<table>
<thead>
<tr>
<th>Network(U) Naval Postgraduate School Monterey CA</th>
<th>J G Boynton et al. Mar 84</th>
</tr>
</thead>
</table>

The creation of a central database on a microcomputer network (U).
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
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

Forald 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

Authors: ________________________________

______________________________

Approved by: ________________________________

Thesis Advisor

______________________________

Second Reader

______________________________

Chairman, Department of Administrative Science

______________________________

Dean of Information and Policy Sciences
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
BIBLIOGRAPHY ............... 217
INITIAL DISTRIBUTION LIST .......... 218
# LIST OF FIGURES

<table>
<thead>
<tr>
<th>Figure</th>
<th>Description</th>
<th>Page</th>
</tr>
</thead>
<tbody>
<tr>
<td>2.1</td>
<td>Basic QDR Flcw and Initial QDR Receipt</td>
<td>10</td>
</tr>
<tr>
<td>2.2</td>
<td>Qdr Update and Close</td>
<td>11</td>
</tr>
<tr>
<td>2.3</td>
<td>User's System Hierarchy</td>
<td>13</td>
</tr>
<tr>
<td>2.4</td>
<td>Supervisor's System Hierarchy</td>
<td>14</td>
</tr>
<tr>
<td>3.1</td>
<td>Logon Prompt</td>
<td>18</td>
</tr>
<tr>
<td>3.2</td>
<td>Invalid I.D. Message</td>
<td>19</td>
</tr>
<tr>
<td>3.3</td>
<td>Currently Logged Message</td>
<td>19</td>
</tr>
<tr>
<td>3.4</td>
<td>Login Prompt</td>
<td>19</td>
</tr>
<tr>
<td>3.5</td>
<td>Main Menu</td>
<td>20</td>
</tr>
<tr>
<td>3.6</td>
<td>Verification Message</td>
<td>21</td>
</tr>
<tr>
<td>3.7</td>
<td>Initial Entry Screen</td>
<td>22</td>
</tr>
<tr>
<td>3.8</td>
<td>Bailout /Change Option</td>
<td>22</td>
</tr>
<tr>
<td>3.9</td>
<td>First Screen of data</td>
<td>23</td>
</tr>
<tr>
<td>3.10</td>
<td>Second Screen of Data</td>
<td>24</td>
</tr>
<tr>
<td>3.11</td>
<td>Response to EOST Choice</td>
<td>25</td>
</tr>
<tr>
<td>3.12</td>
<td>Feedback to Analyst</td>
<td>25</td>
</tr>
<tr>
<td>3.13</td>
<td>Database Handler Access Codes</td>
<td>27</td>
</tr>
<tr>
<td>3.14</td>
<td>Database Handler Return Codes</td>
<td>28</td>
</tr>
<tr>
<td>3.15</td>
<td>Response Times</td>
<td>38</td>
</tr>
</tbody>
</table>
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 (FMSO), 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 Upgrading 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 CDR 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 database 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 (PDREP) increased the files to forty-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.

16
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 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 3.3). If the correct Access I.D. is entered then the system will prompt to enter the password. The password requires


center

enter password followed by <cr>

center

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).

<table>
<thead>
<tr>
<th>WELCOME TO THE QDR AUTOMATED TRACKING SYSTEM</th>
</tr>
</thead>
<tbody>
<tr>
<td>1 - Open New Record</td>
</tr>
<tr>
<td>2 - Close Record</td>
</tr>
<tr>
<td>3 - Update Record</td>
</tr>
<tr>
<td>4 - Originate Letter</td>
</tr>
<tr>
<td>5 - Report Generation</td>
</tr>
<tr>
<td>6 - Query</td>
</tr>
<tr>
<td>7 - Exit from the System</td>
</tr>
<tr>
<td>Enter Your Choice</td>
</tr>
</tbody>
</table>

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).
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.
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 CDR System.
<table>
<thead>
<tr>
<th>#</th>
<th>Field Description</th>
<th>Example</th>
</tr>
</thead>
<tbody>
<tr>
<td>1</td>
<td>NSN</td>
<td></td>
</tr>
<tr>
<td>2</td>
<td>CATEGORY</td>
<td></td>
</tr>
<tr>
<td>3</td>
<td>SMIC</td>
<td></td>
</tr>
<tr>
<td>4</td>
<td>UIC</td>
<td></td>
</tr>
<tr>
<td>5</td>
<td>REPORT CONTROL</td>
<td></td>
</tr>
<tr>
<td>6</td>
<td>DATE DISCOVERED</td>
<td>MMDDYY</td>
</tr>
<tr>
<td>7</td>
<td>NOMENCLATURE</td>
<td></td>
</tr>
<tr>
<td>8</td>
<td>FSCM</td>
<td>&lt;0&gt;</td>
</tr>
<tr>
<td>9</td>
<td>MFG. PART NUMBER</td>
<td>&lt;0&gt;</td>
</tr>
<tr>
<td>10</td>
<td>SERIAL/LOT/BATCH</td>
<td>&lt;0&gt;</td>
</tr>
<tr>
<td>11</td>
<td>CONTRACT/PO</td>
<td>&lt;0&gt;</td>
</tr>
<tr>
<td>12</td>
<td>DOCUMENT NUMBER</td>
<td>&lt;0&gt;</td>
</tr>
<tr>
<td>13</td>
<td>ITEM N OR O</td>
<td>&lt;0&gt;</td>
</tr>
<tr>
<td>14</td>
<td>DATE MFG/REP/OVHL</td>
<td>&lt;0&gt;</td>
</tr>
<tr>
<td>15</td>
<td>CFN TIME AT FAILURE</td>
<td>&lt;0&gt;</td>
</tr>
<tr>
<td>16</td>
<td>GOV FURNISHED MATL</td>
<td>&lt;0&gt;</td>
</tr>
<tr>
<td>17</td>
<td>CTY: REC/INSPECT/DEF/STK</td>
<td>&lt;0&gt;</td>
</tr>
<tr>
<td>18</td>
<td>TYPE/MODEL/SERIES</td>
<td>&lt;0&gt;</td>
</tr>
<tr>
<td>19</td>
<td>SERIAL NUMBER</td>
<td>&lt;0&gt;</td>
</tr>
<tr>
<td>20</td>
<td>NEXT HIGHER ASSY</td>
<td>&lt;0&gt;</td>
</tr>
<tr>
<td>21</td>
<td>SUB-ASSEMBLY</td>
<td>&lt;0&gt;</td>
</tr>
</tbody>
</table>

****** CHECK PREVIOUS ENTRIES ******

CHOOSE 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.
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.

<table>
<thead>
<tr>
<th>First Character</th>
<th>Database File Used</th>
</tr>
</thead>
<tbody>
<tr>
<td>1</td>
<td>OPEN1</td>
</tr>
<tr>
<td>2</td>
<td>OPEN2</td>
</tr>
<tr>
<td>3</td>
<td>CLOSE1</td>
</tr>
<tr>
<td>4</td>
<td>CLOSE2</td>
</tr>
</tbody>
</table>

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

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 Figure 3.14).

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

<table>
<thead>
<tr>
<th>Return Code</th>
<th>Definition</th>
</tr>
</thead>
<tbody>
<tr>
<td>0</td>
<td>Data Base Access Successful</td>
</tr>
<tr>
<td>1</td>
<td>Record Currently Locked</td>
</tr>
<tr>
<td>2-8</td>
<td>Unassigned (Available for future growth)</td>
</tr>
<tr>
<td>9</td>
<td>Record Not Found</td>
</tr>
</tbody>
</table>

Figure 3.14  Data Base Handler Return Codes.

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 XEBHNDLR (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 ADDITION

  2 - List of assigned open cases for specific COG

***** VALIDATING ENTRY

STORE T TO BADCCG
DO WHILE BADCOG
  @ LINE,COLUMN SAY 'Enter COG' GET ANSWER
  READ
  USE D:COGS
  FIND ANSWER
  IF @ 25,20 SAY 'COG NOT FOUND, TRY AGAIN'
  ELSE
    STORE F TO BADCCG
  ENDIF
ENDDO <BADCCG>

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

USE D:OPEN
DISPLAY CASE, NSN, WOMEN,5(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
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 microcomputer 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 MegaFlus board which provided 64 KB of the 320 KE 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 (XOPENZ). 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 FC 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.

<table>
<thead>
<tr>
<th>OPERATION</th>
<th>SIZE</th>
<th>100 rcds</th>
<th>6000 rcds</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td>Users</td>
<td>1</td>
<td>2</td>
</tr>
<tr>
<td>2. Case Update</td>
<td></td>
<td>:38</td>
<td>1:27</td>
</tr>
<tr>
<td>2nd screen</td>
<td></td>
<td>:17</td>
<td>:37</td>
</tr>
</tbody>
</table>

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 ME 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:

<table>
<thead>
<tr>
<th>Variable Name</th>
<th>Using Modules</th>
<th>Description Of Variable</th>
</tr>
</thead>
<tbody>
<tr>
<td>C:JULIAN</td>
<td>(CLOSREC, LOGON, MENU1, XDBHNDLR, XOPEN2, XUPDAT, XXBISTAT, XXMNSTAT)</td>
<td>MEMORY VARIABLE WHICH HOLDS TODAY'S JULIAN DATE. THIS DATE IS GENERATED BY ACCESSING THE SYSTEM CALENDAR AND CONVERTING TO A JULIAN DATE.</td>
</tr>
<tr>
<td>C:WHC</td>
<td>(CLOSREC, COGNT, C-REASGN, DEPACK, LOGON, MENU1, OCASERPT, QUERY, RPTMENU, STATGEN, SUPMNU1, SUPRETS, SUPRPT2, UTILMENU, UTILNDX, XDBHNDLR, XOPEN2, XUPDAT, XXBISTAT, XXMNSTAT)</td>
<td>MEMORY VARIABLE THAT HOLDS THE LOGON IDENTIFICATION OF THE ANALYST. THIS IS CAPTURED DURING THE LOGON PROCESS.</td>
</tr>
<tr>
<td>M:ACTDISF</td>
<td>(XDBHNDLR, XOPEN2, XUPDAT)</td>
<td>MEMORY VARIABLE WHICH IDENTIFIES THE ACTION/DISPOSITION INSTRUCTIONS. ORIGINALLY CAPTURED FROM BLOCK 21 OF THE SF 368.</td>
</tr>
<tr>
<td>M:ACTPT</td>
<td>(XDBHNDLR, XOPEN2, XUPDAT)</td>
<td>MEMORY VARIABLE WHICH IDENTIFIES THE ACTION POINT.</td>
</tr>
</tbody>
</table>
M:ACTKTN (XDBHNDLR, XUPDAT)
MEMORY VARIABLE WHICH IDENTIFIES THE ACTION CODE. ORIGINALLY CAPTURED FROM BLOCK 21 OF THE SF 368.

M:CASE (CLOSREC, 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 (CLOSREC, XOPEN2, XUPDAT)
MEMORY VARIABLE WHICH HOLDS THE DATE THE CASE WAS CLOSED. ORIGINALLY ENTERED WHEN CLOSING THE CASE.

M:COG (C-REASGN, XDBHNDLR, XOPEN2, XUPDAT, XXBISTAT)
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 (CLOSREC, XDBHNDLR, XUPDAT)
MEMORY VARIABLE WHICH IDENTIFIES THE CREDIT CODE.

M:DATES (CLOSREC, C-REASGN, XDBHNDLR, XOPEN2, XUPDAT, XXBISTAT, XXMNSTAT)
MEMORY VARIABLE WHICH HOLDS THE CONCATINATION OF THE MAJOR LATES 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 CODE.

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 CDR.
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:EPAC (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
(CLOSEREC, C-REASGN, XDBHNDLR, XOPEN2)
MEMORY VARIABLE WHICH CONTAINS THE DATABASE ACCESS KEY.

M:DATE
(CLOSEREC, 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 165(3) OF SF 368.

M:MFG
(XDBHNDLR, XOPEN2, XUPDAT)
MEMORY VARIABLE WHICH IDENTIFIES THE MANUFACTURERS PART NUMBER. ORIGINALLY CAPTURED FROM BLOCK 165(3) OF SF 368.

M:NOMEN
(XDBHNDLR, XOPEN2, XUPDAT)

M:NSN
(C-REASGN, XDBHNDLR, XOPEN2, XUPDAT)
MEMORY VARIABLE WHICH IDENTIFIES THE NATIONAL STOCK NUMBER. ORIGINALLY CAPTURED FROM BLOCK 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:O9Q** *(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/COVERHAUL. 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 BLOCK 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:REFCON (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:RETIC (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 RECEIVING A RESPONSE FROM THE ITEM MANAGER.

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

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

M:STATSC (XEBHNDLB, XUPDAT)
MEMORY VARIABLE WHICH IDENTIFIES THE STATUS CODE.

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

M:TYPEE (CLOSREC, C-REASON, XEBHNDLB, XOPEN2, XUPDAT)
MEMORY VARIABLE WHICH HOLDS THE CODE SPECIFYING THE DATABASE HANDLER ACCESS CODE.

M:UI (XEBHNDLE, 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:UIIC (XEBHNDLB, 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:UIPEC (XEBHNDLB, 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:VLIC (CLOSREC, XEBHNDLA, 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

<table>
<thead>
<tr>
<th>FLD</th>
<th>NAME</th>
<th>TYPE</th>
<th>WIDTH</th>
<th>DEC</th>
</tr>
</thead>
<tbody>
<tr>
<td>001</td>
<td>CASE</td>
<td>C</td>
<td>007</td>
<td></td>
</tr>
<tr>
<td>002</td>
<td>CCG</td>
<td>C</td>
<td>002</td>
<td></td>
</tr>
<tr>
<td>003</td>
<td>NSN</td>
<td>C</td>
<td>013</td>
<td></td>
</tr>
<tr>
<td>004</td>
<td>CAT</td>
<td>C</td>
<td>001</td>
<td></td>
</tr>
<tr>
<td>005</td>
<td>NCMEN</td>
<td>C</td>
<td>019</td>
<td></td>
</tr>
<tr>
<td>006</td>
<td>UIC</td>
<td>C</td>
<td>006</td>
<td></td>
</tr>
<tr>
<td>007</td>
<td>UI</td>
<td>C</td>
<td>002</td>
<td></td>
</tr>
<tr>
<td>008</td>
<td>QTYDEF</td>
<td>N</td>
<td>009</td>
<td>002</td>
</tr>
<tr>
<td>009</td>
<td>OPRC</td>
<td>N</td>
<td>012</td>
<td>002</td>
</tr>
<tr>
<td>010</td>
<td>EERC</td>
<td>N</td>
<td>003</td>
<td></td>
</tr>
<tr>
<td>011</td>
<td>ORG</td>
<td>C</td>
<td>001</td>
<td></td>
</tr>
<tr>
<td>012</td>
<td>DCC</td>
<td>C</td>
<td>014</td>
<td></td>
</tr>
<tr>
<td>013</td>
<td>DCCNO</td>
<td>C</td>
<td>014</td>
<td></td>
</tr>
<tr>
<td>014</td>
<td>DATES</td>
<td>C</td>
<td>046</td>
<td></td>
</tr>
<tr>
<td>015</td>
<td>REPCON</td>
<td>C</td>
<td>012</td>
<td></td>
</tr>
<tr>
<td>016</td>
<td>FSCM</td>
<td>C</td>
<td>006</td>
<td></td>
</tr>
<tr>
<td>017</td>
<td>TIME</td>
<td>C</td>
<td>011</td>
<td></td>
</tr>
<tr>
<td>018</td>
<td>WHO</td>
<td>C</td>
<td>004</td>
<td></td>
</tr>
<tr>
<td>019</td>
<td>NUM</td>
<td>C</td>
<td>017</td>
<td></td>
</tr>
<tr>
<td>020</td>
<td>CS</td>
<td>C</td>
<td>001</td>
<td></td>
</tr>
<tr>
<td>021</td>
<td>SCR</td>
<td>C</td>
<td>003</td>
<td></td>
</tr>
<tr>
<td>022</td>
<td>SN</td>
<td>C</td>
<td>002</td>
<td></td>
</tr>
<tr>
<td>023</td>
<td>OSQ</td>
<td>C</td>
<td>001</td>
<td></td>
</tr>
<tr>
<td>024</td>
<td>DIC</td>
<td>C</td>
<td>002</td>
<td></td>
</tr>
<tr>
<td>025</td>
<td>VUC</td>
<td>C</td>
<td>011</td>
<td></td>
</tr>
<tr>
<td>026</td>
<td>ACTPT</td>
<td>C</td>
<td>011</td>
<td></td>
</tr>
<tr>
<td>027</td>
<td>SCRQTY</td>
<td>N</td>
<td>006</td>
<td></td>
</tr>
</tbody>
</table>

** TOTAL ** 00219 BYTES

### STRUCTURE FOR FILE: D:OPEN2 .DBF

<table>
<thead>
<tr>
<th>FLD</th>
<th>NAME</th>
<th>TYPE</th>
<th>WIDTH</th>
<th>DEC</th>
</tr>
</thead>
<tbody>
<tr>
<td>001</td>
<td>CASE</td>
<td>C</td>
<td>007</td>
<td></td>
</tr>
<tr>
<td>002</td>
<td>QTYENS</td>
<td>N</td>
<td>006</td>
<td></td>
</tr>
<tr>
<td>003</td>
<td>QTYREC</td>
<td>N</td>
<td>006</td>
<td></td>
</tr>
<tr>
<td>004</td>
<td>QTYSTK</td>
<td>N</td>
<td>006</td>
<td></td>
</tr>
<tr>
<td>005</td>
<td>DEFV</td>
<td>C</td>
<td>001</td>
<td></td>
</tr>
<tr>
<td>006</td>
<td>DFR</td>
<td>C</td>
<td>001</td>
<td></td>
</tr>
<tr>
<td>007</td>
<td>TITEM</td>
<td>C</td>
<td>001</td>
<td></td>
</tr>
<tr>
<td>008</td>
<td>OVER</td>
<td>C</td>
<td>005</td>
<td></td>
</tr>
<tr>
<td>009</td>
<td>OTP</td>
<td>C</td>
<td>005</td>
<td></td>
</tr>
<tr>
<td>010</td>
<td>GCV</td>
<td>C</td>
<td>001</td>
<td></td>
</tr>
<tr>
<td>011</td>
<td>TIME</td>
<td>C</td>
<td>011</td>
<td></td>
</tr>
<tr>
<td>012</td>
<td>WHO</td>
<td>C</td>
<td>004</td>
<td></td>
</tr>
<tr>
<td>013</td>
<td>DIITEM</td>
<td>C</td>
<td>035</td>
<td></td>
</tr>
<tr>
<td>014</td>
<td>CCGST</td>
<td>N</td>
<td>012</td>
<td>002</td>
</tr>
<tr>
<td>015</td>
<td>WNTY</td>
<td>C</td>
<td>001</td>
<td></td>
</tr>
<tr>
<td>016</td>
<td>WUC</td>
<td>C</td>
<td>007</td>
<td></td>
</tr>
<tr>
<td>017</td>
<td>DIS</td>
<td>C</td>
<td>002</td>
<td></td>
</tr>
<tr>
<td>FLD</td>
<td>NAME</td>
<td>TYPE</td>
<td>WIDTH</td>
<td>DEC</td>
</tr>
<tr>
<td>-----</td>
<td>--------</td>
<td>------</td>
<td>-------</td>
<td>-----</td>
</tr>
<tr>
<td>001</td>
<td>CASE</td>
<td>C</td>
<td>007</td>
<td></td>
</tr>
<tr>
<td>002</td>
<td>CCG</td>
<td>C</td>
<td>002</td>
<td></td>
</tr>
<tr>
<td>003</td>
<td>NSN</td>
<td>C</td>
<td>013</td>
<td></td>
</tr>
<tr>
<td>004</td>
<td>CAT</td>
<td>C</td>
<td>001</td>
<td></td>
</tr>
<tr>
<td>005</td>
<td>KCMEN</td>
<td>C</td>
<td>019</td>
<td></td>
</tr>
<tr>
<td>006</td>
<td>UIC</td>
<td>C</td>
<td>006</td>
<td></td>
</tr>
<tr>
<td>007</td>
<td>UI</td>
<td>C</td>
<td>002</td>
<td></td>
</tr>
<tr>
<td>008</td>
<td>OTYDEF</td>
<td>N</td>
<td>009</td>
<td>002</td>
</tr>
<tr>
<td>009</td>
<td>UERF</td>
<td>N</td>
<td>012</td>
<td>002</td>
</tr>
<tr>
<td>010</td>
<td>EFRE</td>
<td>N</td>
<td>003</td>
<td></td>
</tr>
<tr>
<td>011</td>
<td>DCC</td>
<td>C</td>
<td>001</td>
<td></td>
</tr>
<tr>
<td>012</td>
<td>DCCNO</td>
<td>C</td>
<td>014</td>
<td></td>
</tr>
<tr>
<td>013</td>
<td>DATES</td>
<td>C</td>
<td>046</td>
<td></td>
</tr>
<tr>
<td>014</td>
<td>ERPCON</td>
<td>C</td>
<td>012</td>
<td></td>
</tr>
<tr>
<td>015</td>
<td>FSCM</td>
<td>C</td>
<td>006</td>
<td></td>
</tr>
<tr>
<td>016</td>
<td>TIME</td>
<td>C</td>
<td>011</td>
<td></td>
</tr>
<tr>
<td>017</td>
<td>WHO</td>
<td>C</td>
<td>017</td>
<td></td>
</tr>
<tr>
<td>018</td>
<td>NUM</td>
<td>C</td>
<td>017</td>
<td></td>
</tr>
<tr>
<td>019</td>
<td>SCR</td>
<td>C</td>
<td>001</td>
<td></td>
</tr>
<tr>
<td>020</td>
<td>SQR</td>
<td>C</td>
<td>003</td>
<td></td>
</tr>
<tr>
<td>021</td>
<td>SNQ</td>
<td>C</td>
<td>002</td>
<td></td>
</tr>
<tr>
<td>022</td>
<td>DEF</td>
<td>C</td>
<td>012</td>
<td></td>
</tr>
<tr>
<td>023</td>
<td>RSM</td>
<td>C</td>
<td>002</td>
<td></td>
</tr>
<tr>
<td>024</td>
<td>VIC</td>
<td>C</td>
<td>014</td>
<td></td>
</tr>
<tr>
<td>025</td>
<td>ACTPT</td>
<td>N</td>
<td>001</td>
<td></td>
</tr>
<tr>
<td>026</td>
<td>SCRQTY</td>
<td>S</td>
<td>006</td>
<td></td>
</tr>
</tbody>
</table>

** TOTAL ** 00219 BYTES
<table>
<thead>
<tr>
<th>FLD</th>
<th>NAME</th>
<th>TYPE</th>
<th>WIDTH</th>
<th>DEC</th>
</tr>
</thead>
<tbody>
<tr>
<td>001</td>
<td>NAME</td>
<td>C</td>
<td>020</td>
<td></td>
</tr>
<tr>
<td>002</td>
<td>TECHCODE</td>
<td>C</td>
<td>004</td>
<td></td>
</tr>
<tr>
<td>003</td>
<td>PSWD</td>
<td>C</td>
<td>008</td>
<td></td>
</tr>
<tr>
<td>004</td>
<td>LOGGED</td>
<td>C</td>
<td>001</td>
<td></td>
</tr>
<tr>
<td>005</td>
<td>ASSIGNED</td>
<td>N</td>
<td>004</td>
<td></td>
</tr>
<tr>
<td>006</td>
<td>ACTIVE</td>
<td>N</td>
<td>004</td>
<td></td>
</tr>
<tr>
<td>007</td>
<td>TRANSMIT</td>
<td>N</td>
<td>004</td>
<td></td>
</tr>
<tr>
<td>008</td>
<td>RESPOND</td>
<td>N</td>
<td>004</td>
<td></td>
</tr>
<tr>
<td>009</td>
<td>CLOSED</td>
<td>N</td>
<td>004</td>
<td></td>
</tr>
</tbody>
</table>

** TOTAL ** 00054 BYTES
** TOTAL **
00172 BYTES

** STRUCTURE FOR FILE: D:WERDIS .DBF **

<table>
<thead>
<tr>
<th>FLD</th>
<th>NAME</th>
<th>TYPE</th>
<th>WIDTH</th>
<th>DEC</th>
</tr>
</thead>
<tbody>
<tr>
<td>001</td>
<td>CCDE</td>
<td>C</td>
<td>002</td>
<td></td>
</tr>
<tr>
<td>002</td>
<td>TEXT</td>
<td>C</td>
<td>020</td>
<td></td>
</tr>
</tbody>
</table>

** TOTAL **
00023 BYTES

** STRUCTURE FOR FILE: D:FILESTAT .DBF **

<table>
<thead>
<tr>
<th>FLD</th>
<th>NAME</th>
<th>TYPE</th>
<th>WIDTH</th>
<th>DEC</th>
</tr>
</thead>
<tbody>
<tr>
<td>001</td>
<td>OPEN1</td>
<td>C</td>
<td>004</td>
<td></td>
</tr>
<tr>
<td>002</td>
<td>OPEN2</td>
<td>C</td>
<td>004</td>
<td></td>
</tr>
<tr>
<td>003</td>
<td>CLOSE1</td>
<td>C</td>
<td>004</td>
<td></td>
</tr>
<tr>
<td>004</td>
<td>CLOSE2</td>
<td>C</td>
<td>004</td>
<td></td>
</tr>
</tbody>
</table>

** TOTAL **
00017 BYTES

** STRUCTURE FOR FILE: D:BIWKSTAT .DBF **

<table>
<thead>
<tr>
<th>FLD</th>
<th>NAME</th>
<th>TYPE</th>
<th>WIDTH</th>
<th>DEC</th>
</tr>
</thead>
<tbody>
<tr>
<td>001</td>
<td>YEAR</td>
<td>C</td>
<td>004</td>
<td></td>
</tr>
<tr>
<td>002</td>
<td>TOTALS</td>
<td>N</td>
<td>005</td>
<td></td>
</tr>
<tr>
<td>003</td>
<td>LAST</td>
<td>C</td>
<td>005</td>
<td></td>
</tr>
</tbody>
</table>

** 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

<table>
<thead>
<tr>
<th>Data Element</th>
<th>ID</th>
<th>Picture</th>
<th>Type</th>
</tr>
</thead>
<tbody>
<tr>
<td>Case Number</td>
<td>M:CASE</td>
<td>'999999A'</td>
<td>A7</td>
</tr>
<tr>
<td>Cognizance</td>
<td>M:CCG</td>
<td>'XX'</td>
<td>A2</td>
</tr>
<tr>
<td>Category</td>
<td>M:CAT</td>
<td>'9'</td>
<td>A1</td>
</tr>
<tr>
<td>Nomenclature</td>
<td>M:NCMEN</td>
<td>'XXX...XX'</td>
<td>A19</td>
</tr>
<tr>
<td>UIC</td>
<td>M:UIC</td>
<td>'AXXXX'</td>
<td>A6</td>
</tr>
<tr>
<td>Unit of issue</td>
<td>M:UI</td>
<td>'AA'</td>
<td>A2</td>
</tr>
<tr>
<td>Unit Price</td>
<td>M:UPRC</td>
<td>'999999.99'</td>
<td>N9</td>
</tr>
<tr>
<td>Quantity Def.</td>
<td>M:QTYDEF</td>
<td>'999999'</td>
<td>N6</td>
</tr>
<tr>
<td>Quantity Insp.</td>
<td>M:QTYINS</td>
<td>'999999'</td>
<td>N6</td>
</tr>
<tr>
<td>Quantity Recvd</td>
<td>M:QTRYEC</td>
<td>'999999'</td>
<td>N6</td>
</tr>
<tr>
<td>Qty in Stock</td>
<td>M:QTYSTK</td>
<td>'999999'</td>
<td>N6</td>
</tr>
<tr>
<td>Extended Price</td>
<td>M:EPRE</td>
<td>'9999999999.99'</td>
<td>N12</td>
</tr>
<tr>
<td>Origin</td>
<td>M:OOG</td>
<td>'XXX'</td>
<td>A3</td>
</tr>
<tr>
<td>Deficiency Ver</td>
<td>M:DEFV</td>
<td>'A'</td>
<td>A1</td>
</tr>
<tr>
<td>Deficiency Resp</td>
<td>M:DEFR</td>
<td>'A'</td>
<td>A1</td>
</tr>
<tr>
<td>Type Document</td>
<td>M:DCC</td>
<td>'9'</td>
<td>A1</td>
</tr>
<tr>
<td>Discovery Date</td>
<td>M:DATE</td>
<td>'99999'</td>
<td>A5</td>
</tr>
<tr>
<td>Date Rcvd fm Org</td>
<td>M:RDAT</td>
<td>'99999'</td>
<td>A5</td>
</tr>
<tr>
<td>Open Date</td>
<td>M:OPEN</td>
<td>'99999'</td>
<td>A5</td>
</tr>
<tr>
<td>Date Ltr Typed</td>
<td>M:LTET</td>
<td>'99999'</td>
<td>A5</td>
</tr>
<tr>
<td>Screen Art date</td>
<td>M:SCEDAT</td>
<td>'99999'</td>
<td>A5</td>
</tr>
<tr>
<td>Interim Resp Dat</td>
<td>M:IRDAT</td>
<td>'99999'</td>
<td>A5</td>
</tr>
<tr>
<td>Date rtn fm IM</td>
<td>M:RIMDAT</td>
<td>'999999'</td>
<td>A5</td>
</tr>
<tr>
<td>Close Date</td>
<td>M:CLOSE</td>
<td>'999999'</td>
<td>A5</td>
</tr>
<tr>
<td>Reopen Date</td>
<td>M:REOPEN</td>
<td>'999999'</td>
<td>A5</td>
</tr>
<tr>
<td>Dates concat'n</td>
<td>M:DATES</td>
<td>See note 1</td>
<td>A46</td>
</tr>
<tr>
<td>Date change ID</td>
<td>M:DATECI</td>
<td>'X'</td>
<td>See note 2 A1</td>
</tr>
</tbody>
</table>

Report Control #      | M:RECON | See note 3                   | A12  |
Document Number        | M:DCCNO | See note 4                   | A14  |
FSCM                   | M:FSCM  | 'XXXXXXX'                    | A6   |
Time Start             | M:TIME  | '999999999999'              | A11  |
Analyst Code           | M:WHC   | 'XXXX'                       | A4   |
New-Repair/Cvhl        | M:ITEM  | 'A'                         | A1   |
Date Mfg/Cvhl          | M:OVER  | '999999'                    | A5   |
Opn Time-Failure       | M:OTF   | 'A99999'                    | A5   |
Govtment Furnish       | M:GCV   | 'X'                         | A1   |
Work Unit Code         | M:WUC   | 'XXXXXXXX'                  | A7   |

55
**Discovery Code** | M: DIS | 'AA' | A2  
**Details Section** | M: DETAILS | 'XX...198..X' | A198  
**Return Code** | M: REHC | Q1 | A1  
**Record Variable** | M: REC1 | '99999A' | A6  
**Contract Number** | M: NUM | See note 5 | A17  
**Credit Code** | M: CR | 'A' | A1  
**Screening Code** | M: SCE | 'XXX' | A3  
**Reply Section** | M: REPLY | 'XX...198..X' | A198  
**Action Code** | M: ACTTN | '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 Effect** | M: DIF | '99' | A2  
**Vendor Lab Code** | M: VIC | 'A' | A1  
**Action Point** | M: ACTPT | 'XXXXXX99999' | A11  
**Part Number** | M: MFG | 'XX..16..XX' | A16  
**Lot/ser/latch** | M: LCT | 'XXXXXXXXX' | A9  
**NSN** | M: NSN | See note 6 | A13  
**Type/Model/Ser** | O: MCDCL | 'XXXXXXXXX' | A7  
**Def Item Ser** | O: DEFSER | 'XXXXXXXX' | A6  
**Higher Assy** | O: HASSY | 'XXXXXXXXXX' | A10  
**Sub assembly** | O: SASSY | 'XXXXXXXXXXXXX' | A12  
**Def Item** | M: DITEM | 'XXXXX 35 XXXX' | A35  
**Est Curr Cost** | M: CCCST | '999999999.99' | N12  
**Warranty** | M: WNTY | 'A' | A1  
**Screen Quantity** | M: SCFQTY | '999999' | N6  

**NOTES:**

1. All dates, followed by M:DATECT, 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) 'XXXXXXXX-99-9999'

4. M:DOCNO PICTURE 'XXXXXXXX-99999-9999'


6. NSN (FSC+NATO+FIN) '99999-XX-XXX-9999'
APPENDIX D
QDR PROGRAMS

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

57
XIX. CASE REASIGNMENT 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 REPORT ...................... 214
I. LOGON MODULE

************************************************************************************************************
**
** Date: 23 Nov 1983
** Version: 1.0
** Module Name: LOGON
** 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, Validates The Password, and Calls The Necessary Modules
**
** Superordinate Modules: None
** Subordinate Modules: SUPMENU1, MENU1, LOCKOUT
**
** Author: R. G. NICHOLS
**
************************************************************************************************************

SET TALK OFF
SET BELL OFF
SET COLOR TO 112,3
SET EXACT ON
SET CCLCN 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 ' ' TO C:WHO
   @ 11,37 GET C:WHO
   READ
   STORE '!(C:WHO) TO C:WHO
   IF C:WHO = 'QUIT'
      QUIT
   ENDF
FIND &C:WHO
ENDDC
STORE F TO V:LOGGED
***** Check To See If Previously Logged On

IF LOGGED <> 1
   @ 16,28 SAY 'USER CURRENTLY LOGGED ON'
   @ 17,22 SAY 'LOGON TERMINATED'
   @ 23,0 SAY ' '*
STORE T TO V:LOGGED
ENDIF

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

IF NOT V:LOGGED
   STORE 2 TO V:ATTEMPTS
   @ 16,30 SAY 'ENTER YOUR PASSWORD'
   @ 17,30 SAY 'followed by <CR>'
   SET CONSOLE OFF
   @ 19,35 SAY ' '*
STORE T TO V:TRUE
DO WHILE V:TRUE
   ACCEPT TO V:PSWD
   IF V:ATTEMPTS = 0 .AND. PSWD <> V:PSWD
      SET CONSOLE ON
   ELSE
      STORE 1 TO V:ATTEMPTS
      STORE V:ATTEMPTS TO V:TRUE
   ENDIF
ENDDO
SET CONSOLE ON
REPEL LOGGED WITH ' '*
USE
SET EXACT OFF
RELEASE ALL LIKE V:*

***** 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 T TO V:TRUE
   ENDIF
ENDIF
ENDDO
SET CONSOLE ON
REPEL 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 = '000C' .OR. C:WHO = '0001'
   DO C:SUPMENU1
ELSE
   DO C:MENU1
ENDDO
STORE T TO V:CONTINUE

***** Allow The Logged User To Logoff

USE D:TECHCODE INDEX D:TECH
FIND 6C:WHC
IF # = 0
   ERASE
   @ 10,32 SAY 'LOG OFF FAILURE'*CHR(7)
   @ 11,27 SAY 'CONTACT SYSTEM SUPERVISOR'
   @ 22,26 SAY 'STRIKE ANY KEY TO CONTINUE',
   WAIT
ELSE
   REPEL 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, 295, 33, 137, 22, 13, 240,

SET CALL TO 61440
CALL V:DUMMY
STORE STR(PEEK(61456),2) TO V:HOUR
STORE STR(PEEK(61455),2) TO V:MIN
STORE STR(PEEK(61454),2) TO V:SEC
IF $$(V:HOUR,1,1)=""
STORE "0"+$$(V:HOUR,2,1) TO V:HOUR
ENDIP
IF $$((V:MIN,1,1)=""
STORE "0"+$$(V:MIN,2,1) TO V:MIN
ENDIP
IF $$((V:SEC,1,1)=""
STORE "0"+$$(V:SEC,2,1) TO V:SEC
ENDIP
STORE V:HOUR+"."+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
ELSEIF
R 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: LOCKOUT **
** 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 CCICE 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
DC WHILE V:INNER > 0
IF V:TOGGLE
  21,0,29 SAY 'ILLEGAL ACCESS ATTEMPT'
  21,0,29 SAY '+CHR(7)
ELSE
  21,0,29 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

```
*** THIS SECTION ACCESSES THE SYSTEM DATE
STORE TO V: DUMMY
POKE 61440, 180, 42, 205, 33, 137, 22, 13, 240, 137;
14, 15, 240, 193
SET CALL TO 61446
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
  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 C:JULIAN
RELEASE ALL EXCEPT C:*
STORE T TO V:CONTINUE
DO WHILE V:CONTINUE
ERASE
SET TALK CFF
STORE ' ' TO V:CHOICE
TEXT

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

@ 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:CLCSEPRG
CASE V:CHOICE = 3
RELEASE ALL EXCEPT C:*
DO C:XUPDAT.PRG
CASE V:CHOICE = 4
RELEASE ALL EXCEPT C:*
DO C:LIB.PRG
CASE V:CHOICE = 5
RELEASE ALL EXCEPT C:*
DO C:RPMENU
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:CHOICE>
ENDDO <V:CONTINUE>
***** END OF PROGRAM
IV. NEW CASE INPUT MODULE

********************************************
** DATE: 18 NOV 1983 **
** VERSION: 1.0 **
** MODULE NAME: OPEN **
** MODULE PURPOSE: NEW QDR CASE CREATION **
** MODULE INTERFACE DEFINITION **
** INPUTS: C:WHO, C:JULIAN, V:JULDATE **
** DATA ELEMENTS IN OPEN1 AND OPEN2. **
** MODULE PROCESSING NARRATIVE DESCRIPTION: **
** PROMPTS THE USER FOR INPUT OF ALL DATA FROM **
** SF 368 IN ORDER TO CREATE A NEW QDR CASE. **
** VALIDATION OF DATA ITEMS OCCURS UPON INPUT AND **
** IS BASED ON CURRENT GE TIMESHARE VALIDATION. **
** AS MODIFIED BY FMSO TECHNICAL BRANCH. DATES **
** ARE CAPTURED FOR MANAGEMENT STATISTICS. **
** SUPERORDINATE MODULES: MENU1 **
** SUBORDINATE MODULES: CJULIAN, XDBHNDLR **
** AUTHOR: 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
ENDPHASE
IF C:CHOICE = '2'
RELEASE ALL EXCEPT C:
RETURN
ENDIF

***** INITIALIZE MEMORY VARIABLES
STORE ' ' TO O:KEY

66
STORE ' TO M:CASE
STORE ' TO M:COG
STORE ' TO M:SM
STORE ' TO M:CAT
STORE TC M:UIC
STORE 0 TO M:QTYBEC
STORE 0 TO M:QTYINS
STORE 0 TO M:QTYLEF
STORE 0 TO M:QTYSTK
STORE ' TO M:NUM
STORE TC M:OPEN
STORE TC M:OVER
STORE ' TO M:REPCON
STORE TO M:TIME
STORE ' TO M:MPG
STORE ' TO M:LOT
STORE ' TO M:ITEM
STORE ' TO M:OTF
STORE ' TO M:GCV
STORE ' TO M:LEP
STORE ' TO M:DOCNO
STORE ' TO M:ITEM
STORE ' TO M:LOT
STORE ' TO M:NOMEN
STORE IC M:UIC
STORE 0 TO M:QTYFEC
STORE 0 TO M:QTYINS
STORE 0 TO M:QTYSTK
STORE ' TO M:NUM
STORE ' TO M:OPEN
STORE ' TO M:OVER
STORE ' TO M:REPCON
STORE ' TO M:MPG
STORE ' TO M:LOT
STORE ' TO M:ITEM
STORE ' TO M:OTF
STORE ' TO M:GCV
STORE ' TO M:LEP
STORE ' TO M:DOCNO
STORE ' TO M:ITEM
STORE ' TO M:LOT
STORE ' TO M:NOMEN

***** THIS SEQUENCE CALCULATES THE UPPER AND LOWER YEARS
***** FOR INPUT AND IS BASED ON THE CURRENT JULIAN DATE
***** C:JULIAN. O:ULIMIT= YEAR MINUS TWO YEARS
***** O: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 ILMT
STORE TE.P1 A+HIGH TO ULMT
STORE STF(ILMT,2) TO C:LLIMIT
STORE STF(ULMT,2) TO C:ULIMIT
RELEASE TEMP1,TEMP1A,LOW,HIGH,ILMT,ULMT

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

STORE ' TC O:FLDATE
STORE T TO C:ANSWER
DO WHILE C:ANSWER
  @ 5,20 SAY '******** ENTER DATA FOR THE NEW CASE ********'
  @ 6,20 SAY '******** FROM SF 368 ********'
STORE T TO C:RDATET
DO WHILE O:RDATET
  @ 8,20 SAY 'DATE RECEIVED BY FMSO MMDDYY' :
  READ
  IF $(O:RDAT,1,2)<'01' OR 5(O:RDAT,1,2)>'12':
    OR $(O:RDAT,3,2)<'01' OR $(O:RDAT,3,2)>'31':
      OR $(O:RDAT,4,2)<O:LLIMIT:
        OR $(O:RDAT,4,2)>C:ULIMIT:
        @ 23,30 SAY 'DATE OUT OF RANGE'
  ELSE
    STORE F TO O:RDATET
  ENDDC <O:RDATET>
**ENTE THE CALL TC C:JULIAN TO CHANGE MMDDYY TC**

**JULIAN FORMATT, STORE TO M:RDATE THEN RELEASE C:RDATE**

STORE VAL($O:RDATE, 1,2)) TC V:MM  
STORE VAL($O:RDATE, 3,2)) TC V:DD  
STORE VAL($O:RDATE, 5,2)) TC V:YY  
DO C:JULIAN  
STORE V:JULIAN 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'  
.CR. 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  
DO  
WHILE O:COG1  
@ 12,20 SAY 'COG'  
@ 12,50 SAY 'GET M:COG PICTURE '9X' READ  
IF $M:COG, 2,1) = ''  
@ 23,30 SAY 'NO BLANKS IN 2D POSITION'  
ELSE  
STORE F TO C:COG1  
STORE $(M:CCG) TO M:COG  
ENDIF  
ENDDO <O:COG1>  
@ 23,30 SAY '  
**CHECKS THAT COG IS VALID IN CURRENT COG TABLE... MUST BE VALID TO CONTINUE**  
USE C:COG INDEX D:COGS  
FIND E:*: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, 5 O:CCG2>  
RELEASE O:COG1, O:CCG2  
@ 23,40 SAY '  
@ 14,20 SAY 'NSN'  
@ 14,50 SAY 'GET O:KEY PICTURE '9999-XX-XXX-9999' READ  
STORE T TO C: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  

69
23 20 SAY 'STOR! F TO O: NATOT
ENDIF
ENDDC <O: NATOT>
RELEASE C: NATOT
@ 23,20 SAY 'STORE T 10 C: FIINT
£0 WHILE O: FIINT
"C-
IF $(O: KEY, 7, 1) = " " OR. $(O: KEY, 8, 1) = " ;
@ 23,40 SAY 'NC BLANKS IN THE FIRST 3 POSITIONS'
@ 14,50 SAY 'GET O: KEY PICTURE '9999-XX-XXX-9999'
READ
ELSE
@ 23,40 SAY '
STORE F TC O: FIINT
ENDIF FIINT
ENDDC <C: FIINT>
RELEASE C: FIINT
1
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'
STCRE F TO O: ANSWER
@ 23, 20 SAY 'SEARCHING FOR ANOTHER CASE WITH THIS NSN'
RELEASE C: RDATE
ELSE
IF O: REPLY = '3'
RELEASE ALL EXCEPT C:*
RETURN
ELSE
CLEAR GETS
@ 22,25 SAY '
@ 22,25 SAY '
ENDIF
ENDDIF
ENDDC O: ANSWER
STORE $(O: KEY, 11, 4) +$(O: KEY, 6, 2) + $(O: KEY, 9, 3) ;
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: XISBNDIR.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: FREVREC
STORE T TO O: WHICH
DC WHILE O: WHICH
69
STCRE '11' TO M:TYPE
DO C:XDBHNDLR.PRG

IF M:NSN <> M:KEY .OR. ECF
STCRE O:PREVFEC TO M:REC1
STCRE '1H' TC M:TYPE
DO C:XDBHNDLR.PRG
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
STCRE '1' TO M:DATES
STCRE '1' TO M:CLOSE
STCRE '1' TO M:UIC
STCRE '1' TO M:PSCM
STCRE '1' TO M:MFG
STCRE '1' TO M:REOPEN
STCRE '1' TO M:DOCNO
STCRE '1' TO M:SCRDAT
STCRE '1' TO M:REPCON
STCRE 0 TO M:QTYINS
STCRE 0 TO M:QTYREC
STCRE 0 TO M:QTYSTK
STCRE 0 TO M:QTYDF

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
TO C:XEBENDLR.PRG

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

IF M:TYPE= '9'
STCRE 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 '31' TO M:TYPE
DO C:XDBHNDLR.PRG
STCRE M:REC1 TO O:PREVREC
IF M:NSN <> M:KEY .OR. ECF
STCRE O:PREVFEC TO M:REC1
STORE '1H' TC M:TYPE
DO C:XDBHNDLR.PRG
STORE F TO C:WHICH

70
***THESE MUST BE RELEASED OR ELSE TOO MANY VARIABLES WILL BE ASSIGNED (IE >64)


STORE ' ' TO M:DATES
STORE ' ' TO M:CLOSE
STORE ' ' TO M:UIC
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 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 <CLOSEFILE>

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

IF O:OCASE > O:CCASE
RELEASE O:CCASE,O:OCASE
ELSE
STORE O:OCASE 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 = '1'
STORE $(M:CASE,1,6) + 'A' TO M:CASE
ELSE
STORE $RANK(C:LAST) +1 TO O:SUFFIX
STORE CHR(O:SUFFIX) TO O:LETTER
STORE $(M:CASE,1,6) + O:LETTER TO M:CASE
ENDIF
ENDIF
RELEASE C:LAST,O:LETTER,O:KEY,O:PREVREC,O:CNONE,O:CNCNE,O:WHICH
***** START OF NEW CASE DATA ENTRY

0, 1 SAY 'SF368'
1, 2 SAY 'NSN'
2, 2 SAY 'CATEGORY'
3, 2 SAY 'SMIC'
4, 2 SAY 'UIC'
5, 2 SAY 'REICRT CONTROL

6, 2 SAY 'DATE DISCOVERED MMDDYY'
7, 2 SAY 'NOMENCLATURE'
8, 2 SAY 'FSCM'<0>
9, 2 SAY 'MFG. PART NUMBER'<0>
10, 2 SAY 'SPECIAL/LOT/BATCH'
11, 2 SAY 'CONTRACT/EO'<0>
12, 2 SAY 'DOCUMENT NUMBER'<0>
13, 2 SAY 'ITEM N OR O'<0>
14, 2 SAY 'DATE MFG/REP/OVHL'
15, 2 SAY 'OPEN TIME AT FAILURE'
16, 2 SAY 'GOV FURNISHED MATL'
17, 2 SAY 'QTY: REC/INSPE/DEF/STK'
18, 2 SAY 'TYPE/MODEL/SERIES'<0>
19, 2 SAY 'NEXT HIGHER ASSY'<0>
20, 2 SAY 'SUE-ASSEMBLY'<0>
21, 2 SAY 'A2. SERIAL NUMBER'<0>

2, 38 SAY $(M:KEY, 1,4) + '-' + $(M:KEY, 5,2) + '-' + $(M:KEY, 7,3):
2, 38 SAY $(M:KEY, 10,4):

CLEAR GETS

STORE T TO C:CORRECT
LO WHILE O:CORRECT

IF M:CCG = '1H' .OR. M:COG = '2H' .OR. M:COG = '7H'
STORE T TO O:SMIC1
DC WHILE O:SMIC1
@ 3, 2 SAY 'SMIC
2, 2 SAY 'SMIC PICTURE 'AX'

READ
STORE ! (M:SM) TO M:SM
IF $(M:SM, 1,1) = 'X' .OR. $(M:SM, 1,1) = 'L'
STORE F TO O:SMIC1
ELSE
ENDIF
ENDIF

RELEASE O:SMIC1
23, 30 SAY

STORE T TO O:UIC1
LO WHILE O:UIC1
@ 4,35 SAY '

STORE T TO O:UIC1
LO WHILE O:UIC1

72
STOR!E !(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) = ' ' .OR.
23,20 SAY ' NO BLANKS ALLOWED IN UIC,
ELSE
STORE F TO O:UIC1
ENDIF
ENDDO <O:UIC1>
@ 23,30 SAY '?
RELEASE C:UIC1

***** REPORT CONTROL

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

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

***** 1.TAKE DATE IC JULIAN FORMAT 2. NUMERIC DATA
STORE T TO C:DDATE
LO WHILE O:DDATE
@ 6,35 SAY ' GET O:DDATE PICTURE '9999999'
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:DEATE,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:DDATE
ENDIF
ENDIF
ENDDO <O:DDATE>
@ 23,30 SAY '?
RELEASE C:DDATE

***** 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
LO C:OJULIAN
STORE V:JULIAN TO M:ELDATE
RELEASE ALL LIKE V:*
STORE T TO C:NOMEN
LO WHILE O:NOMEN
@ 7,35 SAY ' GET M:NOMEN PICTURE 'XXXXXXXXXXXXXXXXXXXXXXXXX'
READ
IF $(M:NOMEN, 1,1) = ' ' .OR. $(M:NOMEN, 2,1) = ' ' ;
@ 23,30 SAY ' NO BLANKS IN FIRST 3 POSITIONS'
ELSE
STORE P TO O:NOMEN
ENDIF
ENDDO <O:NOMEN>
@ 23,30 SAY '?
RELEASE C:NOMEN

***** 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:UITC  
STORE T TO O:PREPT  
DO WHILE C:DOCT .OR. C:UITC  
& 12.35 SAY GET M:DOCNO PICTURE 'AXXXXX-9999-9999'  
READ  
IF M:DOCNO =  
STORE F TO O:DOCT  
STORE F TO O:UITC  
ELSE  
IF $(M:DOCNO,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 O:DOCT  
ENDIF  
IF $(M:DCCNO,12,3) > '366':  
.OR. $(M:LCNO,12,3) =  
.OR. $(M:LCNO,11,4) =  
& 23,50 SAY 'PREP DATE OUT OF RANGE.'  
ELSE  
STORE F TO O:UITC  
ENDIF  
ENDP <ALL BLANKS>  
ENDDO <O:DOCT AND. C:UITC>  
RELEASE C:UITC, 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 LCH  
STORE TEMP1A-LOW TO TEMP2  
74
STORE STR(TEMP2, 2) TC O: TENVRS
RELEASE TEMP1 TEMP1A TEMP2, LOW
STORE T TO O: OVER
DO WHILE O: OVER
  & 14.35 SAY ' ' GET M: OVER PICTURE '99999'
  READ
  IF M: OVER = ' ' STORE P TC O: OVER
  ELSE IF $(M: OVER, 3, 3) > '365':
    . OR. $(M: OVER, 1, 2) < O: TENVRS ;
    . OR. $(M: OVER, 1, 2) > O: ULIMIT :
    ELSE STORE F TO O: OVER
    ENDIF
  ENDIF
ENDDC <O: OVER>
@ 23.30 SAY ' '
ENDIF
RELEASE C: OVER, O: TENVRS
STORE T TO C: OTF
DO WHILE O: OTF
  & 15.35 SAY ' ' GET M: OTF PICTURE 'A9999'
  READ
  IF M: OTF = ' ' STORE P TC O: OTF
  ELSE IF S(M: OTF, 1, 1) = ' N' . OR. S(M: OTF, 1, 1) = ' O':
    AND. $(M: OTF, 2, 4) > '0000':
    STORE F TO C: OTF
  ELSE @ 23.30 SAY ' USE N OR O AND THEN TIME (A9999)' ENDIF
ENDIF
ENDDO <C: OTF>
@ 23.30 SAY ' '
ENDIF
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 <C: GOV>
@ 23.30 SAY ' '
ENDIF
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 ' '
ENDIF
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
ELSE
STORE F TO O:QTYINS
ENDIF
ENDDO <O:QTYINS>
@ 23,30 SAY 'OUT OF RANGE'

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 OF RANGE
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
RELEASE C:QTYSTK
@ 18,35 SAY ' GET C:MODEL PICTURE 'XXXXXXXX'
READ
@ 19,35 SAY ' GET C:DEFSER PICTURE 'XXXXXXXX'
READ
@ 20,35 SAY ' GET C:HASSY PICTURE 'XXXXXXX'
READ
@ 21,35 SAY ' GET C:SASSY PICTURE 'XXXXXXXXX'
READ
STORE O:MODEL+O:DEFSER+O:HASSY+O:SASSY TO M:DITEM

***** ERASE USER FOR RESPONSE

STORE T TO C:END
DO WHILE O:END

STORE ' ' TO 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
STORE F TO O:REPLY
ENDIF
ENDDO <O:END>

IF O:REPLY = '2'
STORE T TO O:CORRECT
@ 24,10 SAY '
@ 23,10 SAY '
ELSE

STORE F TO O:CORRECT
ENDIF
ENDDO <C:CORRECT>
ERASE
RELEASE C:MCD$EDEL,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:REPCON TC M:REPCON
STORE $(M:NUM,1,6)+$(M:NUM,8,2)+$(M:NUM,11,4)+$(M:NUM,13,4)+$(M:NUM,18,4) TO O:NUM
STORE O:NUM TO M:NUM
STORE $(M:DOCNO,1,6)+$(M:DOCN0,8,4)+$(M:DOCN0,13,4) TO ;
O:DOCNO
STORP C:DOCNO TO M:DOCNO
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+1 TO M:DATES
RELEASE M:DATES
M:DDATE,M:RDATE,M:OPEN,O:ACTPTT,O:PREPTT,
O:LLIMIT,O:ULIMIT

***** THIS IS THE START OF THE SECOND SCREEN OF DATA ENTRY
STORE ' ' TO M:UPRC
STORE ' ' TO M:WUC
STORE ' ' TO M:ACTDISP
STORE ' ' TO M:ACTPT
STORE ' ' TO M:DETAILS
STORE ' ' TO M:DEFV
STORE ' ' TO M:DEFR
STORE ' ' TO M:O9Q
STORE ' ' TO M:DOC
STORE ' ' TO M:ORG
STORE ' ' TO M:CCOST
STORE T TO C:PAGE2
DO WHILE O:PAGE2
   a 0,10 SAY ' UI ',
   a 1,10 SAY ' UNIT PRICE ' GET M:UPRC PICTURE ;
   a 2,10 SAY ' EST. CORRECTION COST ' GET M:CCOST PICTURE '999999999.99'<
   a 3,10 SAY ' WARRANTY - Y/N/U ' GET;
   a 4,10 SAY ' WORK UNIT CODE ' GET ;
   a 5,10 SAY ' DEFECT VERIFICATION CODE - N/O/U/Y ' GET ;
   a 6,10 SAY ' DETAILS OF DISCREPANCY - FIRST 2 ' ;
   a 7,10 SAY ' LETTERS MUST BE DISCOVERY CODE ' ;
   a 8,10 SAY ' GET M:DETAILS
   a 9,10 SAY ' ACTION POINT ' GET M:ACTPT PICTURE 'AXXXXX99999'
   a 10,10 SAY ' DEFECT VERIFICATION CODE - N/O/U/Y ' GET ;
   a 11,10 SAY ' DEFECT RESPONSIBILITY - C/N/S/U/X ' GET ;
   a 12,10 SAY ' 23A. ACTION POINT ' ;
   a 13,10 SAY ' 24. DEFECT VERIFICATION CODE ' ;
   a 14,10 SAY ' 25. DEFECT RESPONSIBILITY ' ;
   a 15,10 SAY ' 9Q ' ;
CET M:09Q PICTURE 'X';
GET M:ORIGIN PICTURE 'AA' 
GET M:ORG PICTURE 'AAX';
GET M:DOC PICTURE '9';
GET M:DEF PICTURE '99';

CLEAR GETS
STORE T TO O:UI
DO WHILE O:UI
  @ 0,10 SAY 'UI'
  READ
  GET M:UI PICTURE 'AA'
  IF $(M:UI,1,1) = '' OR $(M:UI,2,1) = '' @ 23,30 SAY 'NO BLANKS'
  ELSE
    STORE F TO O:UI
    ENDF.
  ENDDO O:UI
  @ 23,30 SAY 'RELEAS O:UI
STORE T TO C:EPRC
STORE T TO C:UPRC
DO WHILE O:UPRC OR O:EPRC
  DO WHILE O:UPRC
    @ 1,10 SAY 'UNIT PRICE'
    READ
    M:UPRC PICTURE '999999.99'
    IF M:UPRC < .01 OR M:UPPC > 999999.99 @ 23,30 SAY 'AMOUNT OUT OF RANGE'
    ELSE
      STORE F TO O:UPRC
    ENDF.
  ENDDO '<C:UPRC>
  @ 23,30 SAY 'STORE (M:UPRC * M:QTYDEF) TO M:EPRC
  @ 1,43 SAY 'EXT PPICE $'
  @ 1,54 SAY M:EPRC PICTURE '999999999.99'
  IF M:EPPC >= 1000000000 @ 23,30 SAY 'EXTENDED PRICE OUT OF RANGE'
  ELSE
    STORE F TO O:EPRC
  ENDF.
  ENDDO '<O:UPRC & O:EPRC>
RELEAS O:UPRC,O:EPRC
  @ 23,30 SAY 'NET CORRECTION COST $' '<O>
  GET M:CCOST PICTURE '999999999.99'
  READ
STORE T TO C:WNTY
DO WHILE O:WNTY
  @ 3,10 SAY 'WARRANTY - Y/N/U'
  STORE ! (M:WNTY) TO M:WNTY
  READ
  IF M:WNTY <> 'Y' AND M:WNTY <> 'N' @ 23,30 SAY 'USE Y,N OR U'
  ELSE
    STORE F TO O:WNTY
  ENDF.
  ENDDO '<C:WNTY>
  @ 23,30 SAY 
RELEAS O:WNTY

78
@ 4,10 SAY '20. WORK UNIT CODE
    O> GET M:WUC PICTURE 'XXXXXX'
STORE T TO C:ACTDISP
DO WHILE O:ACTDISP
    @ 5,10 SAY '21. ACTION/DISPOSITION :
        'H/I/E/E/O' O> GET M:ACTDISP PICTURE 'X'
READ
    IF M:ACTDISP = 'H' OR. M:ACTDISP = 'I' :
        +'-H/I/E/E/O'<O>
    ELSE STORE F TO O:ACTDISP
ENDIF
ENDDO <O:ACTDISP>
@ 23,30 SAY ' ERROR IN CODE'
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:DETAIS
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:ACTPTI
DO WHILE O:ACTPTI
    @ 12,10 SAY '23A. ACTION POINT
    GET M:ACTPT PICTURE 'AXXXXX9999'
READ
    IF M:ACTPT = '
        @ 23,30 SAY 'MAY NOT BE BLANK'
    ELSE
        STORE F TO O:ACTPTI
    ENDIF
ENDDO <O:ACTPTI>
@ 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
DC WHILE O:DEFR
  @ 14,10 SAY 'DEFECT RESPONSIBILITY:'
  +"C/N/S/U/X <0>" 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 = 'I' .;  
  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 = '9C'
STORE T TO O:9Q
DO WHILE O:9Q
  @ 15,10 SAY '9Q '+'
  GET M:9Q PICTURE 'X';
READ
IF $ (M:9Q 1,1) = '1' .OR. $ (M:9Q 2,1) = '1' .;  
  '+ CONTAIN BLANKS'
ELSE STORE F TO O:9Q
ENDIF
ENDDO <C:9Q>
@ 23,30 SAY '
RELEASE C:9Q
ENDIF <M:COG = '9Q>
STORE T TO C:ORG
DC WHILE O:CFG
  @ 16,10 SAY 'ORIGIN CODE:+'  
  'GET M:ORG PICTURE 'AAX';'
READ
IF $ (M:CFG 1,1) = '1' .OR. $ (M:ORG 1,1) = '1' .;  
  '+ CONTAIN BLANKS'
ELSE STORE F TO O:CFG
ENDIF
ENDDO <O:ORG>
@ 23,20 SAY '
RELEASE O:ORG
STORE T TO C:TYPE
DC WHILE O:TYPE
  @ 17,10 SAY 'TYPE DOC:+'  
  'GET M:DOC PICTURE '9';'
READ
IF M:DOC < '1' .OR. M:DOC > '7' .;  
  'OUT OF RANGE'
ELSE STORE F TO O:TYPE
ENDIF
ENDDO <C:TYPE>
@ 23,30 SAY '
RELEASE O:UIC,O:UIC2,O:PREP,O:DOC,O:SERNO,C: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 '

***** ECMFT USER FOR RESPONSE
STORE T TO C:END
DC WHILE O:END
STORE ' ' TO O:REPLY
@ <0,20 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:ENL
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 ' IF ' TO M:TYPE
DO C:XDEHNDLR.PRG
STORE ' IF ' 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  
** VERSION: 1.0  
** MODULE NAME: UPDATE  
** MODULE PURPOSE: ALLOW ADDITION AND/OR CORRECTION OF  
** DATA IN QDR CASE CURRENTLY IN QDR  
** SYSTEM  
** MODULE INTERFACE DEFINITION  
** INPUTS: CASE, C:WHO, C:JULIAN  
** OUTPUTS: ALL DATA ELEMENTS IN OPEN1 & OPEN2,  
** M:TYPE  
** MODULE PROCESSING NARRATIVE DESCRIPTION:  
** USER ENTERS CASE NUMBER OF CASE TO BE CHANGED.  
** MODULE SEARCHES DATA BASE FOR CASE AND  
** DISPLAYS 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  
** MODULE UTILIZATION.  
** SUPERORDINATE MODULES: MENU1  
** SUBORDINATE MODULES: XDEHNLLR  
** AUTHOR: J.G. BOYNTON  
************************************************************************************

STORE T TO U:UPDATE  
DO WHILE U:UPDATE  
STORE T TO U:TRUE  
DO WHILE U:TRUE  
ERASE  
STORE '1' 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'  
@ 20,20 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

**** SELECT RECORD FOR UPDATE ****

ENTER THE CASE NUMBER
OF THE RECORD TO BE UPDATED

ENDTEXT
STORE ' ' TO 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:XIBENDLR
IF M:TYPE = '9'
@ 12,25 SAY 'RECORD NOT FOUND IN OPEN FILE '
@ 13,21 SAY 'DO YOU 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
ENDIF
IF !(U:REPLY) = 'Y'
STORE '3E' TO M:TYPE
DC 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 TO U:TRUE
STORE 'CLOSE' TO U:FILE
ENDIF
ENDIF
ENDIF
ENDIF
ELSE IF M:TYPE = '1'
@ 12,26 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' TO U:FILE
ENDIF
ENDIF
ENDDO <U:TRUE>

**** THIS SECTION FOR CURRENT DATES VALUE CAPTURE

ERASE
STORE $M:DATES,1,5) TO M:ELATE
STORE \$(M:DATES,6,5) TO M:BLATE
STORE \$(M:DATES,11,5) TO M:OPEN
STORE \$(M:DATES,16,5) TO M:LLATE
STORE \$(M:DATES,21,5) TO M:SCFDATE
STORE \$(M:DATES,26,5) TO M:IRDATE
STORE \$(M:DATES,31,5) TO M:RIMDATE
STORE \$(M:DATES,36,5) TO M:CLOSE
STORE M:DDATE TO T:LLDATE
STORE M:IRDATE TO T:BLATE
STORE M:OPEN TO T:OPEN
STORE M:LLDATE TO T:LLDATE
STORE M:SCFDATE TO T:SCFDATE
STORE M:IRDATE TO T:IRDATE
STORE M:RIMDATE TO T:RIMDATE
STORE M:CLOSE TO T:CLOSE
STORE M:RECPEN TO T:REOPEN

***** THIS SEQUENCE CALCULATES THE UPPER AND LOWER YEARS FOR
***** INPUT AND IS BASED ON THE CURRENT JULIAN DATE
***** U:ULIMIT = YEAR PLUS ONE YEAR
***** U:ULIMIT = YEAR MINUS TWO YEARS
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(ULMT,2) TO U:ULIMIT
STORE STR(LLMT,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
@ 9,2 SAY 'RECEIVED FROM ORIGIN' 'GET M:DDATE
@ 10,2 SAY * OPENING DATE
@ 10,36 SAY M:OPEN
@ 11,2 SAY 'DISCOVERY DATE' 'GET M:DDATE
@ 12,2 SAY 'SCREEN REPORT DATE' 'GET M:SCFDATE
@ 14,2 SAY 'INTERIM RESPONSE DATE' 'GET M:IRDATE
@ 15,2 SAY 'CLOSE' *
@ 15,36 SAY M:CLOSE
@ 16,2 SAY 'RECPEN' 'GET M:RECPEN
@ 18,2 SAY 'MAY NOT CHANGE THESE DATES'> CLEAR GETS

STORE ' ' TC U:REPLY
STORE T TO U:DDATE
DO WHILE U:DDATE
STORE T TO U:DDATE
READ
@ 8,35 SAY 'GET M:DDATE
READ IF M:DDATE <> '
IF \$(M:DATE,1,2) < U:LLIMIT ;
.CR. \$(M:DATE,1,2) > U:ULIMIT ;
.CR. \$(M:DATE,3,3) < '001' ;
.CR. \$(M:DATE,3,3) > '365' ;
.CR. M:DATE > C:JULIAN

84
@ 23.30 SAY 'DATE OUT OF RANGE'
ELSE
STORE F TO U:DDATET
ENDIF
ELSE
STORE F TO U:DDATET
ENDIF <ELANK>
ENDO <U:DDATET>
@ 23.30 SAY '
STORE T TO U:RDATET
DO WHILE U:FDATET
@ 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:RDATET
ENDIF
ENDO <U:RDATET>
@ 23.30 SAY '
RELEASE U:DEATET,U:RDATET
STORE T TO U:LDATET
DO WHILE U:FDATET
@ 11.35 SAY '' GET M:LDATE
READ
IF $(M:LDATE) <> ''
IF $(M:LDATE,1,2) < U:LLIMIT ;
.OR. $(M:LDATE,1,2) > U:ULIMIT ;
.OR. $(M:LDATE,3,3) < '001' ;
.OR. $(M:LDATE,3,3) > '365' ;
.OR. M:LDATE > C:JULIAN ;
.OR. M:LDATE < M:OPEN
@ 23.30 SAY 'DATE OUT OF RANGE'
ELSE
STORE F TO U:LDATET
ENDIF
ELSE
STORE F TO U:LDATET
ENDIF
ENDO <U:LDATET>
@ 23.30 SAY '
STORE T TO U:SCDATET
DO WHILE U:SCDATET
@ 12.35 SAY '' GET M:SCRDATE
READ
IF $(M:SCRDATE) <> ''
IF $(M:SCRDATE,1,2) < U:LLIMIT ;
.OR. $(M:SCRDATE,1,2) > U:ULIMIT ;
.OR. $(M:SCRDATE,3,3) < '001' ;
.OR. $(M:SCRDATE,3,3) > '365' ;
.OR. M:SCRDATE < M:LDATE
@ 23.30 SAY 'DATE OUT OF RANGE'
ELSE
STORE F TO U:SCDATET
ENDIF
ENDIF
ELSE
STORE F TO U:SCDATET
ENDIF
ENDDO <U:SCDATET>
@ 23,30 SAY ''
RELEASE U:LLDATET, U:SCDATET
STORE T TO U:IRDATET
DO WHILE U:IRDATET
@ 13,35 SAY '' GET M:IRDAT
READ
IF M:IRDAT <> ''

IF $ (M:IRDAT,1,2) < U:ULLIMIT;
.C.R. $ (M:IRDAT,1,2) > U:ULLIMIT;
.C.R. $ (M:IRDAT,3,3) > '001';
.C.R. $ (M:IRDAT,3,3) > '365';
@ 23,30 SAY 'DATE OUT OF RANGE'
ELSE
STORE F TO U:IRDATET
ENDIF
ELSE
STORE F TO U:IRDATET
ENDIF
ENDDO <U:IRDATET>
@ 23,30 SAY ''
STORE T TO U:RIMDAT
DO WHILE U:RIMDAT
@ 14,35 SAY '' GET M:RIMDAT
READ
IF M:RIMDAT <> ''

IF $ (M:RIMDAT,1,2) < U:ULLIMIT;
.C.R. $ (M:RIMDAT,1,2) > U:ULLIMIT;
.C.R. $ (M:RIMDAT,3,3) > '001';
.C.R. $ (M:RIMDAT,3,3) > '365';
@ 23,30 SAY 'DATE OUT OF RANGE'
ELSE
IF M:RIMDAT < M:ILDAT
@ 23,30 SAY 'RTN DATE NOT '' BEFORE TRANSMITTAL DATE'
ELSE
STORE F TO U:RIMDAT
ENDIF
ENDIF
ELSE
STORE F TO U:RIMDAT
ENDIF
ENDDO <U:RIMDAT>
@ 23,30 SAY ''
RELEASE U:IRDATET, U:RIMDAT
STORE T TO U:REOPEN
DO WHILE U:REOPEN
@ 16,35 SAY '' GET M:REOPEN
READ
IF M:REOPEN <> ''

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

86
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
ELSE
  STORE F TO U:REOPENT
ENDIF
ENDDO <U:RECPENT>
@ 23,30 SAY ""

STORE T TO U:RECENT
DO WHILE U:END
  @ 21,10 SAY '* * * * * * CHECK DATES ABOVE * * * * * *'*
  @ 22,10 SAY '<CHOOSE> 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:DATET
  STORE T TO U:CONT1
    STORE ' ' TO M:DATECI
  ELSE
    STORE ' * IC M:DATECI
  ENDIF
  RELEASE ALL LIKE T:*
ELSE
  IF U:REPLY = '3'
    STORE F TO U:DATET
    STORE P TC U:CONT1
    STORE P TC U:CONT2
    STORE P TC U:CONT3
    IF U:FILE = 'OPEN'
      STORE '1G' TO M:TYPE
    ELSE
      STORE '3G' TO M:TYPE
    ENDIF
  DO C:XXBHNLLR
    RELEASE ALL EXCEPT C:*
  RETURN
  ENDIF
ENDIF

ENDIF
ENDDO <U:CATET>
RELEASE U:DATET, U:END
ERASE
DO WHILE U:CONT1

***** DISPLAY OF CASE DATA FROM FIRST DATABASE
@ 1,2 SAY 'NSN:' d 1,17 SAY $ (M:NSN, 3) + '-' + $ (M:NSN, 5) + '-' + $ (M:NSN, 10)
@ 1,42 SAY 'CAT' GET 4:CAT
@ 1,55 SAY 'CASE NUMBER:' d 1,67 SAY M:CASE
@ 2,2 SAY 'COG' d 2,41 SAY 'SMIC' GET M:SM PICTURE 'AX' d 2,41 SAY 'UIC' d 2,41 SAY 'RE#RT CONTROL'
@ 5,2 SAY 'ACTION POINT' d 5,2 SAY 'ACTION POINT' d 5,2 SAY 'NOMENCLATURE' d 5,2 SAY 'SERVICE NUMBER' d 6,2 SAY 'UNIT OF ISSUE' d 6,2 SAY 'UNIT PRICE' d 7,2 SAY 'ORIGIN' d 7,2 SAY 'DOCUMENT'
@ 9,2 SAY 'QUANTITY EFFICIENT' d 10,2 SAY 'UNIT OF ISSUE' d 11,2 SAY 'UNIT PRICE' d 12,2 SAY 'ORIGIN'
@ 14,2 SAY '9Q REGION CODE' d 14,2 SAY '9Q REGION CODE'
@ 15,2 SAY 'SCREEN QUANTITY' d 15,2 SAY 'SCREEN QUANTITY'
@ 16,2 SAY 'SCREEN CODE' d 16,2 SAY 'SCREEN CODE'
@ 17,2 SAY 'TYPE DOCUMENT' d 17,2 SAY 'TYPE DOCUMENT'
@ 18,2 SAY 'VENDOR LIABILITY CODE' d 18,2 SAY 'VENDOR LIABILITY CODE'
@ 19,2 SAY 'CREET COLE' d 19,2 SAY 'CREET COLE'
@ 20,2 SAY 'TYPE DEFECT' d 20,2 SAY 'TYPE DEFECT'
CLEAR GETS
STORE ' ' TO U:REPLY
@ 22,10 SAY 'ENTER <N> TO SKIP' d 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 F TO U:CONT2
IF U:FILE = 'CPEN'

STORC 'IC' TO M:TYPE
ELSE
STORC '3C' TO M:TYPE
ENDIF
ELSE
STORC T TO U:FIRSTPG
ENDIF
@ 22, 10 SAY ';
@ 23, 10 SAY ';

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

DO WHILE U:FIRSTPG
STORC 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'
STORC F TO U:CAT
ELSE
@ 23, 20 SAY ' 1 OR 2 ONLY'
ENDIF
ENDDO
U:CAT
@ 23, 20 SAY ';
RELEASE U:CAT

STORC T TO U:COG1
STORC T TO U:COG2
DC WHILE U:CCG1 OR U:CCG2
DO WHILE U:COG1
@ 2, 35 SAY ' GET M:COG PICTURE '9X'
READ
STORC !(M:COG) TO M:COG
IF !(M:COG, 2, 1) = '
@ 23, 20 SAY ' NO BLANKS IN 2D ';
+'POSITION'
ELSE
STORC 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 U:COG INDEX D:COGS
FIND &M:COG
IF # = 0
@ 23, 20 SAY ' COG INVALID - ENTER ';
+'CORRECTED ENTRY'
ELSE
STORC 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'
STORC I TO U:SMIC1
DO WHILE U:SMIC1
@ 2, 45 SAY ' GET M:SM PICTURE 'AX'
READ
STORC !(M:SM) TO M:SM
IF !(M:SM, 1, 1) = 'X' OR !(M:SM, 1, 1) = 'L'
STORC F TO U:SMIC1
89
ELSE
@ 23,30 SAY 'X OR L ONLY'
ENDIF
ENDDO <U:SMIC1>
ENDIF
RELEASE U:SMIC1
@ 23,30 SAY '

STORE T TO U:UIC1
DC 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) = ' ' @ 25,20 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:REFCON PICTURE 'XAXXXXX999999'
READ
STORE T TO U:ACTPTT
DO WHILE U:ACTPTT
@ 5,35 SAY ' ' GET M:ACTPT PICTURE 'AXXXXX999999'
READ
IF M:ACTPT = ' ' @ 25,30 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:NOMEN
@ 6,35 SAY ' ' GET M:NOMEN PICTURE 'XXXXXXXXXXXXXXXXXX'
READ
IF T(M:NOMEN, 1, 1) = ' ' OR. T(M:NOMEN, 2, 1) = ' ' OR. T(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:NOMEN

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

***** INPUT CONTRACT NUMBER
@ 8,35 SAY ' ' GET M:NUM PICTURE 'XXXXXX999XXXXXXX'

***** INPUT DOCUMENT NUMBER
STORE T TO U:UICT
STORE U:UICT TO U:PREPT
STORE U:PREPT TO U:DCCT
DO WHILE U:DOCT .OR. U:UICT
@ 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:ECCNO,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:ECCNO,12,3) >'366' i .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
STCFE 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 'STCFE 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 STCFE F TC U:UI
ENDIF
ENDDO U:UI
@ 23,30 SAY 'RELEASE U:UI
STCFE T TO U:EPRC
STCFE T TO U:UPRC
DO WHILE U:UPRC .OR. U:EPFC
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 
STCFE F TO U:UPRC
ENDIF
ENDDO U:UPRC
@ 23,30 SAY 'STORE (M:UPRC * M:QTYDEF) TO M:EPFC
STORE T TC U:EPFC
91
DO WHILE U:EPRC
  12,50 SAY 'EXT PRICE 3'
  12,61 SAY M:EPRC PICTURE '9999999999.99'
  IF M:EPRC >= 100000000
    ELSE
      STORE F TO U:EPRC
  ENDIF
ENDDO <U:EPRC>
ENDDO <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:ORG,1) = ' ' OR $(M:ORG,2) = ' ' 
     ELSE STORE F TO U:ORG
  ENDIF
ENDDO <U:ORG>
STORE T TO U:9Q
DO WHILE U:9Q
  14,35 SAY 'GET M:9Q PICTURE 'X'
  READ
  IF M:9Q = '2' OR M:9Q = '4'
     OR M:9Q = '5' OR M:9Q = '7' 
     ELSE STORE F TO U:9Q
  ENDIF
ENDDO <U:9Q>
RELEASE U:9Q
ENDIF <M:COG = '9Q>
STORE T TO U:DOC
DO WHILE U:DOC
  17,35 SAY 'GET M:DOC PICTURE '9'
  READ
  IF M:DOC < '7' 
     ELSE STORE F TO U:DOC
  ENDIF
ENDDO <U:DOC>
RELEASE U:DOC
STORE T TO U:DEF
DO WHILE U:DEF
  18,35 SAY 'GET M:VLC PICTURE 'A'
  19,35 SAY 'GET M:CR PICTURE 'A'
  20,35 SAY 'GET M:DEF PICTURE '99'
  92
READ
STORE T TO U:END
DO WHILE U:END
STORE ' ' TO U:REPLY
@ 24,10 SAY 'CHOOSE> 1- CONTINUE 2- CHANGE';
@ 23,30 SAY '* 3- EXIT';
READ U:REPLY
IF U:REPLY <> '1' .AND. U:REPLY <> '2' .AND.:
@ 23,10 SAY 'ANSWER WITH A 1- 2- 3 ONLY'
ELSE
STORE F TO U:END
ENDIF
ENDDO <U:END>
@ 23,10 SAY 'CHOOSE 1- CONTINUE 2- CHANGE';
@ 23,10 SAY '3- EXIT';
READ U:REPLY
IF U:REPLY = '2'
STORE T TO U:FIRSTPG
@ 22,10 SAY 'CHOOSE 1- CONTINUE 2- CHANGE';
@ 23,10 SAY '3- EXIT';
ELSE
IF U:REPLY = '3'
STORE P TO U:FIRSTPG
STORE P TO U:CONT1
STORE P TO U:CONT2
STORE P TO U:CONT3
IF U:FILE = 'OPEN'
STORE 'IG' TO M:TYPE
ELSE
STORE '3G' TO M:TYPE
ENDIF
DO C:KDBHNLEL
RELEASE ALL EXCEPT C:*
RETURN
ELSE
STORE P TO U:FIRSTPG
STORE T TO U:CONT1
IF U:FILE = 'OPEN'
STORE '1C' TO M:TYPE
ELSE
STORE '3C' TO M:TYPE
ENDIF
ENDIF
ENDDO <U:FIRSTPG>
ERASE
RELEASE U:END, U:CONT1, U:FIRSTPG

***** CHICE ABOVE ALLOWS ANALYST TO ABANDON OR TO CUST
***** CHANGES MADE THUS FAR
ENDDO <U:CONT1>
RELEASE U:CONT1
IF U:REPLY <> '3'
@ 10,20 SAY 'RECCFD BEING PARTIALLY UPDATED'
@ 11,20 SAY 'PLEASE STANDBY'

***** WRITE DATA TO OPEN1/CLOSE1 AND RELEASE UNNECESSARY
***** VARIABLES BEFORE READING OPEN2/CLOSE2 FOR FURTHER
***** UPDATE INFORMATION
**DC C:XEHDNLDA**

**RELEASE C:OGC, CAT. M: NUMER. M: TITLE**

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

**DO WHILE U:CONT2**

```plaintext
STORE (M:ITEM, 7) TO U:TYPE
STORE (M:ITEM, 6) TO U:SERNO
STORE (M:ITEM, 14, 12) TO U:HASSY
STORE * TO M:OVER

***** DISPLAY FOR SECOND SCREEN OF UPDATE PROGRAM *****

**ERASE**

```plaintext
1, 2 SAY 'NSN:
1, 16 SAY $(M:NSN,1,4) + ' + $(M:NSN,5,2) + ' + $(M:NSN,7,3) :
1, 55 SAY 'CASE NUMBER:
1, 67 SAY M:CASE
2, 2 SAY 'MFG. PART NUMBER
3, 2 SAY 'SEFIAL/LOT/BATCH
4, 2 SAY 'ITEM
5, 2 SAY 'DATE MFG/REF/OVHL
6, 2 SAY 'OPEN TIME AT FAILURE
7, 2 SAY 'GOV FURNISHED MATL
8, 2 SAY 'QUANTITY: RECEV/INSIP/STK
8, 43 SAY '/ GET M:QTYREC:
8, 50 SAY '/ GET M:QTYREC:
9, 2 SAY 'TYPE/MODEL/SERIES
10, 2 SAY 'SERIAL NUMBER
11, 2 SAY 'NEXT HIGHER ASSY
12, 2 SAY 'SUB-ASSEMBLY
13, 2 SAY 'ESTIMATED CORRECTION COST
14, 2 SAY 'WORK UNIT CODE
15, 2 SAY 'DEFECT VERIF - N/O/J/Y
16, 2 SAY 'DEFECT RESP - C/N/S/U/X
17, 2 SAY 'STATUS CODE
18, 2 SAY 'CAUSE CODE
19, 2 SAY 'ACTION/DISP - H/I/D/R/O
```

94
M:ACTDISP PICTURE 'X'
M:WARRANTY PICTURE 'A'
M:COST C PICTURE 'N'
M:RETURN CODE PICTURE 'A'
M:ACTION CODE PICTURE 'A'
M:ACTTN PICTURE 'A'
CLEAR GETS

@22,10 SAY 'ENTER <N> TO SKIP'
READ
IF !(U:REPLY) = 'N'
STORE F TO U:PAGE2
STORE F TO U:CONT2
STORE T TO U:CONT3
ELSE
STORE T TO U:PAGE2
ENDIF
@22,10 SAY 'COST C>
@23,10 SAY 'G71
ENDW

***** INPUT MANUFACTURERS PART NUMBER

@23,35 SAY 'GET M:MFC PICTURE ''XXXXXXXXXXXX'''
@23,35 SAY 'GET M:LOL PICTURE ''XXXXXXXX'''
STORE T TO U:ITEM
DO WHILE U:ITEM
  @23,35 SAY 'GET M:ITEM PICTURE 'A'
  READ
  IF M:ITEM = 'N' OR M:ITEM = '0' OR M:ITEM = '
    STORE F TO U:ITEM
  ELSE
    STORE F TO U:ITEM
  ENDIF
ENDDO <U:ITEM>
@23,30 SAY 'USE N OR 0'
RELEASE U: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 LOW
STORE TEMP1A-LOW TO TEMP2
STORE SIR(TEMP2.2) TO U:TENYRS
RELEASE TEMP1,TEMP1A,TEMP2,LOW
STORE T TO U:OVER
DO WHILE U:OVER
  @23,35 SAY 'GET M:OVER PICTURE ''99999''
  READ
  IF $(M:OVER,3,3) > '365':
    STORE F TO U:OVER
  ELSE
    STORE F TO U:OVER
  ENDIF
ENDDO <U:OVER>
RELEASE U:OVER,U:TENYRS
THE CREATION OF A CENTRAL DATABASE ON A MICROCOMPUTER NETWORK
NAVAL POSTGRADUATE SCHOOL MONTEREY CA
J G BOYNTON ET AL. MAR 84
UNCLASSIFIED
STORE T TO U:OTF
DO WHILE U:OTF > 6,35
  SAY ' ' GET M:OTF PICTURE 'A99999'
  READ
  IF M:OTF = ''
    STORE T TO U:OTF
  ELSE
    IF $(M:OTF,1,1) = 'N' .OR. $(M:OTF,1,1) = 'C'
      .AND. $(M:OTF,2,4) > '0000'
      STORE T TO U:OTF
    ELSE
      USE N OR O AND THEN TIME (A99999)
  ENDIF
ENDDO
@ 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 T TO U:GOV
  ELSE
    USE EITHER Y OR N
  ENDIF
ENDDO
@ 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
    USE OUT OF RANGE
  ELSE
    STORE T 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
    USE OUT OF RANGE
  ELSE
    STORE T TO U:QTYINS
  ENDIF
ENDDO
@ 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
    USE IN STOCK & OUT OF RANGE
  ELSE
    STORE T TO U:QTYSTK
  ENDIF

96
ENDO U:<QTSTK>
@ 23.30 SAY ' 
RELEASE U:QTSTK
@ 9.35 SAY ' GET U:TYPE PICTURE 'XXXXXXXX'
@ 10.35 SAY ' GET U:SERNO PICTURE 'XXXXXX'
@ 11.35 SAY ' GET U:HASSY PICTURE 'XXXXXXXXXX'
@ 12.35 SAY ' GET U:SASSY PICTURE 'XXXXXXXXXXXX'
READ
STORE U:TYPE+U:SERNO+U:HASSY+U:SASSY TO M:DITEM
@ 13.35 SAY ' GET M:CCOST PICTURE '9999999999.99'
READ
@ 14.35 SAY ' GET M:WUC PICTURE 'XXXXXXXX'
READ
@ 15.35 SAY ' GET M:DEFV PICTURE 'A'
STORE T TO U:DEFV
DO WHILE U:DEFV
@ 12.35 SAY ' GET M:DEFV PICTURE 'A'
READ
IF M:DEFV = 'N'.OR. M:DEFV = 'Q';
.OR. M:DEFV = 'Y':
.OR. M:DEFV = 'I':
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 = 'Y':
.OR. M:DEFR = 'S';
.OR. M:DEFR = 'U':
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'
END IF
END DO
3 23,30 SAY '
RELEASE U:END
@ 20,19 SAY 'COST CODE' GET M:COSTC PICTURE 'A'
@ 20,30 SAY 'ACTION CODE' GET M:ACTTXN PICTURE 'AAA'
@ 24,35 SAY 'GET M:RETNC PICTURE '9'
READ
STOR T TO U:END
DO WHILE U:END
  STORE ' ' TO U:REPLY
  23,10 SAY 'CHOOSE 1- CONTINUE, 2- CHANGE'
  23,35 SAY '. GET U:REPLY PICTURE '9'
  READ
  IF U:REPLY <> '1' .AND. U:REPLY <> '2'
    ELSE
      STORE F TO U:END
    ENDIF
  END DO
  23,05 SAY 'I
  IF U:REPLY = '2'
    STORE T TO U:PAGE2
    22,10 SAY '
    ELSE
      STORE F TO U:CGNT2
      STORE T TO U:CONT3
      STORE F TO U:PAGE2
  ENDIF
END DO
  U:PAGE2>
ERASE
END DO <U:CONT2>

***** START OF THE THIRD SCREEN FOR THE UPDATE PROGRAM
ERASE
DO WHILE U:CONT3

***** DISPLAY OF CASE DATA FROM OPEN2 DETAILS & REPLY
2 1,2 SAY 'NSN:
  1,17 SAY $M:NSN,5,8$<DEL>'+$$(M:NSN,5,2)$<DEL>+'$$(M:NSN,10,4)$
  1,55 SAY 'CASE NUMBERS:
  1,48 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

STOR ' ' TO U:REPLY
  22,10 SAY 'ENTER <N> TO SKIP & UPDATE RECORD'
  22,30 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
ENDDF
@ 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
    @ 21, 10 SAY 'GET M:DETAILS
    READ M:DETAILS
    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
    ENDDF
ENDDO <U:DISCODE>
@ 23, 30 SAY
RELEASE U:DISCODE
@ 14, 10 SAY 'GET M:REPLY
REAL STORE T TO U:END
DO WHILE U:END
STORE 'I' TO U:REPLY
@ 21, 10 SAY 'CHECK PREVIOUS';
+ ' ENTRIES ****** CHECK PREVIOUS';
@ 22, 10 SAY '<CHOOSE> 1- CONTINUE';
@ 23, 35 SAY 'GET U:REPLY
READ U:REPLY
IF U:REPLY <> '1' AND U:REPLY <> '2'
    ELSE
    STORE F TO U:END
ENDDIF
ENDDC <U:END>
@ 23, 10 SAY
IF U:REPLY = '2'
    STORE T TO U:PAGE3
    @ 24, 10 SAY
    ELSE
    STORE F TO U:PAGE3
    STORE F TO U:CONT3
ENDDF
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
ENDDIF
IF U:REPLY <> '3'
IF U:FILE = 'OPEN'
STORE '2C' TO M:TYPE
ELSE
STORE '4C' TO M:TYPE
ENDIF
DC C:xDBHND16
ENDIF
RELEASE U:CONTENT 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: CLOSEC **
** 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 CLOSE2 Databases and is marked for **
** deletion in the OPEN1 and OPEN2. **
** **
** Superordinate Modules: MENU1 **
** Subordinate Modules: XDBHNDLR **
** Author: J.G. BOYNION **
******************************************************************************
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 @: ' TC LC:CLDATE
STORE @: ' TO CL:VC
STORE @: ' TO CL:CF

@ 10.25 SAY '***** CLOSE CASE *****'
STORE T TO CL: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:DATE

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)

@ 23.30 SAY 'DATE OUT OF RANGE'
ELSE
STORE F TO CL:DATET
ENDIF

ENDDO <CL:DATET>

@ 23.30 SAY '
RELEASE CL:DATET,CL:DATE,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 CL: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 CL:AGAIN <> '1'
  RELEASE ALL EXCEPT C:*
RETURN
ENDIF
ELSE
IF .NOT. *
  STORE F TO CL:REPLY

102
STORE T TO CL:FILLED

ENDIF

IF *:
STORE ' ' TO CL:AGAIN
& 20,22 SAY 'That Case Already';
& 22,22 SAY 'Closed';
& 23,33 SAY 'Return To Menu'
READ
IF CL:AGAIN<>'1'
RELEASE ALL EXCEPT C:
RETURN
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 'IE' TC M:TYPE
DO C:XBHNDLR

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:CLOSE
STORE $(M:DATES,21,5) TO M:SCRDATE
STORE $(M:DATES,26,5) TO M:RIMDATE
STORE $(M:DATES,31,5) TO M:REOPEN
STORE $(M:DATES,36,5) TO M:CLOSE
STORE $(M:DATES,41,5) TO M:REOPEN

IF M:RDATE = ' ' OR M:OPEN = ' ' OR M:RIMDATE = ' '
& 20,20 SAY 'Key Date/Dates Are Missing.';
& 21,20 SAY 'Case may Not';
& 23,20 SAY 'Closed Until Update Is ';
& 23,20 SAY 'Accomplished';
@ 23,20 SAY 'Press Any Key To Continue';
WAIT
ERASE
& 10,25 SAY '***** Please Standby *****'
STORE '1G' TO M:TYPE
DO C:XBHNDLR
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:VLCT PICTURE 'A'
READ
IF CL:VLCT = ''

103
& 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 NOT BE BLANK'
ELSE
STORE F TO CL:CRT
ENDIF
,DDO <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 LC:CLDATE FROM MMDDYY TO JULIAN FORM
STCBE VAL($ (LC:CLDATE,1,2)) TO V:MM
STORR VAL($ (LC:CLDATE,3,2)) TO V:DD
STCBE VAL($ (LC:CLDATE,5,2)) TO V:YY
DO C:OJULIAN
STCBE V:JULDATE TO M:CLOSE
RELEASE ALL LIKE V:*  
STCBE M:CASE TO M:KEY
***** PUT CLOSING DATE INTO PROPER FORMAT FOR STORAGE
STCBE $(M:DATES,41,5) TO M:REOPEN  
STORR $(M:DATES,1,35) + M:CLOSE +;  
M:REOPEN TO CL:DATES
STCBE CL:DATES TO M:DATES
STCBE CL:VLC TO M:VLC
STCBE CL:CR TO M:CR
STCBE 'IC' TO M:TYPE
STCBE M:REC1 TO T:REC1
DO C:XDBHNDLR
***** CREATE RECORD IN CLOSE1
STORE '3F' TO M:TYPE

104
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:ONS!!
GOTC T:REC1
DELETE
STORE '23' TO M:TYPE
STORE T:CASE TO M:KEY
DO C:XDBHNDLR
STORE # TO T:REC2
STORE '4P' 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
END IF
@ 20,22 SAY ' ;
@ 22,22 SAY ' ;
ERASE
ENDIF <CL:FILLED>
ENDDC <CL:MOEE>
RELEASE ALL LIKE CL:*
RELEASE ALL LIKE M:*
RELEASE ALL LIKE T:*
STORE T TO CL:CLOSE
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:UTC, M:UT, M:QTYDEF, M:UPRC, M:EPSC, **
M:ORG, M:DOC, M:DOCNO, M:DATES, M:REPCON, **
M:SM, M:O9Q, M:DEF, M:VLC, M:ACTST, **
M:SCRTY, M:REC1, M:QTYINS, M:QTYREC, **
M:QTYSTK, M:DEFVR, M:DEFR, M:ITEM, M:OVER, **
M:QTYFP, M:GOV, M:ITEM, M:CCOST, M:WNTY, **
M:UUC, M:DIS, M:DETAILS, M:REPLY, **
M:ACTKN, 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:UTC, M:UT, M:QTYDEF, M:UPRC, M:EPSC, **
M:ORG, M:DOC, M:DOCNO, M:DATES, **
M:REPCON, M:PSCHA, M:TIME, M:WHO, M:NUM, **
M:ACTST, M:SCRTY, M:REC1, M:QTYINS, **
M:QTYSTK, M:DEFVR, M:DEFR, M:ITEM, M:OVER, **
M:QTYFP, M:GOV, M:ITEM, M:CCOST, M:WNTY, **
M:UUC, M:DIS, M:DETAILS, M:REPLY, **
M:ACTKN, 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/LOCK, 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 **
**
** AUTHOR: 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/LOCK **
** WRITE NEW RECORD, WRITE UPDATE UNLOCK, RECORD UNLOCK...
DO CASE

USE OPEN1 DATA BASE FILE

CASE $(M:TYPE,2,1) = '1'
   IF $(M:TYPE,2,1) = 'A'
   OR. $(M:TYPE,2,1) = 'E'

CASE $(M:TYPE,2,1) = 'A'
   STORE 'USE D:OPEN1 INDEX D:ONSN' TO H:USEFILE
   ELSE

CASE $(M:TYPE,2,1) = 'E'
   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:REC1
   STORE CASE TO M:CASE
   STORE COG TO M:CCG
   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 DEP TO M:DEP
   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'

107
***** READ RECORD SPECIFIED BY M:REC1 (RECORD NUMBER)
***** IF RECORD NOT FOUND RETURN M:TYPE = 9 OTHERWISE
***** RETURN THE RECORD ELEMENTS

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

***** IF M:TYPE = 1 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:CCG
STORE ASN TO M:ASN
STORE CAT TO M:CAT
STORE NCNEN TO M:NSN
STORE NCMEN TO M:NOMEN
STORE UIC TO M:UIC
STORE U1 TO M:U1
STORE CYDEP TO M:QTYDEF
STORE UPRC TO M:UPRC
STORE EPRC TO M:EPRC
STORE CCG TO M:ORG
STORE DOC TO M:DOC
STORE DOCNO TO M:DOCNO
STORE DATES TO M:DATES
STORE REPCON TO M:REPCON
STORE FSCK TO M:FSCK
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:SQ9Q
STORE DEP TO M:DEP
STORE VTC TO M:VTC
STORE ACTPT TO M:ACTPT
STORE SCRQTY TO M:SCRQTY
STORE 'Q' 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 TO H:FAIL
DO WHILE H:FAIL
STORE 0 TO H:CNTR
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 LOOP A SHORT PERIOD OF TIME DISPLAY THE FACT THAT THE FILE IS CURRENTLY IN USE

IF H:LOCPCNTF = 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
  STORE H:DELAY + 1 TO H:DELAY
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 OPEN1 = C:WHO
STORE ' ' TO M:TIME
GOTO M:BEC1
REPL CASE WITH !(M:CASE), Cog WITH !(M:COG), NSN WITH:
  !(M:NSN), CAT WITH !(M:CAT), NOMEN WITH !(M:NOMEN), UIC :
  !(M:UIC), UI WITH !(M:UI), QTYDEF WITH M:QTYDEF, UERC :
  WITH M:UERC, EPRC WITH M:EPRC, ORG WITH !(M:ORG), DOC WITH :
  !(M:DOC)
REPL LOCNO WITH !(M:LOCNO), DATES WITH !(M:DATES), REPCON :
  WITH !(M:REPCON), FSCM WITH !(M:FSCM), TIME WITH !(M:TIME) :
  WITH !(M:SRC), SM WITH !(M:SM), 09Q WITH !(M:09Q)
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) = 'D'
        STORE 'USE D:OPEN1 INDEX D:CASE1, ';
        + 'D:OCASE1' TO H:USEFILE

***** IF TYPE C THEN PERFORM READ/LOCK WITH CASE ACCESS KEY

    ELSE
        STORE 'USE D:OPEN1 INDEX D:CASE1, ';
        + 'D:ONSN' TO H:USEFILE
    ENDIF

USE D:FILESTAT

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

    IF OPEN1 = C:WHO
        FIND &M:KEY
    ELSE
        CEECK TO SEE IF THE USER HAS THE FILE LOCKED FOR WAITING

    ***** 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: HOUR
        STORE STR(PEEK(61455), 2) TO ;
        H: MIN
        STORE STR(PEEK(61454), 2) TC ;
        H: SEC
        IF $(H: HOUR, 1, 1) =
            STORE 0 +$(H: HOUR, 2, 1) ;
            TO H: HOUR
        ENDF

110
IF $(H:MIN,1,1)$ =
  STORE $(H:MIN,2,1)$ TO :
  H:MIN
ENDIF
IF $(H:SEC,1,1)$ =
  STORE $(H:SEC,2,1)$ TO :
  H:SEC
ENDIF
STORE C:JULIAN+H:HOURE +H:MIN+;
  H:SEC TO M:TIME
REPL TIME WITH M:TIME
STORE # TO M:RECP
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 FSCN TO M:FSCN
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 OSG TO M:OSG
STORE DEF TO M:DEF
STORE VLC TO M:VLC
STORE ACTPT TO M:ACTPT
STORE SCRQTY TO M:SCRQTY
STORE 'O' TO M:TYPE
USE D:FILESTAT
REPL OPEN1 WITH *
USE RELASE ALL LIKE H:*
RETURN
ENDIF
ELSE
ENDIF
***** 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 NUMBER HAS BEEN ASSIGNED BECAUSE OF A
***** PREVIOUS CASE, ASSIGN A NEW CASE NUMBER
IF M:CASE = '

111
**** 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 + 1;
STR(H:SERIAL, 5, 0) TO M:CASE
ELSE
IF H:SERIAL > 999
STORE H:YR + '0' + 1;
STR(H:SERIAL, 4, 0) TO M:CASE
ELSE
IF H:SERIAL > 99
STORE H:YR + '00' + 1;
STR(H:SERIAL, 3, 0) TO M:CASE
ELSE
STORE H:YR + '000' + 1;
STR(H:SERIAL, 2, 0) TO M:CASE
ELSE
STORE H:YR + '0000' + 1;
STR(H:SERIAL, 1, 0) TO M:CASE
ENDIF
ENDIF
ENDIF
ENDIF
ENDIF
ENDIF
ENDIF
ENDIF
ENDIF
ENDIF
ENDIF
ENDIF
& 23, 26 SAY 'CREATING NEW RECORD';
+' - OPEN!

**** CREATE NEW RECORD AND FILL WITH DATA

APPEND BLANK
@ 23, 26 SAY 'O' TO M:TYPE
USE D:FILESTAT
REPL OPEN1 WITH 'O'
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

112
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:* 
RETURN
ENDIF
ENDIF
ENDIF
ENDDO

***** 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:* 
  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 DEFP TO M:DEFP
  STORE DEFR TO M:DEFR
  STORE ITEM TO M:ITEM
  STORE OVER TO M:OVER
  STORE OTF IC 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 ACTKN TO M:ACTKN
  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
  113
STORE MFG TO M:MFG
STORE LOT TO M:LCT
RELEASE ALL LIKE H:* RET
ENDIF
ELSE

***** IF TYPE H OR I ACCESS BY RECORD NUMBER (M:REC1)

IF $(M:TYPE,2,1) = 'H' .OR. $(M:TYPE,2,1) = 'I'

USE D:OPEN2
GOT M:REC1
IF # <> M:REC1
STORE 'S' TO M:TYPE
RELEASE ALL LIKE H:* 
RETURN
ELSE

***** IF TYPE I, SKIF TO NEXT RECORD AND READ DATA

IF $(M:TYPE,2,1) = 'I'

SKIF
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 DEPVR TO M:DEPVR
STORE DEFRR TO M:DEFRR
STORE ITEM TO M:ITEM
STORE OVER TO M:OVER
STORE CTF TO M:OTF
STORE GCV TO M:GCV
STORE TIME TO M:TIME
STORE WHO TO M:WHO
STORE ITEM TO M:ITEM
STORE CCOST TO M:CCOST
STORE WNTY TO M:WNTY
STORE WOC TO M:WOC
STORE DIS TO M:DIS
STORE DETAILS TO M:DETAILS
STORE RPR TO M:RPR
STORE ATTKN TO M:ATTKN
STORE CCSTC TO M:CCSTC
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 ICT TO M:LOT
STORE '0' TO M:TYPE
RELEASE ALL LIKE H:*
RETURN
ENDIF
ELSE

***** THE FOLLOWING SECTION REQUIRES THAT THE DATA BASE
***** BE LOCKED TO ENSURE ONLY A SINGLE UPDATE IS
***** PERFORMED AT A TIME

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

***** LOOP WHILE OPEN2 IS LOCKED BY ANOTHER USER
DO WHILE OPEN2<>
  STORE H:CNTR-1 TO H:CNTR
  IF H:CNTR=0
    STORE 2 TO H:CNTR
    STORE H:LOOPCNTR+1 TO H:LOOKCNTR
  ENDF
  IF H:LOOKCNTR=2
    @ 23,16 SAY 'OPEN CASE FILE CURRENTLY IN':
      'USE - PLEASE STANDBY'
  ENDF

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

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
ENDDO

@ 23,16 SAY '+

@ 23,16 SAY '+ FILE LOCKED'+

***** WRITE LOCK TO FILESTAT

REPL OPEN2 WITH C:WHO

***** IF TYPE C PERFCEM WRITE/UNLOCK

IF S(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 '1' TO M:TIME
  USE D:OPEN2 INDEX D:OCASE2

***** WRITE UPDATE INFORMATION TO THE FILE

GCTO M:REC1
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) ,
CTP WITH ! (M:CTP) GOV WITH ! (M:GOV) TIME WITH ! (M:TIME)
REPL WHO WITH ! (M:WHO) DITEM WITH ! (M:DITEM) COSTC WITH :
M:CCOST WNTY WITH ! (M:WNTY) WUC WITH ! (M:WUC) DIS WITH :
! (M:DIS) DETAILS WITH ! (M:DETAILS) REPLY WITH ! (M:REPLY) ,
ACTKN WITH ! (M:ACTKN) COSTC WITH ! (M:ACTCOST) CR
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
    ENDIF 

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

***** IF DESIRED RECORD FOUND VERIFY THAT RECORD IS NOT CURRENTLY IN USE - IF NOT FOUND RETURN TYPE = 9
IF OPEN2 = C:WHO
    USE D:OPEN2 INDEX D:OCASE2 
    FIND &M:KEY
    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) TC ; 
H: HOUR
STORE STR(PEEK(61455), 2) TO ; 
H: MIN
STORE STR(PEEK(61454), 2) TC ; 
H: SEC
IF $(H: HOUR, 1, 1) = 
    STORE 0 *$(H: HOUR, 2, 1) ; 
    TO H: HOUR
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: HOUR +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 DEFV TO M:DEFV
STORE DEF3 TO M:DEF3
STORE ITEM TO M:ITEM
STORE OVER TO M:OVER
STORE OTP TO M:OTP
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 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 OPENZ WITH ' '
USE
RELEASE ALL LIKE H:*
RETURN
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 OPENZ = 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
DEFV WITH !(M:DEFV)
DEF3 WITH !(M:DEF3)
ITEM WITH !(M:ITEM)
OVER WITH !(M:OVER)
OTP WITH !(M:OTP)
GOV WITH !(M:GOV)
TIME WITH !(M:TIME)
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

117
****** UNLOCK DATA BASE FOR OTHERS

USE D:FILESTAT
RELEASE OPEN2 WITH
USE
2 23, 25 SAY ":
RELEASE ALL LIKE H:*
RETURN
ENDIF
ELSE
****** IF TYPE G PERFORM UNLOCK (NO UPDATE)

IF $(M:TYPE, 2, 1) = 'G'
USE D:FILESTAT
ENDIF
USE D:OPEN2
GOTO M:REC1
ENDIF

****** UNLOCK DATA BASE FOR OTHERS

USE D:FILESTAT
RELEASE OPEN2 WITH
USE
RELEASE ALL LIKE H:*
RETURN
ENDIF
ENDIF
ENDIF
ENDIF
ENDIF
ENDIF
ENDIF
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
****** EASIC COMMENTS ARE THE SAME AS ABOVE

****** TYPE 3 USES CLSE1

CASE $((M:TYPE, 1, 1) = '3'
IF $(M:TYPE, 2, 1) = 'A', OR. $(M:TYPE, 2, 1) = 'B'
STORE 'USE D:CLOSE1 INDEX D:CNSN' TO H:USEFILE
ELSE
STORE 'USE D:CLOSE1 INDEX D:CCASE1' TO H:USEFILE
ENDIF
ENDIF
ENDIF
ENDIF
ENDIF
ENDIF
ENDIF
ENDIF
ENDIF
ENDIF
ENDIF
ENDIF
ENDIF
ENDIF
ENDIF
ENDIF
ENDIF
ENDIF
ENDIF
ENDIF
ENDIF
ENDIF
ENDIF
ENDIF
ENDIF
ENDIF
ENDIF
ENDIF
ENDIF
ENDIF
ENDIF
ENDIF
ENDIF
ENDIF
ENDIF
ENDIF
ENDIF
ENDIF
ENDIF
ENDIF
ENDIF
ENDIF
ENDIF
ENDIF
ENDIF
ENDIF
ENDIF
ENDIF
ENDIF
ENDIF
ENDIF
ENDIF
ENDIF
ENDIF
ENDIF
ENDIF
ENDIF
ENDIF
ENDIF
ENDIF
ENDIF
ENDIF
ENDIF
ENDIF
ENDIF
ENDIF
ENDIF
ENDIF
ENDIF
ENDIF
ENDIF
ENDIF
ENDIF
ENDIF
ENDIF
ENDIF
ENDIF
ENDIF
ENDIF
ENDIF
ENDIF
ENDIF
ENDIF
ENDIF
ENDIF
ENDIF
ENDIF
ENDIF
ENDIF
ENDIF
ENDIF
ENDIF
ENDIF
ENDIF
ENDIF
ENDIF
ENDIF
ENDIF
ENDIF
ENDIF
ENDIF
ENDIF
ENDIF
ENDIF
ENDIF
ENDIF
ENDIF
ENDIF
ENDIF
ENDIF
ENDIF
ENDIF
ENDIF
ENDIF
ENDIF
ENDIF
ENDIF
ENDIF
ENDIF
ENDIF
ENDIF
ENDIF
ENDIF
ENDIF
ENDIF
ENDIF
ENDIF
ENDIF
ENDIF
ENDIF
ENDIF
ENDIF
ENDIF
ENDIF
ENDIF
ENDIF
ENDIF
ENDIF
ENDIF
ENDIF
ENDIF
ENDIF
ENDIF
ENDIF
ENDIF
ENDIF
ENDIF
ENDIF
ENDIF
ENDIF
ENDIF
ENDIF
ENDIF
ENDIF
ENDIF
ENDIF
ENDIF
ENDIF
ENDIF
ENDIF
ENDIF
ENDIF
ENDIF
ENDIF
ENDIF
ENDIF
ENDIF
ENDIF
ENDIF
ENDIF
ENDIF
ENDIF
ENDIF
ENDIF
ENDIF
ENDIF
ENDIF
ENDIF
ENDIF
ENDIF
ENDIF
ENDIF
ENDIF
ENDIF
ENDIF
ENDIF
ENDIF
ENDIF
ENDIF
ENDIF
ENDIF
ENDIF
ENDIF
ENDIF
ENDIF
ENDIF
ENDIF
ENDIF
ENDIF
ENDIF
ENDIF
ENDIF
ENDIF
ENDIF
ENDIF
ENDIF
ENDIF
ENDIF
ENDIF
ENDIF
ENDIF
ENDIF
ENDIF
ENDIF
ENDIF
ENDIF
ENDIF
ENDIF
ENDIF
ENDIF
ENDIF
ENDIF
ENDIF
ENDIF
ENDIF
ENDIF
ENDIF
ENDIF
ENDIF
ENDIF
ENDIF
ENDIF
ENDIF
ENDIF
ENDIF
ENDIF
ENDIF
ENDIF
ENDIF
ENDIF
ENDIF
ENDIF
ENDIF
ENDIF
ENDIF
ENDIF
ENDIF
ENDIF
ENDIF
ENDIF
ENDIF
ENDIF
ENDIF
ENDIF
ENDIF
ENDIF
ENDIF
ENDIF
ENDIF
ENDIF
ENDIF
ENDIF
ENDIF
ENDIF
ENDIF
ENDIF
ENDIF
ENDIF
ENDIF
ENDIF
ENDIF
ENDIF
ENDIF
ENDIF
ENDIF
ENDIF
ENDIF
ENDIF
ENDIF
ENDIF
ENDIF
ENDIF
ENDIF
ENDIF
ENDIF
ENDIF
ENDIF
ENDIF
ENDIF
ENDIF
ENDIF
ENDIF
ENDIF
ENDIF
ENDIF
ENDIF
ENDIF
ENDIF
ENDIF
ENDIF
ENDIF
ENDIF
ENDIF
ENDIF
ENDIF
ENDIF
ENDIF
ENDIF
ENDIF
ENDIF
ENDIF
ENDIF
ENDIF
ENDIF
ENDIF
ENDIF
ENDIF
ENDIF
ENDIF
ENDIF
ENDIF
ENDIF
ENDIF
ENDIF
ENDIF
ENDIF
ENDIF
ENDIF
ENDIF
ENDIF
ENDIF
ENDIF
ENDIF
ENDIF
ENDIF
ENDIF
ENDIF
ENDIF
ENDIF
ENDIF
ENDIF
ENDIF
ENDIF
ENDIF
ENDIF
ENDIF
ENDIF
ENDIF
ENDIF
ENDIF
ENDIF
ENDIF
ENDIF
ENDIF
ENDIF
ENDIF
ENDIF
ENDIF
ENDIF
ENDIF
ENDIF
ENDIF
ENDIF
ENDIF
ENDIF
ENDIF
ENDIF
ENDIF
ENDIF
ENDIF
ENDIF
ENDIF
ENDIF
ENDIF
ENDIF
ENDIF
ENDIF
ENDIF
ENDIF
ENDIF
ENDIF
ENDIF
ENDIF
ENDIF
ENDIF
ENDIF
ENDIF
ENDIF
ENDIF
ENDIF
ENDIF
ENDIF
ENDIF
ENDIF
ENDIF
ENDIF
ENDIF
ENDIF
ENDIF
ENDIF
ENDIF
ENDIF
ENDIF
ENDIF
ENDIF
ENDIF
ENDIF
ENDIF
ENDIF
ENDIF
ENDIF
ENDIF
ENDIF
ENDIF
ENDIF
ENDIF
ENDIF
ENDIF
ENDIF
ENDIF
ENDIF
ENDIF
ENDIF
ENDIF
ENDIF
ENDIF
ENDIF
ENDIF
ENDIF
ENDIF
ENDIF
ENDIF
ENDIF
ENDIF
ENDIF
ENDIF
ENDIF
ENDIF
ENDIF
ENDIF
ENDIF
ENDIF
ENDIF
ENDIF
ENDIF
ENDIF
ENDIF
ENDIF
ENDIF
ENDIF
ENDIF
ENDIF
ENDIF
ENDIF
ENDIF
ENDIF
ENDIF
ENDIF
ENDIF
ENDIF
ENDIF
ENDIF
ENDIF
ENDIF
ENDIF
ENDIF
ENDIF
ENDIF
ENDIF
ENDIF
ENDIF
ENDIF
ENDIF
ENDIF
ENDIF
ENDIF
ENDIF
ENDIF
ENDIF
ENDIF
ENDIF
ENDIF
ENDIF
ENDIF
ENDIF
ENDIF
ENDIF
ENDIF
ENDIF
ENDIF
ENDIF
ENDIF
ENDIF
ENDIF
ENDIF
ENDIF
ENDIF
ENDIF
ENDIF
ENDIF
ENDIF
ENDIF
ENDIF
ENDIF
ENDIF
ENDIF
ENDIF
ENDIF
ENDIF
ENDIF
ENDIF
ENDIF
ENDIF
ENDIF
ENDIF
ENDIF
ENDIF
ENDIF
ENDIF
ENDIF
ENDIF
ENDIF
ENDIF
ENDIF
ENDIF
ENDIF
ENDIF
ENDIF
ENDIF
ENDIF
ENDIF
ENDIF
ENDIF
ENDIF
ENDIF
ENDIF
ENDIF
ENDIF
ENDIF
ENDIF
ENDIF
ENDIF
ENDIF
ENDIF
ENDIF
ENDIF
ENDIF
ENDIF
ENDIF
ENDIF
ENDIF
ENDIF
ENDIF
ENDIF
ENDIF
ENDIF
ENDIF
ENDIF
ENDIF
ENDIF
ENDIF
ENDIF
ENDIF
ENDIF
ENDIF
ENDIF
ENDIF
ENDIF
ENDIF
ENDIF
ENDIF
ENDIF
ENDIF
ENDIF
ENDIF
ENDIF
ENDIF
ENDIF
ENDIF
ENDIF
ENDIF
ENDIF
ENDIF
ENDIF
ENDIF
ENDIF
ENDIF
ENDIF
ENDIF
ENDIF
ENDIF
ENDIF
ENDIF
ENDIF
ENDIF
ENDIF
ENDIF
ENDIF
ENDIF
ENDIF
ENDIF
ENDIF
ENDIF
ENDIF
ENDIF
ENDIF
ENDIF
ENDIF
ENDIF
ENDIF
ENDIF
ENDIF
ENDIF
ENDIF
ENDIF
ENDIF
ENDIF
ENDIF
ENDIF
ENDIF
ENDIF
ENDIF
ENDIF
ENDIF
ENDIF
ENDIF
ENDIF
ENDIF
ENDIF
ENDIF
ENDIF
ENDIF
ENDIF
ENDIF
ENDIF
ENDIF
ENDIF
ENDIF
ENDIF
ENDIF
ENDIF
ENDIF
ENDIF
ENDIF
ENDIF
ENDIF
ENDIF
ENDIF
ENDIF
ENDIF
ENDIF
ENDIF
ENDIF
ENDIF
ENDIF
ENDIF
ENDIF
ENDIF
ENDIF
ENDIF
ENDIF
ENDIF
ENDIF
ENDIF
ENDIF
ENDIF
ENDIF
ENDIF
ENDIF
ENDIF
ENDIF
ENDIF
ENDIF
ENDIF
ENDIF
ENDIF
ENDIF
ENDIF
ENDIF
ENDIF
ENDIF
ENDIF
ENDIF
ENDIF
ENDIF
ENDIF
ENDIF
ENDIF
ENDIF
ENDIF
ENDIF
ENDIF
ENDIF
ENDIF
ENDIF
ENDIF
ENDIF
ENDIF
ENDIF
ENDIF
ENDIF
ENDIF
ENDIF
ENDIF
ENDIF
ENDIF
ENDIF
ENDIF
ENDIF
ENDIF
ENDIF
ENDIF
ENDIF
ENDIF
ENDIF
ENDIF
ENDIF
ENDIF
ENDIF
ENDIF
ENDIF
ENDIF
ENDIF
ENDIF
ENDIF
ENDIF
ENDIF
ENDIF
ENDIF
ENDIF
ENDIF
ENDIF
ENDIF
ENDIF
ENDIF
ENDIF
ENDIF
ENDIF
ENDIF
ENDIF
ENDIF
ENDIF
ENDIF
ENDIF
ENDIF
ENDIF
ENDIF
ENDIF
ENDIF
ENDIF
ENDIF
ENDIF
ENDIF
ENDIF
ENDIF
ENDIF
ENDIF
ENDIF
ENDIF
ENDIF
ENDIF
ENDIF
ENDIF
ENDIF
ENDIF
ENDIF
ENDIF
ENDIF
ENDIF
ENDIF
ENDIF
ENDIF
ENDIF
ENDIF
ENDIF
ENDIF
ENDIF
ENDIF
ENDIF
ENDIF
ENDIF
ENDIF
ENDIF
ENDIF
ENDIF
ENDIF
ENDIF
ENDIF
ENDIF
ENDIF
ENDIF
ENDIF
ENDIF
ENDIF
ENDIF
ENDIF
ENDIF
ENDIF
ENDIF
ENDIF
ENDIF
ENDIF
ENDIF
ENDIF
ENDIF
ENDIF
ENDIF
ENDIF
ENDIF
ENDIF
ENDIF
ENDIF
ENDIF
ENDIF
ENDIF
ENDIF
ENDIF
ENDIF
ENDIF
ENDIF
ENDIF
ENDIF
ENDIF
ENDIF
ENDIF
ENDIF
ENDIF
ENDIF
ENDIF
ENDIF
ENDIF
ENDIF
ENDIF
ENDIF
ENDIF
ENDIF
ENDIF
ENDIF
ENDIF
ENDIF
ENDIF
ENDIF
ENDIF
ENDIF
ENDIF
ENDIF
ENDIF
ENDIF
ENDIF
ENDIF
ENDIF
ENDIF
ENDIF
ENDIF
ENDIF
ENDIF
ENDIF
ENDIF
ENDIF
ENDIF
ENDIF
ENDIF
ENDIF
ENDIF
ENDIF
ENDIF
ENDIF
ENDIF
ENDIF
ENDIF
ENDIF
ENDIF
ENDIF
ENDIF
ENDIF
ENDIF
ENDIF
ENDIF
ENDIF
ENDIF
ENDIF
ENDIF
ENDIF
ENDIF
ENDIF
ENDIF
ENDIF
ENDIF
ENDIF
ENDIF
ENDIF
ENDIF
ENDIF
ENDIF
ENDIF
ENDIF
ENDIF
ENDIF
ENDIF
ENDIF
ENDIF
ENDIF
ENDIF
ENDIF
ENDIF
ENDIF
ENDIF
ENDIF
ENDIF
ENDIF
ENDIF
ENDIF
ENDIF
ENDIF
ENDIF
ENDIF
ENDIF
ENDIF
ENDIF
ENDIF
ENDIF
ENDIF
ENDIF
ENDIF
ENDIF
ENDIF
ENDIF
ENDIF
ENDIF
ENDIF
ENDIF
ENDIF
ENDIF
ENDIF
ENDIF
ENDIF
ENDIF
ENDIF
ENDIF
ENDIF
ENDIF
ENDIF
ENDIF
ENDIF
ENDIF
ENDIF
ENDIF
ENDIF
ENDIF
ENDIF
ENDIF
ENDIF
ENDIF
ENDIF
ENDIF
ENDIF
ENDIF
ENDIF
ENDIF
ENDIF
ENDIF
ENDIF
ENDIF
ENDIF
ENDIF
ENDIF
ENDIF
ENDIF
ENDIF
ENDIF
ENDIF
ENDIF
ENDIF
ENDIF
ENDIF
ENDIF
ENDIF
ENDIF
ENDIF
ENDIF
ENDIF
ENDIF
ENDIF
ENDIF
ENDIF
ENDIF
ENDER

118
STORE UIC TO M:UIC
STORE UI TO M:UI
STORE OTYPEF TO M:OTYPEF
STORE UPRC TO M:UPRC
STORE EPRC TO M:EPRC
STORE ORG TO M:ORG
STORE DOC TO M:DOC
STORE DOCNC TO M:DOCNC
STORE DATES TO M:DATES
STORE REPCN TO M:REPCN
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 HIC TO M:HIC
STORE DEF TO M:DEF
STORE VLC TO M:VLC
STORE ACTIT TO M:ACTIT
STORE SCRCTY TO M:SCRCTY
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 # TO M:TYPE
RELEASE ALL LIKE H:*
RETURN
ELSE IF $(M:TYPE,2,1) = 'I'
SKIF
ENDIF
STORE # TO M:REC1
STORE CASE TO M:CASE
STORE CCG TO M:CCG
STORE NSN TO M:NSN
STORE CAT TO M:CAT
STORE NCN TO M:NCMEN
STORE UI TO M:UI
STORE CTYDEF TO M:CTYDEF
STORE UPRC TO M:UPRC
STORE EPRC TO M:EPRC
STORE CBG TO M:CBG
STORE DCCNO TO M:CDOC
STORE DATE TO M:DATES
STORE REPCN TO M:REPCN
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:CSQ
STORE DEF TO M:DEF
STORE VLC TO M:VLC
STORE ACTIT TO M:ACTIT
STORE SCRQTY TO M:SCRQTY
STORE '0' TO M:TYPE
RELEASE ALL LIKE H:*
RETURN
119
END IF
ELSE
STORE T TO H:FAIL
DO WHILE H:FAIL
STORE 0 TO H:LOOPECNTR
STORE 2 TO H:CNTR
DO WHILE H:CLOSE1
STORE H:CNTR-1 TO H:CNTR
IF H:CNTR=0
STORE 2 TO H:CNTR
STORE H:LOOPECNTR+1 TO H:LOOPECNTR
ENDIF
IF H:LOOPECNTR=2
@ 23,16 SAY 'CLOSE CASE FILE CURRENTLY IN';
' USE - PLEASE CLEAR!' ENDIF
USE D:FILESTAT
STORE 1 TO H:DELAY
DO WHILE H:DELAY < 5
STORE H:DELAY + 1 TO H:DELAY
ENDDO
END DO
@ 23,16 SAY 'CLOSE1
FILE LOCKED BY ';
REPL 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:CNNSN, D:CASE1
GOTO M:RENC1
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, EERC WITH 'M:EERC, ORG WITH 'M:ORG, DOC WITH 'M:DOC
REPL LOCNO WITH 'M:LOCNO, DATES WITH 'M:DATES, REPCON ;
WITH 'M:REPCON, FSCH WITH 'M:FSCH, TIME WITH 'M:TIME',
WHO WITH 'M:WHO, NUM WITH 'M:NUM', OR WITH 'M:OR, SCR ;
WITH 'M:SCR', SM WITH 'M:SM', O9O WITH 'M:O9O, REPCON
REPL DEF WITH M:DEF, VLC WITH 'M:VLC, ACTPT WITH ;
'M:ACTPT), SCRQTY WITH M:SCRQTY
STORE 'O' TO M:TYPE
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:CNNSN, ';
'D:CASE1' TO H:USEFILE
ELSE
STORE 'USE D:CLOSE1 INDEX D:CASE1',
'D:CNNSN' TO H:USEFILE
ENDIF
USE D:FILESTAT
IF CLOSE1 = C:WHO
USE D:USEFILE
IF C:WHO
FIND &M:KEY
IF # = 0
STORE '9' TO M:TYPE
120
USE D:FILESTAT
REPLACE CLOSET WITH ' '
USE
RELEASE ALL LIKE H:*
USE
RETURN
ELSE
USE <>
STORE '1' TO M:TYPE
RELEASE ALL LIKE H:*
USE D:FILESTAT
REPLACE CLOSET WITH ' '
USE
RETURN
ELSE
STORE TO H:DU:MY
POKE 1440, 140, 245, 137, 141, 15, 195
SET CALL TO 61440
CALL H:DU:MY
STORE STR(PEEK(61456),2) TC ; H:HOUR
STORE STR(PEEK(61455),2) TC ; H:MIN
STORE STR(PEEK(61454),2) TC ; H:SEC
IF $(H:HOUR,1,1) =
STORE 0 +$ (H:HOUR,2,1) ; TO H:HOUR
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:HOUR +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 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 DEP TO M:DEP
STORE VLC TO M:VLC
STORE ACTPT TO M:ACTPT
STORE SCRQTY TO M:SCRQTY
121
STORE 'O' TO M:TYPE
USE D:FILESTAT
REPL CLOSE1 WITH ' ' USE
RELEASE ALL LIKE H:* RETURN

ENDIF
ENDIF
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, ;
L:CNSN APPEND BLANK REPL CASE WITH !(M:CASE), COG WITH !(M:COG), NSN WITH !
!(M:NSN), CAT WITH !(M:CAT), NOMEN WITH !(M:NOMEN), UI WITH !
!(M:UI), UI WITH !(M:UI), OTY WITH !(M:OTY), UPRC WITH !
M:UPRC, EPIC WITH !(M:EPIC), ORG WITH !(M:ORG), DOC WITH !(M:DOC)
REPL LOCNO WITH !(M:LOCNO), DATES WITH !(M:DATES), REPCON
WITH !(M:REPCON), FSC WITH !(M:FSC), 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 'O' TO M:TYPE
USE D:FILESTAT
REPL CLOSE1 WITH ' ' USE
@ 23,25 SAY ' ' RETURN
ENDIF
ELSE

***** TYPE 4 USES CLCSE2 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
122
FIND &M:KEY
IF * = 0
STORE '9' TO M:TYPE
RELEASE ALL LIKE H:* RETURN
ELSE
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 DEPFV TO M:DEPFV
STORE DEPR TO M:DEPR
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 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
END IF
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
END IF
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 DEPFV TO M:DEPFV
STORE DEPR TO M:DEPR
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 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
END IF
ENDIF
STORE AC@KxN TO M:AC@KxN
STORE CCOST TO M:CCOST
STORE STATUSC TO M:STATUSC
STORE CAUSEC TO M:CAUSEC
SIC@E 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:LOOPCNTR
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:LOOPCNTR+1 TO H:LOOPCNTR
ENDIF
IF H:LOOPCNTR=2
@ 23,16 SAY 'CLOSE CASE FILE CURRENTLY IN USE';
+ 'PLEASE STANDBY'
ENDIF
USE F:FILESTAT
STORE 1 TO H:DELAY
DO WHILE H:DELAY < 5
STORE H:DELAY + 1 TO H:DELAY
ENDEC
ENDDO
@ 23,16 SAY 'REPL CLOSE2 WITH C:WHO
USE IF $ (M:TYPE,2,1) = 'C'
USE D:FILESTAT
IF CLOSE2 = C:WHO
STORE 1 TO M:TIME
STORE D:CLOSE2 INDEX D:CASE2
GOTO M:RETC
REPL CASE WITH ! (M:CASE), QTYINS WITH M:QTYINS, QTYREC :
WITH M:QTYREC, QTYSTK WITH M:QTYSTK, DEFV WITH !(M:DEFV),
DEPR WITH !(M:DEPR), ITEM WITH !(M:ITEM), OVER WITH !
!(M:OVER), OTF WITH !(GTF), GOV WITH !(M:GOV), TIME WITH !
!(M:TENS)
REPL WHO WITH !(M:WHO), 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),
ACTKN WITH !(M:ACTKN), 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 !
STORE '0' TO M:TYPE
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

124
USE D:CLOSE2 INDEX D:CCASE2
FIND &M:KEY
IF # = 0
STORE '9' TO M:TYPE
USE D:FILESTAT
REPLACE CLOSE2 WITH ' *
RELEASE ALL LIKE H: *
RETURN
ELSE
IF TIME <> '
STORE '1' TO M:TYPE
RELEASE ALL LIKE H: *
USE D:FILESTAT
REPLACE CLOSE2 WITH ' *
RETURN
ELSE
STORE TO H:DUMMY
ECKE 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:HOUR
STORE STR(PEEK(61455), 2) TO H:MIN
STORE STR(PEEK(61454), 2) TO H:SEC
IF $(H:HOUR, 1, 1) =
STORE 0 + $(H:HOUR, 2, 1) TO H:HOUR
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:HOUR + 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 DEFPV TO M:DEFPV
STORE DEFPF TO M:DEFPF
STORE ITEM TO M:ITEM
STORE OVER TO M:OVER
STORE OTP TO M:OTP
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 ACTKN TO M:ACTKN
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 ' *
**** FCRTYPE 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
**IF CLOSE2 = C:WHO USE D:CLOSE2 INDEX D:CASE2 FIND &M:CASE IF & <> J
**USE D:CLOSE2 INDEX D:CASE2 APPEND BLANK
**STORE '0' TO M:TYPE
**USE D:FILESTAT
**REPL CLOSE2 WITH ' USE
**& 23,25 SAY ' + ' 
**RELEASE ALL LIKE H:* RETURN
**ENDIF
**ELSE
**IF $(M:TYPE, 2, 1) = 'G' USE D:FILESTAT
**IF CLOSE2 = C:WHO USE D:CLOSE2 GOTO M:REC IF TIME = M:TIME REPL TIME WITH ' 
**ENDIF USE D:FILESTAT REPL CLOSE2 WITH ' USE
**RELEASE ALL LIKE H:* RETURN
**ENDIF
**ENDIF
**ENDIF
**ENDIF
**ENDIF
**ENDIF
**ENDIF
**ENDIF
**ENDDO
**ENDIF
**ENDIF
**ENDCASE
**RETURN

***** END OF PROGRAM
VIII. SUPERVISOR MENU

**SUPERVISOR MENU**

**DATE: 11 JANUARY 1984**

**VERSION: 1.0**

**MODULE NAME: SUPMENU1**

**MODULE PURPOSE: PROVIDE MENU FOR SUPERVISOR TO ACCESS QDF 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 PROCESSING CAPABILITIES AVAILABLE. AFTER ONE IS CHOSEN, THE MODULE THEN CALLS THE DESIGNATED PROGRAM INTO ACTION OR LOGS THE SUPERVISOR CUT OF THE QDF SYSTEM.

**SUPERORDINATE MODULES: LOGON**

**SUBORDINATE MODULES: MEnUI, C-REASGN, UTILMENU, SUPRPTS, SUPRT2**

**AUTHOR: J.G. Bouncy**

---

```
WELCOME TO THE QDF SUPERVISOR MENU

1 - MAIN MENU PROCESSING
2 - CASE REALLOCATION
3 - ANALYST WORKLOAD STATISTICS
4 - UTILITY PROGRAMS
5 - REPORT GENERATION
6 - YEAR END PROCESSING
7 - SOURCE 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

127
```
CASE V:CHOICE= 2
  RELEASE ALL LIKE V:*
  DO C:C-5EASGN.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:SUBREPS.PRG
CASE V:CHOICE= 6
  RELEASE ALL LIKE V:*
  DO C:YEAREND.PRG
CASE V:CHOICE= 7
  RELEASE ALL LIKE V:*
  DO C:SUBRT2.PRG
CASE V:CHOICE= 8
  RELEASE ALL EXCEPT C:*
  ERASE
  RETURN
ENDCASE
ERASE
STORE I TO V:CONTINUE
STORE I TO V:CONTINUE
ELSE
  ?' < PLEASE ANSWER WITH 1 - 8 ONLY >'
ENDIF <V:CHOICE>
ENDDO <V:CONTINUE>
***** ENI 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 available 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, COGCT, 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
   @ 6,26 SAY '***** Utility Processing *****'
   @ 9,29 SAY '1 - Analyst Update'
   @ $$,29 SAY '2 - Password Processing'
   @ $$,29 SAY '3 - COG Update'
   @ $$,29 SAY '4 - Address File Update'
   @ $$,29 SAY '5 - Internal Statistics Update'
   @ $$,29 SAY '6 - COG Count'
   @ $$,29 SAY '7 - Fe-Index Index Files For The System'
   @ $$,29 SAY '8 - Clean Up The Database (Pack)'
   @ $$,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'
   129
RELEASE ALL LIKE U:*  
RETURN  
CASE U:REPLY = '1' 
DO C:ANALYST  
CASE U:REPLY = '2' 
DO C:CHECK  
CASE U:REPLY = '3' 
DO C:COGUPDT  
CASE U:REPLY = '4' 
DO C:ADDRUPDT  
CASE U:REPLY = '5' 
DO C:STATGEN  
CASE U:REPLY = '6' 
DO C:COGCNT  
CASE U:REPLY = '7' 

***** Display Warning To The Operator  
ERASE  
@ 1,25 SAY '***** Data Base Reindex *****'  
@ 3,24 SAY '*'  
@ 4,24 SAY '*' WARNING  
@ 5,24 SAY '*'  
@ 6,24 SAY '*'  
@ 7,24 SAY '*' This Program Will Delete  
@ 8,24 SAY '*' All Index Files and Then  
@ 9,24 SAY '*' Will Re-Index All 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 '*'  
@ 16,24 SAY '*' Are You SURE You Want To'  
@ 17,24 SAY '*' Continue'  
@ 18,24 SAY '*'  
@ 19,24 SAY '<Enter Y or N>' + CHR(7)  
STORE 'TC U:REPLY2  
@ 21,40 GET U:REPLY2  
READ  

***** Accept Response From User  
DO WHILE !(U:REPLY2)<>'Y' .AND. !(U:REPLY2)<>'N'  
@ 23,32 SAY 'Enter Y or N Only' + CHR(7)  
@ 24,40 GET U:REPLY2 PICTURE 'A'  
READ  
ENDDO  
@ 23,32 SAY '  
@ 24,40 SAY '  

***** Accept and Verify Password Before Executing Request  
IF U:REPLY2 = 'Y'  
@ 21,30 SAY 'Enter Your Password '  
STORE ' ' TO U:PASSWORD  
SET CONSOLE OFF  
ACCEPT TC U:PASSWORD  
SET CONSOLE ON  
IF U:PASSWORD <> ' '  
USE D:TECHCODE INDEX D:TECH  
FIND &C:WHO  
IF PSWD = U:PASSWORD .AND. # <> 0  
DO C:UTILNDX  
ELSE  
@ 23,18 SAY 'Request ABORTED - Strike ';  
+'</Any Key To Continue

130
WAIT
ENDIF
ENDIF
CASH
REPLY = '8'
DO CJ.DPACK
*
,NECASE
-NDDC
*,NE CF
PROGRAM
ENDCASE
END
END CASE PROGRAM

**** ENC CP PROGRAM

131
**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.
**
** Subordinate Modules: OCASERPT
**
** Author: J.G. BCYNTON
**
******************************************************************************

STORE T TO C:TRUE
DO WHILE C:TRUE
ERASE *
STORE ' ' TC V: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 >

132
? ' PRESS ANY KEY TO CONTINUE'

WAIT

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

***** END OF PROGRAM
**SUPERVISOR REPORT MENU**

***SUPERVISOR REPORT MENU***

**Date:** 15 January 1984
**Version:** 1.0
**Module Name:** SUPRTE
**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 choose one or return to SUPMNU.
- 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:** SUPMNU1
**Subordinate Modules:** XXBNSSTAT, XXMNSTAT, CATIRPT, EXTVAL
**Author:** J.G. BOYNTON

```
ERASE
STORE T TO V:CONTINUE
DO WHILE V:CONTINUE
SET TALK OFF
STORE ' ' TO 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'
12.10 SAY ' ENDF

DO CASE
CASE V:CHOICE = "1"
REPLACE ALL LIKE V:*
DO C:XXEISTAT.PRG
CASE V:CHOICE = "2"
REPLACE ALL LIKE V:*
DO C:XXNSTAT.PRG
CASE V:CHOICE = "3"
REPLACE ALL LIKE V:* USE D:CEEN1 INDEX D:OCASE1
SET TALK OFF
STORE 0 TO P:COUNT
STORE C TO P:TOTAL
SET FORMAT TO SCREEN
ERASE
SET ALTERNATE TO D:CATIRPT
SET ALTERNATE ON
'TDATE: ',DATE()

**** 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,' 3(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 # ended price
OPEN DATE COG
STORE ROW+4 TO ROW
ENDIF <PAGE IS FULL>
ELSE
SKIP
ENDIF <NOT CAT 1>
ENDDO

? CAT 1 CASES:', P:COUNT
? TOTAL CASES:', P:TOTAL

**** END OF CAT';
EGORY I REPORT  

SET ALTERNATE OFF
SET ALTERNATE TO
? CHR(7)
? CHR(7)

@ 12,2C SAY 'You May Receive Your Cat i';
@ 13,2C SAY 'Report On ' D:CATIRPT.TXT ' 
@ 20,2C SAY ' Press Any Key To Continue'
WAIT

CASE V:CHOICE= "4"
USE D:ENI INDEX D:EXTVAL
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:EXTVALUE
SET ALTERNATE ON
'Date: ',DATE()

' EXTENDED VALUE REPORT ' **** Q3R';

'? CASE # COG SM ' SCREENING';
'+ NSN CAT NOMEN UIC ' EXT PRICE OPEN DATE CODE/DATE';
? STORE 0 TO P:PAGE
STORE 6 TO ROW
DO WHILE NOT. EOF
STORE P:TOTAL+1 TO P:TOTAL
? 'CASE ', COG ', SM ',
$ (NSN, 1, 4), ', $ (NSN, S, 2), '
$ (NSN, 7, 7), ', CAT ',
$ (NOMEN, 1, 10), ', UIE ',
$ (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';
? CASE # COG SM ';
? NSN CAT NOMEN ';
? UIC EXT PRICE OPEN';
? 'DATE CODE/DATE';
? STORE ROW+4 TO ROW
ENDIF <PAGE IS FULL>

SKIP

ENDIF

ENDIF <V:CONTINUE>

ENDIF <V:CONTINUE>

**** END OF PROGRAM
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 CQLCR TO 112, 6
STORE ' ' TC Q:REPLY
ERASE
@ 6, 26 SAY '*--- Query Processing *****'
@ 10, 27 SAY 'THIS PROGRAM ALLOWS YOU TO'
@ 13, 28 SAY 'OPEN THE QDR DATA BASE'
@ 15, 29 SAY '1 - Continue'
@ 17, 30 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 C:*
   RETURN
ENDIF

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

Y 7 -7 7. 7- -7
[101x727]1425
SAY '2
CLOSED
FILE'
@ 16,25 SAY '3
- Merced 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 'MUST
1 2, or 3' + CHR(7)
@ 20,40 GET Q:REPLY PICTURE '9'
READ
ENDDO

STORE 10 TO Q:CNTR
STORE 'Q:1' + 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:NFFASSES
STORE 'O' TO Q:FILE

****** If CLOSE File Is Selected Indicate C File Selection
CASE Q:REPLY = '2'
STORE 1 TO Q:NFFASSES
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 Q:REPLY = '3'
STORE 2 TO Q:NFFASSES
STORE 'O' TO Q:FILE
ENDCASE
STORE ' ' TC Q:SELCLE1
STORE ' ' TC Q:SELCLE2
STORE '00' TO Q:SELECT
STORE 0 TC 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 '56 End Element Select'
@ 5,33 SAY '59 Abandon Query'
* SET COLOR TO 112,6
@ 5,51 SAY '!
@ 6,25 SAY '1 Case Number'
@ 6,51 SAY '!
@ 7,1 SAY '01 Origin Code'
@ 7,25 SAY '11 Interim Report'
@ 7,31 SAY '2 Date'
@ 8,1 SAY '02 Cog'
@ 8,25 SAY '12 Type Document'
139
SAY '10 Origin Prep Date'
SAY '03 Category'
SAY '14 Date Received'
SAY '12 Number'
SAY '05 Nomenclature'
SAY '15 Open Date'
SAY '25 FSCH'
SAY '06 UIC of Origin'
SAY '16 Transmittal Date'
SAY '20 Contract Number'
SAY '07 Unit of Issue'
SAY '25 Unit Issue'
SAY '08 Unit Price'
SAY '18 Close Date'
SAY '28 Screening Code'
SAY '09 Quantity Deficient'
SAY '19 Reopen Date'
SAY '10 Extended Price'
SAY '20 Screen Report'
SAY '30 Next Page of Elements'
SAY '58 End Element Select'
SAY '59 Abandon Query'

READ DO WHILE Q:SELECT < '00' .OR. Q:SELECT > '59'
  READ ENEDO
  ELSE
    SAY 'relations a - include b - exclude'
  READ

IF Q:SELECT = '30'
  ERASE
  SAY 'enter selection criteria for this'
  SAY 'a maximum of 5 items may be selected'
  SAY 'data elements'
  SET COLCR TC 112,2
  SAY 'end element select'
  SET COLCR TC 112.6
  SAY 'region'
  SAY 'deficiency ver'
  SAY 'action code'
  SAY 'type defect'
  SAY 'deficiency resp'
  SAY 'cost code'
  SAY 'vendor liab code'
  SAY 'new-repair/ovhl'

140
+ "51 Status Code"
@ 10,1 SAY '34 Action Point'
@ 10,25 SAY '"43 Date Seq/No/Her'
@ 10,35 SAY '"2 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 QFM'
@ 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 Abandoned Query'
@ 16,51 SAY '"00 Prev Page of Elements'
@ 17,0 SAY '-----------------------------'
+ '"100 Prev Page of Elements'
@ 18,0 SAY '"Relations a - Include b - Exclude'
@ 18,4 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'
+ '"33,26 SAY 'Select From Above (00 - 59) '+CHR(7)
@ 21,27 SAY '"Enter Data Element Number ' GET Q:SELECT PICTURE '99'
READ ENDDO
ELSE
@ 23,26 SAY '

***** 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'
+ '"33,26 SAY '"Return'
ENDCASE

***** 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 '"99999A' TO Q:SELPIC
STORE '"' TO Q:INIT
STORE T TO Q:CHAR
STORE 'C:SELCMD1' TO Q:SELCMD
CASE Q:SELECT = '"02'
STORE 'COG' TO Q:SELITEM
STORE '"9A'" TO Q:SELPIC
STORE T TO Q:CHAR

141
STORE 'S (DATES, 1.5) TO Q:SELITEM
STORE '99999' TO Q:SELPIC
STORE '99999' TO Q:INIT1
STORE 'Q:CHAR
STORE 'C:SELCMD1' TO Q:SELCMD
CASE Q:SELECT = '14'
STORE 'S (DATES, 5.5) TO Q:SELITEM
STORE '99999' TO Q:SELPIC
STORE '99999' TO Q:INIT1
STORE 'Q:CHAR
STORE 'C:SELCMD1' TO Q:SELCMD
CASE Q:SELECT = '15'
STORE 'S (DATES, 11.5) TO Q:SELITEM
STORE '99999' TO Q:SELPIC
STORE '99999' TO Q:INIT1
STORE 'Q:CHAR
STORE 'C:SELCMD1' TO Q:SELCMD
CASE Q:SELECT = '16'
STORE 'S (DATES, 16.5) TO Q:SELITEM
STORE '99999' TO Q:SELPIC
STORE '99999' TO Q:INIT1
STORE 'Q:CHAR
STORE 'C:SELCMD1' TO Q:SELCMD
CASE Q:SELECT = '17'
STORE 'S (DATES, 26.5) TO Q:SELITEM
STORE '99999' TO Q:SELPIC
STORE '99999' TO Q:INIT1
STORE 'Q:CHAR
STORE 'C:SELCMD1' TO Q:SELCMD
CASE Q:SELECT = '18'
STORE 'S (DATES, 36.5) TO Q:SELITEM
STORE '99999' TO Q:SELPIC
STORE '99999' TO Q:INIT1
STORE 'Q:CHAR
STORE 'C:SELCMD1' TO Q:SELCMD
CASE Q:SELECT = '19'
STORE 'S (DATES, 41.5) TO Q:SELITEM
STORE '99999' TO Q:SELPIC
STORE '99999' TO Q:INIT1
STORE 'Q:CHAR
STORE 'C:SELCMD1' TO Q:SELCMD
CASE Q:SELECT = '20'
STORE 'S (DATES, 21.5) TO Q:SELITEM
STORE '99999' TO Q:SELPIC
STORE '99999' TO Q:INIT1
STORE 'Q:CHAR
STORE 'C:SELCMD1' TO Q:SELCMD
CASE Q:SELECT = '21'
STORE 'S (DATES, 31.5) TO Q:SELITEM
STORE '99999' TO Q:SELPIC
STORE '99999' TO Q:INIT1
STORE 'Q:CHAR
STORE 'C:SELCMD1' TO Q:SELCMD
CASE Q:SELECT = '22'
STORE 'S (DATES, 46.5) TO Q:SELITEM
STORE '99999' TO Q:SELPIC
STORE '99999' TO Q:INIT1
STORE 'Q:CHAR
STORE 'C:SELCMD1' TO Q:SELCMD
CASE Q:SELECT = '23'
STORE 'DOCON' TO Q:SELITEM
STORE "XXXXX999999999" TO Q:SELPIC
STORE '9999999999' TO Q:INIT1
STORE 'Q:CHAR
STORE 'C:SELCMD1' TO Q:SELCMD
CASE Q:SELECT = '24'
STORE 'BEPCON' TO Q:SELITEM
STORE "XXXXX999999999" TO Q:SELPIC
STORE '9999999999' TO Q:INIT1
STORE 'Q:CHAR
STORE 'C:SELCMD1' TO Q:SELCMD
CASE Q:SELECT = '25'
STORE 'DOCON' TO Q:SELITEM
STORE "XXXXX999999999" TO Q:SELPIC
STORE '9999999999' TO Q:INIT1
STORE 'Q:CHAR
STORE 'C:SELCMD1' TO Q:SELCMD
CASE Q:SELECT = '26'
STORE 'BEPCON' TO Q:SELITEM
STORE "XXXXX999999999" TO Q:SELPIC
STORE '9999999999' TO Q:INIT1
STORE 'Q:CHAR
STORE 'C:SELCMD1' TO Q:SELCMD
CASE Q:SELECT = '27'
STORE 'DOCON' TO Q:SELITEM
STORE "XXXXX999999999" TO Q:SELPIC
STORE '9999999999' TO Q:INIT1
STORE 'Q:CHAR
STORE 'C:SELCMD1' TO Q:SELCMD
CASE Q:SELECT = '28'
STORE 'BEPCON' TO Q:SELITEM
STORE "XXXXX999999999" TO Q:SELPIC
STORE '9999999999' TO Q:INIT1
STORE 'Q:CHAR
STORE 'C:SELCMD1' TO Q:SELCMD
CASE Q:SELECT = '29'
STORE 'DOCON' TO Q:SELITEM
STORE "XXXXX999999999" TO Q:SELPIC
STORE '9999999999' TO Q:INIT1
STORE 'Q:CHAR
STORE 'C:SELCMD1' TO Q:SELCMD
CASE Q:SELECT = '30'
STORE 'BEPCON' TO Q:SELITEM
STORE "XXXXX999999999" TO Q:SELPIC
STORE '9999999999' TO Q:INIT1
STORE 'Q:CHAR
STORE 'C:SELCMD1' TO Q:SELCMD
CASE Q:SELECT = '31'
STORE 'DOCON' TO Q:SELITEM
STORE "XXXXX999999999" TO Q:SELPIC
STORE '9999999999' TO Q:INIT1
STORE 'Q:CHAR
STORE 'C:SELCMD1' TO Q:SELCMD
CASE Q:SELECT = '32'
STORE 'BEPCON' TO Q:SELITEM
STORE "XXXXX999999999" TO Q:SELPIC
STORE '9999999999' TO Q:INIT1
STORE 'Q:CHAR
STORE 'C:SELCMD1' TO Q:SELCMD
CASE Q:SELECT = '33'
STORE 'DOCON' TO Q:SELITEM
STORE "XXXXX999999999" TO Q:SELPIC
STORE '9999999999' TO Q:INIT1
STORE 'Q:CHAR
STORE 'C:SELCMD1' TO Q:SELCMD
CASE Q:SELECT = '34'
STORE 'BEPCON' TO Q:SELITEM
STORE "XXXXX999999999" TO Q:SELPIC
STORE '9999999999' TO Q:INIT1
STORE 'Q:CHAR
STORE 'C:SELCMD1' TO Q:SELCMD
STORE T TO Q:CHAR
STORE "C:SELCMD1" TO Q:SELCMD
CASE Q:SELECT = "25"
STORE "FSCA" TO Q:SELITEM
STORE "XXXXXX" TO Q:SELPIC
STORE "T" TO Q:INIT1
STORE "C:SELCMD1" TO Q:SELCMD
CASE Q:SELECT = "26"
STORE "NUM" TO Q:SELITEM
STORE "XXXXX999XXXXXXX" TO Q:SELPIC
STORE "T" TO Q:INIT1
STORE "C:SELCMD1" TO Q:SELCMD
CASE Q:SELECT = "27"
STORE "CR" TO Q:SELITEM
STORE "A" TO Q:SELPIC
STORE "T" TO Q:INIT1
STORE "C:SELCMD1" TO Q:SELCMD
CASE Q:SELECT = "28"
STORE "SCR" TO Q:SELITEM
STORE "XXX" TO Q:SELPIC
STORE "T" TO Q:INIT1
STORE "C:SELCMD1" TO Q:SELCMD
CASE Q:SELECT = "29"
STORE "SN" TO Q:SELITEM
STORE "AX" TO Q:SELPIC
STORE "T" TO Q:INIT1
STORE "C:SELCMD1" TO Q:SELCMD
CASE Q:SELECT = "31"
STORE "C90" TO Q:SELITEM
STORE "XX" TO Q:SELPIC
STORE "T" TO Q:INIT1
STORE "C:SELCMD1" TO Q:SELCMD
CASE Q:SELECT = "32"
STORE "DEF" TO Q:SELITEM
STORE "99" TO Q:SELPIC
STORE "T" TO Q:INIT1
STORE "C:SELCMD1" TO Q:SELCMD
CASE Q:SELECT = "33"
STORE "VLC" TO Q:SELITEM
STORE "A" TO Q:SELPIC
STORE "T" TO Q:INIT1
STORE "C:SELCMD1" TO Q:SELCMD
CASE Q:SELECT = "34"
STORE "ACTPT" TO Q:SELITEM
STORE "AXXX999999" TO Q:SELPIC
STORE "T" TO Q:INIT1
STORE "C:SELCMD1" TO Q:SELCMD
CASE Q:SELECT = "35"
STORE "SCOTT" TO Q:SELITEM
STORE "9999999" TO Q:SELPIC
STORE "6" TO Q:NR
STORE "O" TO Q:LEC
STORE "O" TO Q:INIT1
STORE "C:SELCMD1" TO Q:SELCMD
CASE Q:SELECT = "36"
STORE "WHO" TO Q:SELITEM
STORE "XXXX" TO Q:SELPIC
STORE "T" TO Q:INIT1
STORE "C:SELCMD1" TO Q:SELCMD
CASE Q:SELECT = "37"
STORE "T" TO Q:CHAR
144
STORE 'Q:SELCMD1' TO Q:SELCMD
CASE Q:SELECT = '37'
STORE 'CTYINS' TO Q:SELITEM
STORE '#999999' TO Q:SELPIC
STORE 6 TO Q:NR
STORE 0 TO Q:DEC
STORE 0 TO Q:INIT1
STORE 'C:SELCMD2' TO Q:SELCMD
CASE Q:SELECT = '38'
STORE 'CTYREC' TO Q:SELITEM
STORE '#999999' TO Q:SELPIC
STORE 6 TO Q:NR
STORE 0 TO Q:DEC
STORE 0 TO Q:INIT1
STORE 'C:SELCMD2' TO Q:SELCMD
CASE Q:SELECT = '39'
STORE 'CTYSK' TO Q:SELITEM
STORE '#999999' TO Q:SELPIC
STORE 6 TO Q:NR
STORE 0 TO Q:DEC
STORE 0 TO Q:INIT1
STORE 'C:SELCMD2' TO Q:SELCMD
CASE Q:SELECT = '40'
STORE 'CLEFY' TO Q:SELITEM
STORE '#A' TO Q:SELPIC
STORE '#' TO Q:INIT1
STORE 'C:SELCMD2' TO Q:SELCMD
CASE Q:SELECT = '41'
STORE 'LEPR' TO Q:SELITEM
STORE '#A' TO Q:SELPIC
STORE '#' TO Q:INIT1
STORE 'C:SELCMD2' TO Q:SELCMD
CASE Q:SELECT = '42'
STORE 'ITEM' TO Q:SELITEM
STORE '#A' TO Q:SELPIC
STORE '#' TO Q:INIT1
STORE 'C:SELCMD2' TO Q:SELCMD
CASE Q:SELECT = '43'
STORE 'CVER' TO Q:SELITEM
STORE '#99999' TO Q:SELPIC
STORE '' TO Q:INIT1
STORE 'C:SELCMD2' TO Q:SELCMD
CASE Q:SELECT = '44'
STORE 'CTF' TO Q:SELITEM
STORE '#A9999' TO Q:SELPIC
STORE '' TO Q:INIT1
STORE 'C:SELCMD2' TO Q:SELCMD
CASE Q:SELECT = '45'
STORE 'GOV' TO Q:SELITEM
STORE '#X' TO Q:SELPIC
STORE '' TO Q:INIT1
STORE 'C:SELCMD2' TO Q:SELCMD
CASE Q:SELECT = '46'
STORE 'WUC' TO Q:SELITEM
STORE '#XXXXXX' TO Q:SELPIC
STORE '' TO Q:INIT1
STORE 'C:SELCMD2' TO Q:SELCMD
**** Display Selection Relations and Accept Relationship
and Initial Search Values

IF Q:SELECT<>'00' .AND. Q:SELECT<>'30' .AND.;
Q:SELECT<>'58'
STORE Q:ITEM + 1 TO Q:ITEM
SELECT '58' TO Q:SELECT
Co.0 SAY 'Relations a - Include  b - Excludes'
C 06,'12 SAY 'c - Range  d - Equal'
C 07,'10 SAY 'e - Not Equal  f - Less Than'
C 10,'10 SAY 'Greater Than'
A 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'
  23,13 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'
STORE Q:INIT1 TO Q:INIT2
A 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
RELEASE Q:TEMP
ENDIF

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

****** Format Numerics to be Characters For Code Generation
STORE STF(Q:INIT1,Q:NR,Q:DEC) TO Q:INIT1T
STORE Q:INIT1 TO Q:INIT1T
STORE STF(Q:INIT2,Q:NR,Q:DEC) TO Q:INIT2T
STORE Q:INIT2 TO Q:INIT2T
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; 
  + '+'+Q:SELITEM+<'+Q:INIT2 TO &Q:SELCMD
  147
**** Accept Selection Values For Query (Single Value)
  2 14.10 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 STE(Q:INIT1,Q:NR,Q:DEC) TO Q:INIT1
  RELEASE Q:INIT1
ENEIF

**** 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

  DC CASE
  CASE ! (Q:SELECT) = 'A'
    STORE &Q:SELCMD-Q:SELTEN+'='+Q:INIT1 TO &Q:SELCMD
  CASE ! (Q:SELECT) = 'B'
    STORE &Q:SELCMD-Q:SELTEN.='>'+Q:INIT1 TO &Q:SELCMD
  CASE ! (Q:SELECT) = 'C'
    STORE &Q:SELCMD-Q:SELTEN+'<>'+Q:INIT1 TO &Q:SELCMD
  CASE ! (Q:SELECT) = 'D'
    STORE &Q:SELCMD-Q:SELTEN+'='+Q:INIT1 TO &Q:SELCMD
  CASE ! (Q:SELECT) = 'E'
    STORE &Q:SELCMD-Q:SELTEN+'<>'+Q:INIT1 TO &Q:SELCMD
  CASE ! (Q:SELECT) = 'F'
    STORE &Q:SELCMD-Q:SELTEN.'>'+Q:INIT1 TO &Q:SELCMD
  CASE ! (Q:SELECT) = 'G'
    STORE &Q:SELCMD-Q:SELTEN+'='+Q:INIT1 TO &Q:SELCMD
  CASE ! (Q:SELECT) = 'H'
    STORE &Q:SELCMD-Q:SELTEN+'='+Q:INIT1 TO &Q:SELCMD
  ENDCASE

  STORE '00' TO Q:SELECT
ENDIF
ENDIF
ENDDO

**SET COLCR TO 112, 3
RELEASE Q:SELTEM, Q:SELPIC, Q:CHAR, Q:INIT1, Q:INIT2, Q:NR
STORE P TC Q:DATESEL
STORE 'CASE 1 TO Q:FLCMD1
STORE 'CASE 1 TO Q:FLCMD2
STORE ' TO Q:FIELD
STORE '0' TO Q:ITEM
STORE '0' TO Q:SELECT
STORE ' ' TO 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'
    STORE Q:ITEM TO Q:ITEM
  ELSE SAY 'Enter Field Selection For This Query'
ENDDO

148
I% S%"2,19 ;(A Maximum of 10 Fields May Be Selected)'
9 SAY 'Data Elements'
9 SAY 'Fields Currently Selected' + Q:CHOSEN
9 SAY 'Enter Field Number' GET Q:SELECT PICTURE '99'
READ

**** Display Previously Selected Fields
9 SAY 'Fields Currently Selected' + Q:CHOSEN
9 SAY 'Enter Field Number' GET Q:SELECT PICTURE '99'
READ

**** Validate Field Selection
DO WHILE Q:SELECT < '00' OR Q:SELECT > '59'
9 SAY 'Select From Above (00 - 59)' +%CHR(7)
9 SAY 'Enter Field Number' GET ;
Q:SELECT PICTURE '99'
READ
ENDDO

**** Display Second Screen For Selection
IF Q:SELECT = '30'
FRASE
9 SAY 'Enter Field Selection For This Query'
9 SAY '(A Maximum of 10 Fields May Be Selected)'
@ 4,0 SAY 'Data Elements'
+--------------------------:
@ 5,0 SAY 'End Element Select'
* SET COLOR TO 112, 2
@ 5,1 SAY '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
@ 8,1 SAY '32 Type Defect'
@ 8,25 SAY '!!' 41 Deficiency Resp
@ 9,1 SAY '33 Vendor Liab Code'
@ 9,25 SAY '!!' 42 New-Repair/Ovhl
@ 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!'
@ 12,1 SAY '36 Analyst Code'
@ 12,25 SAY '!!' 45 GFM'
@ 12,51 SAY '!!' 54 Part Number'
@ 13,1 SAY '37 Quantity Inspected
@ 13,58 SAY '!!' 46 Work Unit Code
@ 14,1 SAY '38 Quantity Received
@ 14,58 SAY '!!' 47 Discovery Code
@ 15,1 SAY '39 Quantity in Stock
@ 15,58 SAY '!!' 48 Return Code
@ 16,1 SAY '58 End Element Select
@ 16,51 SAY '!!' 59 Abandon Query'
@ 17,1 SAY '00 Prev Page of Elements'
@ 17,0 SAY '!!' 58 SELECT PICTURE '99:
READ

***** Validate Field Selection

DC 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

150
**** Store Selection Name to Q:SELITEM
**** Store Field Selection Pointer To Q:FLDCMD

CASE Q:SELECT = '01'
STORE 'CASE' TO Q:SELITEM
STORE 'C:FLDCMD1' TO Q:FLDCMD

CASE Q:SELECT = '02'
STORE 'CG' TO Q:SELITEM
STORE 'C:FLDCMD1' TO Q:FLDCMD

CASE Q:SELECT = '03'
STORE 'NSN' TO Q:SELITEM
STORE 'C:FLDCMD1' TO Q:FLDCMD

CASE Q:SELECT = '04'
STORE 'CAT' TO Q:SELITEM
STORE 'C:FLDCMD1' TO Q:FLDCMD

CASE Q:SELECT = '05'
STORE 'NOMEN' TO Q:SELITEM
STORE 'C:FLDCMD1' TO Q:FLDCMD

CASE Q:SELECT = '06'
STORE 'UIC' TO Q:SELITEM
STORE 'C:FLDCMD1' TO Q:FLDCMD

CASE Q:SELECT = '07'
STORE 'UI' TO Q:SELITEM
STORE 'C:FLDCMD1' TO Q:FLDCMD

CASE Q:SELECT = '08'
STORE 'UPRC' TO Q:SELITEM
STORE 'C:FLDCMD1' TO Q:FLDCMD

CASE Q:SELECT = '09'
STORE 'CTYDEF' TO Q:SELITEM
STORE 'C:FLDCMD1' TO Q:FLDCMD

CASE Q:SELECT = '10'
STORE 'EPBC' TO Q:SELITEM
STORE 'C:FLDCMD1' TO Q:FLDCMD

CASE Q:SELECT = '11'
STORE 'CRG' TO Q:SELITEM
STORE 'C:FLDCMD1' TO Q:FLDCMD

CASE Q:SELECT = '12'
STORE 'DOC' TO Q:SELITEM
STORE 'C:FLDCMD1' TO Q:FLDCMD

CASE Q:SELECT = '13'
STORE '$(DATES, 1,5)' TO Q:SELITEM
STORE 'C:FLDCMD1' TO Q:FLDCMD

CASE Q:SELECT = '14'
STORE '$(DATES, 6,5)' TO Q:SELITEM
STORE 'C:FLDCMD1' TO Q:FLDCMD

CASE Q:SELECT = '15'
STORE '$(DATES, 11,5)' TO Q:SELITEM
STORE 'C:FLDCMD1' TO Q:FLDCMD

CASE Q:SELECT = '16'
STORE '$(DATES, 16,5)' TO Q:SELITEM
STORE 'C:FLDCMD1' TO Q:FLDCMD

CASE Q:SELECT = '17'
STORE '$(DATES, 21,5)' TO Q:SELITEM
STORE 'C:FLDCMD1' TO Q:FLDCMD

CASE Q:SELECT = '18'
STORE '$(DATES, 26,5)' TO Q:SELITEM
STORE 'C:FLDCMD1' TO Q:FLDCMD

CASE Q:SELECT = '19'
STORE '$(DATES, 31,5)' TO Q:SELITEM
STORE 'C:FLDCMD1' TO Q:FLDCMD

CASE Q:SELECT = '20'
STORE '$(DATES, 36,5)' TO Q:SELITEM
STORE 'C:FLDCMD1' TO Q:FLDCMD

CASE Q:SELECT = '21'
STORE '$(DATES, 41,5)' TO Q:SELITEM
STORE 'C:FLDCMD1' TO Q:FLDCMD

CASE Q:SELECT = '22'
STORE '$(DATES, 46,5)' TO Q:SELITEM
STORE 'C:FLDCMD1' TO Q:FLDCMD
CASE Q: SELECT = '23' STORE '0': SELITE, ETCRC CF LDC "IJ1" VTLD' N: SBrCT Ll = LITZ TO Q:DLCMID1 TO Q:FLDCMD
CASE Q: SELECT = '7' STORE 'Q:SELITEM' STORE 'C:FLDCMD1' TO Q:FLDCMD
CASE Q: SELECT = '1' STORE '5CR' TO Q:SELITEM STORE 'C: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 'VLC' 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 'WHN' TO Q:SELITEM STORE 'C:FLDCMD1' TO Q:FLDCMD
CASE Q: SELECT = '38' STORE 'CTYREC' TO Q:SELITEM STORE 'C:FLDCMD2' TO Q:FLDCMD
CASE Q: SELECT = '39' STORE 'QTXSTK1' 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 'CEFR' 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 'OVER' TO Q:SELITEM STORE 'O: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 = '6' STORE 'IWUC' TO Q:SELITEM STORE 'C:FLDCMD2' TO Q:FLDCMD
CASE Q:SELECT = '47'
STORE 'DIS' TO Q:SELITEM
STORE 'C:FLDCMD2' TO Q:FLDCMD
CASE Q:SELECT = '48'
STORE 'SEC' TO Q:SELITEM
STORE 'C:FLDCMD2' TO Q:FLDCMD
CASE Q:SELECT = '49'
STORE 'ACTKN1' TO Q:SELITEM
STORE 'C:FLDCMD2' TO Q:FLDCMD
CASE Q:SELECT = '50'
STORE 'COSTC' TO Q:SELITEM
STORE 'C:FLDCMD2' TO Q:FLDCMD
CASE Q:SELECT = '51'
STORE 'STATUSC' TO Q:SELITEM
STORE 'C:FLDCMD2' TO Q:FLDCMD
CASE Q:SELECT = '52'
STORE 'CAUSEC' TO Q:SELITEM
STORE 'C:FLDCMD2' TO Q:FLDCMD
CASE Q:SELECT = '53'
STORE 'ACTDISP' TO Q:SELITEM
STORE 'C:FLDCMD2' TO Q:FLDCMD
CASE Q:SELECT = '54'
STORE 'MFG' TO Q:SELITEM
STORE 'C:FLDCMD2' TO Q:FLDCMD
CASE Q:SELECT = '55'
STORE 'LOT1' TO Q:SELITEM
STORE 'C:FLDCMD2' TO Q:FLDCMD
CASE Q:SELECT = '56'
STORE 'ITEM' TO Q:SELITEM
STORE 'C:FLDCMD2' TO Q:FLDCMD
CASE Q:SELECT = '57'
STORE 'WNYT' TO Q:SELITEM
STORE 'C:FLDCMD2' TO Q:FLDCMD
ENDIFCASE

IF Q:SELECT <> '00' AND. Q:SELECT <> '30' .AND.;
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+1 TO Q:CHosen
STORE Q:DISPLAY+1 TO Q:SELITEM TO Q:DISPLAY
ENDIF
IF Q:SELECT <> '01'
IF $(Q:SELITEM, 1)<>$ .AND. .NOT. Q:DATESel
STORE 'DATES' TO Q:SELITEM
STORE 'C:FLDCMD2' TO Q:FLDCMD
STORE 'T' TO Q:DATESel
ELSE
IF $(Q:SELITEM, 1)<>$ ,
STORE 'C:FLDCMD2' TO Q:FLDCMD
ENDIF
ENDIF

**** Generate Code For Field Selection
IF Q:FIELD = 1
STORE C:FIELD-Q:SELITEM TO Q:FIELD
ELSE
STORE Q:FIELD-1 , , -Q:SELITEM TO Q:FIELD
ENDIF
STORE '00' TO Q:SELECT
ENDIF
ENDDC

153
***** Release All Unnecessary Memory Variables

RELEASE Q:REPLY, Q:SELECT, Q:ITEM, Q:SELITEM, Q:SELCMD,
Q:FLDCMD, Q:CHOSEN

STORE Q:NREFASSES TO Q:LOOFCNT

***** If Both OPEN and CLOSED Files are Selected
***** The Execution Loop Will Be Run Twice

DO WHILE Q:LOOFCNT >= 1

***** Select File To Be Used In This Query

IF Q:FILE = 'O'
STORE 'D:' 'C:WHO-' 'CPEN' TO Q:TEMP3
ELSE
STORE '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'
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
STORE 'D:' 'C:WHO-' 'IMP1' TO Q:TEMP1
STORE 'CCPY TO ' + Q:TEMP1 TO &Q:LINE
IF Q:FLDCMD1 <> 'CASE'
STORE 68:LINE+ '+Q:FLDCMD1+' FOR '+Q:SELCMD1:' TO &Q:LINE
ELSE
STORE 68:LINE+ FOR '+Q:SELCMD1 TO &Q:LINE
ENDIF
STORE Q:CNTR + 1 TO Q:CNTR
STORE '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'
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 'D:' 'C:WHO-' 'IMP2' TO Q:TEMP2
STORE 'CCPY TO ' + Q:TEMP2 TO &Q:LINE
IF Q:FLDCMD2 <> 'CASE'
STORE 68:LINE+ '+Q:FLDCMD2+' FOR '+Q:SELCMD2:' TO &Q:LINE
ELSE
STORE 68:LINE+ FOR '+Q:SELCMD2 TO &Q:LINE
ENDIF
STORE Q:CNTR + 1 TO Q:CNTR
STORE 'Q:L'+STR(Q:CNTR,2) TO Q:LINE

154
**** Generate Code to Join Together Both Halves of the
**** Selected Files

IF Q:SELCMD1 <> ' ' AND Q:SELCMD2 <> ' '
  STORE 'SELECT PRIMARY' TO &Q:LINE
  STORE Q:CNTR + 1 TC Q:CNTR
  STORE Q:LINE + STR(Q:CNTR, 2) TO Q:LINE
  STORE 'USE ' + Q:TEMP1 TO EQ:LINE
  STORE Q:CNTR + 1 TC Q:CNTR
  STORE Q:LINE + STR(Q:CNTR, 2) TO Q:LINE
  STORE 'SELECT SECONDARY' TO &Q:LINE
  STORE Q:CNTR + 1 TC Q:CNTR
  STORE Q:LINE + STR(Q:CNTR, 2) TO Q:LINE
  STORE 'USE ' + Q:TEMP2 TO EQ:LINE
  STORE Q:CNTR + 1 TC Q:CNTR
  STORE Q:LINE + 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 EQ:LINE
    STORE Q:CNTR + 1 TO Q:CNTR
    STORE Q:LINE + STR(Q:CNTR, 2) TO Q:LINE
    ELSE
      STORE 'SELECT SECONDARY' TO &Q:LINE
      STORE Q:CNTR + 1 TC Q:CNTR
      STORE Q:LINE + STR(Q:CNTR, 2) TO Q:LINE
      STORE 'USE ' + Q:TEMP1 TO EQ:LINE
      STORE Q:CNTR + 1 TC Q:CNTR
      STORE Q:LINE + STR(Q:CNTR, 2) TO Q:LINE
      STORE 'SELECT PRIMARY' TO &Q:LINE
      STORE Q:CNTR + 1 TC Q:CNTR
      STORE Q:LINE + STR(Q:CNTR, 2) TO Q:LINE
      IF Q:FILE = 'O'
        STORE 'USE D:OPEN2' TO &Q:LINE
        STORE Q:CNTR + 1 TC Q:CNTR
        STORE Q:LINE + STR(Q:CNTR, 2) TO Q:LINE
      ELSE
        STORE 'USE D:CLOSE2' TO &Q:LINE
        STORE Q:CNTR + 1 TO Q:CNTR
        STORE Q:LINE + 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
  STORE Q:LINE + STR(Q:CNTR, 2) TO Q:LINE
  STORE 'USE ' TO EQ:LINE
  STORE Q:CNTR + 1 TC Q:CNTR
  STORE Q:LINE + 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:LINE + STR(Q:CNTR, 2) TO Q:LINE
  ELSE
    STORE 'SELECT SECONDARY' TO &Q:LINE
    STORE Q:CNTR + 1 TC Q:CNTR
    STORE Q:LINE + STR(Q:CNTR, 2) TO Q:LINE
    STORE 'USE ' TO EQ:LINE
    STORE Q:CNTR + 1 TC Q:CNTR
    STORE Q:LINE + 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:LINE + STR(Q:CNTR, 2) TO Q:LINE
ENDIF
ELSE IF \texttt{Q:FILE} = 0'\texttt{\} STORE 'USE D:OPEN1' TO \&Q:LINE

STORE Q:CNTR + 1 TO Q:CNTR

STORE 'Q:L' + STR(Q:CNTR,2) TO Q:LINE

ENDIF

ENLIF

ENDF

STORE Q:LOOFCNT-1 TO Q:LOOPCNT

***** If Two Passes Required, Switch to Closed File

***** For Second Pass

IF Q:NRPASSES = 2

STORE 'C' TO Q:FILE

ENDIF

ENDO

***** If Two Passes Required, Generate Code To Join Files

***** Created by Individual Passes

IF Q:NRPASSES = 2

STORE 'USE D:\*C:\WHC\*OPEN' TO \&Q:LINE

STORE Q:CNTR + 1 TO Q:CNTR

STORE 'Q:L' + STR(Q:CNTR,2) TO Q:LINE

STORE 'APPEND FROM '+'D:\*C:\WHO\*CLOS' TO \&Q:LINE

STORE Q:CNTR + 1 TO Q:CNTR

STORE '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:EXECNT

TO WHILE Q:EXECNTR <= Q:CNTR-1

STORE 'Q:L' + STR(Q:EXECNTR,2) TO Q:EXELINE
&Q:EXELINE
STORE Q:EXECNTR + 1 TO Q:EXECNTR
ENDIF
STORE "'F':*C:WHC:'OPEN.DBF'"' TO Q:OPENFILE
STORE "'F':*C:WHC:'CLOSE.DBF'"' TO Q:CLOSEFILE
IF FILE(Q:OPENFILE) AND FILE(Q:CLOSEFILE)
STORE 'D':*C:WHC:'OPEN' TO Q:USEFILE
DELETE FILE &Q:DELETEF
ELSE
IF FILE(Q:OPENFILE)
STORE 'D':*C:WHC:'OPEN' TO Q:USEFILE
ELSE
STORE 'D':*C:WHC:'CLOSE' 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 BOTTOM
ERASE
@ 10, 16 SAY 'Records Have Been Selected For This Query'
@ 10, 22 SAY '1. Print Hard Copy'
@ 10, 30 SAY '2. Display To Screen'
@ 10, 38 SAY '3. Abort Query'
STORE 'T':*C: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 Q:REPLY 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
XIII. STATISTICS GENERATION MODULE

******************************************************************************
** Date: 19 Jan 1984
** Version: 1.0
** Module Name: STATGEN
** Module Purpose: Generate Count of Cases in Each
** Processing Phase and Create Time
** Framework Statistics
** Module Interface Definition
** Inputs: None
** Outputs: None
** Module Processing Narrative Description:
** Calculates the Time Span Between Operations
** and Updates the TECHCODE File to Indicate the
** Number of Cases in Each Processing Status
** Superordinate Modules: UTILMENU
** Subordinate Modules: None
** Author: R. G. NICHOLS
****** Display Warning
ERASE
17,19 SAY '***** Statistics Generation Processing *****'
21,24 SAY '* * * * * * * * * * * * * * * *
19,24 SAY '**** WARNING
20,25 SAY 'This Program Will Read
24,25 SAY 'All Records While
28,25 SAY 'Processing
32,25 SAY 'If Existing Files Are
36,25 SAY 'Large, This Could Take
40,25 SAY 'Hours
44,25 SAY 'Are You SURE You Want To'
55,25 SAY 'Continue'
57,25 SAY '<Enter Y or N>' + CHR(7)
STORE 'TC S:REPLY2
21,40 GET S:REPLY2
READ
DO WHILE !(S:REPLY2) <> 'Y' .AND. !(S:REPLY2) <> 'N'
@ 23,32 SAY 'Enter Y or N Only' + CHR(7)
@ 21,40 GET S:REPLY2 PICTURE 'A'
READ
ENDDO
@ 23,32 SAY '
@ 17,40 SAY ': :

***** Verify Password Prior to Beginning Computations
IF S:REPLY2 = 'Y'
@ 21,30 SAY 'Enter Your Password'
STORE 'TC S:PASSWORD
SET CCNSCLE OFF

158
ACCEPT TC S:PASSWORD
SUB C:CONSUMER ON
IF S:PASSWORD <> "" THEN
USE D:TECHCODE INDEX D:TECH
FIND FC:HC
IF FS <> S:PASSWORD OR. # = 0
AND 23,18 SAY 'Request ABORTED',
+ '- Strike Any Key To Continue'
WAIT
RELEASE ALL LIKE S:
RETURN
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 PRIMARY
USE L:OPEN1 INDEX D:CASE1
***** Begin Computing Loop
DO WHILE NOT. EOF
DO WHILE $(DATES,46,1) <> ' ' AND. NOT. EOF
STORE CASE TO S:CASE
STORE WHO TO S:WHO
STORE 0 TO S:ASSIGNED
STORE 0 TO S:ACTIVE
STORE 0 TO S:TRANSMIT
STORE 0 TO S:RESPOND
STORE 0 TO S:CLOSED
STORE $(DATES,46,1) TO S:DATECHG
STORE $(DATES,6,5) TO S:RECEIPT
STORE $(DATES,11,5) TO S:OPEN
STORE $(DATES,16,5) TO S:XMIT
STORE $(DATES,26,5) TO S:RESPONSE
STORE $(DATES,36,5) TO S:CLOSE
STORE 0 TO S:MAILDLAY
STORE 0 TO S:XMITDLAY
STORE 0 TO S:RESPDLAY
STORE 0 TO S:CLOSEDLAY
STORE 0 TO S:PROCDELAY
****** Calculate Time Span From Receipt to Case Open
IF S:RECEIPT <> ' ' AND. S:OPEN <> ' ' THEN
IF $(S:RECEIPT,1,2) < $(S:OPEN,1,2)
IF VAL $(S:OPEN,1,2) - VAL $(S:RECEIPT,1,2) = 1
STORE VAL $(S:RECEIPT,3,3) + 365 \nVAL $(S:RECEIPT,3,3) + 1 TO S:MAILDLAY
ELSE
IF VAL $(S:OPEN,1,2) - VAL $(S:RECEIPT,1,2) = 2
STORE VAL $(S:RECEIPT,3,3) + 730;
-VAL $(S:RECEIPT,3,3) + 1 TO S:MAILDLAY
ELSE
STORE 999 TO S:MAILDLAY
159
Calculate Time Span From Case Open to Letter Transmit

**Calculate Time Span From Case Open to Letter Transmit**

IF S:XMIT <> ' ' IF $(S:OPEN,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

Calculate Time Span From Letter Transmit to Item Manager Response

**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

Calculate Time Span From Item Manager Response to Case Close

**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($(S:RESPONSE,3,3)) + 1 TO S:CLOSEDLAY ELSE IF VAL($(S:CLOSE,1,2)) - 

VAL($(S:RESPONSE,1,2)) = 2 

STORE VAL($(S:CLOSE,3,3)) + 730 - 

VAL($(S:RESPONSE,3,3)) + 1 TO S:CLOSEDLAY ELSE STORE 999 TO S:CLOSEDLAY ENDIF ENDIF ELSE 160
STORE VAL($S:CLOSE, 3, 3)) - VAL($S:RESPONSE, 3, 3)) + 1 TO $:PROCDELAY

**** 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 - 1 TO $:PROCDELAY
    ELSE
        IF VAL($S:CLOSE, 1, 2) - VAL($S:OPEN, 1, 2)) = 2
            STORE VAL($S:CLOSE, 3, 3)) + 730 - 1 TO $:PROCDELAY
        ELSE
            STORE 999 TO $:PROCDELAY
    ENDIF
ELSE
    STORE VAL($S:CLOSE, 3, 3)) - VAL($S:OPEN, 3, 3));
        + 1 TO $:PROCDELAY
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'
    STORE 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
    STORE 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
    STORE 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:CLOSED > 0 .AND. CLOSDLAY = 0
    STORE S:RESPOND + 1 TO S:RESPOND
ELSE
    IF S:CLOSED = 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
WITH S:MAILDLAY, XMITDLAY WITH S:XMITDLAY
S:MAILDLAY, 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

SELECT PRIMARY
REPLACE DATES WITH $(DATES, 1, 45) +''

SKIP
ENDDC
SKIP
ENDDC

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

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

***** END CF 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
END
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 '***** CCG Count Processing *****'
3,24 SAY '****************************'
4,24 SAY 'WARNING'
5,24 SAY 'This Program Will Read All Records While Processing'
6,24 SAY 'If Existing Files Are Large, This Could Take Hours'
7,24 SAY 'Are You SURE You Want To Continue'
8,24 SAY '<Enter Y or N>' + CHR(7)
STORE ' ' TC CC:REPLY2
READ DO WHILE !(CC:REPLY2) <> 'Y' .AND. !(CC:REPLY2) <> 'N'
17,24 SAY 'Enter Y or N Only' + CHR(7)
21,40 GET CC:REPLY2 PICTURE 'A'
READ ENDDO
23,32 SAY '**** Prompt For and Accept Password Verification

IF CC:REPLY2 = 'Y'
24,30 SAY 'Enter Your Password'
SET CC:SCREEN OFF
ACCEPT TC CC:PSWD
SET CCNSCLE ON
IF CC:FSW <> ,
  USE D:TECHCODE INDEX D:TECH
FIND CC:WHO
IF TSW <> CC:FSWD OR. = 0
  WAIT
  RELEASE ALL LIKE CC:* 
  RETURN
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 '***** DO NOT INTERRUPT WHILE PROCESSING *****'

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

***** Count the COGs Assigned Until End Of File Found
DO WHILE .NCT. EOF
  STORE CC:CURRENT TO CC:CURRENT
  STORE 0 TO CC:COUNI

***** Increment Counter Until a Different COG or End Of File Found
DC WHILE COG = CC:CURRENT .AND. .NOT. EOF
  STORE CC:CCUNT + 1 TO CC:COUNI
  SKIP
ENDDO
SELECT SECONDARY

***** Update IM Record
FIND &CC:CURRENT
  IF # <> 0
    REPLACE COUNT WITH CC:COUNT
    ENDF
    SELECT PRIMARY
ENDF
USE
SELECT SECONDARY
USE
RELEASE ALL LIKE CC:* 
RETURN

***** END OF PROGRAM
**BI-WEEKLY STATISTICS REPORT MODULE**

**DATE:** 27 JANUARY 1984

**VERSION:** 1.0

**MODULE NAME:** XXBISTAT

**MODULE PURPOSE:** CALCULATE BI-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 CLEAREST YEAR ARE CALCULATED AND STORED INTO MEMORY VARIABLES: OPEN AND CLOSE DATABASES ARE SEARCHED SEQUENTIALLY FOR ANY CASES WHICH WERE OPENED OR CLOSED 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

**SUBORDINATE MODULES:** NONE

**AUTHOR:** J.G. BOYNTON

**** INITIALIZATION OF VARIABLES

STORE ' ' TO BW:CURR
STORE ' ' TO BW:PREV
STORE ' ' TO BW:CID
STORE 0 TO EW:CREC
STORE 0 TO EW:CCLOS
STORE 0 TO EW:C9COG
STORE 0 TO EW:CSECC
STORE 0 TO EW:PREC
STORE 0 TO EW:PRECC
STORE 0 TO EW:PREOS
STORE 0 TO EW:PSCC
STORE 0 TO EW:OREC
STORE 0 TO EW:OCCLOS
STORE 0 TO EW:OCCLOS
STORE 0 TO EW:O9COG
STORE 0 TO EW:OSEC
STORE 0 TO EW:CREC
STORE 0 TO EW:PRECC
STORE 0 TO EW:ORECC
STORE 0 TO EW:OPERR
STORE 0 TO EW:PEROR
STORE 0 TO EW:OEROR

166
*** THIS SEQUENCE CALCULATES THE UPPER AND LOWER YEARS
*** FOR INPUT AND IS BASED ON THE CURRENT JULIAN
*** C:JULIAN, BW:LLIMIT = YEAR MINUS TWO YEARS
*** BW:ULIMIT = YEAR PLUS ONE YEAR

STORE $((C:JULIAN, 1, 2)) TO TEMP1
STORE VAL($((C:JULIAN, 1, 2))) TO C:JULIAN
STORE VAL($((C:JULIAN, 1, 2))) TO C:JULIAN
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>,'
@ 13,30 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
END IF
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:CURRE, BW:PREV, BW:OLD
*** ENTER THE CALL TO C:OJULIAN TO CHANGE MMDDYY TO C
*** 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:JULIANATE TC EW:CURRE
RELEASE ALL LIKE V:*

STORE $((BW:CURRE, 1, 2)) TO BW:TYR
STORE VAL((BW:TYR)) TO BW:TYR3
STORE BW:TYR3-1 TO BW:TYR1
STORE BW:TYR3-2 TO BW:TYR2
STORE STR((BW:TYR1, 3, 2)) TO BW:PREVT
STORE STR((BW:TYR2, 3, 2)) TO BW:OLDT
STORE BW:PREVT+$((BW:CURRE, 3, 3)) TO BW:PREV
STORE BW:OLDT+$((BW:CURRE, 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 ';
@ 13,30 SAY 'PROCESSED'
@ 14,20 SAY ' PLEASE STAND BY'
@ 23,30 SAY ' **** DO NOT INTERRUPT WHILE'

167
**PROCESSING ****

***** END DATE CHANGE AND ASSIGNMENT HERE

USE D:OPEN1

DO WHILE .NOT. EOF
STORE DATES TO M:DATES
STORE COG TO M:COG

STORE $(M:DATES, 1, 5) TO BW:CDAT
STORE $(M:DATES, 36, 5) TO BW:CDAT
IF $(BW:CDAT, 1, 2) = $(BW:CRR, 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 BW:C9COG + 1 TO BW:C9COG
ELSE
STORE BW:CSPCC + 1 TO BW:CSPCC
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 BW:P9COG + 1 TO BW:P9COG
ELSE
STORE BW:PSPCC + 1 TO BW:PSPCC
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 < THIS CASE IN OLDEST YEAR COUNT>

IF $(BW:CDAT, 1, 2) <> $(BW:CRR, 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 .NOT. EOF
STORE DATES TO M:DATES
STORE COG TO M:COG
STORE $ ( M:DATES, 11, 5 ) TO BW: CDAT
STORE $ ( M:DATES, 36, 5 ) TO BW: CDAT
IF $ ( BW: CDAT, 1, 2 ) = $ ( BW: CURR, 1, 2 )
STORE BW: CRECC + 1 TO BW: CRECC
IF BW: CDAT <> TYPE
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 <> TYPE
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 <> TYPE
STORE BW: OERROR + 1 TO BW: OERROR
ENDIF
ENDIF < THIS CASE IN OLDEST YEAR COUNT>
SKIP
ENDDO < SEARCH OF CLOSE1.DBF >
STORE BW: C9CG + BW: C9SPCC TO BW: CTOT
STORE BW: B9CG + BW: B9SPCC TO BW: PTOT
STORE BW: O9CG + BW: O9SPCC TO BW: OTOT
STORE BW: OERROR + 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: BWWKSTAT
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: SOTCT - BW: CTCT TO BW: OTRD
STORE BW: SPTCT - BW: PTCT TO BW: PTRD
STORE BW:SCICT-BW:CICI TO BW:CTRD

SET FOAMAT TO PRINT

@ 2,30 SAY " QCDE 9142 TECHNICAL BRANCH";
@ 4,30 SAY " QUALITY DEFICIENCY REPORT";
@ 6,30 SAY " BIWEEKLY STATUS REPORT";
@ 8,30 SAY " THRU";
@ 8,47 SAY " (BW:EDATE, 1, 2) /" + " (BW:EDATE, 3, 2) +";
@ 10,30 SAY " JULIAN DATE";
@ 10,50 SAY " SPCC 9-COG TOTAL";
@ 12,10 SAY " CALENDAR CASES OPEN CASES";
@ 13,10 SAY " CALENDAR CASES CLOSED CASES";
@ 14,10 SAY " YEAR RECEIVED OPEN CLOSED";
@ 15,10 SAY " CASES CASES CASES";
@ 18,10 SAY " E:YEAR STCRE BW:OREC+BW:ORECC TO BW:TOREC";
@ 18,18 SAY " BW:E C:RECC TO BW:TCREC";
@ 18,18 SAY " BW:E:PRECC TO BW:TPREC";
@ 18,30 SAY " BW:E:PRECC TO BW:TPREC";
@ 18,40 SAY " BW:E:OSPCC";
@ 18,50 SAY " BW:E:O9COG";
@ 18,60 SAY " BW:E:OTOT";
@ 18,70 SAY " BW:E:OTR";
@ 18,80 SAY " BW:E:OLOA";

STORE BW:OREC-BW:TPREC TO BW:TCREC

IF BW:TPREC > BW:C REC:
STORE "ECWN" TO BW:TLABEL
ENDIF
IF BW:TPREC < BW:C REC:
STORE "UP" TO BW:TLABEL
ENDIF
STORE BW:C REC-BW:TPREC TO BW:TTRD

EJECT

***** PAGE TWO

170
STORE BW:CREC+BW:PREC+BW:OREC TO BW:TRREC
STORE BW:CRECC+BW:PRECC+BW:ORECC TO BW:TRRECC
@ 2,30 SAY ' CODE 9142 TECHNICAL BRANCH'
@ 6,30 SAY ' WEEKLY STATUS REPORT'
@ 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 BW:TRREC
@ 16,25 SAY ' TOTAL RECORDS ON CLOSED FILES:'
@ 16,70 SAY BW:TRRECC
@ 18,25 SAY ' RECORDS WITH INVALID DATES,OPEN FILE:'
@ 18,69 SAY BW:OPER - 1
@ 20,25 SAY ' RECORDS WITH INVALID DATES,CLOSED FILE:'
@ 20,70 SAY BW:TEROR
@ 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 EW: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
**
** VERSION: 1.0
**
** MODULE NAME: XXMNSTAT
**
** MODULE PURPOSE: CALCULATE MONTHLY STATISTICS REPORT
**
** MODULE INTERFACE DEFINITION
**
** INPUTS: C:WHIC, 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 RETRIEVED IN D:MSBAD.TXT
** BY 'TYPING' USING THE OPERATING SYSTEM.
**
** SUPERORDINATE MODULES: SUPRPTS
** SUERORDINATE MODULES: NONE
** AUTHOR: J.G. BOYNTON
**
******************************************************************************

STORE 1 TO MS:ROW

****** INITIALIZATION CF VARIABLES

STORE 0 TO MS:CRCVD
STORE 0 TO MS:OPOTOT
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
****** MS:ULIMIT = YEAR PLUS ONE YEAR
****** MS:LLIMIT = YEAR MINUS TWO YEARS

STORE $(C:JULIAN,1,2) TO TEMP1
STORE VAL(TEMP1) TO TEMP1A
STORE VAL('1') TO LOW
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,LLMT,ULMT

****** INPUT OF REPORT CLOSING DATE

172
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
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($1(MS:EDATE,1,2)) TO V:MM
STORE VAL($1(MS:EDATE,3,2)) TO V:DD
STORE VAL($1(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 $1(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:PREDICT
STORE STR(MS:TYR2,2) TO MS:OLDT

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

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

***** Calculates FIRST DAY OF MONTH IN PREVIOUS YEAR

STORE VAL('C1') TO V:DD
DC C:OJULIAN
STORE V:JULDATE TO MS:PMUL1

***** Calculates END DATE OF MONTH IN OLDEST YEAR

STORE VAL($1(MS:EDATE,1,2)) TO V:MM

173
STORE VAL({MS:EDATE,3}), TO V:DD
STORE VAL({MS:OLDT}) TO V:VV
DC C:OJULIAN
STORE V:JULDATE TO MS:OJUL

***** CALCULATES FIRST DAY OF NEXT IN OLDEST YEAR

STORE VAL('01') TO V:DD
DC C:OJULIAN
STORE V:JULDATE TO MS:OJUL1
RELEASE MS:FEVT, MS:OLDT, MS:TYR, MS:TYR1, MS:TYR2,
MS:TYR3
RELEASE ALL LIKE V:*

IF @ 12, 20 SAY ' MONTHLY STATUS REPORT IS BEING ; + PROCESSING' ,
@ 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:CO1
STORE 0 TO MS:CO2
STORE 0 TO MS:CO3
STORE 0 TO MS:CO4
STORE 0 TO MS:PTOT
STORE 0 TO MS:OTOT

USE D:OPEN1

DO WHILE .NOT. EOF
STORE DATES TO M:DATES
STORE $(M:DATES,1,5) 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:CRCVCD + 1 TO MS:CRCVCD

****** IF CASE IS STILL OPEN IT SHOULD BE IN THIS FILE

IF MS:CRCV = 1
STORE VAL({MS:CJUL,3}) TO V:CDAT
STORE VAL(${MS:ODAT,3}) TO V:CDAT
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'

174
*** IF CASE IS IN PREVIOUS YEAR

IF $(MS:CDAT,1,2) = $(MS:PJUL,1,2)
    IF MS:ODAT > MS:PJUL AND MS:ODAT < MS:PJUL
        STORE MS:PTOT + 1 TO MS:PTOT

*** 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
ENDIF
ENDIF < CASE OPENED IN THE PREVIOUS YEAR >

*** IF CASE IS IN CLOSEST YEAR

IF $(MS:CDAT,1,2) = $(MS:OJUL,1,2)
    IF MS:ODAT > MS:OJUL AND MS:ODAT < MS:OJUL
        STORE MS:OTOT + 1 TO MS:PTOT

*** 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: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

175
ENDCASE

ENDIF

ENDIF < CASE OPENED IN THE OLDEST YEAR >

IF $\langle MS:CDAT, 1, 2 \rangle < \langle MS:CJUL, 1, 2 \rangle$ AND;
$\langle MS:CDAT, 2, 2 \rangle < \langle MS:CJUL, 1, 2 \rangle$ AND;
STORE MS:OPERR + 1 TO MS:OPERR
STORE CASE TO MS:CASE
SET ALTERNATE ON
? MS:ROW, 10 SAY 'OCASE!'
? MS:ROW, 16 SAY MS:CASE
? MS:ROW, 20 SAY $\langle MS:ODAT, 1, 2 \rangle + '/ '
+ $\langle MS:CDAT, 3, 2 \rangle + '/ ' + $\langle MS:ODAT, 5, 2 \rangle
? MS:CDAT, 40 SAY $\langle MS:CDAT, 1, 2 \rangle + '/ '
+ $\langle MS:CDAT, 3, 2 \rangle + '/ ' + $\langle MS:CDAT, 5, 2 \rangle
STORE MS:ROW + 1 TO MS:ROW
SET ALTERNATE OFF

ENDIF

STORE MS:CPTOT + 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

STORE 0 TO MS:OCL1
STORE 0 TO MS:OCL2
STORE 0 TO MS:OCL3
STORE 0 TO MS:OCL4
STORE 0 TO MS:PCL1
STORE 0 TO MS:PCL2
STORE 0 TO MS:PCL3
STORE 0 TO MS:PCL4
STORE 0 TO MS:CCL1
STORE 0 TO MS:CCL2
STORE 0 TO MS:CCL3
STORE 0 TO MS:CCL4

USE D;CLOSE1

DO WHILE NOT EOF IN D:DATES TO M:DATES
STORE $\langle M:DATES, 11, 5 \rangle$ TO MS:CDAT
STORE $\langle M:DATES, 36, 5 \rangle$ TO MS:CJUL

IF $\langle MS:CDAT, 1, 2 \rangle = $\langle MS:CJUL, 1, 2 \rangle$

IF MS:CDAT < MS:CJUL AND MS:CDAT < MS:CJUL
STORE MS:CRCVD + 1 TO MS:CRCVD

IF MS:CDAT <> $' ',
STORE VAL($\langle MS:CDAT, 3, 3 \rangle$) TO V:CDAT
STORE VAL($\langle MS:CDAT, 3, 3 \rangle$) TO V:CJUL
STORE V:CDAT - V:CDAT TO V:TIME
STORE STR(V:TIME, 3) TO MS:TIME
RELEASE ALL LIKE V:*

DO CASE
CASE MS:TIME < $' 61 '$
STORE MS:CCL1 + 1 TO MS:CCL1
CASE MS:TIME < $' 121 '$

176
END CASE
END IF
ENDIF < CASE OPENED IN THE CURRENT YEAR >

***** IF CASE IS IN PREVIOUS YEAR

IF $(MS:CDAT,1,2) = $(MS:OJUL,1,2)

IF MS:CDAT > MS:OJUL1 . AND. MS:CDAT<MS:OJUL
STORE MS:PTOT + 1 TO MS:PTOT

IF MS:CDAT <> '1'
STORE VAL($MS:CDAT,3,3) TO V:CDAT
STORE VAL($MS:OJUL,3,3) TO V:OJUL
STORE V:CDAT - V:OJUL TO V:TIME
STORE STR(V:TIME,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

END IF
ENDIF < CASE CLOSED IN THE PREVIOUS YEAR >

***** IF CASE IS IN CLOSEST YEAR

IF $(MS:CDAT,1,2) = $(MS:OJUL,1,2)

IF MS:CDAT > MS:OJUL1 . AND. MS:CDAT<MS:OJUL
STORE MS:PTOT + 1 TO MS:PTOT

***** IF CASE IS CLOSED < IT SHOULD BE TO BE IN THIS FILE >

IF MS:CDAT <> '1'
STORE VAL($MS:CDAT,3,3) TO V:CDAT
STORE VAL($MS:OJUL,3,3) TO V:OJUL
STORE V:CDAT - V:OJUL TO V:TIME
STORE STR(V:TIME,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

END IF
ENDDIF < CASE CLOSED IN THE:

IF T (MS: EDATE, 1, 2) <> S (MS: CDATE, 1, 2) OR S (MS: CDATE, 1, 2) <> S (MS: EDATE, 1, 2)
STORE CASE TO MS: CLERR + 1
STORE ALTERNATE ON
? MS: ROW, 10 SAY 'C:
? MS: ROW, 18 SAY MS.
? MS: ROW, 10 SAY S:
? +$ (MS: CDATE, 3, 2) +
+$ (MS: EDATE, 3, 2) +
STORE MS: EDATE + 1 TO
SET ALTERNATE OFF

ENDDIF
STORE MS: CLTOT + 1 TO MS:

SKIP
ENDDO < SEARCH FOR CASE IN T
RELEASE MS: CJUL, MS: JUL, MS

***** DETERMINE WHICH MONTH IT IS FOR

DO CASE
CASE $ (MS: EDATE, 1, 2) = '0'
STORE 'JANUARY' TO M:
CASE $ (MS: EDATE, 1, 2) = '1'
STORE 'FEBRUARY' TO M:
CASE $ (MS: EDATE, 1, 2) = '2'
STORE 'MARCH' TO MS:
CASE $ (MS: EDATE, 1, 2) = '3'
STORE 'APRIL' TO MS:
CASE $ (MS: EDATE, 1, 2) = '4'
STORE 'MAY' TO MS:
CASE $ (MS: EDATE, 1, 2) = '5'
STORE 'JUNE' TO MS:
CASE $ (MS: EDATE, 1, 2) = '6'
STORE 'JULY' TO MS:
CASE $ (MS: EDATE, 1, 2) = '7'
STORE 'AUGUST' TO MS:
CASE $ (MS: EDATE, 1, 2) = '8'
STORE 'SEPTEMBER' TO
CASE $ (MS: EDATE, 1, 2) = '9'
STORE 'OCTOBER' TO MS:
CASE $ (MS: EDATE, 1, 2) = '10'
STORE 'NOVEMBER' TO
CASE $ (MS: EDATE, 1, 2) = '11'
STORE 'DECEMBER' TO
ENDCASE

STORE 19 + $ (MS: OJUL, 1, 2)
STORE 19 + $ (MS: PJUL, 1, 2)
STORE 19 + $ (MS: CJUL, 1, 2)
STORE MS: OCL1 + MS: OCL2 + MS: OCL
STORE MS: O01 + MS: O02 + MS: O1
STORE MS: O05 + MS: OCL5 TO MS:
STORE MS: PCL1 + MS: PCL2 + MS: PCL
STORE MS: P01 + MS: P02 + MS: P
STORE MS: P05 + MS: PCL5 TO MS:
SET FORMAT TO PRINT

**** DATA FOR THE SECOND YEAR OUTPUT

**** DATA FOR THE CURRENT YEAR OUTPUT
CODE 9142 TECHNICAL BRANCH
QUALITY DEFICIENT MATERIAL

COMMAND KEY INDICATORS

AS OF:

(CODE: EDATE, 1, 2) */+*/ (CODE: EDATE, 3, 2) */+*/ (CODE: EDATE, 5, 2)

CASES RECEIVED IN:

CASES CLOSED:

TOTAL RECORDS ON OPEN FILE:

TOTAL RECORDS ON CLOSED FILES:

JULIAN DATE

MONTHLY STATUS REPORT

TOTAL RECORDS ON OPEN FILE:

TOTAL RECORDS ON CLOSED FILES:
\[ 16,20 \text{ SAY 'RECORDS WITH INVALID DATES, OPEN ';} \\
+ 'FILE:' \]
\[ 18,64 \text{ SAY MS:STOPPER = 1} \]
\[ 18,20 \text{ SAY 'RECORDS WITH INVALID DATES, CLOSED';} \\
+ 'FILE:' \]
\[ 19,65 \text{ SAY MS:CLEAR} \]
\[ 22,35 \text{ SAY 'END OF REPORT'} \]
\[ \text{SET ALTERNATE OFF} \]
\[ \text{SET ALTERNATE TO} \]

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: SUPET2
** MODULE 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: SUPENU1
** SUBORDINATE MODULES: SUPRT1, SUPRT2, SUPRT3, SUPRT4
**
** AUTHOR: 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 ';
@ 23,15 SAY 'PRESS 1 - TO ABORT, ANY OTHER KEY TC';
+COFFEE

WAIT TO V:BAIL
IF V:BAIL = '1'  
  RELEASE ALL LIKE V:*  
  RELEASE C:TRUE  
  RETURN  
ENDIF  
DO 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  
  STORE 0 TO P:CCOUNT  
  STORE 0 TO P:TOTAL  
  SET FORMAT TO SCREEN  
  ERASE  
  SET ALTERNATE TO D:SUPRPT1  
  SET ALTERNATE ON  
  8  
  ? 'DATE: ', DATE()  
  ?  
  8  
  GDR CFEN FILE BY ANALYST & CASE  
  ??  
  ? NSN / PART:  
  ??  
  + QTY     EXT 9  OPEN SCREENING  
  ??  
  - CASE:  
  + COG  SM  FSC  NATO PIIN  
  + CAT  NOMEN  UIC  
  + UI PRICE  Q ORG  DEF  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, OTYDEF, EPRC, O9O, ORG, DEF, DOC, NUM,  
    $(DATES, 11,5), SCR, $/S(DATES, 21,5)  
    STORE ROW+1 TO ROW  
    SKIP  
    STORE P:COUNT+1 TO P:COUNT  
    IF ROW > 60  
      ERASE  
      STORE CHR(12) TO ROW  
      STORE P:PAGE+1 TO P:PAGE  
      ?  
    8  
    PAGE ', P:PAGE  
    ?  
    + 'NSN / PART';  
    + 'QTY';  
    + 'UNIT';  
    + 'EXT 9';  
    + 'OPEN SCREENING';  
    ? CASE;  
    8  
 183
TOTAL CASES: ' $T:TOTAL

ENDDO

ENDIF <PAGE IS FULL>

" TOTAL CASES: ' $T:TOTAL

ENDOF

ENDDO

ENDIF <PAGE IS FULL>

TOTAL CASES: ' $T:TOTAL

ENDDO

ENDIF <PAGE IS FULL>

TOTAL CASES: ' $T:TOTAL

ENDDO

ENDIF <PAGE IS FULL>

TOTAL CASES: ' $T:TOTAL

ENDDO

ENDIF <PAGE IS FULL>

TOTAL CASES: ' $T:TOTAL

ENDDO

ENDIF <PAGE IS FULL>

TOTAL CASES: ' $T:TOTAL

ENDDO

ENDIF <PAGE IS FULL>

TOTAL CASES: ' $T:TOTAL

ENDDO

ENDIF <PAGE IS FULL>

TOTAL CASES: ' $T:TOTAL

ENDDO

ENDIF <PAGE IS FULL>

TOTAL CASES: ' $T:TOTAL

ENDDO

ENDIF <PAGE IS FULL>

TOTAL CASES: ' $T:TOTAL

ENDDO

ENDIF <PAGE IS FULL>

TOTAL CASES: ' $T:TOTAL

ENDDO

ENDIF <PAGE IS FULL>

TOTAL CASES: ' $T:TOTAL

ENDDO

ENDIF <PAGE IS FULL>

TOTAL CASES: ' $T:TOTAL

ENDDO

ENDIF <PAGE IS FULL>

TOTAL CASES: ' $T:TOTAL

ENDDO

ENDIF <PAGE IS FULL>

TOTAL CASES: ' $T:TOTAL

ENDDO

ENDIF <PAGE IS FULL>

TOTAL CASES: ' $T:TOTAL

ENDDO

ENDIF <PAGE IS FULL>

TOTAL CASES: ' $T:TOTAL

ENDDO

ENDIF <PAGE IS FULL>

TOTAL CASES: ' $T:TOTAL

ENDDO

ENDIF <PAGE IS FULL>

TOTAL CASES: ' $T:TOTAL

ENDDO

ENDIF <PAGE IS FULL>

TOTAL CASES: ' $T:TOTAL

ENDDO

ENDIF <PAGE IS FULL>
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, VIC;
CASE CAT, $NOMEN, 9, TIC;
CASE UI, $UPRC, QTYDEF, DOC;
CASE EPRC, $ORG, 09Q, $DEF, SCR;
CASE $(DATES, 21, 5)
STORE ROW+1 TO ROW
SKIP
STORE P:COUNT+1 TO P:COUNT
IF ROW > 60
ERASE
STORE 0 TO ROW
STORE P:PAGE+1 TO P:PAGE
PAGE ', P:PAGE
STORE ROW+1 TO ROW
ENDIF <PAGE IS FULL>
ENDDO

STORE ROW+4 TO ROW
ENDIF <PAGE IS FULL>

TOTAL CASES:', P:TOTAL

*********** END CF OPEN FILE REPORT BY ITEM MANAGER & ANALYST ***********

CASE V:CHOICE = '3'

FILE D:SUPTPT2.TXT IS CREATED TO PROVIDE THE REPORT
FILE D:SUPTPT2.NDX IS INDEXED ON COG*SM*$(DATES, 11, 5)*CASE

185
RELEASE ALL LIKE V:
USE D:GEN INDEX D:SUPRPT3
SET TALK OFF
STORE C TO P:COUNT
STORE C TO P:TOTAL
SET FORMAT TO SCREEN
ERASE
SET ALTERNATE TO D:SUPRPT3
SET ALTERNATE ON
"DATE: " "DATE()"

*** OPEN FILE BY 
"COG,SMIC,OPEN DATE & CASE ***

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: 
"UI: " "UPRC: 
"QTY: " "PRC: 
"DATE: " "SCR: 
$(DATES,215)
STORE ROW+4 TO ROW
IF ROW > 60 ERASE chr(12) STORE 0 TO ROW
STORE P:PAGE+1 TO P:PAGE
PAGE 'P:PAGE
"PAGE " "P:PAGE
"NSN / PART":

QTY EXT 9
OPEN SCREENING
CASE:
"COG,SM,FSC,NATO,PIIN: 
"NOMEN: 
"UI: 
"UPRC: 
"QTY: 
"PRICE: 
"DEPT: 
"DOC: 
"NU: 
$(DATES,115)
STORE ROW+1 TO ROW
STORE P:COUNT+1 TO P:COUNT
ENDIF <PAGE IS FULL>

186
STORE P:COUNT+1 TO P:COUNT
IF ROW > 60
    STORE 0 TO ROW
    STORE P:PAGE+1 TO P:PAGE
* PAGE ', P:PAGE
* ' NSN / PART';
* ' UNIT
* ' CTY
* ' OPEN SCREENING';
* ' CASE Cfg SM FS';
* ' C NATO FIN CAT NOMEN
* ' UIC JIT PRICE
* ' DFNT PRICE C OR
* ' G DF DF C CONTRAC
* ' T NUMBER DATE CODE/DATE
STORE ROW+4 TO ROW
ENDIF <PAGE IS FULL>
ENDDO

TOTAL CASES: ', P:TOTAL

END OF CLOSED FILE REPORT BY CREDIT CODE
& CASE
CHR(12)
SET ALTERNATE TO
CHR(7)
ERASE
@ 12,20 SAY 'YOU MAY RECEIVE YOUR REPORT BY':
@ 13,20 SAY ' CREDIT CODE & CASE ON';
@ 20,20 SAY ' D:SUPPT4.TXT';
WAIT
CASE V:CHOICE = '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
ELSE:
    PLEASE ANSWER WITH A 1 - 5 ONLY:
    PRESS ANY KEY TO CONTINUE
ENDIF
ENDDO <C:TRUE>
RELEASE ALL LIKE V:*  
RELEASE C:TRUE  
RETURN  

***** END OF PROGRAM
XIX. CASE REASSIGNMENT MODULE

DATE: 22 JANUARY 1984

VERSION: 1.0

MODULE NAME: C-REASN

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.

SUPERIOR INTEGRATED MODULES: UTILMENU

SUBLORDINATE MODULES: XDBHNDLR

AUTHOR: J.G. BOYNTON

***********

STORE T TO R:CONTINUE
DO WHILE R:CONTINUE
ERASE
@ 6,24 SAY '***** CASE RE-ASSIGNMENT PROCESSING *****'
@ 9,26 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'
@ 13,32 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'
STORE ' ' TO R:CASE
ERASE
@ 6,24 SAY 'ENTER DATA FOR CASE BEING RE-ASSIGNED'
@ 9,32 SAY 'CASE NUMBER' GET R:CASE
READ
STORE ! (R:CASE) TO R:CASE
STORE R:CASE TO M:KEY
STORE '1E' TO 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'
@ 21,14 SAY 'CASE DOES NOT EXIST IN OPEN'
190
+ '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
   @ 11,12 SAY 'NSN: ' M:NSN
   @ 11,14 SAY '$(M:NSN,5,2) +$(M:NSN,7,3) +1 +$(M:NSN,10,4)
   @ 11,15 SAY 'COG: ' M:COG
   @ 11,16 SAY 'CAT: ' M:CAT
   @ 14,10 SAY 'CURRENTLY ASSIGNED TO: ' M:WHO
   @ 15,10 SAY 'RE-ASSIGN TO: ' M:WHO
READ STORE ' ' TO R:REPLY,
DC WHILE R:REPLY < '1' .OR. R:REPLY > '3'
   @ 21,20 SAY '1 - RE-ASSIGN'
   @ 22,20 SAY '2 - CHANGE'
   @ 23,20 SAY '3 - EXIT'
   @ 24,20 SAY 'GET R:REPLY
READ ENDC
IF R:REPLY = '3'
  STORE '1G' TO M:TYPE
  DO XEBHNDLR.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 ';
  @ 11,20 SAY '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
  @ 21,19 SAY '**** DC NOT '
  @ 22,19 SAY 'INTERRUPT ****'
  USE D:OPEN1 INDEX D:OCASE1,D:NSN
  STORE R:NEW TO M:WHO
  STORE '1C' TO M:TYPE
  DO XEBHNDLR.PRG
  STORE M:CASE TO R:CASE
  RELEASE ALL LIKE M:*
  STORE R:CASE TO M:KEY
  STORE '2E' TO M:TYPE
  DO XEBHNDLR.PRG
  STORE '2C' TO M:TYPE
  STORE R:NEW TO M:WHO
  DO XEBHNDLR.PRG
  RELEASE ALL LIKE M:*
  ERASE
  STORE F TO R:REPLY
ENDIF
ENDDC <R:REPLY>
ENDCASE
ENDDC <CONTINUE>

***** END OF PROGRAM
THE CREATION OF A CENTRAL DATABASE ON A MICROCOMPUTER NETWORK(U) NAVAL POSTGRADUATE SCHOOL MONTEREY CA
J G BOYNTON ET AL. MAR 84
UNCLASSIFIED

F/G 9/2

NL
ITEM MANAGER FILE UPDATE

DATE: 12 JAN 84
VERSION: 1.0
MODULE NAME: ADDFUPDT
MODULE 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
AUTHOR: R. G. NICHOLS

STORE T TO A:CONTINUE
DO WHILE A:CONTINUE

****** DISPLAY SELECTION OPTIONS 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 RESPONSE
DC WHILE A:REPLY < '1' . OR. A:REPLY > '4'.
@ 43,32 SAY 'ENTER 1 - 4 ONLY' + CHR(7)
@ 15,40 GET A:REPLY PICTURE '9'
READ
ENDDO
DC CASE

****** IF CHOICE IS CUIT, 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'
192
STORE "" TO A:IM
ERASE
@ 6,20 SAY 'ENTER DATA FOR ITEM MANAGER 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:STATE
  STORE "" TO A:CITY
  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 - CREATE 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

193
***** REQUEST FOR AND ACCEPT IM TO BE UPDATED
ERASE
@ 6,19 SAY 'ENTER DATA FOR ITEM MANAGER BEING ';
* 'UPDATED';
@ 9,32 SAY 'ITEM MANAGER' GET A:IM
READ
STORE 1(A:IM) TO A:IM
SELECT PRIMARY

***** RETRIEVE IM RECORD BEING UPDATED
USE D:ADDRESS INDEX D: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, PROMPT 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 ';
+ 'REACTIVATED <Y OR N>'; 
@ 21,40 GET A:REPLY2 PICTURE 'A'
READ
DC 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
ENDIF
ENDDO

***** IF REACTIVATION REQUESTED
 IF ! (A:REPLY2) = 'Y'
@ 18,22 SAY 'RECALL ';
@ 19,17 SAY 'RECALL ';
@ 21,40 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 COMMAND2 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 'STATE ; GET STATE
@ 19,56 SAY 'ZIF CODE ; GET ZIP ;
READ 21, 20 SAY '1 - POST UPDATED RECORD ';
+ '2 - EXIT'
STORE ' ' TO A:REPLY3
@ 23, 40 GET A:REPLY3 PICTURE '9'

***** PROVIDE UPDATE/EXIT CFCTION
READ DO WHILE A:REPLY3 <> '1' .OR. A:REPLY3 > '2'
@ 23, 40 GET A:REPLY3
READ ENDDO

***** IF UPDATE
IF A:REPLY3 <> '1'
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), * ZIP WITH A:ZIP
ENDIF
STORE F TO A:PROCESS
ENDIF

***** IF CHOICE IS DELETE
CASE A:REPLY = '3'
STORE ' ' TO A:IM

***** ERASE FOR ANY ACCEPT IM BEING DELETED
ERASE
@ 6, 19 SAY 'ENTER DATA FOR ITEM MANAGER BEING ';
+ 'DELETED'
@ 9, 32 SAY 'ITEM MANAGER ' GET A:IM
READ STORE ! (A:TF) 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 'COMMAND2 ' + 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 ';

195
* CHF (7)

**WAIT**

ELSE

****** IF PREVIOUSLY DELETED, NOTIFY OPERATOR

IF *  
  @ 23,13 SAY 'RECORD PREVIOUSLY DELETED';  
  + '1' - 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  
  @ 17,20 SAY '1 - DELETE THIS ITEM MANAGER  
  + '2 - EXIT'  
  STORE ' ' TO A:REPLY2  
  @ 21,40 GET A:REPLY2 PICTURE '9'

****** ACCEPT DELETE/EXIT OPTION

REAL 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  

**ENDCASE**

**ENDIF**  

**ENDDC**

****** END OF PROGRAM
**COG FILE UPDATE MODULE**

*****************************************************
**
** DATE: 18 JAN 1984
** VERSION: 1.0
** MODULE NAME: COGUPDT
** MODULE PURPOSE: ALLOWS ADDITION, DELETION, OR
** UPDATING OF COG FILE
**
** MODULE INTERFACE DEFINITION
** INPUTS: NONE
** OUTPUTS: NONE
**
** MODULE PROCESSING NARRATIVE DESCRIPTION:
**
** ACCEPTS NEW COG DATA, VERIFIES THAT THE NEW
** COG IS NOT A DUPLICATE, AND VERIFIES THE
** EXISTENCE OF AN ITEM MANAGER RECORD. ACCEPTS
** UPDATE INFORMATION ON COG-ITEM MANAGER ASSIGN-
** MENTS, VALIDATES THE EXISTENCE OF AN ITEM
** MANAGER RECORD, 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 VERIFIES THAT NO ACTIVE CASES EXIST FOR
** THIS COG.
**
** SUPERORDINATE MODULES: UTILMENU
** SUBORDINATE MODULES: NONE
** AUTHOR: R. G. NICHOLS
**
*****************************************************

* COGUPDT.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 - ALL COG'
@ 10.28 SAY '2 - UPDATE COG'
@ 11.28 SAY '3 - DELETE COG'
@ 12.28 SAY '4 - RETURN TO UTILITY MENU'
STORE '1 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 TO QUIT, RELEASE LOCAL MEMORY VARIABLES
AND RETURN TO CALLING PROGRAM

CASE G:REPLY = '4'
RELEASE ALL LIKE G:

197
RETURN

***** IF CHOICE IS TC ADD, PROMPT FOR AND ACCEPT INPUT
CASE G:REPLY = '1'
STORE ' ' TO G:IM
STORE ' ' TO G:COG
PROMPT
@ 6,24 SAY 'ENTER DATA FOR COG BEING ADDED'
@ 9,32 SAY 'COG' GET G:COG PICTURE '9A'
READ
STORE I(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 TO 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 I(G:IM) TO G:IM

***** CHECK TO SEE IF IM RECORD EXISTS
FIND SG:IM
IF # <> 0
   STORE F TO 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 '9A'
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
   CASE G:REPLY2 = '2'
   STORE F TO G:GETIM
STORE ' ' TO G:TITLE
STORE ' ' TO G:COMMAND
STORE ' ' TO G:COMMAND2
STORE ' ' TO G:ATTN
STORE ' ' TO G:STREET
STORE ' ' TO G:CITY
STORE ' ' TO G:STATE
STORE ' ' TO G:ZIP

READ

***** ACCEPT NEW DATA AND PROMPT FOR CREATE/EXIT OPTION

@ 16,27 SAY '1 - POST NEW RECORD';
@ 19,30 SAY '+';
@ 20,30 SAY '2 - EXIT';
@ 23,40 GET G:REPLY3 PICTURE '9'
READ
DO WHILE G:REPLY3 < '1' .OR.;
   @ 23,40 GET G:REPLY3
END DO

***** 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'

199
STORE F TO G:GETIM
ENDCASE
ENDIF
ENDIF

***** IF CHOICE IS TC UPDATE
CASE G:REPLY = '2':
STORE ' ' TO G:IM
STORE ' ' TC G:GCOG

***** PROMPT FOR AND ACCEPT COG BEING UPDATED
ERASE
@ 9.32 SAY 'ENTER DATA FOR COG BEING UPDATED':
GET G:GCOG PICTURE '9A'
READ
STORE ! (G:CCG) TO G:GCOG
SELECT PRIMARY

***** RETRIEVE RECORD TO BE UPDATED
USE D:GCOG INDEX D:COGS
FIND &G:GCOG
IF # = 0
  @ 22.17 SAY 'RECORD NOT FOUND':
  + ' - S R I K E A N Y K E Y T O C O N T I N U E ' + CHR(7)
  WAIT
ELSE
  STORE T ? :PROCESS
ENDIF

***** 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)
  @ 24.40 GET G:REPLY3 PICTURE 'A'
  READ
ENDDO

***** REACTIVATE RECORD IF REQUESTED
IF !(G:REPLY3) = 'Y':
  @ 18.22 SAY ' 
  @ 19.19 SAY ' 
  @ 21.40 SAY ' 
  RECALL
ELSE
  STORE F TC G:PROCESS
ENDIF
ENDIF
DO WHILE G:PROCESS
STORE IN TO G:IM

***** PROMPT FOR AND ACCEPT UPDATE INFORMATION
@ 10.32 SAY 'ITEM MANAGER': GET IM
READ
@ 21.20 SAY '1 - POST UPDATE INFORMATION':
200
STORE '2 - EXIT' TO G:REPLY3
@ 23,40 GET G:REPLY3 PICTURE '9'

***** ACCEPT UPDATE/EXIT SELECTION
READ DC 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

***** POMPT 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:CCG INEX 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';
   @ 14,25 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 OPERATECF THAT RECORD PREVIOUSLY DELETED

IF * 23,13 SAY 'RECORD PREVIOUSLY DELETED - ';
   + 'STRIKE ANY KEY TO CONTINUE';
   WAIT
   ELSE

201
@ 10,32 SAY 'ITEM MANAGER? ' + CR
@ 17,20 SAY '1 - DELETE THIS COG ' + CR
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
ENDIF
ENDIF

ENDBLOCK CASE
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 **
** USERS PASSWORD IS VERIFIED TO ENSURE THAT HE **
** IS AUTHORIZED TO PERFORM THE PACK. **
** **
** SUPERIOR MODULES: UTILIMENU **
** SUBORDINATE MODULES: NONE **
** AUTHOR: R. G. NICHOLS **
**
*******************************************************************************

***** DISPLAY WARNING MESSAGE

ERASE
@ 1,25 SAY '***** DATA BASE PACKING *****'
@ 3,24 SAY **
@ 4,24 SAY ** WARNING **
@ 5,24 SAY **
@ 6,24 SAY ** THIS PROGRAM WILL PACK **
@ 7,24 SAY ** ALL DELETED CASES AND THEN **
@ 8,24 SAY ** WILL RE-INDEX THE FILES **
@ 9,24 SAY **
@ 10,24 SAY ** IF EXISTING FILES ARE **
@ 11,24 SAY ** LARGE, THIS COULD TAKE **
@ 12,24 SAY ** HOURS **
@ 13,24 SAY **
@ 14,24 SAY **
@ 15,24 SAY ** ARE YOU SURE YOU WANT TO' **
@ 16,24 SAY 'CONTINUE' **
@ 17,24 SAY '<ENTER Y OR N>' + CHR (7)
STORE ' ' TC P:REPLY2
@ 18,40 GET P:REPLY2

***** ACCEPT RESPONSE FROM USER

READ
DO WHILE ! (P:REPLY2) <> 'Y' .AND. ! (P:REPLY2) <> 'N'
@ 20,32 SAY 'ENTER Y OR N ONLY' + CHR (7)
@ 21,40 GET P:REPLY2 PICTURE 'A'
READ
ENDDO
@ 22,32 SAY :.
@ 23,40 SAY :.

***** ACCEPT AND VERIFY PASSWORD PRIOR TO EXECUTION

203
IF P:REPLY2 = 'Y'
  10 21.30 SAY 'ENTER YOUR PASSWORD '
  STOP ' ' TO P:PASSWORD
  SET CONSOLE OFF
  ACCEPT TC P:PASSWORD
  SET CONSOLE ON
  IF P:PASSWORD <> ' ' 
  USE D:TECHCODE INDEX D:TECH
  FIND 6C:WHO
  IF PSWE = 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:CASE1, D:ONSN
  PACK
  USE D:OPEN2 INDEX D:CASE2
  PACK
  ELSE
  @ 22.18 SAY 'REQUEST ABORTED';
    + ' ' - STRIKE ANY KEY TO CONTINUE'
  WAIT
  ENDIF
ENDIF
ENDIF
USE
RELEASE ALL LIKE P:*
RETURN

***** END OF PROGRAM
ANALYST FILE UPDATE MODULE

STORE T TO A:CONTINUE
DO WHILE A:CONTINUE

**** DISPLAY OPTIONS AVAILABLE TO THE USER AND ACCEPT SELECTION

ERASE
@ 6,26 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'
STOR ' ' 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 ENDO DC CASE

**** IF OUT put 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'
STORE '1' TO A:TECHCODE 
STORE ' ' TO A:PASSWORD 
STORE ' ' 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 EXCEEDED BLANKS 

DO WHILE $(A:TECHCODE,1,1)=' ' OR $(A:TECHCODE,2,1)=' ' OR $(A:TECHCODE,3,1): 
@ 23, 23 SAY 'ANALYST CODE CANNOT CONTAIN': 
+ 'BLANKS' + CHR(7) 
READ 
ENDO 
@ 23, 23 SAY 'STORE 'A:TECHCODE) TO A:TECHCODE 

**** VALIDATE FOR DUPLICATE USER ID 

USE D:TECHCODE 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 ' ' TO A:ENTERPSW 

**** FMCRT 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) 
STORE ' ' TO A:PASSWORD 
ELSE STORE ' ' TO A:PASSWORD 
ENDIF 
ENDO 
SET CONSOLE ON 

**** CREATE THE NEW ANALYST RECORD 

APPEND BLANK REPLACE TECHCODE WITH ! (A:TECHCODE), NAME WITH: 

206
A: NAME), PSWD WITH A: PASSWORD

****** IF UPDATE ANALYST SELECTED

CASE A: REPLY = '2'
STORE ' ' TO A: NAME
STORE ' ' TO A: TECHCODE

****** FORMAT FOR AND ACCEPT ANALYST CODE

ERASE
@ 6, 15 SAY 'ENTER ANALYST CODE FOR RECORD TO BE':
+ ' UPDATED' GET A: TECHCODE
READ
STORE 1 (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
ENDIF

****** 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<>'!' 
@ 23, 31 SAY 'ENTER Y OR N ONLY' + CHR(7)
@ 21, 40 GET A: REPLY2 PICTURE 'A'
REAL
ENDDO
IF (A: REPLY2) = 'Y'
RECALL
@ 18, 22 SAY ' ';
@ 19, 18 SAY ' ';
@ 21, 40 SAY ' ';
ELSE
STORE F TC A: PROCESS
ENDIF
ENDIF

****** FORMAT 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

207
READ
DO WHILE A:REPLY2 < '1' .OR. A:REPLY2 > '2'
  @ 23,40 GET A:REPLY2 PICTURE '9'
  REAL
ENDDO
IF A:REPLY2 = '1'
   REPLACE NAME WITH !(NAME)
ELSE
   REPLACE NAME WITH A:NAME
ENDIF
STORE F TO A:PROCESS
ENDIF
USE
ENDDC

***** IF DELETE OPTION SELECTED
CASE A:REPLY = '3'
STORE ' ' TO A:NAME
STORE ' ' 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
STORE !(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'
   STORE ' ' 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
**** IF LIST OPTION

CASE A:REPLY = "u"
  USE D:TECHCODE INDEX D:TECH
  SET DELETED CN
  ERASE
  DISPLAY ALL FIELD TECHCODE, NAME OFF
  STRIKE ANY KEY TO CONTINUE
  SET DELETED OFF
  WAIT
ENDCASE
ENDDC

**** END OF PROGRAM
**XXIV. PASSWORD FILE UPDATE MODULE**

***Date: 15 Jan 1984***  
***Version: 1.0***  
***Module Name: PASS***  
***Module Purpose: PASSWORD 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  
**Subordinate Modules:** None  
**Author:** R. G. Nichols

***** CLEAR SCREEN AND PROMPT FOR USER ID *****

ERASE  
STORE ':' TO P: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:TECHCODE INDEX D:TECH
FIND &P:TECHCODE
IF # = 0
    @ 22.14 SAY 'RECORD DOES NOT EXIST'  
        ' - STIRKE ANY KEY TO CONTINUE' +CHR(7)
    WAIT
ELSE
    SET CONSOLE 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:  
        ' - STIRKE ANY KEY TO CONTINUE' +CHR(7)
    WAIT
ELSE

    210
STORE T TO P:GETPASWD

**** ACCEPT NEW PASSWORD AND VALIDATION OF NEW PASSWORD

DO WHILE P:GETPASWD
  @ 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 TO P:GETPASWD
  ENDIF
ENDDO

REPLACE PSWD WITH P:PASSWORD
USE
ENDIF
SET EXACT OFF
SET CONSOLE ON
ENDIF

***** RELEASE LOCAL MEMORY VARIABLES AND RETURN TO
***** CALLING PROGRAM

RELEASE ALL LIKE P:*
RETURN

***** END CF PROGRAM
**MODULE NAME: UTILINDEX**
**MODULE PURPOSE: RE-INDEX ALL INDEX FILES**
**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 QDP system.

**SUPERSITVE NOODLE: UTILMENU**
**SUERORDOINE NOODLES: NONE**
**AUTHOR: J.G. BOYNTON**

**DELETE CURRENT INDICES**

DELETE FILE D:OCASE1.NDX
DELETE FILE D:ONS.NDX
DELETE FILE D:OCASE2.NDX
DELETE FILE D:CCASE1.NDX
DELETE FILE D:CCASE2.NDX
DELETE FILE D:TECHCODE

**BEGIN REINDEX OF FILES**

USE D:OPEN1
INDEX ON CASE TO D:OCASE1
INDEX CN NSN TO D:ONS

USE D:OPEN2
INDEX ON CASE TO D:OCASE2

USE D:CCASE1
INDEX ON CASE TO D:CCASE1
INDEX ON NSN TO D:CCASE2

USE D:CCASE2
INDEX ON CASE TO D:CCASE2

USE D:TECHCODE
INDEX ON TECHCODE TO D:TECH

USE D:COG
INDEX ON COG TO D:COGS

USE D:WHEREIS
INDEX ON CODE TO D:DISCODE
USE E:ADDRESS
INDEX ON IM TO D:IM
RETURN
***** END OF PROGRAM
OPEN CASE REPORT

******************************************************************************
** DATE: 5 JANUARY 1984  
** VERSION: 1.0  
** MODULE NAME: OCASERT  
** MODULE PURPOSE: PROVIDE ANALYST WITH LISTING OF ALL  
** 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  
** SUBORDINATE 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 >  

ENDIFTEXT  
@ 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  

214
USE D:OPEN 1
SET PRINT CN
REPORT FORM OPENCASE FOR WHO = C:WHO
EJECT
SET PRINT CPP
? 'PRESS ANY KEY TO CONTINUE'
WAIT
ENDIF
ENDIF
RELEASE V:PRINT
RETURN

***** END OF PROGRAM
LIST OF REFERENCES


BIBLIOGRAPHY


**INITIAL DISTRIBUTION LIST**

<table>
<thead>
<tr>
<th>No.</th>
<th>Copies</th>
<th>Recipient Details</th>
</tr>
</thead>
</table>
| 1.  | 2      | Defense Technical Information Center  
 |     |        | Cameron Station  
 |     |        | Alexandria, Virginia 22314 |
| 2.  | 2      | Library (Code 0142)  
 |     |        | Naval Postgraduate School  
 |     |        | Monterey, CA 93943 |
| 3.  | 1      | Naval Postgraduate School (Code 37)  
 |     |        | Computer Technologies Curriculum Office  
 |     |        | Monterey, CA 93943 |
| 4.  | 1      | Associate Professor Norman Lyons (Code 541b)  
 |     |        | Department of Administrative Science  
 |     |        | Naval Postgraduate School  
 |     |        | Monterey, CA 93943 |
| 5.  | 5      | Commanding Officer  
 |     |        | Navy Fleet Material Support Office (Code 93)  
 |     |        | SPCC Complex, Bldg 409  
 |     |        | Mechanicsburg, PA 17055 |
| 6.  | 1      | Major John G. Boynton  
 |     |        | U.S. Army Command & General Staff College  
 |     |        | Class 84/85  
 |     |        | Fort Leavenworth, Kansas 66027 |
| 7.  | 1      | LCDR Ronald G. Nichols  
 |     |        | Navy Fleet Material Support Office  
 |     |        | SPCC Complex, Bldg 409  
 |     |        | Mechanicsburg, PA 17055 |
| 8.  | 1      | Capt. M. D. Carriger  
 |     |        | Headquarters, U.S. Marine Corps  
 |     |        | (Attn: Code CCIS, Room 3037)  
 |     |        | Washington, D.C. 20380 |