AN EXERCISE IN DATABASE CUSTOMIZED PROGRAMMING TO COMPARE THE SMART DATA MANAGER AND dBASEIII

by

AMY LYNN FITZGERALD

B. S., Kansas State University, 1983

A MASTER'S REPORT

submitted in partial fulfillment of the

requirements for the degree

MASTER OF SCIENCE

Department of Industrial Engineering

KANSAS STATE UNIVERSITY
Manhattan, Kansas

1985

Approved by:

Major Professor

TABLE OF CONTENTS

AKNOWLEDGEMENTS

I would like to express my gratitude to Dr. Raj Vaithianathan and Dr. L.E. Grosh for their help. Without their guidance, this paper would not be possible. A special thanks to Raj; without his enthusiasm for education, I would not have been motivated to begin graduate school. But most of all, I would like to thank my family and my future husband, Hürriyet Necdet Aydoğan. Their love, patience, and most of all, support, helped me to achieve this goal.

LIST OF FIGURES

LIST OF TABLES

1. INTRODUCTION

Every organization must arrange information for quick and useful retrieval. Without the ability to retrieve the necessary information, the information is of no use to the organization. A database is simply a collection of data. A database management system (DBMS) provides the means to organize and access the database. The DBMS may be thought of as a data librarian. It stores and retrieves data.

1.1 DBMS Terminology

There are many terms used in DBMSs that sound familiar, but have somewhat unique meanings. Most people familiar with computer terminology have heard or used the term file. In DBMS terms, a file is a collection of information about entities with similar attributes. For example, a stock room of a retail auto parts store may have a file of suppliers, a file of parts-in-inventory, and a file of purchase orders. These files contain information on each entity according to the entity's type. If information about air filters is needed, it will be found in the parts-in-inventory file; not in either the suppliers file or the purchase orders file. Information about an individual entity is called a record. All the information about Ace Supplies may be found in the suppliers file in the Ace Supplies record. Individual pieces of data about a

record are kept in fields. The price field in the air filter's record will indicate the air filter's selling price.

A DBMS avoids repeating information as often as possible. So even though a purchase order may contain the supplier's name, it will not contain all the data about the supplier. This data will, of course, be found in the suppliers file. The supplier's name is common to a record in the purchase orders file and a record in the suppliers file. The use of common fields is the method by which DBMSs reduce data duplication. A relationship is established between two records via the common field - the supplier's name. If while using the purchase order file any additional information is needed about the supplier, the supplier's name is used to access the information in the suppliers file.

1.2 Database Models

Not all DBMSs structure and process the data in the same way. There are six common methods, or database models. Only two of these six models are common to microcomputer use. These are the Conference on Data System Languages, Database Task Group (CODASYL DBTG) model and the relational model.

1.2.1 The CODASYL DBTG Model

The CODASYL DBTG data model was developed during the late

1960s and is the oldest data model. Many DBMSs for large computer systems are based on this model. The CODASYL DBTG model is a very physical database model. Physical, in terms of models, means that the model is oriented toward machines and machine specifications. The opposite of a physical database model is a logical database model. A logical database model is oriented toward humans and human understanding. Due to the CODASYL DBTG model's physical nature, this model is not very popular with users.

1.2.2 The Relational Model

The relational model has both logical and physical characteristics. The relational model represents data in a familiar format - - a table as shown in Figure 1. The table is called a relation. A relation consists of rows, called tuples, and columns, called attributes. In Figure 1, the tuples are the part numbers and the attributes are price, number in inventory, and the reorder point. The significance of the relation is that the relationships are considered to be implied in the data values by their location in the relation. To demonstrate how the relationships are implied, note the value 200 in Figure 1. From evaluation of the table (or relation) it can be seen that the 200 represents the quantity of part number 1 in inventory. Though it may sound redundent, this is actually looking

at the quantity 200 from the point of view of the inventory, and the point of view of part number 1; thus the relationship is established.

PART #	PRICE	QTY ON HAND	REORDER POINT
1	0.50	200	75
2	0.90	540	100
3	3.60	25	5
4	12.40	60	10
5	1.25	980	90

Figure 1. PARTS-IN-INVENTORY RELATION

More research has been done regarding the relational model than any other model, and it has become the most popular model for microcomputer use.

1.3 Problem

The purpose of this master's report is to evaluate the Smart Data Manager. The means to accomplish this are simple. The Salina, Kansas YWCA (Young Women's Christian Association) recently decided to computerize some of its operations. Included in the computerization is its membership file system. In order to satisfy the varied needs at the YWCA, Innovative Software's the Smart System has been purchased. The membership files are to be handled by the Smart Data Manager. This situation provides the means to evaluate the Smart Data Manager;

but in order to judge this software effectively, it should be compared with another DBMS. Ashton-Tate's dBaseIII was chosen to serve this purpose because it is one of the market leaders.

2. THE SOFTWARE

2.1 The Smart System

The Smart System is an integrated software package; where integrated software merely implies that the package contains many types of independent subsystems for specialized functions and also that information may be easily passed between the independent subsystems. The Smart System consists of the Smart Word Processor, the Smart Spreadsheet with Graphics, and the Smart Data Manager. Each segment of the system may be used individually, or as an integrated whole.

2.1.1 The Smart Data Manager

The Smart Data Manager is primarily a menu-driven DBMS based on the relational model. The database management system software was written with the novice in mind. Due to the menu-driven format, the database may be used with very little training. The Data Manager, like the other programs in the Smart System, has confidence levels. The programs can be adjusted to be used by anyone in the range of novice to programmer. Each new level makes more commands and more sophisicated procedures available to the user. Project processing is available through either the remember mode, which records each key stroke, or through programming. Project Processing is what the Smart Data

Manager calls a user's application programming in the Smart System.

2.2 dBaseIII

dBaseIII is Ashton-Tate's new version of its popular dBaseII. This software, like the Smart Data Manager, is based on the relational model. dBaseIII is a more powerful version of dBaseII with fewer restrictions. There are two primary reasons why dBaseIII was chosen against which to compare the Smart Data Manager. First, although dBaseII is still the system with the largest client base; Ashton-Tate has replaced dBaseII with dBaseIII. Second, dBaseII is old-fashioned in comparison to the Smart Data Manager's capabilities. Most of the software that is new on the market have more capabilities and more power. dBaseIII is of approximately the same generation as the Smart Data Manager, so its choice is appropriate.

3. METHOD

The solution methodology consists on the following tasks:

1. Logical design of the system
2. Physical design of the Smart Data Manager software
3. Physically design of the dBaseIII software
4. Software implementation
5. Objective comparisons
6. Subjective comparisons
7. Conclusions

There are two approaches to the logical design. The first, is the ideal approach. In this approach, the software language that best fits the logical design is chosen for implementation after the design is completed. The second and the most common approach is to do the design with prior knowledge of the likely implementation language.

The system should be designed logically without taking the software into consideration. Realistically, this is not possible. The best way to do a design is always affected by the language's capabilities. In other words, a designer would not set up the logical design in a way that he/she knows is not possible to accomplish in the language to be used.

The physical design, in both languages, requires the adjustment in the logical design to fit the constraints of the software. These adjustments must be made for a variety of reasons, but the most common reason is that the software does not have the capability to do all that is required in the logical design. Once the adjustments have been made, the

design is implemented in the two chosen languages.

The comparisons are made on objective matters first. The objective comparisons include hardware and software characteristics. Such characteristics as the memory requirements, the number of disks required, the maximum number of records per file are compared. Once this is complete, the subjective comparisons are made. The subjective comparisons include those qualities that are judged by the user, and may vary from one user to another. These include such qualities as: ease in development of application programs, ease in learning, and documentation. When both comparisons are complete, conclusions are made.

4. DATABASE DESIGN

Database design is a two-phased process. First, users' requirements are examined and a conceptual database structure (or a model of the organization) is built. This phase of database design is called the logical database design. Once the logical design of the database is complete, the design is translated into the constraints of the particular DBMS. This process of formulating the logical design in terms of the DBMS is called the physical database design.

4.1 Logical Database Design

The logical database design, as stated above, specifies the needs of the user. There are five stages of logical database design. They are as follows:

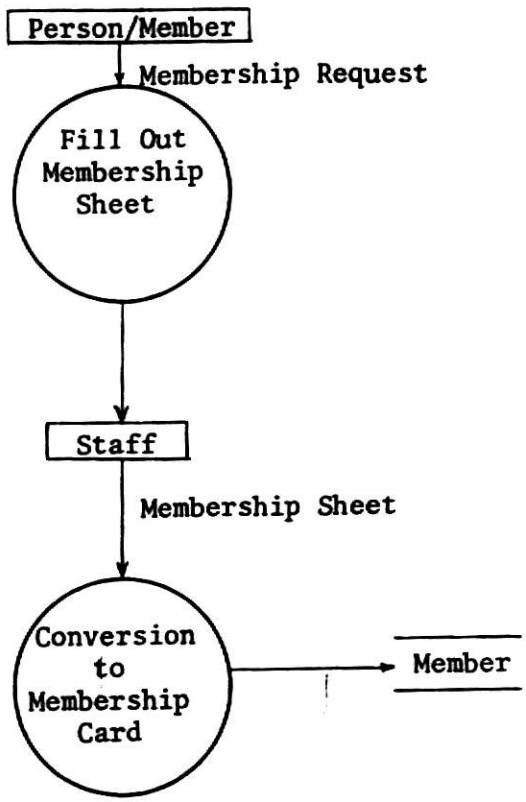
1. Identify data to be stored
2. Consolidate and clarify data names
3. Develop the logical schema
4. Define processing
5. Review the design

Stages one and two involve the data and the associated data names. The data needed are identified and aliases are eliminated or reduced. Aliases are common in most organizations. For example, what one group may call sales, another group may call future income. Identifying aliases is very much like detective work. It is time consuming and requires keen

investigative work. Aliases and data names need to be identified early, so unnecessary work will not be done in the future. The data dictionary is created in order to summarize and catalog the data names and their definitions. Each entry in the dictionary provides the data name, the data type, the data length, and any information that helps in the definition of the data item. An example of a data dictionary is detailed in Figure 2. The third stage, development of the logical schema, is accomplished by defining records and relationships. Records are defined by determining the data items they will contain. The fourth stage, define processing, is to examine the requirements to determine how the database may be manipulated to produce the required results. This stage is important in helping the designer to identify design flaws. Often process flow charts or psuedo code are completed during this stage in order to clarify the processes needed. Figures 3 and 4 contain examples of flow chart and psuedo code respectively. The fifth stage is not really a stage at all; it is a constant process throughout the logical design. The review of the design takes place from the moment designing begins to long after the design has been implemented. An important aspect of design review is the users' input. No one knows better what is needed than the one who needs it. The logical database design phase is very important. If it is done well, the physical database design is greatly simplified.

```
LNAME
      INDIVIDUAL'S LAST NAME
      TYPE:  CHARACTER
      LENGTH = 20

FNAME
      INDIVIDUAL'S FIRST NAME
      TYPE:  CHARACTER
      LENGTH = 10

SEX
      GENDER OF INDIVIDUAL
      TYPE:  CHARACTER
      VALUES:  M = MALE
               F = FEMALE
      LENGTH = 1

DATE
      CURRENT DATE
      TYPE:  DATE
      FORMAT:  MM/DD/YY
      LENGTH = 8

STATUS
      MARRIAGE STATUS
      TYPE:  CHARACTER
      VALUES:  M = MARRIED
               S = SINGLE
               W = WIDOWED
               D = DIVORCED
      LENGTH = 1
```

Figure 2. DATA DICTIONARY SAMPLE



Key:

File

Entity Outside System

Figure 3. PROCESS FLOW CHART FOR NEW MEMBER

```
PROSPECT FILE:  EXISTING DONOR DONATION, PLEDGE OR INSTALL-
-----------------------------------------------------------
MENT
----

LOCATE RECORD IN THE PROSPECT FILE
IF MAKING A DONATION
     CREATE RECORD IN THE GIVEN FILE
     ENTER DATA
     IF LETTER AND RECEIPT SENT
          SET LETTER TO "Y"
          SET RECPT TO "Y"
IF MAKING AN INSTALLMENT PAYMENT
     CREATE RECORD IN THE INSTALL FILE
     ENTER DATA
     STORE INSTL VALUE
     LOCATE RECORD IN THE PLEDGE FILE
     CALCULATE OUTSTD
          OUTSTD = AMT - INSTL
     IF OUTSTD <= 0
          SET PLDG IN RECORD IN THE PROSPECT FILE TO "N"
IF MAKING A PLEDGE
     SET PLDG TO "Y" IN RECORD IN THE PROSPECT FILE
     CREATE A RECORD IN THE PLEDGE FILE
     ENTER DATA
     IF FIRST INSTALLMENT PAID
          STORE INSTL VALUE
          CALCULATE OUTSTD:
               OUTSTD = AMT - INSTL
          CREATE INSTALL RECORD
          ENTER DATA
     ELSE
          OUTSTD = AMT
```

Figure 4. SAMPLE PSEUDOCODE FOR DONATION OR PLEDGE

4.2 The Physical Database Design

The physical database design begins where the logical database design ends. This phase of database design is a phase of transformation. The logical schema is transformed into the data constructs that are available with the DBMS in use. There are two major results of the database design. The first is the physical schema. The physical schema contains such information as the definition of the record contents, the name and the format of the fields of each record, and the values that each field may contain. If the logical schema, particularly the data dictionary, were done completely, the physical schema should require little effort. The second result is the definition of the content of the users' views. A user view is the portion of the database the user needs to see in order to perform the necessary tasks. The completion of the physical database design brings to a completion the design of the database.

5. LOGICAL DESIGN

The logical design, as stated in the database design section, specifies the needs of the user. In order to discover the users' needs, much of the work in this phase involves interviewing users and locating existing documents used by the user. There are many methods to define and document user needs. The traditional method is to interview users individually, and then to document, in ordinary English, the user needs. Recently, new procedures have been developed that allow users to be interviewed in groups with the use of such graphic tools as the data flow diagram. The data flow diagram shows how data and the processes that change data are connected. In this study, the traditional method is used. This is a sucessful method, in this case, because there are so few users. At the YWCA, there are only approximately two or three employees who will be affected by the installation of the computer.

To begin the logical design, the current procedures are evaluated. All forms and documents currently in use are gathered for future evaluation. The primary sources of information at YWCA, are the membership cards, donor information cards, and the volunteer time sheets. The membership cards are a color coded system. Adult members, those over 17 years old, are color coded by sex. Men, the nonvoting members, have

orange cards; and the women have blue cards. The cards are identical, except for the color. A typical adult membership card is shown in Figure 5. The youth memberships, those under 18 years old, are divided into three catagories: registrant, both sexes ages three to eleven; teen male, men ages 12 to 17; and teen female, women ages 12 to 17. These cards are all identical except for the color. The registrant card is green, the teen male card is yellow, and the teen female card is white. A typical youth membership card is shown in Figure 6.

The donor information cards, the prospect cultivation sûmmary as they are called, are kept for anyone who gave a donation, or made a pledge to the YWCA. A card is also kept on anyone who may be a potential source for a solicitation of a contribution in the future. An example of a donor information card is shown in Figure 7.

The volunteer time sheets are a record of the volunteer hours given. These sheets are completed by all volunteers, and include the time worked, at which department of the YWCA the volunteer worked, and the volunteer's name. These records allow the YWCA the means by which to recognize those who are donating their time to the organization. A sample volunteer time sheet is showm in Figure 8.

With the gathering of the documents complete, the processes in which these cards are used is evaluated. How the information for the documents is gathered, how the information is

THIS BOOK CONTAINS NUMEROUS PAGES WITH DIAGRAMS THAT ARE CROOKED COMPARED TO THE REST OF THE INFORMATION ON THE PAGE.

THIS IS AS RECEIVED FROM CUSTOMER.

Jan. Feb. Mar. Apr. May June July Aug. Sept. Oct. Nov. Dec.

Member - Community YWCA (Voting_____/Associate_____)

NAME _____
(Last) (First) (Middle)

Address _____
(Street & P.O. Box) (City, State & Zip) (Zone Code)

TELEPHONE: Home _____ Work _____

AGE (Check) 15-17_____ 18-24_____ 25-29_____ 30-34_____ 35-39_____ 60-64_____ 65 + _____

YOUR OCCUPATION & PLACE OF WORK _____

TODAY'S DATE _____

DO NOT WRITE IN THIS SPACE

EXPIRATION DATE

DATE PAID

Date _____ _____

Date _____ _____

Date _____ _____

Date _____ _____

Date _____ _____

YWCA RECORD (Special committees, staff, offices, board members, etc.:

Continue on Back

STATISTICAL SECTION

Please complete the following because the YWCA seeks to serve persons of all ages, creeds, races and backgrounds. This information is a very helpful tool to monitor this goal.

Brithdate _____/_____/_____
Month Day Year

Major Emphasis in Education: _____

Family Status: _____ Names/Ages of Children: _____

Year Moved to Salina: _____

YWCA Involvement: (current & past) _____

Racial Background: Asian Amer._____ Black_____ White_____
Native Amer _____ Hispanic ___ Other_____

PARTICIPATION

Date	Activity

Figure 5. ADULT MEMBERSHIP CARD

Jan. Feb. Mar. Apr. May June July Aug. Sept. Oct. Nov. Dec.

Member - Community YWCA (Registrant_____/Teen_____Teen Assoc._____)

NAME _____
(Last) (First) (Middle)

AGE_____ BIRTHDATE_____

ADDRESS _____

TELEPHONE _____

FIRST & LAST NAMES OF BOTH PARENTS OR LEGAL GUARDIANS_____

OCCUPATION & PLACE OF WORK IF ANY _____

NAME OF SCHOOL _____

NAME OF FAMILY DOCTOR_____ PHONE _____

PERSON TO CONTACT IN CASE OF ILLNESS OR ACCIDENT (Other than Parents)_____

_____ PHONE _____

SPECIAL SKILLS & TALENTS and/or OTHER GROUPS YOU BELONG TO:

PARTICIPATION

Date	Activity

DATE _____

DO NOT WRITE IN THIS SPACE

RENEWALS	PAID
Date _____	___
Date _____	___
Date _____	___
Date _____	___
Date _____	___
Date _____	___
Date _____	___
Date _____	___
Date _____	___
Date _____	___
Date _____	___
Date _____	___
Date _____	___
Date _____	___
Date _____	___
Date _____	___

Figure 6. YOUTH MEMBERSHIP CARD

PROSPECT CULTIVATION SUMMARY

Date_____

Name of Individual or Firm_____

Address_____ Phone_____

Name of Contact Person_____ Phone_____

Title_____ Contact Preference: __Home __Work

Information:	AGE RANGE	EDUCATION	FIELD OF STUDY	RELIG AFFILIATION
Prospect				
Spouse				
Family				

Special Interest_____

Past and Present YMCA Participation (Board, Committee, Club, etc.)_____

Comments_____

Past and Present Participation and Support of Other Organizations_____

Known well by_____

Record of Giving:

DATE	AMOUNT	PURPOSE	CONTACTED BY	COMMENTS

Outstanding Pledge:

DATE	AMOUNT	PURPOSE	INSTALLMENTS	COMMENTS

Recommended Solicitor_____

Figure 7. PROSPECT CULTIVATION SUMMARY

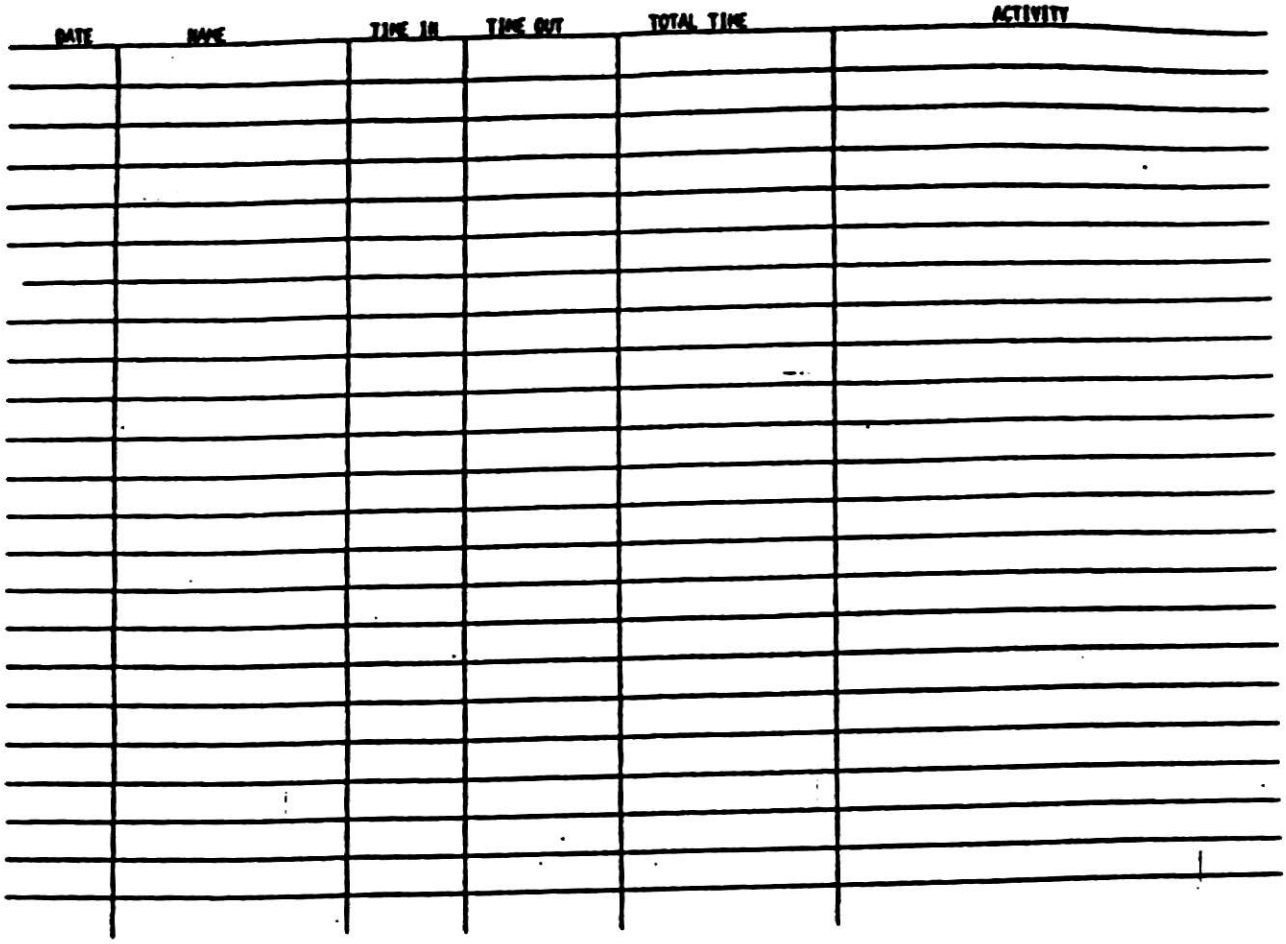


Figure 8. VOLUNTEER TIME SHEET

manipulated, and how it is stored is studied. For example, it is important to know that when a new member joins the YWCA, he/she fills out a sheet of paper similar to the membership card. The information on the sheet of paper is later typed onto a permanent membership card and kept in a file. Such processes as the new membership registration are often written in such a way that the database designer can easily recall and duplicate the processes. One popular method of writing processes is pseudo code. Pseudo code is a combination of the English laguage and computer programming logic. This pseudo code may later be translated into a computer program. The pseudo code for some of the processes is shown in Appendix A. It should be noted that as the design progresses, the logic behind the psuedo code also changes; so, the pseudo code shown is not necessarily the final logic used, nor is it complete.

The most time consuming and seemingly never ending stage of the logical design is the refining of data names. Each data name must be analyzed and recorded. Often information on data items is kept in a data dictionary. The data dictionary is a means of noting a data item and recording its definition, type, length, and any other pertinent information. This document is helpful as a reference for when the designer finds a data item that is unfamiliar to him/her. The data dictionary from this study is found in Appendix B. The data types used are character, numeric, and date. These three are chosen because they are common to almost all database software.

The completion of the gathering of the documents, the defining of the processes, and the refining of the data names concludes the logical database design. The physical database design begins where the logical design ends.

6. PHYSICAL DESIGN

The physical design is a manipulation of the logical design so that it fits into the constraints of the software. The largest portion of the effort expended in the physical design is the definition and naming of the records and the record fields. Most of the background information is acquired from the data dictionary. Since this study involves two DBMSs, the physical design will be developed twice; once for the Smart Data Manager, and once for dBaseIII.

6.1 The Smart Data Manager

The custom fitting of the logical design to the Smart Data Manager is very straight forward. The data items become the fields of the records, and the record structures given in the data dictionary are used almost directly. The only changes necessary are in the data types. When creating the data dictionary, it was assumed that only the data types character, numeric, and date existed. The Smart Data Manager supports these data types, but also has more sophisticated data types. These more sophisticated data types allow easier manipulation of the more specific data types. These types include the inverted name field, counter field, phone field, social security number field, and the time field.

The logical design is changed in only two ways. First, the inverted name field is used. This field allows a name to be entered as it would be said (firstname lastname or firstname middlename lastname). The field is a character field that sorts on the last word in the field. This eliminates the need to have a separate field for the first name, the middle name, and the last name. All of these fields merge to become the field called Name. Second, those fields that are phone numbers are changed from the character field type to the phone field type. The final Smart Data Manager data structures are detailed in Appendix C.

6.2 dBaseIII

The physical database design for the dBaseIII based system is almost as short and simple as is for the Smart Data Manager system. The record structures in the data dictionary are used almost directly. The only changes that occur are with the field types. dBaseIII supports the character, numeric, and the date types; but it also has more sophisticated types. dBaseIII uses logical fields and memo fields. Logical fields represent true/false values. Memo fields are designed to accomodate large blocks of textual information. These blocks are stored in an auxiliary file. These two field types are included in the record structures to give the final dBaseIII structures. These structures are documented in Appendix D.

7. IMPLEMENTATION

The software implementation process, which is the process of coding, debugging, and ensuring the functional integrity of the system must be done twice because two different DBMSs are used. This is by far the most time consuming stage of designing customized application programs. Included in this stage is the learning of the programming language, which may or may not be lengthy depending on the designer's experience.

7.1 The Smart Data Manager

The complete set of programs is documented in Appendix E. Included are the application programs to handle those processes that the YWCA requested. A list if the needed application programs is shown in Table 1.

1. New membership
2. Membership renewal
3. New donor
4. A donation or a pledge
5. A new volunteer
6. Entry of volunteer hours
7. Monthly processes
8. Generation of information for annual report

Table 1. Application programs required at the YWCA

7.2 dBaseIII

The final coding used for dBaseIII may be seen in Appendix F. Included are the application programs to handle the processes shown in Table 1.

8. COMPARISON

This chapter is written in order to present the contrast of the two DBMSs that are being evaluated. This section in no way claims to cover all differences in the two systems, but merely presents the differences encountered when completing this study.

8.1 Differences in Physical Design

The only subject affected in the physical design is the data types or field types. The field types that the Smart Data Manager and the dBaseIII support are shown in Figure 9.

Field Type	Smart Data Manager	dBaseIII
Character	YES	YES
Numeric	YES	YES
Date	YES	YES
Counter	YES	NO
Inverted Name	YES	NO
Logical	NO	YES
Memo	NO	YES
Phone	YES	NO
Social Security	YES	NO

Figure 9. SUPPORTED FIELD TYPES.

The inverted name field available in the Smart Data Manager is very convenient, and reduces the number of fields

required. Some confusion may result, however, if names are inadvertently entered with the last name first. The phone field, also available in the Smart Data Manager, reduces key strokes when inputting because the format for phone numbers is already presented. However, space is automatically allotted for the area code; so if the user will have phone numbers all in the same area code, the advantages of the phone field are lost.

The logical field and the memo field available in dBaseIII are both beneficial. The logical field makes logical testing simple. The memo field allows for input of textual information. The default length of the memo is 10 characters (or bytes), so if no information is input, the field only requires 10 bytes. If a character field is used for the same purpose, adequate space, say 50 bytes, must be reserved for that field. If no information is input, the field still requires 50 bytes. Therefore, if conservation of valuable disk space is important, the memo field will help effort in utilization of limited disk space.

8.2 Differences During Implementation

This section is a comparison of the Smart Data Manager and dBaseIII implementation. When viewing the application programs, it should be noted that these programs do not

represent the only applications needed at the YWCA. The YWCA only desires to have applications written for frequent processes that would require knowledge of the language in order to execute. Infrequent, or one-of-a-kind, requests may be satisfied by the use of the menu feature in the Smart Data Manager software. Though dBaseIII does not have the menu-driven capabilities of the Smart Data Manager, only those application programs needed for Smart are written in dBaseIII.

8.2.1 Storage of Programs into the DBMS

The storage of programs involves how the programs are entered into the DBMS, so that they may later be executed. dBaseIII provides full screen editing when using the dBase word processor, and programs are input as on a typed page. The Smart Data Manager provides two options. The user may either input a program by editing only, or by using the remember mode to do the typing for him/her. The remember mode is designed to handle frequent tasks. After entering this mode by merely typing "start", the frequent task is completed by the user. When the user is done, the remember mode is exited by typing "finished". While in the remember mode, the DBMS has recorded every key stroke entered and stored it in a file. This file may then be edited to erase any mistakes made, or to add logic statements to provide decision making capabilities.

8.2.2 User Involvement in Execution

The user involvement in execution includes the requirements demanded of the user in order to execute the programs. In the dBaseIII application the user must only type "DO MAIN". The programs will all automatically execute as needed. This is not always true with the Smart Data Manager. Due to Smart's interactive requirements, there are situations in which the user must work directly with the system rather than solely with the application programs. An example of this is the entry of a new record. The application program brings the user to the new record, but once the record is completed, it is up to the user to tell the system that he/she is done. The application program then regains control and continues execution. If the system is not notified that the entry is complete, it will attempt to enter another new record. This is not the case in dBaseIII. The system "knows" when the entry into the record is complete, so it then continues execution.

8.2.3 Communication Between User and Programs

It is often necessary for the application program to question the user about what is needed, or to explain a procedure. dBaseIII allows the entire screen to be used by the programs to display questions or messages. This provides

virtually unlimited space for communication with the user. In the Smart Data Manager, however, the available screen space is restricted. Approximately 80 percent of the Smart screen is reserved for viewing of the databases in what is called the window area. Below the window is either the command list area, if in the interactive mode, or a blank area, if executing an application program. If the program asks a question, or gives an explanation to the user, it must fit in only one line of the blank area. This restriction may sometimes become frustrating to the programmer.

8.2.4 Use of Variables

Variables are constantly used in application programs in order to store information. They are an important aspect of a DBMS's programming. The Smart Data Manager allows the use of two text variables, and two numeric variables. These variables are called TEXT1, TEXT2, VALUE1, VALUE2. Other variables may be introduced, but only after intialization. dBaseIII allows the use of up to 256 variables at one time. If a specific program requires the use of many variables, the initialization needed by Smart may become burdensome.

8.2.5 Use of Database Files

In order to perform any manipulation of data in a

database, the records of the database need to be loaded, or activated. In dBaseIII the user, or programmer, merely tells the system which database to use. However, the Smart Data Manager also requires that a screen design be designated. The screen format is a predefined custom data entry screen. This causes no problem, except when more than one screen design may be needed. In that case the first screen must be unloaded, and the new one must then be loaded.

8.2.6 The Date Field Type

Both dBaseIII and the Smart Data Manager provide the date field type, but neither provide the same abilities to manipulate the date field. The Smart Data Manager supplies many more date functions than dBaseIII. In Smart, for example, the function ADDYEAR allows years to be added to a date. The ADD function is also available for days and months. The MONTHNAME function returns the name of the month. For example, the function MONTHNAME(8/11/85) will yield August. These functions, among others, make the manipulation of date fields much easier in the Smart Data Manager.

8.2.7 Entry of a New Record

Updating of a database is a constant process, so new records are often added. dBaseIII supplies two methods to

accomplish the entry of new records. Either a blank record can be shown, or the user may input answers to questions. These answers are either stored in variables for later entry, or placed directly into the blank record. This provides the programmer flexibility to request information when appropriate. The Smart Data Manager, however, only has the ability to allow entry of a new record by having the user input directly into the blank record.

9.2.8 Key Fields

Key fields are fields by which the database may be sorted. So if NAME is a key field, the records may be put in alphabetical order by name. The Smart Data Manager requires that key fields be designated when the fields are defined. This is not only awkward, but it also is time consuming when entering new records into the database. After the entry of each record, the question, "Update Keys now (Y/N)?", is asked. If the user answers "yes", he/she must wait while the entire index is adjusted for each key field. When the Smart was implemented a separate program was included that will update the keys for all new records. This saves time when entering records, but still requires time at the completion of all the records to do the updating.

8.2.9 File Space Requirements

The required file space for all the files other than the application programs was evaluated. It was found that the Smart Data Manager required 78 files, while dBaseIII required only 28 files. This amounted to an approximate difference of 100,000 bytes. This extremely large difference may be explained by the Smart System's use of key fields. Each designated key field has a file that is responsible for the location of all the records in the database using that field. Therefore if a database has 15 key fields, ther are 15 extra files associated with that particular database. This is the primary reason for the large differences in the number of files.

8.2.10 Debugging

Debugging, or the elimination of programming errors, is sometimes frustrating, but Smart provides a means to reduce the frustration. It is called single step execution. This type of execution of programs requires the programmer to press a key in order for the next command to be executed. Single step execution allows the programmer time to observe each command as it is executed, and to evaluate its results.

9. DIFFERENCES DURING EXECUTION

It is intended that the application programs in both languages be identical. However, this is impossible; the different languages do not permit identical programs to be written. It was attempted to make the two programs as similar as possible, while using the capabilities of each DBMSs to its fullest. The differences in the programs make a definitive comparison of execution times debatable. The reader should, therefore keep the appropriateness of the comparison in mind. The programs were executed on an IBM PC equipped with dual diskette drives.

9.1. Entry Of New Records

Excluding the time to type the data into the blank record, the input of one record per trial was timed. Ten trials were performed in each language to achieve the average time per trial. The averages are very different; this is due to the key updating as mentioned in section 8.2.9. The average time to input one new record into the Smart Data Manager with keys updated, is two minutes and 23 seconds. This is extremely high when compared to the dBASEIII time of only slightly over one second. It should be noted that no further indexing is needed in Smart, but it is necessary in dBASEIII, so some of the time descrepancy may balance out.

9.2. Renewal of Current Record

The renewal of a current record is very similar to the process involved in the entry of a new member, except that the record must be located. This increases the average time to compute the renewal, but the large difference is still obvious. The Smart Data Manager took approximately two minutes and 38 seconds to execute; while dBASEIII required only 28 seconds. Again, the key updating by Smart is an important reason for the descrepancy.

10. OBJECTIVE COMPARISON

The objective comparisons include those characteristics that are inherent in the software. These characteristics are not capable of being judged, but are considered given. As can be seen in Table 2 on the following page, the two DBMSs, the Smart Data Manager and dBASEIII, are objectively very similar.

Table 2. COMPARISON OF OBJECTIVE CHARACTERISTICS

Characteristic	dBASEIII	Smart DM
Vendor	Ashton-Tate	Innovative Software, Inc.
Price	$695	$495
Memory Req (in bytes)	256K	256K
PC AT Compatible	No	Yes
Disks Required	2	2
Copy Protected	Yes	Yes
Command Driven	Yes	Yes
Menu Driven	Yes	Yes
Provides Help Screens	Yes	Yes
Provides On-Disk Tutorial	Yes	Yes
Multi-User Capability	No	Yes
Max # of Fields/Record	128	255
Max Field Size (Characters)	4000	1000
Max # of Records/File	Unlimited	100,000
Max Record Size (Characters)	4000	4096
Allows Full-Screen Editing	Yes	Yes
Number of Files/Screen	10	10
Simulates Paper Forms on Screen	Yes	Yes
Max # of Fields Sorted	10	15
Automatic Indexing	Yes	Yes
Add or Change Indexes	Yes	Yes
Splits and Merges Files	Yes	Yes
Provides Query Language	Yes	Yes
Provides Procedural Language	Yes	Yes
ASCII	No	Yes
DIF	No	Yes
Other	Binary	Sylk, 1-2-3
Allows Encryption	No	No
Password Access	No	Yes
Activity Log	No	Yes

11. SUBJECTIVE COMPARISONS

Though the Smart Data Manager and dBASEIII look similar on paper, differences exist. Smart is very user friendly. The novice should feel at ease after one session. If the user makes a mistake when executing a command, the system will tell the user why what he/she did was wrong, and how to do it correctly. When the system is involved with the execution of a lengthy command, it does not print, "wait;" instead it prints, "busy-do not disturb." These touches, plus vivid color and bright screens, make the program enjoyable with which to work. When comparing dBASEIII to Smart's showmanship, it is similar to comparing night and day. dBASEIII does not concern itself with nice looking screen output, or bright colors. dBASEIII has made improvements, however, in user friendliness. The assist command provides menus to help in the completion of common tasks, and the help command can aid in the execution of commands.

The documentation, or manuals, for both of these systems is excellent by comparison to most manuals. They both are easy to follow, and well indexed. Smart is, however, in need of sample application programs. The learning process is greatly hindered when no in-depth examples are available to study.

The ease in learning of the two systems is very difficult

to estimate; the ease or difficulty would depend on past database experience. The true beginner, with no computer experience of any kind, would most likely do better with the Smart Data Manager because there is nothing intimidating about it. Someone with computer experience, but none with databases would probably be just as comfortable with either system. His/her lack of database knowledge would make Smart appealing, but his/her software development experience could make dBASEIII more appealing. The real computer programmer would more than likely prefer dBASEIII. There are no bells or whistles, but it gets the job done in a straight forward fashion.

12. CONCLUSIONS

The purpose of this master's report was to evaluate the Smart Data Manager. As a basis of comparison, one of the most popular DBMSs, dBaseIII, is chosen. The logical design is completed, and next the physical design of the database is done in both languages. The only significant differences between the Smart Data Manager's physical design and that of dBaseIII are in the field types. dBaseIII offers a logical and memo field in addition to the standard fields of date, character, and numeric. The Smart Data Manager offers a counter, an inverted name, a phone, a time, and a social security field type. The adjustment of the data types concludes the physical design and signals the beginning of implementation. Many differences were found when making a comparison of the languages during implementation of the application programs. Generally, the Smart Data Manager requires more interaction with the user. There are times when the user must assume responsibility for segments of the execution of the application programs. The application program language in Smart is sometimes awkward. For example, Smart only defines four variables. Any others that may be needed must be initialized prior to their use. The Smart system provides a means to debug a program that is not available in dBaseIII; This is called single step execu

tion. This is a valuable tool for examining the application program's logic command-by-command. When evaluating the file space requirements, they are found to be very different. The Smart Data Manager files (excluding the application programs) requires 78 files compared to dBaseIII's requirement of only 28 files. Most of this descrepancy is possibly due to Smart's use of predefined key fields. Objectively the two systems are very similar, but subjectively there are many differences. The Smart Data Manager may be better suited for the novice computer user, or someone with no database experience. The application programming language in Smart is somewhat cumbersome. dBaseIII, though requiring more programming knowledge, is much more powerful. When deciding which of the two software systems to choose, the following should be considered: the file space requirements of the databases to be used, the execution time needed, and the experience of the users.

BIBLIOGRAPHY

Burch, John G. Jr., Felix R. Straler, Gary Grudnitski. Information Systems: Theory and Practice, 2nd ed. John Wiley and Sons, New York, New York, 1979.

Byers, Robert A. Everyman's Database Primer, Ashton-Tate Publishing Group, Culver City, California, 1984.

dBASEIII Manual, Ashton-Tate Publishing Group, Culver City, California, 1984.

Gabel, D. "Organizational Pursuit," PC Week (Buyers' Guide), (May 7, 1985)

Gare, Chris and Trish Sarson Structured Systems Analysis, Prentice Hall, Inc., Englewood Cliffs, New Jersey, 1979.

Kindred, Alton R. Data Systems and Management, 2nd ed. Prentice Hall, Inc., Englewood Clifs, New Jersey, 1980.

Krajewski, R. "Database Types", BYTE, (October 1984), pp.137.

Kroenke, David Database Processing, 2nd ed. SRA, Inc., Chicago, 1983.

Schwartz, Andrew N. "Get Smart To Fight Chaos," PC Week, (December 25, 1984)

Smart Data Manager Manual, Innovative Software, Inc., Overland Park, Kansas, 1984.

Urschel, W. "Data Base Management Software," Personal Computing, (March 1985), pp. 241-249.

APPENDIX A

PSEUDO CODE

MEMBERSHIP FILES: NEW MEMBER

```
ENTER DATA INTO BLANK RECORD
INVOLVEMENT
   NOTE NOINV (NUMBER OF AREAS INVOLVED)
   NEW RECORD IN INVOLVE FILE FOR EACH
      PLACE LNAME, FNAME, MNAME, AND INT INTO
      BLANK INVOLVE RECORD
INTEREST AND SKILLS
   NOTE NOINT (NUMBER OF AREAS INTERESTED)
   NEW RECORD IN INTEREST FILE FOR EACH
      PLACE LNAME, FNAME, MNAME, AND INT INTO
      BLANK INTEREST RECORD
CALCULATE AGE
   COMPARE DATE WITH DOB
      IF MM OF DOB >= MM OF DATE
         AGE = YY(DATE)-YY(DOB)
      ELSE
         AGE = YY(DATE)-YY(DOB)-1
      ENDIF
      IF AGE >= 65
         SET SENIOR TO "Y"
      ENDIF
SET ORIG = DATE
EXPIRE = MM/DD/YY(DATE)  (YY=YY+1)
```

MEMBERSHIP FILES: RENEWAL / NON-EXPIRED

```
VERIFY EXISTING DATA VIA PRINTOUT OF MEMBERSHIP,
      INTEREST,INVOLVE
ENTER DATE
EXPIRE = OLD EXPIRE(YY)+1
IF INTERESTS CHANGED
   IF THE CHANGE IS AN INCREASE IN AREAS
      .CREATE RECORD IN INTEREST FILE FOR EACH NEW INTEREST
       CONTAINING LNAME, FNAME, MNAME, AND INT
      .CHANGE NOINT
   IF THE CHANGE IS A DECREASE IN AREAS
      .LOCATE RECORD IN INTEREST FILE WITH THE LNAME,
        FNAME, INT TO BE ELIMINATED
      .ERASE THE RECORD IN INTEREST
      .CHANGE NOINT
IF INVOLVEMENT HAS CHANGED
   IF THE CHANGE IS AN INCREASE IN THE AREAS
      .CREATE RECORD IN INVOLVE FOR EACH NEW INVOLVEMENT
       CONTAINING LNAME, FNAME, MNAME, AND INV
      .CHANGE THE NOINV
   IF THE CHANGE IS A DECREASE IN AREAS
      .LOCATE RECORD IN INVOLVE WITH THE LNAME, FNAME, MNAME
       AND INV TO BE ELIMINATED
      .ERASE THE RECOD IN INVOLVE
      .CHANGE NOINV
ENTER NEW ACNOTE AND ACDATE
```

MEMBERSHIP FILES: RENEWAL / EXPIRED

```
VERIFY EXISTING DATA VIA
   MEMBERSHIP FILE PRINTOUT
   INTEREST FILE PRINTOUT
   INVOLVEMENT FILE PRINTOUT
ENTER DATE
EXPIRE = DATE(YY)+1
IF INTERESTS CHANGED
   IF THE CHANGE IS AN INCREASE IN AREAS
      .CREATE RECORD IN INTEREST FILE FOR EACH NEW INTEREST
       CONTAINING LNAME, FNAME, MNAME, AND INT
      .CHANGE NOINT
   IF THE CHANGE IS A DECREASE IN AREAS
      .LOCATE RECORD IN INTEREST FILE WITH THE LNAME,
       FNAME, MNAME, MNAME, INT TO BE ELIMINATED
      .ERASE THE RECORD IN INTEREST
      .CHANGE NOINT
IF INVOLVEMENT HAS CHANGED
   IF THE CHANGE IS AN INCREASE IN AREAS
      .CREATE RECORD IN INVOLVE FOR EACH NEW
       INVOLVEMENT CONTAINING LNAME,FNAME, MNAME, AND INV
      .CHANGE THE NOINV
   IF THE CHANGE IS A DECREASE IN AREAS
      .LOCATE RECORD IN INVOLVE WITH THE LNAME, FNAME, AND
       INV TO BE ELIMINATED
      .ERASE THE RECORD IN INVOLVE
      .CHANGE NOINV
ENTER NEW ACNOTE AND ACDATE
REMOVE FLAG
```

LIST OF "ABOUT" DUE (DONE ONE MONTH PRIOR)

```
CLEAR ABOUT FILE
SEARCH MEMBERSHIP FILES FOR RECORDS WITH
   MM(EXPIRE) = NEXT MONTH
   YY(EXPIRE) = THIS YEAR-1
IF FIND RECORD MEETING CRITERIA
   .COPY LNAME, FNAME, MNAME, STREET, CITYST, EXPIRE
    TO ABOUT FILE
NEXT RECORD
ONCE EOF IS REACHED
   .SORT ABOUT IN INCREASING ORDER ON LNAME THEN FNAME
   .THEN PRINT
       FNAME LNAME
       STREET
       CITYST
       EXPIRE
   FOR EACH RECORD
```

FLAG PAST DUE (MONTHLY)

```
IF EXPIRE < DATE
   FLAG RECORD FOR DELETION
CONTINUE UNTIL EOF
```

VOTER (DONE ANNUALLY DURING ELECTIONS)

```
LOCATE VOTER'S RECORD
ENTER CURRENT YEAR INTO VOTE
```

PROSPECT FILE : NEW DONOR

```
ENTER DATA IN PROSPECT
IF MAKING A DONATION
   CREATE A RECORD IN GIVEN
   ENTER DATA IN GIVEN
   IF LETTER AND RECEIPT SENT
      SET LETTER TO "Y" IN GIVEN
      SET RECEIPT TO "Y" IN GIVEN
IF MAKING A PLEDGE
   SET PLDG TO "Y" IN PROSPECT
   CREATE A RECORD IN PLEDGE
   ENTER DATA GIVEN
   CALCULATE OUTSTD: (IF 1ST INSTALLMENT PAID)
      AMT-INSTALLMENT = OUTSTD
      CREATE AN INSTALL RECORD
      ENTER DATA GIVEN
      ELSE OUTSD = AMT
```

PROSPECT FILE: EXISTING DONOR DONATION OR PLEDGE OR INSTALLMENT

```
LOCATE RECORD IN PROSPECT FILE
IF MAKING A DONATION
   CREATE A RECORD IN GIVEN
   ENTER DATA
   IF LETTER AND RECEIPT SENT
      SET LETTER TO "Y"
      SET RECEIPT TO "Y"
IF MAKING AN INSTALLMENT PAYMENT
   CREATE RECORD IN INSTALL
   ENTER DATA
   STORE INSTL VALUE
   LOCATE RECORD IN PLEDGE
   CALCULATE OUTSTD IN PLEDGE
      AMT-INSTL = OUTSTD
   IF OUTSTD <= 0
      SET PLDG IN PROSPECT TO "N"
IF MAKING A PLEDGE
   SET PLDG TO "Y" IN PROSPECT
   CREATE A RECORD IN PLEDGE
   ENTER DATA GIVEN
   IF 1ST INSTALLMENT PAID
      CALCULATE OUTSTD: AMT-INSTL = OUTSTD
      CREATE INSTALL RECORD
      ENTER DATA
   ELSE
      OUTSTD = AMT
```

APPENDIX B

DATA DICTIONARY

DATA DICTIONARY

ACDATE
DATE OF ISSUE OF ACTIVITY CARD
TYPE: DATE
FORMAT
LENGTH = 8

ACNOTE
NOTES WHAT TYPE OF ACTIVITY CARD WAS ISSUED
TYPE: CHARACTER
VALUES: FAC = FAMILY ACTIVITY CARD
AC = INDIVIDUAL ACTIVITY CARD
LENGTH = 3

ADULT
TYPE: DATABASE RECORD STRUCTURE OF ADULT MEMBER
LNAME
FNAME
MNAME
SEX
DATE
STREET
CITYST
ZONE
HTELE
WTELE
OCCUP
PLOFWK
AGE
EDUC
STATUS
NOCH
DOB
YRSAL
RACE
SENIOR
VOTE
ORIG
EXPIRE
ACDATE
ACNOTE

AGE
AGE OF INDIVIDUAL
TYPE: NUMERIC
VALUES: 0-99
LENGTH = 2

```
AMT
      DOLLAR AMOUNT GIVEN OR PLEDGED TO YWCA
      TYPE:  NUMERIC
      LENGTH = 9

APLTOT
      VOLUNTEER HOURS FOR APRIL
      TYPE:  NUMERIC
      VALUES:  0-999
      LENGTH = 3

ARTS/CRAFTS
      INDICATES INTEREST OF MEMBER
      TYPE:  CHARACTER
      LENGTH = 1

AUGTOT
      VOLUNTEER HOURS FOR AUGUST
      TYPE:  NUMERIC
      VALUES:  0-999
      LENGTH = 3

BOARD
      INDICATES INVOLVEMENT AT YWCA
      TYPE:  CHARACTER
      LENGTH = 1

CHILDREN
      INDICATES INTEREST OF MEMBER
      TYPE:  CHARACTER
      LENGTH = 1

CITYST
      CITY, ST ZIP
      TYPE:  CHARACTER
      LENGTH = 20

CLASS
      INDICATES INVOLVEMENT AT YWCA
      TYPE:  CHARACTER
      LENGTH = 1

CLERICAL
      INDICATES INTEREST OF MEMBER
      TYPE:  CHARACTER
      LENGTH = 1
```

```
CLUB
      INDICATES INVOLVEMENT AT YWCA
      TYPE:  CHARACTER
      LENGTH = 1

COMM
      COMMENTS
      TYPE:  CHARACTER
      LENGTH = 25

COMMITTEE
      INDICATES INVOLVEMENT AT YWCA
      TYPE:  CHARACTER
      LENGTH = 1

CONNAME
      NAME OF PERSON TO CONTACT ABOUT DONATION
      TYPE:  CHARACTER
      LENGTH = 20

CONPREF
      WHERE CONTACT PERSON PREFERS TO BE CONTACTED
      TYPE:  CHARACTER
      VALUES:  H = HOME
               W = WORK
      LENGTH = 1

CONTACT
      PERSON (OTHER THAN PARENTS) FOR EMERGENCIES
      TYPE:  CHARACTER
      LENGTH = 20

CONTEL
      TELEPHONE NUMBER FOR CONTACT
      TYPE:  CHARACTER
      LENGTH = 14

COOKING/BAKING
      INDICATES INTEREST OF MEMBER
      TYPE:  CHARACTER
      LENGTH = 1

DATE
      CURRENT DATE
      TYPE:  DATE
      LENGTH = 8
```

```
DECTOT
      VOLUNTEER HOURS FOR DECEMBER
      TYPE:  NUMERIC
      LENGTH = 9

DOB
      DATE OF BIRTH
      TYPE:  DATE
      LENGTH = 8

DOC
      NAME OF FAMILY DOCTOR
      TYPE:  CHARACTER
      LENGTH = 20

DOCTEL
      TELEPHONE NUMBER OF FAMILY DOCTOR
      TYPE:  CHARACTER
      LENGTH = 14

DONEBY
      PERSON WHO RECEIVED DONATION FOR YWCA
      TYPE:  CHARACTER
      LENGTH = 15

DONOR
      NAME OF INDIVIDUAL OR FIRM MAKING DONATION
      TYPE:  CHARACTER
      LENGTH = 20

EDUC
      EDUCATIONAL BACKGROUND
      TYPE:  CHARACTER
      VALUES:  A = ARTS
               C = CLERICAL
               E = EDUCATIONAL
               L = LEGAL
               M = MEDICAL
               S = SERVICE
               T = TECHNICAL
               O = OTHER
               N = NOT APPLICABLE
      LENGTH = 1

EXPIRE
      DATE THAT MEMBERSHIP EXPIRES
      TYPE:  DATE
      LENGTH = 8
```

```
FEBTOT
      VOLUNTEER HOURS FOR FEBRUARY
      TYPE:  NUMERIC
      VALUES:  0-999
      LENGTH = 3

FNAME
      INDIVIDUAL'S FIRST NAME
      TYPE:  CHARACTER
      LENGTH = 10

GIVEN
      TYPE:  DATABASE RECORD STRUCTURE FOR DONATION
            DONOR
            DATE
            AMT
            RECPT
            LETTER
            PURPOSE
            DONEBY

GUARD
      FIRST AND LAST NAMES OF PARENTS OR GUARDIANS
      TYPE: CHARACTER
      LENGTH = 35

HTELE
      HOME TELEPHONE NUMBER
      TYPE:  CHARACTER
      LENGTH = 14

INTEREST
      TYPE:  DATABASE RECORD STRUCTURE FOR MEMBER INTEREST
            LNAME
            FNAME
            MNAME
            ARTS/CRAFTS
            CLERICAL
            CHILDREN
            COOKING/BAKING
            P.E./SWIM/DANCE
            TELEPHONING
            UPKEEP/CLEANING
            VOLUNTR

INVOLVE
      TYPE:  DATABASE RECORD STRUCTURE FOR INVOLVEMENT
            LNAME
            FNAME
```

```
          MNAME
          BOARD
          COMMITTEE
          CLUB
          CLASS
          RECREATION
          VOLUNTR

JANTOT
     VOLUNTEER HOURS FOR JANUARY
     TYPE:  NUMERIC
     VALUES:  0-999
     LENGTH = 3

JULTOT
     VOLUNTEER HOURS FOR JULY
     TYPE:  NUMERIC
     VALUES:  0-999
     LENGTH = 3

JUNTOT
     VOLUNTEER HOURS FOR JUNE
     TYPE:  NUMERIC
     VALUES:  0-999
     LENGTH = 3

KNOWN
     WHO KNOWS PROSPECTIVE DONOR WELL
     TYPE:  CHARACTER
     LENGTH = 20

LETTER
     DENOTES WHETHER LETTER SENT TO DONOR
     TYPE:  CHARACTER
     LENGTH = 1

LNAME
     INDIVIDUAL'S LAST NAME
     TYPE:  CHARACTER
     LENGTH = 20

MARTOT
     VOLUNTEER HOURS FOR MARCH
     TYPE:  NUMERIC
     VALUES:  0-999
     LENGTH = 3
```

```
MAYTOT
      VOLUNTEER HOURS FOR MAY
      TYPE:  NUMERIC
      VALUES:  0-999
      LENGTH = 3

MINOR
      TYPE:  DATABASE RECORD STRUCTURE OF MINORS
            LNAME
            FNAME
            MNAME
            SEX
            DATE
            STREET
            CITYST
            ZONE
            HTELE
            PLOFWK
            AGE
            DOB
            RACE
            GUARD
            SCHOOL
            DOC
            DOCTEL
            CONTACT
            CONTEL
            ORIG
            EXPIRE
            ACDATE
            ACNOTE

MNAME
      INDIVIDUAL'S MIDDLE NAME
      TYPE:  CHARACTER
      LENGTH = 10

NOCH
      NUMBER OF CHILDREN
      TYPE:  NUMERIC
      LENGTH = 2

NOVTOT
      VOLUNTEER HOURS FOR NOVEMBER
      TYPE:  NUMERIC
      VALUES:  0-999
      LENGTH = 3
```

```
OCCUP
      OCCUPATION OF INDIVIDUAL
      TYPE:  CHARACTER
      LENGTH = 10

OCTTOT
      VOLUNTEER HOURS FOR OCTOBER
      TYPE:  NUMERIC
      VALUES:  0-999
      LENGTH = 3

ORIG
      ORIGINAL DATE OF MEMBERSHIP
      TYPE:  DATE
      LENGTH = 8

OTHERS
      SUPPORT OF OTHER ORGANIZATIONS
      TYPE:  CHARACTER
      LENGTH = 25

OUSTD
      AMOUNT OUTSTANDING ON A DONATION PLEDGE
      TYPE:  NUMERIC
      LENGTH = 9

PAID
      AMOUNT OF INSTALLMENT DONATION PAID
      TYPE:  NUMERIC
      LENGTH = 9

PAMT
      AMOUNT PLEDGED
      TYPE:  NUMERIC
      LENGTH = 9

P.E./SWIM/DANCE
      INDICATES INTEREST OF MEMBER
      TYPE:  CHARACTER
      LENGTH = 1

PLEDGE
      TYPE:  DATABASE RECORD STRUCTURE FOR PLEDGED DONATION
            DONOR
            AMT
            PURPOSE
            DATE
            OUTSTD
            DONEBY
```

```
            PAID

POUT
      AMOUNT OF PLEDGE OUTSTANDING
      TYPE:  NUMERIC
      LENGTH = 9

PLOFWK
      PLACE OF WORK
      TYPE:  CHARACTER
      LENGTH = 15

PROINT
      INTERESTS OF PROSPECTIVE DONOR
      TYPE:  CHARACTER
      LENGTH = 20

PROSAGE
      AGE OF PROSPECTIVE DONOR
      TYPE:  NUMERIC
      LENGTH = 2

PROSED
      EDUCATION OF PROSPECTIVE DONR
      TYPE:  CHARACTER
      LENGTH = 10

PROSPECT
      TYPE:  DATABASE RECORD STRUCTURE OF PROSPECTIVE DONOR
            DONOR
            CONNAME
            TITLE
            CONPREF
            PROSAGE
            PROSED
            PROSTUD
            PROREL
            SPAGE
            SPSTUD
            SPNAME
            NOCH
            PROINT
            COMM
            OTHERS
            KNOWN
            SOLIC
            PAMT
            POUT
```

```
PROREL
      RELIGION AFFILIATION OF PROSPECTIVE DONOR
      TYPE:  CHARACTER
      LENGTH = 10

PROSTUD
      FIELD OF STUDY OF PROSPECTIVE DONOR
      TYPE:  CHARACTER
      LENGTH = 10

PURPOSE
      FOR WHAT PURPOSE DONATION TO BE USED
      TYPE:  CHARACTER
      LENGTH = 15

RACE
      RACIAL BACKGROUND
      TYPE:  CHARACTER
      VALUES:  A = ASIAN AMERICAN
               B = BLACK
               H = HISPANIC
               N = NATIVE AMERICAN
               W = WHITE
               O = OTHER
      LENGTH = 1

RECPT
      DENOTES WHETHER RECEIPT SENT FOR DONATION
      TYPE:  CHARACTER
      LENGTH = 1

RECREATION
      INDICATES INVOLVEMENT AT YWCA
      TYPE:  CHARACTER
      LENGTH = 1

SCHOOL
      SCHOOL ATTENDING
      TYPE:  CHARACTER
      LENGTH = 10

SENIOR
      INDICATES WHETHER SENIOR (OVER 65)
      TYPE:  CHARACTER
      LENGTH = 1

SEPTOT
      VOLUNTEER HOURS FOR SEPTEMBER
      TYPE:  NUMERIC
```

```
      VALUES:  0-999
      LENGTH = 3

SEX
      GENDER OF INDIVIDUAL
      TYPE:  CHARACTER
      VALUES:  M = MALE
               F = FEMALE
      LENGTH = 1

SOLIC
      RECOMMENDED SOLICITOR FOR DONATION
      TYPE:  CHARACTER
      LENGTH = 15

SPED
      EDUCATION OF PROSPECTIVE DONOR'S SPOUSE
      TYPE:  CHARACTER
      LENGTH = 10

SPNAME
      PROSPECTIVE DONOR'S SPOUSE'S NAME
      TYPE:  CHARACTER
      LENGTH = 10

SPSTUD
      FIELD OF STUDY OF PROSPECTIVE DONOR'S SPOUSE
      TYPE:  CHARACTER
      LENGTH = 10

STATUS
      MARRIAGE STATUS
      TYPE:  CHARACTER
      VALUES:  D = DIVORCED
               M = MARRIED
               S = SINGLE
               W = WIDOWED
      LENGTH = 1

STREET
      STREET ADDRESS
      TYPE:  CHARACTER
      LENGTH = 20

TELEPHONING
      INDICATES MEMBER INTEREST
      TYPE: CHARACTER
      LENGTH = 1
```

```
TITLE
      TITLE OF CONTACT PERSON FOR DONATION
      TYPE:  CHARACTER
      LENGTH = 10

TOTAL
      INDICATES LAST MONTH VOLUNTEER GAVE TIME
      TYPE:  NUMERIC
      LENGTH = 2

UPKEEP/CLEANING
      INDICATES MEMBER INTEREST
      TYPE:  CHARACTER
      LENGTH = 1

VOLUNTR
      INDICATES MEMBER INTEREST AND/OR INVOLVEMENT
      TYPE:  CHARACTER
      LENGTH = 1

VOLUNTEER
TYPE:  DATABASE RECORD STRUCTURE FOR VOLUNTEER
            LNAME
            FNAME
            MNAME
            TOTAL
            JANTOT
            FEBTOT
            MARTOT
            APLTOT
            MAYTOT
            JUNTOT
            JULTOT
            AUGTOT
            SEPTOT
            OCTTOT
            NOVTOT
            DECTOT

VOTE
      LAST YEAR MEMBER VOTED
      TYPE:  CHARACTER
      LENGTH = 4

WTELE
      WORK TELEPHONE NUMBER
      TYPE:  CHARACTER
      LENGTH = 14
```

```
YRSAL
      YEAR MOVED TO SALINA
      TYPE:  CHARACTER
      LENGTH = 4


XPIRAD
      TYPE:  DATABASE RECORD STRUCTURE OF EXPIRED ADULT MEMBER
            LNAME
            FNAME
            MNAME
            SEX
            DATE
            STREET
            CITYST
            ZONE
            HTELE
            WTELE
            OCCUP
            PLOFWK
            AGE
            EDUC
            STATUS
            NOCH
            DOB
            YRSAL
            RACE
            SENIOR
            VOTE
            ORIG
            EXPIRE
            ACDATE
            ACNOTE


XPIRMIN
      TYPE:  DATABASE RECORD STRUCTURE OF EXPIRED MINORS
            LNAME
            FNAME
            MNAME
            SEX
            DATE
            STREET
            CITYST
            ZONE
            HTELE
            PLOFWK
            AGE
            DOB
```

RACE
GUARD
SCHOOL
DOC
DOCTEL
CONTACT
CONTEL
ORIG
EXPIRE
ACDATE
ACNOTE

ZONE
AREA ZONE OF SALINA
TYPE: CHARACTER
VALUES: 1 - 7
LENGTH = 1

APPENDIX C

SMART DATA STRUCTURES

SMART DATA STRUCTURES

Database Name: ADULT

Field Number	Field Name	Type	Length
1	NAME	INVERTED	30
2	STREET	CHARACTER	20
3	CITYST	CHARACTER	20
4	ZONE	CHARACTER	1
5	HTELE	PHONE	14
6	OCCUP	CHARACTER	15
7	PLOFWK	CHARACTER	15
8	WTELE	PHONE	14
9	DATE	DATE	8
10	SEX	CHARACTER	1
11	AGE	NUMERIC	2
12	EDUC	CHARACTER	1
13	STATUS	CHARACTER	1
14	NOCH	NUMERIC	2
15	DOB	DATE	8
16	YRSAL	CHARACTER	4
17	RACE	CHARACTER	1
18	SENIOR	LOGICAL	1
19	VOTE	CHARACTER	4
20	ORIG	DATE	8
21	EXPIRE	DATE	8
22	ACDATE	DATE	8
23	ACNOTE	CHARACTER	3
		TOTAL:	189

Database Name: GIVEN

Field Number	Field Name	Type	Length
1	DONOR	CHARACTER	20
2	DATE	DATE	8
3	AMT	NUMERIC	9
4	RECPT	CHARACTER	1
5	LETTER	CHARACTER	1
6	PURPOSE	CHARACTER	15
7	DONEBY	CHARACTER	15
		TOTAL:	69

Database Name: INTEREST

Field Number	Field Name	Type	Length
1	NAME	INVERTED	30
2	ARTS/CRAFTS	CHARACTER	1
3	CLERICAL	CHARACTER	1
4	CHILDREN	CHARACTER	1
5	COOKING/BAKING	CHARACTER	1
6	P.E./SWIM/DANCE	CHARACTER	1
7	TELEPHONING	CHARACTER	1
8	UPKEEP/CLEANING	CHARACTER	1
9	VOLUNTR	CHARACTER	1
		TOTAL:	38

Database Name: INVOLVE

Field Number	Field Name	Type	Length
1	NAME	INVERTED	30
2	BOARD	CHARACTER	1
3	COMMITTEE	CHARACTER	1
4	CLASS	CHARACTER	1
5	CLUB	CHARACTER	1
6	RECREATION	CHARACTER	1
7	VOLUNTR	CHARACTER	1
		TOTAL:	36

Database Name: MINOR

Field Number	Field Name	Type	Length
1	NAME	INVERTED	30
2	STREET	CHARACTER	20
3	CITYST	CHARACTER	20
4	ZONE	CHARACTER	1
5	HTELE	PHONE	14
6	PLOFWK	CHARACTER	15
7	DATE	DATE	8
8	AGE	NUMERIC	2
9	SEX	CHARACTER	1
10	DOB	DATE	14
11	RACE	CHARACTER	1
12	GUARD	CHARACTER	30
13	SCHOOL	CHARACTER	10
14	DOC	CHARACTER	20
15	DOCTEL	PHONE	14
16	CONTACT	CHARACTER	30
17	CONTEL	PHONE	14
18	ORIG	DATE	8
19	EXPIRE	DATE	8
20	ACDATE	DATE	8
21	ACNOTE	CHARACTER	3
		TOTAL:	271

Database Name: PLEDGES

Field Number	Field Name	Type	Length
1	DONOR	CHARACTER	20
2	AMT	NUMERIC	9
3	PURPOSE	CHARACTER	15
4	DATE	DATE	14
5	PAID	NUMERIC	9
6	OUTSTD	NUMERIC	9
7	DONEBY	CHARACTER	20
		TOTAL:	96

Database Name: PROSPECT

Field Number	Field Name	Type	Length
1	DONOR	CHARACTER	30
2	CONNAME	CHARACTER	20
3	TITLE	CHARACTER	10
4	STREET	CHARACTER	20
5	CITYST	CHARACTER	20
6	CONPREF	CHARACTER	1
7	PROSAGE	NUMERIC	2
8	PROSED	CHARACTER	10
9	PROSTUD	CHARACTER	10
10	PROREL	CHARACTER	10
11	SPNAME	CHARACTER	10
12	SPAGE	NUMERIC	2
13	SPED	CHARACTER	10
14	SPSTUD	CHARACTER	10
15	NOCH	NUMERIC	2
16	PROINT	CHARACTER	20
17	COMM	CHARACTER	25
18	OTHERS	CHARACTER	25
19	KNOWN	CHARACTER	20
20	SOLIC	CHARACTER	15
21	PAMT	NUMERIC	9
22	POUT	NUMERIC	9
		TOTAL:	190

Database Name: VOLUNTEER

Field Number	Field Name	Type	Length
1	NAME	INVERTED	30
2	TOTAL	NUMERIC	6
3	MONTH	NUMERIC	2
4	JANTOT	NUMERIC	3
5	FEBTOT	NUMERIC	3
6	MARTOT	NUMERIC	3
7	APLTOT	NUMERIC	3
8	MAYTOT	NUMERIC	3
9	JUNTOT	NUMERIC	3
10	JULTOT	NUMERIC	3
11	AUGTOT	NUMERIC	3
12	SEPTOT	NUMERIC	3
13	OCTTOT	NUMERIC	3
14	NOVTOT	NUMERIC	3
15	DECTOT	NUMERIC	3
		TOTAL:	74

Database Name: XPIRAD

Field Number	Field Name	Type	Length
1	NAME	INVERTED	30
2	STREET	CHARACTER	20
3	CITYST	CHARACTER	20
4	ZONE	CHARACTER	1
5	HTELE	PHONE	14
6	OCCUP	CHARACTER	15
7	PLOFWK	CHARACTER	15
8	WTELE	PHONE	14
9	DATE	DATE	8
10	SEX	CHARACTER	1
11	AGE	NUMERIC	2
12	EDUC	CHARACTER	1
13	STATUS	CHARACTER	1
14	NOCH	NUMERIC	2
15	DOB	DATE	8
16	YRSAL	CHARACTER	4
17	RACE	CHARACTER	1
18	SENIOR	LOGICAL	1
19	VOTE	CHARACTER	4
20	ORIG	DATE	8
21	EXPIRE	DATE	8
22	ACDATE	DATE	8
23	ACNOTE	CHARACTER	3
		TOTAL:	189

Database Name: XPIRMIN

Field Number	Field Name	Type	Length
1	NAME	INVERTED	30
2	STREET	CHARACTER	20
3	CITYST	CHARACTER	20
4	ZONE	CHARACTER	1
5	HTELE	PHONE	14
6	PLOFWK	CHARACTER	15
7	DATE	DATE	8
8	AGE	NUMERIC	2
9	SEX	CHARACTER	1
10	DOB	DATE	14
11	RACE	CHARACTER	1
12	GUARD	CHARACTER	30
13	SCHOOL	CHARACTER	10
14	DOC	CHARACTER	20
15	DOCTEL	PHONE	14
16	CONTACT	CHARACTER	30
17	CONTEL	PHONE	14
18	ORIG	DATE	8
19	EXPIRE	DATE	8
20	ACDATE	DATE	8
21	ACNOTE	CHARACTER	3
		TOTAL:	271

APPENDIX D

dBASEIII DATA STRUCTURES

dBASEIII DATA STRUCTURES

Database Name: ADULT

Field Number	Field Name	Type	Length
1	LNAME	CHARACTER	20
2	FNAME	CHARACTER	10
3	MNAME	CHARACTER	10
4	STREET	CHARACTER	20
5	CITYST	CHARACTER	20
6	ZONE	CHARACTER	1
7	HTELE	CHARACTER	14
8	OCCUP	CHARACTER	15
9	PLOFWK	CHARACTER	15
10	WTELE	CHARACTER	14
11	DATE	DATE	8
12	SEX	CHARACTER	1
13	AGE	NUMERIC	2
14	EDUC	CHARACTER	1
15	STATUS	CHARACTER	1
16	NOCH	NUMERIC	2
17	DOB	DATE	8
18	YRSAL	CHARACTER	4
19	RACE	CHARACTER	1
20	SENIOR	LOGICAL	1
21	VOTE	CHARACTER	4
22	ORIG	DATE	8
23	EXPIRE	DATE	8
24	ACDATE	DATE	8
25	ACNOTE	CHARACTER	3
		TOTAL:	199

Database Name: GIVEN

Field Number	Field Name	Type	Length
1	DONOR	CHARACTER	20
2	DATE	DATE	8
3	AMT	NUMERIC	9
4	RECPT	LOGICAL	1
5	LETTER	LOGICAL	1
6	PURPOSE	CHARACTER	15
7	DONEBY	CHARACTER	15
		TOTAL:	69

Database Name: INTEREST

Field Number	Field Name	Type	Length
1	LNAME	CHARACTER	20
2	FNAME	CHARACTER	10
3	MNAME	CHARACTER	10
4	ARTS	LOGICAL	1
5	CLERICAL	LOGICAL	1
6	CHILDREN	LOGICAL	1
7	COOKING	LOGICAL	1
8	PE	LOGICAL	1
9	TELE	LOGICAL	1
10	UPKEEP	LOGICAL	1
11	VOLUNTR	LOGICAL	1
		TOTAL:	48

Database Name: INVOLVE

Field Number	Field Name	Type	Length
1	LNAME	CHARACTER	20
2	FNAME	CHARACTER	10
3	MNAME	CHARACTER	10
4	BOARD	LOGICAL	1
5	COMMITTEE	LOGICAL	1
6	CLASS	LOGICAL	1
7	CLUB	LOGICAL	1
8	RECREATION	LOGICAL	1
9	VOLUNTR	LOGICAL	1
		TOTAL:	46

Database Name: MINOR

Field Number	Field Name	Type	Length
1	LNAME	CHARACTER	20
2	FNAME	CHARACTER	10
3	MNAME	CHARACTER	10
4	STREET	CHARACTER	20
5	CITYST	CHARACTER	20
6	ZONE	CHARACTER	1
7	HTELE	CHARACTER	14
8	PLOFWK	CHARACTER	15
9	DATE	DATE	8
10	AGE	NUMERIC	2
11	SEX	CHARACTER	1
12	DOB	DATE	14
13	RACE	CHARACTER	1
14	GUARD	CHARACTER	30
15	SCHOOL	CHARACTER	10
16	DOC	CHARACTER	20
17	DOCTEL	CHARACTER	14
18	CONTACT	CHARACTER	30
19	CONTEL	CHARACTER	14
20	ORIG	DATE	8
21	EXPIRE	DATE	8
22	ACDATE	DATE	8
23	ACNOTE	CHARACTER	3
		TOTAL:	281

Database Name: PLEDGES

Field Number	Field Name	Type	Length
1	DONOR	CHARACTER	20
2	AMT	NUMERIC	9
3	PURPOSE	CHARACTER	20
4	DATE	DATE	14
5	PAID	NUMERIC	9
6	OUTSTD	NUMERIC	9
7	DONEBY	CHARACTER	20
		TOTAL:	110

Database Name: PROSPECT

Field Number	Field Name	Type	Length
1	DONOR	CHARACTER	30
2	CONNAME	CHARACTER	20
3	TITLE	CHARACTER	10
4	STREET	CHARACTER	20
5	CITYST	CHARACTER	20
6	CONPREF	CHARACTER	1
7	PROSAGE	NUMERIC	2
8	PROSED	CHARACTER	10
9	PROSTUD	CHARACTER	10
10	PROREL	CHARACTER	10
11	SPNAME	CHARACTER	10
12	SPAGE	NUMERIC	2
13	SPED	CHARACTER	10
14	SPSTUD	CHARACTER	10
15	NOCH	NUMERIC	2
16	PROINT	CHARACTER	20
17	COMM	MEMO	10
18	OTHERS	MEMO	10
19	KNOWN	CHARACTER	20
20	SOLIC	CHARACTER	15
21	PAMT	NUMERIC	9
22	POUT	NUMERIC	9
		TOTAL:	160

Database Name: VOLUNTEER

Field Number	Field Name	Type	Length
1	LNAME	CHARACTER	20
2	FNAME	CHARACTER	10
3	MNAME	CHARACTER	10
4	TOTAL	NUMERIC	6
5	JANTOT	NUMERIC	3
6	FEBTOT	NUMERIC	3
7	MARTOT	NUMERIC	3
8	APLTOT	NUMERIC	3
9	MAYTOT	NUMERIC	3
10	JUNTOT	NUMERIC	3
11	JULTOT	NUMERIC	3
12	AUGTOT	NUMERIC	3
13	SEPTOT	NUMERIC	3
14	OCTTOT	NUMERIC	3
15	NOVTOT	NUMERIC	3
16	DECTOT	NUMERIC	3
		TOTAL:	82

Database Name: XPIRAD

Field Number	Field Name	Type	Length
1	LNAME	CHARACTER	20
2	FNAME	CHARACTER	10
3	MNAME	CHARACTER	10
4	STREET	CHARACTER	20
5	CITYST	CHARACTER	20
6	ZONE	CHARACTER	1
7	HTELE	CHARACTER	14
8	OCCUP	CHARACTER	15
9	PLOFWK	CHARACTER	15
10	WTELE	CHARACTER	14
11	DATE	DATE	8
12	SEX	CHARACTER	1
13	AGE	NUMERIC	2
14	EDUC	CHARACTER	1
15	STATUS	CHARACTER	1
16	NOCH	NUMERIC	2
17	DOB	DATE	8
18	YRSAL	CHARACTER	4
19	RACE	CHARACTER	1
20	SENIOR	LOGICAL	1
21	VOTE	CHARACTER	4
22	ORIG	DATE	8
23	EXPIRE	DATE	8
24	ACDATE	DATE	8
25	ACNOTE	CHARACTER	3
		TOTAL:	199

Database Name: XPIRMIN

Field Number	Field Name	Type	Length
1	LNAME	CHARACTER	20
2	FNAME	CHARACTER	10
3	MNAME	CHARACTER	10
4	STREET	CHARACTER	20
5	CITYST	CHARACTER	20
6	ZONE	CHARACTER	1
7	HTELE	CHARACTER	14
8	PLOFWK	CHARACTER	15
9	DATE	DATE	8
10	AGE	NUMERIC	2
11	SEX	CHARACTER	1
12	DOB	DATE	14
13	RACE	CHARACTER	1
14	GUARD	CHARACTER	30
15	SCHOOL	CHARACTER	10
16	DOC	CHARACTER	20
17	DOCTEL	CHARACTER	14
18	CONTACT	CHARACTER	30
19	CONTEL	CHARACTER	14
20	ORIG	DATE	8
21	EXPIRE	DATE	8
22	ACDATE	DATE	8
23	ACNOTE	CHARACTER	3
		TOTAL:	281

APPENDIX E

SMART APPLICATION PROGRAMS

NEW MEMBERSHIP APPLICATION PROGRAM

```
UNLOAD ALL
LABEL START
INPUT TEXT1 ENTER A FOR ADULT OR M FOR MINOR
IF TEXT1="A" THEN JUMP ADULT
IF TEXT1="M" THEN JUMP MINOR
WAIT 5 NOT A CHOICE! TRY AGAIN
JUMP START
LABEL ADULT
LOAD ADULT SCREEN MINORS
JUMP COMMON
LABEL MINOR
LOAD MINOR SCREEN MINOR
LABEL COMMON
SPLIT VERTICAL 2 45
GOTO WINDOW 2
LOAD INTBACK SCREEN INTBACK2
SPLIT HORIZANTAL 12 47
GOTO WINDOW 3
LOAD INVBACK SCREEN INVBACK
GOTO WINDOW 1
KEY UPDATE
ORDER KEY [1]
GOTO WINDOW 2
KEY UPDATE
ORDER KEY [1]
GOTO WINDOW 3
KEY UPDATE
ORDER KEY [1]
GOTO WINDOW 1
LINK 2 FIELD [1]
ENTER
```

MEMBERSHIP RENEWAL APPLICATION PROGRAM

```
UNLOAD ALL
LABEL START2
INPUT TEXT1 ENTER A FOR ADULT OR M FOR MINOR
IF TEXT1="A" THEN JUMP ADULT
IF TEXT1="M" THEN JUMP MINOR
WAIT 5 NOT A CHOICE!  TRY AGAIN.
JUMP START2
LABEL ADULT
LOAD ADULT SCREEN MINORS
JUMP COMMON
LABEL MINOR
LOAD MINOR SCREEN MINOR
LABEL COMMON
ORDER KEY [1]
SPLIT VERTICAL 2 39
GOTO WINDOW 2
LOAD INTBACK SCREEN INTBACK2
ORDER KEY [1]
SPLIT HORIZANTAL 12 41
GOTO WINDOW 3
LOAD INVBACK SCREEN INVBACK
ORDER KEY [1]
GOTO WINDOW 1
LINK 2 FIELD [1]
GOTO WINDOW 2
LINK 3 FIELD [1]
GOTO WINDOW 1
EXECUTE UPDATE
EXECUTE MAINFILE
```

MEMBERSHIP RENEWAL APPLICATION PROGRAM (CONT'D.)

```
LABEL START
INPUT TEXT1 ARE ALL THE INTERESTS CORRECT (Y/N)?
IF TEXT1 ="Y" THEN JUMP OKAY
IF TEXT1 ="N" THEN JUMP INTER
WAIT 5 THIS IS A YES OR NO QUESTION.  TRY AGAIN!
JUMP START
LABEL INTER
GOTO WINDOW 2
UPDATE
LABEL OKAY
INPUT TEXT2 ARE ALL THE INVOLVEMENTS CORRECT (Y/N)?
IF TEXT2 ="Y" THEN JUMP OUT
IF TEXT2 ="N" THEN JUMP INVOLVE
WAIT 5 THIS IS A YES OR NO QUESTION.  TRY AGAIN!
JUMP OKAY
LABEL INVOLVE
GOTO WINDOW 3
UPDATE
LABEL OUT
EXECUTE MAINFILE
```

```
LABEL START1
GOTO WINDOW 1
INPUT TEXT1 IS ALL THE INFORMATION CURRENT (Y/N)?
IF TEXT1 ="Y" THEN JUMP AUTODATE
IF TEXT1= "N" THEN JUMP CURRENT
WAIT 5 THIS IS A YES OR NO QUESTION!  TRY AGAIN.
JUMP START1
LABEL CURRENT
WAIT 5 MAKE CORRECTIONS ON THE FOLLOWING RECORD
UPDATE
LABEL AUTODATE
INPUT TEXT2 ENTER THE DATE TO BE USED (MM/DD/YY).
LET [DATE] = TEXT2
LET [ACDATE] = TEXT2
LET [EXPIRE] = ADDYEARS(TEXT2,1)
WAIT 5 UPDATE COMPLETE
REPAINT
```

QUITING MEMBERSHIP UPDATES APPLICATION PROGRAM

```
GOTO WINDOW 3
KEY UPDATE
CLOSE
KEY UPDATE
CLOSE
KEY UPDATE
UNLOAD ALL
```

NEW VOLUNTEER APPLICATION PROGRAM

```
UNLOAD ALL
LOAD VOLUN SCREEN NEW
ENTER
LABEL START
INPUT TEXT1 WAS ANY TIME GIVEN NOW (Y/N)?
IF TEXT1 = "Y" EXECUTE VOLUN
IF TEXT2 = "N" JUMP OUT
WAIT 5 THIS IS ONLY A YES OR NO QUESTION!
JUMP START
LABEL OUT
 LET [TOTAL]=0
 LET [MONTH]=0
```

VOLUNTEER APPLICATION PROGRAM

```
UNLOAD ALL
LABEL START
LOAD VOLUN SCREEN VOLUN
INPUT TEXT1 ENTER THE NAME OF THE VOLUNTEER
INPUT VALUE1 ENTER THIS MONTH'S NUMERIC EQUIVALENT (I.E. JAN =
   1)
QUERY PREDEFINED VOLFIND SCREEN
INPUT VALUE2 ENTER THIS MONTH'S HOURS
LET [TOTAL] = [TOTAL] +VALUE2
LET [MONTH] = VALUE1
IF [MONTH] = 1 THEN JUMP JAN
IF [MONTH] = 2 THEN JUMP FEB
IF [MONTH] = 3 THEN JUMP MARCH
IF [MONTH] = 4 THEN JUMP APRIL
IF [MONTH] = 5 THEN JUMP MAY
IF [MONTH] = 6 THEN JUMP JUNE
IF [MONTH] = 7 THEN JUMP JULY
IF [MONTH] = 8 THEN JUMP AUG
IF [MONTH] = 9 THEN JUMP SEPT
IF [MONTH] = 10 THEN JUMP OCT
IF [MONTH] = 11 THEN JUMP NOV
IF [MONTH] = 12 THEN JUMP DEC
LABEL JAN
  LET [JANTOT] = VALUE2
  JUMP END
LABEL FEB
  LET [FEBTOT] = VALUE2
  JUMP END
LABEL MARCH
  LET [MARTOT] = VALUE2
  JUMP END
LABEL APRIL
  LET [APLTOT] = VALUE2
  JUMP END
LABEL MAY
  LET [MAYTOT] = VALUE2
  JUMP END
LABEL JUNE
  LET [JUNTOT] = VALUE2
  JUMP END
LABEL JULY
  LET [JULTOT] = VALUE2
  JUMP END
LABEL AUG
```

```
  LET [AUGTOT] = VALUE2
  JUMP END
LABEL SEPT
  LET [SETTOT] = VALUE2
  JUMP END
LABEL OCT
  LET [OCTTOT] = VALUE2
  JUMP END
LABEL NOV
  LET [NOVTOT] = VALUE2
  JUMP END
LABEL DEC
  LET [DECTOT] = VALUE2
  JUMP END
LABEL END
REPAINT
```

NEW DONOR APPLICATION PROGRAM

```
UNLOAD ALL
LOAD DONORS SCREEN DONOR
ENTER
INPUT TEXT1 WAS A PLEDGE MADE AT THIS TIME (Y/N)?
IF TEXT1 = "Y" EXECUTE PLEDGE
INPUT TEXT2 WAS A DONATION MADE AT THIS TIME (Y/N)?
IF TEXT2 ="Y" EXECUTE DONATE
```

PLEDGE APPLICATION PROGRAM

```
UNLOAD ALL
LOAD PLEDGES SCREEN STANDARD
SPLIT VERTICAL 2 42
GOTO WINDOW 2
LOAD DONORS SCREEN DONOR
GOTO WINDOW 1
WAIT 5 PLEASE ENTER DONOR, DATE, PURPOSE, AND DONE BY ON THE
        FOLLOWING
ENTER
LET TEXT2 = [DONOR]
GOTO WINDOW 2
QUERY PREDEFINED PLEDGE SCREEN
GOTO WINDOW 1
LABEL START
INPUT TEXT1 IS THIS THE ORIGINAL PLEDGE (Y/N)?
IF TEXT1 = "Y" THEN JUMP ORIGINAL
IF TEXT1 = "N" THEN JUMP HISTORY
WAIT 5 THIS IS A YES OR NO QUESTION!  TRY AGAIN.
JUMP START
LABEL ORIGINAL
INPUT VALUE1 ENTER THE AMOUNT OF THE PLEDGE.
LET [AMT] = VALUE1
LABEL TRY
INPUT TEXT2 WAS PART OF THE PLEDGE GIVEN AT THIS TIME (Y/N)?
IF TEXT2 = "Y" THEN JUMP AMOUNT
IF TEXT2 = "N" THEN JUMP FINISHED
WAIT 5 THIS IS A YES OR NO QUESTION!  TRY AGAIN.
JUMP TRY
LABEL AMOUNT
INPUT VALUE2 ENTER THE AMOUNT GIVEN AT THIS TIME.
LET [OUTSTD] = [AMT] - VALUE2
LET [PAID] = VALUE2
REPAINT
JUMP END
LABEL FINISHED
LET [OUTSTD] = [AMT]
LET [PAID] = 0
JUMP END
LABEL HISTORY
GOTO WINDOW 2
LET VALUE1 = [POUT]
LET VALUE2 = [PAMT]
GOTO WINDOW 1
WAIT 5 ENTER DONOR, DATE, PURPOSE, AND DONEBY IN THE FOLLOWING
```

```
ENTER
LET [OUTSTD] = VALUE1
LET [AMT] = VALUE2
INPUT VALUE2 ENTER THE AMOUNT GIVEN AT THIS TIME
LET [OUTSTD] = [OUTSTD] - VALUE2
LET [PAID] = VALUE2
REPAINT
LABEL END
LET VALUE1 = [OUTSTD]
GOTO WINDOW 2
LET [POUT] = VALUE1
REPAINT
```

DONATION APPLICATION PROGRAM

```
UNLOAD ALL
LOAD DONORS SCREEN DONOR
LABEL START
INPUT TEXT1 DID THIS PROSPECT MAKE A PLEDGE (Y/N)?
IF TEXT1 = "N" THEN JUMP DONATION
IF TEXT1 = "Y" THEN JUMP PLEDGE
WAIT 5 THIS IS A YES OR NO QUESTION!  TRY AGAIN.
JUMP START
LABEL PLEDGE
SPLIT VERTICAL 2 42
GOTO WINDOW 2
LOAD PLEDGES SCREEN standard
JUMP NORMAL
LABEL DONATION
INPUT TEXT2 DID THIS PROSPECT MAKE A DONATION (Y/N)?
IF TEXT2 ="Y" THEN JUMP DOIT
IF TEXT2 ="N" THEN JUMP NORMAL
WAIT 5 THIS IS A YES OR NO QUESTION!  TRY AGAIN.
JUMP DONATION
LABEL DOIT
SPLIT VERTICAL 2 42
GOTO WINDOW 2
LOAD GIVEN SCREEN GIVEN
LABEL NORMAL
GOTO WINDOW 1
ENTER
IF TEXT1 = "Y" OR TEXT2 ="Y" THEN JUMP OTHER
JUMP ENDIT
LABEL OTHER
GOTO WINDOW 2
IF TEXT1 ="Y" THEN JUMP OUTSTAND
IF TEXT2 = "Y" THEN JUMP DONATE
LABEL OUTSTAND
WAIT 5 ENTER DONOR, DATE, PURPOSE, AND DONEBY IN THE FOLLOWING
ENTER
INPUT VALUE1 ENTER THE AMOUNT OF THE PLEDGE
LET [AMT] = VALUE1
LABEL TRY
INPUT TEXT1 WAS PART OF THE PLEDGE GIVEN AT THIS TIME (Y/N)?
IF TEXT1 = "Y" THEN JUMP AMOUNT
IF TEXT1 = "N" THEN JUMP FINISHED
WAIT 5 THIS IS A YES OR NO QUESTION!  TRY AGAIN.
JUMP TRY
LABEL AMOUNT
```

```
INPUT VALUE2 ENTER THE AMOUNT GIVEN AT THIS TIME
LET [OUTSTD] = [AMT]- VALUE2
LET [PAID] = VALUE2
REPAINT
GOTO WINDOW 1
LET [POUT] = VALUE1 - VALUE2
LET [PAMT] = VALUE1
REPAINT
CLOSE
JUMP ENDIT
LABEL FINISHED
LET [OUTSTD] = [AMT]
LET [PAID] = 0
GOTO WINDOW 1
LET [PAMT] = VALUE1
LET [POUT] = VALUE1
REPAINT
CLOSE
JUMP ENDIT
LABEL DONATE
WAIT 5 PLEASE ENTER INFO IN FOLLOWING RECORD
ENTER
LABEL ENDIT
```

ANNUAL REPORT APPLICATION PROGRAM

```
UNLOAD ALL
LOAD ADULT SCREEN MINORS
IF MONTH([DOB])=00 THEN JUMP OUT1
ELSE
IF MONTH([DOB]) >= MONTH(TODAY) THEN JUMP MONTH
ELSE
LET [AGE] =YEAR(TODAY)-YEAR([DOB])
JUMP OUT1
LABEL MONTH
LET [AGE] =YEAR(TODAY) -YEAR([DOB]) -1
LABEL OUT1
ORDER KEY   0]  COMMENT THIS SORTS BY SEX
QUERY COUNT [10;11;] COMMENT THIS COUNTS THE AGES WITHIN SEX
ORDER SEQUENTIAL
ORDER KEY [17] COMMENT THIS SORTS BY ETHNIC GROUPS
QUERY COUNT [17]  COMMENT THIS COUNTS THE ETHNIC GROUPS
ORDER SEQUENTIAL
ORDER COUNT [4] COMMENT THIS SORTS BY ZONE
QUERY COUNT [4] COMMENT THIS COUNTS THE ZONES
ORDER SEQUENTIAL
ORDER KEY [20] COMMENT THIS SORTS BY DATE
QUERY COUNT [20] COMMENT THIS COUNTS THE DIFFERENT DATES
ORDER SEQUENTIAL
UNLOAD ALL
LOAD MINOR SCREEN MINOR
IF MONTH([DOB]) = 00 THEN JUMP OUT2
ELSE
IF MONTH([DOB]) >= MONTH(TODAY) THEN JUMP MONTH1
ELSE
LET [AGE] = YEAR(TODAY) -YEAR([DOB])
JUMP OUT2
LABEL MONTH1
LET [AGE] = YEAR(TODAY)-YEAR([DOB]) -1
LABEL OUT2
ORDER KEY [9] COMMENT THIS SORTS BY SEX
QUERY COUNT [9;8;] COMMENT THIS COUNTS THE AGES WITHIN SEX
ORDER SEQUENTIAL
ORDER KEY [11]   COMMENT ETHNIC GROUPS
QUERY COUNT [11]
ORDER SEQUENTIAL
ORDER KEY [4]  COMMENT THIS COUNTS ZONES
QUERY COUNT [4]
ORDER SEQUENTIAL
ORDER KEY [18]   COMMENT THIS COUNTS ORIGINAL DATE OF MEMBERSHIP
```

```
QUERY COUNT [18]
ORDER SEQUENTIAL
```

APPENDIX F

dBASEIII APPLICATION PROGRAMS

MAIN MENU PROGRAM

```
SET TALK OFF
DO WHILE .T.
  CLEAR
  TEXT
**************************************************
*                                                *
*                  MAIN MENU                     *
*                  ---- ----                     *
*                                                *
*           1.  NEW MEMBERSHIP                   *
*           2.  MEMBERSHIP RENEWAL               *
*           3.  NEW VOLUNTEER                    *
*           4.  VOLUNTEER HOURS                  *
*           5.  NEW DONOR                        *
*           6.  DONATION OR PLEDGE               *
*           7.  MONTHLY PROCESSES                *
*           8.  MAILING LIST                     *
*           9.  ANNUAL REPORT                    *
*           0.  EXIT                             *
*                                                *
**************************************************
  ENDTEXT
  ACCEPT "ENTER THE NUMBER OF YOUR SELECTION: " TO SELECT
  DISPLAY MEMORY
  WAIT TO WHY
  CLEAR
    IF SELECT ="1"
      DO NEW
    ENDIF
    IF SELECT ='2'
      DO RENEW
    ENDIF
    IF SELECT ='3'
      DO NEWVOL
    ENDIF
    IF SELECT ='4'
      DO VOLUN
    ENDIF
    IF SELECT ='5'
      DO NEWDON
    ENDIF
    IF SELECT ='6'
      DO GIVING
```

```
    ENDIF
    IF SELECT ='7'
      DO MONTH
    ENDIF
    IF SELECT ='8'
      DO MAIL
    ENDIF
    IF SELECT ='9'
      DO ANNUAL
    ENDIF
    IF SELECT ="0"
      QUIT
    ENDIF
ENDDO
```

NEW MEMBERSHIP APPLICATION PROGRAM

```
CLEAR
WAIT 'PRESS A FOR ADULT, M FOR MINOR:   ' TO TYPE
IF TYPE ='A'
  USE B:ADULT
  CLEAR
  APPEND BLANK
  ACCEPT 'ENTER LAST NAME:' TO LAST
  ACCEPT 'ENTER FIRST NAME:' TO FIRST
  ACCEPT 'ENTER MIDDLE NAME:' TO MIDDLE
  CLEAR
  @ 2,10 SAY 'SEX:' GET SEX
  @ 2,16 SAY 'AGE:' GET AGE
  @ 3,10 SAY 'STATUS (M,S,W,D):' GET STATUS
  @ 4,10 SAY 'EDUCATION CODE:' GET EDUC
  @ 6,10 SAY 'STREET ADDRESS:' GET STREET
  @ 7,10 SAY 'CITY, ST ZIP:' GET CITYST
  @ 8,10 SAY 'ZONE CODE:' GET ZONE
  @ 9,10 SAY 'HOME PHONE:' GET HTELE
  @ 11,10 SAY 'OCCUPATION:' GET OCCUP
  @ 12,10 SAY 'PLACE OF WORK:' GET PLOFWK
  @ 13,10 SAY 'WORK PHONE:' GET WTELE
  @ 15,10 SAY 'NUMBER OF CHILDREN:' GET NOCH
  @ 16,10 SAY 'DATE OF BIRTH (MM/DD/YY):' GET DOB
  @ 17,10 SAY 'YEAR MOVED TO SALINA:' GET YRSAL
  @ 18,10 SAY 'RACE CODE:' GET RACE
  @ 20,10 SAY 'TYPE OF ACTIVITY CARD (FAC OR AC):' GET ACNOTE
  @ 21,10 SAY 'DATE OF EXPIRATION (MM/DD/YY):' GET EXPIRE
  READ
  STORE DATE() TO TODAY
  REPLACE DATE WITH TODAY,ACDATE WITH TODAY,LNAME WITH LAST
  REPLACE FNAME WITH FIRST,MNAME WITH MIDDLE,ORIG WITH TODAY
ENDIF
IF TYPE ='M'
  USE B:MINOR
  CLEAR
  ACCEPT 'ENTER LAST NAME:' TO LAST
  ACCEPT 'ENTER FIRST NAME:' TO FIRST
  ACCEPT 'ENTER MIDDLE NAME:' TO MIDDLE
  APPEND BLANK
  CLEAR
  @ 2,10 SAY 'SEX:' GET SEX
  @ 2,16 SAY 'AGE:' GET AGE
  @ 3,10 SAY 'STREET ADDRESS:' GET STREET
  @ 4,10 SAY 'CITY, ST ZIP:' GET CITYST
  @ 5,10 SAY 'ZONE CODE:' GET ZONE
```

```
   @ 6,10 SAY 'HOME PHONE:' GET HTELE
   @ 8,10 SAY 'PLACE OF WORK:' GET PLOFWK
   @ 9,10 SAY 'SCHOOL:' GET SCHOOL
   @ 10,10 SAY 'RACE CODE:' GET RACE
   @ 11,10 SAY 'DATE OF BIRTH (MM/DD/YY):' GET DOB
   @ 12,10 SAY 'PARENT OR GUARDIAN:' GET GUARD
   @ 13,10 SAY 'DOCTOR:' GET DOC
   @ 14,10 SAY 'DOCTORS PHONE:' GET DOCTEL
   @ 15,10 SAY 'CONTACT PERSON OTHER THAN GUARDIAN:' GET CONTACT
   @ 16,10 SAY 'CONTACTS PHONE:' GET CONTEL
   @ 17,10 SAY 'EXPIRATION DATE (MM/DD/YY):' GET EXPIRE
   @ 18,10 SAY 'TYPE OF ACTIVITY CARD (FAC OR AC):' GET ACNOTE
   READ
   STORE DATE() TO TODAY
   REPLACE DATE WITH TODAY, ACDATE WITH TODAY, LNAME WITH LAST
   REPLACE FNAME WITH FIRST, MNAME WITH MIDDLE,ORIG WITH TODAY
ENDIF
USE B:INTEREST
APPEND BLANK
CLEAR
@ 2,10 SAY 'PLACE T NEXT TO THE APPROPRIATE INTEREST(S)'
@ 3,10 SAY 'PRESS THE CTRL KEY AND THE END KEY AT THE'
@ 4,15 SAY 'SAME TIME WHEN FINISHED'
@ 5,10 SAY 'ARTS/CRAFTS:' GET ARTS
@ 6,10 SAY 'CLERICAL:' GET CLERICAL
@ 7,10 SAY 'CHILDREN:' GET CHILDREN
@ 8,10 SAY 'COOKING AND BAKING:' GET COOKING
@ 9,10 SAY 'P.E./SWIM/DANCE:' GET PE
@ 10,10 SAY 'TELPHONING:' GET TELE
@ 11,10 SAY 'UPKEEP/CLEANING:' GET UPKEEP
@ 12,10 SAY 'VOLUNTEER:' GET VOLUNTR
READ
REPLACE LNAME WITH LAST, FNAME WITH FIRST,MNAME WITH MIDDLE
USE B:INVOLVE
CLEAR
APPEND BLANK
@ 2,10 SAY 'PLACE T NEXT TO THE AREA(S) OF INVOLVMENT'
@ 3,10 SAY 'PRESS THE CTRL KEY AND THE END KEY WHEN DONE'
@ 5,10 SAY 'BOARD:' GET BOARD
@ 6,10 SAY 'COMMITTEE:' GET COMMITTEE
@ 7,10 SAY 'CLASS:' GET CLASS
@ 8,10 SAY 'CLUB :' GET CLUB
@ 9,10 SAY 'RECREATION:' GET RECREATION
@ 10,10 SAY 'VOLUNTEER:' GET VOLUNTR
READ
REPLACE LNAME WITH LAST, FNAME WITH FIRST,MNAME WITH MIDDLE
CLEAR GETS
CLOSE DATABASES
RETURN
```

MEMBERSHIP RENEWAL APPLICATION PROGRAM

```
CLEAR
WAIT 'PRESS A FOR ADULT, OR M FOR MINOR:   ' TO TYPE
IF TYPE = 'A'
  USE B:ADULT INDEX ADNAME
  REINDEX
  ACCEPT 'ENTER LAST NAME:   ' TO LAST
  ACCEPT 'ENTER FIRST NAME:   ' TO FIRST
  ACCEPT 'ENTER MIDDLE NAME:   ' TO MIDDLE
  LOCATE FOR LNAME=LAST .AND. FNAME=FIRST .AND. MNAME=MIDDLE
  TEXT

    THE FOLLOWING IS THE RECORD OF THE RENEWING MEMBER.  PLEASE
    MAKE ANY CORRECTIONS NECESSARY (INCLUDING DATES).  ONCE
    CORRECTIONS ARE COMPLETE, PRESS THE CTRL KEY AND THE END
    KEY SIMULTANEOUSLY.

  ENDTEXT
  WAIT 'HIT ANY KEY TO VIEW RECORD' TO DUMMY
  EDIT
  ENDIF
  IF TYPE = 'M'
    CLEAR
    USE B:MINOR INDEX MINNAME
    REINDEX
    ACCEPT 'ENTER LAST NAME: ' TO LAST
    ACCEPT 'ENTER FIRST NAME : ' TO FIRST
    ACCEPT 'ENTER MIDDLE NAME: ' TO MIDDLE
    LOCATE FOR LNAME=LAST .AND. FNAME=FIRST .AND. MNAME=MIDDLE
  TEXT

    THE FOLLOWING IS THE RECORD OF THE RENEWING MEMBER.  PLEASE
    MAKE ANY CORRECTIONS (INCLUDING DATES).  ONCE CORRECTIONS
    ARE COMPLETE, PRESS THE CTRL KEY AND THE END KEY
    SIMULTANEOUSLY.

  ENDTEXT
  WAIT 'HIT ANY KEY TO VIEW RECORD' TO DUMMY
  EDIT
  ENDIF
  CLEAR
  USE B:INTEREST
  LOCATE FOR LNAME=LAST .AND. FNAME=FIRST .AND. MNAME=MIDDLE
  TEXT

    THE FOLLOWING IS THE MEMBER'S INTEREST RECORD.  PLEASE MAKE
```

```
  ANY CORRECTIONS NECESSARY.  PLACE A T NEXT TO NEW INTERESTS
  AND ERASE THE T NEXT TO THOSE INTERESTS WHICH ARE NO LONGER
  APPLICABLE.  ONCE CORRECTIONS ARE COMPLETE, PRESS THE CTRL
  KEY AND THE END KEY SIMULTANEOUSLY.

ENDTEXT
WAIT 'PRESS ANY KEY TO CONTINUE' TO DUMMY
EDIT
CLEAR
USE B:INVOLVE
LOCATE FOR LNAME=LAST .AND. FNAME=FIRST .AND. MNAME=MIDDLE
TEXT

  THE FOLLOWING IS THE MEMBER'S INVOLVEMENT RECORD.  PLEASE
  MAKE ANY CORRECTIONS NECESSARY.  PLACE A T NEXT TO NEW
  AREAS OF INVOLVEMENT AND ERASE THE T NEXT TO THOSE AREAS
  OF INVOLVEMENT WHICH ARE NO LONGER APPLICABLE.  ONCE COR-
  RECTIONS ARE COMPLETE, PRESS THE CTRL KEY AND THE END KEY
  SIMULTANEOUSLY.

ENDTEXT
WAIT 'PRESS ANY KEY TO CONTINUE ' TO DUMMY
EDIT
CLOSE DATABASES
RETURN
```

NEW VOLUNTEER APPLICATION PROGRAM

```
CLEAR
USE B:VOLUNTEER
APPEND BLANK
@ 7,10 SAY 'ENTER LAST NAME:   ' GET LNAME
@ 9,10 SAY 'ENTER FIRST NAME:   ' GET FNAME
READ
STORE 0 TO AMOUNT
REPLACE TOTAL WITH AMOUNT
REPLACE MONTH WITH 0
ACCEPT 'DID THIS VOLUNTEER DONATE TIME THIS MONTH (Y/N)? ' TO
TIME
IF TIME = 'Y'
  DO B:VOLUN
ENDIF
CLOSE DATABASES
RETURN
```

VOLUNTEER HOURS INPUT APPLICATION PROGRAM

```
CLEAR
USE B:VOLUNTEER
ACCEPT 'ENTER LAST NAME:   ' TO LAST
ACCEPT 'ENTER FIRST NAME:   ' TO FIRST
LOCATE FOR LNAME = LAST.AND.FNAME = FIRST
INPUT 'ENTER THIS MONTHS TOTAL HOURS:   ' TO HOURS
REPLACE TOTAL WITH TOTAL+HOURS
IF MONTH(DATE())=1
  REPLACE JANTOT WITH HOURS
ENDIF
IF MONTH(DATE())=2
  REPLACE FEBTOT WITH HOURS
ENDIF
IF MONTH(DATE()) =3
  REPLACE MARTOT WITH HOURS
ENDIF
IF MONTH(DATE()) =4
  REPLACE APLTOT WITH HOURS
ENDIF
IF MONTH(DATE()) =5
  REPLACE MAYTOT WITH HOURS
ENDIF
IF MONTH(DATE()) =6
  REPLACE JUNTOT WITH HOURS
ENDIF
IF MONTH(DATE())=7
  REPLACE JULTOT WITH HOURS
ENDIF
IF MONTH(DATE())=8
  REPLACE AUGTOT WITH HOURS
ENDIF
IF MONTH(DATE()) =9
  REPLACE SEPTOT WITH HOURS
ENDIF
IF MONTH(DATE())=10
  REPLACE OCTTOT WITH HOURS
ENDIF
IF MONTH(DATE())=11
  REPLACE NOVTOT WITH HOURS
ENDIF
IF MONTH(DATE())=12
  REPLACE DECTOT WITH HOURS
ENDIF
IF MONTH>0
  IF MONTH(DATE())<MONTH
```

```
    STORE MONTH(DATE()) +12 TO DATE
  ELSE
    STORE MONTH(DATE()) TO DATE
  ENDIF
  STORE DATE - MONTH -1 TO N
  DO WHILE N>0
  Q=MONTH +N
  IF Q>12
    STORE Q +12 TO Q
  ENDIF
  IF Q =1
    REPLACE JANTOT WITH 0
  ENDIF
  IF Q =2
    REPLACE FEBTOT WITH 0
  ENDIF
  IF Q = 3
    REPLACE MARTOT WITH 0
  ENDIF
  IF Q = 4
    REPLACE APLTOT WITH 0
  ENDIF
  IF Q = 5
    REPLACE MAYTOT WITH 0
  ENDIF
  IF Q = 6
    REPLACE JUNTOT WITH 0
  ENDIF
  IF Q = 7
    REPLACE JULTOT WITH 0
  ENDIF
  IF Q = 8
    REPLACE AUGTOT WITH 0
  ENDIF
  IF Q = 9
    REPLACE SEPTOT WITH 0
  ENDIF
  IF Q = 10
    REPLACE OCTTOT WITH 0
  ENDIF
  IF Q = 11
    REPLACE NOVTOT WITH 0
  ENDIF
  IF Q = 12
    REPLACE DECTOT WITH 0
  ENDIF
  N = N-1
ENDDO
ENDIF
```

```
REPLACE MONTH WITH MONTH(DATE())
CLOSE DATABASES
RETURN
```

NEW DONOR APPLICATION PROGRAM

```
CLEAR
USE B:PROSPECT
APPEND BLANK
@ 5,10 SAY 'ENTER DONORS NAME:' GET DONOR
@ 6,10 SAY 'ENTER CONTACT PERSONS NAME:' GET CONNAME
@ 7,10 SAY 'CONTACT OR DONORS TITLE:' GET TITLE
@ 8,10 SAY 'STREET ADRESS:' GET STREET
@ 9,10 SAY 'CITY, ST ZIP:' GET CITYST
@ 10,10 SAY 'WHERE PREFERS TO BE CONTACTED (H OR W):' GET
CONPREF
@ 11,10 SAY 'PROSPECTS AGE (APPROXIMATELY):' GET PROSAGE
@ 12,10 SAY 'PROSPECTS EDUCATION:' GET PROSED
@ 13,10 SAY 'PROSPECTS FIELD OF STUDY:' GET PROSTUD
@ 14,10 SAY 'PROSPECTS RELIGIOUS AFFILIATION:' GET PROREL .
@ 15,10 SAY 'SPOUSES NAME:' GET SPNAME
@ 16,10 SAY 'SPOUSES AGE (APPROXIMATELY):' GET SPAGE
@ 17,10 SAY 'SPOUSES EDUCATION:' GET SPED
@ 18,10 SAY 'SPOUSES FIELD OF STUDY:' GET SPSTUD
@ 19,10 SAY 'NUMBER OF CHILDREN:' GET NOCH
@ 20,10 SAY 'PROSPECTS INTERESTS:' GET PROINT
@ 21,10 SAY 'COMMENTS:' GET COMM
@ 22,10 SAY 'INVOLVEMENT WITH OTHER ORGANIZATIONS:' GET OTHERS
@ 23,10 SAY 'KNOWN WELL BY:' GET KNOWN
@ 24,10 SAY 'SOLICITOR:' GET SOLIC
READ
CLEAR GETS
ACCEPT 'WAS A DONATION OR A PLEDGE GIVEN (Y/N)?' TO GIVER
IF GIVER = 'Y'
  DO B:GIVING
ENDIF
CLOSE DATABASES
RETURN
```

DONATION OR PLEDGE APPLICATION PROGRAM

```
CLEAR
ACCEPT 'ENTER DONORS NAME:' TO NAME
ACCEPT 'ENTER P FOR A PLEDGE, OR D FOR DONATION:' TO TYPE
IF TYPE = 'P'
  DO B:PROMISE
ENDIF
IF TYPE = 'D'
  DO B:DONATE
ENDIF
CLOSE DATABASE
RETURN
```

PLEDGE APPLICATION PROGRAM

```
CLEAR
ACCEPT 'ENTER THE PURPOSE:' TO USE
ACCEPT 'ENTER WHO HANDLED THE DONATION:' TO WHO
ACCEPT 'ENTER THE DATE (MM/DD/YY):' TO WHEN
ACCEPT 'IS THIS THE ORIGINAL PLEDGE (Y/N)?:' TO ORIGIN
IF ORIGIN = 'N'
  USE B:PROSPECT
  LOCATE FOR DONOR = NAME
  STORE PAMT TO AMOUNT
  STORE POUT TO OLDOUT
  INPUT 'ENTER THE AMOUNT GIVEN NOW:' TO PAYED
  STORE OLDOUT-PAYED TO OUT
  REPLACE POUT WITH OUT
ENDIF
IF ORIGIN = 'Y'
  INPUT 'ENTER THE AMOUNT PLEDGED:' TO AMOUNT
  ACCEPT 'WAS ANY OF THE PLEDGE GIVEN AT THIS TIME(Y/N)?:'TO
NOW
  IF NOW='Y'
    INPUT 'HOW MUCH WAS GIVEN?:' TO PAYED
    STORE AMOUNT-PAYED TO OUT
  ENDIF
  IF NOW = 'N'
    STORE 0 TO PAYED
    STORE AMOUNT TO OUT
  ENDIF
  CLEAR
  USE B:PLEDGES
  APPEND BLANK
  REPLACE DONOR WITH NAME, AMT WITH AMOUNT
  REPLACE OUTSTD WITH OUT, PAID WITH PAYED
ENDIF
CLOSE DATABASES
RETURN
```

DONATION APPLICATION PROGRAM

```
CLEAR
USE B:GIVEN
APPEND BLANK
REPLACE DONOR WITH NAME
ACCEPT 'ENTER THE DATE (MM/DD/YY):' TO WHEN
INPUT 'ENTER AMOUNT GIVEN:' TO AMOUNT
ACCEPT 'ENTER THE PURPOSE:' TO USE
ACCEPT 'WHO HANDLED THE DONATION?:' TO WHO
ACCEPT 'WAS A RECEIPT GIVEN (Y/N)?:' TO CARBON
ACCEPT 'WAS A LETTER SENT (Y/N)?:' TO PAPER
REPLACE DATE WITH CTOD(WHEN), AMT WITH AMOUNT
REPLACE PURPOSE WITH USE, DONEBY WITH WHO
IF CARBON = 'Y'
  REPLACE RECPT WITH .T.
ENDIF
IF PAPER = 'Y'
  REPLACE LETTER WITH .T.
ENDIF
CLOSE DATABASES
RETURN
```

MONTHLY PROCESSES APPLICATION PROGRAM

```
USE B:ADULT
GO TOP
STORE MONTH(DATE())+1 TO MNEXT
STORE YEAR(DATE()) TO YRNOW
LABEL FORM ADEXPIRE TO PRINT FOR MNEXT=MONTH(EXPIRE);
.AND.YRNOW=YEAR(EXPIRE)
EJECT
USE B:MINOR
LABEL FORM MINEXPIRE TO PRINT FOR MNEXT=MONTH(EXPIRE);
.AND. YRNOW=YEAR(EXPIRE)
USE B:ADULT
IF MONTH(DATE())=1
  STORE 13 TO NOW
ENDIF
IF MONTH(DATE())>1
  STORE MONTH(DATE()) TO NOW
ENDIF
GO TOP
DO WHILE .NOT. EOF()
IF NOW>MONTH(EXPIRE).AND. YEAR(DATE())>=YEAR(EXPIRE)
  STORE LNAME TO LAST
  STORE FNAME TO FIRST
  STORE MNAME TO MIDDLE
  STORE STREET TO ADD
  STORE CITYST TO TOWN
  STORE ZONE TO NUMB
  STORE HTELE TO HOME
  STORE OCCUP TO JOB
  STORE PLOFWK TO WHERE
  STORE WTELE TO WORK
  STORE DATE TO DAY
  STORE SEX TO GENDER
  STORE AGE TO OLDNESS
  STORE EDUC TO DEGREE
  STORE STATUS TO TYPE
  STORE NOCH TO TOOMANY
  STORE DOB TO HOWOLD
  STORE YRSAL TO WHEN
  STORE RACE TO ETHNIC
  STORE SENIOR TO OVER65
  STORE VOTE TO YEAR
  STORE EXPIRE TO THEEND
  STORE ACDATE TO GIVEN
  STORE ACNOTE TO KIND
  STORE ORIG TO ORIGIN
```

```
DELETE
USE B:XPIRAD
APPEND BLANK
  REPLACE LNAME WITH LAST,FNAME WITH FIRST, MNAME WITH MIDDLE
  REPLACE STREET WITH ADD, CITYST WITH TOWN, ZONE WITH NUMB
  REPLACE HTELE WITH HOME, OCCUP WITH JOB, PLOFWK WITH WHERE
  REPLACE WTELE WITH WORK,DATE WITH DAY, SEX WITH GENDER
  REPLACE AGE WITH OLDNESS, EDUC WITH DEGREE, STATUS WITH TYPE
  REPLACE NOCH WITH TOOMANY, DOB WITH HOWOLD, YRSAL WITH WHEN
  REPLACE RACE WITH ETHNIC, SENIOR WITH OVER65, VOTE WITH YEAR
  REPLACE EXPIRE WITH THEEND, ACDATE WITH GIVEN
  REPLACE ACNOTE WITH KIND, ORIG WITH ORIGIN
ENDIF
USE B:ADULT
SKIP
ENDDO
USE B:MINOR
IF MONTH(DATE())=1
  STORE 13 TO NOW
ENDIF
  STORE MONTH(DATE()) TO NOW
ENDIF
GOTOP
DO WHILE .NOT. EOF()
IF NOW > MONTH(EXPIRE).AND. YEAR(DATE())>=YEAR(EXPIRE)
  STORE LNAME TO LAST
  STORE FNAME TO FIRST
  STORE MNAME TO MIDDLE
  STORE STREET TO ADD
  STORE CITYST TO TOWN
  STORE ZONE TO NUMB
  STORE HTELE TO HOME
  STORE PLOFWK TO JOB
  STORE DATE TO DAY
  STORE AGE TO OLDNESS
  STORE SEX TO GENDER
  STORE DOB TO HOWOLD
  STORE RACE TO ETHNIC
  STORE GUARD TO PROTECT
  STORE SCHOOL TO BUILD
  STORE DOC TO DOCTOR
  STORE DOCTEL TO DTELE
  STORE CONTACT TO OTHER
  STORE CONTEL TO CTELE
  STORE ORIG TO ORIGIN
  STORE EXPIRE TO THEEND
  STORE ACDATE TO GIVEN
  STORE ACNOTE TO KIND
 DELETE
```

```
  USE B: XPIRMIN
  APPEND BLANK
  REPLACE LNAME WITH LAST, FNAME WITH FIRST, MNAME WITH MIDDLE
  REPLACE STREET WITH ADD, CITYST WITH TOWN, ZONE WITH NUMB
  REPLACE HTELE WITH HOME, PLOFWK WITH JOB, DATE WITH DAY
  REPLACE AGE WITH OLDNESS, SEX WITH GENDER, DOB WITH HOWOLD
  REPLACE RACE WITH ETHNIC, GUARD WITH PROTECT, SCHOOL WITH
BUILD
  REPLACE DOC WITH DOCTOR, DOCTEL WITH DTELE, CONTACT WITH
OTHER
  REPLACE NONTEL WITH CTELE, ORIG WITH ORIGIN, EXPIRE WITH
THEEND
  REPLACE ACDATE WITH GIVEN, ACNOTE WITH KIND
  ENDIF
  USE B:MINOR
  SKIP
ENDDO
USE B:VOLUNTEER
CLEAR
STORE MONTH(DATE()) TO WHICH
GO TOP
DO WHILE .NOT. EOF()
IF WHICH = 1
  DISPLAY OFF LNAME, FNAME, TOTAL, JANTOT TO PRINT
ENDIF
IF WHICH = 2
  DISPLAY OFF LNAME, FNAME, TOTAL, FEBTOT TO PRINT
ENDIF
IF WHICH = 3
  DISPLAY OFF LNAME, FNAME, TOTAL, MARTOT TO PRINT
ENDIF
IF WHICH = 4
  DISPLAY OFF LNAME, FNAME, TOTAL, APLTOT TO PRINT
ENDIF
IF WHICH = 5
  DISPLAY OFF LNAME, FNAME, TOTAL, MAYTOT TO PRINT
ENDIF
IF WHICH = 6
  DISPLAY OFF LNAME, FNAME, TOTAL, JUNTOT TO PRINT
ENDIF
IF WHICH = 7
  DISPLAY OFF LNAME, FNAME, TOTAL, JULTOT TO PRINT
ENDIF
IF WHICH = 8
  DISPLAY OFF LNAME, FNAME, TOTAL, AUGTOT TO PRINT
ENDIF
IF WHICH = 9
  DISPLAY OFF LNAME, FNAME, TOTAL, SEPTOT TO PRINT
ENDIF
```

```
IF WHICH = 10
  DISPLAY OFF LNAME, FNAME, TOTAL, OCTTOT TO PRINT
ENDIF
IF WHICH = 11
  DISPLAY OFF LNAME, FNAME, TOTAL, NOVTOT TO PRINT
ENDIF
IF WHICH = 12
  DISPLAY OFF LNAME, FNAME, TOTAL, DECTOT TO PRINT
ENDIF
SKIP
ENDDO
CLOSE DATABASES
RETURN
```

MAILING LIST APPLICATION PROGRAM

```
USE B:ADULT
GO TOP
LABEL FORM MAIL TO PRINT
EJECT
USE B:MINOR
LABEL FORM MAIL2 TO PRINT
EJECT
CLOSE DATABASES
RETURN
```

ANNUAL REPORT APPLICATION PROGRAM

```
CLEAR
USE B:ADULT
GO TOP
DO WHILE .NOT. EOF()
  IF MONTH(DOB)>MONTH(DATE()) .AND. MONTH(DOB)<>0
    REPLACE AGE WITH YEAR (DATE())-YEAR(DOB)
  ENDIF
  IF MONTH(DOB)<MONTH(DATE()) .AND. MONTH(DOB)<>0
    REPLACE AGE WITH YEAR(DATE())-YEAR(DOB)-1
  ENDIF
  SKIP
ENDDO
COUNT FOR SEX='F' .AND. AGE>18 .AND. AGE<=34 TO F1834
COUNT FOR SEX='M' .AND. AGE>18 .AND. AGE<=34 TO M1834
COUNT FOR SEX='F'.AND.YEAR(ORIG)=YEAR(DATE()).AND.;
AGE>18.AND.AGE<=34 TO FN1834
COUNT FOR SEX='M' .AND. YEAR(ORIG)=YEAR(DATE()).AND.;
AGE>18.AND.AGE<=34 TO MN1834
COUNT FOR SEX='F' .AND. YEAR(ORIG)<YEAR(DATE()).AND.;
AGE>18.AND.AGE<=34 TO FO1834
COUNT FOR SEX='M' .AND. YEAR(ORIG)<YEAR(DATE()).AND.;
AGE>18.AND.AGE<=34 TO MO1834
COUNT  FOR SEX='F'.AND. AGE>34 .AND. AGE<=59 TO F3559
COUNT  FOR SEX='M'.AND. AGE>34 .AND. AGE<=59 TO M3559
COUNT  FOR SEX='F'.AND. AGE>=60 TO F60
COUNT  FOR SEX='M'.AND. AGE>=60 TO M60
COUNT  FOR SEX='F' .AND. YEAR(ORIG)=YEAR(DATE()).AND.;
AGE>34 .AND. AGE<=59 TO FN3559
COUNT  FOR SEX='M' .AND. YEAR(ORIG)=YEAR(DATE()).AND.;
AGE>34.AND. AGE<=59 TO MN3559
COUNT  FOR SEX='F' .AND. YEAR(ORIG)<YEAR(DATE()).AND.;
AGE>34.AND. AGE<=59 TO FO3559
COUNT  FOR SEX='M' .AND. YEAR(ORIG)<YEAR(DATE()).AND.;
AGE>34.AND.AGE<=59 TO MO3559
COUNT  FOR SEX='F'.AND.YEAR(ORIG)=YEAR(DATE()).AND.AGE>=60 TO
FO60
COUNT  FOR SEX='M'.AND.YEAR(ORIG)=YEAR(DATE()).AND.AGE>=60 TO
MO60
COUNT  FOR SEX='F'.AND.YEAR(ORIG)<YEAR(DATE()).AND.AGE>=60 TO
FN60
COUNT  FOR SEX='M'.AND.YEAR(ORIG)<YEAR(DATE()).AND.AGE>=60 TO;
 MN60
COUNT  FOR RACE='W' TO WHITE
COUNT  FOR RACE='A' TO ASIAN
COUNT  FOR RACE='N' TO NATIVE
```

```
COUNT  FOR RACE='B' TO BLACK
COUNT  FOR RACE='H' TO HISPAN
COUNT  FOR RACE='O' TO OTHER
COUNT  FOR ZONE='1' TO ZONE1
COUNT  FOR ZONE='2' TO ZONE2
COUNT  FOR ZONE='3' TO ZONE3
COUNT  FOR ZONE='4' TO ZONE4
COUNT  FOR ZONE='5' TO ZONE5
COUNT  FOR ZONE='6' TO ZONE6
COUNT  FOR ZONE='7' TO ZONE7
USE B:MINOR
DO WHILE .NOT. EOF()
IF MONTH(DOB)>MONTH(DATE()) .AND. MONTH(DOB)<>0
  REPLACE AGE WITH YEAR(DATE())-YEAR(DOB)
ENDIF
IF MONTH(DOB)<MONTH(DATE()) .AND. MONTH(DOB)<>0
  REPLACE AGE WITH YEAR(DATE())-YEAR(DOB)-1
ENDIF
SKIP
COUNT FOR SEX='F' .AND. AGE<6 TO F6
COUNT FOR SEX='M' .AND. AGE<6 TO M6
COUNT FOR SEX='F'.AND.YEAR(ORIG)=YEAR(DATE()).AND.AGE<6 TO FN6
COUNT FOR SEX='M' .AND. YEAR(ORIG)=YEAR(DATE()).AND.AGE<6 TO
MN6
COUNT FOR SEX='F' .AND. YEAR(ORIG)<YEAR(DATE()).AND.AGE<6 TO
FO6
COUNT FOR SEX='M' .AND. YEAR(ORIG)<YEAR(DATE()).AND.AGE<6 TO
MO6
COUNT FOR SEX='F' .AND. AGE>=6 .AND. AGE<=11 TO F611
COUNT FOR SEX='M' .AND. AGE>=6 .AND. AGE<=11 TO M611
COUNT FOR SEX='F' .AND. AGE>=6 .AND. AGE<=11.AND.;
YEAR(ORIG)=YEAR(DATE()) TO FN611
COUNT FOR SEX='M' .AND. AGE>=6 .AND. AGE<=11.AND.;
YEAR(ORIG)=YEAR(DATE()) TO MN611
COUNT FOR SEX='F' .AND. AGE>=6 .AND. AGE<=11.AND.;
YEAR(ORIG) < YEAR(DATE()) TO FO611
COUNT FOR SEX='M' .AND. AGE>=6 .AND. AGE<=11.AND.;
YEAR(ORIG)<YEAR(DATE()) TO MO611
COUNT FOR SEX='F' .AND. AGE>=12 .AND. AGE<=14 TO F1214
COUNT FOR SEX='M' .AND. AGE>=12 .AND. AGE<=14 TO M1214
COUNT FOR SEX='F' .AND. AGE>=12 .AND. AGE<=14.AND.;
YEAR(ORIG)=YEAR(DATE()) TO FN1214
COUNT FOR SEX='M' .AND. AGE>=12 .AND. AGE<=14.AND.;
YEAR(ORIG)=YEAR(DATE()) TO MN1214
COUNT FOR SEX='F' .AND. AGE>=12 .AND. AGE<=14.AND.;
YEAR(ORIG)<YEAR(DATE()) TO FO1214
COUNT FOR SEX='M' .AND. AGE>=12 .AND. AGE<=14.AND.;
YEAR(ORIG)<YEAR(DATE()) TO MO1214
COUNT FOR SEX='F' .AND. AGE>=15 .AND. AGE<=17 TO F1517
```

```
COUNT FOR SEX='M' .AND. AGE>=15 .AND. AGE<=17 TO M1517
COUNT FOR SEX='F' .AND. AGE>=15 .AND. AGE<=17.AND.;
YEAR(ORIG)=YEAR(DATE()) TO FN1517
COUNT FOR SEX='M' .AND. AGE>=15 .AND. AGE<=17.AND.;
YEAR(ORIG)=YEAR(DATE()) TO MN1517
COUNT FOR SEX='F' .AND. AGE>=15 .AND. AGE<=17.AND.;
YEAR(ORIG)<YEAR(DATE()) TO FO1517
COUNT FOR SEX='M' .AND. AGE>=15 .AND. AGE<=17.AND.;
YEAR(ORIG)<YEAR(DATE()) TO MO1517
COUNT FOR RACE='W' TO MWHITE
COUNT FOR RACE='A' TO MASIAN
COUNT FOR RACE='N' TO MNATIVE
COUNT FOR RACE='B' TO MBLACK
COUNT FOR RACE='H' TO MHISPAN
COUNT FOR RACE='O' TO MOTHER
COUNT FOR ZONE='1' TO MZONE1
COUNT FOR ZONE='2' TO MZONE2
COUNT FOR ZONE='3' TO MZONE3
COUNT FOR ZONE='4' TO MZONE4
COUNT FOR ZONE='5' TO MZONE5
COUNT FOR ZONE='6' TO MZONE6
COUNT FOR ZONE='7' TO MZONE7
DO B:ANMIN
RETURN
```

OUTPUT OF ANNUAL REPORT APPLICATION PROGRAM

```
SET MARGIN TO 10
STORE YEAR (DATE()) TO PERIOD
SET PRINT ON
?
?
?
?
?
?
?'                              ANNUAL SUMMARY'
?
?'SALINA KANSAS YWCA                                    ', PERIOD
?
?
?'MEMBERS (WOMEN)          18-34          35-59          60+'
?'-------------------------------------------------------'
?'NUMBER ON REGISTER       ',STR(F1834,3)+'
 +STR(F3559,3)+'           '+STR(F60,3)
?'NUMBER OF NEW MEMBERS    ',STR(FN1834,3)+'
 +STR(FN3559,3)+'           '+STR(FN60,3)
?'NUMBER OF RENEWALS       ',STR(FO1834,3)+'
 +STR(FO3559,3)+'           '+STR(FO60,3)
?
?
?
?
?
?
?
?'MEMBERS (TEENS)          12-14          15-17 '
?'-------------------------------------------'
?'NUMBER ON REGISTER       ',STR(F1214,3)+'          '+STR(F1517,3)
?'NUMBER OF NEW MEMBERS    ',STR(FN1214,3)+'
 +STR(FN1517,3)
?'NUMBER OF RENEWALS       ',STR(FO1214,3)+'
 +STR(FO1517,3)
?
?
?
?
?
?
?
?
```

```
?'MEMBERS (GIRLS)          UNDER 6          6-11'
?'-----------------------------------------------'
?'NUMBER ON REGISTER       ',STR(F6,3)+'              '+STR(F611,3)
?'NUMBER OF NEW MEMBERS    ',STR(FN6,3)+'                    '
 +STR(FN611,3)
?'NUMBER OF RENEWALS       ',STR(FO6,3)+'                    '
 +STR(FO611,3)
?
EJECT
?
?
?
?
?
?
?
?'NONMEMBERS (MEN)        18-34         35-59         60+'
?'-----------------------------------------------------'
?'NUMBER ON REGISTER      ',STR(M1834,3)+'                  '
 +STR(M3559,3)+'          '+STR(M60,3)
?'NUMBER OF NEW MEMBERS   ',STR(MN1834,3)+'                 '
 +STR(MN3559,3)+'         '+STR(MN60,3)
?'NUMBER OF RENEWALS      ',STR(MO1834,3)+'                 '
 +STR(MO3559,3)+'         '+STR(MO60,3)
?
?
?
?
?
?
?
?'NONMEMBERS (TEENS)       12-14            15-17'
?'-----------------------------------------------'
?'NUMBER ON REGISTER       ',STR(M1214,3)+'                  '
 +STR(M1517,3)
?'NUMBER OF NEW MEMBERS    ',STR(MN1214,3)+'                 '
 +STR(MN1517,3)
?'NUMBER OF RENEWALS       ',STR(MO1214,3)+'                 '
 +STR(MO1517,3)
?
?
?
?
?
?
?
?'NONMEMBERS (BOYS)           UNDER 6          6-11'
?'-----------------------------------------------'
?'NUMBER ON REGISTER    ',STR(M6)+'      '+STR(M611)
```

```
?'NUMBER OF NEW MEMBERS',STR(MN6)+'      '+STR(MN611)
?'NUMBER OF RENEWALS    ',STR(MO6)+'      '+STR(MO611)
?
?
EJECT
?
?
?
?
?
?
?
?
?'RACIAL/ETHNIC GROUP                    TOTAL'
?'---------------------------------------'
?'ASIAN AMERICAN                          ',STR(ASIAN+MASIAN,3)
?'NATIVE AMERICAN                         ',STR(NATIVE+MNATIVE,3)
?'BLACK                                   ',STR(BLACK+MBLACK,3)
?'HISPANIC                                ',STR(HISPAN+MHISPAN,3)
?'WHITE                                   ',STR(WHITE+MWHITE,4)
?'OTHER                                   ',STR(OTHER+MOTHER,3)
?
?
?
?
?
?'ZONES              TOTAL'
?'-------------------------'
?'ZONE 1             ',STR(ZONE1+MZONE1,3)
?'ZONE 2             ',STR(ZONE2+MZONE2,3)
?'ZONE 3             ',STR(ZONE3+MZONE3,3)
?'ZONE 4             ',STR(ZONE4+MZONE4,3)
?'ZONE 5             ',STR(ZONE5+MZONE5,3)
?'ZONE 6             ',STR(ZONE6+MZONE6,3)
?'ZONE 7             ',STR(ZONE7+MZONE7,3)
EJECT
SET PRINT OFF
CLOSE DATABASES
RETURN TO MASTER
```

AN EXERCISE IN DATABASE CUSTOMIZED PROGRAMMING TO COMPARE THE SMART DATA MANAGER AND dBASEIII

by

AMY LYNN FITZGERALD

B. S., Kansas State University, 1983

AN ABSTACT OF A MASTER'S REPORT

submitted in partial fulfillment of the

requirements for the degree

MASTER OF SCIENCE

Department of Industrial Engineering

KANSAS STATE UNIVERSITY
Manhattan, Kansas

1985

ABSTRACT

Every organization must arrange information so that it may be retrieved when necessary. Database technology is one method to handle this task. The purpose of this master's report is to evaluate the database management system known as the Smart Data Manager. As a means for comparison, dBaseIII is chosen as the market standard of microcomputer database software. The methodology used involves the logical design, its adjustment to meet the constraints of the software, implementation of the design, and comparisons. The logical design includes specification of data items needed and processes to manipulate the data in order to achieve the needed results. The adjustment of the logical design to meet the constraints of the system is known as the physical design. Once adjustments have been made, each language is implemented via coding. The comparisons, both objective and subjective, are completed next, and then conclusions are made. When making comparisons, it is found that the Smart Data Manager and dBaseIII are objectively similar. However, subjective differences exist. The Smart Data Manager executes slightly slower than dBaseIII; but its ease in use, due to the fact that it is menu-driven, may prove to be more important to the infrequent or novice user.