 16, 5)' TO Q:SELITEM  
  STORE 'Q:FLDCMD1' TO Q:FLDCMD  
CASE Q:SELECT = '17'  
  STORE '$(DATES, 26, 5)' TO Q:SELITEM  
  STORE 'Q:FLDCMD1' TO Q:FLDCMD  
CASE Q:SELECT = '18'  
  STORE '$(DATES, 36, 5)' TO Q:SELITEM  
  STORE 'Q:FLDCMD1' TO Q:FLDCMD  
CASE Q:SELECT = '19'  
  STORE '$(DATES, 41, 5)' TO Q:SELITEM  
  STORE 'Q:FLDCMD1' TO Q:FLDCMD  
CASE Q:SELECT = '20'  
  STORE '$(DATES, 21, 5)' TO Q:SELITEM  
  STORE 'Q:FLDCMD1' TO Q:FLDCMD  
CASE Q:SELECT = '21'  
  STORE '$(DATES, 31, 5)' TO Q:SELITEM  
  STORE 'Q:FLDCMD1' TO Q:FLDCMD  
CASE Q:SELECT = '22'  
  STORE '$(DATES, 46, 5)' TO Q:SELITEM  
  STORE 'Q:FLDCMD1' TO Q:FLDCMD
```


CASE Q:SELECT = '23'
 STORE 'DOCNO' TO Q:SELITEM
 STORE 'C:FLDCMD1' TO Q:FLDCMD
 CASE Q:SELECT = '24'
 STORE 'REPCON' TO Q:SELITEM
 STORE 'Q:FLDCMD1' TO Q:FLDCMD
 CASE Q:SELECT = '25'
 STORE 'FSCM' TO Q:SELITEM
 STORE 'C:FLDCMD1' TO Q:FLDCMD
 CASE Q:SELECT = '26'
 STORE 'NUM' TO Q:SELITEM
 STORE 'Q:FLDCMD1' TO Q:FLDCMD
 CASE Q:SELECT = '27'
 STORE 'CR' TO Q:SELITEM
 STORE 'C:FLDCMD1' TO Q:FLDCMD
 CASE Q:SELECT = '28'
 STORE 'SCR' TO Q:SELITEM
 STORE 'Q:FLDCMD1' TO Q:FLDCMD
 CASE Q:SELECT = '29'
 STORE 'SM' TO Q:SELITEM
 STORE 'C:FLDCMD1' TO Q:FLDCMD
 CASE Q:SELECT = '31'
 STORE 'C9Q' TO Q:SELITEM
 STORE 'C:FLDCMD1' TO Q:FLDCMD
 CASE Q:SELECT = '32'
 STORE 'DEF' TO Q:SELITEM
 STORE 'C:FLDCMD1' TO Q:FLDCMD
 CASE Q:SELECT = '33'
 STORE 'VLC' TO Q:SELITEM
 STORE 'C:FLDCMD1' TO Q:FLDCMD
 CASE Q:SELECT = '34'
 STORE 'ACTPT' TO Q:SELITEM
 STORE 'C:FLDCMD1' TO Q:FLDCMD
 CASE Q:SELECT = '35'
 STORE 'SCROTQ' TO Q:SELITEM
 STORE 'C:FLDCMD1' TO Q:FLDCMD
 CASE Q:SELECT = '36'
 STORE 'WHO' TO Q:SELITEM
 STORE 'C:FLDCMD1' TO Q:FLDCMD
 CASE Q:SELECT = '37'
 STORE 'CTYINS' TO Q:SELITEM
 STORE 'C:FLDCMD2' TO Q:FLDCMD
 CASE Q:SELECT = '38'
 STORE 'CTYREC' TO Q:SELITEM
 STORE 'C:FLDCMD2' TO Q:FLDCMD
 CASE Q:SELECT = '39'
 STORE 'CTYSTK' TO Q:SELITEM
 STORE 'Q:FLDCMD2' TO Q:FLDCMD
 CASE Q:SELECT = '40'
 STORE 'DEFV' TO Q:SELITEM
 STORE 'C:FLDCMD2' TO Q:FLDCMD
 CASE Q:SELECT = '41'
 STORE 'DEFR' TO Q:SELITEM
 STORE 'C:FLDCMD2' TO Q:FLDCMD
 CASE Q:SELECT = '42'
 STORE 'ITEM' TO Q:SELITEM
 STORE 'C:FLDCMD2' TO Q:FLDCMD
 CASE Q:SELECT = '43'
 STORE 'CVER' TO Q:SELITEM
 STORE 'Q:FLDCMD2' TO Q:FLDCMD
 CASE Q:SELECT = '44'
 STORE 'OIF' TO Q:SELITEM
 STORE 'C:FLDCMD2' TO Q:FLDCMD
 CASE Q:SELECT = '45'
 STORE 'GOV' TO Q:SELITEM
 STORE 'C:FLDCMD2' TO Q:FLDCMD
 CASE Q:SELECT = '46'
 STORE 'WUC' TO Q:SELITEM
 STORE 'C:FLDCMD2' TO Q:FLDCMD


```

CASE Q:SELECT = '47'
  STORE 'DIS' TO Q:SELITEM
  STORE 'Q:FLDCMD2' TO Q:FLDCMD
CASE Q:SELECT = '48'
  STORE 'RETC' TO Q:SELITEM
  STORE 'Q:FLDCMD2' TO Q:FLDCMD
CASE Q:SELECT = '49'
  STORE 'ACTTKN' TO Q:SELITEM
  STORE 'Q:FLDCMD2' TO Q:FLDCMD
CASE Q:SELECT = '50'
  STORE 'COSTC' TO Q:SELITEM
  STORE 'Q:FLDCMD2' TO Q:FLDCMD
CASE Q:SELECT = '51'
  STORE 'STATUSC' TO Q:SELITEM
  STORE 'Q:FLDCMD2' TO Q:FLDCMD
CASE Q:SELECT = '52'
  STORE 'CAUSEC' TO Q:SELITEM
  STORE 'Q:FLDCMD2' TO Q:FLDCMD
CASE Q:SELECT = '53'
  STORE 'ACTDISP' TO Q:SELITEM
  STORE 'Q:FLDCMD2' TO Q:FLDCMD
CASE Q:SELECT = '54'
  STORE 'MFG' TO Q:SELITEM
  STORE 'Q:FLDCMD2' TO Q:FLDCMD
CASE Q:SELECT = '55'
  STORE 'LOT' TO Q:SELITEM
  STORE 'Q:FLDCMD2' TO Q:FLDCMD
CASE Q:SELECT = '56'
  STORE 'DITEM' TO Q:SELITEM
  STORE 'Q:FLDCMD2' TO Q:FLDCMD
CASE Q:SELECT = '57'
  STORE 'WNTY' TO Q:SELITEM
  STORE 'Q:FLDCMD2' TO Q:FLDCMD
ENDCASE
IF Q:SELECT <> '00' .AND. Q:SELECT <> '30' .AND.;
  Q:SELECT <> '58'
  STORE Q:ITEM + 1 TO Q:ITEM
  IF Q:ITEM = 1
    STORE Q:SELECT TO Q:CHOSEN
    STORE Q:SELITEM TO Q:DISPLAY
  ELSE
    STORE Q:CHOSEN+' , '+Q:SELECT TO Q:CHOSEN
    STORE Q:DISPLAY+' , '+Q:SELITEM TO Q:DISPLAY
  ENDIF
  IF Q:SELECT <> '01'
    IF $(Q:SELITEM,1,1)='$' .AND. .NOT. Q:DATESEL
      STORE 'DATES' TO Q:SELITEM
      STORE &Q:FLDCMD-' , '+Q:SELITEM TO &Q:FIDCMD
      STORE ' ' TO Q:SELITEM
      STORE T TO Q:DATESEL
    ELSE
      IF $(Q:SELITEM,1,1)<>'$'
        STORE &Q:FLDCMD-' , '+Q:SELITEM TO &Q:FLDCMD
      ENDIF
    ENDIF
  ENDIF
ENDIF
ENDIF
***** Generate Code For Field Selection
IF Q:FIELD = ' '
  STORE Q:FIELD-Q:SELITEM TO Q:FIELD
ELSE
  STORE Q:FIELD-' , '-Q:SELITEM TO Q:FIELD
ENDIF
STORE '00' TO Q:SELECT
ENDIF
ENDIF
ENDIF
ENDDC

```



```

***** Release All Unnecessary Memory Variables
RELEASE Q:REPLY, Q:SELECT, Q:ITEM, Q:SELITEM, Q:SELCMD, ;
      Q:FLDCMD, Q:CHOSEN
STORE Q:NRPASSES TO Q:LOOPCNT
***** If Both OPEN and CLOSED Files are Selected
***** The Execution Loop Will Be Run Twice
DO WHILE Q:LOOPCNT >= 1
***** Select File To Be Used In This Query
IF Q:FILE = 'O'
  STCRE 'D:'+C:WHO-'OPEN' TO Q:TEMP3
ELSE
  STCRE 'D:'+C:WHO-'CLOS' TO Q:TEMP3
ENDIF
***** Generate Executable Code To Perform Query Selection
***** Check to See If The Selection Deals With The First
***** Half of the Data Base
IF Q:SELCMD1 <> ' '
  IF Q:FILE = 'O'
    STCRE 'USE D:OPEN1' TO &Q:LINE
    SICRE Q:CNTR + 1 TO Q:CNTR
    STCRE 'Q:L' + STR(Q:CNTR,2) TO Q:LINE
  ELSE
    STCRE 'USE D:CLOSE1' TO &Q:LINE
    SICRE Q:CNTR + 1 TO Q:CNTR
    STCRE 'Q:L' + STR(Q:CNTR,2) TO Q:LINE
  ENDIF
  STCRE 'D:'+C:WHO+'IMP1' TO Q:TEMP1
  STCRE 'CCPY TO '+Q:TEMP1 TO &Q:LINE
  IF Q:FLDCMD1 <> 'CASE '
    STCRE &Q:LINE+' FIELD '+Q:FLDCMD1+' FOR '+Q:SELCMD1;
      TO &Q:LINE
  ELSE
    STCRE &Q:LINE+' FOR '+Q:SELCMD1 TO &Q:LINE
  ENDIF
  SICRE Q:CNTR + 1 TO Q:CNTR
  STCRE 'Q:L'+STR(Q:CNTR,2) TO Q:LINE
ENDIF
***** Generate Executable Code To Perform Query Selection
***** Check to See If The Selection Deals With The Second
***** Half of the Data Base
IF Q:SELCMD2 <> ' '
  IF Q:FILE = 'C'
    STCRE 'USE D:OPEN2' TO &Q:LINE
    SICRE Q:CNTR + 1 TO Q:CNTR
    STCRE 'Q:L' + STR(Q:CNTR,2) TO Q:LINE
  ELSE
    STCRE 'USE D:CLOSE2' TO &Q:LINE
    SICRE Q:CNTR + 1 TO Q:CNTR
    STCRE 'Q:L' + STR(Q:CNTR,2) TO Q:LINE
  ENDIF
  STCRE 'D:'+C:WHO+'IMP2' TO Q:TEMP2
  STCRE 'COPY TO '+Q:TEMP2 TO &Q:LINE
  IF Q:FLDCMD2 <> 'CASE '
    STCRE &Q:LINE+' FIELD '+Q:FLDCMD2+' FOR '+Q:SELCMD2;
      TO &Q:LINE
  ELSE
    STCRE &Q:LINE+' FOR '+Q:SELCMD2 TO &Q:LINE
  ENDIF
  SICRE Q:CNTR + 1 TO Q:CNTR
  STCRE 'Q:L'+STR(Q:CNTR,2) TO Q:LINE

```


ENDIF

***** Generate Code to Join Together Both Halves of the
***** Selected Files

```
IF Q:SELCMD1 <> ' ' .AND. Q:SELCMD2 <> ' '  
  STCRE 'SELECT PRIMARY' TO &Q:LINE  
  STCRE Q:CNTR + 1 TO Q:CNTR  
  STCRE 'Q:L'+STR(Q:CNTR,2) TO Q:LINE  
  STCRE 'USE '+Q:TEMP1 TO &Q:LINE  
  STCRE Q:CNTR + 1 TO Q:CNTR  
  STCRE 'Q:L'+STR(Q:CNTR,2) TO Q:LINE  
  STCRE 'SELECT SECONDARY' TO &Q:LINE  
  STCRE Q:CNTR + 1 TO Q:CNTR  
  STCRE 'Q:L'+STR(Q:CNTR,2) TO Q:LINE  
  STCRE 'USE '+Q:TEMP2 TO &Q:LINE  
  STCRE Q:CNTR + 1 TO Q:CNTR  
  STCRE 'Q:L'+STR(Q:CNTR,2) TO Q:LINE  
  STCRE 'JCIN TO '+Q:TEMP3+' FOR P.CASE=S.CASE FIELD '+';  
    Q:FIELD TO &Q:LINE  
  STCRE Q:CNTR + 1 TO Q:CNTR  
  STCRE 'Q:L'+STR(Q:CNTR,2) TO Q:LINE  
  STCRE 'USE' TO &Q:LINE  
  STCRE Q:CNTR + 1 TO Q:CNTR  
  STCRE 'Q:L'+STR(Q:CNTR,2) TO Q:LINE  
  STCRE 'DELETE FILE D: '+Q:TEMP1-' .DBF' TO &Q:LINE  
  STCRE Q:CNTR + 1 TO Q:CNTR  
  STCRE 'Q:L'+STR(Q:CNTR,2) TO Q:LINE  
  STCRE 'DELETE FILE D: '+Q:TEMP2-' .DBF' TO &Q:LINE  
  STCRE Q:CNTR + 1 TO Q:CNTR  
  STCRE 'Q:L'+STR(Q:CNTR,2) TO Q:LINE
```

ELSE

```
IF Q:SELCMD1 <> ' ' .AND. Q:SELCMD2 = ' '
```

***** Generate Codes to Rename Files as Necessary

```
IF Q:FLDCMD2 = 'CASE'  
  STORE 'RENAME '+Q:TEMP1-' .DBF TO '+Q:TEMP3-' .DBF';  
  TO &Q:LINE  
  STCRE Q:CNTR + 1 TO Q:CNTR  
  STORE 'Q:L'+STR(Q:CNTR,2) TO Q:LINE  
ELSE  
  STORE 'SELECT SECONDARY' TO &Q:LINE  
  STCRE Q:CNTR + 1 TO Q:CNTR  
  STORE 'Q:L'+STR(Q:CNTR,2) TO Q:LINE  
  STCRE 'USE '+Q:TEMP1 TO &Q:LINE  
  STORE Q:CNTR + 1 TO Q:CNTR  
  STCRE 'Q:L'+STR(Q:CNTR,2) TO Q:LINE  
  STORE 'SELECT PRIMARY' TO &Q:LINE  
  STCRE Q:CNTR + 1 TO Q:CNTR  
  STORE 'Q:L'+STR(Q:CNTR,2) TO Q:LINE  
  IF Q:FILE = 'O'  
    STORE 'USE D:OPEN2' TO &Q:LINE  
    STORE Q:CNTR + 1 TO Q:CNTR  
    STORE 'Q:L'+STR(Q:CNTR,2) TO Q:LINE  
  ELSE  
    STORE 'USE D:CLOSE2' TO &Q:LINE  
    STORE Q:CNTR + 1 TO Q:CNTR  
    STORE 'Q:L'+STR(Q:CNTR,2) TO Q:LINE  
  ENDIF  
  STORE 'JOIN TO '+Q:TEMP3+;  
    ' FOR P.CASE=S.CASE FIELD '+Q:FIELD TO &Q:LINE  
  STORE Q:CNTR + 1 TO Q:CNTR  
  STCRE 'Q:L'+STR(Q:CNTR,2) TO Q:LINE  
  STORE 'USE' TO &Q:LINE  
  STCRE Q:CNTR + 1 TO Q:CNTR  
  STORE 'Q:L'+STR(Q:CNTR,2) TO Q:LINE  
  STORE 'DELETE FILE D: '+Q:TEMP1-' .DBF' TO &Q:LINE  
  STORE Q:CNTR + 1 TO Q:CNTR  
  STORE 'Q:L'+STR(Q:CNTR,2) TO Q:LINE
```



```

ENDIF
ELSE
IF Q:FLDCMD1 = 'CASE '
STORE 'RENAME '+Q:TEMP2-'.DBF TO '+Q:TEMP3-'.DEF';
TO &Q:LINE
STORE Q:CNTR + 1 TO Q:CNTR
STCRE 'Q:L'+STR(Q:CNTR,2) TO Q:LINE
ELSE
STORE 'SELECT SECONDARY' TO &Q:LINE
STORE Q:CNTR + 1 TO Q:CNTR
STCRE 'Q:L'+STR(Q:CNTR,2) TO Q:LINE
STORE 'USE '+Q:TEMP2 TO &Q:LINE
STORE Q:CNTR + 1 TO Q:CNTR
STCRE 'Q:L'+STR(Q:CNTR,2) TO Q:LINE
STORE 'SELECT PRIMARY' TO &Q:LINE
STORE Q:CNTR + 1 TO Q:CNTR
STCRE 'Q:L'+STR(Q:CNTR,2) TO Q:LINE
IF Q:FILE = 'O'
STORE 'USE D:OPEN1' TO &Q:LINE
STORE Q:CNTR + 1 TO Q:CNTR
STORE 'Q:L'+STR(Q:CNTR,2) TO Q:LINE
ELSE
STORE 'USE D:CLOSE1' TO &Q:LINE
STORE Q:CNTR + 1 TO Q:CNTR
STORE 'Q:L'+STR(Q:CNTR,2) TO Q:LINE
ENDIF
STCRE 'JOIN TO '+Q:TEMP3+' FOR P.CASE=S.CASE '
+'FIELD '+Q:FIELD TO &Q:LINE
STORE Q:CNTR + 1 TO Q:CNTR
STCRE 'Q:L'+STR(Q:CNTR,2) TO Q:LINE
STCRE 'USE' TO &Q:LINE
STORE Q:CNTR + 1 TO Q:CNTR
STORE 'Q:L'+STR(Q:CNTR,2) TO Q:LINE
STORE 'DELETE FILE D: '+Q:TEMP2-'.DBF' TO &Q:LINE
STCRE Q:CNTR + 1 TO Q:CNTR
STORE 'Q:L'+STR(Q:CNTR,2) TO Q:LINE
ENDIF
ENDIF
STORE Q:LOOPCNT-1 TO Q:LOOPCNT

***** If Two Passes Required, Switch to Closed File
***** For Second Pass
IF Q:NRPASSES = 2
STCRE 'C' TO Q:FILE
ENDIF
ENDDO

***** If Two Passes Required, Generate Code To Join Files
***** Created by Individual Passes
IF Q:NRPASSES = 2
STCRE 'USE D: '+C:WHC+'OPEN' TO &Q:LINE
STCRE Q:CNTR + 1 TO Q:CNTR
STCRE 'Q:L'+STR(Q:CNTR,2) TO Q:LINE
STCRE 'APPEND FROM '+D: '+C:WHO+'CLOS' TO &Q:LINE
STCRE Q:CNTR + 1 TO Q:CNTR
STCRE 'Q:L'+STR(Q:CNTR,2) TO Q:LINE
ENDIF
*SET COLCR TO 112, 6
ERASE
@ 10,23 SAY 'Your Query Is Now Being Processed'
@ 12,33 SAY 'Please Standby'

***** Begin Execution of Generated Code
STORE 10 TO Q:EXECNTR
DO WHILE Q:EXECNTR <= Q:CNTR-1
STCRE 'Q:L'+STR(Q:EXECNTR,2) TO Q:EXELINE

```



```

&&Q:EXELINE
STORE Q:EXECNTR + 1 TO Q:EXECNTR
ENDDO
STORE ""+'D:'+C:WHC+'OPEN.DBF'+"" TO Q:OPENFILE
STORE ""+'D:'+C:WHO+'CLOS.DBF'+"" TO Q:CLOSFILE
IF FILE(&Q:CPENFILE) .AND. FILE(&Q:CLOSFILE)
STORE 'D:'+C:WHO+'OPEN' TO Q:USEFILE
STORE 'D:'+C:WHO+'CLOS.DBF' TO Q:DELETEF
DELETE FILE &Q:DELETEF
ELSE
IF FILE(&Q:OPENFILE)
STORE 'D:'+C:WHC+'OPEN' TO Q:USEFILE
ELSE
STORE 'D:'+C:WHO+'CLOS' TO Q:USEFILE
ENDIF
ENDIF

***** Display The Number of Records Selected and
***** Provide the Option of Hard Copy or Screen Reports

USE &Q:USEFILE
GOTO EOTTCM
ERASE
@ 10,16 SAY #
@ 10,22 SAY 'Records Have Been Selected For This Query'
@ 13,30 SAY '1. Print Hard Copy'
@ 15,30 SAY '2. Display To Screen'
@ 17,30 SAY '3. Abort Query'
STORE ' ' TC Q:REPLY
@ 20,40 GET Q:REPLY
READ
DO WHILE Q:REPLY < '1' .OR. Q:REPLY > '3'
@ 23,33 SAY 'Enter 1, 2 or 3'+ CHR(7)
@ 20,40 GET Q:REPLY
READ
ENDDO
DO CASE
CASE Q:REPLY = '1'
SET PRINT ON
USE &Q:USEFILE
DO WHILE .NOT. EOF
ERASE
STORE 0 TO Q:LINENR
DO WHILE .NOT. EOF .AND. Q:LINENR <= 59
DISPLAY ALL FIELD &Q:DISPLAY OFF
SKIP
STORE Q:LINENR + 1 TO Q:LINENR
ENDDO
? CHR(12)
ENDDO
SET PRINT OFF
CASE Q:REPLY = '2'
ERASE
USE &Q:USEFILE
DISPLAY ALL FIELD &Q:DISPLAY OFF
WAIT
CASE Q:REPLY = '3'
USE
DELETE FILE &Q:USEFILE+'.DBF'
RELEASE ALL LIKE Q:*
RETURN
ENDCASE
USE
STORE Q:USEFILE+'.DBF' TO Q:USEFILE
DELETE FILE &Q:USEFILE
RELEASE ALL LIKE Q:*
RETURN

***** END OF PROGRAM

```



```

ACCEPT TC S:PASSWCFD
SET CCNSCLE ON
IF S:PASSWORD <> ' '
  USE D:TECHCODE INDEX D:TECH
  FIND &C:WHO
  IF PSWD <> S:PASSWORD .OR. # = 0
    @ 23,18 SAY 'Request ABORTED';
    + '- Strike Any Key To Continue'
    WAIT
    RELEASE ALL LIKE S:*
    RETURN
  ENLIF
ENDIF
ELSE
  RELEASE ALL LIKE S:*
  RETURN
ENDIF

***** Display Processing Message to User

ERASE
@ 6,27 SAY 'Internal Statistics Update'
@ 8,35 SAY 'In Process'
@ 16,26 SAY '***** DC NOT INTERRUPT *****'

SELECT PFIMARY
USE I:OPEN1 INDEX D:OCASE1

***** Begin Computing Loop

DO WHILE .NOT. EOF
  DO WHILE $(DATES,46,1) <> ' ' .AND. .NOT. EOF
    STCRE CASE TO S:CASE
    STCRE WHO TO S:WHO

    STCRE 0 TO S:ASSIGNED
    STCRE 0 TO S:ACTIVE
    STCRE 0 TO S:TRANSMIT
    STCRE 0 TO S:RESPOND
    STCRE 0 TO S:CLOSED

    STCRE $(DATES,46,1) TC S:DATECHG
    STCRE $(DATES,6,5) TO S:RECEIPT
    STCRE $(DATES,11,5) TC S:OPEN
    STCRE $(DATES,16,5) TO S:XMIT
    STCRE $(DATES,26,5) TC S:RESPONSE
    STCRE $(DATES,36,5) TO S:CLOSE

    STCRE 0 TO S:MAILDLAY
    STCRE 0 TO S:XMITDLAY
    STCRE 0 TO S:RESPDLAY
    STCRE 0 TO S:CLCSDLAY
    STCRE 0 TO S:PFCCDLAY

***** Calculate Time Span From Receipt to Case Open

IF S:RECEIPT <> ' ' .AND. S:OPEN <> ' '
  IF $(S:RECEIPT,1,2) < $(S:OPEN,1,2)
    IF VAL($(S:OPEN,1,2))-VAL($(S:RECEIPT,1,2))=1
      STORE VAL($(S:OPEN,3,3))+365-;
      VAL($(S:RECEIPT,3,3))+1 TO S:MAILDLAY
    ELSE
      IF VAL($(S:OPEN,1,2))-VAL($(S:RECEIPT,1,2))=2
        STCRE VAL($(S:OPEN,3,3))+730;
        -VAL($(S:RECEIPT,3,3))+1 TO S:MAILDLAY
      ELSE
        STCRE 999 TO S:MAILDLAY

```



```

        ENDIF
    ENDIF
ELSE
    STORE VAL($ (S:CPEN,3,3)) -VAL($ (S:RECEIPT,3,3));
    +1 TO S:MAILDLAY
ENDIF
ENDIF
***** Calculate Time Span From Case Open to Letter Transmit
IF S:XMIT <> ' '
    IF $(S:CPEN,1,2) < $(S:XMIT,1,2)
        IF VAL($ (S:XMIT,1,2)) -VAL($ (S:OPEN,1,2)) =1
            STORE VAL($ (S:XMIT,3,3)) +365 -;
            VAL($ (S:OPEN,3,3)) +1 TO S:XMITDLAY
        ELSE
            IF VAL($ (S:XMIT,1,2)) -VAL($ (S:OPEN,1,2)) =2
                STORE VAL($ (S:XMIT,3,3)) +730 -;
                VAL($ (S:OPEN,3,3)) +1 TO S:XMITDLAY
            ELSE
                STORE 999 TO S:XMITDLAY
            ENDIF
        ENDIF
    ELSE
        STORE VAL($ (S:XMIT,3,3)) -VAL($ (S:OPEN,3,3)) +1 ;
        TO S:XMITDLAY
    ENDIF
ENDIF
ENDIF

```

***** Calculate Time Span From Letter Transmit to Item
 ***** Manager Response

```

IF S:RESPONSE <> ' '
    IF $(S:XMIT,1,2) < $(S:RESPONSE,1,2)
        IF VAL($ (S:RESPONSE,1,2)) -VAL($ (S:XMIT,1,2)) =1
            STORE VAL($ (S:RESPONSE,3,3)) +365 -;
            VAL($ (S:XMIT,3,3)) +1 TO S:RESPDLAY
        ELSE
            IF VAL($ (S:RESPONSE,1,2)) -;
            VAL($ (S:XMIT,1,2)) = 2;
            STORE VAL($ (S:RESPONSE,3,3)) +730 -;
            VAL($ (S:XMIT,3,3)) +1 TO S:RESPDLAY
            ELSE
                STORE 999 TO S:RESPDLAY
            ENDIF
        ENDIF
    ELSE
        STORE VAL($ (S:RESPONSE,3,3)) -VAL($ (S:XMIT,3,3));
        + 1 TO S:RESPDLAY
    ENDIF
ENDIF
ENDIF

```

***** Calculate Time Span From Item Manager Response to
 ***** Case Close

```

IF S:CLOSE <> ' '
    IF $(S:RESPONSE,1,2) < $(S:CLOSE,1,2)
        IF VAL($ (S:CLOSE,1,2)) -VAL($ (S:RESPONSE,1,2)) =1
            STORE VAL($ (S:CLOSE,3,3)) +365 -;
            VAL($ (S:RESPONSE,3,3)) +1 TO S:CLOSDLAY
        ELSE
            IF VAL($ (S:CLOSE,1,2)) -;
            VAL($ (S:RESPONSE,1,2)) = 2;
            STORE VAL($ (S:CLOSE,3,3)) +730 -;
            VAL($ (S:RESPONSE,3,3)) +1 TO S:CLCSDLAY
            ELSE
                STORE 999 TO S:CLOSDLAY
            ENDIF
        ENDIF
    ELSE
        STORE 999 TO S:CLOSDLAY
    ENDIF
ENDIF
ENDIF

```



```

        STORE VAL($ (S:CLOSE,3,3)) -;
        VAL($ (S:RESPONSE,3,3)) +1 TO S:CLOSDLAY
    ENDIF

```

***** Calculate Time Span From Case Open to Case Close

```

    IF $(S:OPEN,1,2) < $(S:CLOSE,1,2)
        IF VAL($ (S:CLOSE,1,2)) - VAL($ (S:OPEN,1,2)) = 1
            STORE VAL($ (S:CLOSE,3,3)) + 365 -;
            VAL($ (S:OPEN,3,3)) +1 TO S:PROCDLAY
        ELSE
            IF VAL($ (S:CLOSE,1,2)) - VAL($ (S:OPEN,1,2)) = 2
                STCRE VAL($ (S:CLOSE,3,3)) + 730 -;
                VAL($ (S:OPEN,3,3)) + 1 TO S:PROCDLAY
            ELSE
                STORE 999 TO S:PROCDLAY
            ENDIF
        ENDIF
    ELSE
        STORE VAL($ (S:CLOSE,3,3)) - VAL($ (S:OPEN,3,3));
        + 1 TO S:PROCDLAY
    ENDIF
ENDIF
SELECT SECONDARY
USE D:QTIME INDEX D:QCASE

```

***** Update Time Frame File With New Time Spans
 ***** If a Record Does Not Exist For a Case, Create It

```

FIND &S:CASE
IF # = 0
    APPEND BLANK
    REPLACE CASE WITH S:CASE,WHO WITH S:WHO
    IF S:DATECHG <> 'N'
        STORE '9999' TO S:WHO
    ENDIF
ENDIF
IF S:DATECHG = 'N'
    STCRE 1 TO S:ASSIGNED
    STORE 1 TO S:ACTIVE
ENDIF

```

***** Change Status From Active to Transmitted or
 ***** Vice Versa

```

IF S:XMITDLAY > 0 .AND. XMITDLAY = 0
    STORE S:ACTIVE - 1 TO S:ACTIVE
    STCRE S:TRANSMIT + 1 TO S:TRANSMIT
ELSE
    IF S:XMITDLAY = 0 .AND. XMITDLAY <> 0
        STORE S:ACTIVE + 1 TO S:ACTIVE
        STORE S:TRANSMIT - 1 TO S:TRANSMIT
    ENDIF
ENDIF

```

***** Change Status From Transmitted to Responded or
 ***** Vice Versa

```

IF S:RESPDLAY > 0 .AND. RESPDLAY = 0
    STORE S:TRANSMIT - 1 TO S:TRANSMIT
    STCRE S:RESPOND + 1 TO S:RESPOND
ELSE
    IF S:RESPDLAY = 0 .AND. RESPDLAY <> 0
        STORE S:TRANSMIT + 1 TO S:TRANSMIT
        STORE S:RESPOND - 1 TO S:RESPOND
    ENDIF
ENDIF

```

***** Change Status From Responded to Closed or

***** Vice Versa

```
IF S:CLOSDLAY > 0 .AND. CLOSDLAY = 0
  STORE S:RESPOND - 1 TO S:RESPOND
  STORE S:CLOSED + 1 TO S:CLOSED
ELSE
  IF S:CLOSDLAY = 0 .AND. CLOSDLAY <> 0
    STORE S:RESPOND - 1 TO S:RESPOND
    STORE S:CLOSED + 1 TO S:CLOSED
  ENDIF
ENDIF
```

***** Update Time Span File

```
REPLACE CASE WITH S:CASE,WHO WITH S:WHO,MAILDLAY ;
WITH S:MAILDLAY,XMITDLAY WITH S:XMITDLAY,RESPDLAY WITH ;
S:RESPDLAY,CLOSDLAY WITH S:CLOSDLAY,PROCDLAY WITH S:PROCDLAY
```

```
SELECT SECONDARY
USE D:TECHCODE INDEX D:TECH
```

***** Update Techcode File

```
FIND &S:WHO
IF # <> 0
  REPLACE ASSIGNED WITH ASSIGNED + S:ASSIGNED,;
  ACTIVE WITH ACTIVE + S:ACTIVE,TRANSMIT WITH TRANSMIT ;
+ S:TRANSMIT,RESPOND WITH RESPOND+S:RESPOND,CLOSED WITH ;
CLOSED + S:CLOSED
ENDIF
ENDIF
SELECT PRIMARY
REPLACE DATES WITH $(DATES,1,45)+' '
SKIP
ENDDO
SKIP
ENDDO
```

***** Release All Local Memory Variables and All Files
***** Used During Processing

```
SELECT PRIMARY
USE
SELECT SECONDARY
USE
RELEASE ALL LIKE S:*
RETURN
```

***** END OF PROGRAM

XIV. JULIAN DATE CONVERSION MODULE

```

*****
**
** Date: 18 October 1984
** Version: 1.0
** Module Name: OJULIAN
** Module Purpose: Convert Date (MMDDYY) to Julian
** Module Interface Definition
**   Inputs: V:MM, V:DD, V:YY
**   Outputs: V:JULDATE
** Module Processing Narrative Description:
**
**   Receives a date in MMDDYY format and converts
**   it to a Julian date and returns the date to
**   the calling program.
**
** Superordinate Modules: XOPEN2
** Subordinate Modules: None
** Author: J.G. BOYNTON
**
*****

```

```

DO CASE
CASE V:MM = 01
  STORE V:DD TO V:DAY
CASE V:MM = 02
  STORE V:DD + 31 TO V:DAY
CASE V:MM = 03
  STORE V:DD + 59 TO V:DAY
CASE V:MM = 04
  STORE V:DD + 90 TO V:DAY
CASE V:MM = 05
  STORE V:DD + 120 TO V:DAY
CASE V:MM = 06
  STORE V:DD + 151 TO V:DAY
CASE V:MM = 07
  STORE V:DD + 181 TO V:DAY
CASE V:MM = 08
  STORE V:DD + 212 TO V:DAY
CASE V:MM = 09
  STORE V:DD + 243 TO V:DAY
CASE V:MM = 10
  STORE V:DD + 273 TO V:DAY
CASE V:MM = 11
  STORE V:DD + 304 TO V:DAY
CASE V:MM = 12
  STORE V:DD + 334 TO V:DAY
ENDCASE
IF INT(V:YY/4) * 4 = V:YY .AND. V:DAY >= 60
  IF V:MM= 02 .AND. V:DD= 29
    STORE V:DAY TO V:DAY
  ELSE
    STORE V:DAY + 1 TO V:DAY
  ENDIF
ENDIF
STORE V:YY * 1000 + V:DAY TO V:JULIAN
STORE STR(V:JULIAN,5) TO V:JULDATE
RETURN

***** END OF PROGRAM

```


XV. COG COUNT MODULE

```

*****
**
** Date: 8 Jan 1984
** Version: 1.0
** Module Name: COGCNT
** Module Purpose: Count the Active Cases Assigned to
**                  Any Given COG
**
** Module Interface Definition
**   Inputs: None
**   Outputs: None
**
** Module Processing Narrative Description:
**
**   Indexes the OPEN1 File by COG and Counts
**   The Number of Cases Assigned To Each COG
**
** Superordinate Modules: UTILMENU
** Subordinate Modules: None
** Author: R. G. NICHOLS
**
*****

```

***** Display Warning Message and Accept Continue Request

```

ERASE
@ 1,24 SAY '***** COG Count Processing *****'
@ 3,24 SAY '* * * * *'
@ 4,24 SAY '*
@ 5,24 SAY '                WARNING
@ 6,24 SAY '*
@ 7,24 SAY '*      This Program Will Read
@ 8,24 SAY '*      All Records While
@ 9,24 SAY '*      Processing
@ 10,24 SAY '*
@ 11,24 SAY '*      If Existing Files Are
@ 12,24 SAY '*      Large, This Could Take
@ 13,24 SAY '*      Hours
@ 14,24 SAY '*
@ 15,24 SAY '* * * * *
@ 17,24 SAY '*      Are You SURE You Want To'
@ 18,24 SAY '*      Continue'
@ 19,24 SAY '*      <Enter Y or N>' + CHR(7)
STORE ' ' TO CC:REPLY2
@ 21,40 GET CC:REPLY2
READ
DO WHILE !(CC:REPLY2) <> 'Y' .AND. !(CC:REPLY2) <> 'N'
  @ 23,32 SAY 'Enter Y or N Only' + CHR(7)
  @ 21,40 GET CC:REPLY2 PICTURE 'A'
  READ
ENDDO
@ 23,32 SAY '
@ 17,40 SAY '

```

***** Prompt For and Accept Password Verification

```

IF CC:REPLY2 = 'Y'
  @ 21,30 SAY 'Enter Your Password '
  STORE ' ' TO CC:PSWD
  SET CCNSCLE OFF
  ACCEPT TO CC:PSWD

```



```

SET CCNSCLE ON
IF CC:PSWD <> '
  USE D:TECHCODE INDEX D:TECH
  FIND &C:WHO
  IF PSWD <> CC:PSWD .OR. # = 0
    @ 23,18 SAY 'Request ABORTED - Strike Any Key To Continue'
    WAIT
    RELEASE ALL LIKE CC:*
    RETURN
  ENDDIF
ENDIF
ELSE
  RELEASE ALL LIKE CC:*
  RETURN
ENDIF

```

***** Begin Statistics Update

```

ERASE
@ 12,20 SAY '          CCG STATISTICS BEING PROCESSED'
@ 14,20 SAY '          PLEASE STANDBY'
@ 20,20 SAY '***** DC NOT INTERRUPT WHILE PROCESSING *****'

```

```

SELECT PRIMARY
USE D:OPEN1 INDEX D:CCGCNT
REINDEX
SELECT SECONDARY
USE D:COG INDEX D:COGS
REPLACE COUNT WITH 0 FOR COUNT <> 0
SELECT PRIMARY
GOTO 2

```

***** Count the COGs Assigned Until End Of File Found

```

DO WHILE .NOT. EOF
  STORE CCG TO CC:CURRENT
  STORE 0 TO CC:COUNT

```

***** Increment Counter Until a Different COG or End Of
***** File Found

```

DO WHILE COG = CC:CURRENT .AND. .NOT. EOF
  STORE CC:CCUNT + 1 TO CC:COUNT
  SKIP
ENDDO
SELECT SECONDARY

```

***** Update IM Record

```

FIND &CC:CURRENT
IF # <> 0
  REPLACE COUNT WITH CC:COUNT
ENDIF
SELECT PRIMARY
ENDDO
USE
SELECT SECCNDARY
USE
RELEASE ALL LIKE CC:*
RETURN

```

***** END OF PROGRAM

XVI. BI-WEEKLY STATISTICS REPORT MODULE

```

*****
**
** DATE: 27 JANUARY 1984
** VERSION: 1.0
** MODULE NAME: XXBISTAT
** MCDUIE PURPOSE: CALCULATE EI-WEEKLY STATISTICS
** MODULE INTERFACE DEFINITION
** INPUTS: C:WHO, C;JULIAN
** OUTPUTS:
** MODULE PROCESSING NARRATIVE DESCRIPTION:
**
** ACCEPTS CLOSING DATE FOR THE REPORT IN MMDDYY
** FORMAT. DATE IS CONVERTED TO JULIAN FORMAT BY
** CALLING OJULIAN. DATES FOR PREVIOUS YEAR AND
** CIDEST YEAR ARE CALCULATED AND STORED INTO
** MEMORY VARIABLEES. OPEN AND CLOSE DATABASES ARE
** SEARCHED SEQUENTIALLY FOR ANY CASES WHICH WERE
** OPENED OR CICSSED DURING THE PERIOD IN QUESTION.
** THE BIWKSTAT DATABASE IS READ FOR THE COUNTS
** OF THE LAST REPORT TO CALCULATE THE TREND, AND
** THEN THE CURRENT COUNTS ARE PLACED INTO THE
** BIWKSTAT DATABASE FOR FUTURE REFERENCE. THE
** REPORT IS THEN PRINTED USING THE COUNTS FROM
** THIS PROCESSING. THE PROGRAM SHOULD BE RUN
** IN BATCH, DURING 'OFF' HOURS, AND ONLY ON THE
** SPECIFIC DAY FOR THE CUTOFF TO KEEP THE TREND
** DATA REAL.
** SUPERORDINATE MODULES: SUPRPTS
** SUEORDINATE MODULES: NONE
** AUTHCR: J.G. BOYNTON
**
*****

```

***** INITIALIZATION OF VARIABLES

```

STORE ' ' TO BW:CURR
STORE ' ' TO BW:PREV
STORE ' ' TO BW:CID
STORE 0 TO BW:CREC
STORE 0 TO EW:CCLOS
STORE 0 TO EW:C9COG
STORE 0 TO EW:CSFCC
STORE 0 TO EW:PREC
STORE 0 TO EW:PIN
STORE 0 TO EW:PCLOS
STORE 0 TO EW:P9COG
STORE 0 TO EW:PSFCC
STORE 0 TO EW:OREC
STORE 0 TO EW:OCLOS
STORE 0 TO EW:OCIOS
STORE 0 TO BW:O9COG
STORE 0 TO EW:OSFCC
STORE 0 TO BW:CRECC
STORE 0 TO EW:PRECC
STORE 0 TO BW:ORECC
STORE 0 TO BW:OPERR
STORE 0 TO EW:CERROR
STORE 0 TO EW:PEERROR
STORE 0 TO EW:OERROR

```



```

***** THIS SEQUENCE CALCULATES THE UPPER AND LOWER YEARS
***** FOR INPUT AND IS BASED ON THE CURRENT JULIAN DATE
***** C:JULIAN. BW:LLIMIT= YEAR MINUS TWO YEARS
***** BW:ULIMIT = YEAR PLUS ONE YEAR

```

```

STORE $(C:JULIAN,1,2) TO TEMP1
STORE VAL(TEMP1) TO TEMP1A
STORE VAL('2') TO LOW
STORE VAL('1') TO HIGH
STORE TEMP1A-LOW TO LLMT
STORE TEMP1A+HIGH TO ULMT
STORE STR(LLMT,2) TO EW:LLIMIT
STORE STR(ULMT,2) TO EW:ULIMIT
RELEASE TEMP1,TEMP1A,LOW,HIGH,LLMT,ULMT

```

```

STORE ' ' TO EW:EDATE
STORE T TO EW:CHOOSE
ERASE
DO WHILE BW:CHOOSE

```

```

    @ 10,20 SAY 'PLEASE ENTER THE CLOSING DATE'
    @ 11,20 SAY ' FOR THIS BIWEEKLY REPORT '
    @ 12,20 SAY ' <MMDDYY> '
    @ 14,30 GET BW:EDATE PICTURE '999999'
    READ
    IF $(BW:EDATE,1,2) <'01' ;
        .OR. $(BW:EDATE,1,2) >'12' ;
        .OR. $(BW:EDATE,3,2) <'01' ;
        .OR. $(BW:EDATE,3,2) >'31' ;
        .OR. $(BW:EDATE,5,2) < BW:LLIMIT ;
        .OR. $(BW:EDATE,5,2) > BW:ULIMIT
        @ 23,30 SAY 'DATE OUT OF RANGE'
    ELSE
        STORE F TO EW:CHOOSE
    ENDIF
ENDDO<BW:CHOOSE>
@ 23,30 SAY '
RELEASE BW:CHOOSE, BW:LLIMIT, BW:ULIMIT

```

```

***** CALCULATE THE DATES TO BE SEARCHED FOR AND ASSIGN
***** THEM TO THE VARIABLES: BW:CURR, BW:PREV, BW:OLD
***** ENTER THE CALL TO C:OJULIAN TO CHANGE MMDDYY TO
***** JULIAN FORMAT

```

```

STORE VAL($(BW:EDATE,1,2)) TO V:MM
STORE VAL($(BW:EDATE,3,2)) TO V:DD
STORE VAL($(BW:EDATE,5,2)) TO V:YY
DC C:OJULIAN
STORE V:JULDATE TO BW:CURR
RELEASE ALL LIKE V:*

```

```

STORE $(BW:CURR,1,2) TO BW:TYR
STORE VAL(EW:TYR) TO BW:TYR3
STORE BW:TYR3-1 TO BW:TYR1
STORE BW:TYR3-2 TO BW:TYR2
STORE STR(EW:TYR1,2) TO BW:PREVT
STORE STR(EW:TYR2,2) TO BW:OLDT
STORE BW:PREVT+$(BW:CURR,3,3) TO BW:PREV
STORE BW:OLDT+$(BW:CURR,3,3) TO BW:OLD
RELEASE BW:PREVT, BW:OLDT, BW:TYR, BW:TYR1, BW:TYR2

```

```

ERASE
@ 12,20 SAY ' BIWEEKLY STATUS REPORT IS BEING ' ;
        +' PROCESSED '
@ 14,20 SAY ' PLEASE STANDBY '
@ 23,20 SAY ' ***** DO NOT INTERRUPT WHILE '

```


+ 'PROCESSING ****'

***** END DATE CHANGE AND ASSIGNMENT HERE

USE D:OPEN1

DO WHILE .NCT. EOF

STORE DATES TO M:DATES
STORE COG TO M:COG

STORE \$(M:DATES,11,5) TO BW:ODAT
STORE \$(M:DATES,36,5) TO BW:CDAT
IF \$(BW:CDAT,1,2) = \$(BW:CURR,1,2)
STORE BW:CREC + 1 TO BW:CREC
IF BW:CDAT <> ' '

STORE BW:CCLOS + 1 TO BW:CCLOS

ELSE

IF \$(M:COG,1,1) = '9'

STORE EW:C9COG + 1 TO BW:C9COG

ELSE

STORE BW:CSPCC + 1 TO BW:CSPCC

ENDIF

ENDIF

ENDIF < THIS CASE IN CURRENT YEAR COUNT >

IF \$(BW:CDAT,1,2) = \$(BW:PREV,1,2)

STORE BW:PREC + 1 TO BW:PREC

IF BW:CDAT <> ' '

STORE BW:PCLOS + 1 TO BW:PCLOS

ELSE

IF \$(M:COG,1,1) = '9'

STORE EW:P9COG + 1 TO BW:P9COG

ELSE

STORE EW:PSPCC + 1 TO BW:PSPCC

ENDIF

ENDIF

ENDIF < THIS CASE IN PREVIOUS YEAR COUNT >

IF \$(BW:CDAT,1,2) = \$(BW:OLD,1,2)

STORE BW:OREC + 1 TO BW:OREC

IF BW:CDAT <> ' '

STORE BW:OCLOS + 1 TO BW:OCLOS

ELSE

IF \$(M:COG,1,1) = '9'

STORE BW:O9COG + 1 TO BW:O9COG

ELSE

STORE BW:OSPCC + 1 TO BW:OSPCC

ENDIF

ENDIF

ENDIF < THIS CASE IN OLDEST YEAR COUNT >

IF \$(BW:CDAT,1,2) <> \$(BW:CURR,1,2) .AND. ;

\$(BW:CDAT,1,2) <> \$(BW:PREV,1,2) .AND. ;

\$(BW:CDAT,1,2) <> \$(BW:OLD,1,2)

STORE BW:OPERR + 1 TO BW:OPERR

ENDIF

SKIP

ENDDO <SEARCH OF OPEN1.DBF >

***** END OF THE OPENFILE SEARCH, NOW FOR THE CLOSED FILES

USE D:CLOSE1

DO WHILE .NCT. EOF

STORE DATES TO M:DATES
STORE COG TO M:COG


```

STORE $(M:DATES,11,5) TO BW:ODAT
STORE $(M:DATES,36,5) TO BW:CDAT
IF $(BW:ODAT,1,2) = $(BW:CURR,1,2)
  STORE BW:CRECC + 1 TO BW:CRECC
  IF BW:CDAT <> ' '
    STORE BW:CERROR + 1 TO BW:CERROR
  ENDIF
ENDIF < THIS CASE IN CURRENT YEAR COUNT >

IF $(BW:CDAT,1,2) = $(BW:PREV,1,2)
  STORE BW:PRECC + 1 TO BW:PRECC
  IF BW:CDAT <> ' '
    STORE BW:PERROR + 1 TO BW:PERROR
  ENDIF
ENDIF < THIS CASE IN PREVIOUS YEAR COUNT >

IF $(BW:CDAT,1,2) = $(BW:OLD,1,2)
  STORE BW:ORECC + 1 TO BW:ORECC
  IF BW:CDAT <> ' '
    STORE BW:OERROR + 1 TO BW:OERROR
  ENDIF
ENDIF < THIS CASE IN OLDEST YEAR COUNT >
SKIP

ENDDO <SEARCH OF CLOSE1.DBF >

STORE BW:C9CCG + BW:CSPCC TO BW:CTOT
STORE BW:P9CCG + BW:PSPCC TO BW:PTOT
STORE BW:O9CCG + BW:OSPCC TO BW:OTOT
STORE BW:OERROR+BW:PERROR+BW:CERROR TO BW:TERROR

STORE '19' + $(BW:CURR,1,2) TO BW:CYEAR
STORE '19' + $(BW:PREV,1,2) TO BW:PYEAR
STORE '19' + $(BW:OLD,1,2) TO BW:OYEAR

USE D:BIWKSTAT

STORE TOTALS TO BW:SCTOT
SKIP
STORE TOTALS TO BW:SPTOT
SKIP
STORE TOTALS TO BW:SOTOT

STORE ' ' TO BW:CLABEL
STORE ' ' TO BW:PLABEL
STORE ' ' TO BW:OLABEL

IF BW:SCTOT < BW:CTOT
  STORE 'UP' TO BW:CLABEL
ENDIF
IF BW:SCTOT > BW:CTOT
  STORE 'DOWN' TO BW:CLABEL
ENDIF
IF BW:SPTOT < BW:PTOT
  STORE 'UP' TO BW:PLABEL
ENDIF
IF BW:SPTOT > BW:PTOT
  STORE 'DOWN' TO BW:PLABEL
ENDIF
IF BW:SOTOT < BW:OTOT
  STORE 'UP' TO BW:OLABEL
ENDIF
IF BW:SOTOT > BW:OTOT
  STORE 'DOWN' TO BW:OLABEL
ENDIF

STORE BW:SOTOT-BW:OTOT TO BW:OTRD
STORE BW:SPTOT-BW:PTOT TO BW:PTRD

```


STORE BW:SCIOT-BW:CTCT TO BW:CTRD

SET FORMAT TO PRINT

```
@ 2,30 SAY ' CODE 9142 TECHNICAL BRANCH'
@ 4,30 SAY ' QUALITY DEFICIENT MATERIAL'
@ 6,30 SAY ' BIWEEKLY STATUS REPORT '
@ 8,30 SAY ' THRU'
@ 8,47 SAY '$(BW:EDATE,1,2) + '/' + $(BW:EDATE,3,2) + ;
        '/' + '$(BW:EDATE,5,2)
@ 10,30 SAY ' JULIAN DATE'
@ 10,50 SAY EW:CURR
@ 12,10 SAY '
        +' SPCC 9-COG TOTAL
@ 13,10 SAY 'CALENDAR CASES CASES
        +' CPEN OPEN OPEN TREND'
@ 14,10 SAY ' YEAR RECEIVED CLOSED
        +' CASES CASES CASES
```

```
@ 18,13 SAY EW:OYEAR
STCRE BW:OREC+BW:ORECC TO BW:TOREC
@ 18,18 SAY EW:TOR EC
@ 18,30 SAY EW:ORECC
@ 18,40 SAY EW:OSPCC
@ 18,50 SAY EW:O9COG
@ 18,60 SAY EW:OTOT
@ 18,70 SAY EW:OTRD
@ 18,80 SAY EW:OLABEL
```

```
@ 20,13 SAY EW:PYEAR
STCRE BW:PREC+BW:PRECC TO BW:TPREC
@ 20,18 SAY EW:TPREC
@ 20,30 SAY EW:PRECC
@ 20,40 SAY EW:PSPCC
@ 20,50 SAY EW:P9COG
@ 20,60 SAY EW:PTOT
@ 20,70 SAY EW:PTRD
@ 20,80 SAY EW:PLABEL
```

```
@ 22,13 SAY EW:CYEAR
STCRE BW:CREC+BW:CRECC TO BW:TCREC
@ 22,18 SAY EW:TCREC
@ 22,30 SAY EW:CRECC
@ 22,40 SAY EW:CSPCC
@ 22,50 SAY EW:C9COG
@ 22,60 SAY EW:CTOT
@ 22,70 SAY EW:CTRD
@ 22,80 SAY EW:CLABEL
```

```
@ 36,24 SAY ' CASE INPUT COMPARISON'
@ 40,25 SAY BW:PYEAR
@ 40,40 SAY BW:CYEAR
@ 40,55 SAY 'TREND'
STCRE BW:PREC+BW:PRECC TO BW:TPREC
@ 42,18 SAY EW:TPREC
@ 42,32 SAY EW:CREC
STCRE ' TO BW:TLABEL
IF BW:TPREC > BW:CREC
    STORE 'DCWN' TO BW:TLABEL
ENDIF
IF BW:TPREC < BW:CREC
    STORE 'UP' TO BW:TLABEL
ENDIF
STORE BW:CREC-BW:TPREC TO BW:TTRD
@ 42,48 SAY EW:TTRD
@ 42,59 SAY EW:TLABEL
```

EJECT

***** PAGE TWO

STCRE BW:CREC+BW:PREC+BW:OREC TO BW:TREC
STCRE BW:CRECC+BW:PRECC+BW:ORECC TO BW:TRECC

@ 2,30 SAY ' CODE 9142 TECHNICAL BRANCH'
@ 4,30 SAY ' QUALITY DEFICIENT MATERIAL'
@ 6,30 SAY ' BIWEEKLY STATUS REPORT '
@ 8,30 SAY ' THRU '
@ 8,47 SAY '\$(BW:EDATE,1,2) + '/' + \$(BW:EDATE,3,2) ;
+ '/' + \$(BW:EDATE,5,2)
@ 10,33 SAY ' JULIAN DATE '
@ 10,52 SAY BW:CURR

@ 14,25 SAY 'TOTAL RECORDS ON OPEN FILE: '
@ 14,70 SAY EW:TREC
@ 16,25 SAY 'TOTAL RECORDS ON CLOSED FILES: '
@ 16,70 SAY EW:TRECC
@ 18,25 SAY 'RECORDS WITH INVALID DATES, OPEN FILE: '
@ 18,69 SAY EW:OPERR - 1
@ 20,25 SAY 'RECORDS WITH INVALID DATES, CLOSED '
+ 'FILE: '
@ 20,70 SAY EW:TERROR
@ 28,40 SAY 'END OF REPORT '

EJECT
SET FORMAT TO SCREEN

***** STUFF NEW COUNIS INTO THE BIWKSTAT DATABASE

USE D:BIWKSTAT.DBF
REPLACE TOTALS WITH BW:CTRD
SKIP
REPLACE TOTALS WITH BW:PTRD
SKIP
REPLACE TOTALS WITH BW:OTRD

RELEASE ALL LIKE BW:*

ERASE
RETURN

***** END OF PROGRAM

XVII. MONTHLY STATISTICS REPORT MODULE

```
*****  
**  
** DATE: 8 JANUARY 1984 **  
** VERSICN: 1.0 **  
** MODULE NAME: XYMNSTAT **  
** MODULE PURPOSE: CALCULATE MONTHLY STATISTICS REPORT **  
** MODULE INTERFACE DEFINITION **  
** INPUTS: C:WHC, C:JULIAN **  
** OUTPUTS: NONE **  
** MODULE PROCESSING NARRATIVE DESCRIPTION: **  
**  
** ACCEPTS THE ENDING DATE AND THEN CALCULATES THE **  
** JULIAN DATE FOR THIS YEAR AND THE PRIOR TWO **  
** YEARS. THE OPEN AND CLOSE DATA BASES ARE **  
** SEARCHED SEQUENTIALLY TO FIND THE STATUS OF **  
** EACH CASE IN THE DESIGNATED TIME PERIODS AND **  
** COUNTS ARE SUMMARIZED INTO MEMORY VARIABLES. **  
** AFTER PROCESSING, THE REPORTS ARE GENERATED TO **  
** THE PRINTER. THIS INCLUDES THE MONTHLY STATUS **  
** REPORT BY YEAR, COMMAND KEY INDICATORS FOR **  
** CURRENT YEAR, AND THE SUMMARY REPORT FOR THE **  
** CURRENT YEAR. THIS SHOULD BE DONE 'OFF' TIME **  
** AND WHEN THE SYSTEM IS NOT BEING USED. OUTPUT **  
** IS DIRECTED TO THE PRINTER. **  
** ERROR LISTING CAN BE RETREIVED IN D:MSBAD.TXT **  
** BY 'TYPING' USING THE OPERATING SYSTEM. **  
**  
** SUPERORDINATE MODULES: SUPRPTS **  
** SUBORDINATE MODULES: NONE **  
** AUTHCR: J.G. BOYNTON **  
**  
*****
```

```
SET ALTERNATE TO D:MSEAD  
STORE 1 TO MS:ROW
```

```
***** INITIALIZATION OF VARIABLES
```

```
STORE 0 TO MS:CRCVD  
STORE 0 TO MS:OPTOT  
STORE 0 TO MS:CLTOT  
STORE 0 TO MS:OPERR  
STORE 0 TO MS:CLERR
```

```
***** THIS SEQUENCE CALCULATES THE UPPER AND LOWER YEARS  
***** FOR INPUT AND IS BASED ON THE CURRENT JULIAN DATE  
***** C:JULIAN. MS:LLIMIT= YEAR MINUS TWO YEARS  
***** MS:ULIMIT = YEAR PLUS ONE YEAR
```

```
STORE $(C:JULIAN,1,2) TO TEMP1  
STORE VAI(TEMP1) TO TEMP1A  
STORE VAI('2') TO LOW  
STORE VAI('1') TO HIGH  
STORE TEMP1A-LOW TO LLMT  
STORE TEMP1A+HIGH TO ULMT  
STORE STR(LLMT,2) TO MS:LLIMIT  
STORE STR(ULMT,2) TO MS:ULIMIT  
RELEASE TEMP1,TEMP1A,LOW,HIGH,LLMT,ULMT
```

```
***** INPUT OF REPORT CLOSING DATE
```



```

STORE ' ' TO MS:EDATE
STORE T TO MS:CHOOSE
ERASE
DO WHILE MS:CHOOSE
  @ 10,20 SAY ' PLEASE ENTER THE CLOSING DATE'
  @ 11,20 SAY '   FOR THIS MONTHLY REPORT '
  @ 12,20 SAY '   <MMDDYY>'
  @ 14,31 GET MS:EDATE PICTURE '999999'
  READ
  IF $(MS:EDATE,1,2) < '01' ;
    .OR. $(MS:EDATE,1,2) > '12' ;
    .OR. $(MS:EDATE,3,2) < '01' ;
    .OR. $(MS:EDATE,3,2) > '31' ;
    .OR. $(MS:EDATE,5,2) < MS:LLIMIT ;
    .OR. $(MS:EDATE,5,2) > MS:ULIMIT
    @ 23,30 SAY ' DATE OUT OF RANGE'
  ELSE
    STORE F TO MS:CHOOSE
  ENDIF
ENDDO <MS:CHOOSE>
@ 23,30 SAY '
RELEASE MS:CHOOSE, MS:LLIMIT, MS:ULIMIT

```

***** CALCULATE DATES TO BE SEARCHED FOR AND ASSIGNS THEM
 ***** TO VARIABLES

***** ENTER THE CALL TO C:OJULIAN TO CHANGE MMDDYY TO
 ***** JULIAN FORMAT

```

STORE VAL($ (MS:EDATE,1,2) ) TO V:MM
STORE VAL($ (MS:EDATE,3,2) ) TO V:DD
STORE VAL($ (MS:EDATE,5,2) ) TO V:YY
DC C:OJULIAN
STORE V:JULDATE TO MS:CJUL

```

***** THIS CALCULATES THE JULIAN DATE OF THE FIRST DAY
 ***** OF THE MONTH OF INTEREST

```

STORE VAL('01') TO V:DD
DC C:OJULIAN
STORE V:JULDATE TO MS:CJUL1

```

```

STORE $(MS:CJUL,1,2) TO MS:TYR
STORE VAL(MS:TYR) TO MS:TYR3
STORE MS:TYR3-1 TO MS:TYR1
STORE MS:TYR3-2 TO MS:TYR2
STORE STR(MS:TYR1,2) TO MS:PREVT
STORE STR(MS:TYR2,2) TO MS:OLDT

```

***** CALCULATE THE CALENDAR AND JULIAN DATES FOR THE
 ***** PREVIOUS YEAR

```

STORE VAL($ (MS:EDATE,1,2) ) TO V:MM
STORE VAL($ (MS:EDATE,3,2) ) TO V:DD
STORE VAL(MS:PREVT) TO V:YY
DC C:OJULIAN
STORE V:JULDATE TO MS:PJUL

```

***** CALCULATES FIRST DAY OF MONTH IN PREVIOUS YEAR

```

STORE VAL('01') TO V:DD
DC C:OJULIAN
STORE V:JULDATE TO MS:PJUL1

```

***** CALCULATES ENDDATE OF MONTH IN OLDEST YEAR

```

STORE VAL($ (MS:EDATE,1,2) ) TO V:MM

```



```
STORE VAL($ (MS:EDATE,3,2) ) TO V:DD
STORE VAL(MS:OLDT) TO V:YY
DC C:OJULIAN
STORE V:JUIDATE TO MS:OJUL
```

***** CALCULATES FIRST DAY OF MONTH IN OLDEST YEAR

```
STORE VAL('01') TO V:DD
DC C:OJULIAN
STORE V:JUIDATE TO MS:OJUL1
```

```
RELEASE MS:PREVT, MS:OLDT, MS:TYR, MS:TYR1, MS:TYR2;
MS:TYR3
RELEASE ALL LIKE V:*
```

```
ERASE
@ 12,20 SAY ' MONTHLY STATUS REPORT IS BEING ;
+'PROCESSED'
@ 14,20 SAY ' PLEASE STANDBY '
@ 23,20 SAY '***** DO NOT INTERRUPT WHILE ;
+'PROCESSING *****'
```

***** END DATE CHANGE AND ASSIGNMENT HERE

***** SEARCH THE OPEN DATABASE

```
STORE 0 TO MS:CO1
STORE 0 TO MS:CO2
STORE 0 TO MS:CO3
STORE 0 TO MS:CO4
STORE 0 TO MS:PO1
STORE 0 TO MS:PO2
STORE 0 TO MS:PO3
STORE 0 TO MS:PO4
STORE 0 TO MS:OO1
STORE 0 TO MS:OO2
STORE 0 TO MS:OO3
STORE 0 TO MS:OO4
STORE 0 TO MS:PTOT
STORE 0 TO MS:OTOT
```

USE D:OPEN1

```
DO WHILE .NCT. EOF
STORE DATES TO M:DATES
```

```
STORE $(M:DATES,11,5) TO MS:ODAT
STORE $(M:DATES,36,5) TO MS:CDAT
```

***** IF CASE IS IN CURRENT YEAR

```
IF $(MS:CDAT,1,2) = $(MS:CJUL,1,2)
```

```
IF MS:CDAT > MS:CJUL1 .AND. MS:ODAT < MS:CJUL
STORE MS:CRCVD + 1 TO MS:CRCVD
```

***** IF CASE IS STILL OPEN IT SHOULD BE IN THIS FILE

```
IF MS:CDAT = '
STORE VAL($ (MS:CJUL,3,3) ) TO V:CDAT
STORE VAL($ (MS:ODAT,3,3) ) TO V:ODAT
STORE V:CDAT - V:ODAT TO V:TIMEN
STORE STR(V:TIMEN,3) TO MS:TIME
RELEASE ALL LIKE V:*
```

```
DO CASE
CASE MS:TIME < '61'
```



```

                STORE MS:CO1 + 1 TO MS:CO1
CASE MS:TIME < '121'
                STORE MS:CO2 + 1 TO MS:CO2
CASE MS:TIME < '181'
                STORE MS:CO3 + 1 TO MS:CO3
CASE MS:TIME > '180'
                STORE MS:CO4 + 1 TO MS:CO4
                ENDCASE
            ENDFIF
        ENDFIF
    ENDFIF < CASE OPENED IN THE CURRENT YEAR >

```

***** IF CASE IS IN PREVIOUS YEAR

```

        IF $(MS:CDAT,1,2) = $(MS:PJUL,1,2)
            IF MS:ODAT > MS:PJUL1 .AND. MS:ODAT<MS:PJUL
                STORE MS:PTOT + 1 TO MS:PTOT
            ENDIF
        ENDIF

```

***** IF CASE IS STILL OPEN IT SHOULD BE IN THIS FILE

```

        IF MS:CDAT = ' '
            STORE VAL($ (MS:PJUL,3,3) ) TO V:CDAT
            STORE VAL($ (MS:ODAT,3,3) ) TO V:ODAT
            STORE V:CDAT - V:ODAT TO V:TIMEN
            STORE STR(V:TIMEN,3) TO MS:TIME
            RELEASE ALL LIKE V:*
        DO CASE
        CASE MS:TIME < '61'
            STORE MS:PO1 + 1 TO MS:PO1
        CASE MS:TIME < '121'
            STORE MS:PO2 + 1 TO MS:PO2
        CASE MS:TIME < '181'
            STORE MS:PO3 + 1 TO MS:PO3
        CASE MS:TIME > '180'
            STORE MS:PO4 + 1 TO MS:PO4
        ENDCASE
    ENDFIF
    ENDFIF
    ENDFIF < CASE OPENED IN THE PREVIOUS YEAR >

```

***** IF CASE IS IN CIDEST YEAR

```

        IF $(MS:ODAT,1,2) = $(MS:OJUL,1,2)
            IF MS:ODAT > MS:OJUL1 .AND. MS:ODAT<MS:OJUL
                STORE MS:OTOT + 1 TO MS:OTOT
            ENDIF
        ENDIF

```

***** IF CASE IS STILL OPEN IT SHOULD BE IN THIS FILE

```

        IF MS:CDAT = ' '
            STORE VAL($ (MS:OJUL,3,3) ) TO V:CDAT
            STORE VAL($ (MS:ODAT,3,3) ) TO V:ODAT
            STORE V:CDAT - V:ODAT TO V:TIMEN
            STORE STR(V:TIMEN,3) TO MS:TIME
            RELEASE ALL LIKE V:*
        DO CASE
        CASE MS:TIME < '61'
            STORE MS:OO1 + 1 TO MS:CO1
        CASE MS:TIME < '121'
            STORE MS:OO2 + 1 TO MS:CO2
        CASE MS:TIME < '181'
            STORE MS:OO3 + 1 TO MS:CO3
        CASE MS:TIME > '180'
            STORE MS:OO4 + 1 TO MS:CO4
        ENDCASE
    ENDFIF

```



```

                                ENDCASE
                                ENDIF
                                ENDIF
ENDIF < CASE OPENED IN THE OLDEST YEAR >
IF $(MS:CDAT,1,2) <> $(MS:CJUL,1,2) .AND.:
$(MS:CDAT,1,2) <> $(MS:PJUL,1,2) .AND.:
$(MS:CDAT,1,2) <> $(MS:OJUL,1,2)
STCRE MS:OPERR + 1 TO MS:OPERR
STORE CASE TO MS:CASE
SET ALTERNATE ON
? MS:ROW,10 SAY 'OCASE'
? MS:ROW,18 SAY MS:CASE
? MS:ROW,30 SAY $(MS:ODAT,1,2) + '/' +
+$(MS:ODAT,3,2) + '/' + $(MS:ODAT,5,2)
? MS:ROW,40 SAY $(MS:CDAT,1,2) + '/' +
$(MS:CDAT,3,2) + '/' + $(MS:CDAT,5,2)
STCRE MS:ROW + 1 TO MS:ROW
SET ALTERNATE OFF
ENDIF

```

```

STORE MS:OPTOT + 1 TO MS:OPTOT
SKIP

```

```

ENDDO < WHILE NOT EOF IN OPEN FILE >

```

```

***** START THE SEARCH AND COUNT IN THE CLOSED DATABASE

```

```

***** INITIALIZATION OF VARIABLES FOR CLOSED FILE

```

```

STCRE 0 TO MS:OCL1
STORE 0 TO MS:OCL2
STCRE 0 TO MS:OCL3
STORE 0 TO MS:OCL4
STCRE 0 TO MS:PCL1
STORE 0 TO MS:PCL2
STCRE 0 TO MS:PCL3
STORE 0 TO MS:PCL4
STCRE 0 TO MS:CCL1
STORE 0 TO MS:CCL2
STCRE 0 TO MS:CCL3
STORE 0 TO MS:CCL4

```

```

USE D:CLOSE1

```

```

DO WHILE .NOT. EOF
STORE DATES TO M:DATES

```

```

STORE $(M:DATES,11,5) TO MS:ODAT
STORE $(M:DATES,36,5) TO MS:CDAT

```

```

IF $(MS:CDAT,1,2) = $(MS:CJUL,1,2)

```

```

IF MS:CDAT > MS:CJUL1 .AND. MS:CDAT < MS:CJUL
STORE MS:CRCVD + 1 TO MS:CRCVD

```

```

IF MS:CDAT <> ' '
STORE VAL($ (MS:CDAT,3,3) ) TO V:CDAT
STORE VAL($ (MS:ODAT,3,3) ) TO V:ODAT
STORE V:CDAT - V:ODAT TO V:TIMEN
STORE STR(V:TIMEN,3) TO MS:TIME
RELEASE ALL LIKE V:*

```

```

DO CASE
CASE MS:TIME < '61'
STORE MS:CCL1 + 1 TO MS:CCL1
CASE MS:TIME < '121'

```



```

                STORE MS:CCL2 + 1 TO MS:CCL2
CASE MS:TIME < '181'
                STORE MS:CCL3 + 1 TO MS:CCL3
CASE MS:TIME > '180'
                STORE MS:CCL4 + 1 TO MS:CCL4
            ENDCASE
        ENDIF
    ENDIF
ENDIF < CASE OPENED IN THE CURRENT YEAR >

```

***** IF CASE IS IN PFEVIOUS YEAR

```

IF $(MS:CDAT,1,2) = $(MS:PJUL,1,2)
    IF MS:CDAT > MS:PJUL1 .AND. MS:CDAT<MS:PJUL
        STORE MS:PTOT + 1 TO MS:PTOT
    
```

```

    IF MS:CDAT <> ' '
        STORE VAL($ (MS:CDAT,3,3) ) TO V:CDAT
        STORE VAL($ (MS:ODAT,3,3) ) TO V:ODAT
        STORE V:CDAT - V:ODAT TO V:TIMEN
        STORE STR(V:TIMEN,3) TO MS:TIME
        RELEASE ALL LIKE V:*
    
```

```

    DO CASE
    CASE MS:TIME < '61'
        STORE MS:PCL1 + 1 TO MS:PCL1
    CASE MS:TIME < '121'
        STORE MS:PCL2 + 1 TO MS:PCL2
    CASE MS:TIME < '181'
        STORE MS:PCL3 + 1 TO MS:PCL3
    CASE MS:TIME > '180'
        STORE MS:PCL4 + 1 TO MS:PCL4
    ENDCASE

```

```

        ENDIF
    ENDIF
ENDIF < CASE CLOSED IN THE PREVIOUS YEAR >

```

***** IF CASE IS IN CIDEST YEAR

```

IF $(MS:CIAT,1,2) = $(MS:OJUL,1,2)
    IF MS:CDAT > MS:OJUL1 .AND. MS:CDAT<MS:CJUL
        STORE MS:OTOT + 1 TO MS:OTOT
    
```

***** IF CASE IS CLOSED < IT SHOULD BE TO BE IN THIS FILE >

```

    IF MS:CDAT <> ' '
        STORE VAL($ (MS:CDAT,3,3) ) TO V:CDAT
        STORE VAL($ (MS:ODAT,3,3) ) TO V:ODAT
        STORE V:CDAT - V:ODAT TO V:TIMEN
        STORE STR(V:TIMEN,3) TO MS:TIME
        RELEASE ALL LIKE V:*
    
```

```

    DO CASE
    CASE MS:TIME < '61'
        STORE MS:OCL1 + 1 TO MS:OCL1
    CASE MS:TIME < '121'
        STORE MS:OCL2 + 1 TO MS:OCL2
    CASE MS:TIME < '181'
        STORE MS:OCL3 + 1 TO MS:OCL3
    CASE MS:TIME > '180'
        STORE MS:OCL4 + 1 TO MS:OCL4
    ENDCASE

```

```

        ENDIF
    ENDIF

```


ENDIF < CASE CLOSED IN THE OLDEST YEAR >

```
IF $(MS:CDAT,1,2) <> $(MS:CJUL,1,2) .AND.;
$(MS:CDAT,1,2) <> $(MS:PJUL,1,2) .AND.;
$(MS:CDAT,1,2) <> $(MS:OJUL,1,2)
STCFE MS:CLERR + 1 TO MS:CLERR
STORE CASE TO MS:CASE
SET ALTERNATE ON
? MS:ROW,10 SAY 'CCASE'
? MS:ROW,18 SAY MS:CASE
? MS:ROW,30 SAY $(MS:ODAT,1,2) + '/' +
+$(MS:ODAT,3,2) + '/' + $(MS:ODAT,5,2)
? MS:ROW,40 SAY $(MS:CDAT,1,2) + '/' +
$(MS:CDAT,3,2) + '/' + $(MS:CDAT,5,2)
STCRE MS:ROW + 1 TO ROW
SET ALTERNATE OFF
```

```
ENDIF
STORE MS:CLTOT + 1 TO MS:CLTOT
```

SKIP

```
ENDDO < SEARCH FOR CASE IN THE CLOSED FILE >
RELEASE MS:CJUL1,MS:PJUL1,MS:OJUL1,M:DATES
```

***** DETERMINE WHICH MONTH IT IS FOR THE REPORT

```
DO CASE
CASE $(MS:EDATE,1,2) = '01'
STORE 'JANUARY' TO MS:MONTH
CASE $(MS:EDATE,1,2) = '02'
STORE 'FEBRUARY' TO MS:MONTH
CASE $(MS:EDATE,1,2) = '03'
STORE 'MARCH' TO MS:MONTH
CASE $(MS:EDATE,1,2) = '04'
STORE 'APRIL' TO MS:MONTH
CASE $(MS:EDATE,1,2) = '05'
STORE 'MAY' TO MS:MONTH
CASE $(MS:EDATE,1,2) = '06'
STORE 'JUNE' TO MS:MONTH
CASE $(MS:EDATE,1,2) = '07'
STORE 'JULY' TO MS:MONTH
CASE $(MS:EDATE,1,2) = '08'
STORE 'AUGUST' TO MS:MONTH
CASE $(MS:EDATE,1,2) = '09'
STORE 'SEPTEMBER' TO MS:MONTH
CASE $(MS:EDATE,1,2) = '10'
STORE 'OCTOBER' TO MS:MONTH
CASE $(MS:EDATE,1,2) = '11'
STORE 'NOVEMBER' TO MS:MONTH
CASE $(MS:EDATE,1,2) = '12'
STORE 'DECEMBER' TO MS:MONTH
ENDCASE
```

```
STCRE '19' + $(MS:OJUL,1,2) TO MS:OYR
STORE '19' + $(MS:PJUL,1,2) TO MS:PYR
STCRE '19' + $(MS:CJUL,1,2) TO MS:CYR
```

```
STCRE MS:OCI1+MS:OCL2+MS:OCL3+MS:OCL4 TO MS:OCI5
STCRE MS:OO1 + MS:OO2 + MS:OO3 + MS:OO4 TO MS:OO5
STCRE MS:OO5+MS:OCL5 TO MS:OCASE
```

```
STORE MS:PCL1+MS:PCL2+MS:PCL3+MS:PCL4 TO MS:PCL5
STCRE MS:PO1 + MS:PO2 + MS:PO3 + MS:PO4 TO MS:PO5
STCRE MS:PO5+MS:PCL5 TO MS:PCASE
```


STORE MS:CCL1+MS:CCL2+MS:CCL3+MS:CCL4 TO MS:CCI5
 STCRE MS:CO1 + MS:CO2 + MS:CO3 + MS:CO4 TO MS:CO5
 STCRE MS:CO5+MS:CCI5 TO MS:CCASE

STCRE '0 TO 60 DAYS' TO MS:LBL1
 STORE '61 TO 120 DAYS' TO MS:LBL2
 STCRE '121 TO 180 DAYS' TO MS:LBL3
 STCRE '181 DAYS OR OVER' TO MS:LBL4
 STCRE ' TOTALS:' TO MS:LBL5

SET FORMAT TO PRINT

@ 2,25 SAY ' CODE 9142 TECHNICAL BRANCH'
 @ 3,25 SAY ' QUALITY DEFICIENT MATERIAL'
 @ 4,25 SAY ' MONTHLY STATUS REPORT '
 @ 5,25 SAY ' THRU'
 @ 5,42 SAY '\$(MS:EDATE,1,2) + '/' + \$(MS:EDATE,3,2) ;
 + '/' + \$(MS:EDATE,5,2)
 @ 7,28 SAY ' JULIAN DATE '
 @ 7,48 SAY MS:CJUL
 @ 9,27 SAY ' MONTH OF'
 @ 9,44 SAY MS:MONTH
 @ 10,42 SAY MS:OYR
 @ 11,25 SAY 'TOTAL CASES'
 @ 11,40 SAY MS:OCASE
 @ 13,10 SAY ' CLOSED';
 + ' OPEN'
 @ 15,15 SAY MS:LBL1
 @ 15,33 SAY MS:OCL1
 @ 15,48 SAY MS:OO1
 @ 17,15 SAY MS:LBL2
 @ 17,33 SAY MS:OCL2
 @ 17,48 SAY MS:OO2
 @ 19,15 SAY MS:LBL3
 @ 19,33 SAY MS:OCL3
 @ 19,48 SAY MS:OO3
 @ 21,15 SAY MS:LBL4
 @ 21,33 SAY MS:OCL4
 @ 21,48 SAY MS:OO4
 @ 22,15 SAY MS:LBL5
 @ 22,33 SAY MS:OCL5
 @ 22,48 SAY MS:OO5

***** DATA FOR THE SECOND YEAR OUTPUT

@ 25,42 SAY MS:PYP
 @ 27,25 SAY 'TOTAL CASES'
 @ 27,40 SAY MS:PCASE
 @ 29,10 SAY ' CLOSED';
 + ' OPEN'
 @ 31,15 SAY MS:LBL1
 @ 31,33 SAY MS:PCL1
 @ 31,48 SAY MS:PO1
 @ 33,15 SAY MS:LBL2
 @ 33,33 SAY MS:PCL2
 @ 33,48 SAY MS:PO2
 @ 35,15 SAY MS:LBL3
 @ 35,33 SAY MS:PCL3
 @ 35,48 SAY MS:PO3
 @ 37,15 SAY MS:LBL4
 @ 37,33 SAY MS:PCL4
 @ 37,48 SAY MS:PO4
 @ 39,15 SAY MS:LBL5
 @ 39,33 SAY MS:PCL5
 @ 39,48 SAY MS:PO5

***** DATA FOR THE CURRENT YEAR OUTPUT


```

@ 42,42 SAY MS:CYR
@ 44,25 SAY 'TOTAL CASES'
@ 44,40 SAY MS:CCASE
@ 46,10 SAY '
                                OPEN'
                                CLOSED';
@ 48,15 SAY MS:LBL 1
@ 48,33 SAY MS:CCL 1
@ 48,48 SAY MS:CO1
@ 50,15 SAY MS:LBL 2
@ 50,33 SAY MS:CCL 2
@ 50,48 SAY MS:CO2
@ 52,15 SAY MS:LBL 3
@ 52,33 SAY MS:CCL 3
@ 52,48 SAY MS:CO3
@ 54,15 SAY MS:LBL 4
@ 54,33 SAY MS:CCL 4
@ 54,48 SAY MS:CO4
@ 56,15 SAY MS:LBL 5
@ 56,33 SAY MS:CCL 5
@ 56,48 SAY MS:CO5

```

EJECT
 ***** SECOND PAGE OF REPORT

```

@ 2,25 SAY ' CODE 9142 TECHNICAL BRANCH'
@ 4,25 SAY ' QUALITY DEFICIENT MATERIAL'
@ 6,25 SAY ' COMMAND KEY INDICATORS '
@ 8,25 SAY ' AS OF '
@ 8,42 SAY '$ (MS:EDATE,1,2) + '/' + $ (MS:EDATE,3,2) + ;
          '/' + $ (MS:EDATE,5,2)
@ 10,25 SAY 'CASES RECEIVED IN'
@ 10,44 SAY MS:MONTH
@ 10,62 SAY MS:CYR + ':'
@ 10,72 SAY MS:CRDVD
@ 12,10 SAY '
                                OPEN'
                                CLOSED';
@ 14,15 SAY MS:LBL 1
@ 14,33 SAY MS:CCL 1
@ 14,48 SAY MS:CO1
@ 16,15 SAY MS:LBL 2
@ 16,33 SAY MS:CCL 2
@ 16,48 SAY MS:CO2
@ 18,15 SAY MS:LBL 3
@ 18,33 SAY MS:CCL 3
@ 18,48 SAY MS:CO3
@ 20,15 SAY MS:LBL 4
@ 20,33 SAY MS:CCL 4
@ 20,48 SAY MS:CO4
@ 22,15 SAY MS:LBL 5
@ 22,33 SAY MS:CCL 5
@ 22,48 SAY MS:CO5

```

EJECT
 ***** THIRD PAGE OF REPORT

```

@ 2,25 SAY ' CODE 9142 TECHNICAL BRANCH'
@ 3,25 SAY ' QUALITY DEFICIENT MATERIAL'
@ 4,25 SAY ' MONTHLY STATUS REPORT '
@ 5,25 SAY ' THRU '
@ 5,42 SAY '$ (MS:EDATE,1,2) + '/' + $ (MS:EDATE,3,2) + ;
          '/' + $ (MS:EDATE,5,2)
@ 7,28 SAY ' JULIAN DATE '
@ 7,48 SAY MS:CJUL
@ 9,27 SAY ' MONTH OF '
@ 9,44 SAY MS:MONTH
@ 12,20 SAY 'TOTAL RECORDS ON OPEN FILE:'
@ 12,64 SAY MS:OPTOT - 1
@ 14,20 SAY 'TOTAL RECORDS ON CLOSED FILES:'
@ 14,65 SAY MS:CLTOT

```


@ 16,20 SAY 'RECORDS WITH INVALID DATES, OPEN ';
+' FILE:'
@ 16,64 SAY MS:OPERR - 1
@ 18,20 SAY 'RECORDS WITH INVALID DATES, CLOSED';
+' FILE:'
@ 18,65 SAY MS:CLERR
@ 22,35 SAY 'END OF REPORT'
SET ALTERNATE OFF
SET ALTERNATE TO

EJECT
SET FORMAT TO SCREEN

RELEASE ALL LIKE MS:*

ERASE
RETURN

***** END OF PROGRAM

XVIII. SORTED LISTING REPORT MENU

```
*****  
**  
** DATE: 22 JANUARY 1984 **  
** VERSION: 1.0 **  
** MODULE NAME: SUPRPT2 **  
** MCDUIE PURPOSE: PROVIDE MENU OF SORTED LISTING **  
** REPORTS FOR THE SUPERVISOR **  
**  
** MODULE INTERFACE DEFINITION **  
** INPUTS: C:WHO, C:JULIAN **  
** OUTPUTS: NONE **  
** MODULE PROCESSING NARRATIVE DESCRIPTION: **  
**  
** DISPLAYS MENU FOR SUPERVISOR TO CHOOSE DESIRED **  
** REPORT. CAUSES REINDEXING OF APPROPRIATE FILE **  
** TO PRODUCE CURRENT VALUES FOR REPORT GENERATION **  
** RESULTS ARE STORED ON D: DRIVE AS A 'TXT' FILE **  
** FOR LATER ACCESS. REPORT MAY BE PRINTED BY USING **  
** 'TYPE' FUNCTION OF OPERATING SYSTEM. **  
** PROCESSING SHOULD BE ACCOMPLISHED DURING 'OFF' **  
** TIME PERIOD. **  
**  
** SUPERORDINATE MODULES: SUPMENU1 **  
** SUBORDINATE MODULES: SUPRPT1, SUPRPT2, SUPRPT3, SUPRPT4 **  
** AUTHCR: J.G. BCYNTON **  
**  
*****
```

```
STORE T TO C:TRUE  
DO WHILE C:TRUE  
ERASE  
*  
STORE ' ' TC V:CHOICE  
TEXT
```

***** SORTED LISTING REPORTS AVAILABLE *****

- 1 - OPEN FILE BY CASE AND ANALYST
- 2 - OPEN FILE BY ITEM MGR AND ANALYST
- 3 - OPEN FILE BY COG, SMIC, OPEN DATE
- 4 - CLOSED FILE BY CREDIT CODE, CASE
- 5 - EXIT

```
ENDTEXT  
@ 19,40 GET V:CHOICE  
READ  
*  
IF V:CHOICE >='1' .AND. V:CHOICE < '5'  
ERASE  
@ 12,20 SAY ' THESE REPORTS WILL TAKE SOME TIME TO'  
@ 13,20 SAY ' GENERATE. IF YOU DECIDE TO CONTINUE'  
@ 14,20 SAY ' THE TERMINAL MAY NOT BE USED FOR ANY'  
@ 15,20 SAY ' OTHER PROCESSING UNTIL AFTER ' ;  
+'COMPLETION '  
@ 23,15 SAY 'PRESS 1 - TO ABORT, ANY OTHER KEY TO';  
+' CONTINUE'  
WAIT TO V:BAII
```



```

IF V:BAIL = '1'
  RELEASE ALL LIKE V:*
  RELEASE C:TRUE
  RETURN
ENDIF
DO CASE
  CASE V:CHOICE = '1'
  ***** FILE D:SUPRPT1.TXT IS CREATED TO PROVIDE THE REPORT
  ***** D:SUPRPT1.NDX IS INDEXED ON WHO+CASE
  RELEASE ALL LIKE V:*
  USE D:CFEN1 INDEX D:SUPRPT1
  REINDEX
  GOTO TCF
  SKIP
  SET TALK OFF
  STORE 0 TO P:COUNT
  STORE 0 TO P:TOTAL
  SET FORMAT TO SCREEN
  ERASE
  SET ALTERNATE TO D:SUPRPT1
  SET ALTERNATE ON
  ? 'LATE: ',DATE()
  ?
  ?
  + ' ***** '
  + ' QDR CFEN FILE BY ANALYST & CASE ***** '
  ?
  ? ' NSN / PART';
  + ' UNIT';
  + ' QTY EXT 9 OPEN SCREENING';
  ?
  + ' CASE';
  + ' COG SM FSC NATO FIIN ';
  + ' CAT NOMEN UIC';
  + ' UI PRICE PRICE DEF C DEFNT';
  + ' CONTRACT NUMBER DATE CODE/DATE';
  ?
  STORE 0 TO P:PAGE
  STORE 5 TO ROW
  DO WHILE .NOT. EOF
  STORE P:TOTAL+1 TO P:TOTAL
  ? ' CASE',COG,SM,' NSN,';
  ' CAT','$ (NOMEN,1,9),UIC,UI';
  ' UPRC,QT,DEF,EPRC,09Q';
  ' ORG,DEF,DOC,NUM';
  $(DATE$,11,5),SCR,'/',$ (DATE$,21,5)
  STORE ROW+1 TO ROW
  SKIP
  STORE P:COUNT+1 TO P:COUNT
  IF ROW > 60
  ERASE
  ? CHR(12)
  STORE 0 TO ROW
  STORE P:PAGE+1 TO P:PAGE
  ?
  ?
  ? ' PAGE ',P:PAGE
  ?
  + ' NSN / PART';
  + ' UNIT';
  + ' QTY EXT 9 OPEN';
  + ' D SCREENING';
  ? ' CASE';

```



```

+ '      CCNTRACT NUMBER      DATE      CODE/DATE '
?
STORE 0 TO P:PAGE
STORE 5 TO ROW
DO WHILE .NOT. EOF
  STORE P:TOTAL+1 TO P:TOTAL
  ? '      CASE ' COG ' SM ' NSN '
  ? '      CAT ' $(NOMEN,1,9) ' UIC '
  ? '      UI ' UPRC ' QTYDEF '
  EPRC ' 09Q ' ORG ' DEF ' DOC '
  ? '      NUM ' $(DATES,11,5) ' SCR ' / '
  ? '      $(DATES,21,5)
  STORE ROW+1 TO ROW
  SKIP
  STORE P:COUNT+1 TO P:COUNT
  IF ROW > 60
    ERASE
    ? CHR(12)
    STORE 0 TO ROW
    STORE P:PAGE+1 TO P:PAGE
    ?
    ?
    ? '      PAGE ' , P:PAGE
    ?
    + '      NSN / PART ' ;
    + '
    + '      UNIT Q ' ;
    + '      TY
    + '      EXT 9
    + '
    + '      OPEN SCREENING '
    ? '      CASE ' ;
    + '
    + '      FSC NATO FIIN COG SM ' ;
    + '      EN UIC CAT NOM ' ;
    + '      UI PRICE PRICE DEFNT ' ;
    + '      PRICE Q ORG DF ' ;
    + '      C CONTRACT NUMBER DATE ' ;
    + '      CODE/DATE '
    ?
    STORE ROW+4 TO ROW
  ENDIF <PAGE IS FULL>
ENDDO
?
?
?
?
?
?
? '      TOTAL CASES: ' , P:TOTAL
?
? '      ***** ' ;
+ '      END OF OPEN FILE REPORT BY ITEM MANAGER & ' ;
+ '      ANALYST ***** ' ;
? CHR(12)
SET ALTERNATE TO
? CHR(7)
? CHR(7)
ERASE
@ 12,20 SAY ' YOU MAY RECEIVE YOUR COG,OPEN ' ;
+ '      FILE REPORT ON '
@ 13,20 SAY '
+ '      D: SUPRPT2.TXT '
@ 20,20 SAY ' PRESS ANY KEY TO CONTINUE '
WAIT
CASE V:CHOICE = '3'

```

```

***** FILE D: SUPRPT3.TXT IS CREATED TO PROVIDE THE REPORT
***** D: SUPRPT3.NDX IS INDEXED ON COG+SM+$(DATES,11,5)+CASE

```



```

RELEASE ALL LIKE V:*
USE D:CFEN1 INDEX D:SUPRPT3
REINDEX
GOTO TCF
SKIP
SET TALK OFF
STORE 0 TO P:COUNT
STORE 0 TO P:TOTAL
SET FORMAT TO SCREEN
ERASE
SET ALTERNATE TO D:SUPRPT3
SET ALIERNATE ON
? 'DATE: ',DATE()
?
?
+ ' ***** QDR OPEN FILE BY ' ;
+ ' COG,SMIC,OPEN DATE & CASE ***** ' ;
?
? ' NSN / PART' ;
+ ' UNIT' ;
+ ' QTY EXT 9
+ ' D OPEN SCREENING'
? ' CASE' ;
+ ' CAT NOMEN UIC UI PRIC
+ ' E DEFNT PRICE Q ORG
+ ' DF C CONTRACT NUMBER DATE CODE/' ;
+ ' DATE '
?
STORE 0 TO P:PAGE
STORE 5 TO ROW
DO WHILE .NOT. EOF
STORE P:TOTAL+1 TO P:TOTAL
? ' CASE, COG, SM, NSN ;
' CAT, $(NOMEN,1,9), UIC ;
' UI, UPRC, QTYDEF, EPRC ;
' 09Q, ORG, DEF, DOC ;
' NUM, $(DATES,11,5), SCR, / ;
$(DATES,21,5)
STORE ROW+1 TO ROW
SKIP
STORE P:COUNT+1 TO P:COUNT
IF ROW > 60
ERASE
? CHR(12)
STORE 0 TO ROW
STORE P:PAGE+1 TO P:PAGE
?
? ' PAGE ',P:PAGE
?
? ' NSN / PART' ;
+ ' UNIT
+ ' QTY EXT 9
+ ' D
+ ' OPEN SCREENING'
? ' CASE' ;
+ ' C NATO FIIN CAT NOMEN
+ ' UIC UI PRICE
+ ' DEFNT PRICE C OR
+ ' G DF C CONTRAC
+ ' T NUMBER DATE CODE/DATE '
?
STORE ROW+4 TO ROW
ENDIF <PAGE IS FULL>

```



```

STORE P:COUNT+1 TO P:COUNT
IF ROW > 60
  ERASE
  ? CHR(12)
  STORE 0 TO ROW
  STORE P:PAGE+1 TO P:PAGE
  ?
  ?
  ? ' PAGE ',P:PAGE
  ?
  ?
  ? ' NSN / PART';
  ?
  ?
  ? QTY UNIT EXT 9
  ?
  ? OPEN SCREENING'
  ? ' CASE COG SM FS';
  ? 'C NATO FIIN CAT NOMEN
  ? ' UIC UI PRICE
  ? 'DEFNT PRICE C OR'
  ? 'G DF C CONTRAC'
  ? 'T NUMBER DATE CODE/DATE '
  ?
  STORE ROW+4 TO ROW
ENDIF <PAGE IS FULL>

```

ENDDO

```

?
?
?
?
? ' TOTAL CASES:',P:TOTAL
?
? *****
+ ' END CF CLOSED FILE REPORT BY CREDIT CODE';
+ ' & CASE *****'
? CHR(12)
SET ALTERNATE TO
? CHR(7)
? CHR(7)
ERASE
@ 12,20 SAY ' YOU MAY RECEIVE YOUR REPORT BY';
+ ' CREDIT CODE & CASE ON'
@ 13,20 SAY ' D:SUPRPT4.TXT '
@ 20,20 SAY ' PRESS ANY KEY TO CONTINUE'
WAIT
CASE V:CHCICE = '5'
RELEASE ALL LIKE V:*
RELEASE C:TRUE
RETURN

```

ERASE
ENDCASE

```

ELSE
IF V:CHOICE='5'
RELEASE ALL LIKE V:*
RELEASE C:TRUE
RETURN
ELSE

```

PLEASE ANSWER WITH A 1 - 5 ONLY'
PRESS ANY KEY TO CONTINUE'

```

?
?
?
WAIT
ENDIF
ENDIF <V:CHCICE>
ENDDO <C:TRUE>

```


RELEASE ALL LIKE V:*
RELEASE C:TRUE
RETURN

***** END OF PROGRAM

XIX. CASE REASSIGNMENT MODULE

```

*****
**
** DATE: 22 JANUARY 1984
** VERSICN: 1.0
** MODULE NAME: C-REASGN
** MODULE PURPOSE: REASSIGN CASE FROM ONE ANALYST TO
** ANOTHER
** MODULE INTERFACE DEFINITION
** INPUTS: C:WHO, C:JULIAN
** OUTPUTS:
** MODULE PROCESSING NARRATIVE DESCRIPTION:
**
** RECEIVES THE CASE NUMBER AND TWO ANALYSTS TO BE
** INVOLVED IN THE TRANSFER. BEFORE TRANSFERRING CASE
** THE DATABASE IS CHECKED TO INSURE UPDATE OF
** ANALYST STATISTICS. RETURNS FOR ANOTHER CASE TO
** BE ASSIGNED OR FOR TERMINATION OF PROGRAM.
**
** SUPERORDINATE MODULES: UTILMENU
** SUBORDINATE MODULES: XDBHNDLR
** AUTHCR: J.G. BOYNTON
**
*****

```

```

STORE T TO R:CONTINUE
DO WHILE R:CONTINUE
  ERASE
  @ 6,24 SAY '***** CASE RE-ASSIGNMENT PROCESSING *****'
  @ 9,28 SAY '1 - RE-ASSIGN CASE TO ANOTHER ANALYST'
  @ 10,28 SAY '2 - RETURN TO UTILITY MENU'
  STORE ' ' TO R:REPLY
  @ 15,40 GET R:REPLY
  READ
  DC WHILE R:REPLY < '1' . OR. R:REPLY > '2'
    @ 23,32 SAY 'ENTER 1 - 2 ONLY' + CHR(7)
    @ 15,40 GET R:REPLY
    READ
  ENDDO
  DO CASE
    CASE R:REPLY = '2'
      RELEASE ALL LIKE R:*
      RETURN
    CASE R:REPLY = '1'
      STCRE ' ' TO R:CASE
      ERASE
      @ 6,24 SAY 'ENTER DATA FOR CASE BEING RE-ASSIGNED'
      @ 9,32 SAY 'CASE NUMBER ' GET R:CASE ;
      PICTURE '999999X'
      READ
      STCRE ! (R:CASE) TO R:CASE
      STCRE R:CASE TO M:KEY
      STORE '1E' TC M:TYPE
      DO XDBHNDLR.PRG
      IF M:TYPE = '1'
        @ 20,20 SAY 'CASE IS CURRENTLY LOCKED' + CHR(7)
        @ 22,20 SAY 'PRESS ANY KEY TO CONTINUE'
        WAIT
      ENDIF
      IF M:TYPE <> '1'
        IF M:TYPE = '9'
          @ 22,14 SAY 'CASE DOES NOT EXIST IN OPEN';

```


+ 'FILE - STRIKE ANY KEY TO CONTINUE';
+ CHR(7)

```
WAIT
ELSE
  STORE ' ' TO R:NEW
  STORE ' ' TO R:REPLY
  DO WHILE R:REPLY
    @ 11,10 SAY 'CASE NUMBER ' + M:CASE
    @ 11,32 SAY 'NSN ' + $(M:NSN,1,4) + '-';
      +$(M:NSN,5,2) + '-'+$(M:NSN,7,3);
      +'-'+$(M:NSN,10,4)
    @ 11,55 SAY 'COG ' + M:COG
    @ 11,62 SAY 'CAT ' + M:CAT
    @ 14,10 SAY 'CURRENTLY ASSIGNED TO ' ;
      + M:WHO
    @ 15,10 SAY 'RE-ASSIGN TO ' ;
      GET R:NEW

  READ
  STORE ' ' TO R:REPLY
  DC WHILE R:REPLY <'1' .OR. R:REPLY >'3'
    @ 21,20 SAY '1 - RE-ASSIGN ' ;
      + ' 2 - CHANGE 3 - EXIT'
    @ 23,20 SAY ' ' ;
      GET R:REPLY

  READ
  ENDDO
  IF R:REPLY = '3'
    STORE '1G' TO M:TYPE
    DO XDBHNDLR.PRG
    RELEASE ALL EXCEPT C:*
    RETURN
  ENDIF
  IF $(M:DATES,46,1) = '* '
    DO STATISTICS UPDATE PROGRAM
  ENDIF
  IF R:REPLY = '1'
    ERASE
    @ 10,20 SAY '***** PLEASE ' ;
      + 'STANDBY *****'
    @ 12,20 SAY 'CASE NUMBER ' + M:CASE
    @ 13,20 SAY 'IS BEING RE-ASSIGNED '
    @ 15,20 SAY ' FROM ' + M:WHO
    @ 16,20 SAY ' TO ' + R:NEW
    @ 22,18 SAY ' ***** DO NOT ' ;
      + 'INTERRUPT *****'
    USE D:OPEN1 INDEX D:OCASE1,D:ONSN
    STORE R:NEW TO M:WHO
    STORE '1C' TO M:TYPE
    DO XDBHNDLR.PRG
    STORE M:CASE TO R:CASE
    RELEASE ALL LIKE M:*
    STORE R:CASE TO M:KEY
    STORE '2E' TO M:TYPE
    DO XDBHNDLR.PRG
    STORE '2C' TO M:TYPE
    STORE R:NEW TO M:WHO
    DO XDBHNDLR.PRG
    RELEASE ALL LIKE M:*
    ERASE
    STORE F TO R:REPLY
  ENDIF
  ENDDC <R:REPLY>
ENDIF
ENDCASE
ENDDC <CCONTINUE>
***** END OF PROGRAM
```


XX. ITEM MANAGER FILE UPDATE

```

*****
**
** DATE: 12 JAN 84
** VERSICN: 1.0
** MODULE NAME: ADDRUPDT
** MCDULE PURPOSE: ADD, UPDATE, OR DELETE ITEM MANAGER
** ADDRESS RECORDS
**
** MODULE INTERFACE DEFINITION
** INPUTS: NONE
** OUTPUTS: NONE
**
** MODULE PROCESSING NARRATIVE DESCRIPTION:
**
**     ALLOWS THE ADDITION, DELETION, OR UPDATING
**     OF ITEM MANAGER ADDRESS RECORDS. ACCEPTS THE
**     ITEM MANAGER CODE, VALIDATES THE EXISTENCE OF
**     AN IM WHEN CREATING NEW RECORDS, ALLOWS THE
**     RECALL OF PREVIOUSLY DELETED RECORDS DURING
**     UPDATE, AND VERIFIES THAT NO ACTIVE COGS
**     ARE ASSIGNED TO A RECORD BEING DELETED.
**
** SUPERORDINATE MODULES: UTILMENU
** SUBORDINATE MODULES: NONE
** AUTHCR: R. G. NICHOLS
**
*****

```

```

STORE T TO A:CONTINUE
DO WHILE A:CONTINUE

```

```

***** DISPLAY SELECTION OPTICNS AND ACCEPT CHOICE
ERASE
@ 6,18 SAY '***** ITEM MANAGER ADDRESS PROCESSING *****'
@ 9,28 SAY '1 - ADD ADDRESS'
@ 10,28 SAY '2 - UPDATE ADDRESS'
@ 11,28 SAY '3 - DELETE ADDRESS'
@ 12,28 SAY '4 - RETURN TO UTILITY MENU'
STORE ' ' TO A:REPLY
@ 15,40 GET A:REPLY PICTURE '9'
READ

```

```

***** VALIDATE RESPCNSE
DO WHILE A:REPLY < '1' .OR. A:REPLY > '4'
@ 23,32 SAY 'ENTER 1 - 4 ONLY' + CHR(7)
@ 15,40 GET A:REPLY PICTURE '9'
READ
ENDDO
DC CASE

```

```

***** IF CHOICE IS QUIT, RELEASE ALL LOCAL MEMORY VARIABLES
***** AND RETURN TO CALLING PROGRAM

```

```

CASE A:REPLY = '4'
RELEASE ALL LIKE A:*
RETURN

```

```

***** IF CHOICE IS ADD, PROMPT FOR ITEM MANAGER BEING ADDED
CASE A:REPLY = '1'

```



```

STORE ' ' TO A:IM
ERASE
@ 6,20 SAY 'ENTER DATA FOR ITEM MANGER BEING ADDED'
@ 9,32 SAY 'ITEM MANAGER ' GET A:IM
READ
STORE !(A:IM) TO A:IM
SELECT PRIMARY
USE D:ADDRESS INDEX D:IM

```

***** CHECK FOR EXISTING IM RECORD

```

FIND &A:IM
IF # <> 0
  @ 22,14 SAY 'RECORD CURRENTLY EXISTS -';
  + 'STRIKE ANY KEY TO CONTINUE' + CHR(7)
  WAIT
ELSE
  STORE ' ' TO A:TITLE
  STORE ' ' TO A:COMMAND
  STORE ' ' TO A:COMMAND2
  STORE ' ' TO A:ATTN
  STORE ' ' TO A:STREET
  STORE ' ' TO A:CITY
  STORE ' ' TO A:STATE
  STORE ' ' TO A:ZIP

```

***** PROMPT FOR AND ACCEPT NEW IM DATA

```

@ 14,16 SAY 'TITLE ' GET A:TITLE
@ 15,16 SAY 'COMMAND ' GET A:COMMAND
@ 16,16 SAY 'COMMAND 2 ' GET A:COMMAND2
@ 17,16 SAY 'ATTN ' GET A:ATTN
@ 18,16 SAY 'STREET ' GET A:STREET
@ 19,16 SAY 'CITY ' GET A:CITY
@ 19,44 SAY 'STATE ' GET A:STATE
@ 19,56 SAY 'ZIP CODE ' GET A:ZIP ;
          PICTURE '99999'
READ

```

***** VERIFY POST OR EXIT

```

@ 21,20 SAY '1 - POST NEW RECORD ' ;
          + '2 - EXIT'
STORE ' ' TO A:REPLY3
@ 23,40 GET A:REPLY3 PICTURE '9'
READ
DO WHILE A:REPLY3 < '1' .OR. A:REPLY3 > '2'
  @ 23,40 GET A:REPLY3
  READ
ENDDC

```

***** CREATE NEW IM RECORD

```

IF A:REPLY3 = '1'
  APPEND BLANK
  REPLACE IM WITH !(A:IM), TITLE WITH ;
  !(A:TITLE), COMMAND WITH !(A:COMMAND), COMMAND2 WITH ;
  !(A:COMMAND2), ATTN WITH !(A:ATTN), STREET WITH !(A:STREET), ;
  CITY WITH !(A:CITY), STATE WITH !(A:STATE), ZIP WITH A:ZIP
ENDIF
ENDIF

```

***** IF CHOICE IS UPDATE

```

CASE A:REPLY = '2'
  STORE ' ' TO A:IM

```



```

***** REQUEST FOR AND ACCEPT IM TO BE UPDATED
      ERASE
      @ 6,19 SAY 'ENTER DATA FOR ITEM MANGER BEING ';
        + 'UPDATED'
      @ 9,32 SAY 'ITEM MANAGER ' GET A:IM
      READ
      STORE !(A:IM) TO A:IM
      SELECT PRIMARY

```

```

***** RETREIVE IM RECORD BEING UPDATED

      USE D:ADDRESS INDEX D:IM
      FIND &A:IM
      IF # <> 0
        @ 22,17 SAY 'RECORD NOT FOUND ';
          + '- STRIKE ANY KEY TO CONTINUE'+CHR(7)
          WAIT
      ELSE
        STORE T TO A:PROCESS

```

```

***** IF RECORD DELETED, PRCPMT FOR REACTIVATION OPTION

      IF *
        STORE ' ' TO A:REPLY2
        @ 18,22 SAY 'RECORD HAS BEEN MARKED FOR ';
          + 'DELETION'
        @ 19,17 SAY 'DO YOU WANT THIS RECORD ';
          + 'REACTIIVATED <Y OR N>'
        @ 21,40 GET A:REPLY2 PICTURE 'A'
        READ
        DO WHILE A:REPLY2<>'Y' .AND. A:REPLY2<>'N'
          @ 23,31 SAY 'ENTER Y OR N ONLY'+CHR(7)
          @ 21,40 GET A:REPLY2 PICTURE 'A'
          READ
        ENDDO

```

```

***** IF REACTIVATION REQUESTED

      IF !(A:REPLY2) = 'Y'
        @ 18,22 SAY ' ';
          + ' '
        @ 19,17 SAY ' ';
          + ' '
        @ 21,40 SAY ' '
        RECALL
      ELSE
        STORE F TO A:PROCESS
      ENDIF
    ENDIF

```

```

***** PROMPT FOR AND ACCEPT UPDATE INFORMATION

      DO WHILE A:PROCESS
        STORE TITLE TO A:TITLE
        STORE COMMAND TO A:COMMAND
        STORE CCMMAND2 TO A:COMMAND2
        STORE ATTN TO A:ATTN
        STORE STREET TO A:STREET
        STORE CITY TO A:CITY
        STORE STATE TO A:STATE
        STORE ZIP TO A:ZIP
        @ 14,16 SAY 'TITLE ' GET TITLE
        @ 15,16 SAY 'COMMAND ' GET COMMAND
        @ 16,16 SAY 'COMMAND2 ' GET COMMAND2
        @ 17,16 SAY 'ATTN ' GET ATTN
        @ 18,16 SAY 'STREET ' GET STREET
        @ 19,16 SAY 'CITY ' GET CITY
        @ 19,44 SAY 'SIATE ' GET STATE
        @ 19,56 SAY 'ZIP CODE ' GET ZIP ;

```



```

                PICTURE '99999'
READ
@ 21,20 SAY '1 - POST UPDATED RECORD ';
+ '2 - EXIT'
STORE ' ' TO A:REPLY3
@ 23,40 GET A:REPLY3 PICTURE '9'

***** PROVIDE UPDATE/EXIT OPTION

READ
DO WHILE A:REPLY3 < '1' .OR. A:REPLY3 > '2'
    @ 23,40 GET A:REPLY3
    READ
ENDDO

***** IF UPDATE

    IF A:REPLY3 <> '1'
        REPIACE IM WITH !(A:IM), TITLE WITH ;
!(A:TITLE), COMMAND WITH !(A:COMMAND), COMMAND2 WITH ;
!(A:CCMMAND2), ATTN WITH !(A:ATTN), STREET WITH !(A:STREET), ;
CITY WITH !(A:CITY), ZIP WITH A:ZIP
        ENDIF
        STORE F TO A:PROCESS
    ENDDO
ENDIF

***** IF CHOICE IS DELETE

CASE A:REPLY = '3'
    STORE ' ' TO A:IM

***** PROMPT FOR AND ACCEPT IM BEING DELETED

    ERASE
    @ 6,19 SAY 'ENTER DATA FOR ITEM MANGER BEING ';
    + 'DELETED'
    @ 9,32 SAY 'ITEM MANAGER ' GET A:IM
    READ
    STCRE !(A:IM) TO A:IM
    SELECT PRIMARY
    USE D:ADDRESS INDEX D:IM

***** RETRIEVE RECORD BEING DELETED

    FIND &A:IM
    IF # <> 0
        @ 22,17 SAY 'RECORD NOT FOUND ';
        + '- STRIKE ANY KEY TO CONTINUE'+CHR(7)
        WAIT
    ELSE

***** IF ACTIVE COGS ARE ASSIGNED, DO NOT ALLOW DELETE

        IF COUNT > 0
            @ 11,16 SAY 'TITLE ' + TITLE
            @ 12,16 SAY 'COMMAND ' + COMMAND
            @ 13,16 SAY 'CCMMAND 2 ' + COMMAND2
            @ 14,16 SAY 'ATTN ' + ATTN
            @ 15,16 SAY 'STREET ' + STREET
            @ 16,16 SAY 'CITY '+CITY-' STATE '+STATE+;
            + 'ZIP CODE ' + ZIP
            @ 18,20 SAY 'ACTIVE COGS EXIST FOR THIS ';
            + 'ITEM MANAGER'
            @ 19,15 SAY 'ALL ACTIVE COGS MUST BE ';
            + 'REASSIGNED TO ANOTHER I.M.'
            @ 20,21 SAY 'BEFORE DELETE ACTION CAN BE';
            + 'COMPLETED'
            @ 22,27 SAY 'STRIKE ANY KEY TO CONTINUE';

```


+ CHR(7)

WAIT
ELSE

***** IF PREVIOUSLY DELETED, NOTIFY OPERATOR

```
IF *
@ 23,13 SAY 'RECORD PREVIOUSLY DELETED ';
+ '- STRIKE ANY KEY TO CONTINUE'+CHR(7)
WAIT
ELSE
@ 11,16 SAY 'TITLE ' + TITLE
@ 12,16 SAY 'COMMAND ' + COMMAND
@ 13,16 SAY 'COMMAND 2 ' + COMMAND2
@ 14,16 SAY 'ATTN ' + ATTN
@ 15,16 SAY 'STREET ' + STREET
@ 16,16 SAY 'CITY '+CITY-' STATE '+STATE;
- ' '+ZIP CODE ' + ZIP
@ 19,20 SAY '1 - DELETE THIS ITEM MANAGER
+ ' 2 - EXIT'
STORE ' ' TO A:REPLY2
@ 21,40 GET A:REPLY2 PICTURE '9'
```

***** ACCEPT DELETE/EXIT OPTION

```
READ
DO WHILE A:REPLY2 < '1' .OR. A:REPLY2 > '2'
@ 23,40 GET A:REPLY2
READ
ENDDO
```

***** IF DELETE REQUESTED

```
IF A:REPLY2 = '1'
DELETE
ENDIF
ENDIF
ENDIF
ENDIF
ENDCASE
USE
ENDDC
```

***** ENI OF PROGRAM

XXI. COG FILE UPDATE MODULE

```
*****
**
** DATE: 18 JAN 1984
** VERSICN: 1.0
** MODULE NAME: COGUEDT
** MCDUIE PURPOSE: AILLOWS ADDITION, DELETION, OR
**                  UPDATING OF COG FILE
**
** MODULE INTERFACE DEFINITION
**   INPUTS: NCNE
**   OUTPUTS: NONE
**
** MODULE PROCESSING NARRATIVE DESCRIPTION:
**
**   ACCEPTS NEW COG DATA, VERIFIES THAT THE NEW
**   CCG IS NOT A DUPLICATE, AND VERIFIES THE
**   EXISTENCE CF AN ITEM MANAGER RECORD. ACCEPTS
**   UPDATE INFCMATION ON COG-ITEM MANAGER ASSIGN-
**   MENTS, VALIDATES THE EXISTENCE OF AN ITEM
**   MANAGER RECCRD, AND UPDATES THE COG FILE.
**   PREVIOUSLY DELETED COGS MAY BE REACTIVATED
**   WITH THE UPDATE OPTION. ACCEPTS THE COG TO
**   BE DELETED, VERIFIES THE RECORDS EXISTENCE,
**   AND VERIFIÉS THAT NO ACTIVE CASES EXIST FOR
**   THIS COG.
**
** SUPERORDINATE MODULES: UTILMENU
** SUBORDINATE MODUIES: NONE
** AUTHCR: R. G. NICHOLS
**
*****
```

```
* COGUEDT.PRG
* LAST UPDATE 18 JAN 84
```

```
*
STORE T TO G:CONTINUE
DO WHILE G:CONTINUE
```

```
***** DISPLAY OPTIONS AND ACCEPT CHOICE
```

```
ERASE
@ 6,24 SAY '***** COG FILE PROCESSING *****'
@ 9,28 SAY '1 - ADD COG'
@ 10,28 SAY '2 - UPDATE COG'
@ 11,28 SAY '3 - DELETE COG'
@ 12,28 SAY '4 - RETURN TO UTILITY MENU'
STORE ' ' TO G:REPLY
@ 15,40 GET G:REPLY
READ
DO WHILE G:REPLY < '1' .OR. G:REPLY > '4'
  @ 23,32 SAY 'ENTER 1 - 4 ONLY' + CHR(7)
  @ 15,40 GET G:REPLY
  READ
ENDDO
DO CASE
```

```
***** IF CHOICE IS TC QUIT, RELEASE LOCAL MEMORY VARIABLES
***** AND RETURN TO CALLING PROGRAM
```

```
CASE G:REPLY = '4'
  RELEASE ALL LIKE G:*
```


RETURN

***** IF CHOICE IS TC ADD, PROMPT FOR AND ACCEPT INPUT

```
CASE G:REPLY = '1'
  STORE ' ' TO G:IM
  STORE ' ' TC G:COG
  ERASE
  @ 6,24 SAY 'ENTER DATA FOR COG BEING ADDED'
  @ 9,32 SAY 'COG ' GET G:COG PICTURE '9A'
  READ
  STORE ! (G:CCG) TO G:COG
  SELECT PRIMARY
```

***** CHECK FOR DUPLICATE RECORD

```
USE D:COG INDEX D:COGS
FIND &G:COG
IF # <> 0
  @ 22,14 SAY 'RECORD CURRENTLY EXISTS ' ;
  + ' - STRIKE ANY KEY TO CONTINUE' + CHR(7)
  WAIT
ELSE
  STORE T TC G:GETIM
  DO WHILE G:GETIM
```

***** PROMPT FOR AND ACCEPT ITEM MANAGER

```
@ 10,32 SAY 'ITEM MANAGER ' GET G:IM
SELECT SECONDARY
USE D:ADDRESS INDEX D:IM
READ
STORE !(G:IM) TC G:IM
```

***** CHECK TO SEE IF IM RECORD EXISTS

```
FIND &G:IM
IF # <> 0
  STORE F TC G:GETIM
  SELECT PRIMARY
  APPEND BLANK
  REPLACE COG WITH G:COG, IM WITH G:IM
ELSE
```

***** IF ITEM MANAGER NOT ON FILE PROVIDE OPTION TO
***** CORRECT IM CODE, ADD THE IM RECORD OR EXIT WITHOUT
***** UPDATE

```
@ 16,27 SAY 'ITEM MANAGER NOT ON FILE'
@ 18,30 SAY '1 - CHANGE I.M. CODE'
@ 19,30 SAY '2 - ADD ITEM MANAGER'
@ 20,30 SAY '3 - EXIT' + CHR(7)
STORE ' ' TO G:REPLY2
@ 23,40 GET G:REPLY2 PICTURE '9'
READ
DO WHILE G:REPLY2<'1' .OR. G:REPLY2>'3'
  @ 23,40 GET G:REPLY2
  READ
ENDDO
DO CASE
  CASE G:REPLY2 = '1'
    @ 16,27 SAY ' ' ;
    + ' ' ;
    @ 18,30 SAY ' ' ;
    @ 19,30 SAY ' ' ;
    @ 20,30 SAY ' ' ;
    @ 23,40 SAY ' ' ;
  CASE G:REPLY2 = '2'
    STORE F TO G:GETIM
```



```

STORE ' ' ;
TO G:TITLE
STORE ' ' ;
+ ' TO G:COMMAND
STORE ' ' ;
G:TITLE TC;
STORE ' ' ;
+ ' TO G:COMMAND2
STORE ' ' ;
TO G:ATTN
STORE ' ' ;
TO G:STREET
STORE ' ' ;
TO G:CITY
STORE ' ' TO G:STATE
STORE ' ' TO G:ZIP
@ 16,27 SAY ' ' ;
+ ' '
@ 18,30 SAY ' ' ;
@ 19,30 SAY ' ' ;
@ 20,30 SAY ' ' ;
@ 23,40 SAY ' ' ;
@ 14,16 SAY ' TITLE ' GET G:TITLE
@ 15,16 SAY ' COMMAND ' GET ;
G:COMMAND
@ 16,16 SAY ' COMMAND 2 ' GET ;
G:COMMAND2
@ 17,16 SAY ' ATTN ' GET ;
G:ATTN
@ 18,16 SAY ' STREET ' GET ;
G:STREET
@ 19,16 SAY ' CITY ' GET G:CITY
@ 19,44 SAY ' STATE ' GET G:STATE
@ 19,56 SAY ' ZIP CODE ' GET G:ZIP;
PICTURE '99999'
READ

```

***** ACCEPT NEW IM DATA AND PROMPT FOR CREATE/EXIT OPTION

```

@ 21,20 SAY '1 - POST NEW RECORD';
+ ' 2 - EXIT'
STORE ' ' TO G:REPLY3
@ 23,40 GET G:REPLY3 PICTURE '9'
READ
DO WHILE G:REPLY3 < '1' .OR.;
G:REPLY3 > '2'
@ 23,40 GET G:REPLY3
READ
ENDDO

```

***** CREATE A NEW RECORD

```

IF G:REPLY3 = '1'
SELECT PRIMARY
APPEND BLANK
REPLACE COG WITH G:COG,;
IM WITH G:IM
SELECT SECONDARY
APPEND BLANK
REPLACE IM WITH !(G:IM), TITLE;
WITH !(G:TITLE), COMMAND WITH !(G:COMMAND), COMMAND2 WITH;
!(G:COMMAND2), ATTN WITH !(G:ATTN), STREET WITH !(G:STREET),;
CITY WITH !(G:CITY), ZIP WITH G:ZIP
ELSE
STORE F TO G:GETIM
ENDIF

```

***** EXIT WITHOUT CREATING RECORD

```

CASE G:REPLY2 = '3'

```



```

                                STORE F TO G:GETIM
                                ENICASE
                                ENDIF
                                ENDDO
                                ENDIF

```

```

***** IF CHOICE IS TO UPDATE

```

```

CASE G:REPLY = '2'
  STCRE ' ' TO G:IM
  STORE ' ' TO G:COG

```

```

***** PROMPT FOR AND ACCEPT COG BEING UPDATED

```

```

ERASE
@ 6,24 SAY 'ENTER DATA FOR COG BEING UPDATED'
@ 9,32 SAY 'COG ' GET G:COG PICTURE '9A'
READ
STORE ! (G:CCG) TO G:COG
SELECT PRIMARY

```

```

***** RETREIVE RECORD TO BE UPDATED

```

```

USE D:COG INDEX D:COGS
FIND &G:COG
IF # = 0
  @ 22,17 SAY 'RECORD NOT FOUND ' ;
  + ' - STRIKE ANY KEY TO CONTINUE' + CHR(7)
  WAIT
ELSE
  STORE T TO G:PROCESS

```

```

***** IF RECORD DELETED PROMPT FOR REACTIVATION

```

```

IF *
  STORE ' ' TO G:REPLY3
  @ 18,22 SAY 'RECORD HAS BEEN MARKED FOR';
  + ' DELETION ' ;
  @ 19,19 SAY 'DO YOU WANT THIS COG ' ;
  + ' REACTIVATED <Y OR N>' ;
  @ 21,40 GET G:REPLY3 PICTURE 'A'
  READ
  DC WHILE ! (G:REPLY3) <> 'Y' .AND. ! (G:REPLY3) <> 'N'
    @ 23,31 SAY 'ENTER Y OR N ONLY' + CHR(7)
    @ 21,40 GET G:REPLY3 PICTURE 'A'
    READ
  ENDDO

```

```

***** REACTIVATE RECCRD IF REQUESTED

```

```

IF ! (G:REPLY3) = 'Y'
  @ 18,22 SAY ' ' ;
  + ' ' ;
  @ 19,19 SAY ' ' ;
  + ' ' ;
  @ 21,40 SAY ' ' ;
  RECALL
ELSE
  STCRE F TO G:PROCESS
ENDIF
ENDIF
DO WHILE G:PROCESS
  STORE IM TO G:IM

```

```

***** PROMPT FOR AND ACCEPT UPDATE INFORMATION

```

```

@ 10,32 SAY 'ITEM MANAGER ' GET IM
READ
@ 21,20 SAY '1 - POST UPDATE INFORMATION';

```



```

                + '          2 - EXIT'
STORE ' ' TO G:REPLY3
@ 23,40 GET G:REPLY3 PICTURE '9'

***** ACCEPT UPDATE/EXIT SELECTION

READ
DO WHILE G:REPLY3 < '1' .OR. G:REPLY3 > '2'
  @ 23,40 GET G:REPLY3
  READ
ENDDO

***** IF EXIT WITHOUT UPDATE, RESTORE RECORD TO ORIGINAL
***** VALUE

IF G:REPLY3 <> '1'
  REPLACE IM WITH G:IM
ENDIF
USE
STORE F TO G:PROCESS
ENDDO
ENDIF

***** IF CHOICE IS TC DELETE

CASE G:REPLY = '3'
  STORE ' ' TO G:IM
  STORE ' ' TC G:COG

***** PROMPT FOR AND ACCEPT COG BEING DELETED

ERASE
@ 6,28 SAY 'ENTER COG BEING DELETED'
@ 9,32 SAY 'COG          ' GET G:COG PICTURE '9A'
READ
STORE ! (G:CCG) TO G:COG
SELECT PRIMARY
USE D:COG INDEX D:COGS

***** VERIFY COGS EXISTENCE

FIND &G:COG
IF # = 0
  @ 22,17 SAY 'RECORD NOT FOUND ';
  + '- STRIKE ANY KEY TO CONTINUE' + CHR(7)
  WAIT
ELSE

***** VERIFY THAT NC ACTIVE CASES ARE ASSIGNED TO THIS COG

IF COUNT > 0
  @ 10,32 SAY 'ITEM MANAGER ' + IM
  @ 13,25 SAY 'ACTIVE CASES EXIST FOR THIS COG'
  @ 14,15 SAY 'ALL ACTIVE CASES MUST BE ';
  + 'REASSIGNED TO ANOTHER COG'
  @ 15,21 SAY 'BEFORE DELETE ACTION CAN BE ';
  + 'COMPLETED'
  @ 18,27 SAY 'STRIKE ANY KEY TO CONTINUE' ;
  + CHR(7)
  WAIT
ELSE

***** NOTIFY OPERATOR THAT RECORD PREVIOUSLY DELETED

IF *
  @ 23,13 SAY 'RECORD PREVIOUSLY DELETED - ';
  + 'STRIKE ANY KEY TO CONTINUE'
  WAIT
ELSE

```



```
@ 10,32 SAY 'ITEM MANAGER ' + IM
@ 17,20 SAY '1 - DELETE THIS COG';
      + '      2 - EXIT'
STORE ' ' TO G:REPLY3
@ 19,40 GET G:REPLY3 PICTURE '9'
```

```
***** ACCEPT DELETE/EXIT SELECTION
```

```
      READ
      DO WHILE G:REPLY3 < '1' .OR. G:REPLY3 > '2'
        @ 23,32 SAY 'ENTER 1 - 2 ONLY' + CHR(7)
        @ 19,40 GET G:REPLY3
        READ
      ENDDO
      IF G:REPLY3 = '1'
        DELETE
        USE
      ENLIF
    ENLIF
  ENLIF
ENICASE
USE
ENDDC
***** END OF PROGRAM
```


XXII. DATA BASE PACK MODULE

```

*****
**
** DATE: 15 JAN 1984
** VERSION: 1.0
** MODULE NAME: DBPACK
** MODULE PURPOSE: PACK THE DATA BASE AND REMOVE
** RECORDS TAGGED FOR DELETION
**
** MODULE INTERFACE DEFINITION
** INPUTS: C:WHO
** OUTPUTS: NONE
**
** MODULE PROCESSING NARRATIVE DESCRIPTION:
** COMPRESSES THE DATA BASES BY REMOVING RECORDS
** MARKED FOR DELETION. PRIOR TO EXECUTION, THE
** USER'S PASSWORD IS VERIFIED TO ENSURE THAT HE
** IS AUTHORIZED TO PERFORM THE PACK.
**
** SUPERORDINATE MODULES: UTILMENU
** SUBORDINATE MODULES: NONE
** AUTHCR: R. G. NICHOLS
**
*****

```

***** DISPLAY WARNING MESSAGE

```

ERASE
@ 1,25 SAY '***** DATA BASE PACKING *****'
@ 3,24 SAY '* * * * *'
@ 4,24 SAY '*
@ 5,24 SAY '          WARNING
@ 6,24 SAY '*
@ 7,24 SAY '*      THIS PROGRAM WILL PACK
@ 8,24 SAY '*      ALL DELETED CASES AND THEN
@ 9,24 SAY '*      WILL RE-INDEX THE FILES
@ 10,24 SAY '*
@ 11,24 SAY '*      IF EXISTING FILES ARE
@ 12,24 SAY '*      LARGE, THIS COULD TAKE
@ 13,24 SAY '*      HOURS
@ 14,24 SAY '*
@ 15,24 SAY '* * * * *
@ 17,24 SAY '*      ARE YOU SURE YOU WANT TO'
@ 18,24 SAY '*      CONTINUE'
@ 19,24 SAY '*      <ENTER Y OR N>' + CHR(7)
STCRÉ ' ' TC P:REPLY2
@ 21,40 GET P:REPLY2

```

***** ACCEPT RESPONSE FROM USER

```

READ
DO WHILE !(P:REPLY2)<>'Y' .AND. !(P:REPLY2)<>'N'
@ 23,32 SAY 'ENTER Y OR N ONLY' + CHR(7)
@ 21,40 GET P:REPLY2 PICTURE 'A'
READ
ENDDO
@ 23,32 SAY '
@ 17,40 SAY '

```

***** ACCEPT AND VERIFY PASSWORD PRIOR TO EXECUTION


```

IF P:REPLY2 = 'Y'
@ 21,30 SAY 'ENTER YOUR PASSWORD '
STORE ' ' TO P:PASSWORD
SET CONSCIE OFF
ACCEPT TC P:PASSWORD
SET CONSCIE ON
IF P:PASSWORD <> ' '
USE D:TECHCODE INDEX D:TECH
FIND &C:WHO
IF PSWD = P:PASSWORD .AND. # <> 0

```

***** DISPLAY PROCESSING MESSAGE

```

ERASE
@ 6,26 SAY ' OPEN DATA BASE BEING PURGED'
@ 8,32 SAY 'OF CLOSED CASES'
@ 16,29 SAY '***** DO NOT INTERRUPT *****'
USE D:OPEN1 INDEX D:OCASE1, D:ONSN
PACK
USE D:OPEN2 INDEX D:OCASE2
PACK
ELSE
@ 23,18 SAY 'REQUEST ABORTED ':
+ '- STRIKE ANY KEY TO CONTINUE'
WAIT
ENDIF
ENDIF
USE
RELEASE ALL IIKE P:*
RETURN

```

***** END OF PROGRAM

XXIII. ANALYST FILE UPDATE MODULE

```
*****  
**  
** DATE: 15 JAN 1984  
** VERSION: 1.0  
** MODULE NAME: ANALYST  
** MODULE PURPOSE: TO ADD, UPDATE, DELETE, AND LIST  
** ANALYST INFORMATION  
**  
** MODULE INTERFACE DEFINITION  
** INPUTS: NCNE  
** OUTPUTS: NONE  
**  
** MODULE PROCESSING NARRATIVE DESCRIPTION:  
** PROVIDE CAPABILITY TO ADD NEW ANALYST CODES,  
** UPDATE EXISTING ANALYST RECORDS, DELETE  
** ANALYST RECORDS, OR DISPLAY ANALYST RECORDS.  
** NEW ANALYST IDS ARE VALIDATED TO ENSURE THAT  
** DUPLICATE RECORDS ARE NOT CREATED AND THAT NO  
** EMBEDDED BLANKS APPEAR IN THE ID. DELETED  
** ANALYST RECORDS MAY BE RECALLED BY UPDATING  
** THE RECORD. PRIOR TO DELETION OF A RECORD,  
** THE ANALYST IS VERIFIED TO HAVE NO ACTIVE  
** CASES ASSIGNED.  
**  
** SUPERORDINATE MODULES: UTILMENU  
** SUBORDINATE MODULES: NONE  
** AUTHCR: R. G. NICHOLS  
**  
*****
```

```
STORE T TO A:CONTINUE  
DO WHILE A:CONTINUE
```

```
***** DISPLAY OPTIONS AVAILABLE TO THE USER AND ACCEPT  
***** SELECTION
```

```
ERASE  
@ 6,25 SAY ' ***** ANALYST FILE UPDATE ***** '  
@ 9,28 SAY '1 - ADD ANALYST'  
@ 10,28 SAY '2 - UPDATE ANALYST'  
@ 11,28 SAY '3 - DELETE ANALYST'  
@ 12,28 SAY '4 - LIST ANALYST'  
@ 13,28 SAY '5 - RETURN TO UTILITY MENU'  
STORE ' ' TO A:REPLY  
@ 16,40 GET A:REPLY PICTURE '9'  
READ
```

```
***** VALIDATE SELECTION
```

```
DO WHILE A:REPLY < '1' .OR. A:REPLY > '5'  
@ 23,32 SAY 'ENTER 1 - 5 ONLY' + CHR(7)  
@ 16,40 GET A:REPLY PICTURE '9'  
READ  
ENDDO  
DC CASE
```

```
***** IF QUIT REQUEST, RELEASE LOCAL MEMORY VARIABLES AND  
***** RETURN TO CALLING PROGRAM
```

```
CASE A:REPLY = '5'  
RELEASE ALL LIKE A:*
```


RETURN

***** IF ADD NEW ANALYST SELECTED

```
CASE A:REPLY = '1'  
  STCRE ' ' TO A:TECHCODE  
  STORE ' ' TO A:PASSWORD  
  STCRE ' ' TO A:NAME
```

***** CLEAR SCREEN AND PROMPT FOR NEW ANALYST INFORMATION

```
ERASE  
@ 6,22 SAY 'ENTER DATA FOR ANALYST BEING ADDED'  
@ 7,22 SAY ' FOLLOW EACH ENTRY WITH A <CR>'  
@ 10,28 SAY 'ANALYST CODE ' GET A:TECHCODE  
READ
```

***** VALIDATE NO ENEEDED BLANKS

```
DO WHILE $(A:TECHCODE,1,1)=' ' .OR. ;  
  $(A:TECHCCDE,2,1)=' ' .OR. $(A:TECHCODE,3,1);  
  =' ' .OR. $(A:TECHCODE,4,1)=' ' ;  
  @ 23,23 SAY 'ANALYST CODE CANNOT CONTAIN';  
  + ' BLANKS' + CHR(7)  
  @ 10,45 GET A:TECHCODE  
  READ  
ENDDO  
@ 23,23 SAY '  
STORE ! (A:TECHCODE) TO A:TECHCODE
```

***** VALIDATE FOR DUPLICATE USER ID

```
USE D:TECHCCDE INDEX D:TECH  
FIND &A:TECHCODE  
IF # <> 0  
  @ 22,14 SAY 'RECORD CURRENTLY EXISTS '  
  + '- STRIKE ANY KEY TO CONTINUE'  
  WAIT  
ELSE  
  @ 12,28 SAY 'ANALYST NAME ' GET A:NAME  
  READ  
  SET CONSOLE OFF  
  STORE T TC A:ENTERPSW
```

***** PROMPT FOR USER PASSWORD AND VERIFICATION OF THE
***** PASSWORD

```
@ 14,28 SAY 'PASSWORD '  
DO WHILE A:ENTERPSW  
  @ 14,44 SAY ' '  
  ACCEPT TO A:PASSWORD  
  STORE ' ' TO A:VERIFY  
  @ 16,28 SAY 'VERIFY PASSWORD '  
  ACCEPT TO A:VERIFY  
  IF A:PASSWORD <> A:VERIFY  
    @ 23,5 SAY 'VERIFICATION PASSWORD DOES '  
    + 'NOT MATCH - REENTER PASSWORD';  
    + ' AND REVERIFY' + CHR(7)  
    STCRE ' ' TO A:PASSWORD  
  ELSE  
    STCRE F TC A:ENTERPSW  
  ENDIF  
ENDDO  
SET CONSOLE ON
```

***** CREATE THE NEW ANALYST RECORD

```
APPEND BLANK  
REPLACE TECHCODE WITH ! (A:TECHCODE) , NAME WITH;
```



```

                ! (A:NAME), PSWD WITH A:PASSWORD
            USE
        ENDIF

***** IF UPDATE ANALYST SELECTED
        CASE A:REPLY = '2'
            STORE ' ' TO A:NAME
            STORE ' ' TO A:TECHCODE

***** PFOMPT FOR AND ACCEPT ANALYST CODE
            ERASE
            @ 6,15 SAY 'ENTER ANALYST CODE FOR RECORD TO BE':
                + ' UPDATED ' GET A:TECHCODE
            READ
            STORE ! (A:TECHCODE) TO A:TECHCODE

***** VALIDATE CODES EXISTENCE
            USE D:TECHCODE INDEX D:TECH
            FIND &A:TECHCODE
            IF # = 0
                @ 22,17 SAY 'RECORD NOT FOUND ':
                    + '- STRIKE ANY KEY TO CONTINUE'
                WAIT
            ELSE
                STORE T TC A:PROCESS

***** IF MARKED FOR DELETION, SEE IF RECORD SHOULD BE
***** REACTIVATED
            IF *
                STORE ' ' TO A:REPLY2
                @ 18,22 SAY 'RECORD HAS BEEN MARKED FOR':
                    + ' DELETION'
                @ 19,18 SAY 'DO YOU WANT THIS ANALYST ':
                    + 'REACTIVATED <Y OR N>'
                @ 21,40 GET A:REPLY2 PICTURE 'A'
                READ
                DO WHILE A:REPLY2<>'Y' .AND. A:REPLY2<>'N'
                    @ 23,31 SAY 'ENTER Y OR N ONLY' + CHR(7)
                    @ 21,40 GET A:REPLY2 PICTURE 'A'
                    READ
                ENDDO
                IF ! (A:REPLY2) = 'Y'
                    RECALL
                    @ 18,22 SAY ' ';
                    + ' ';
                    @ 19,18 SAY ' ';
                    + ' ';
                    @ 21,40 SAY ' '
                ELSE
                    STORE F TC A:PROCESS
                ENDIF
            ENDIF

***** PFOMPT FOR AND ACCEPT UPDATE INFORMATION
            DO WHILE A:PROCESS
                STORE NAME TO A:NAME
                @ 8,15 SAY 'ENTER NEW NAME DATA ' GET NAME
                READ
                @ 21,20 SAY '1 - POST UPDATE INFORMATION':
                    + ' 2 - EXIT'
                STORE ' ' TO A:REPLY2
                @ 23,40 GET A:REPLY2 PICTURE '9'
            ***** ACCEPT UPDATE/EXIT OPTION SELECTION

```



```

      READ
      DO WHILE A:REPLY2 < '1' .OR. A:REPLY2 > '2'
        @ 23,40 GET A:REPLY2 PICTURE '9'
      READ
    ENDDO
    IF A:REPLY2 = '1'
      REPLACE NAME WITH !(NAME)
    ELSE
      REPLACE NAME WITH A:NAME
    ENDIF
    STORE F TO A:PROCESS
  ENDDO
USE
ENDIF

```

***** IF DELETE OPTION SELECTED

```

CASE A:REPLY = '3'
  STCRE ' ' TO A:NAME
  STCRE ' ' TO A:TECHCODE

```

***** PROMPT FOR AND ACCEPT ANALYST CODE

```

ERASE
@ 6,15 SAY 'ENTER ANALYST CODE FOR RECORD TO BE DELETED ';
  GET A:TECHCODE
READ
STCRE !(A:TECHCODE) TO A:TECHCODE
USE D:TECHCODE INDEX D:TECH
FIND &A:TECHCODE
IF # = 0
  @ 22,17 SAY 'RECORD NOT FOUND - STRIKE ANY KEY';
    + ' TC CONTINUE'
  WAIT
ELSE

```

***** CHECK FOR ACTIVE RECORDS ASSIGNED

```

IF ACTIVE>0 .OR. TRANSMIT>0 .OR. RESPOND>0
  @ 9,21 SAY 'ACTIVE RECORDS EXIST FOR ';
    + ' THIS ANALYST'
  @ 10,21 SAY 'ALL ACTIVE RECORDS MUST BE';
    + ' REASSIGNED'
  @ 11,21 SAY ' PRIOR TO DELETION'
  WAIT
ELSE

```

***** INDICATE IF RECORD PREVIOUSLY DELETED

```

IF *
  @ 23,13 SAY 'RECORD PREVIOUSLY DELETED ';
    + '- STRIKE ANY KEY TO CONTINUE'
  WAIT
ELSE

```

***** PROVIDE OPTION TO DELETE OR EXIT

```

  @ 8,32 SAY 'ANALYST ' + NAME
  @ 17,20 SAY '1 - DELETE THIS ANALYST ' ;
    + ' 2 - EXIT'
  STCRE ' ' TO A:REPLY2
  @ 23,40 GET A:REPLY2 PICTURE '9'
  READ
  DO WHILE A:REPLY2 < '1' .OR. A:REPLY2 > '2'
    @ 23,40 GET A:REPLY2 PICTURE '9'
  READ
  ENDDO
  IF A:REPLY2 = '1'
    DELETE

```



```
                ENCLIF
            ENCLIF
        ENCLIF
    ENCLIF
```

```
***** IF LIST OPTION
```

```
    CASE A:REPLY = '4'
        USE D:TECHCODE INDEX D:TECH
        SET DELETED CN
        ERASE
        DISPLAY ALL FIELD TECHCODE, NAME OFF
        ?
        ?
        ?
        ?
        SET DELETED CFF
        WAIT
```

```
                STRIKE ANY KEY TO CCNTINUE'
```

```
    ENDCASE
ENDDC
```

```
***** END OF PROGRAM
```


XXIV. PASSWORD FILE UPDATE MODULE

```

*****
**
** DATE: 15 JAN 1984
** VERSION: 1.0
** MODULE NAME: PASS
** MCDUIE FURPOSE: FASSWORD UPDATING
**
** MODULE INTERFACE DEFINITION
** INPUTS: NONE
** OUTPUTS: NONE
**
** MODULE PROCESSING NARRATIVE DESCRIPTION:
** ACCEPTS THE USER ID AS INPUT, REQUESTS THE
** CURRENT PASSWORD, VALIDATES IT, AND REQUESTS
** THE ENTRY AND VALIDATION OF THE NEW PASSWORD.
** AN ILLEGAL USER ID OR AN ILLEGAL PASSWORD WILL
** CAUSE THE PASSWORD UPDATE TO TERMINATE.
**
** SUPERORDINATE MODULES: UTILMENU
** SUECRDINATE MODUIES: NONE
** AUTHCR: R. G. NICHOLS
**
*****

```

***** CLEAR SCREEN AND PROMPT FOR USER ID

```

ERASE
STORE ' ' TO F:PASSWORD
STORE ' ' TO P:TECHCODE
@ 6,21 SAY '***** PASSWORD UPDATE PROCESSING *****'
@ 9,24 SAY 'ENTER DESIRED ANALYST CODE ' GET P:TECHCODE

```

***** ACCEPT AND VALIDATE USER ID

```

READ
STORE !(P:TECHCODE) TC P:TECHCODE
USE D:TECHCCDE INDEX D:TECH
FIND &P:TECHCODE
IF # = 0
@ 22,14 SAY 'RECORD DOES NOT EXIST ' ;
+ '- STRIKE ANY KEY TO CONTINUE'+CHR(7)
WAIT
ELSE
SET CCNSCLE OFF
SET EXACT ON

```

***** ACCEPT AND VALIDATE PASSWORD

```

@ 11,24 SAY 'ENTER CURRENT PASSWORD '
ACCEPT TO P:PASSWORD
IF P:PASSWORD = ' '
SET CONSOLE ON
RELEASE ALL LIKE P:*
RETURN
ENDIF
STORE P:PASSWORD+' ' TO P:PASSWORD
IF $(P:PASSWORD,1,8) <> PSWD
@ 22,8 SAY 'INVALID PASSWORD FOR ANALYST '+P:TECHCODE;
+ '- STRIKE ANY KEY TO CONTINUE' + CHR(7)
WAIT
ELSE

```



```

STORE T TO P:GETIPASWD
***** ACCEPT NEW PASSWORD AND VALIDATION OF NEW PASSWORD
DC WHILE P:GETIPASWD
  @ 13,24 SAY 'ENTER NEW PASSWORD '
  ACCEPT TO P:PASSWORD
  @ 15,24 SAY 'VERIFY NEW PASSWORD '
  ACCEPT TO P:VERIFYPW
  IF P:PASSWORD <> P:VERIFYPW
    @ 23,5 SAY 'VERIFICATION PASSWORD DOES NOT MATCH';
    + ' - REENTER PASSWORD AND REVERIFY' + CHR(7)
  ELSE
    STORE F IC P:GETIPASWD
  ENDIF
ENDDO
REPLACE PSWD WITH P:PASSWORD
USE
ENDIF
SET EXACT OFF
SET CCNSCLE ON
ENDIF

***** RELEASE LOCAL MEMORY VARIABLES AND RETURN TO
***** CALLING PROGRAM

RELEASE ALL LIKE P:*
RETURN

***** END OF PROGRAM

```


XXV. DATA BASE RE-INDEX MODULE

```
*****
**
** DATE: 20 JANUARY 1984
** VERSICN: 1.0
** MODULE NAME: UTIINDX
** MODULE PURPOSE: RE-INDEX ALL INDEX FILES
** MODULE INTERFACE DEFINITION
** INPUTS: C:WHO, C:JULIAN
** OUTPUTS: NONE
** MODULE PROCESSING NARRATIVE DESCRIPTION:
**
** A UTILITY FOR THE SUPERVISOR TO RECONSTITUTE
** THE INDEX FILES WHEN THE SYSTEM DESTROYS THE
** CURRENT INDEXES. AFTER ACCEPTANCE OF THE
** SUPERVISOR'S CHOICE TO PROCEED, EACH INDEX FILE
** IS DELETED AND THEN REBUILT USING THE DATA IN
** ALL OF THE DATABASE FILES. THIS TAKES A LONG
** TIME TO PROCESS, AND CAN BE ACCOMPLISHED ONLY
** WHEN IT IS THE ONLY PROGRAM RUNNING ON THE
** QDR SYSTEM.
**
** SUPERORDINATE MODULES: UTILMENU
** SUBORDINATE MODULES: NONE
** AUTHCR: J.G. BOYNTON
**
*****
```

***** DELETE CURRENT INDICES

```
DELETE FILE D:OCASE1.NDX
DELETE FILE D:ONSN.NDX
DELETE FILE D:OCASE2.NDX
DELETE FILE D:CCASE1.NDX
DELETE FILE D:CNSN.NDX
DELETE FILE D:CCASE2.NDX
```

***** BEGIN REINDEX OF FILES

```
USE D:OPEN1
INDEX ON CASE TO D:OCASE1
INDEX ON NSN TO D:ONSN
```

```
USE D:OPEN2
INDEX ON CASE TO D:OCASE2
```

```
USE D:CLCSE1
INDEX ON CASE TO D:CCASE1
INDEX ON NSN TO D:CNSN
```

```
USE D:CLCSE2
INDEX ON CASE TO D:CCASE2
```

```
USE D:TECHCCDE
INDEX ON TECHCODE TO D:TECH
```

```
USE D:COG
INDEX ON COG TO D:COGS
```

```
USE D:WHEREEIS
INDEX ON CODE TO D:DISCODE
```


USE D:ADDRESS
INDEX ON IM TO D:IM

RETURN

***** END OF PROGRAM

XXVI. OPEN CASE REPORT

```
*****  
**  
** DATE: 5 JANUARY 1984 **  
** VERSICN: 1.0 **  
** MODUIE NAME: OCASERPT **  
** MODULE PURPOSE: PROVIDE ANALYST WITH LISTING OF ALL **  
** CF HIS OPEN CASES IN THE DATA BASE. **  
** MODULE INTERFACE DEFINITION **  
** INPUTS: C:WHO, C:JULIAN **  
** OUTPUTS: NONE **  
** MODULE PROCESSING NARRATIVE DESCRIPTION: **  
**  
** ALLOWS THE ANALYST TO CHOOSE BETWEEN RECEIVING **  
** A LIST OF HIS CURRENTLY OPEN CASES OR TO RETURN **  
** TO THE MAIN PROCESSING MENU. THE PROGRAM DOES A **  
** SEQUENTIAL SEARCH OF THE DATA BASE TO IDENTIFY **  
** THE APPROPRIATE CASES, AND LISTS THEM TO EITHER **  
** THE SCREEN OR THE PRINTER. LIST SHOULD NOT BE **  
** SENT TO THE PRINTER IF ANYONE ELSE WILL BE **  
** USING IT BEFORE THE PROCESS IS COMPLETED. **  
**  
** SUPERORDINATE MODULES: MENU1 **  
** SUEORDINATE MODULES: NONE **  
** AUTHOR: J.G. BOYNTON **  
**  
*****  
ERASE  
STORE ' ' TO V:PRINT  
TEXT
```

YOU MAY RECEIVE THE REPORT ON THE SCREEN OR
AT THE PRINTER

- 1 - SCREEN
- 2 - PRINTER
- 3 - EXIT

< ENTER YOUR CHOICE >

```
ENDTEXT  
@ 22,35 SAY ' ' GET V:PRINT  
READ
```

```
IF V:PRINT = '1'  
ERASE  
USE D:CPEN1  
REPORT FORM OPENCASE FOR WHO = C:WHO  
? 'PRESS ANY KEY TO CONTINUE'  
WAIT
```

```
ELSE  
IF V:PRINT = '2'  
ERASE
```



```
USE D:OPEN1
SET PRINT CN
REPORT FORM OPENCASE FOR WHO = C:WHO
EJECT
SET PRINT OFF
? 'PRESS ANY KEY TO CONTINUE'
WAIT
    ENDIF
ENDIF
RELEASE V:PRINT
RETURN

***** END OF PROGRAM
```


LIST OF REFERENCES

1. Carriger, M. D., A System Analysis and Design For Updating the Internal Tracking of the Quality Deficiency Reporting System at the Navy's Fleet Material Support Office, M. S. Thesis, Naval Postgraduate School, Monterey, Ca., 1983
2. Sommerville, I., Software Engineering, Addison-Wesley Publishers Limited, 1982
3. Parnas, D. I., "Designing Software for Ease of Extension and Contraction," Tutorial on Software Design Techniques, Third Edition, IEEE Computer Society, 1980
4. Pressman, R. S., Software Engineering: A Practitioner's Approach, McGraw-Hill Book Company, 1982

BIBLIOGRAPHY

Freeman, Peter, The Context of Design, Tutorial on Software Design Techniques, Third Edition, IEEE Computer Society, 1980.

IBM, Technical Reference Manual, Version 2.02, Revised Edition, April 1983.

Navy Fleet Material Support Office, Defective Material Report Program Users Manual, FMSO Document No. F9333-I32-9703-UM-01, 15 June 1980.

Orchid Technology, ECnet: A Local Area Network For The IBM PC/XT, 1983.

Ratliff, Wayne, dBase II: Assembly Language Relational Database Management System, Ashton-Tate, 1982.

Shooman, Martin I., Software Engineering, Design, Reliability and Managemnt, McGraw-Hill Book Company, 1983.

INITIAL DISTRIBUTION LIST

		No. Copies
1.	Defense Technical Information Center Cameron Station Alexandria, Virginia 22314	2
2.	Library (Code 0142) Naval Postgraduate School Monterey, Ca 93943	2
3.	Naval Postgraduate School (Code 37) Computer Technologies Curriculum Office Monterey, Ca 93943	1
4.	Associate Professor Norman Lyons (Code 541b) Department of Administrative Science Naval Postgraduate School Monterey, Ca 93943	1
5.	Commanding Officer Navy Fleet Material Support Office (Code 93) SPCC Complex, Bldg 409 Mechanicsburg, Pa 17055	5
6.	Majcr John G. Bcynton U. S. Army Command & General Staff College Class 84/85 Fort Leavenworth, Kansas 66027	1
7.	LCDR Ronald G. Nichols Navy Fleet Material Support Office SPCC Complex, Bldg 409 Mechanicsburg, Pa 17055	1
8.	Capt M. D. Carriger Headquarters, U. S. Marine Corps (Attn: Code CCIS, Room 3037) Washington, D. C. 20380	1

205390

Thesis
B79565
c. 1

Boynton

The creation of a da-
tabase on a microcompu-
ter network.

SEP 23 85
22 MAR 88

30624
32179

205390

Thesis
B79565
c. 1

Boynton

The creation of a da-
tabase on a microcompu-
ter network.



thesB79565

The creation of a database on a micro-co



3 2768 002 07419 7

DUDLEY KNOX LIBRARY