QUERY/45
User's Guide

Part No. 09845-91056
Microfiche No. 09845-98056

Hewlett-Packard Desktop Computer Division
3404 East Harmony Road, Fort Collins, Colorado 80525
(For World-wide Sales and Service Offices see back of manual.)

Important

The tape cartridge or disc containing the programs is very reliable, but being a mechanical device, is subject to wear over a period of time. To avoid having to purchase a replacement medium, we recommend that you immediately duplicate the contents of the tape onto a permanent backup tape or disc. You should also keep backup copies of your important programs and data on a separate medium to minimize the risk of permanent loss.

Table of Contents

Chapter 3: Data Base Concepts

Chapter 4: Basic Operations

Chapter 5: Retrieving Data Meeting Certain Criteria

Chapter 6: Changing the Data in the Data Base

Chapter 7: Using Forms with your Data Base

Chapter 8: Defining a Data Base

Chapter 9: Linking Other Subprograms to QUERY/45

Chapter 10: Using the Formal Command Language

APPENDIX C: The "ICAT" Catalog Data Base

APPENDIX D: Tables

Printing History

New editions of this manual will incorporate all material updated since the previous edition. Update packages may be issued between editions and contain replacement and additional pages to be merged into the manual by the user. Each updated page will be indicated by a revision date at the bottom of the page. A vertical bar in the margin indicates the changes on each page. Note that pages which are rearranged due to changes on a previous page are not considered revised.

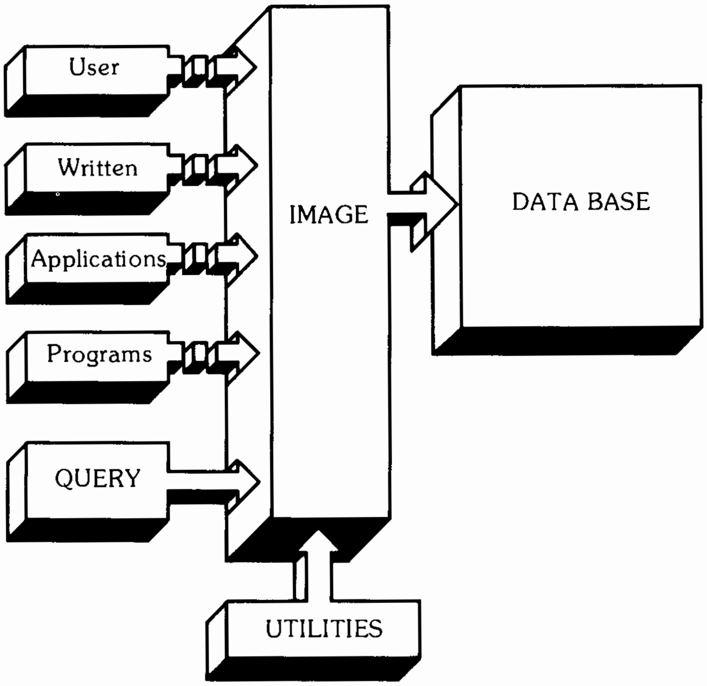
The manual printing date and part number indicate its current edition. The printing date changes when a new edition is printed. (Minor corrections and updates which are incorporated at reprint do not cause the date to change.) The manual part number changes when extensive technical changes are incorporated.

June 1980...First Edition
March 1981...Second Editon, Updated pages: 5, 6, 9, 13, 14, 75-78, 80, 97, 98, 100-106, 112
(Chapter 9 renumbered)

Chapter 1
General Information

What is QUERY/45?

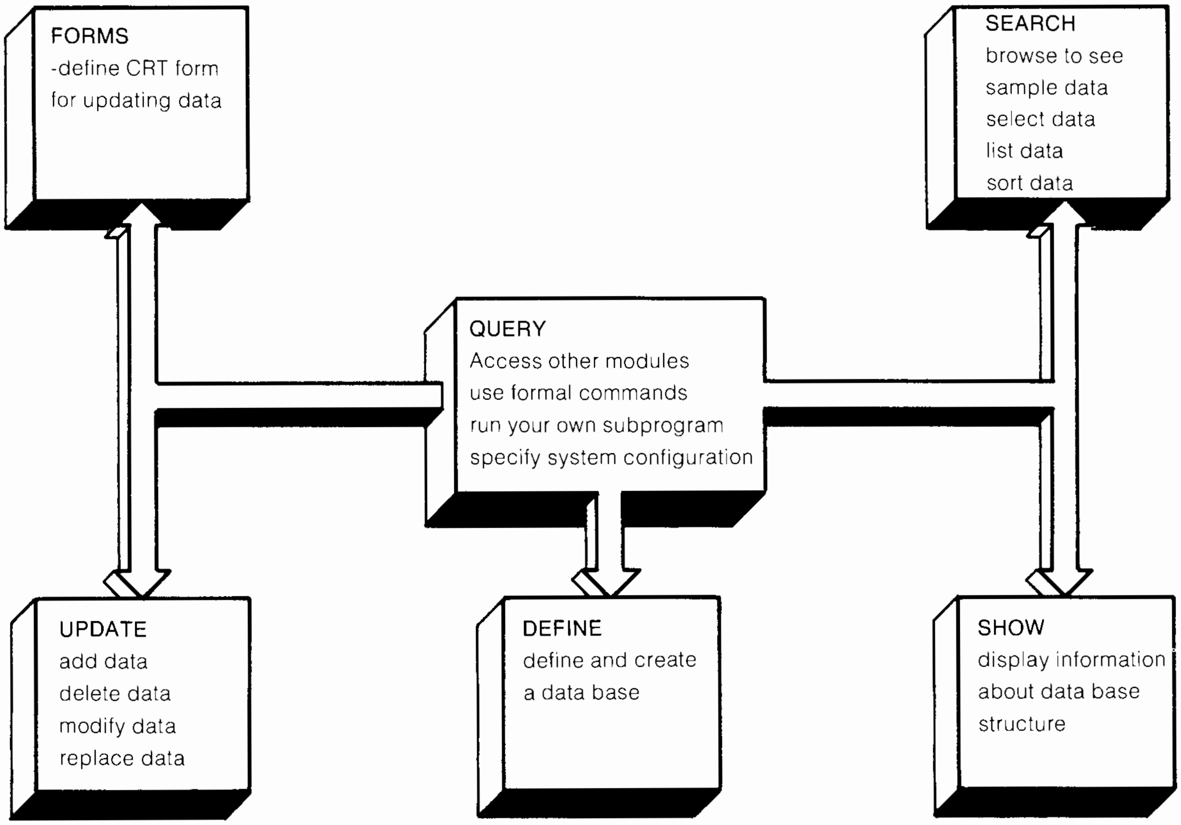
The HP System 45 Data Base Management System enables you to set up, access and maintain a data base. A **data base** is a group of logically related files containing data and information about how that data is interrelated.



QUERY/45 is part of the Data Base Management System that enables you to perform data base management operations without having to write a program. IMAGE/45 integrates the data files for an application and relieves you of the task of data organization and management. The Utility programs provide an easy means for maintaining data bases.

QUERY/45 is a BASIC language program which enables you to define a data base, put information into the data base and query or ask for information from the data base, thus saving the time it takes to write and debug a program.

QUERY/45 consists of five subsystems — SHOW, SEARCH, UPDATE, FORMS, and DEFINE — which are linked together and accessed through a central subsystem called QUERY. The following diagram represents QUERY/45 and gives a brief description of each subsystem.



How QUERY/45 Works

The various features of QUERY/45 are accessed by pressing **softkeys** (the Special Function Keys 0 through 15). At any particular point in the program, the available options (called a **menu**) are displayed and are accessed by pressing the corresponding key. The EXIT softkey is used to return to the previous menu.

The HELP Softkey

If you are unfamiliar with the options while using QUERY/45, press the HELP softkey. This prints an explanation of the current options.

Data Base Management Steps

There are six basic procedures involved in using QUERY/45 for data base management.

1. Become Familiar with QUERY/45

Chapter 2 contains step-by-step example operations to familiarize you with QUERY/45.

2. Understand Data Base Concepts and Terms

It is essential to understand the basic concepts and terms in order to use QUERY/45 effectively. Chapter 3 describes these concepts.

3. Design the Data Base

If you are going to use a data base of your own, rather than using the data base of someone else in your organization, you must convert your application needs into an design that QUERY/45 can understand. This is the most important step in using your data base management system. The Data Base Design Kit provides procedures and tools to help you do this.

4. Define and Create the Data Base

Once you have designed your data base, the DEFINE subsystem is used to describe the data base to QUERY/45. The DEFINE subsystem is also used to create the data base once it is defined. Chapter 8 covers the DEFINE subsystem.

5. Access the Data Base

Once the data base is created, data can be stored, retrieved and modified. The UPDATE and SEARCH subsystems are used to perform these operations. If you are unfamiliar with a data base, the SHOW subsystem provides information about the structure of the data base. These subsystems are introduced in Chapter 2 and are covered in more detail in Chapters 4, 5 and 6. In addition, CRT forms can be used to enhance data entry and updating. The FORMS subsystem is covered in Chapter 7.

6. Maintain the Data Base

Backing up your data base should be a regular part of your data base operations. This information can be found in Chapter 2.

Advanced Operations

QUERY/45 enables you to perform more advanced operations such as linking in your own subprogram using the RUN USER PROGRAM feature. If you are very familiar with various QUERY/45 operations, you can use the FORMAL COMMAND mode to bypass the softkeys and type in the operations directly. These are covered in Chapters 9 and 10.

The Sample Data Base

Throughout this manual, a sample data base is used in the examples. This sample data base (called LIBR) was shipped with your Data Base Management System and enables you to become familiar with and perform the QUERY/45 operations before you create your own data base.

The Scenario

The NOP Manufacturing Company has eight plants. Each plant has a small library of reference books. The books may be checked out by any employee from any plant and kept as long as needed.

NOP would like to use a System 45 to keep the following information:

- Information about each book (title, author, call number[1], publication date, publisher, cost, and subject).
- A list of every book, identifying which plant owns it.
- Whether or not a book has been checked out, and when.
- Which employee has borrowed a book (including name, employee identification number, department and phone number).
- General information about each library branch (plant name and address, librarian and phone r.umber).
- Classification of each book by subject.

All of this information has been arranged logically and stored on a disc to become your LIBR Data Base. Books can be found by the author, subject, title, or call number. The call number can be used to determine whether or not a book is checked out (and to whom) and if there are any additional copies available. Additionally, all books on a particular subject can be found. Exercises using this sample data base are presented in Chapter 2.

The detailed information about the LIBR data base is:

data base name: LIBR
root file volume name: LIBRARY
passwords: LIBRMGR (READ/WRITE access to all sets)
ENGINEER (READ-ONLY access to all sets)
maintenance word: BOOKS

[1] Ϝ. simple numbering scheme is used as a call number to identify each unique author-title combination in the library. This scheme is not related to any particular library numbering system.

Getting Started

Before using QUERY/45, you need to perform these three operations:

1. Install the IMAGE/45, Mass Storage and Advanced Programming ROMs, as covered in your HP 9845 Installation, Operation and Test Manual.
2. Transfer the QUERY/45 program from the tape cartridges to a disc.
3. Recover the backup version of the sample data base and its Information file, putting them on a disc.

Steps 2 and 3 are discussed in detail in the following sections.

Transferring QUERY/45 from Tapes to a Disc

QUERY/45 comes to you on three tape cartridges, but it must be on a disc to use it. A program is provided to transfer it for you. If you are using an HP 9885 Flexible Disc Drive as your mass storage device, you need **two** initialized discs to contain QUERY/45 since it is too large to fit on one. If you are using an HP 9895 Flexible Disc Drive or an HP 7900-series hard disc, you need only one initialized platter.

Before tranferring QUERY/45, ensure that your mass storage device is properly connected to the System 45 and that they are both turned on. Then follow these steps as the program prompts you.

1. Insert the tape cartridge labeled 'System 45 DBMS – QUERY/45, English Tape #1' into the righthand tape drive.
2. Type LOAD "QCOPY:T15" and press EXECUTE.
3. When the program is loaded (the run light in the lower right corner of the CRT goes out), press RUN.
4. Enter the mass storage unit specifier of the disc drive on which you are storing QUERY/45, then press CONTINUE. :F8 is an example mass storage unit specifier for a 9885.
5. Enter the volume label to be printed on your disc, then press CONTINUE. The program suggests that you use the label "QUERY/45", but you can enter a different one.

Note

The program now checks to see if there is already a version of QUERY/45 on the disc. If there is, the old version is purged.

6. The files on Tape #1 are then transferred to the disc. When the transfer is complete, you are asked to insert QUERY/45 Tape #2 into the right-hand tape drive and press CONTINUE.
7. When this information is transferred, you are asked to insert QUERY/45 Tape #3 into the right-hand drive and press CONTINUE. If you are using a 9885 Flexible Disc, you are asked to insert the second disc into the disc drive and press CONTINUE.
8. When all of QUERY/45 is transferred, 'QUERY/45 transferred; QCOPY terminated.' is displayed.

Transferring the Sample Data Base to a Disc

To transfer the sample data base LIBR from the tape cartridge to a disc, you use a program on the Utilities tape called TBKUP to RECOVER the data base.

Labeling your Disc

If you are using a hard disc, you may put the sample data base on the same disc as QUERY/45. If you are using a **flexible disc** or you want to put LIBR on a **different hard disc**, you should label [1] the disc first. It is suggested that you use the label LIBRARY. To label the disc, type the following, then press EXECUTE.

```
PRINT LABEL "LIBRARY" ON "msus"
```

The msus specifies the type and location of the disc drive; :F8 is an example.

Directions

1. Insert the tape cartridge containing the sample data base into the lefthand tape drive (:T14) and insert the Utilities tape cartridge into the righthand tape drive (:T15).
2. Type LOAD "TBKUP:T15" and press EXECUTE.
3. After the program is loaded (the run light in the lower right corner of the CRT goes out), press RUN.
4. The program asks you for various information. Enter the following in response to the questions.

- 2 – to indicate recovery.
- CONTINUE – to indicate a single-volume data base.
- LIBR – the name of the data base.
- LIBRBK:T15 – the name and location of the backup file.
- msus – the mass storage unit specifier of the disc on which you want to place the data base. :F8 is an example.
- Y – to indicate there is no previous version of the data base.
- Y – to recover the QUERY/45-related files after the root file and data files are recovered.

[1] An explanation of volume labels is found in Chapter 8

Running QUERY/45

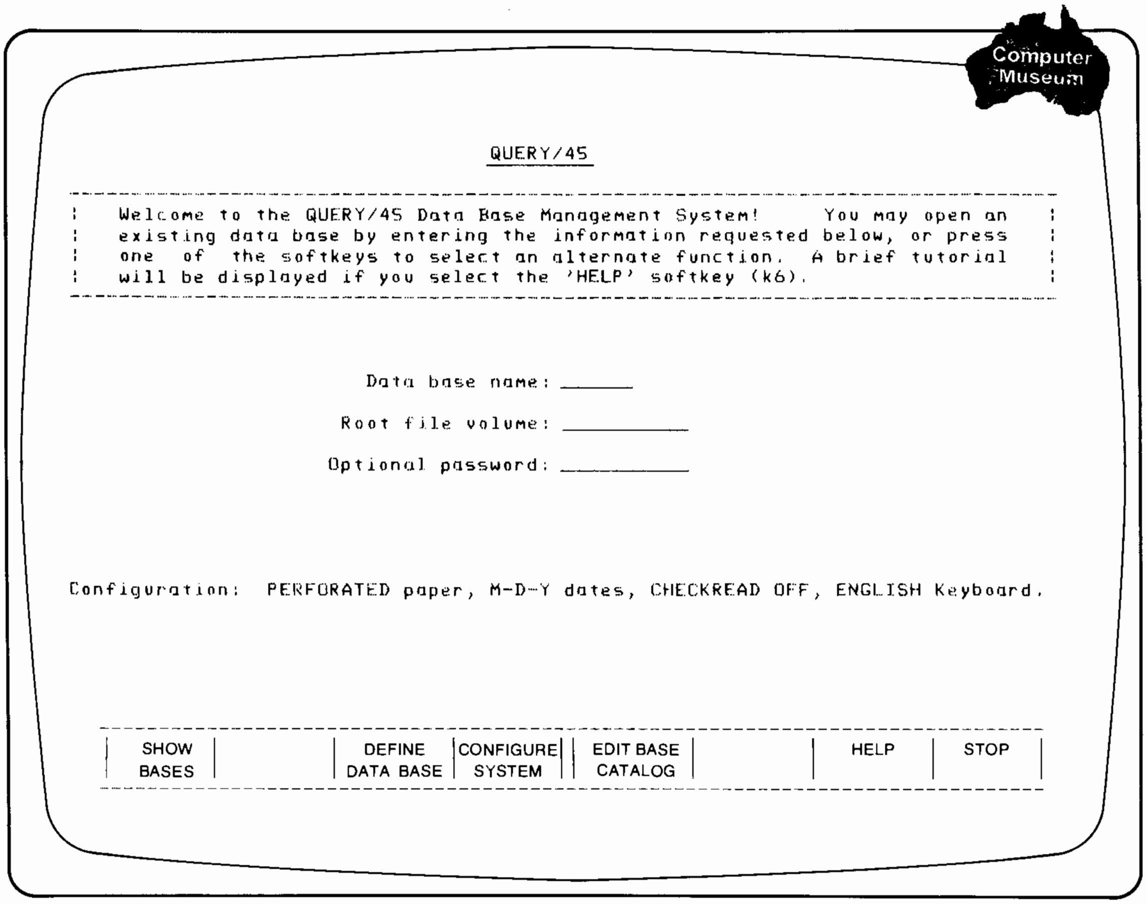
To use QUERY/45, you need to load the program into the System 45 and run it. To load QUERY/45 make sure that the disc with QUERY/45 is inserted into a mass storage device, then type the following and press EXECUTE:

```
MSI ":msus"
```

(MSI is used instead of MASS STORAGE IS) then type:

```
LOAD "QUERY"
```

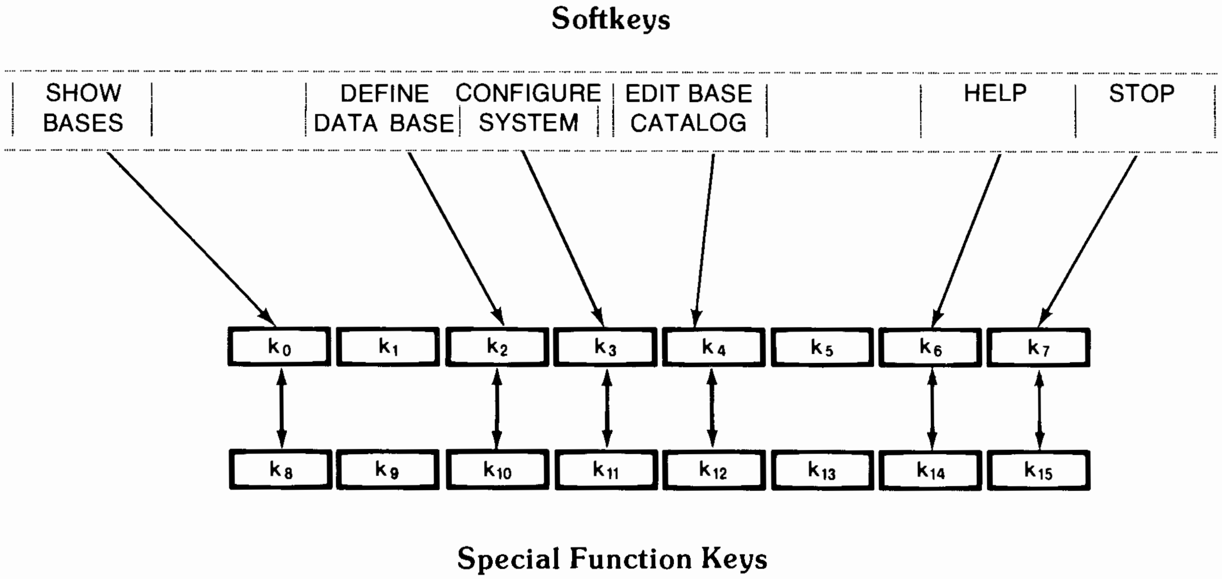
The mass storage unit specifier (msus) indicates the type of disc and where it is located. For example, type `MSI ":F8"` if you are using a 9885 Flexible Disc Drive with the select code 8. For more information refer to the 9845 Operating and Programming manual. After QUERY/45 is loaded, press the RUN key. The program is ready when you see the following displayed on the CRT.



```
                              QUERY/45

-------------------------------------------------------------------------------
:   Welcome to the QUERY/45 Data Base Management System!     You may open an   :
:   existing data base by entering the information requested below, or press   :
:   one  of  the softkeys to select an alternate function.  A brief tutorial   :
:   will be displayed if you select the 'HELP' softkey (k6).                    :
-------------------------------------------------------------------------------


                       Data base name: ______

                     Root file volume: __________

                    Optional password: __________


Configuration:  PERFORATED paper, M-D-Y dates, CHECKREAD OFF, ENGLISH Keyboard.
```

The Softkeys

The softkeys are used to access the features of QUERY/45 and are located at the bottom of the CRT and relate to the Special Function Keys on the keyboard, as shown here:



Softkeys

Special Function Keys

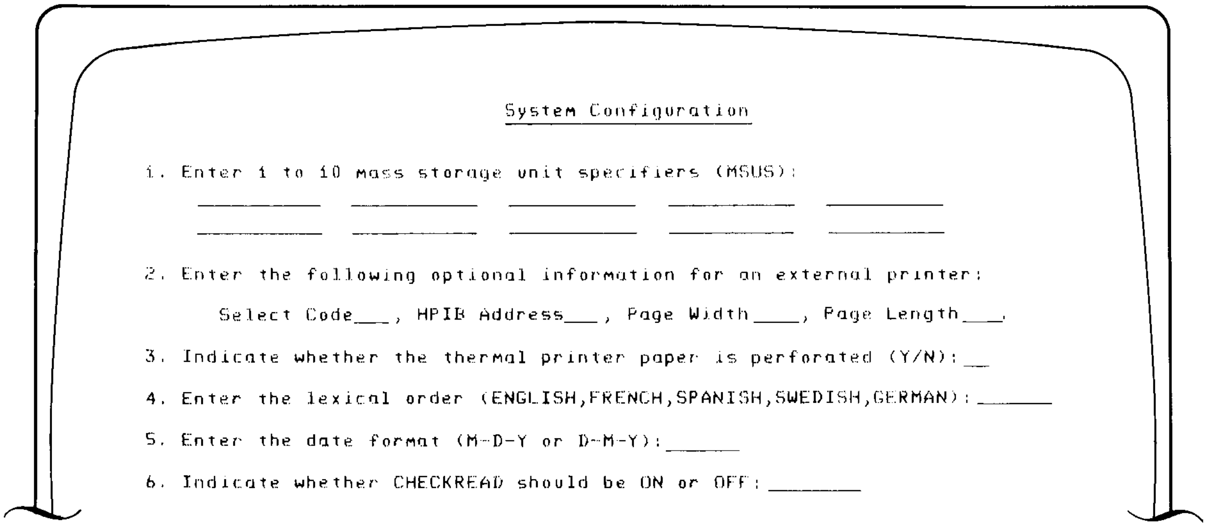
k8 through k15 are represented by the same softkey functions as k0 through k7; using the second row of Special Function Keys is a matter of preference.

Entering Information

While you are using QUERY/45, press the CONTINUE, STORE or the TAB key to enter information into the system as it is requested.

Configuring your System

If you are running QUERY/45 for the first time, you must specify the configuration of your system, such as the location of the optional printer. This should also be done any time you change the configuration. To do this, press the CONFIGURE SYSTEM softkey. The CRT now looks like:

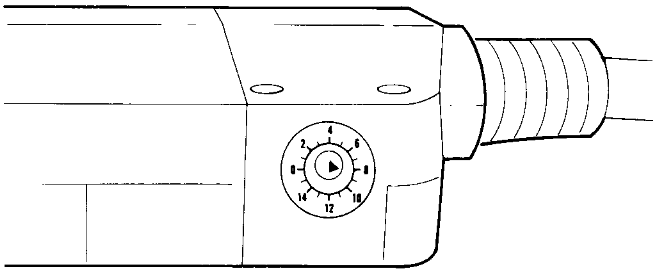


```
                    System Configuration

 1. Enter 1 to 10 mass storage unit specifiers (MSUS):
     _________   _________   _________   _________   _________
     _________   _________   _________   _________   _________

 2. Enter the following optional information for an external printer:
       Select Code___, HPIB Address___, Page Width____, Page Length___.

 3. Indicate whether the thermal printer paper is perforated (Y/N):__

 4. Enter the lexical order (ENGLISH,FRENCH,SPANISH,SWEDISH,GERMAN):______

 5. Enter the date format (M-D-Y or D-M-Y):______

 6. Indicate whether CHECKREAD should be ON or OFF:________
```

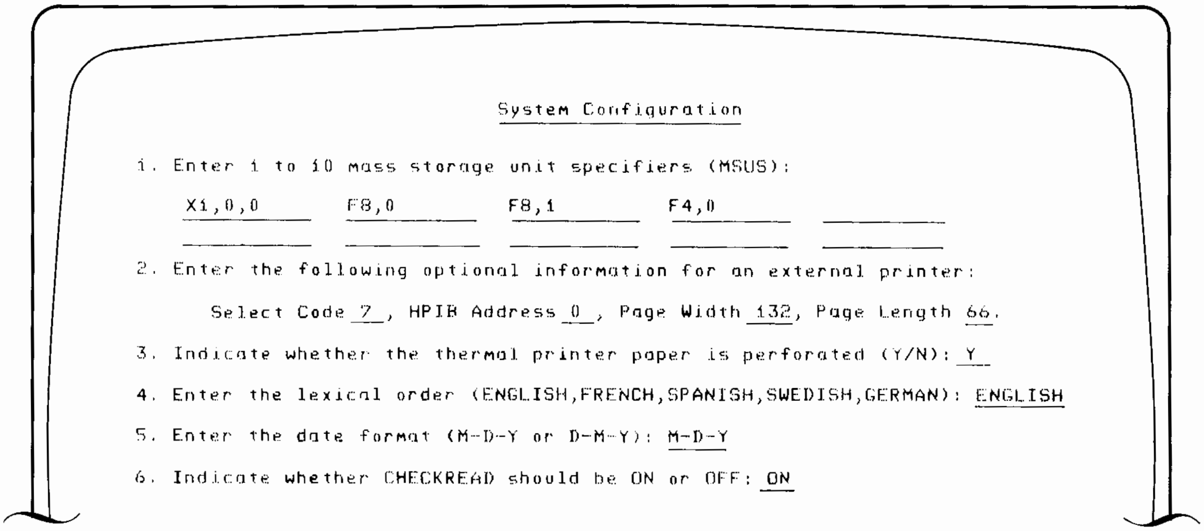
The first step is to indicate the mass storage devices that you are using with your system, including the disc which contains the sample data base. The table shows the device type for five of the more common devices.

HP Device	Device Type
9885 M/S Flexible Disc	F
9895 M/S Flexible Disc	H
7906 Removable Disc	C
7906 Fixed Disc	C
7920 M/S Disc Pack	P
7925 Fixed Disc	X

In addition to the device type, you must specify the select code currently set on the interface card. For example, the following interface card has the select code of 8.



Next, you can specify an optional printer for your output, other than the internal printer. For its page length, specify the total number of lines per page. For example, enter 66 for 6 lines per inch on 11-inch paper. This includes top and bottom margins. The third line on the screen indicates if the internal printer has perforated paper (type Y or N). Line 4 specifies which keyboard you are using while working with QUERY/45 (ENGLISH, FRENCH, SPANISH, SWEDISH, GERMAN). Additionally, you must choose between month, day, year (M-D-Y) or day, month, year (D-M-Y) for a date format on line 5. This enables QUERY/45 to correctly interpret dates you enter. (This is associated with the Date type data item.) The last line specifies error-checking on disc accesses and should perferably be "ON" if you are using flexible discs, as shown in the example.



```
                              System Configuration

1. Enter 1 to 10 mass storage unit specifiers (MSUS):

   X1,0,0      F8,0        F8,1        F4,0        __________

   __________  __________  __________  __________  __________

2. Enter the following optional information for an external printer:

     Select Code 7 , HPIB Address 0 , Page Width 132, Page Length 66.

3. Indicate whether the thermal printer paper is perforated (Y/N): Y

4. Enter the lexical order (ENGLISH,FRENCH,SPANISH,SWEDISH,GERMAN): ENGLISH

5. Enter the date format (M-D-Y or D-M-Y): M-D-Y

6. Indicate whether CHECKREAD should be ON or OFF: ON
```

When all the information is entered, record it by pressing the STORE CONFIG softkey.

The System Configuration information needs to be entered only once unless you add or remove mass storage devices or change any other information on this display. When you need to edit the System Configuration screen, the CONFIGURE SYSTEM softkey is found at the beginning of QUERY/45. If the data base is already opened, the CONFIGURE SYSTEM softkey is found after pressing MORE KEYS softkey.

Chapter 2

Getting Acquainted with QUERY/45

Introduction

The purpose of this chapter is to enable you to become familiar with the major capabilities of QUERY/45. This is done using a series of exercises which consist of simple operations for you to perform. Important data base management terms are also introduced as needed. After performing these exercises, you should be familiar enough with QUERY/45 to use it for the majority of data base operations. Once you are familiar with the information in this chapter, you can refer to the remaining chapters for more detailed information about how QUERY/45 works.

After the exercises, there is a section on backing up the sample data base. Backup is an essential operation; you should perform the backup procedure as indicated. The last section of the chapter introduces the other capabilities of QUERY/45.

Performing the Exercises

Each exercise contains step-by-step instructions. In general, when you are asked to enter the specified data, you should use the CONTINUE key to enter it. Any special instructions are included with the exercise.

Using the Sample Data Base

The exercises in this chapter use the sample data base, LIBR, which is provided with your Data Base Management System package. These exercises enable you to perform data base operations using actual data. To perform the exercises correctly, the LIBR data base must be in exactly the same form as it was when it was shipped to you on the tape cartridge. If other people have performed these exercises or the LIBR data base contents have been changed in any way, follow these steps to restore the LIBR data base to its original state:

1. Use the backup version of the LIBR data base. This version was supplied on a tape cartridge with your Data Base Management System package. You may also use a copy of this, if one was made.
2. Since a disc may contain only one copy of a data base having the same name, you may not place this new version of the data base on the same disc as the altered version. In this case, you have two choices. The first is to place the new data base on another disc. This may not be possible if you are using a hard disc, so the second choice is to purge the altered version from the disc. To do this, type:

   ```
   DBPURGE "LIBR:msus";"BOOKS"
   ```

   and press EXECUTE.
3. Follow the steps listed in Chapter 1 under Transferring the Sample Data Base, to perform the recovery.

Loading QUERY/45

Before you can access a data base, you must load QUERY/45 into the System 45. To do this, insert the QUERY/45 disc into a mass storage device and type:

```
MSI ":msus"
```

(MSI may be used in place of MASS STORAGE IS) and press EXECUTE.

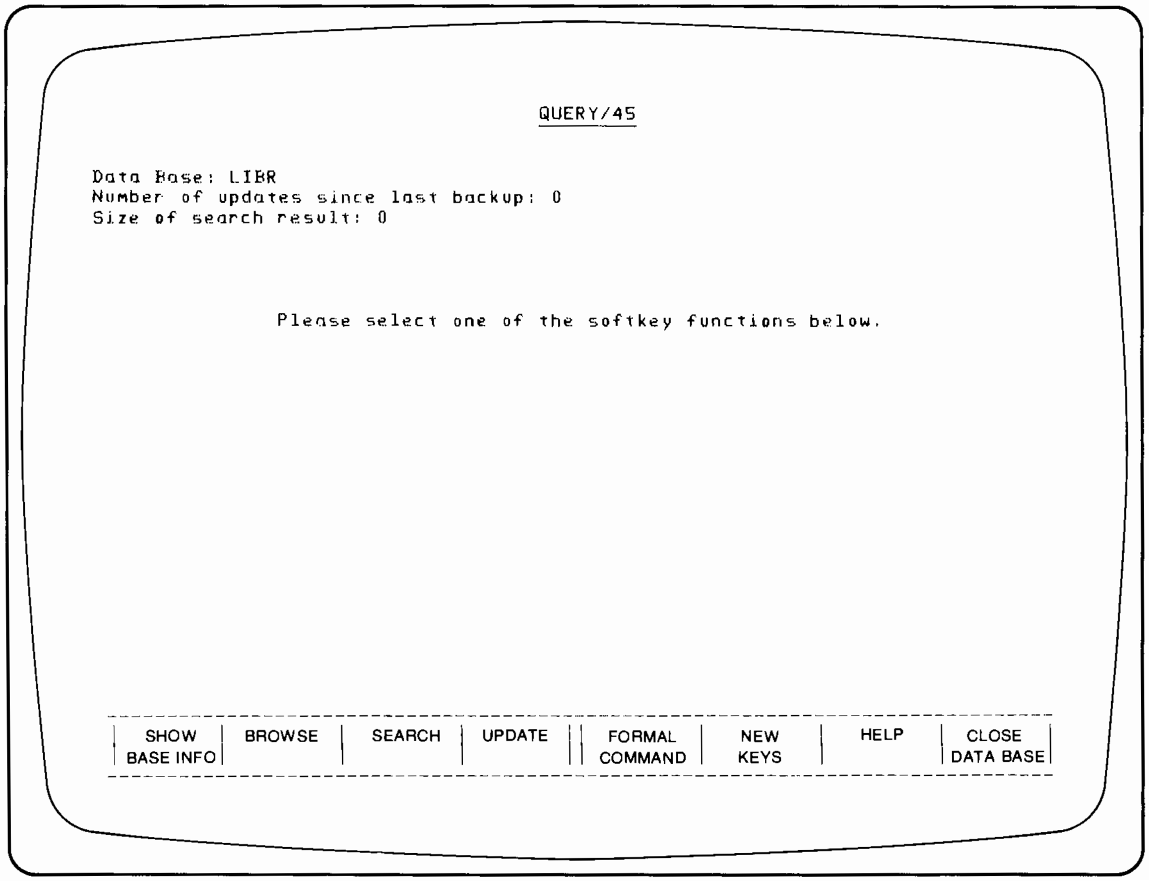
Now type:

```
LOAD "QUERY"
```

and press EXECUTE. Press RUN after the run light goes out.

Accessing a Data Base

In using QUERY/45 to access a data base the first step is to open it. To open the LIBR data base, type `LIBR` for the data base name, the volume name for your medium, and `LIBRMGR` for the password, pressing CONTINUE after each one. Notice that when you typed the password, it was not printed on the screen. This is to ensure that only authorized people know the password of the data base. Now that the LIBR data base is open, the CRT displays:



```
                    QUERY/45

Data Base: LIBR
Number of updates since last backup: 0
Size of search result: 0


        Please select one of the softkey functions below.
```

Finding Out What the Data Base Looks Like

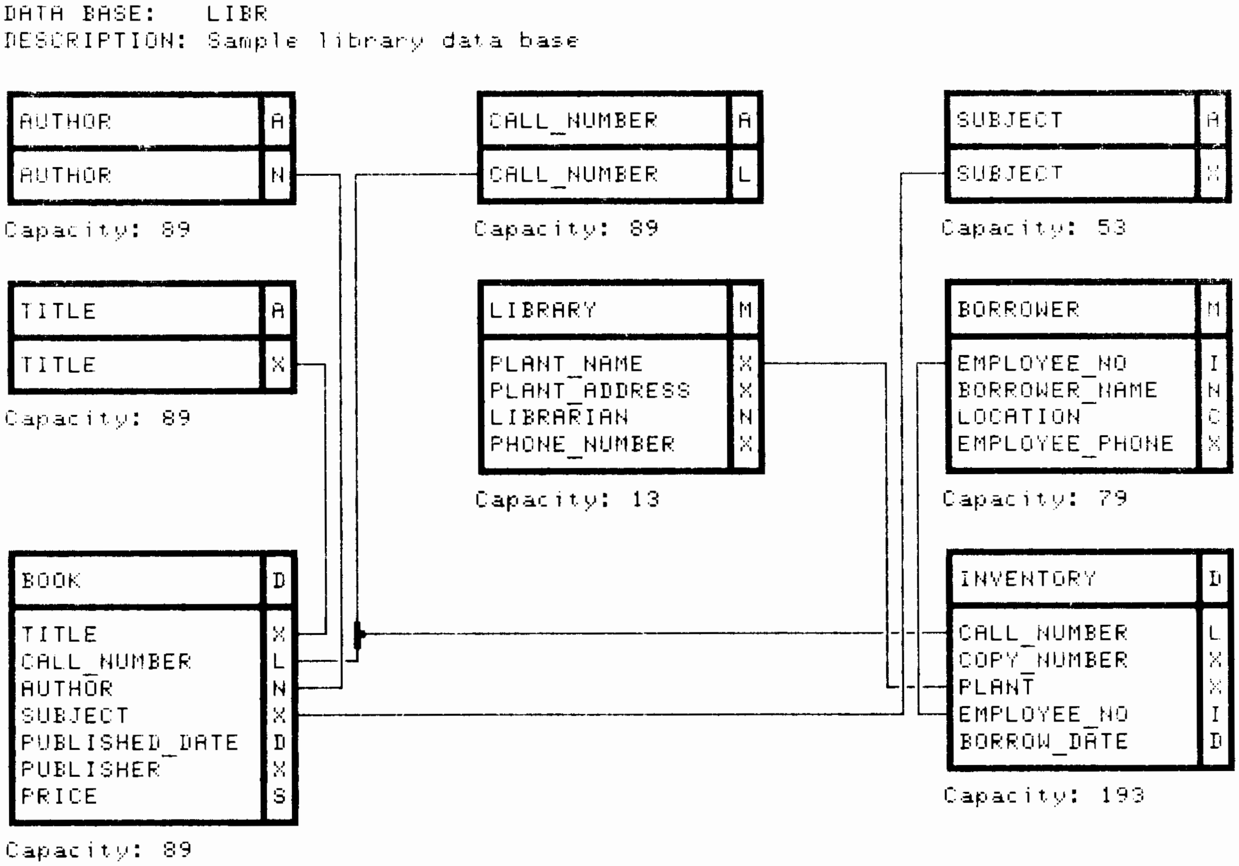
When you encounter a data base for the first time, you might want to know the structure of the data base and look at a few entries to see samples of the data base contents. By pressing the SHOW BASE INFO softkey you are able to display the data base in four different ways. Only two of the ways are discussed in this chapter, the remaining ways are discussed in Chapter 4.

1. Press the SHOW BASE INFO softkey (after the data base is opened).

Structure

First let's look at the structure of the LIBR data base. The softkey labeled GRAPHIC STRUCTURE prints a diagram of the data base on the printer. This diagram is useful during other data base operations.

1. Press the GRAPHIC STRUCTURE softkey and the LIBR data base is drawn as:



```
DATA BASE:   LIBR
DESCRIPTION: Sample library data base

AUTHOR           A     CALL_NUMBER      A     SUBJECT          A
AUTHOR           N     CALL_NUMBER      L     SUBJECT          X
Capacity: 89           Capacity: 89           Capacity: 53

TITLE            A     LIBRARY          M     BORROWER         M
TITLE            X     PLANT_NAME       X     EMPLOYEE_NO      I
Capacity: 89           PLANT_ADDRESS    X     BORROWER_NAME    N
                       LIBRARIAN        N     LOCATION         C
                       PHONE_NUMBER     X     EMPLOYEE_PHONE   X
                       Capacity: 13           Capacity: 79

BOOK             D                            INVENTORY        D
TITLE            X                            CALL_NUMBER      L
CALL_NUMBER      L                            COPY_NUMBER      X
AUTHOR           N                            PLANT            X
SUBJECT          X                            EMPLOYEE_NO      I
PUBLISHED_DATE   D                            BORROW_DATE      D
PUBLISHER        X                            Capacity: 193
PRICE            S
Capacity: 89
```

The diagram of the data base shows the data sets within the data base. A **data set** contains information about something, such as a BOOK or BORROWER. Each data set contains **data items** which describe the thing; TITLE and AUTHOR are two of the data items used to describe a book. The diagram of the data base also shows the **data paths** which connect the data sets to each other.

This diagram also shows the maximum number of entries that can be entered for each set, called **capacity**. For example, the BOOK data set can contain, at most, 89 books.

Data Sets

If you just want to know what data sets are in the LIBR data base, the SET INFO softkey gives you this information.

1. Type T to indicate that you want the set information listed on the internal printer.
2. Press the SET INFO softkey.

This is what is listed:

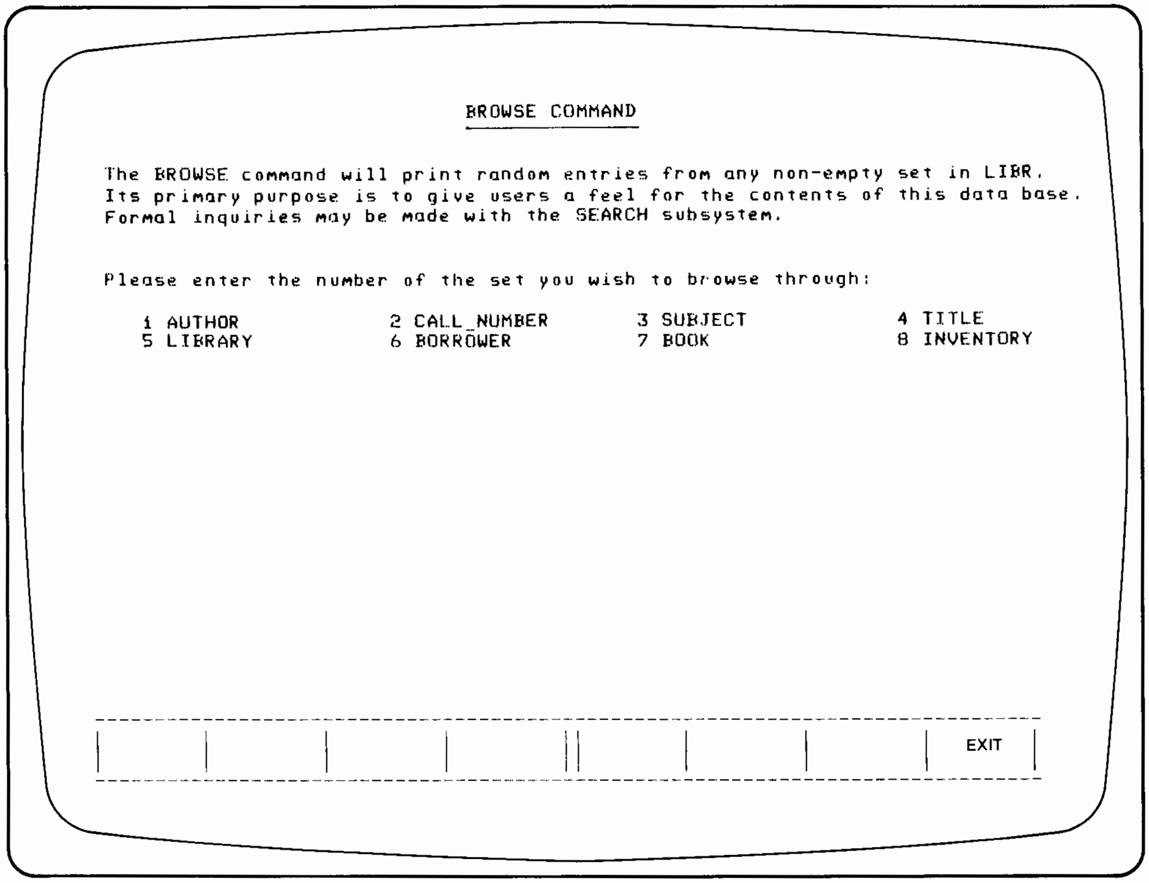
```
                 Set Information for Data Base LIBR

Set Name         Type            Volume     Entries  Capacity
-------------------------------------------------------------
AUTHOR           AUTO MASTER     LIBRARY       68        89
CALL_NUMBER      AUTO MASTER     LIBRARY       71        89
SUBJECT          AUTO MASTER     LIBRARY       45        53
TITLE            AUTO MASTER     LIBRARY       69        89
LIBRARY          MANUAL MASTER   LIBRARY        9        13
BORROWER         MANUAL MASTER   LIBRARY       61        79
BOOK             DETAIL          LIBRARY       69        89
INVENTORY        DETAIL          LIBRARY      148       193
```

3. Press the EXIT softkey to return to these softkeys:

```
| SHOW     | BROWSE | SEARCH | UPDATE | | FORMAL  | MORE | HELP | CLOSE     |
|BASE INFO |        |        |        | |COMMAND  | KEYS |      |DATA BASE  |
```

Browsing Through the Data

The BROWSE softkey enables you to view all entries in a particular data set one at a time. The primary purpose of BROWSE is to help you become familiar with the data in an unfamiliar data base. The BROWSE softkey is displayed after the data base is opened. After pressing it this screen is displayed:



```
                          BROWSE COMMAND

The BROWSE command will print random entries from any non-empty set in LIBR.
Its primary purpose is to give users a feel for the contents of this data base.
Formal inquiries may be made with the SEARCH subsystem.

Please enter the number of the set you wish to browse through:

   1 AUTHOR            2 CALL_NUMBER        3 SUBJECT          4 TITLE
   5 LIBRARY           6 BORROWER           7 BOOK             8 INVENTORY
```

You can now specify the set to be examined by entering its corresponding number. After you enter the appropriate set number and press CONTINUE, entries from that set are displayed on the screen. If an entry does not fit on one screen, you can see the rest of the entry by pressing the NEXT PAGE softkey.

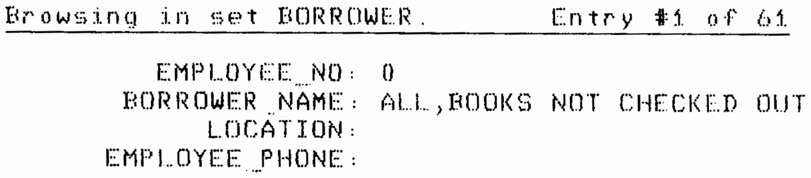
Exercise #1: Browsing Through the BORROWER Data Set

To browse through the BORROWER data set, you first must open the LIBR data base (as discussed previously), which displays these softkeys:

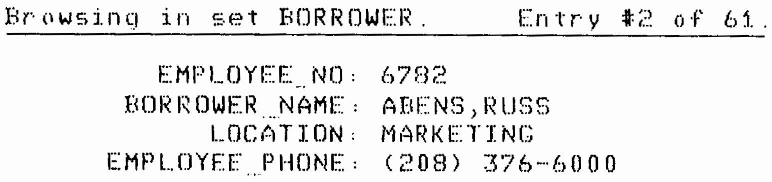
```
-------------------------------------------------------------------------------------
|   SHOW   |  BROWSE  |  SEARCH  |  UPDATE  ||  FORMAL  |  MORE  |   HELP   |   CLOSE   |
|BASE INFO |          |          |          ||COMMAND   |  KEYS  |          | DATA BASE |
-------------------------------------------------------------------------------------
```

1. Now press the BROWSE softkey.
2. Specify the BORROWER data set by typing the number 6 and pressing CONTINUE.
3. As entries appear on the screen, press the NEXT PAGE softkey to advance through the entries.

The first two entries displayed are:



```
Browsing in set BORROWER.         Entry #1 of 61
_____________________________________________________

      EMPLOYEE_NO: 0
    BORROWER_NAME: ALL,BOOKS NOT CHECKED OUT
         LOCATION:
   EMPLOYEE_PHONE:
```



```
Browsing in set BORROWER.       Entry #2 of 61.
___________________________________________________

      EMPLOYEE_NO: 6782
    BORROWER_NAME: ABENS,RUSS
         LOCATION: MARKETING
   EMPLOYEE_PHONE: (208) 376-6000
```

While browsing the BORROWER data set, you might notice that AMY HARRISON's name is in the data set and therefore, she has checked out a book. If you want to find out the particular books checked out by Amy, then the SEARCH subsystem is used as explained in the following section.

When you reach the eighth entry in the data set, press the ABORT softkey to leave the BORROWER data set. Now that you have seen the contents of the BORROWER set, BROWSE some other data sets. Use the DUMP SCREEN softkey to get a copy of the screen printed on the internal printer. When you are ready to get back to the beginning of QUERY/45, press the EXIT softkey.

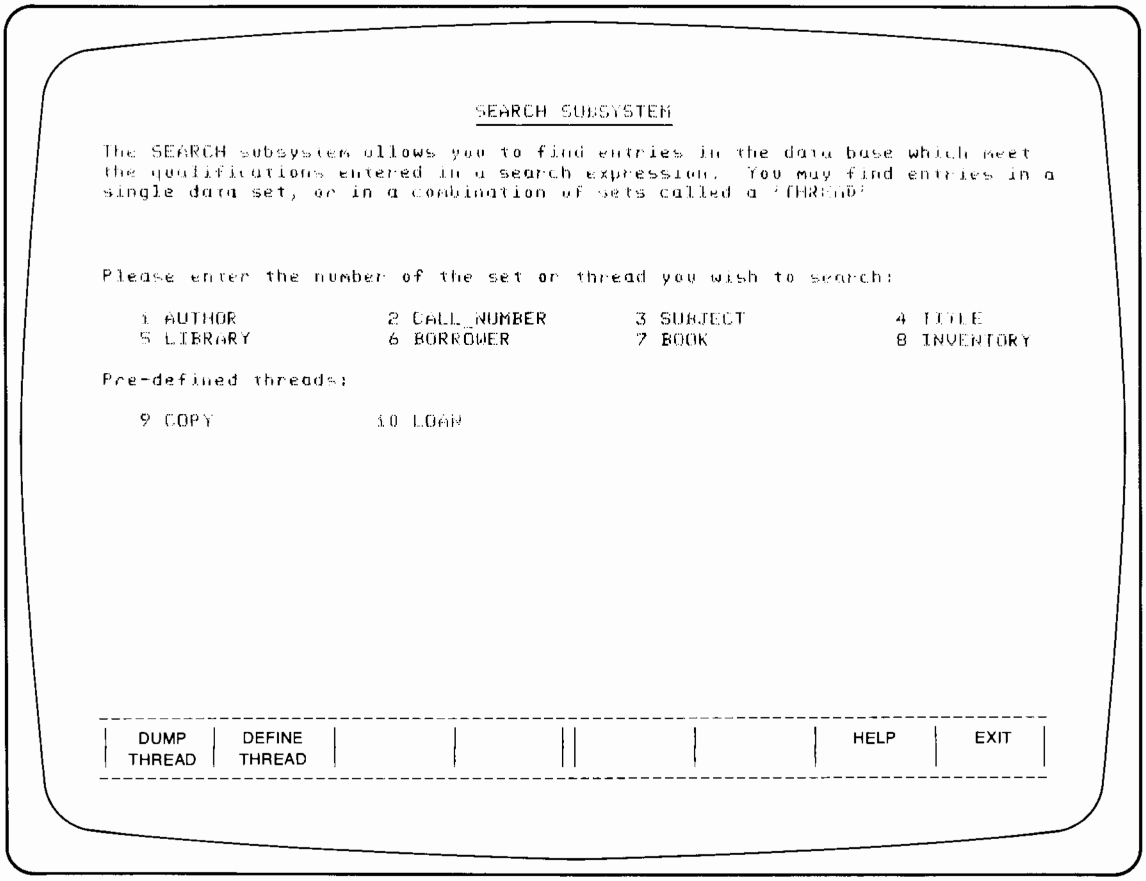
Retrieving Data Meeting Certain Criteria

Once you have used BROWSE to become familiar with the data in the data base, you can use the SEARCH softkey to retrieve selected data entries. A **search expression** specifies the criteria for selecting the entries. For example, you could retrieve all the books by the author William M. Newman by creating this search expression:

AUTHOR = WILLIAM M. NEWMAN

After selected data is retrieved, the SEARCH subsystem also enables you to sort and print this data.

The SEARCH softkey is displayed after the data base is opened. After pressing it, this screen is displayed:

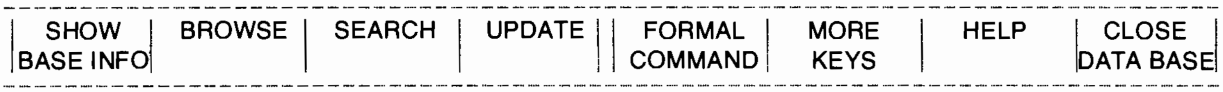


```
                              SEARCH SUBSYSTEM

The SEARCH subsystem allows you to find entries in the data base which meet
the qualifications entered in a search expression.  You may find entries in a
single data set, or in a combination of sets called a 'THREAD'


Please enter the number of the set or thread you wish to search:

    1 AUTHOR              2 CALL_NUMBER         3 SUBJECT            4 TITLE
    5 LIBRARY             6 BORROWER            7 BOOK               8 INVENTORY

Pre-defined threads:

    9 COPY               10 LOAN
```

You can now specify the data set you want to search. After you type the number that corresponds to a set, the screen is cleared and you can type the search expression. A search expression can be specified in many different ways; these are explained in Chapter 5.

Exercise #2: Searching for Books on Computer Programming

To search for books on computer programming, the LIBR data base must be opened and these softkeys displayed:



```
----------------------------------------------------------------------------------------
|   SHOW   | BROWSE |  SEARCH  |  UPDATE  ||  FORMAL  |  MORE  |   HELP   |  CLOSE   |
|BASE INFO|         |          |          || COMMAND  |  KEYS  |          |DATA BASE|
----------------------------------------------------------------------------------------
```

1. Press the SEARCH softkey.
2. Enter 7 to indicate that BOOK is the data set to be searched.
3. Press the PROMPT softkey
4. Type SUBJECT for the item name and press CONTINUE.
5. Type 1 to indicate the equal sign (=) and press CONTINUE.
6. Type COMPUTER PROGRAMMING for the value and press CONTINUE.

The search expression now looks like:

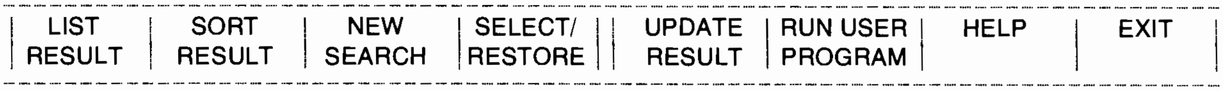
SUBJECT = COMPUTER PROGRAMMING

7. Type 3 to indicate that the search expression is complete and press CONTINUE.

When the search is complete, the screen displays:

Size of search result: 12

This indicates that 12 data entries were found with the SUBJECT = COMPUTER PROGRAMMING. Now that a search has been performed, these softkeys are displayed:

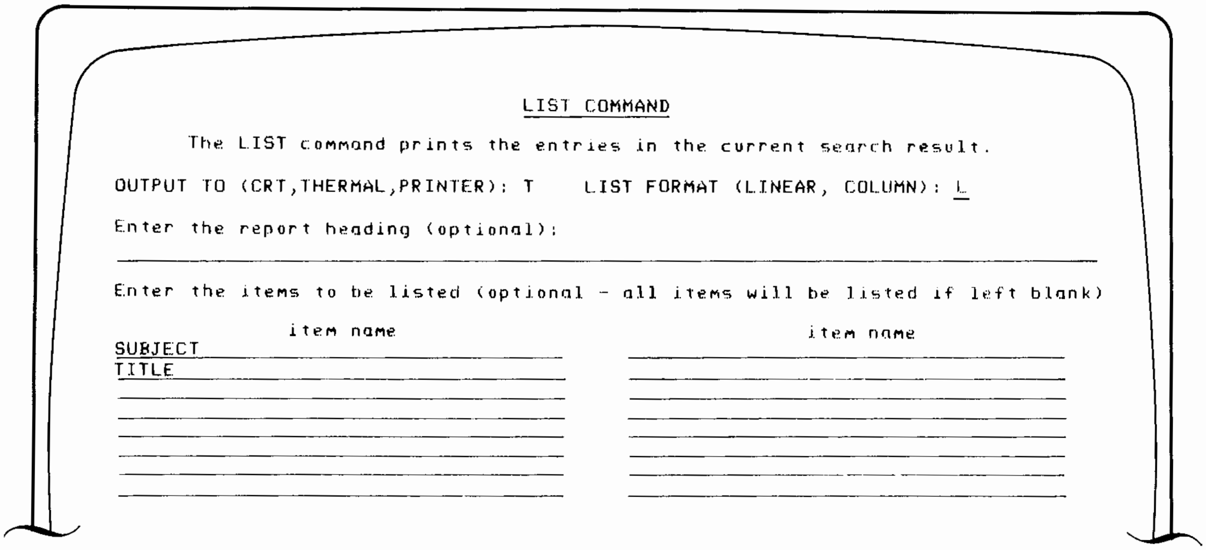


```
----------------------------------------------------------------------------------------
|  LIST  |  SORT  |  NEW   | SELECT/ ||  UPDATE  | RUN USER |   HELP   |   EXIT   |
| RESULT | RESULT | SEARCH |RESTORE  ||  RESULT  | PROGRAM  |          |          |
----------------------------------------------------------------------------------------
```

Exercise #3: Listing the Books on Computer Programming

In order to see the titles of the books found in the previous exercise:

1. Press the LIST RESULT softkey.
2. Fill in the LIST COMMAND screen as shown next. This causes the subjects and titles of the selected books to be printed.
3. Press the EXECUTE LIST softkey.
4. Press EXIT to get out of the LIST command.



```
                         LIST COMMAND

     The LIST command prints the entries in the current search result.

OUTPUT TO (CRT,THERMAL,PRINTER): T    LIST FORMAT (LINEAR, COLUMN): L

Enter the report heading (optional):
_____________________________________________________________________

Enter the items to be listed (optional - all items will be listed if left blank)

               item name                               item name
SUBJECT_____________________________     _____________________________
TITLE_______________________________     _____________________________
____________________________________     _____________________________
____________________________________     _____________________________
____________________________________     _____________________________
____________________________________     _____________________________
____________________________________     _____________________________
____________________________________     _____________________________
```

How many of the titles, out of the 12 data entries found, actually had the word COMPUTER in them? You should find four:

FUN AND GAMES WITH THE COMPUTER
TECHNIQUES IN COMPUTER PROGRAMMING
STANDARDIZED DEVELOPMENT OF COMPUTER SOFTWARE
THE PSYCHOLOGY OF COMPUTER PROGRAMMING

Listing the Data Entries

The LIST RESULT softkey enables you to examine all or selected entries found by the search. You can print the values in linear or columnar format with a optional heading printed at the top of the listing.

Exercise #4: Listing a Search Result

In this exercise, you will search the INVENTORY data set and find the entries indicating books checked out by employees with employee number 9905 or 2843. The resulting information is printed in columnar format.

1. Press the NEW SEARCH softkey.
2. Type 8 to indicate the INVENTORY data set.
3. For the search expression, type:

   ```
   EMPLOYEE_NO = 9905 OR EMPLOYEE_NO = 2843
   ```

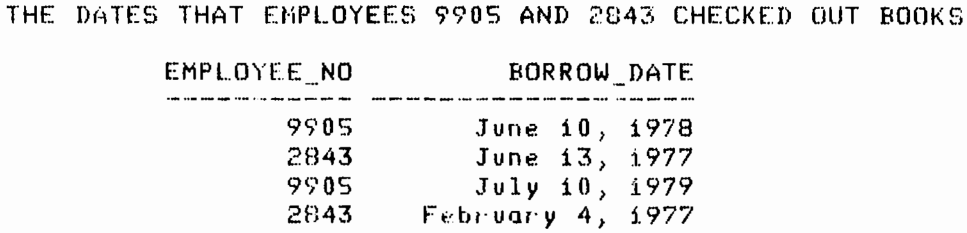
   and press CONTINUE.
4. When the search result is found (4 entries), press the LIST RESULT softkey to examine these entries.
5. Type C to indicate the listing to be on the CRT, and type C to indicate columnar format.

6. For the heading to be printed at the top of the pages, type the following in the HEADING field.

   THE DATES THAT EMPLOYEES 9905 AND 2843 CHECKED OUT BOOKS

7. Now type EMPLOYEE_NO and BORROW_DATE as the item values that are to be printed out.
8. Press the EXECUTE LIST softkey.

The listing looks like:



```
THE DATES THAT EMPLOYEES 9905 AND 2843 CHECKED OUT BOOKS

                EMPLOYEE_NO          BORROW_DATE
                -----------  -------------------
                       9905        June 10, 1978
                       2843        June 13, 1977
                       9905        July 10, 1979
                       2843     February 4, 1977
```

9. Press the EXIT softkey to get out of the LIST command

Sorting the Selected Data Entries

After a search has been performed, you may also sort the resulting entries into a certain order based on a particular item. The sort is performed in ascending order unless you specify descending order. Additionally, you can sort using more than one item. For example, the books found in Exercise #2 on computer programming could be sorted alphabetically by author and chronologically by publication date. If two or more books have the same author, these books would be sorted by publication date.

To perform a sort, press the SORT RESULT softkey which appears after a search has been performed. You are asked to enter the data items on which you want to sort. Remember that sorting works only on data entries that were found by the search.

Exercise #5: Sorting the Books in the BOOK Data Set

Now let's search for all the books in the BOOK data set that have a price less than $10.00, then sort these books by SUBJECT and AUTHOR in ascending order and print the result on the internal printer.

1. Press the NEW SEARCH softkey to indicate you want to perform a new search.

```
---------------------------------------------------------------------------------------
| LIST   |  SORT   |  NEW   | SELECT/ ||  UPDATE | RUN USER |   HELP   |    EXIT    |
| RESULT |  RESULT | SEARCH | RESTORE ||  RESULT | PROGRAM  |          |            |
---------------------------------------------------------------------------------------
```

2. Type 7 to specify that the BOOK data set is to be searched.

3. To find all the books with prices less than $10.00, type the search expression as:

   PRICE < 10.00

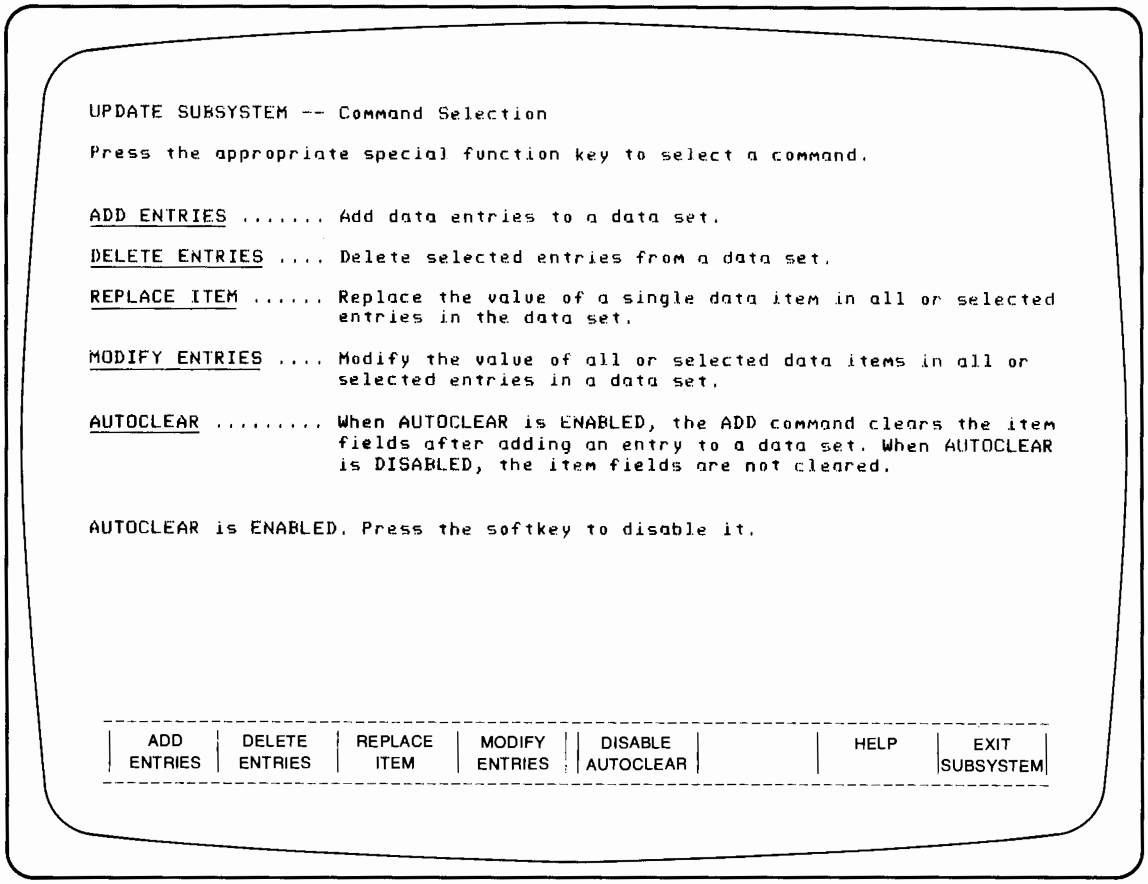
   and press CONTINUE.
4. When the search is complete the screen displays that 8 entries are in the search result.
5. Now press the SORT RESULT softkey to sort the search result
6. To indicate that you want the result sorted by SUBJECT and arranged in ascending order, type SUBJECT and press CONTINUE twice to bypass the ORDER field. Now type AUTHOR and press the EXECUTE SORT softkey. Remember, unless you specify descending order the sort is done in ascending order.
7. To print the sorted data, press the LIST RESULT softkey.
8. Type T to specify output to the thermal printer and type L to specify linear format.
9. Press the EXECUTE LIST softkey.

After the listing is done, notice the order of the books. All the subjects are in ascending alphabetical order and where the subject is the same for two books, the author is arranged in ascending order.

10. Press the EXIT softkey to get out of the SORT command.
11. Press the EXIT softkey to get back to the SEARCH subsystem.
12. Now press the EXIT softkey to exit from the SEARCH subsystem and return you to the beginning of QUERY/45.

Changing the Data in the Data Base

The UPDATE softkey enables you to change the data in a data set by adding complete entries, deleting complete entries, or changing data item values. When you press the UPDATE softkey after you open the data base, this screen is displayed:



```
UPDATE SUBSYSTEM -- Command Selection

Press the appropriate special function key to select a command.

ADD ENTRIES ....... Add data entries to a data set.

DELETE ENTRIES .... Delete selected entries from a data set.

REPLACE ITEM ...... Replace the value of a single data item in all or selected
                    entries in the data set.

MODIFY ENTRIES .... Modify the value of all or selected data items in all or
                    selected entries in a data set.

AUTOCLEAR ......... When AUTOCLEAR is ENABLED, the ADD command clears the item
                    fields after adding an entry to a data set. When AUTOCLEAR
                    is DISABLED, the item fields are not cleared.

AUTOCLEAR is ENABLED. Press the softkey to disable it.
```

Adding Data Entries

Pressing the ADD ENTRIES softkey enables you to add entries to a manual master or a detail data set. You need to specify the data set to which you want to add entries. After a set is selected, the item names in that set are displayed, and now you may enter the values. After you have typed a value, press CONTINUE, TAB, STORE, EXERCUTE or the down arrow (↓) key. If you need to move the cursor to a previous field, press the SHIFT key and the TAB key together, or press the up arrow (↑) key. After all the values have been entered, press the ADD ENTRY softkey to store the entry. Once the entry is stored, the screen is cleared and another entry can be added.

Exercise #6: Adding a New Book to the BOOK Data Set

In this exercise, you can add a new book to the BOOK data set. This can be done after the UPDATE softkey is pressed.

1. Press the ADD ENTRIES softkey.
2. Enter 3 to indicate the BOOK data set is the one to which data is to be added.
3. Type the information about the book as described next, pressing the CONTINUE key after each data item value.

Title: THE MYTHICAL MAN-MONTH

Call number: 744714

Author: FREDERICK BROOKS

Subject: COMPUTER PROGRAMMING

Date Published: August 22,1975

Publisher: ADDISON-WESLEY

Price: 15.30

4. Press the ADD ENTRY softkey to store the entry into the data base.
5. If you wish, you may add another book. If not, press the OPTION SELECTION softkey to return you to the beginning of the UPDATE subsystem.

Deleting Data Entries

Pressing the DELETE ENTRIES softkey enables you to select and delete entries from a data set. Only complete data entries can be deleted; you cannot delete individual data item values. For example, all the data about a book has to be deleted, not just the publisher or the price.

To delete an entry, you must first specify the data set from which you want to delete entries. Next, specify which entries are to be deleted by entering a search expression. You can either delete all the entries found or delete only certain entries.

Exercise #7: Deleting a Book from the BOOK Data Set

In this exercise, you can delete the entry in the BOOK data set which has the subject MATHEMATICS and price 15.25.

1. Press the DELETE ENTRIES softkey.
2. Type 3 to indicate the BOOK data set.
3. Type SUBJECT = MATHEMATICS for the search expression and press CONTINUE.
4. Press the PAUSE DELETE softkey to view each of the three entries in order to find the exact entry.

If the entry displayed on the CRT is not the one you want to delete, press the NEXT ENTRY softkey to view the next entry in the search result.

5. Press the DELETE ENTRY softkey when the following entry is displayed on the CRT.

```
TITLE: HANDBOOK OF MATHEMATICAL FUNCTIONS

      CALL_NUMBER: 6460036
           AUTHOR: ABRAMOWITZ,MILTON
          SUBJECT: MATHEMATICS
   PUBLISHED_DATE: April 1, 1964
        PUBLISHER: U.S. GOVT. PRINT. OFFICE
            PRICE: 15.25
```

NOTE

Another way to find the above entry is to use:

SUBJECT = MATHEMATICS AND PRICE = 15.25

as the search expression. This returns one entry in the search result and then you can press the DELETE ALL softkey. Using this search expression saves you time since you don't have to examine each entry found for the particular entry you want.

6. Press the OPTION SELECTION softkey to return the program to the beginning of the UPDATE subsystem.

Backing up the Sample Data Base

Now that you have performed a few simple operations with the sample data base you should make a copy of the data in LIBR in case some failure occurs which could cause you to lose all or part of the data base. This is called **data backup** and it is an essential part of data base management.

To run the backup program:

1. Press the CLOSE DATA BASE softkey to get out of the present subsystem.
2. Press the STOP softkey to stop the QUERY/45 program.
3. Insert the IMAGE/45 Utility tape into the right tape drive and type – `LOAD "TBKUP:T15"`
4. When the run light goes out press the RUN key.

The program then asks you if you want to RECOVER or BACKUP your data base.

5. Type `1` for BACKUP.
6. Presss CONTINUE since the data base is a single volume data base.
7. Type `LIBR:`msus for the data base name to be backed up, then press CONTINUE.
8. Type `LIBRMGR` for the password.

9. Type BOOKS for the maintenance word[1].
10. Type LIBKUP:msus for the backup file name.
11. Enter N to indicate that the backup file is not to be protected.

The program is now backing up the LIBR data base; it displays each set number as it is being backed up. When the process is completed, you can type CAT and press EXECUTE and a new file call LIBKUP is listed as a backup file.

The root file, which contains all the structural information about a data base, and the data sets are the only files that are backed up into the new file LIBKUP. The program then asks you if you want the QUERY related files to be backed up.

12. Type Y to indicate YES and press CONTINUE.

The Information file is given the name of the data base followed by 00 (LIBR00). When TBKUP backs it up, the last two characters of the back up file name are 99 (LIBR99).

When backing up is completed, type CAT to display the files that are stored on your medium. Identify which files are the backed up copies and which are the original data base files.

Other Capabilities of QUERY/45

As explained earlier in this chapter, QUERY/45 enables you to BROWSE through the data, SEARCH for and list specific data, and UPDATE data that is found in a data base. There are several other features associated with QUERY/45, but they are more complex than BROWSE, SEARCH, and UPDATE. This section briefly discusses these additional features in general terms so that you are aware that they exist. Detailed explanations of each feature are found in the following chapters.

Using Forms with your Data Base

A **form** is an image on the CRT used while adding or modifying entries. You can define a form for any data set except an automatic master data set (since data in automatic masters is maintained automatically by the IMAGE/45 Data Base Management System). By defining your own form, you can display additional information about the data to be entered. For example, you can clarify the CALL_NUMBER item by also specifying that it is a number with a maximum of 12 digits. Forms are explained in Chapter 7.

Defining a Data Base

When creating a new data base, you must first design it (draw the data base diagram) and then interactively specify the definition to QUERY/45, interpreting the diagram to QUERY/45. You can use QUERY/45 to interactively define or redefine a data base. The Data Base Design Kit helps you to develop a data base diagram; Chapter 8 explains how to define and create a data base using QUERY/45.

[1] Maintenance words are described in Chapter 8.

Using the Formal Command Language

QUERY/45 enables you to use the formal command language to perform QUERY/45 operations rather than use the softkeys. Formal commands can be used when you are familiar with an operation, thus saving the time taken by using softkeys. After you open a data base, there is a softkey labeled FORMAL COMMAND; pressing it enables you to type and execute the formal commands. For example, you could type:

```
FIND SUBJECT = COMPUTER PROGRAMMING
```

and

```
LIST TITLE, AUTHOR
```

instead of using the softkeys. The command language is explained in greater detail in Chapter 10.

Linking Other Programs to QUERY/45

Another feature of QUERY/45 enables you to execute subprograms which are written to extend the feature of QUERY/45. For example, if you wrote a program which produces formatted reports, QUERY/45 can use this program to output data in a data base. You would open the data base, enter the SEARCH subsystem, and establish a search result. Then, by pressing the RUN USER PROGRAM softkey, QUERY/45 calls your reporting subprogram and the search result is used as the data for reporting.

There are two places within QUERY/45 that a subprogram can be linked. One is at the beginning of QUERY/45, after you have opened the data base, and the other is in the SEARCH subsystem. Extensions to QUERY/45 should be written by experienced programmers. Anyone can use the extension once it is written. The details about RUN USER PROGRAM are found in Chapter 9.

The Information File

The Information File is a file created by the DEFINE subsystem of QUERY/45. There is one Information File for each data base; it contains detailed information about the unique enhancements provided by QUERY/45 and how they are used in a particular data base. For example, the Information File specifies whether or not a form exists for a certain data set. It also keeps information about the Name, Date, and Code data item types. Explanation of this file is found in Appendix B.

Chapter 3
Data Base Concepts

If this is the first time you have used the IMAGE/45 Data Base Management System, you may encounter many unfamiliar terms. This chapter defines these terms and concepts as they apply to QUERY/45.

Data Base

A **data base** is a collection of related information that has been stored in a logical manner so that data can be easily accessed. For example, you can think of a company's library as being a data base with information about the books, who borrowed the books, and some information about each company's library branch.

Data Item

A **data item** is the smallest accessible data element in the data base. Each data item is given a value. In the library data base, examples of data items are the title, price, and author of a book.

Data Item	Data Item Value
CALL_NUMBER	5712902
TITLE	Engineering Electromagnetics
AUTHOR	William Hayt
SUBJECT	Electromagnetics Theory
PUBLISHED_DATE	December 26, 1956
PUBLISHER	McGraw-Hill
PRICE	8.50

Each data item has a name associated with it. For example, the data item containing the title of a book has the data item name TITLE. The format for data item names is found in Chapter 8.

Synonyms

When referring to data items,QUERY/45 enables you to use synonyms for the item name. A **synonym** is a name that can be used interchangeably with the item name as a memory aid or a typing aid; such as one or two letter abbreviations for a long item name. For example, the item name TITLE has the synonyms, HEADING and NAME_OF_BOOK. A synonym can be used instead of the item name TITLE while using QUERY/45. Up to five synonyms may be defined for each data item.

Types of Data Items

There are seven types of data items used with QUERY/45. The first four (Integer, Short, Long, and Character) are the basic types provided by IMAGE/45. The remaining three (Name, Date, and Code) are unique to QUERY/45. Each type is represented by a unique capital letter used when defining a data base. The four basic data item types are:

Character string (X) – a string of ASCII characters. A table of ASCII characters is found in Appendix D. A maximum length must be specified for each string; it can be an even number from 2 to 1022.

Integer (I) – a whole number in the range −32 767 through 32 767.

Short Numeric (S) – a short precision number with six significant digits and an exponent in the range −63 and 63.

Long Numeric (L) – a full precision real number with twelve significant digits and an exponent in the range −99 and 99.

These types are explained in greater detail in Chapter 8.

Name Type

The **NAME type (N)** data item is used to store personal names in a consistent format. However, names can be entered in one of two ways:

"last name, first name [middle name] [,suffix]"

or

"first name [middle name] last name [,suffix]"

No matter which way a name is entered, it is stored by QUERY/45 using the first format shown above. The part of the name enclosed in square brackets is optional. Some examples of how QUERY/45 converts names are:

Sam Smith	is converted to	Smith,Sam
Sam Smith, Jr.	is converted to	Smith,Sam,Jr.
Sam E. Smith, Jr.	is converted to	Smith,Sam E.,Jr.
Mr. & Mrs. Sam E. Smith	is converted to	Smith,Mr. & Mrs. Sam E.

If a person's last name has two or more parts, such as Von Grendy, the last name should be entered with the underscore character (_) between the parts of the name, such as Von_Grendy.

Code Type

The **CODE type (C)** data item enables you to establish a set of allowable values for a data item. Up to 35 string values, with a maximum of 15 characters each, can be specified. This enables you to ensure data validity by specifying the only allowable values for the item.

For example, the data item LOCATION in the sample data base is defined as a Code type. Its allowable values are: MANAGEMENT, MARKETING, PRODUCTION, R&D, FINANCE, PERSONNEL, and QA. These seven values are the only acceptable inputs that can be entered for the data item LOCATION.

Another advantage of using a Code item type is that it saves storage space on the disc. As code values are specified, they are given an integer value. It is the integer that is stored on the disc rather than the string value. QUERY/45 automatically performs the conversion. For the data item LOCATION, the related integers are:

MANAGEMENT: 1
MARKETING: 2
PRODUCTION: 3
R&D: 4
FINANCE: 5
PERSONNEL: 6
QA: 7

Date Type

The BOOK data set has a data item (PUBLISHED_DATE) of **type DATE (D)**. This enables you to enter dates for this item in any one of the following formats. The order in which the month and day are interpreted is specified in the CONFIGURE SYSTEM screen at the beginning of QUERY/45.

M – D – Y		D – M – Y	
January 9,1978	1/9/78	9 January 1978	9/1/78
January 9,78	1/9/1978	9 Jan 78	9/1/1978
Jan 9,1978	1-9-78	9 January 1978	9-1-78
Jan 9,78	1-9-1978	9 January 78	9-1-1978
1 9 78	1.9.78	9 1 78	9.1.78
1 9 1978	1.9.1978	9 1 1978	9.1.1978

The range of Date types is 1/1/0000 to 9/11/5475 and when the Date types are converted and stored as short-precision numbers, the range is −999998 to 999999. The advantage of using the Date type is that the dates can be conveniently searched and sorted in chronological order.

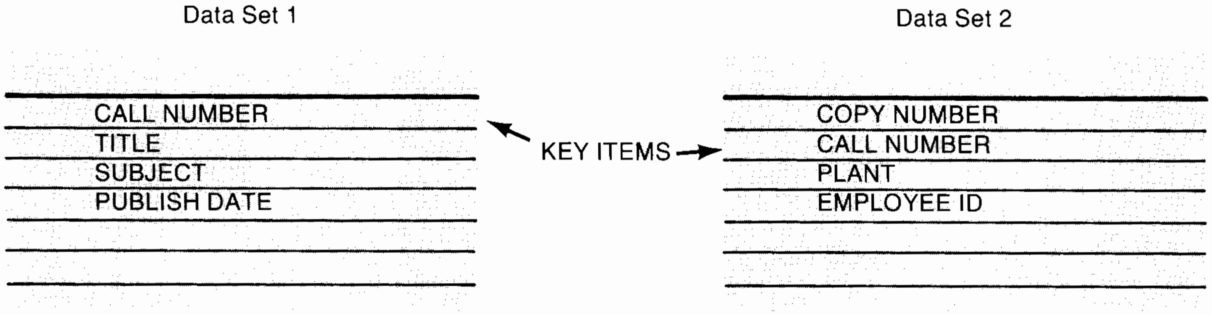
Compound Items

In the LIBRARY data set there is an item named PLANT_ADDRESS. This item is defined as: CHARACTER(3)[40],(req). The 3 in parentheses indicates that this item is a **compound item**. A compound item enables an item to have its value broken down into logical parts, called **subitems**, effectively making a one-dimensional array. Each subitem must be the same type and length.

PLANT_ADDRESS has three subitems with each one being 40 characters long. One is for the address and street, the second is for city and state, and the third is for the zip code. A compound item is used when you want to perform a search using just one part of an item. For example, to find the plant in Fort Collins, CO, you can search on ADDRESS(2) which contains the plant's city and state. The 2 in parentheses indicates the second subitem.

Key Item

A **key item** is used to **identify** data entries for quick **retrieval**. For example, a book can be identified by its call number, so this is defined to be a key item. Additionally, since one use of the LIBR data base is to retrieve all books on a particular subject, SUBJECT is defined to be a key item. Key items also link data sets together as illustrated next. A key item cannot be a compound item.



Ranges

Each numeric or date data item can be given a **range** which indicates the upper and lower bounds of allowable values. For example, the data item EMPLOYEE_NO should always be a positive number, so the range is defined as 0 to 32 767 instead of the default range of −32 767 to 32 767. When a data item value is entered, its value is checked to see if it is within the range. If the value isn't within the range, an error message is displayed and a new value must be entered.

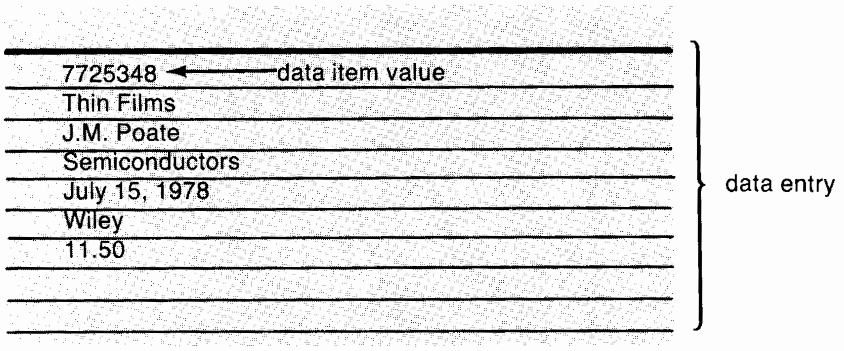
Null Values

When a value is not entered for a data item, a **null value** is stored. The following table shows the null values for each of the data item types.

Data Item Type	Null Value
integer	−32 768
short	−9.99999E63
long (real)	−9.99999999999E99
date	1/0/0000
code	0
character	" "
name	" "

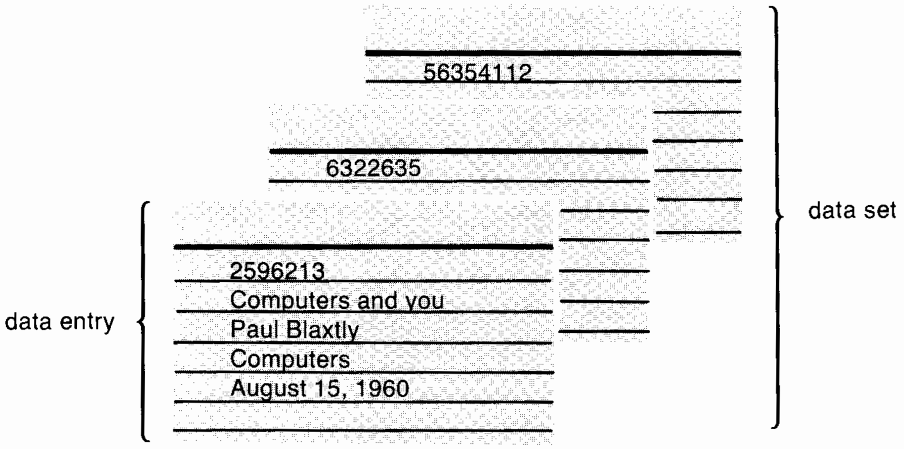
Data Entry

A collection of related data item values is called a **data entry**. An entry can be thought of as a card found in the card catalog at the library. For example, here is a data entry for the BOOK data set:



Data Set

A collection of similar data entries is called a **data set**. The next example contains three data entries which together make up a particular data set, the BOOK data set.

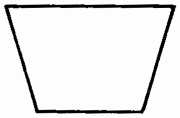


Data Set Types

There are two types of data sets used by QUERY/45: **detail sets** and **master sets. Detail sets** contain the bulk of the information. A **master set** is generally used as an index for fast access to the information in detail sets. There are two types of master data sets: **automatic** and **manual**.

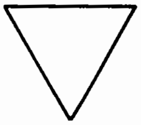
Each master data set contains one key item. Each detail data set can contain up to 16 key items and other non-key items. Each of the key items in a detail set is linked to the key item in one of the master data sets. Key items can be used to link together the information in two or more data sets. A detail data set need not contain any key items if you do not wish to index or link it.

Detail Data Sets



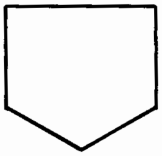
A **detail data set** generally contains data items which describe one "thing" about which information is kept. For example, the BOOK data set contains data about the books, but nothing about the library branches or borrowers.

Automatic Master Data Sets



An **automatic master data set** contains one data item, the key item. (Remember that a key item is an item common to two or more sets to "link" master and detail sets together.) When a new value is added for the same key item in a detail data set, the value is automatically added to the automatic master data set. Deletions from the automatic master data set are made when the last data entry with the particular key item value is deleted from the detail data set. Automatic masters are used when key item values are too numerous or unpredictable, making manual entry unreasonable.

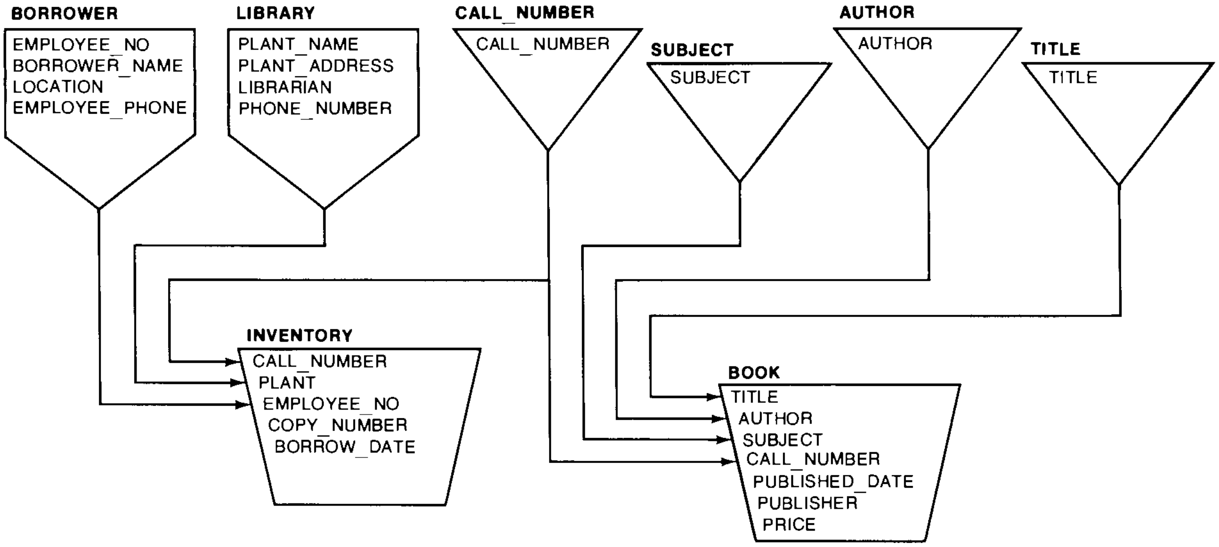
Manual Master Data Sets



A **manual master data set** can contain one or more data items in addition to the one key item. Before a new key item value can be added to a detail data set that is linked to a manual master data set, the value must first be added to the manual master data set. This enables you to control what data is entered into a manual master data set since you have to enter the values. Typing errors in a manual master data entry won't be automatically entered as they would if you were using a automatic master data set.

The LIBR Diagram

The following diagram illustrates the LIBR data base using the "boxes" defined for each data set. It is simular to the diagram produced by the Data Base Design Kit and shows the different sets and their relationships.


BORROWER
EMPLOYEE_NO
BORROWER_NAME
LOCATION
EMPLOYEE_PHONE
LIBRARY
PLANT_NAME
PLANT_ADDRESS
LIBRARIAN
PHONE_NUMBER
CALL_NUMBER
CALL_NUMBER
SUBJECT
SUBJECT
AUTHOR
AUTHOR
TITLE
TITLE
INVENTORY
CALL_NUMBER
PLANT
EMPLOYEE_NO
COPY_NUMBER
BORROW_DATE
BOOK
TITLE
AUTHOR
SUBJECT
CALL_NUMBER
PUBLISHED_DATE
PUBLISHER
PRICE

Chapter 4
Basic Operations

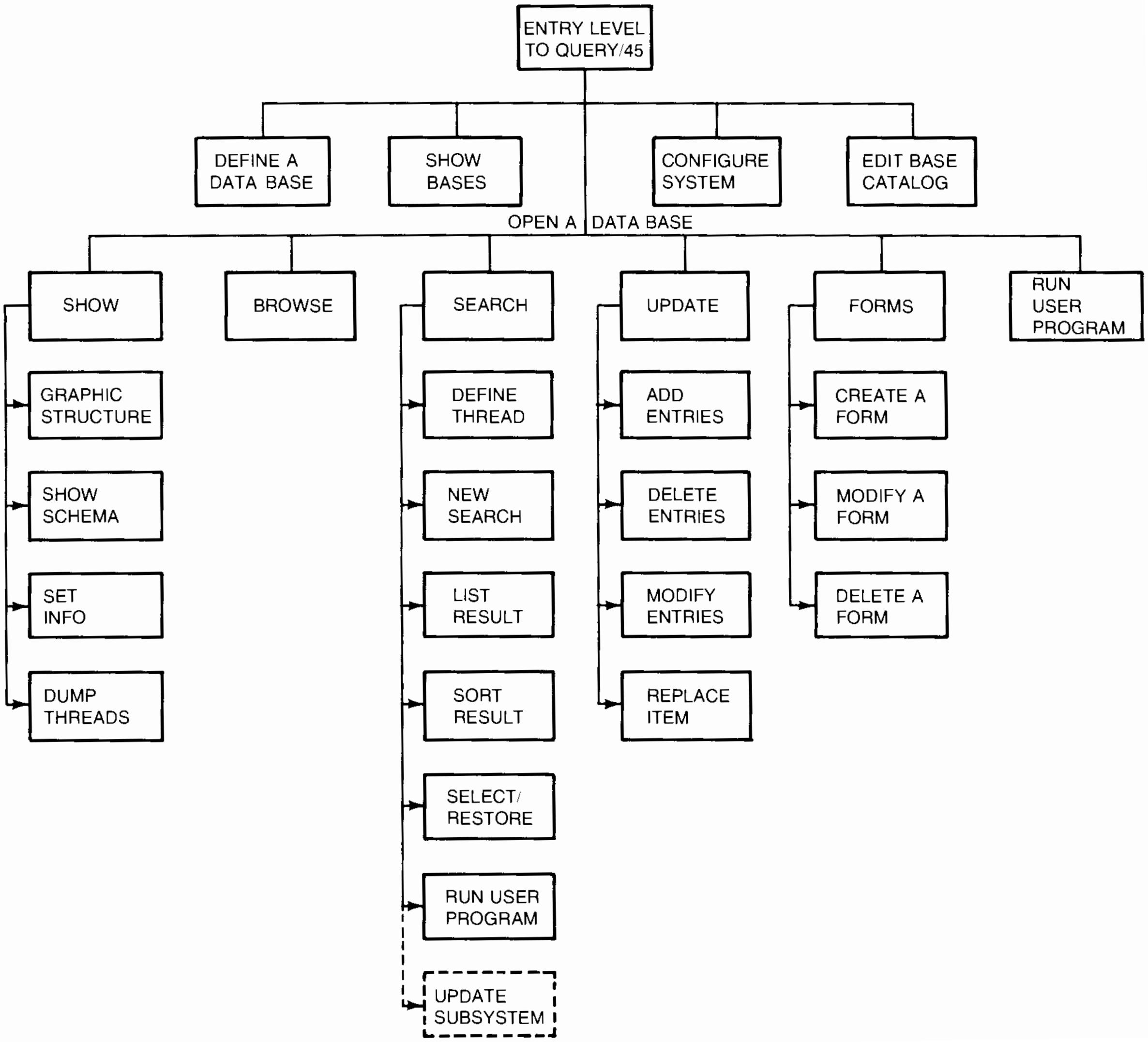
Introduction

Chapters 4 through 7 are written to give a more detailed explanation of the QUERY/45 features and capabilities. An overview of these features was presented by the exercises in Chapter 2. The following chapters contain helpful hints and other information important when using QUERY/45. This chapter contains:

- an overview of QUERY/45.
- a review of general features of QUERY/45.
- an explanation of basic data base operations: opening and closing the data base and looking at the structure and contents of the data base. (SHOW BASE INFO and BROWSE)

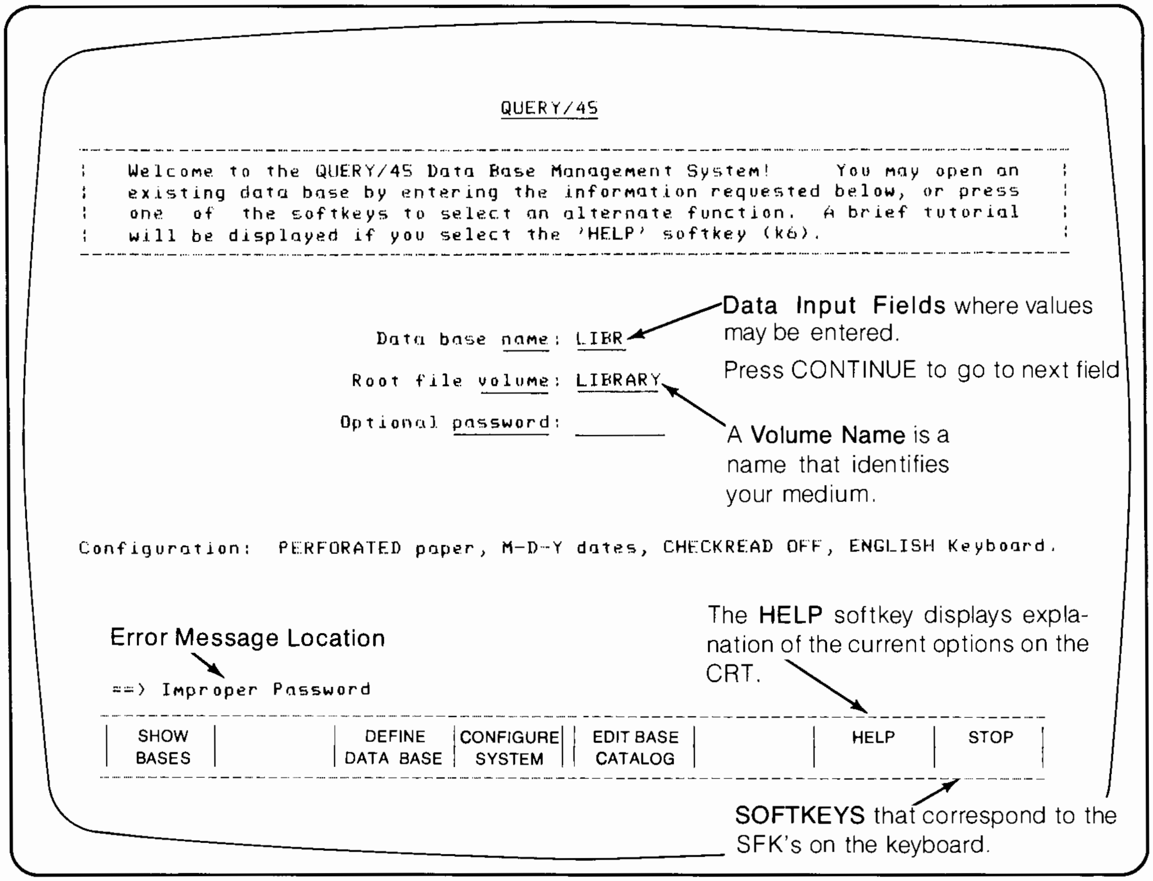
Overview of QUERY/45

QUERY/45 is organized as a **tree structure** enabling you to branch to subsequent levels, then return to access other features. The top of the tree enables you to open a data base, define a data base, or configure your system. If you choose to open a data base, you then can branch down to perform different operations on the data base. When you want to exit from the program, you have to exit back up through the different levels of the tree structure to close the data base and stop the program. Following is a general diagram of QUERY/45 showing the different levels of the tree structure:


ENTRY LEVEL TO QUERY/45
DEFINE A DATA BASE
SHOW BASES
CONFIGURE SYSTEM
EDIT BASE CATALOG
OPEN A DATA BASE
SHOW
BROWSE
SEARCH
UPDATE
FORMS
RUN USER PROGRAM
GRAPHIC STRUCTURE
SHOW SCHEMA
SET INFO
DUMP THREADS
DEFINE THREAD
NEW SEARCH
LIST RESULT
SORT RESULT
SELECT/ RESTORE
RUN USER PROGRAM
UPDATE SUBSYSTEM
ADD ENTRIES
DELETE ENTRIES
MODIFY ENTRIES
REPLACE ITEM
CREATE A FORM
MODIFY A FORM
DELETE A FORM

General QUERY/45 Features

When running QUERY/45, various features are included to make using QUERY/45 as simple as possible. The features (softkeys, fields, error message location, and volume names) are described in Chapters 1 and 2, and are summarized in the following diagram.



```
                              QUERY/45
-------------------------------------------------------------------------
:  Welcome to the QUERY/45 Data Base Management System!    You may open an  :
:  existing data base by entering the information requested below, or press :
:  one  of  the softkeys to select an alternate function.  A brief tutorial :
:  will be displayed if you select the 'HELP' softkey (k6).                 :
-------------------------------------------------------------------------

                 Data base name:  LIBR
               Root file volume:  LIBRARY
              Optional password:  ________

Configuration:  PERFORATED paper, M-D-Y dates, CHECKREAD OFF, ENGLISH Keyboard.

==> Improper Password
```

Data Input Fields where values may be entered.

Press CONTINUE to go to next field

A **Volume Name** is a name that identifies your medium.

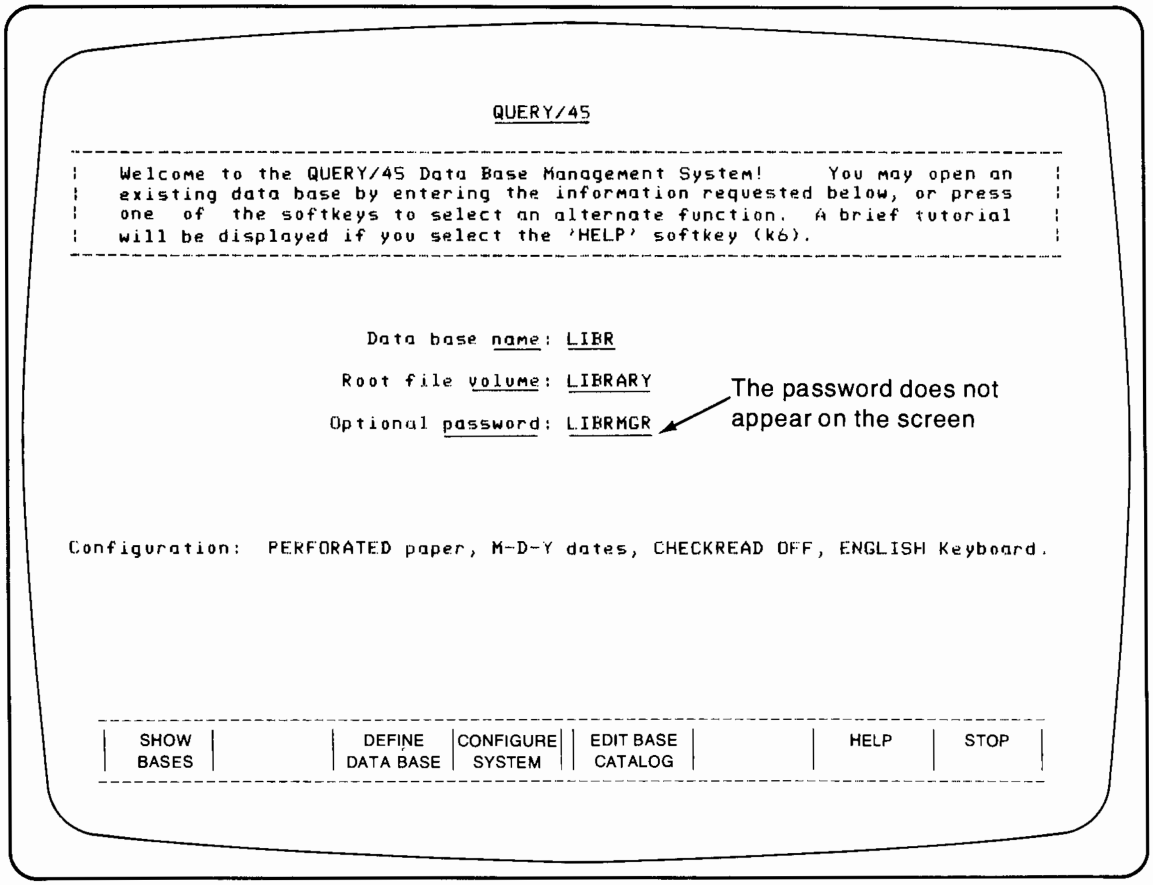
Error Message Location

The **HELP** softkey displays explanation of the current options on the CRT.

SOFTKEYS that correspond to the SFK's on the keyboard.

Accessing a Data Base

In order to access a data base with QUERY/45, the first step you take must be to open the data base. This operation involves specifying the name of the data base, the volume name of the device which contains the root file of the data base, and a password. When you start running the QUERY/45 program, you can open the data base by filling in the screen as shown below:



```
                                   QUERY/45
--------------------------------------------------------------------------------
!   Welcome to the QUERY/45 Data Base Management System!     You may open an    !
!   existing data base by entering the information requested below, or press    !
!   one  of  the softkeys to select an alternate function.  A brief tutorial    !
!   will be displayed if you select the 'HELP' softkey (k6).                     !
--------------------------------------------------------------------------------

                           Data base name: LIBR

                         Root file volume: LIBRARY

                        Optional password: LIBRMGR


Configuration:  PERFORATED paper, M-D-Y dates, CHECKREAD OFF, ENGLISH Keyboard.
```

The password does not appear on the screen

The reverse of opening a data base is closing a data base. This is also a necessary operation; QUERY/45 performs this operation automatically when you press the CLOSE DATA BASE softkey.

Looking at the Data Base

After you have opened a data base, the SHOW BASE INFO softkey is used to obtain information about the data base in four different ways.

- The GRAPHIC STRUCTURE softkey prints a diagram of the data base on the printer.
- The SCHEMA softkey prints the schema, or formal definition of the data base.
- The SET INFO softkey prints information about the individual sets.
- The THREADS softkey prints the threads that have been defined while using QUERY/45.

Graphic Structure

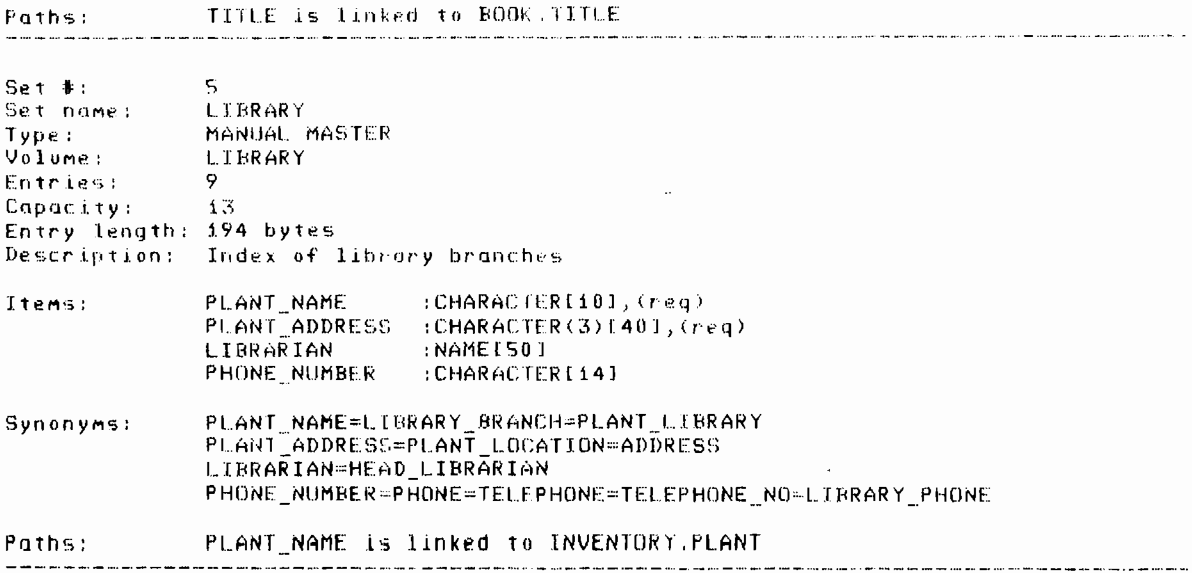
The diagram produced by the GRAPHIC STRUCTURE softkey is useful during other data base operations to get information like data item names, quickly. If the password specified does not have access to some of the data sets, these sets are not printed. Within this diagram, the data set names, data items and their types, the data set capacity, and the data paths are shown in the diagram.

Set Information

If you just want to know about the data sets in a data base, use the SET INFO softkey. For each data set the following information is printed: the data set name, the type of set, the volume name, how many entries are in the set and the capacity of the set are printed on the CRT or a printer. You specify where you want the information to be printed.

The Data Base Schema

After pressing the SHOW BASE INFO softkey, the SCHEMA softkey is displayed. Pressing SCHEMA prints all the data sets, items, codes, synonyms and ranges of the data base in a format similar to the schema, or formal definition, used by IMAGE/45. The information can be printed on an external printer, the internal printer or the CRT. Here is part of the information about the LIBR data base obtained by pressing SCHEMA:



```
Paths:          TITLE is linked to BOOK.TITLE
--------------------------------------------------------------------------------

Set #:          5
Set name:       LIBRARY
Type:           MANUAL MASTER
Volume:         LIBRARY
Entries:        9
Capacity:       13
Entry length:   194 bytes
Description:    Index of library branches

Items:          PLANT_NAME        :CHARACTER[10],(req)
                PLANT_ADDRESS     :CHARACTER(3)[40],(req)
                LIBRARIAN         :NAME[50]
                PHONE_NUMBER      :CHARACTER[14]

Synonyms:       PLANT_NAME=LIBRARY_BRANCH=PLANT_LIBRARY
                PLANT_ADDRESS=PLANT_LOCATION=ADDRESS
                LIBRARIAN=HEAD_LIBRARIAN
                PHONE_NUMBER=PHONE=TELEPHONE=TELEPHONE_NO=LIBRARY_PHONE

Paths:          PLANT_NAME is linked to INVENTORY.PLANT
--------------------------------------------------------------------------------
```

```
Set #:          6
Set name:       BORROWER
Type:           MANUAL MASTER
Volume:         LIBRARY
Entries:        61
Capacity:       79
Entry length:   68 bytes
Description:    Index of borrowers of books

Items:          EMPLOYEE_NO     :INTEGER[0:32767],(req)
                BORROWER_NAME   :NAME[50],(req)
                LOCATION        :CODE
                EMPLOYEE_PHONE  :CHARACTER[14]

Synonyms:       EMPLOYEE_NO=EMPLOYEE
                BORROWER_NAME=BORROWER
                LOCATION=AREA_LOCATED=DEPARTMENT
                EMPLOYEE_PHONE=PHONE_NO

Codes:          LOCATION=(MANAGEMENT,MARKETING,PRODUCTION,R&D,FINANCE,PERSONNEL,QA)

Paths:          EMPLOYEE_NO is linked to INVENTORY.EMPLOYEE_NO
------------------------------------------------------------------------------------

Set #:          7
Set name:       BOOK
Type:           DETAIL
Volume:         LIBRARY
Entries:        69
Capacity:       89
Entry length:   196 bytes
Description:    Book description

Items:          TITLE           :CHARACTER[60],(req)
                CALL_NUMBER     :LONG[0:9.99999999999E+99],(req)
                AUTHOR          :NAME[50],(req)
                SUBJECT         :CHARACTER[40]
                PUBLISHED_DATE  :DATE[January 1, 1900:December 31, 2000]
                PUBLISHER       :CHARACTER[30]
                PRICE           :SHORT[.01:500]

Synonyms:       TITLE=HEADING=NAME_OF_BOOK
                AUTHOR=NOVELIST=WRITER=POET=AUTHOR_NAME
                SUBJECT=TOPIC
                PUBLISHED_DATE=PUBLISH_DATE=DATE_PUBLISHED
                PUBLISHER=PUBLISHER_NAME
                PRICE=COST=BOOK_PRICE=PRICE_OF_BOOK=COST_OF_BOOK=BOOK_COST

Paths:          TITLE is linked to TITLE.TITLE
                CALL_NUMBER is linked to CALL_NUMBER.CALL_NUMBER
                AUTHOR is linked to AUTHOR.AUTHOR
                SUBJECT is linked to SUBJECT.SUBJECT
------------------------------------------------------------------------------------

Set #:          8
Set name:       INVENTORY
Type:           DETAIL
Volume:         LIBRARY
Entries:        148
Capacity:       193
Entry length:   34 bytes
Description:    Books in the library

Items:          CALL_NUMBER     :LONG[0:9.99999999999E+99],(req)
                COPY_NUMBER     :CHARACTER[10],(req)
                PLANT           :CHARACTER[10],(req)
                EMPLOYEE_NO     :INTEGER[0:32767],(req)
                BORROW_DATE     :DATE[January 1, 1937:December 31, 2000]
```

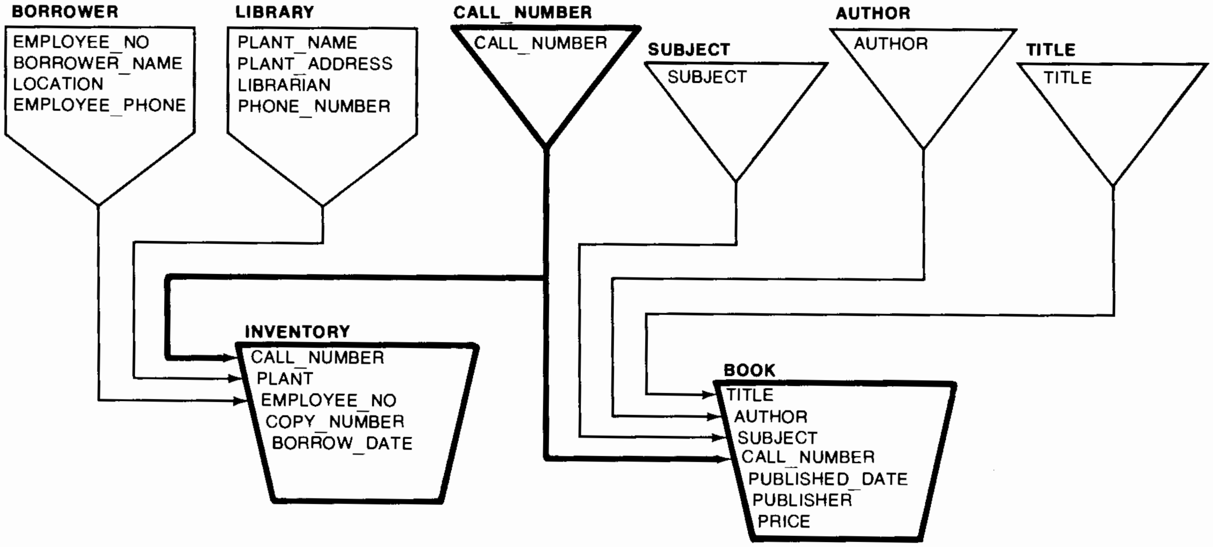
```
Synonyms:       COPY_NUMBER=COPY=COPY_NO=NUMBER_OF_COPY
                EMPLOYEE_NO=EMPLOYEE
                BORROW_DATE=DATE_BORROWED

Paths:          CALL_NUMBER is linked to CALL_NUMBER.CALL_NUMBER
                PLANT is linked to LIBRARY.PLANT_NAME
                EMPLOYEE_NO is linked to BORROWER.EMPLOYEE_NO
------------------------------------------------------------------------------------
```

Listing the Defined Threads

A thread is required when performing a SEARCH that uses information from more than one data set. The THREADS softkey lists all the existing threads for the specified data base. When you press the THREADS softkey using the LIBR data base, these threads are printed:

```
                  Pre-defined threads for LIBR
                  ____________________________

      COPY=BOOK to CALL_NUMBER to INVENTORY

      LOAN=BORROWER to INVENTORY to CALL_NUMBER to BOOK
```

This example shows one thread which was given the name COPY. Each thread is given a name to identify it. Here is a diagram showing how the thread links the sets together.



Thread definition is covered in Chapter 5.

Browsing Through the Data

If you want to know what entries are found in a particular data set, press the BROWSE softkey at the beginning of QUERY/45. After you specify the set to be examined, the entries are displayed one at a time on the CRT. The entries are read serially from the data set and are displayed in that order. The BROWSE softkey can help you become familiar with the data in an unfamiliar data base.

Chapter 5
Retrieving Data Meeting Certain Criteria

Introduction

The SEARCH subsystem enables you to retrieve selected data entries from the data base, then display, list, sort, or update them. The desired entries are retrieved by means of "translating" a natural language question into something that QUERY/45 can understand. This is done by using a **search expression** which consists of an item name, a relational operator, and a value or an item name. For example, if you want to know what author wrote the book Programming Proverbs, the search expression would look like:

```
TITLE = PROGRAMMING PROVERBS
```

It is helpful to be familiar with the data base when using the SEARCH subsystem. If you do not know what type of data is stored in the data base, use the BROWSE feature, which is explained in Chapter 4.

The **search result** consists of pointers to the entries which meet the search criteria. Up to 7560 pointers can be stored in memory at one time. If more than one set is involved in the search, the maximum search result size is reduced to 7560 divided by the number of sets in the search. If the search result is larger than the maximum size, QUERY/45 creates a file to hold the remainder. This file is created on the root file volume. The maximum number of entries in a search result is 32 767.

Accessing the SEARCH Subsystem

The following softkeys are displayed at the beginning of the SEARCH subsystem. You can then press the SEARCH softkey to specify a search expression.

SHOW BASE INFO	BROWSE	SEARCH	UPDATE	FORMAL COMMAND	MORE KEYS	HELP	CLOSE DATA BASE

After a search has been performed, the following softkeys are displayed. Another search can be performed by pressing the NEW SEARCH softkey.

LIST RESULT	SORT RESULT	NEW SEARCH	SELECT/ RESTORE	UPDATE RESULT	RUN USER PROGRAM	HELP	EXIT

If you are familiar with QUERY/45, you can use the formal command mode to perform a search. This enables you to save time by typing the command rather than using the softkeys. If you're not sure about the information needed to make up a command, you can press the PROMPT softkey which helps you compose the expression. A listing of all the formal commands is found in Chapter 10.

Performing a Search

A search is implemented using a **search expression** to select data entries. The simplest form of a search expression is:

item relational operator value or item you are searching for

A few examples of simple search expressions are:

AUTHOR WITH LASTNAME ANDERSON,

PRICE > 15.35

LOCATION = MARKETING

The following relational operators can be used in search expressions:

Relational Operator	**Explanation**
= , IS	retrieves entries whose item value exactly matches the search value.
<>,#,IS NOT	retrieves entries where the item value and the search value do not match.
< , BEFORE	retrieves entries whose item value is less than the search value.
> , AFTER	retrieves entries whose item value is greater than the search value.

<= retrieves entries whose item value is less than or equal to the search value.

>= retrieves entries whose item value is greater than or equal to the search value.

FROM ... TO retrieves entries with values in the specified range, including the boundary conditions named.

BETWEEN ... AND retrieves entries with values in the specified range, excluding the boundary conditions named.

WITH retrieves entries where the item value (character or name type) contains the string specified by the search value anywhere within it.

STARTING WITH retrieves entries where the item value (character or name type) starts with the string specified by the search value.

WITH FIRSTNAME retrieves entries where the item value (name type only) contains the value ",''&search value.

WITH LASTNAME retrieves entries where the item value (name type only) starts with search value&",''.

Considerations

Any character-string data that is entered with lowercase letters is stored with lowercase letters. Similarly, uppercase are stored as uppercase. This is important when you are specifying the string for search expressions; for example, a capital letter (A) does not equal a lowercase letter (a).

Compound Searches

Sometimes it is convenient to search for entries that meet several conditions rather than just one. QUERY/45 enables **compound search expressions** to be used when the search involves more than one search condition. Compound search expressions are formed by connecting two simple expressions together using the conjunctions "AND" and "OR".

The "AND" operator specifies that both criteria must be met for the search expression to be true. The "OR" specifies that if either criteria is met, the search expression is true. For example, if you type the search expression:

SUBJECT = ROBOTS AND PRICE > 20.00

and an entry has the SUBJECT = ROBOTS but the PRICE = 15.95, then the search expression is evaluated to be false. Only one of the criteria is met.

If you type this search expression:

SUBJECT = ROBOTS OR PRICE > 20.00

and find the same entry described above, the search expression is evaluated true. The "OR" specifies that only one criterion must be met (SUBJECT = ROBOTS).

Search expressions are evaluated left to right with "AND" taking priority over "OR", but parentheses can be used to establish a different order. For example, using the LIBR data base, you can type in the following compound search expression:

PUBLISHED_DATE > 1-1-76 AND PRICE = 20.00 OR SUBJECT = ROBOTS

and press CONTINUE. When the search is complete, the search result contains twelve entries that have qualified. If you press the NEW SEARCH softkey, indicate the BOOK data set again and type:

PUBLISHED_DATE > 1-1-76 AND (PRICE = 20.00 OR SUBJECT = ROBOTS)

the search result contains three entries. This shows that the SEARCH program evaluates two nearly identical searches with greatly varying results. In the first example, parentheses are implicity placed like:

(PUBLISHED_DATE > 1-1-76 AND PRICE = 20.00) OR SUBJECT = ROBOTS

Some other examples of compound searches are:

PUBLISHER = MCGRAW-HILL OR PUBLISHER = ADDISON-WESLEY

PLANT_ADDRESS(3) IS 97330 OR LIBRARIAN WITH LASTNAME ROSS,

PRICE FROM 10.00 TO 20.00 AND TITLE WITH COMPUTER

Multiple Values

If you are using a search expression involving the "=" or "WITH" operators, multiple values can be specified. For example, this search expression:

SUBJECT = MATHEMATICS ; INDUSTRIAL MANAGEMENT

is the same as:

(SUBJECT = MATHEMATICS OR SUBJECT = INDUSTRIAL MANAGEMENT)

If you have a multiple value search along with another search criteria such as:

PRICE=15.00;20.00 AND SUBJECT=COLOR

then the multiple-value search is evaluated first. Multiple values have the highest priority in a compound search expression if parentheses are not used. For example, the previous example is the same as:

(PRICE=15.00 OR PRICE=20.00) AND SUBJECT=COLOR

Using Quotes in a Search Expression

You can use quotes when specifying names or character strings but they are not required. For example, the following search expressions perform the same search:

PLANT = DCD

PLANT = "DCD"

There are, however, a few cases where quotes are required.

- When a search string contains reserved words or characters such as AND, OR, TO, FOR, $NULL, ;,or). $NULL is used to check if the search item has no value.
- When using the equals operator (=) to search for a character string whose value contains leading or trailing blanks. For example, "MARKETING" does not equal " MARKETING ".
- If the string value contains any quote marks, they must be doubled and the entire string must be enclosed in quotes. For example, the string AB"C would be typed "AB""C".
- If the search string is the same as an item name or a synonym name, quotes are needed. For example, TITLE = "SUBJECT" finds the books where the title is the string "SUBJECT". TITLE = SUBJECT finds any books where TITLE and SUBJECT are identical.

Item Names

Item names can be used as the search value in a search expression. For example, if you want to check if the librarian has borrowed a book, you would type: LIBRARIAN = BORROWER_NAME.

Null Values

A data item contains a **null value** if nothing is entered for the item using the UPDATE subsystem. The word $NULL is a reserved keyword which indicates null data. A search expression can be entered to find items with a data value. $NULL can be used for any data item; codes, names and strings can also use " " for the null value.

For example, a search may be executed to find the BORROW_DATEs that have no value by typing:

```
BORROW_DATE = $NULL
```

Arithmetic Expressions

Arithmetic expressions can also be used in a search expression, for either side of the relational operator, if the data item on which you are searching is integer, short, long or date type. The following examples are valid search expressions:

```
PRICE + 5.00 > 20.00
EMPLOYEE_NO / 2 FROM 1200 TO 3090
CALL_NUMBER BETWEEN 100000 AND CALL_NUMBER * 3
```

Recalling Search Expressions

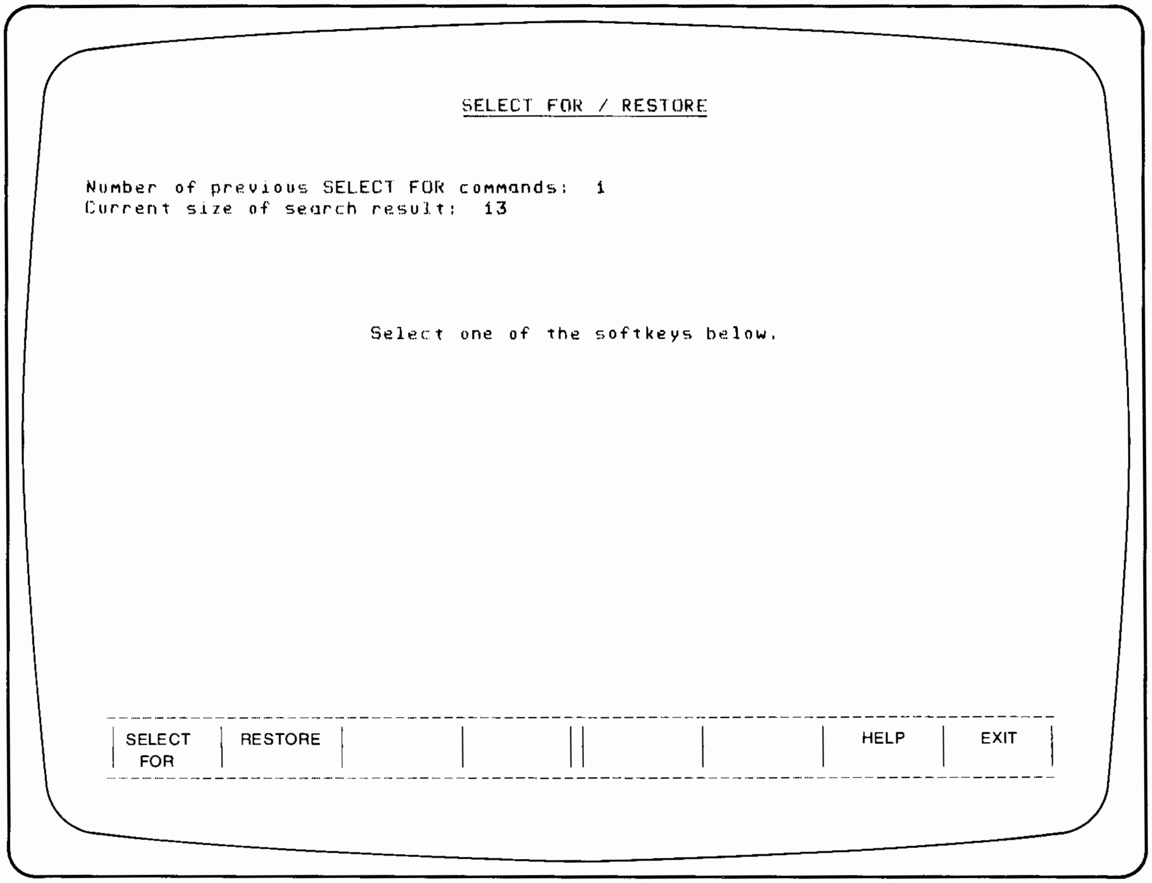
There is a softkey, RECALL, which is displayed as you are specifing the search expression. This softkey is used to re-display search expressions that have been previously entered allowing you to view, modify and reuse previous search expressions. Search expressions are stored into a 321-character buffer on a last in, first out basis. Each time the RECALL softkey is pressed, a previous search expression is displayed in the search expression area on the CRT.

When the recall buffer becomes full, each new search expression causes one or more of the oldest search expression, depending on size, to be lost.

Narrowing Down the Search Result

After you have performed a search and established a search result, you can press the SELECT/RESTORE softkey to narrow down your current search result by selecting the entries that meet a new search criterion. The new search result is a subset of and replaces the previous search result. If no entries qualify, the previous search result is kept.

After you have narrowed down the original search result once, you can narrow down the new search result further or restore the original search result. After pressing the SELECT/RESTORE softkey the second time, this screen is displayed.



```
                         SELECT FOR / RESTORE


Number of previous SELECT FOR commands:  1
Current size of search result:  13


                    Select one of the softkeys below.
```

The SELECT FOR softkey is pressed if you want to narrow down the search result further and the RESTORE softkey performs the converse operation of the SELECT FOR softkey. Every time you press the RESTORE softkey, the previous search result is restored as the current search result.

Hints for Reducing Search Times

The search expression specified for a search affects how long the search takes. Most searches are performed by starting at the first entry in the set and reading each entry successively, determining whether the entry meets the specified criteria (serial searching). Here are a few hints to use when searching so that the searches are performed faster. They apply to single-set searches only.

- Searches are faster if a key item is used as the search item with "=" or "IS" as the operator. This enables QUERY/45 to access the desired entry directly by way of calculated access. (Calculated access is explained in the IMAGE/45 Programming manual.) For example, if you have ten thousand entries in a detail set and you are searching for entries with SUBJECT BEGINNING WITH DATA, QUERY/45 has to read each entry (ten thousand) to check if it meets the search criterion. But, by using the search expression SUBJECT = DATA PROCESSING, QUERY/45 accesses the occurrence of the subject in the master data set, then accesses all the detail entries directly because they are automatically chained together. This may mean reading only twenty entries. Explanations of chained access are found in the IMAGE/45 Programming manual.

NOTE

When using LIBR to do the example searches, the time difference is not noticeable, but when you are searching a large data base and using large compound search expressions, the time difference may be significant.

- Here is another form of a compound search expression that enhances the speed of the search.

  PLANT = DTD AND (BORROW_DATE IS Feb 1,1979 OR COPY_NUMBER >2)

- If you are using parentheses in the search expression, place the key item and the equals operator outside the parentheses (outer level).

Considerations

Search expressions can be constructed in many, many forms. Remember to compare data items of the same type to avoid an error message to be printed.

Multiple-Set Searches

When the criteria for retrieving entries is based on the information found in two or more sets, a **thread** must be specified in order to link these sets together through the key items. Specifying a thread enables QUERY/45 to perform multiple-set searches.

Example

Suppose you know the call number of a book (7725348) and you want to know the names of the employees who have checked out that book. Borrower names are found in the BORROWER data set, but the call number is not. How do you find the necessary information?

Two separate searches need to be performed to find the employee number, then the borrower's name.

1. Search the INVENTORY data set for CALL_NUMBER = 7725348 and from the entry found, you know the EMPLOYEE_NO = 2843.
2. Search the BORROWER data set for EMPLOYEE_NO = 2843 to find the borrower's name.

The common data item is EMPLOYEE_NO; the information needed about EMPLOYEE_NO was found in the first search and used in the second search.

What if the search is more complex? Given a specific subject, (COMPUTER PROGRAMMING) how do you find out who has checked out books with this subject. The search expressions needed are:

1. Search BOOK data set to find SUBJECT = COMPUTER PROGRAMMING to get a list of the CALL_NUMBERs.
2. Search INVENTORY for CALL_NUMBER = ...;...;..., listing all the EMPLOYEE_NOs.
3. Search BORROWER for EMPLOYEE_NO = ...;...;... to find all the BORROWER_NAMEs.

If several CALL_NUMBERs or EMPLOYEE_NOs need to be listed for the searches, then many searches need to be made to find all the borrower names.

QUERY/45 provides a feature to perform these multiple-set searches automatically in one pass using **threads**.

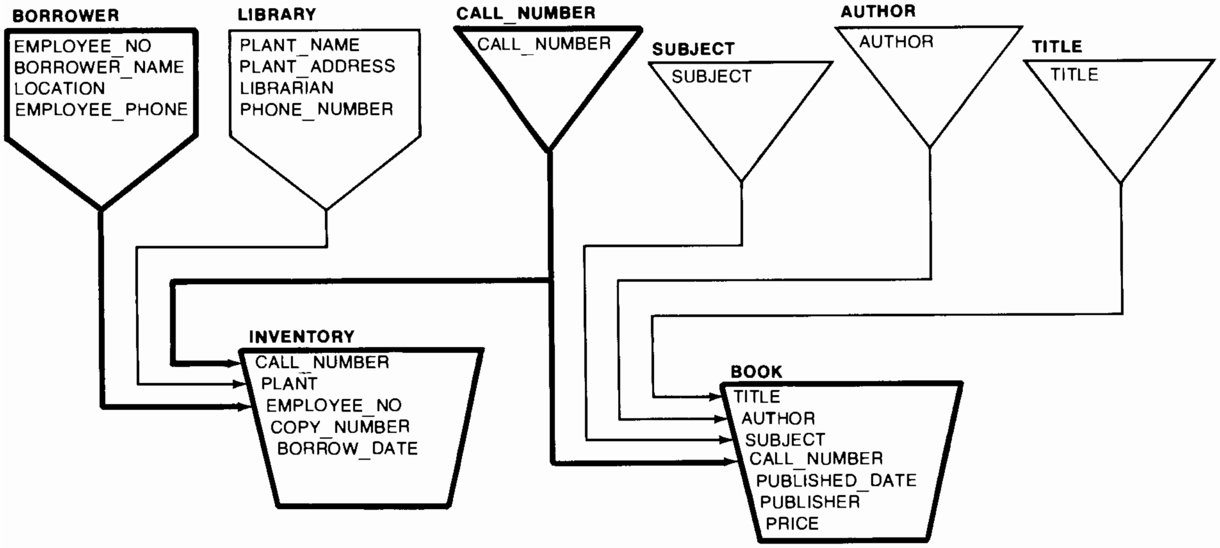
Threads

A **thread** is a named combination of data sets, specifying a unique path from one data set to the next based on search items. A master set can only be linked to a detail set and a detail set can only be linked to a master set. QUERY/45 enables you to name, define and store up to four threads with a maximum number of ten data sets linked in each thread.

In the previous example, a thread would need to be defined linking the BOOK data set to the BORROWER data set using common data items, CALL_NUMBER and EMPLOYEE_NO. Here is what the thread would look like:

LOAN = BOOK to CALL_NUMBER to INVENTORY to BORROWER

Here is a diagram of the thread defined:



A multiple-set search could then be performed to solve the above example by using this search command:

FIND LOAN FOR SUBJECT = COMPUTER PROGRAMMING

Because you used a thread to perform the multiple-set search, you can list the data items TITLE and BORROWER_NAME using the list command.

Using Threads to Shorten the Searching Time

The normal method for performing searches is to do serial reads in a data set to find the entries that meet the specified criteria. The only exception to this is if an equality operator is used with a key item in the search expression like: SUBJECT = PHYSICS.

Suppose you want to find all the entries in the BOOK data set that have "SUBJECT WITH COMPUTER"? This is not optimized because "WITH" is a relational operator. If there are 10,000 entries in the BOOK data set, a serial search can be time consuming.

However, the following thread can be defined to optimize the search time:

QUICK = SUBJECT to BOOK

Serial reads are now performed in the SUBJECT data set which has 100 entries. When a desired entry is found, the BOOK entry is accessed by way of a path. Since SUBJECT is much smaller than BOOK, the search is performed quicker.

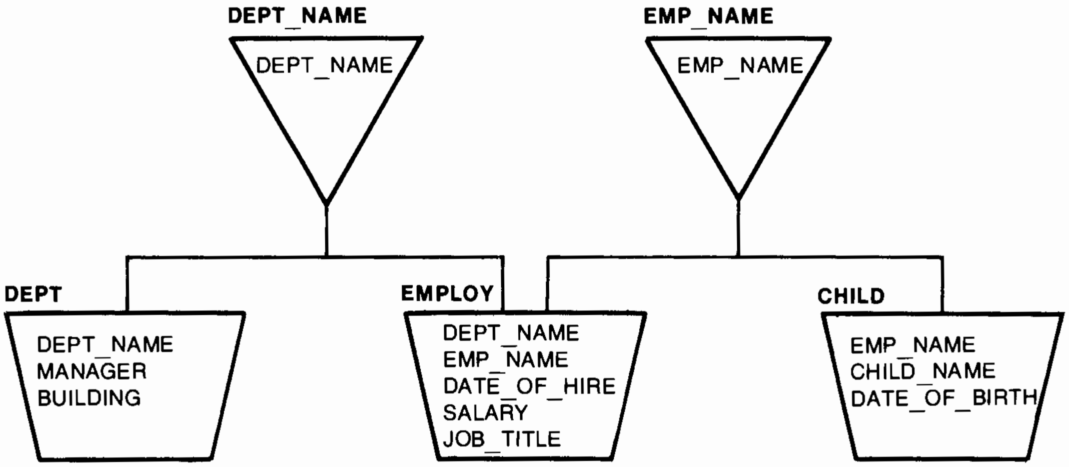
Optimizing Threads

When you are defining threads, the search is performed faster if:

- the thread starts from the smallest set and works toward the largest set.
- the sets involved in the search expression are defined towards the front of the thread. This means that QUERY/45 does not have to read as many item values before making a true or false decision about the search criterion.
- the thread doesn't end with an automatic master data set since no new data is stored in an automatic master.

Searching a Hierarchy

When defining a thread, it should contain only those sets necessary to find the desired information. For example, suppose you have a hierarchical data base that contains information about departments, employees and the employee's children; its diagram looks like:



If you wanted to find all the children of those employees who work in BUILDING #1, this thread is defined:

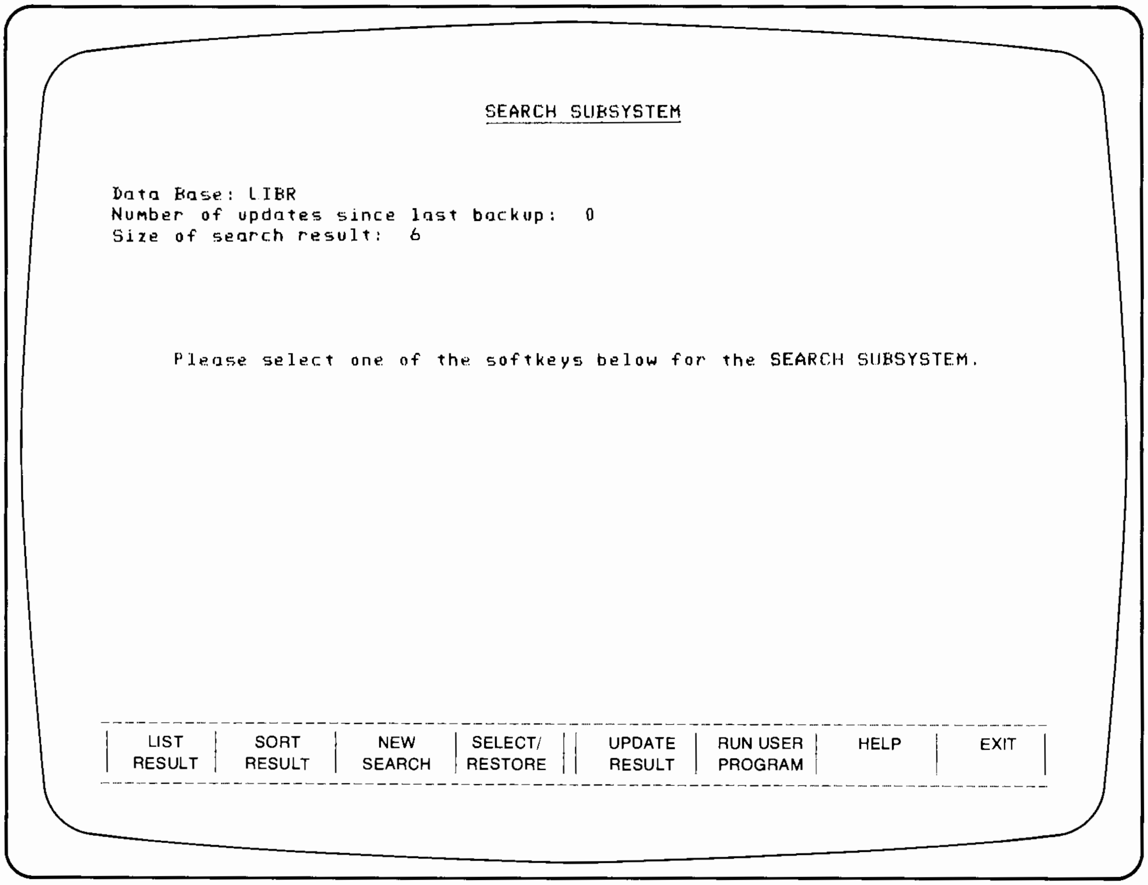
HIERARCHY = DEPT to DEPT_NAME to EMPLOY to EMP_NAME to CHILD

After a search is performed using this thread, an employee with no children does not appear in the search result. This illustrates that an entry is put in the search result only if the thread can be completely traversed from one end to the other. For example, if you wanted to find all employees working for the MANAGER Jack Smith, the HIERARCHY thread should not be used. Using the HIERARCHY thread, any employees without children who work for Jack Smith are not included in the search result. The following thread could be defined to find the employees:

EMPLOYEES = DEPT to DEPT_NAME to EMPLOY

Updating Entries Found by a Search

After you have performed a search, the entries found from the search can be updated. This is done by pressing the UPDATE RESULT softkey defined when this screen is displayed:



```
                         SEARCH SUBSYSTEM

Data Base: LIBR
Number of updates since last backup:   0
Size of search result:   6


     Please select one of the softkeys below for the SEARCH SUBSYSTEM.
```

The UPDATE RESULT softkey causes the UPDATE subsystem to be loaded into memory, enabling you to delete or modify entries or replace items that are contained in the search result.

Sorting Data Entries

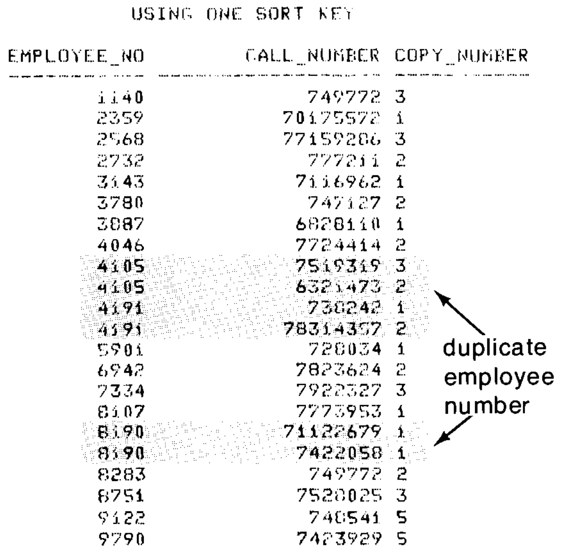
After a search has been performed, the SORT RESULT softkey may be used to sort the entries in the search result into a specific order. Sorting can be done using a single key by which to sort the entries; this is called a **simple sort**. For example, after a search to find all the books published after Jan 1, 1970, you can sort these books by author in ascending (alphabetical) or descending order. If the order is not specified, the entries are arranged in ascending order.

Sorting Using More Than One Key

Sorting the search result using many keys is called a **multiple-key sort**. With a multiple-key sort, the search result is sorted by the first key. Entries having the same value for the first key are then sorted by the second key. This process may continue for a total of **ten** keys. Ascending or descending order is specified for each of the keys.

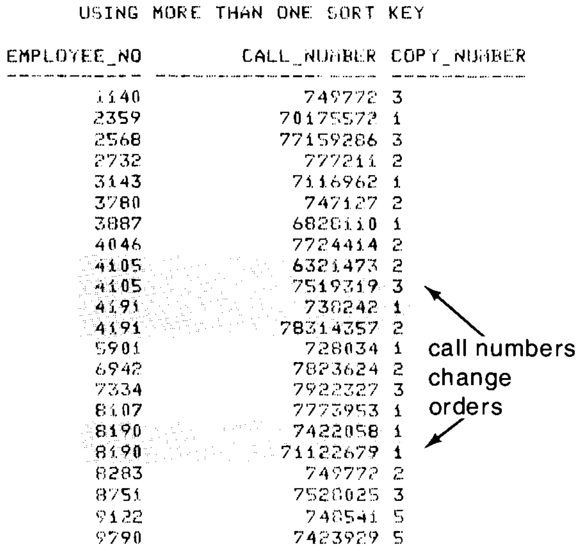
For example, suppose a search found all the books which were borrowed after Jan 1,1980. These books could be sorted by employee number and call number. All the employee numbers would be in ascending order, but if one employee checked out more than one book after Jan 1,1980, the book with the lowest call number would be listed first.

The following sorts show the difference between single and multiple-key sorts. The single key search was sorted by EMPLOYEE_NO. The multiple-key search was sorted by EMPLOYEE_NO and CALL_NUMBER.



```
        USING ONE SORT KEY

EMPLOYEE_NO          CALL_NUMBER COPY_NUMBER
-----------   ------------------ -----------
       1140               749772 3
       2359             70175572 1
       2568             77159286 3
       2732               777211 2
       3143              7116962 1
       3780               747127 2
       3887              6828110 1
       4046              7724414 2
       4105              7519319 3
       4105              6321473 2
       4191               738242 1
       4191             78314357 2
       5901               728034 1
       6942              7823624 2
       7334              7922327 3
       8107              7773953 1
       8190             71122679 1
       8190              7422058 1
       8283               749772 2
       8751              7528025 3
       9122               740541 5
       9790              7423929 5
```

duplicate employee number

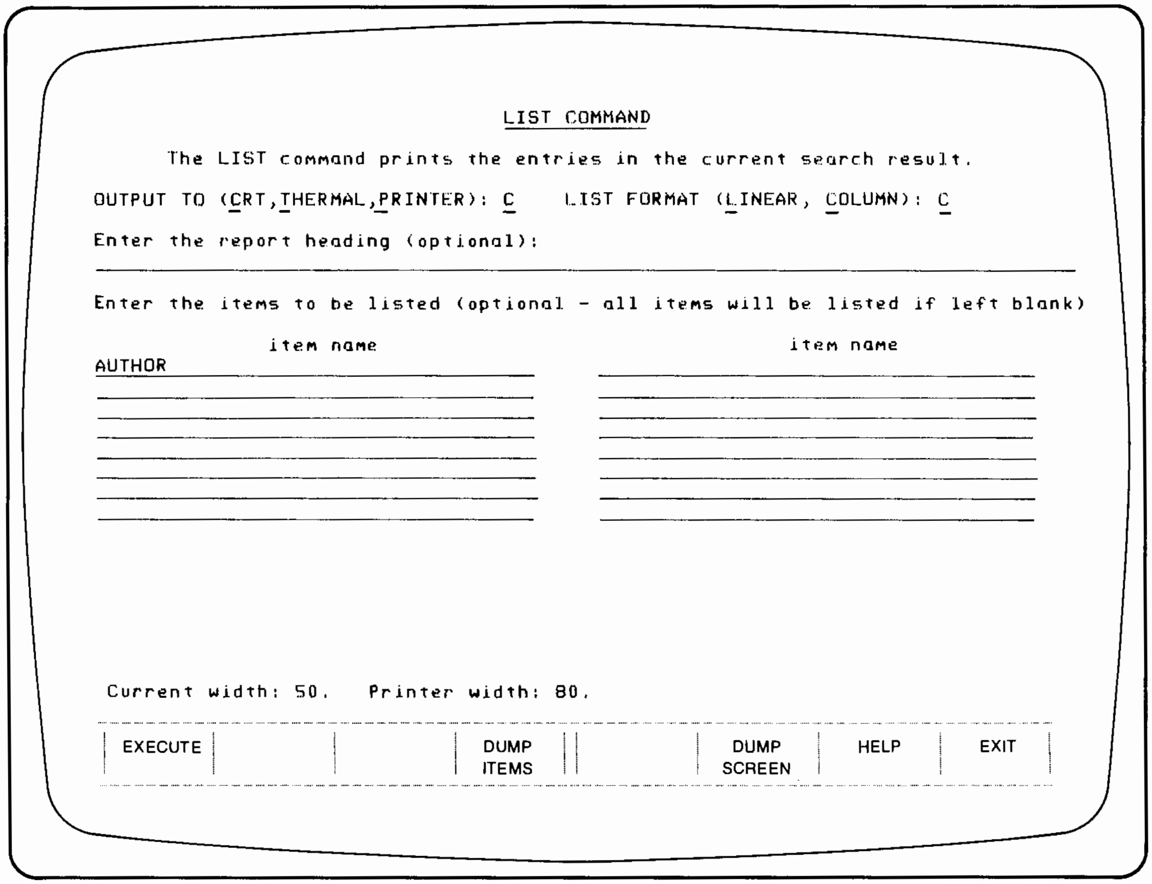


```
      USING MORE THAN ONE SORT KEY

EMPLOYEE_NO          CALL_NUMBER COPY_NUMBER
-----------   ------------------ -----------
       1140               749772 3
       2359             70175572 1
       2568             77159286 3
       2732               777211 2
       3143              7116962 1
       3780               747127 2
       3887              6828110 1
       4046              7724414 2
       4105              6321473 2
       4105              7519319 3
       4191               738242 1
       4191             78314357 2
       5901               728034 1
       6942              7823624 2
       7334              7922327 3
       8107              7773953 1
       8190              7422058 1
       8190             71122679 1
       8283               749772 2
       8751              7528025 3
       9122               740541 5
       9790              7423929 5
```

call numbers change orders

Listing the Data Entries

After a search is performed, use the LIST softkey to print the values of all or selected items from the entries in the search result. Each item value is preceded by the item name and set name printed as SET.ITEM. The printout can be in either linear or columnar format. Linear format prints one item per line; any value longer than one line is printed on the next line. Columnar format prints several items in a simple table. Here is a diagram of the list screen when using columnar format.



```
                              LIST COMMAND

     The LIST command prints the entries in the current search result.

OUTPUT TO (CRT,THERMAL,PRINTER): C     LIST FORMAT (LINEAR, COLUMN): C

Enter the report heading (optional):

Enter the items to be listed (optional - all items will be listed if left blank)

            item name                              item name
AUTHOR

 Current width: 50.   Printer width: 80.
```

Considerations for Columnar Format

When you specify columnar format, you must make sure that the sum of the maximum lengths of the items you want to list doesn't exceed the maximum line width. As you enter each item, this sum is indicated by the current line length. If the maximum length is exceeded, the items are cleared, an error message is printed and the cursor returns to the first line for entry.

The maximum line length of the CRT or internal printer is 80 characters. If you are using an external printer, the maximum line length was indicated during the System Configuration part at the beginning of QUERY/45.

For example, if you wanted to print the TITLE and AUTHOR of particular books on the CRT, then you would need to use linear format. TITLE was defined to be 60 characters long and BOOK was defined as 50 characters long. This totals 110 characters and the maximum line length of the CRT is 80 characters.

Specifying the Set

When item values are listed, the data item name is also listed, preceded by the set name and a period (set.item). When you specify the items to be listed, you can precede any item name with a set name and a period. If no set is specified, the set closest to the end of the thread that contains the item is used.

Chapter 6

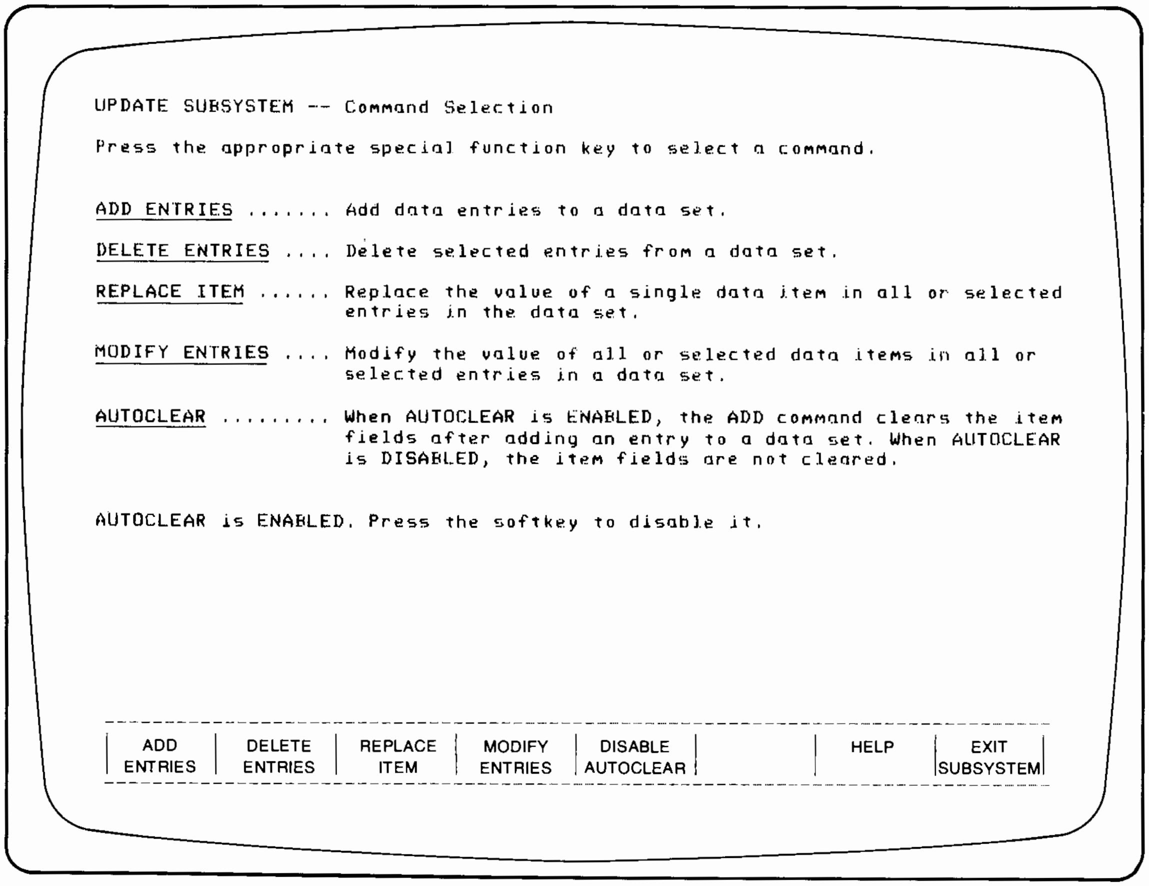
Changing the Data in the Data Base

The UPDATE subsystem assists you in modifying the data stored in your data base. There are four operations that can be performed on a data set. You can:

- add a data entry
- delete a data entry
- modify a data entry (change one or more item values in an entry)
- replace an item (change the value of a single data item in one or more entries)

Accessing the UPDATE Subsystem

To access the UPDATE subsystem, open the data base as explained in Chapter 2. Then press the UPDATE softkey. The screen now looks like:



```
UPDATE SUBSYSTEM -- Command Selection

Press the appropriate special function key to select a command.


ADD ENTRIES ....... Add data entries to a data set.

DELETE ENTRIES .... Delete selected entries from a data set.

REPLACE ITEM ...... Replace the value of a single data item in all or selected
                    entries in the data set.

MODIFY ENTRIES .... Modify the value of all or selected data items in all or
                    selected entries in a data set.

AUTOCLEAR ......... When AUTOCLEAR is ENABLED, the ADD command clears the item
                    fields after adding an entry to a data set. When AUTOCLEAR
                    is DISABLED, the item fields are not cleared.


AUTOCLEAR is ENABLED. Press the softkey to disable it.
```

After a softkey is pressed (ADD ENTRIES, DELETE ENTRIES, REPLACE ITEM, MODIFY ENTRIES), you are asked to select a data set to update from the data sets displayed on the screen. If any data sets are filled to their capacity (when adding entries) or empty (when deleting entries), they are listed next. The last line on the screen displays a message similar to:

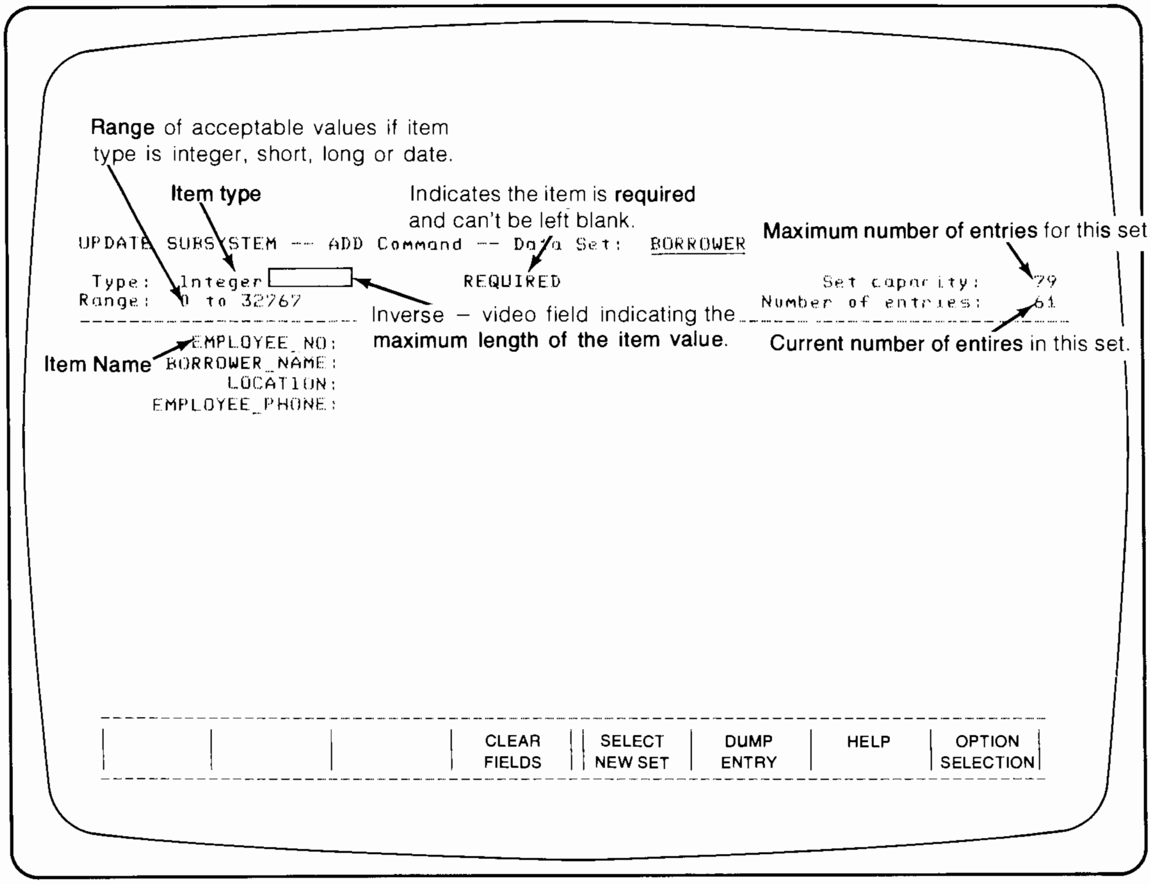
> "The remaining 4 data sets are automatic masters or are password protected"

This number refers to all automatic master sets (since these are not directly updated) and all data sets to which you do not have write access with the password used to open the data base.

Adding Data Entries

To add entries to a manual master or detail data set press the ADD ENTRIES softkey. After a set is specified, the screen displays the data item names on the left side of the CRT in the same order as the data base definition.

As you access each item, the following information is displayed:



```
UPDATE SUBSYSTEM -- ADD Command -- Data Set:  BORROWER

 Type:   Integer                    REQUIRED                       Set capacity:    79
Range:   0 to 32767                                           Number of entries:    61
-------------------------                                     ------------------------
       EMPLOYEE_NO:
     BORROWER_NAME:
          LOCATION:
    EMPLOYEE_PHONE:
```

Range of acceptable values if item type is integer, short, long or date.

Item type

Indicates the item is **required** and can't be left blank.

Maximum number of entries for this set.

Inverse – video field indicating the **maximum length of the item value.**

Item Name

Current number of entires in this set.

If the item is a code type, you may press the DUMP CODES softkey for a listing of the acceptable input values.

After a value is typed, press the CONTINUE, STORE, EXECUTE, DOWN ARROW (↓), or TAB key. QUERY/45 checks the value for correct type, range, code values, and makes sure that a non-blank value is entered for a required item. If the value is valid, the cursor moves to the next item. To move the cursor to a previously filled field, hold down the SHIFT key, and press TAB, or press the UP ARROW (↑) key.

After you have entered values for all the required items, an ADD ENTRY softkey is defined on the screen. Press this key to store the data entry into the data set.

Autoclear

There is a softkey defined at the beginning of the UPDATE subsystem that gives you the option of having AUTOCLEAR ENABLE or AUTOCLEAR DISABLE while adding entries. If AUTOCLEAR is enabled, all fields are cleared after you store an entry. If AUTOCLEAR is disabled, all fields are not cleared after you store the entry. This is useful when only one or two data item values change between entries.

Considerations

In general, always try to fill the manual master data sets first. When adding an entry to a detail data set, make sure that the values of key items are already stored in the related manual master(s). If you try to add an entry when the key value doesn't exist in the manual master, a error message is displayed. In this case, you can either add the key item value to the manual master without losing the detail data item values (press the ADD TO MANUAL softkey), or not add the entry into the detail data set (press the CLEAR FIELDS, SELECT NEW SET, or OPTION SELECTION softkey). Another choice is to change the field value for the item which is missing from the entry in the master data set (press the CHANGE KEY ITEM softkey).

Using a Form for Adding Entries

If you have defined a form (forms are covered in Chapter 7) for the data set, QUERY/45 asks if you want to use it for data input. If you choose to use it, the screen is cleared and the form is displayed. The information that is normally displayed at the top of the screen when using the default screen image is not displayed while using a pre-defined form.

Deleting Data Entries

By pressing the DELETE ENTRIES softkey at the beginning of the UPDATE subsystem, you can search for and delete entries from a selected data set. You can only delete complete entries in a set, not individual data items.

A search expression is used to select the entries to be deleted. When the search is completed, the number of entries meeting the search criteria is displayed. Pressing the PAUSE DELETE softkey enables you to examine each of these entries. If the complete entry doesn't fit on the screen, the DOWN ARROW (↓) and UP ARROW (↑) keys enables you to scroll the entry. If you want to delete the entry displayed on the screen press the DELETE ENTRY softkey; otherwise press the NEXT ENTRY softkey. If all the entries found by the search are to be deleted, press the DELETE ALL softkey.

Consideration

When deleting entries in a manual master data set, you need to ensure that the key item value in a particular entry is not found in a related detail set entry. If you try to delete a master entry that has detail entries linked to it, a warning is displayed on the CRT. You must delete or change the entries in the detail data sets before the manual master entry can be deleted.

Modifying Data Entries

By pressing the MODIFY ENTRIES softkey, you can change the values of data items in one or more data entries. For example, MODIFY might be used if a particular employee moves to a different department. A search expression is entered to select the entries to be modified. When the search is completed, each entry can be modified.

After a set is specified and a search is performed, the screen displays each entry in the same format as in the ADD command. If the current data item is a code type, you may press the DUMP CODES softkey for a listing of the acceptable values.

You now may edit the entry being displayed. Whether or not a data item value is modified, press CONTINUE to enter the value and move the cursor to the next item. This causes the value to be checked for correct type, range and code values. If the item is required, QUERY/45 checks to make sure a value was entered.

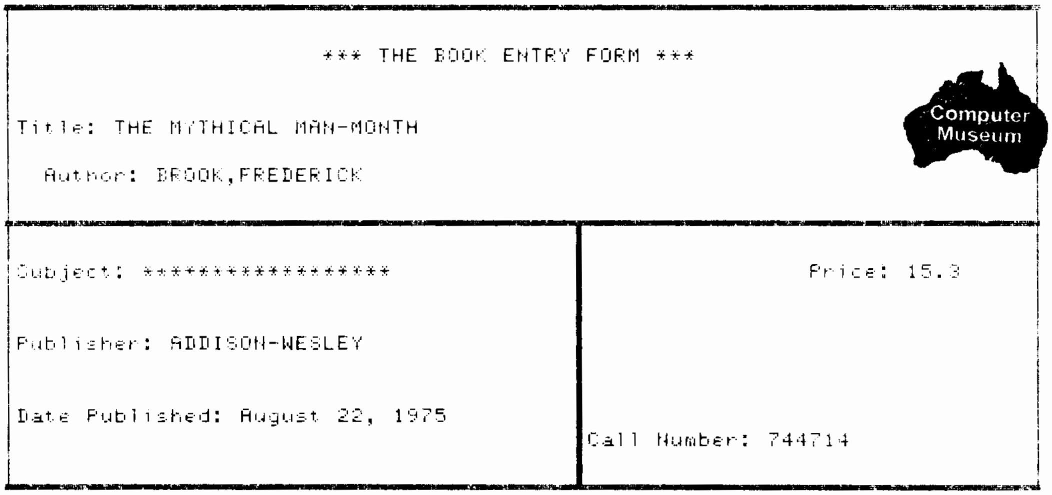
If you don't want to modify the entry displayed, press the NEXT ENTRY softkey to display the next entry in the search result.

When you have changed all the desired items in a particular data entry, press the MODIFY ENTRY softkey. This replaces the old entry in the data base with the new entry, and then displays the next entry from the search result for editing.

Using a Form for Modifying Entries

If you have defined a form for the data set being modified, QUERY/45 asks if you want to use it for data modification. If you select to use the form, the form for the specified data set is displayed on the screen. The information that is displayed at the top of the screen if you are not using a form is not displayed while using a pre-defined form.

Using a form for modification has one limitation. If the current value for a data item is longer than the field defined on the form, the field is filled with asterisks (*) and you cannot move the cursor into the field to modify it. For example, when defining a form for the BOOK set, you may have thought that all the values for SUBJECT were less than 15 characters long. In one entry, however, the SUBJECT has 17 characters. When modifying this particular entry, the form would be displayed as shown next and you couldn't modify SUBJECT for this entry.



```
                    *** THE BOOK ENTRY FORM ***

Title: THE MYTHICAL MAN-MONTH

  Author: BROOK,FREDERICK

Subject: *****************                        Price: 15.3

Publisher: ADDISON-WESLEY

Date Published: August 22, 1975
                                   Call Number: 744714
```

Considerations

When modifying an entry in a detail data set, make sure that the values of key items are already stored in the related manual master(s). If you try to modify an entry when the key value doesn't exist, the choices QUERY/45 allows you to make are the same as in ADD.

Replacing Data Items

Press the REPLACE ITEM softkey when you want to change one particular value for a data item which is found in all or selected entries of a data set. For example, if the person entering the date on which certain books were borrowed entered Sunday's date instead of Monday's date, you could correct all of the occurrences of the date by using the REPLACE ITEM softkey. You can replace any item value in a detail data set including the key item value. When replacing the key item value in a manual master data set, the key item can only be modified if one entry is found in the search result.

When replacing an item, you need to specify the data set containing the data item. You are then asked for the data item to be replaced. If it is a compound item, the particular subitem must be specified, such as ADDRESS(2). If you are not sure whether an item is compound or which items are in the data set, press the DUMP ITEMS softkey to produce a listing of the items on the internal printer.

After you have entered the data item name, enter its new value. If the item is not required, the new value can be left blank to store a null value for the item.

A search expression is then used to select the data entries in which the data item value is to be replaced. After the search is performed, press the REPLACE ALL softkey to replace the item value in all the entries found by the search, or press the PAUSE REPLACE softkey to see each entry. If you want to replace the entry displayed on the screen, press the REPLACE ENTRY softkey; otherwise press the NEXT ENTRY softkey.

Example

Replacing an item in the LIBR data base would be needed if DCD changed their telephone number from (303)226-3800 to (303)524-5683. Thus, all the entries in the BORROWER set with the old telephone number (303)226-3800 need to be changed to the new telephone number (303)524-5683. You would press the REPLACE ITEM softkey and specify the BORROWER data set by typing the set number 2. For the name of the item to be replaced, type EMPLOYEE_PHONE; for the new value type (303)524-5683. When prompted for the search expression, type EMPLOYEE_PHONE = (303)226-3800. When the search is complete, the screen displays the number of entries found. You really don't need to press the PAUSE REPLACE softkey because you know that all phone numbers with (303)226-3800 need to be changed. Press the REPLACE ALL softkey and the replace is done for all the entries.

Considerations

When replacing a key item value in a detail data set, make sure that the new value for the key item is already stored in the related manual master. If the new key value does not exist in the manual master, an error message is displayed and the replace operation is ignored.

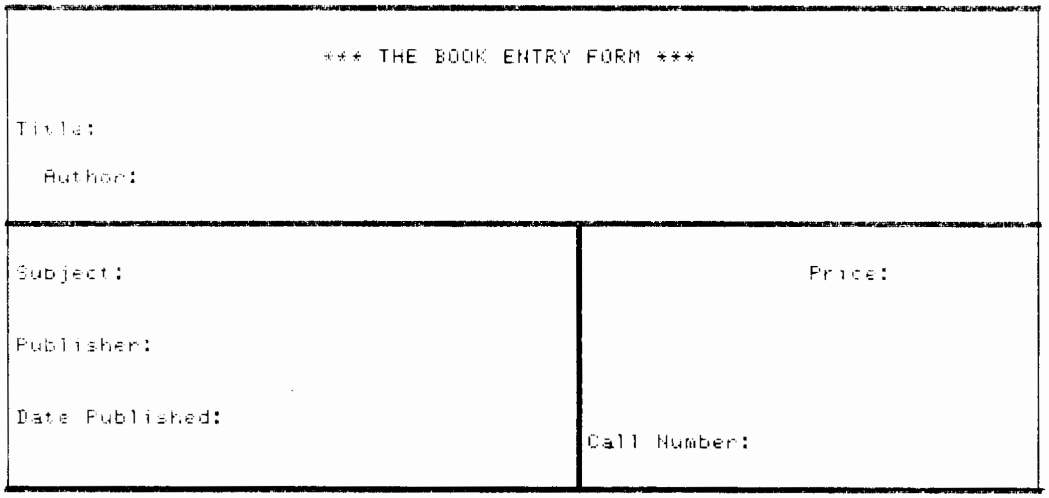
Chapter 7

Using Forms with your Data Base

Introduction

The FORMS subsystem enables you to define a form for each manual master or detail data set. The **form** is an image displayed on the CRT and can include heading, text, lines for dividing the form into sections, a field for entering each item value and additional information, such as a more complete description of a data item name. It can be used while adding or modifying data entries during the UPDATE subsystem. The advantage of using a form is that additional information can be displayed when entering data that is not available if a form is not used.

The following form is used for the examples in this chapter.



```
                    *** THE BOOK ENTRY FORM ***

Title:

  Author:

Subject:                                    Price:

Publisher:

Date Published:
                            Call Number:
```

Forms can include the following features:

- Video highlighting for text and input fields (such as blinking)
- Line drawing characters to partition the form
- Definition of fields for data item values
- Definition of tab order for entering the data items

The FORMS subsystem is run in two stages. In the first stage you create the form. The second stage occurs after the form image has been created and consists of defining the tab order and storing the form.

Limitations

Each form has a maximum of 66 lines. One field must be defined for each item and subitem in the data set. This field cannot be longer than the maximum length of the item defined during the definition of the data base, but it can be shorter. For example, you might want to make the field shorter for the item called PHONE_NO. A phone number with an area code preceding it has 13 characters — (303)987−6543. The DEFINE subsystem requires a character string to be an even number of characters, so PHONE_NO is defined as 14 characters long. When you create a form for the data set which contains PHONE_NO, you would make the field only 13 characters long.

Creating and Modifying a Form

Pressing the DEFINE FORM softkey accesses the FORMS subsystem. This softkey is accessed by pressing the MORE KEYS softkey in the main QUERY subsystem. You then specify the data set for which you want to create a form. The only sets that you can create forms for are manual masters and detail data sets to which you have read/write access with the password. The CRT is cleared and when the cursor appears at the upper left corner of the CRT, you are ready to begin creating the form.

A form is defined by drawing lines to divide the form into sections, entering text to name a data item field or provide additional information and defining a field for each data item. Additionally, any of the text can be highlighted with video highlights such as inverse video. Text and line segments are entered at the current cursor position.

If a form exists for the specified data set, you can RECALL the form to modify it, DELETE the form so that it no longer is defined for that data set, define a NEW FORM which replaces the present form for that data set, or select a new data set.

If you RECALL the form and it has more than 20 lines, the bottom line of the CRT displays the loading of the extra lines of the form. When the form loading is complete, the first twenty lines are displayed on the CRT. When the cursor appears at the upper left corner of the CRT, you are ready to begin editing the form. The keys have the same definintion as when creating a form, as explained in the next section.

Key Definitions

All alphanumeric keys and the following keys retain their normal definitions while defining a form.

BACK SPACE	INS LN	TAB
DEL CHAR	REPEAT	TAB CLEAR
DEL LN	SHIFT	TAB SET
INS CHR	TYPWTR	

The following keys have been redefined so that pressing:

HOME — Moves the cursor to the left-most position of the current line.

SHIFT HOME — Moves the cursor to the top left of the screen.

STORE — Moves the cursor to the left-most position of the next line of the form.

CLR→END — Clears the screen image from the cursor to the end of the line or to the next field, whichever occurs first.

CLEAR LINE — Clears the current line of the form, excluding fields.

SHIFT TAB — Moves the cursor backward to the previous tab position.

RECALL — Moves the cursor to the top left of the form. The top left of the form may not be visible on the screen if the form is longer than 20 lines.

SHIFT RECALL — Moves the cursor to the bottom left of the form. The bottom left of the form may not be visible on the screen if the form is longer than 20 lines.

Moving the Cursor

The ARROW keys are used to move the cursor on the screen in the direction specified by the arrow. The ROLL UP key moves the form up ten lines. The cursor remains in its relative screen position (the form moves with respect to the cursor). The ROLL DOWN key moves the form down ten lines. The cursor remains in its screen relative position.

Video Highlights

Video highlights can be used to accent important items in the form. The video highlights are accessed as they normally are on the System 45, by holding down CONTROL and pressing the appropriate SFKs. The highlight remains in effect until you press CONTROL and the SFK or SFKs which you selected to turn it on.

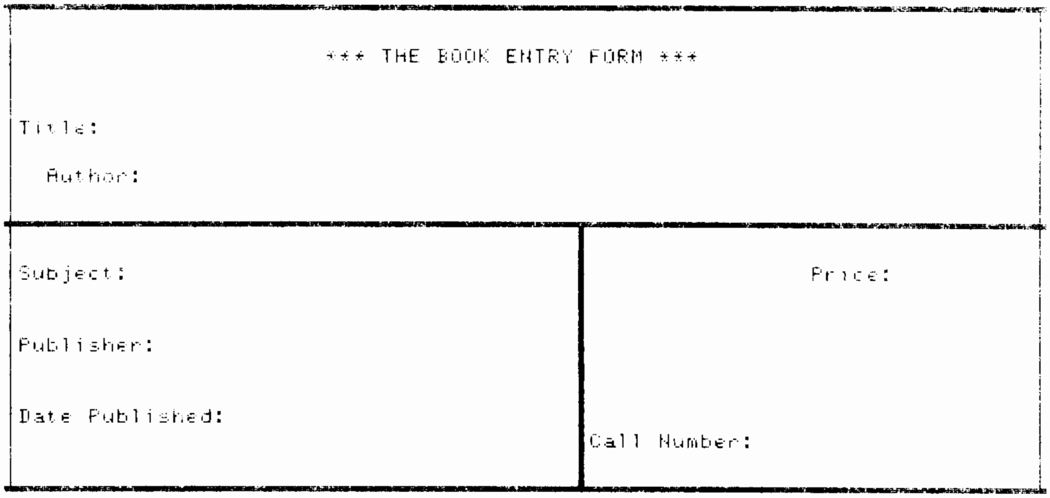
CONTROL-SFK 0	Inverse Video
CONTROL-SFK 1	Blinking
CONTROL-SFK 2	Underline

Drawing Lines

After pressing the LINE DRAW CHARS softkey, the softkeys are defined with the line drawing characters. Pressing any of the first six softkeys draws that line segment at the present cursor location. Pressing MORE KEYS enables you to access the other sets of line drawing characters. Pressing EXIT returns you from the line drawing keys to the previous level of FORMS.

Example

Specify the BOOK data set when QUERY/45 asks you for a data set. Use the LINE DRAW CHARS sets to draw the border around the example form, as shown here.



*** THE BOOK ENTRY FORM ***

Title:

Author:

Subject:

Price:

Publisher:

Date Published:

Call Number:

NOTE

The horizontal lines appear as dashed lines on the CRT. When the form is output to the thermal printer, the lines are solid as shown.

Defining Fields

Each item and subitem in the set must have a field to contain the value when data is entered. A field can be defined at any point in the creation of the form. Defining a field consists of specifying the starting position, the corresponding data item and the length.

To define a field, position the cursor where you would like the field to begin and press the START A FIELD softkey. QUERY/45 then asks you for the item number (which is based on the order of the items defined in the data set). The ITEM LISTING softkey lists the item names, their corresponding numbers, and maximum lengths. You need to enter the maximum number of characters for each field. The maximum length of the field is the maximum length specified when the item was defined.

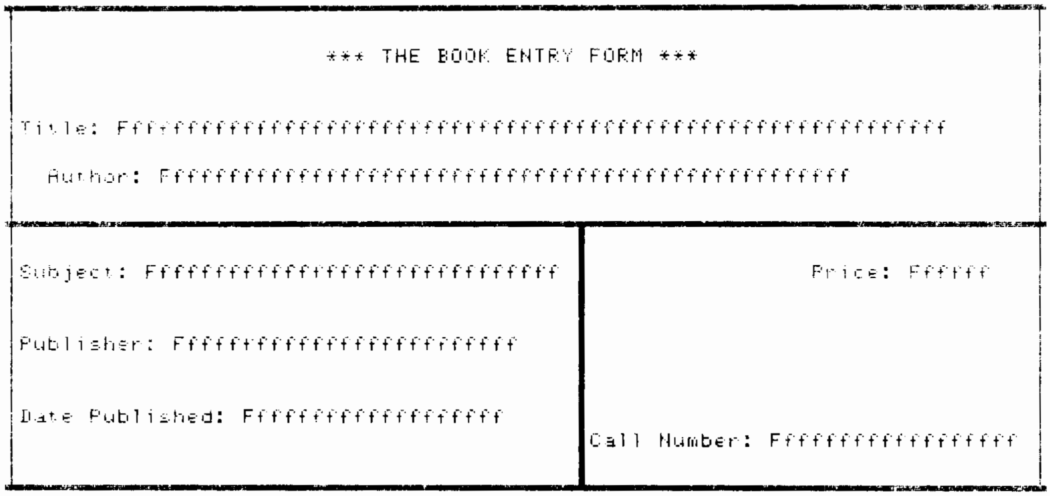
The first character in a field is identified by an uppercase F. All subsequent characters are identified by lowercase f's. The order in which fields are defined is not important; order specification occurs after all fields are defined.

NOTE

No area in a form can be shared by two or more fields.

Example

After all fields in the example form have been defined, the form appears as:



```
                         *** THE BOOK ENTRY FORM ***

Title: Ffffffffffffffffffffffffffffffffffffffffffffffffffffffffffff

  Author: Fffffffffffffffffffffffffffffffffffffffffffffffff

Subject: Fffffffffffffffffffffffffffff                   Price: Ffffff

Publisher: Ffffffffffffffffffffffff

Date Published: Ffffffffffffffffff
                                          Call Number: Fffffffffffffffff
```

Deleting a Field

Deleting a field is accomplished by positioning the cursor anywhere within the field and pressing the DELETE A FIELD softkey.

Moving a Field

To move a field, position the cursor anywhere within the field and press the MOVE A FIELD softkey. Move the cursor to the new location on the form and press the PLACE FIELD softkey. The field is placed in the new location unless it overlaps another field or it is at the end of the form. In this case, an error message is displayed and you must move the field to another location on the form. The CANCEL MOVE softkey puts the field back to its original position before the MOVE A FIELD softkey was pressed.

Processing the Form

After you have finished defining the lines, text and fields on the form, press the PROCESS FIELDS softkey which appears after you have defined a field for every data item. The FORMS subsystem then enters the portion of the program where you can define tab order.

Defining the Tab Order

Tab order is the sequence in which the cursor is moved from field to field when entering data using a form. There is a default order which corresponds to the order in which items appear in the definition of the set.

Pressing ORDER FORM enables you to define a different tab order. The form now has an arrow symbol (↑) pointing to the first character of the first field on the form. At this time you can change the tab order by typing in a number which specifies when, in the tab order, this field should be accessed, then press CONTINUE. If you do not want to change the tab number of a field, press CONTINUE.

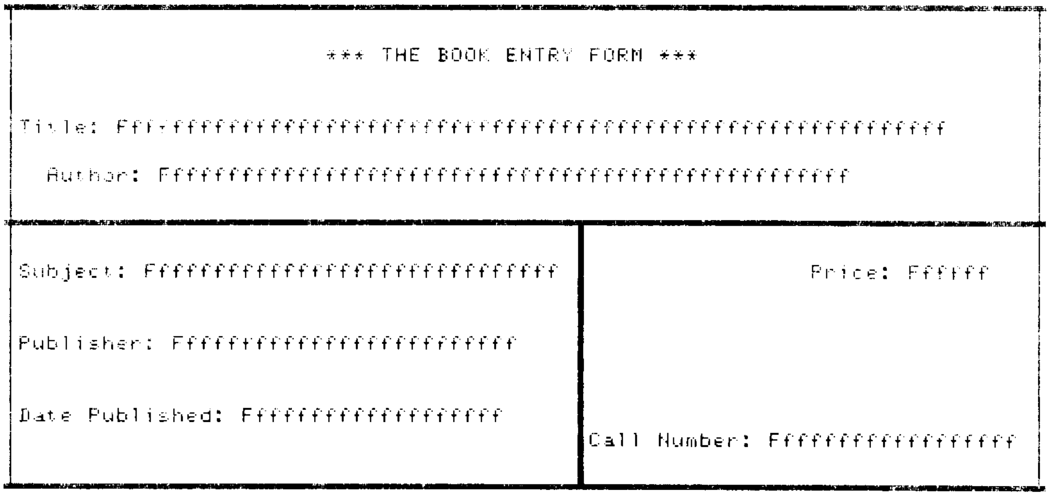
The process of specifying tab order continues until each field has the desired values. Press PROCESS TABS to leave the tab definition mode.

Considerations

You can use either real numbers or integers for tab order. However, when the tab numbers are processed, they are renumbered with integers. When two fields have the same tab number, they are renumbered using the default order. For example, if two fields have a value of 4, the field with the lowest item number would be numbered 4, the other with 5.

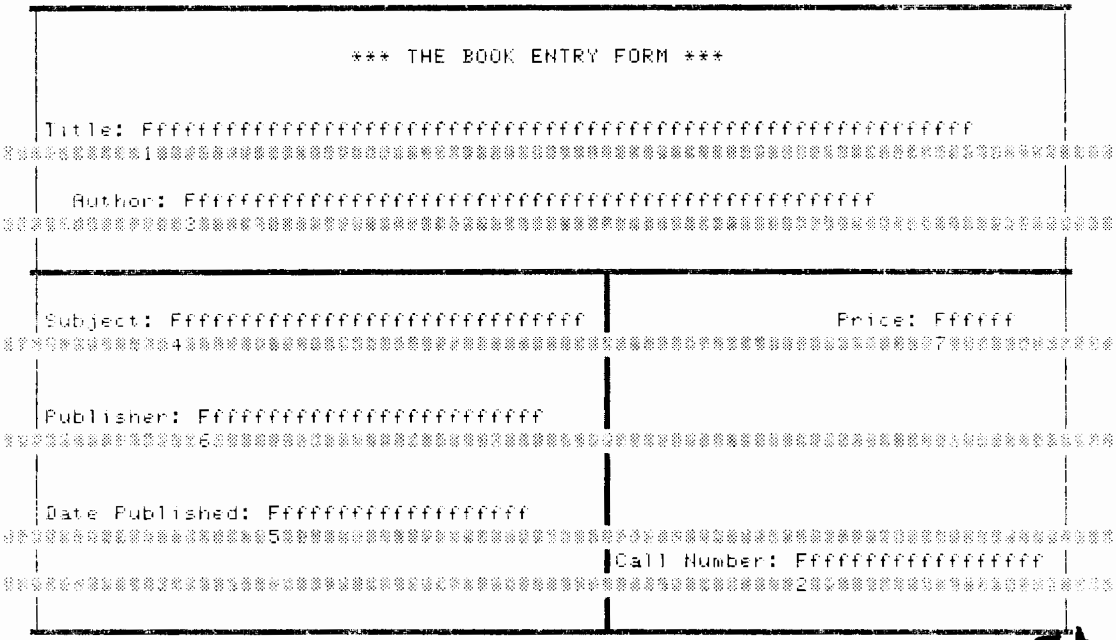
Printing the Form

To see what the form looks like, press the PRINT REPORT softkey and three copies of the form are printed on the internal printer. The first copy shows you what the complete form looks like. The second copy shows you the item numbers as they are currently defined on the form. The third copy of the form shows you the tab order as it is currently defined for the form. The printout produced for the example form look like this.



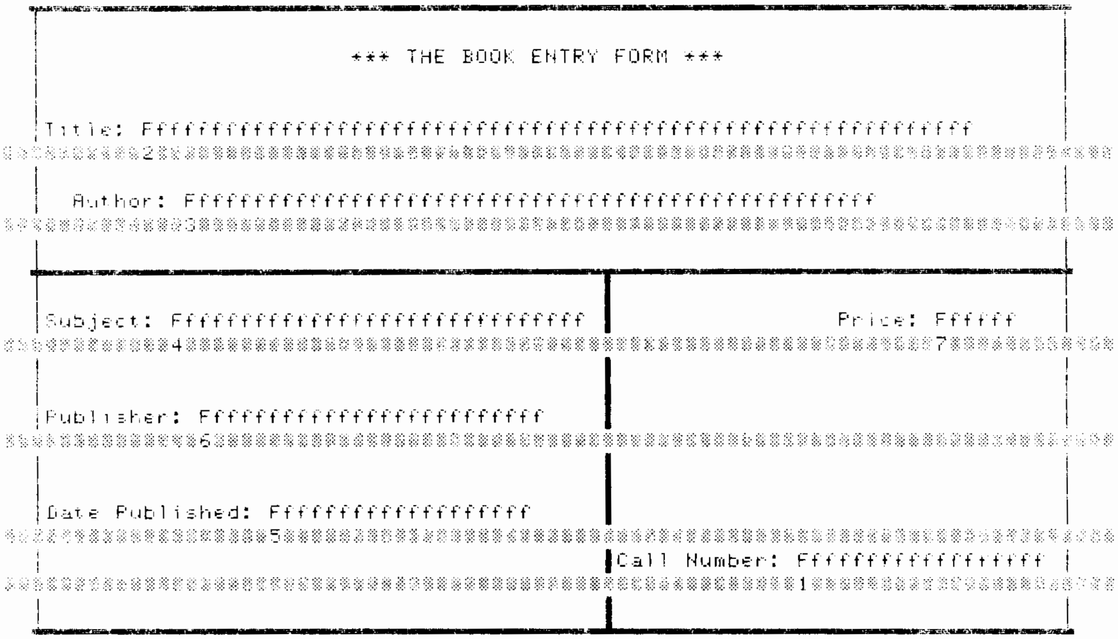
```
                         *** THE BOOK ENTRY FORM ***

Title: Fffffffffffffffffffffffffffffffffffffffffffffffffffffffffff

  Author: Fffffffffffffffffffffffffffffffffffffffffffffffff

Subject: Fffffffffffffffffffffffffffff              Price: Ffffff

Publisher: Ffffffffffffffffffffffff

Date Published: Ffffffffffffffffff
                                     Call Number: Fffffffffffffffff
```

Item Numbers



```
                         *** THE BOOK ENTRY FORM ***

   Title: Fffffffffffffffffffffffffffffffffffffffffffffffffffffffffff
▒▒▒▒▒▒▒▒▒▒1▒▒▒▒▒▒▒▒▒▒▒▒▒▒▒▒▒▒▒▒▒▒▒▒▒▒▒▒▒▒▒▒▒▒▒▒▒▒▒▒▒▒▒▒▒▒▒▒▒▒▒▒▒▒▒▒▒▒▒▒▒▒▒▒▒
     Author: Fffffffffffffffffffffffffffffffffffffffffffffffff
▒▒▒▒▒▒▒▒▒▒▒▒▒3▒▒▒▒▒▒▒▒▒▒▒▒▒▒▒▒▒▒▒▒▒▒▒▒▒▒▒▒▒▒▒▒▒▒▒▒▒▒▒▒▒▒▒▒▒▒▒▒▒▒▒▒▒▒▒▒▒▒▒▒

   Subject: Fffffffffffffffffffffffffffff                  Price: Ffffff
▒▒▒▒▒▒▒▒▒▒▒▒4▒▒▒▒▒▒▒▒▒▒▒▒▒▒▒▒▒▒▒▒▒▒▒▒▒▒▒▒▒▒▒▒▒▒▒▒▒▒▒▒▒▒▒▒7▒▒▒▒▒▒▒▒▒▒▒▒

   Publisher: Ffffffffffffffffffffffff
▒▒▒▒▒▒▒▒▒▒▒▒▒▒6▒▒▒▒▒▒▒▒▒▒▒▒▒▒▒▒▒▒▒▒▒▒▒▒▒▒▒▒▒▒▒▒▒▒▒▒▒▒▒▒▒▒▒▒▒▒▒▒▒▒▒▒▒▒▒▒▒

   Date Published: Ffffffffffffffffff
▒▒▒▒▒▒▒▒▒▒▒▒▒▒▒▒▒▒▒5▒▒▒▒▒▒▒▒▒▒▒▒▒▒▒▒▒▒▒▒▒▒▒▒▒▒▒▒▒▒▒▒▒▒▒▒▒▒▒▒▒▒▒▒▒▒▒▒▒▒
                                            Call Number: Fffffffffffffffff
▒▒▒▒▒▒▒▒▒▒▒▒▒▒▒▒▒▒▒▒▒▒▒▒▒▒▒▒▒▒▒▒▒▒▒▒▒▒▒▒▒▒▒▒▒▒▒▒▒▒▒▒▒▒▒▒▒2▒▒▒▒▒▒▒▒▒▒▒▒▒▒▒▒▒▒▒▒▒▒
```

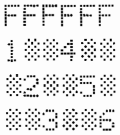
Tab Numbers



```
                         *** THE BOOK ENTRY FORM ***

   Title: Fffffffffffffffffffffffffffffffffffffffffffffffffffffffffff
▒▒▒▒▒▒▒▒▒▒2▒▒▒▒▒▒▒▒▒▒▒▒▒▒▒▒▒▒▒▒▒▒▒▒▒▒▒▒▒▒▒▒▒▒▒▒▒▒▒▒▒▒▒▒▒▒▒▒▒▒▒▒▒▒▒▒▒▒▒▒▒▒▒▒▒
     Author: Fffffffffffffffffffffffffffffffffffffffffffffffff
▒▒▒▒▒▒▒▒▒▒▒▒▒3▒▒▒▒▒▒▒▒▒▒▒▒▒▒▒▒▒▒▒▒▒▒▒▒▒▒▒▒▒▒▒▒▒▒▒▒▒▒▒▒▒▒▒▒▒▒▒▒▒▒▒▒▒▒▒▒▒▒▒▒

   Subject: Fffffffffffffffffffffffffffff                  Price: Ffffff
▒▒▒▒▒▒▒▒▒▒▒▒4▒▒▒▒▒▒▒▒▒▒▒▒▒▒▒▒▒▒▒▒▒▒▒▒▒▒▒▒▒▒▒▒▒▒▒▒▒▒▒▒▒▒▒▒7▒▒▒▒▒▒▒▒▒▒▒▒

   Publisher: Ffffffffffffffffffffffff
▒▒▒▒▒▒▒▒▒▒▒▒▒▒6▒▒▒▒▒▒▒▒▒▒▒▒▒▒▒▒▒▒▒▒▒▒▒▒▒▒▒▒▒▒▒▒▒▒▒▒▒▒▒▒▒▒▒▒▒▒▒▒▒▒▒▒▒▒▒▒▒

   Date Published: Ffffffffffffffffff
▒▒▒▒▒▒▒▒▒▒▒▒▒▒▒▒▒▒▒5▒▒▒▒▒▒▒▒▒▒▒▒▒▒▒▒▒▒▒▒▒▒▒▒▒▒▒▒▒▒▒▒▒▒▒▒▒▒▒▒▒▒▒▒▒▒▒▒▒▒
                                            Call Number: Fffffffffffffffff
▒▒▒▒▒▒▒▒▒▒▒▒▒▒▒▒▒▒▒▒▒▒▒▒▒▒▒▒▒▒▒▒▒▒▒▒▒▒▒▒▒▒▒▒▒▒▒▒▒▒▒▒▒▒▒▒▒1▒▒▒▒▒▒▒▒▒▒▒▒▒▒▒▒▒▒▒▒▒▒
```

```
Data Items for Data Set:  BOOK

Item      Item Name                Maximum   Item      Item Name            Maximum
Number                             Length    Number                         Length
------------------------------------------------------------------------------------
    1  TITLE                        60           5  PUBLISHED_DATE           19
    2  CALL_NUMBER                  18           6  PUBLISHER                30
    3  AUTHOR                       50           7  PRICE                    14
    4  SUBJECT                      40
```

The item and tab numbers are read similarly; a line of null characters is printed under each line of the form which contains a field. At the position under the first character of the field is a number which represents its item number or the tab order. When there are several character fields that are adjacent to each other, item and tab numbers are specified by multiple lines of null characters under the fields. The numbers are staggered so that the initial number of each field is under the first character of its field as shown here.



```
FFFFFF
1▒▒4▒▒
▒2▒▒5▒
▒▒3▒▒6
```

Storing the Form

The final step in the creation of a form is to store it on a mass storage medium. The form is stored by pressing the STORE FORM softkey. You are asked for the volume name of the disc on which the form is to be stored. If you are modifying a form and you are going to store the form on the same volume as the original form, the original form is purged before the modified form is stored.

Press the EXIT softkey to return to the following softkey selections:

DEFINE FORM	DEFINE THREAD			RUN USER PROGRAM	PREVIOUS KEYS	HELP	CLOSE DATA BASE

Chapter 8
Defining a Data Base

Introduction

The DEFINE subsystem is used to interactively define or redefine a data base to QUERY/45. The final output of the DEFINE subsystem is a data base **root file** (containing all the structural information about the data base), an **information file** for the data base (containing information about the data base and data set descriptions, name, code and date type data items, synonyms, ranges, forms and threads), and optionally, the data set files into which the data is stored. The information file contains supplementary information about a data base; this file is not used by IMAGE/45.

An advantage of using the DEFINE subsystem over the IMAGE/45 Schema Processer is that you can use the special data types (code, name, date), synonyms, and ranges. When using the DEFINE subsystem, your only permanent record of the data base definition is the data base root file and the information file that are produced. There is no schema text file. Copies of each file must be made in order to preserve a copy of the data base definition.

This chapter has three main sections:

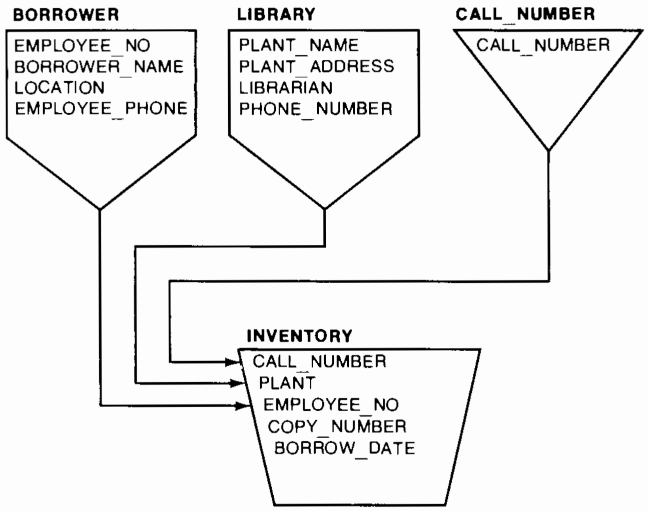
- Planning before you define a data base (definitions and terms)
- Using QUERY/45 to define a data base
- A step-by-step example

Once you have designed a data base, the softkey definition section gives you the basic information needed to define a data base. Other information is provided on the screen. You may want to turn to the definitions or the example first to obtain added information before beginning.

Planning Before you Define a Data Base

The Data Base Diagram

The first step in defining a data base is to use the Data Base Design Kit to take you from a situation where a data base is needed to a diagram similar to this:



Item and Set Information

What does the previous diagram show? There are four different sets. A name is associated with each data set (LIBRARY, CALL_NUMBER, BORROWER and INVENTORY). Their shapes indicate that two sets are manual master data sets (LIBRARY and BORROWER), one is an automatic master data set (CALL_NUMBER),and one is a detail data set (INVENTORY). Additionally, the diagram contains all the item names. The key items are items on either end of the paths which link the sets together. Here is a list of all the names with the key items starred (*).

Data Set Name	Item Name	Key Item
LIBRARY	PLANT_NAME	(*)
	PLANT_ADDRESS	
	LIBRARIAN	
	PHONE_NUMBER	
CALL_NUMBER	CALL_NUMBER	(*)
BORROWER	EMPLOYEE_NUMBER	(*)
	BORROWER_NAME	
	LOCATION	
	EMPLOYEE_PHONE	
INVENTORY	CALL_NUMBER	(*)
	COPY_NUMBER	
	PLANT	(*)
	EMPLOYEE_NO	(*)
	BORROWER_DATE	

Beyond the Diagram

After the diagram is complete, the following things must be determined. These items are explained in detail in the following sections. This information can also be found on the screen during the appropriate part of DEFINE.

data base name
1 to 4 alphanumeric characters; the first must be a letter. This name is used to identify the data base.

data base description
1 to 60 optional character string. This description is used to help identify the data base since the data base name is so short.

volume label
1 to 8 ASCII character string excluding blanks, colons, and commas. This label must be printed on the disc before you start running the DEFINE subsystem. A data base may reside on more than one disc and therefore have many labels. You should label each disc with a different name. You specify which data set or sets are to be stored on which disc when defining the data sets.

passwords
1 to 8 characters in length including upper and lowercase letters, digits 0 thru 9, and all other printable characters excluding blanks and semicolons.

To access the data base, a valid password must be be given. Thus, you need only tell passwords to those people you want accessing the data base.

If no passwords are defined, unrestricted access is given for every data set in the data base. Any particular password can be defined to give access to any of the data sets. Each password can have READ-ONLY access, READ/WRITE access, or NO access to a particular data set.

The different types of access are specified with R for READ-ONLY, W for READ/WRITE, and nothing is typed for NO access.

Up to 31 passwords can be defined per data base.

data set name
1 to 15 characters, including letters, digits 0 thru 9, and the underscore (_) character. The first character must be a letter. Each set name must be unique in the data base. Up to 32 data sets can be defined per data base.

data set description
1 to 60 character string which describes the data stored in the data set. This description is optional.

data set type
M specifies MANUAL
A specifies AUTOMATIC
D specifies DETAIL

data set capacity

Integer from 1 to 32 767 (fractions are truncated). For master data sets, choose a prime number and a capacity which is 25% greater than the number of entries you intend to have. This helps ensure rapid searches.

data item name

1 to 15 characters, including letters, digits 0 thru 9, and the underscore (_) character. The first character must be a letter. Each item name must be unique in the data set and in the data base.

A maximum of one data item can be defined for an AUTOMATIC data set, 127 items for MANUAL and DETAIL data sets, and 255 items for a data base.

synonyms

1 to 15 characters, including upper and lowercase letters, digits 0 thru 9, and the underscore (_) character. The first character must be a letter. Each synonym must be unique and cannot match an existing data item name.

Up to five synonyms can be defined for any data item and no more than 250 synonyms can be defined for the data base.

data item types:

character string

X represents CHARACTER string. Any even number of characters from 2 to 1022. If an odd number is entered, QUERY/45 changes it to the next higher number. The actual upper bound is determined by the amount of room remaining in the data set entry.

long

L represents LONG numeric. A full-precision real number with 12 significant digits and an exponent in the range −99 to 99.

short

S represents SHORT numeric. A short-precision number with 6 significant digits and an exponent with the range −63 to 63.

integer

I represents INTEGER. A whole number between −32 767 and 32 767.

code

C represents CODE. 1 to 15 character string using any printing character including blanks. A code value need only be unique to a particular data item.

Up to 35 code values can be defined for a data item. A maximum of 250 code values can be defined for each data base.

name

N represents NAME. The length is any even number of characters from 4 to 80. The actual upper bound is determined by the amount of room remaining in the data set entry.

date

D represents DATE. It is converted and stored as a short numeric. Any date can be entered between January 1, 0000 and September 11, 5475.

subitem count

The number of subitems in addition to the original, simple data item. The size of each subitem is the same as defined for the original data item. The maximum number of subitems depends on the amount of room available in the data entry.

maintenance word

1 to 6 ASCII character string. A maintenance word is used to protect the entire data base against unauthorized people purging, erasing, backing up, or modifying the data base. It is similar to a password, except passwords enable only certain people to update the data in a set; a maintenance word helps prevent purging or restructuring of the whole data base. If the field is left blank, anyone can purge or restructure the data base.

Using QUERY/45 to Define the Data Base

You are now ready to run the DEFINE subsystem of QUERY/45. To load the DEFINE subsystem, press the DEFINE DATA BASE softkey at the beginning of QUERY/45 when these softkeys are displayed:

SHOW BASES		DEFINE DATA BASE	CONFIGURE SYSTEM	EDIT BASE CATALOG		HELP	STOP

When the DEFINE sybsystem is loaded, press the DEFINE A DATA BASE softkey. Keep the diagram you drew handy to use as a reference when QUERY/45 asks you for information.

The steps for defining a data base are:

1. Enter the data base information. Press PROCESS BASE INFO when the information is correct.
2. Enter the information for the first data set. Press PROCESS SET INFO when the information is correct.
3. Define the data items for the first data set.
4. Repeat steps 2 and 3 for each data set.

Note

It is simplest to define master sets first, then detail sets. Then, when you want to enter a key item for a detail set, press LINK TO MASTER instead of defining or adding a new item.

5. Create the root file and information file.
6. Create the data set files.

The softkeys for defining a data base are discussed in the following sections.

Data Base Options

When you begin defining a data base, these softkeys are displayed:

```
--------------------------------------------------------------------------------------------
| DEFINE A |        | MODIFY A |      | |       |         |   HELP   |   EXIT    |
|DATA BASE|         |DATA  BASE|      | |       |         |          | SUBSYSTEM
--------------------------------------------------------------------------------------------
```

Defining a New Data Base

If you want to define a new data base, press the **DEFINE A DATA BASE** softkey. After this softkey is pressed, QUERY/45 asks you for the information about the data base: the data base name, description, volume label and passwords. Press PROCESS BASE INFO when the information is correct. You are then asked for the first set name.

Modifying an Existing Data Base Definition

The **MODIFY A DATA BASE** softkey enables you to modify an existing data base. You can either create a new data base from an existing data base definition or you can just change parts of the current definition.

If you want to change parts of the current data base definition, then keep using the data base, you must take some extra steps so that your data is not lost. (Modifying a data base modifies the root file which, in turn, would make the data inaccessible.) To preserve the data, follow these steps:

1. Unload the data out of the data base using the utility program DBUNLD.
2. Run QUERY, press DEFINE A DATA BASE to access the DEFINE subsystem, then press MODIFY A DATA BASE.
3. Make the desired changes.
4. Press CREATE ROOT FILE, then press the PURGE DATA BASE softkey to purge the old root file and data sets (which are empty).
5. Create the root file and the data sets.
6. Exit QUERY, then load the data back into the new data base using the utility program DBLOAD.

For explanations of DBUNLD and DBLOAD, refer to the IMAGE/45 Programming Manual.

After you specify which data base to restructure, DEFINE asks you to enter a maintenance word if one was used when the data base was created. A data base created with a maintenance word cannot be modified unless the same maintenance word is specified. When the maintenance word is specified, the Information file [1] is read. If any error is found, it indicates a bad root file or a fatal disc error, an error message is displayed and the program returns to the beginning of DEFINE. If the data base is opened successfully, the data base information is displayed and you begin editing the definition.

Returning to QUERY

Press the **EXIT SUBSYSTEM** softkey to return to the beginning of QUERY/45.

[1] If the information file cannot be read or is not present, all synonyms, codes, ranges, and special item types that were previously defined for the data base are not used.

Data Set Options

When you begin defining a data set, these softkeys are displayed:

DEFINE A DATA SET	ALTER & RESTORE	MODIFY & RENAME	DELETE DATA SET	CREATE ROOT FILE	SHOW BASE INFO	HELP	EDIT BASE INFO

Define a New Data Set

Press the **DEFINE A DATA SET** softkey to define a new data set. You need to enter the set name, description (optional), type, capacity, volume name, and password access. Then press PROCESS SET INFO. You are then asked to define the items for that set.

Changing the Definition of an Existing Data Set

If you want to change any part of an existing data set definition except the type of the set, the **ALTER & RESTORE** softkey is used.The altered data set definition is stored in place of the original set definition.

Create a Data Set from an Existing Definition

The **MODIFY & RENAME** softkey defines a new data set from an existing data set definition. The original data set definition is not changed with the MODIFY & RENAME function. The new set is the same type (automatic, manual or detail) as the selected set, and it contains the same data items, although none of the data set links are copied.

Deleting a Data Set Definition

The **DELETE DATA SET** softkey is used to remove an existing data set definition. The entire definition of the selected data set is deleted. After the data set is deleted, the remaining data sets are listed and you are asked to select another set to be deleted. Press the DATA SET OPTIONS softkey to cancel the delete option.

Creating the Data Base

When the data base definition is complete, press the **CREATE ROOT FILE** softkey. This procedure is discussed later in the chapter.

Show Data Base Information

When the **SHOW BASE INFO** softkey is pressed, the following softkeys are displayed:

GRAPHIC STRUCTURE	SCHEMA					HELP	DATA SET OPTIONS

Data Base Structure

The **GRAPHIC STRUCTURE** softkey is used to print the current structure of the data base on the internal printer. This diagram contains the defined data sets, the data items contained in each set and the links defined between the sets.

Schema Listing

The **SCHEMA** softkey is used to print a word description of the data base on the internal printer. The schema listing contains the current data sets, data items, synonyms, ranges, code values, and links.

Data Item Options

At the beginning of defining a data item, one or more of these softkeys are displayed. The softkeys displayed depends on if there are other items defined in the data base or that data set.

DEFINE A DATA ITEM	ALTER & RESTORE	MODIFY & RENAME	DELETE DATA ITEM	ADD A DATA ITEM	LINK TO[1] MASTER	HELP	DATA SET OPTIONS

When you have defined all the items for the set you are defining, press **DATA SET OPTIONS**. A data set must contain at least one data item, so if no data items were defined for a data set, QUERY/45 asks you to:

- edit the definition of the data set (**EDIT SET INFO** softkey)
- select the data item options (**DATA ITEM OPTIONS** softkey)
- delete the definition of the data set (**DATA SET OPTIONS** softkey)

Every manual master data set must have a key item. A list of the items contained in the set is displayed and you need to select the search item. Once the search item is selected, the data set options are redisplayed.

Defining a Data Item

To define a new data item, press the **DEFINE A DATA ITEM** softkey. Once the data item definition is complete, press the **PROCESS ITEM INFO** softkey.

The information specified for each data item is:

- data item name
- synonyms
- whether or not the item is required
- data item type
- range of the data item if the type is integer, long, short or date
- length of the data item in characters if the type is character or name type
- code values if the type is code
- a subitem count if the data item is a compound item

Changing the Definition of an Existing Data Item

To change the definition of a data item which has been previously defined for the current data set, press the **ALTER & RESTORE** softkey. If the selected data item is found in more than one data set or if it is a key item linked to other data sets, the type and the subitem count for this item cannot be changed. If you are altering a key item in a detail data set and the data item type is integer, short, long or date, the default range of the item is the range specified for the master data set key item. When you complete the changes, the new definition replaces the original definition of the data item. If you alter a data item which is also found in other data sets, those data items are also changed.

Create a Data Item from an Existing Data Item

If you want to create a new data item from an existing data item definition, press the **MODIFY & RENAME** softkey. The definition of the original data item is unchanged. The screen displays the definition of the selected data item with the data item name field and the synonym fields blank.

[1] This key is not displayed when you are defining a master set.

When you modify a data item which is code type, there must be enough room to store a maximum of 35 more codes in the code table (which contains the code values for the entire data base). If 35 code values cannot be stored in the code table, the maximum that can be stored is indicated by the number of fields displayed on the screen. For example, if 245 code values have already been defined for the data base and you want to MODIFY & RENAME a data item of the type code, a maximum of five code values can be defined for the specified data item. Therefore, when the data item information is displayed on the screen, only five fields are displayed.

If the new data item is too large to fit in the data set entry, it can be made smaller by pressing the **CHANGE ITEM TYPE** softkey. For example, a long data type can be changed to short. If the subitem count originally defined for the data item is too large to fit in the new data set entry, the subitem count for the new data item is automatically set to zero.

You cannot define or modify & rename a data item if:

- a data item has already been defined for the automatic master data set being defined.
- 127 data items have already been defined for a manual master or detail data set.
- 255 data items have already been defined for the data base.
- there is no room left in the data set entry.

Deleting a Data Item Definition

The **DELETE DATA ITEM** softkey is used to delete a data item from the current data set. You select the data item to be deleted from a list of the defined data items listed on the screen. The selected data item is deleted from the current data set along with its associated links. If the selected item is found only in the current data set, the data item is deleted from the data base. The DELETE DATA ITEM softkey always appears at the beginning of defining a data item except when no items are defined for the current data set.

Adding an Existing Data Item to a Data Set

Press the **ADD A DATA ITEM** softkey to add a previously defined data item to the data set being defined. The selected item is then included in the definition of the current data set if there is room in the data set. If there is not enough room for the selected item or if the selected item is already found in the current data set, an error message is displayed and the data item is not added. Press the DATA ITEM OPTIONS softkey to cancel the ADD A DATA ITEM command.

Linking a Detail Data Set to a Master Data Set

When you want to specify a path between the key items of a master and a detail data set, press the **LINK TO MASTER** softkey. This softkey is present only when the current data set is a detail set which has less than 16 paths defined and room for a link, and at least one master set has been defined. When the LINK TO MASTER softkey is pressed, a list of the currently defined master sets is displayed on the screen, and you are asked to select the master set to which the current detail set is to be linked. The screen also lists the master sets which already have 16 paths defined or have no room in the data set for another path definition. If none of the master data sets can be linked to, an error message is displayed and the LINK TO MASTER operation is not performed.

After the master data set is selected, you are asked to enter the name of the key item in the detail data set. The name displayed in the field is the same as the master data set key item name. You can change the item name as long as the new name is not the same as any previously defined item name or synonym in the data base or the new name may be same as an existing data item which has the same definition as the master set key item. Once you enter an item name, the new item is displayed unless:

- there is not enough room in the detail set for the new item.
- the data set already has 127 defined data items.
- 255 data items have already been defined for the entire data base.

An error message is displayed if the new data item cannot be entered into the current data set.

When the detail key item has been entered, the screen displays the definition of the selected master data set's key item. You can then change any of the synonyms, the required status, or the range or code values. The information that cannot be changed is the item type, string length, and subitem count.

Creating the Data Base

When you press the **CREATE ROOT FILE** softkey, QUERY/45 checks the data base definition to determine that the definition represents a valid data base. It checks to insure that:

- all automatic master data sets are linked to at least one detail data set.
- all manual master and detail data sets have read/write access assigned to at least one password.
- no data set requires more than 32 767 physical disc sectors (records).
- the ranges of master and detail set key items are compatible.
- the code values of master and detail set key items are compatible.
- the required status of master and detail set key items are compatible.
- there are no unreferenced passwords.

If these conditions are not met, press the **DATA SET OPTIONS** softkey to redisplay the data set options.

If it is valid and the volume name can be found, QUERY/45 attempts to create the root file and the information file. If the volume name cannot be found, you can insert the correct volume in the device and press the **VOLUME MOUNTED** softkey.

If a disc error occurs while creating the root file and information file, an error message is displayed and the data base definition screen is displayed. When the root file and information file are successfully created, you can:

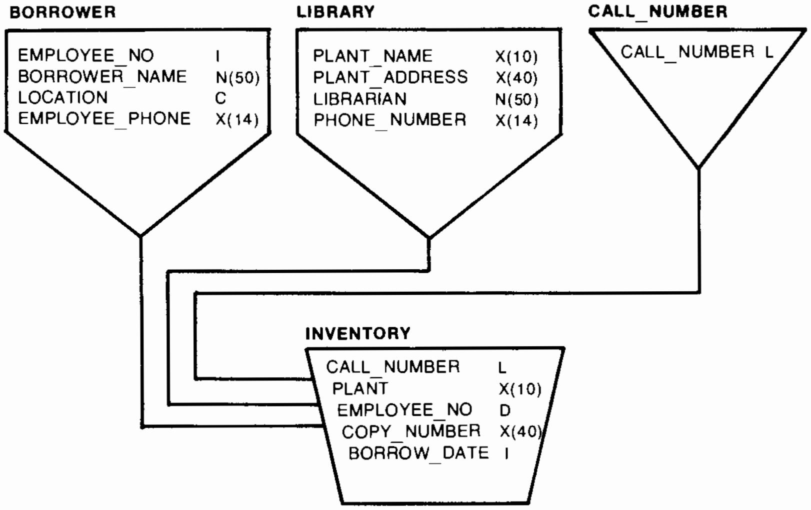
- create the data sets in the data base (**CREATE DATA SETS** softkey)
- define another data base (**DATA BASE OPTIONS** softkey)
- exit the DEFINE subsystem (**EXIT SUBSYSTEM** softkey)

When you press the **CREATE DATA SETS** softkey, you are asked to enter a maintenance word which is used to protect the root file and data set files. Purging, erasing, backing up, or modifying the data base cannot be done without specifying the maintenance word. If you specify a maintenance word and later forget it, the data base can never be purged, backed up, or modified.

After the data sets have been created, the data base options are displayed on the screen.

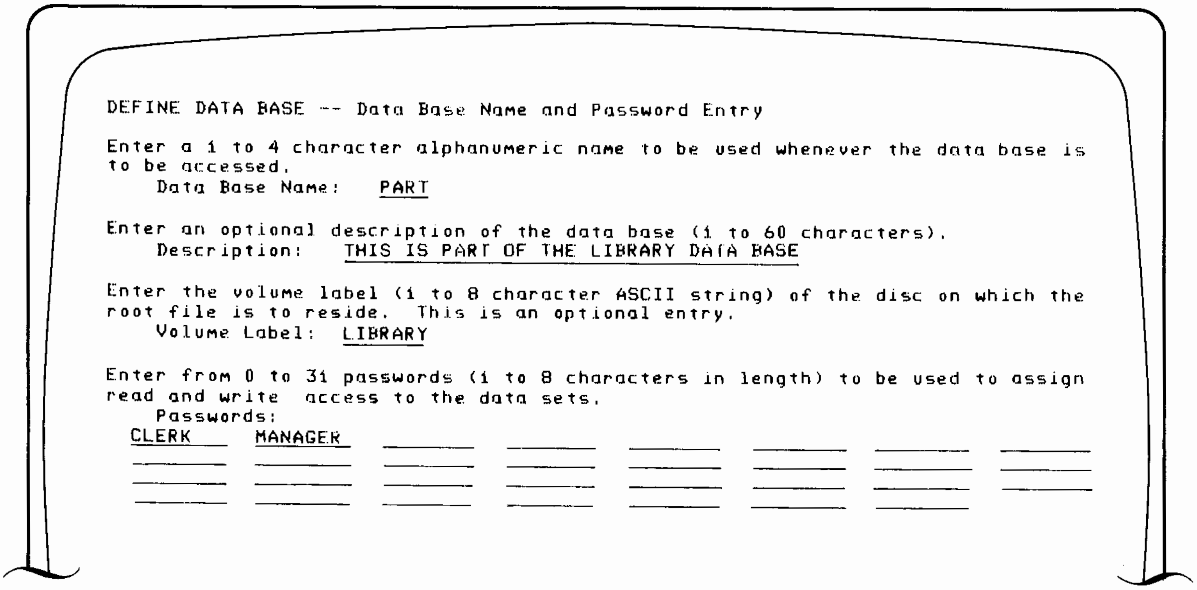
Example

The following example guides you through the definition of part of the Library data base. Three 9885 flexible discs are used; two discs contain QUERY/45 and have the volume label QUERY/45 and the other disc stores the new data base and has the label LIBRARY. This section also includes brief definitions (in shaded boxes) of important terms used while defining a data base. The diagram of the data base you are going to create is:



Data Base Information

1. For the data base name, type PART and press CONTINUE.
2. Fill out the rest of the screen to look like:



```
DEFINE DATA BASE -- Data Base Name and Password Entry

Enter a 1 to 4 character alphanumeric name to be used whenever the data base is
to be accessed.
    Data Base Name:   PART

Enter an optional description of the data base (1 to 60 characters).
    Description:   THIS IS PART OF THE LIBRARY DATA BASE

Enter the volume label (1 to 8 character ASCII string) of the disc on which the
root file is to reside.  This is an optional entry.
    Volume Label:  LIBRARY

Enter from 0 to 31 passwords (1 to 8 characters in length) to be used to assign
read and write  access to the data sets.
    Passwords:
  CLERK____  MANAGER__  _________  _________  _________  _________  _________  _________
  _________  _________  _________  _________  _________  _________  _________  _________
  _________  _________  _________  _________  _________  _________  _________  _________
  _________  _________  _________  _________  _________  _________  _________
```

Defining the CALL_NUMBER Data Set

3. For the first data set name, type CALL_NUMBER and press CONTINUE. The character between the two words is an underscore (_).
4. Type SET CONTAINING ALL THE CALL NUMBERS for the description and press CONTINUE. As with the data base description, the optional set description helps you identify the data that is stored in the data set.
5. The data set type is automatic master, so type A. This type is indicated by the triangle shape in the diagram.
6. For the maximum capacity, type 23. Remember, since this is a master data set, the capacity should be a prime number.
7. Press CONTINUE to enter the volume name of LIBRARY.
8. For passwords, type R (for READ access) for CLERK and keep the W (for READ/WRITE access) for MANAGER.
9. Press the PROCESS SET INFO softkey to record the information.

The information about the first data set has been entered; now the data items in this data set (CALL_NUMBER) need to be entered.

Defining the Data Items for CALL_NUMBER

10. Type CALL_NUMBER for the data item name.
11. You are now asked for the synonyms for the data item name, CALL_NUMBER.

Synonyms are names that are used interchangeably with item names. For example, WRITER can be used instead of AUTHOR. Synonyms are useful when several people use a data base and the exact item names might not be remembered, or to abbreviate a long item name to just a few characters.

12. Press CONTINUE to leave all the synonyms fields blank so that no synonyms are defined for the item name CALL_NUMBER.
13. Type Y to indicate Yes for the required item field.

When an item is **required**, it cannot be left blank (or null) when a new entry is entered into a data base.

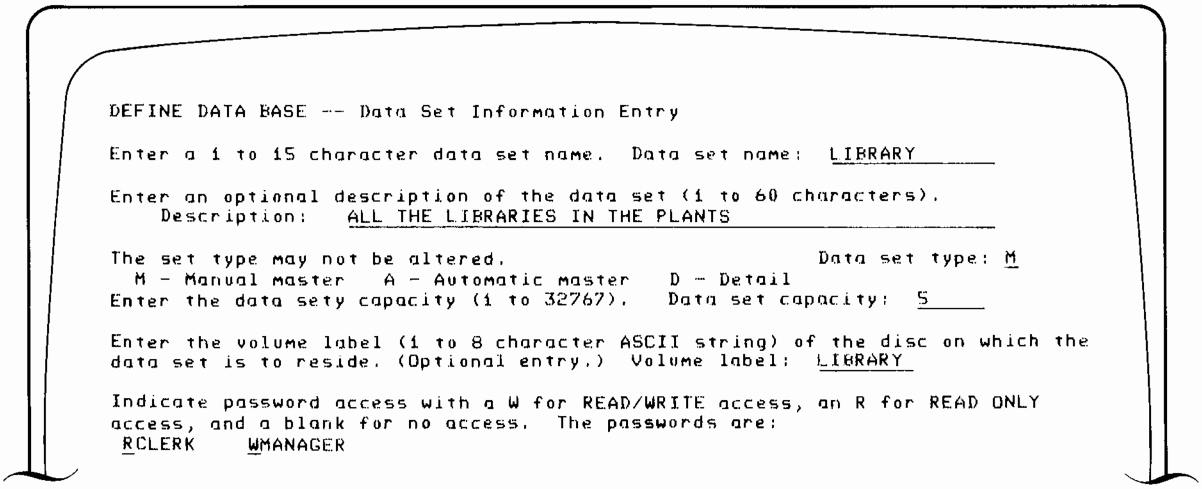
14. Indicate that this data item is of type LONG by typing an L.
15. The maximum range of a long real number is displayed (with lower and upper bounds). Call numbers are positive, so change the lower bound to 0.

The cursor then returns to the first entry on this screen to enable you to edit any values. If you don't want to edit anything, press the PROCESS ITEM INFO softkey.

Since this is an automatic master set, only one data item can be defined. DEFINE then gives you the choice of changing the definition of the item or deleting the item. You want to leave the definition as it is, so now you are ready to define another data set.

Defining the LIBRARY Data Set

16. Press the DATA SET OPTIONS softkey.
17. Now press the DEFINE A DATA SET softkey.
18. Enter the following information for the LIBRARY manual master data set. When the screen is completed, press the PROCESS SET INFO softkey.



```
DEFINE DATA BASE -- Data Set Information Entry

Enter a 1 to 15 character data set name.  Data set name:  LIBRARY

Enter an optional description of the data set (1 to 60 characters).
    Description:   ALL THE LIBRARIES IN THE PLANTS

The set type may not be altered.                         Data set type:  M
  M - Manual master    A - Automatic master    D - Detail
Enter the data sety capacity (1 to 32767).    Data set capacity:  5

Enter the volume label (1 to 8 character ASCII string) of the disc on which the
data set is to reside. (Optional entry.)  Volume label:  LIBRARY

Indicate password access with a W for READ/WRITE access, an R for READ ONLY
access, and a blank for no access.  The passwords are:
 RCLERK     WMANAGER
```

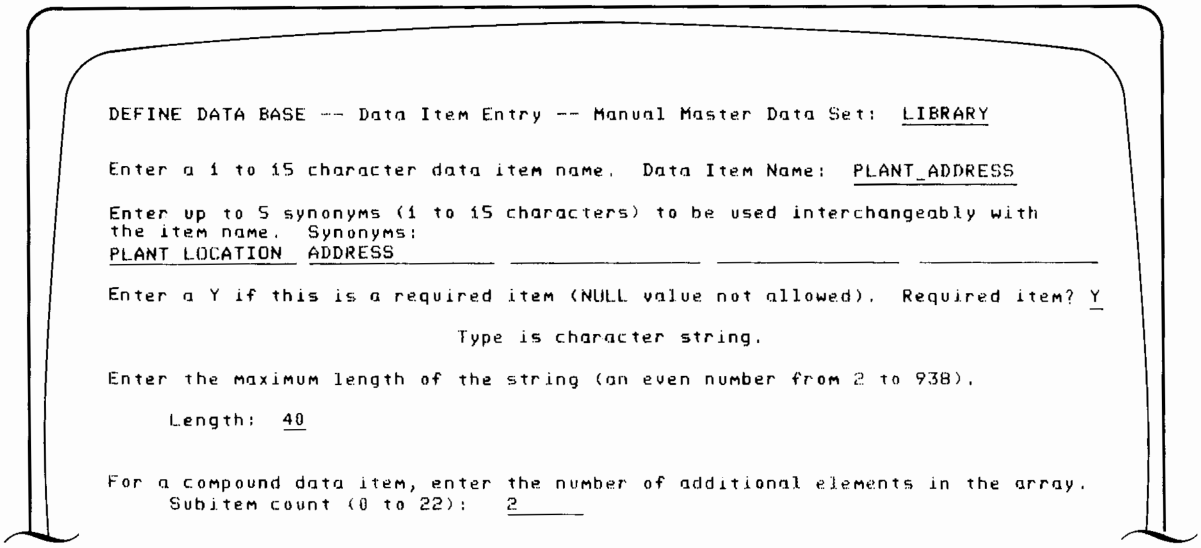
Defining the Data Items for LIBRARY

19. Now press the DEFINE A DATA ITEM softkey to define the data items in the LIBRARY set.
20. Type PLANT_NAME for the item name.
21. For synonyms, type LIBRARY_BRANCH and press CONTINUE, and type PLANT_LIBRARY.
22. Press CONTINUE twice to move past the synonym section.
23. Type Y to indicate that it is a required item. A key item should always be a required item.
24. Type X to specify that PLANT_NAME is character-type (string) data item.
25. For the length, type 10. The length of each item is indicated on the data base diagram shown previously.

The **length of a character string has to be an even number**. If you type an odd number, QUERY/45 converts it to the next highest number.

26. Now press the PROCESS ITEM INFO softkey, leaving the subitem count at 0.

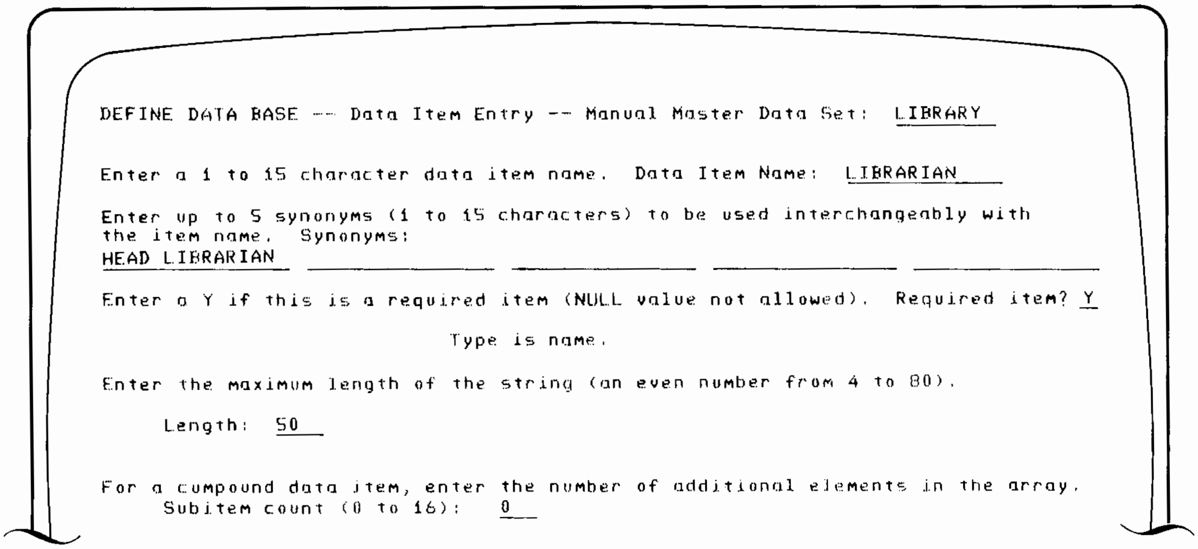
The screen for the next item should be filled out as follows. When it is completed, press the PROCESS ITEM INFO softkey.



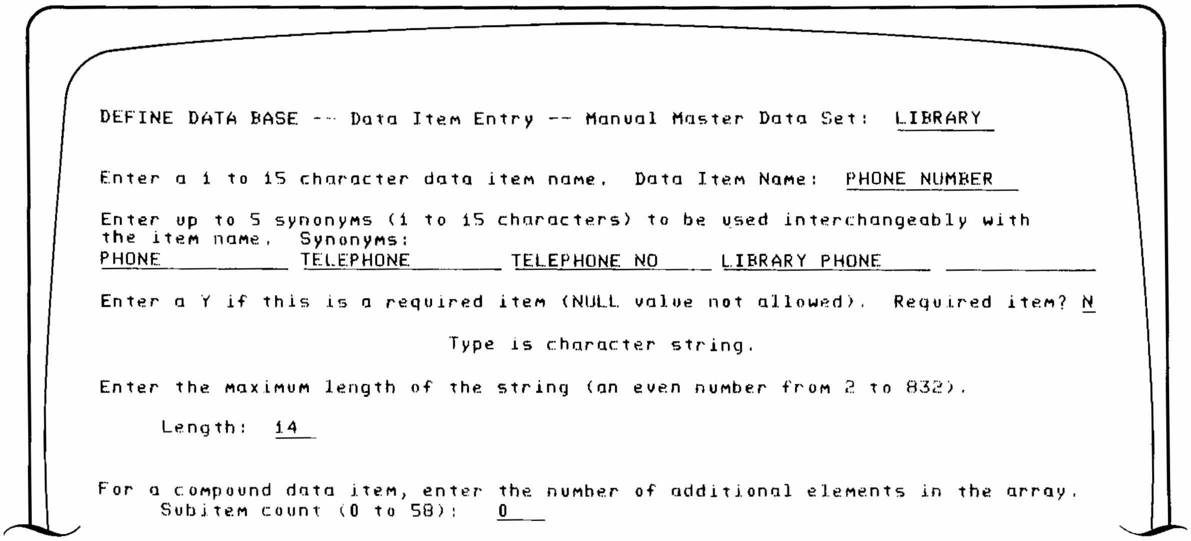
```
DEFINE DATA BASE -- Data Item Entry -- Manual Master Data Set:  LIBRARY

Enter a 1 to 15 character data item name.  Data Item Name:  PLANT_ADDRESS

Enter up to 5 synonyms (1 to 15 characters) to be used interchangeably with
the item name.  Synonyms:
PLANT_LOCATION  ADDRESS

Enter a Y if this is a required item (NULL value not allowed).  Required item? Y

                              Type is character string.

Enter the maximum length of the string (an even number from 2 to 938).

     Length:  40

For a compound data item, enter the number of additional elements in the array.
     Subitem count (0 to 22):    2
```

Note that the subitem count is 2. This indicates the PLANT_ADDRESS is a **compound item** made up of three 40-character strings. A compound data item is made up of a number of parts or **subitems**. The subitem count entered here indicates the number of subitems **in addition to** the original, simple data item. The first part contains the street address, the second the city and state and the third contains the zip code.

For the next two items, fill in the screens as shown, pressing the PROCESS ITEM INFO softkey after each one.



```
DEFINE DATA BASE -- Data Item Entry -- Manual Master Data Set:  LIBRARY

Enter a 1 to 15 character data item name.  Data Item Name:  LIBRARIAN

Enter up to 5 synonyms (1 to 15 characters) to be used interchangeably with
the item name.  Synonyms:
HEAD_LIBRARIAN

Enter a Y if this is a required item (NULL value not allowed).  Required item? Y

                              Type is name.

Enter the maximum length of the string (an even number from 4 to 80).

     Length:  50

For a compound data item, enter the number of additional elements in the array.
     Subitem count (0 to 16):    0
```



```
DEFINE DATA BASE -- Data Item Entry -- Manual Master Data Set:  LIBRARY

Enter a 1 to 15 character data item name.  Data Item Name:  PHONE NUMBER

Enter up to 5 synonyms (1 to 15 characters) to be used interchangeably with
the item name.  Synonyms:
PHONE          TELEPHONE          TELEPHONE NO     LIBRARY PHONE

Enter a Y if this is a required item (NULL value not allowed).  Required item? N

                          Type is character string.

Enter the maximum length of the string (an even number from 2 to 832).

     Length:  14

For a compound data item, enter the number of additional elements in the array.
     Subitem count (0 to 58):   0
```

Now that you have defined the data items, press the DATA SET OPTIONS softkey. At this point, DEFINE asks you for the key or search item.

27. Type 1 to indicate PLANT_NAME is the key item. 1 is the item number of PLANT_NAME.

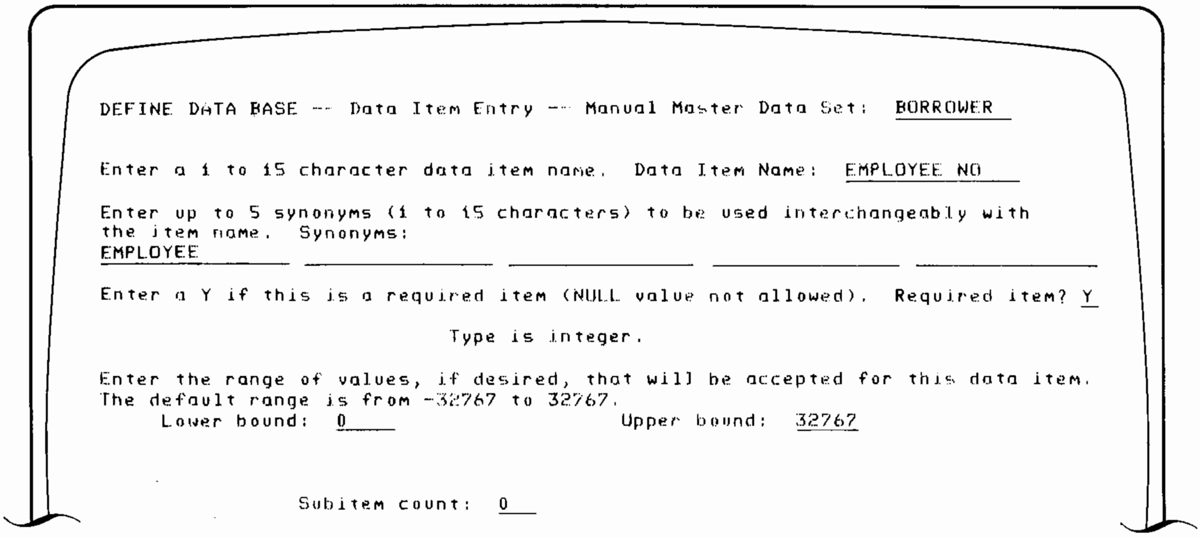
You are now ready to define another data set, so press the DEFINE A DATA SET softkey.

Defining the BORROWER Data Set

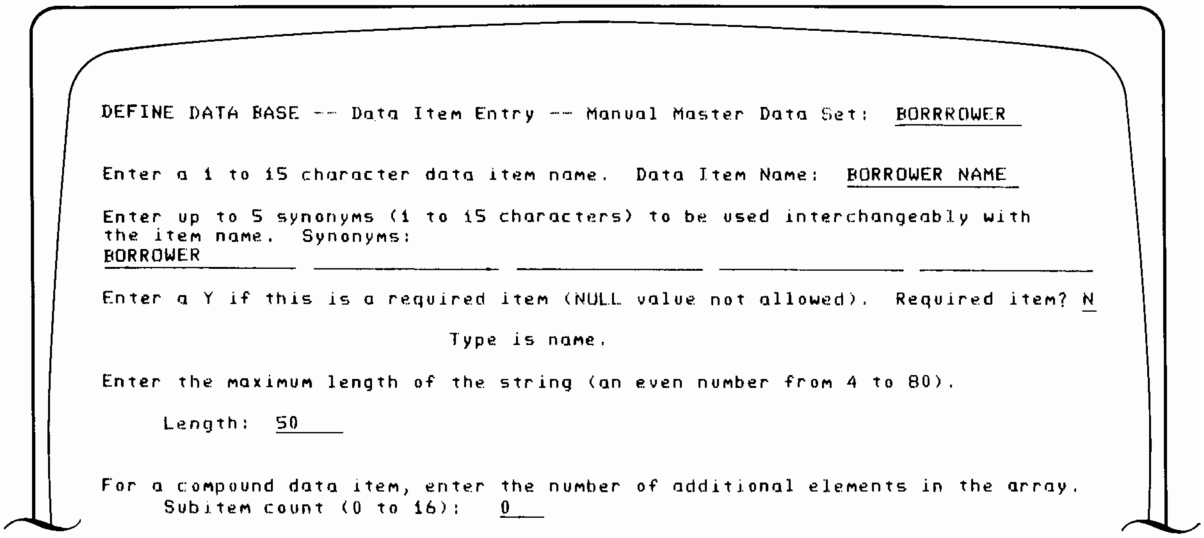
28. Type BORROWER for the name of the data set.
29. When asked for the description, type RECORD OF THE EMPLOYEES WHO BORROW THE BOOKS.
30. This is a manual master data set, so type M for the set type.
31. Type 37 for the capacity.
32. Press CONTINUE to leave the volume name as LIBRARY; change the CLERK access to READ ONLY access by typing R before CLERK and leave the W before MANAGER.
33. Press the PROCESS SET INFO softkey.

Defining the Data Items for BORROWER

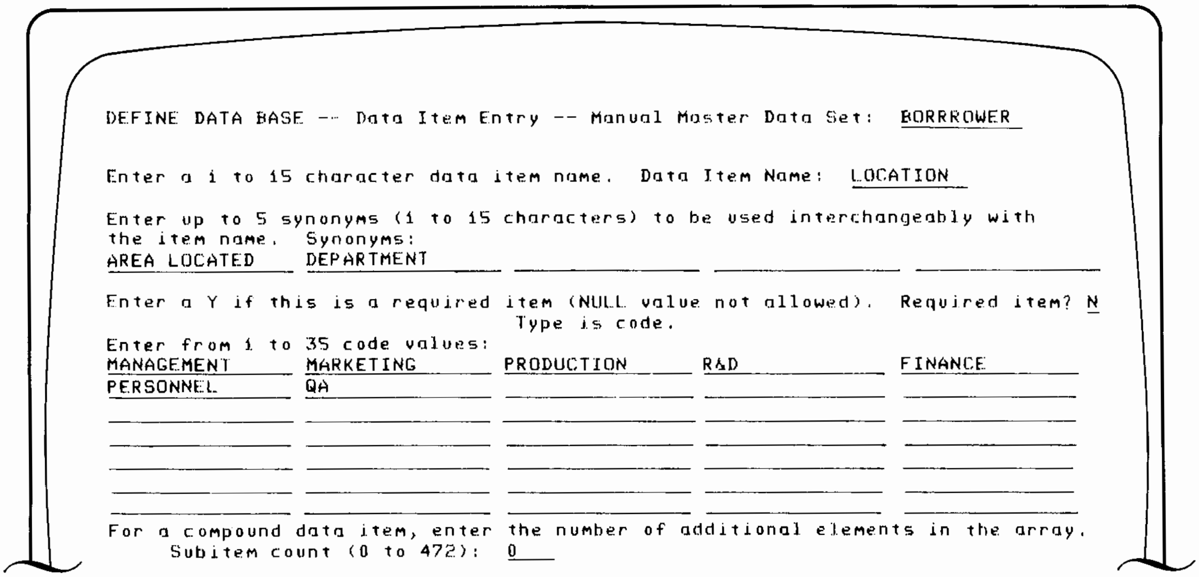
You are now ready to define the items in the BORROWER data set. Complete the screens as shown:



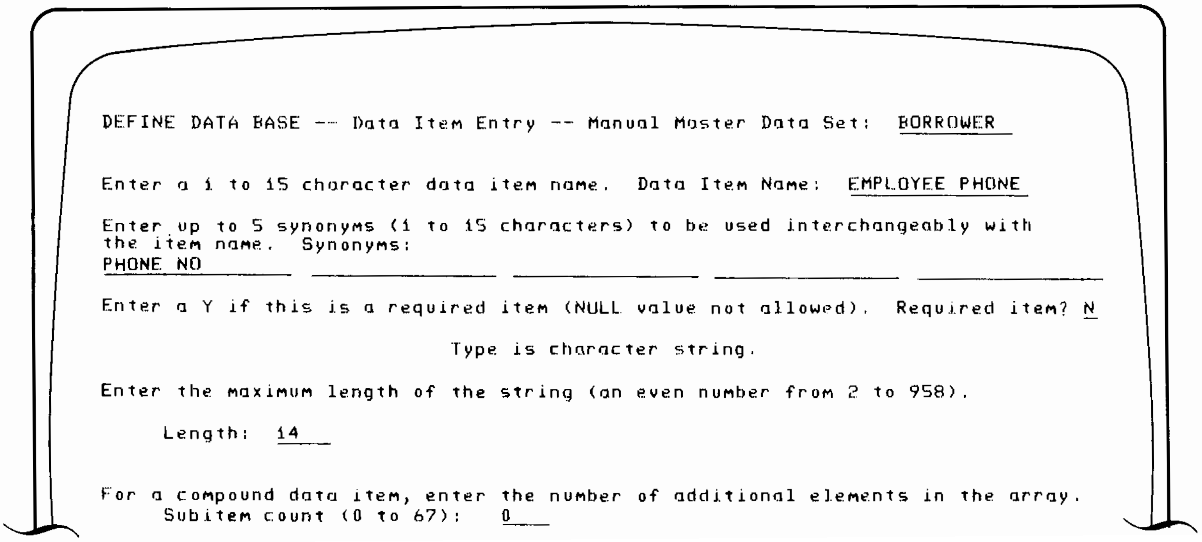
```
DEFINE DATA BASE -- Data Item Entry -- Manual Master Data Set: BORROWER

Enter a 1 to 15 character data item name.  Data Item Name: EMPLOYEE NO

Enter up to 5 synonyms (1 to 15 characters) to be used interchangeably with
the item name.  Synonyms:
EMPLOYEE

Enter a Y if this is a required item (NULL value not allowed).  Required item? Y

                       Type is integer.

Enter the range of values, if desired, that will be accepted for this data item.
The default range is from -32767 to 32767.
    Lower bound:  0                       Upper bound:  32767


                Subitem count:  0
```



```
DEFINE DATA BASE -- Data Item Entry -- Manual Master Data Set: BORRROWER

Enter a 1 to 15 character data item name.  Data Item Name: BORROWER NAME

Enter up to 5 synonyms (1 to 15 characters) to be used interchangeably with
the item name.  Synonyms:
BORROWER

Enter a Y if this is a required item (NULL value not allowed).  Required item? N

                       Type is name.

Enter the maximum length of the string (an even number from 4 to 80).

    Length:  50

For a compound data item, enter the number of additional elements in the array.
    Subitem count (0 to 16):  0
```



```
DEFINE DATA BASE -- Data Item Entry -- Manual Master Data Set:  BORRROWER

Enter a 1 to 15 character data item name.  Data Item Name:  LOCATION

Enter up to 5 synonyms (1 to 15 characters) to be used interchangeably with
the item name.  Synonyms:
AREA LOCATED    DEPARTMENT

Enter a Y if this is a required item (NULL value not allowed).  Required item? N
                                   Type is code.
Enter from 1 to 35 code values:
MANAGEMENT      MARKETING       PRODUCTION      R&D             FINANCE
PERSONNEL       QA

For a compound data item, enter the number of additional elements in the array.
     Subitem count (0 to 472):  0
```



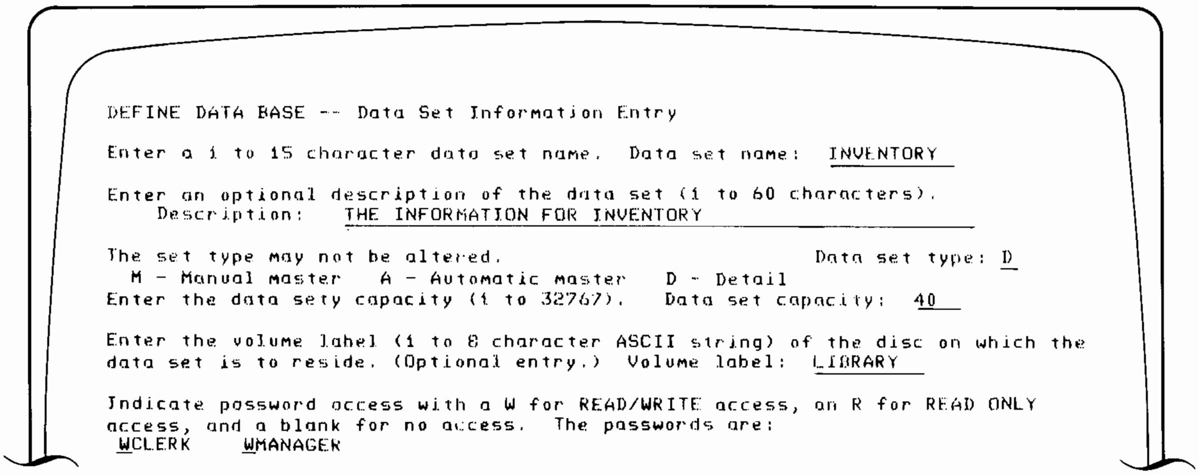
```
DEFINE DATA BASE -- Data Item Entry -- Manual Master Data Set:  BORROWER

Enter a 1 to 15 character data item name.  Data Item Name:  EMPLOYEE PHONE

Enter up to 5 synonyms (1 to 15 characters) to be used interchangeably with
the item name.  Synonyms:
PHONE NO

Enter a Y if this is a required item (NULL value not allowed).  Required item? N

                              Type is character string.

Enter the maximum length of the string (an even number from 2 to 958).

     Length:  14

For a compound data item, enter the number of additional elements in the array.
     Subitem count (0 to 67):  0
```

34. After all the data items are entered, press the DATA SET OPTIONS softkey.
35. Type 1 to specify that the key item for the BORROWER data set is EMPLOYEE_NO.

Defining the INVENTORY Data Set

36. Press DEFINE DATA SET to define the last data set.
37. Type INVENTORY for the name of the data set.

Fill in the rest of the screen as shown:



```
DEFINE DATA BASE -- Data Set Information Entry

Enter a 1 to 15 character data set name.  Data set name:  INVENTORY

Enter an optional description of the data set (1 to 60 characters).
    Description:   THE INFORMATION FOR INVENTORY

The set type may not be altered.                          Data set type: D
  M - Manual master    A - Automatic master    D - Detail
Enter the data sety capacity (1 to 32767).    Data set capacity:  40

Enter the volume label (1 to 8 character ASCII string) of the disc on which the
data set is to reside. (Optional entry.)  Volume label:  LIBRARY

Indicate password access with a W for READ/WRITE access, on R for READ ONLY
access, and a blank for no access.  The passwords are:
 WCLERK    WMANAGER
```

38. Press PROCESS SET INFO softkey to enter the data set information.

Defining the Data Items for INVENTORY

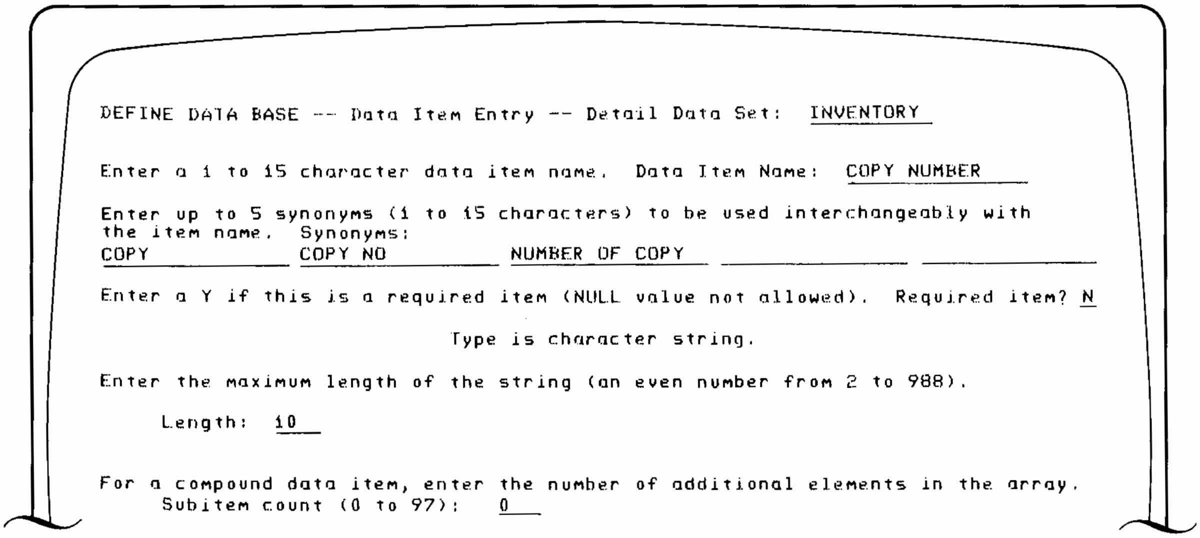
Since this is a detail set and the master sets that relate to INVENTORY have already been defined, the process of defining the data items is slightly different. The first step is to indicate the key items and related master sets.

39. Press the LINK TO MASTER softkey.

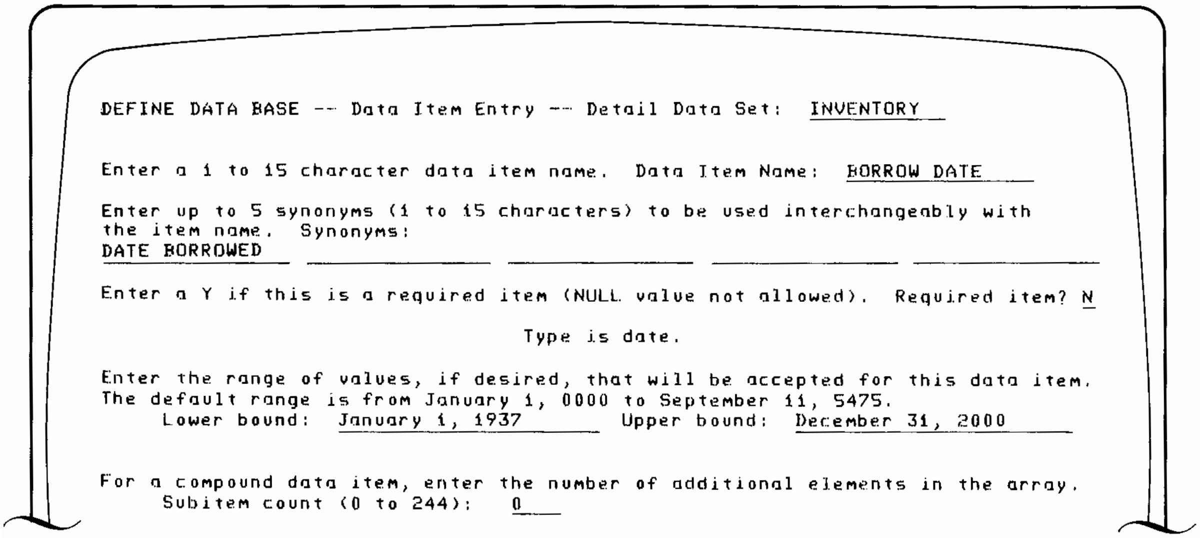
All of the data set names that you have already defined are listed on the screen.

40. Type 1 to indicate that you want INVENTORY linked to the CALL_NUMBER set; the CALL_NUMBER item is then placed into INVENTORY.
41. The item name CALL_NUMBER is displayed on the screen. This enables you to change the name of the key item in the detail set so that it differs from the name of the key item in the master set.
42. Press CONTINUE to indicate that you want the item name to be the same in both the detail data set and the master data set.
43. When the rest of the information about the CALL_NUMBER item is displayed, press the PROCESS ITEM INFO softkey without changing any information.

44. Now press the DATA ITEM OPTIONS softkey to define other data items contained in INVENTORY.
45. Then press the DEFINE A DATA ITEM softkey.
46. Define the next two items as follows, pressing the PROCESS ITEM INFO softkey after each screen is completed.



```
DEFINE DATA BASE -- Data Item Entry -- Detail Data Set:  INVENTORY

Enter a 1 to 15 character data item name.  Data Item Name:  COPY NUMBER

Enter up to 5 synonyms (1 to 15 characters) to be used interchangeably with
the item name.  Synonyms:
COPY                COPY NO               NUMBER OF COPY

Enter a Y if this is a required item (NULL value not allowed).  Required item? N

                            Type is character string.

Enter the maximum length of the string (an even number from 2 to 988).

     Length:  10

For a compound data item, enter the number of additional elements in the array.
     Subitem count (0 to 97):   0
```



```
DEFINE DATA BASE -- Data Item Entry -- Detail Data Set:  INVENTORY

Enter a 1 to 15 character data item name.  Data Item Name:  BORROW DATE

Enter up to 5 synonyms (1 to 15 characters) to be used interchangeably with
the item name.  Synonyms:
DATE BORROWED

Enter a Y if this is a required item (NULL value not allowed).  Required item? N

                               Type is date.

Enter the range of values, if desired, that will be accepted for this data item.
The default range is from January 1, 0000 to September 11, 5475.
     Lower bound:  January 1, 1937        Upper bound:  December 31, 2000


For a compound data item, enter the number of additional elements in the array.
     Subitem count (0 to 244):   0
```

47. Now press the DATA ITEM OPTIONS since you need to define the two remaining key items.
48. Press the LINK TO MASTER softkey.
49. Type 2 to indicate that you want INVENTORY linked to the LIBRARY data set.
50. When the data item PLANT_NAME is displayed, change the name of the item to PLANT.
51. Press the PROCESS ITEM INFO softkey to enter this item into the INVENTORY data set without changing any of the remaining information.
52. Now type 3 to link INVENTORY to the BORROWER data set.
53. Press CONTINUE when the item name EMPLOYEE_NO is displayed.
54. Then press the PROCESS ITEM INFO softkey after the rest of the information is displayed. This enters the item EMPLOYEE_NO into the INVENTORY data set without changing any of the remaining information.

Creating the Data Base

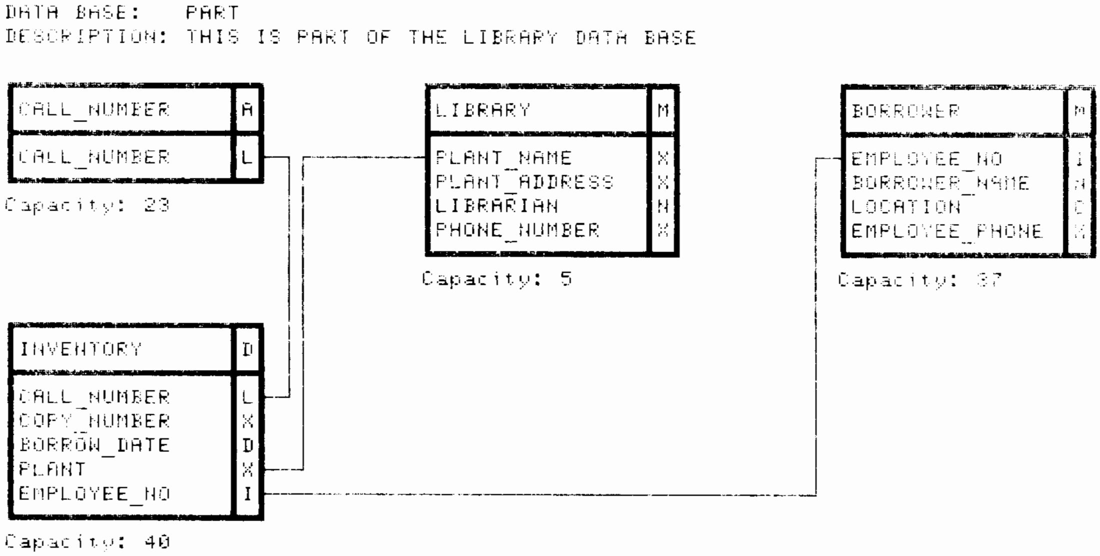
The PART data base is now ready to be created. This involves processing the set and item definitions, then creating the root file, information file and data set files.

55. Press the DATA ITEM OPTIONS softkey.
56. Press the DATA SET OPTIONS softkey.
57. Press the CREATE ROOT FILE softkey.
58. When the root file and information file have been created, press the CREATE DATA SETS softkey.
59. For the maintenance word, just press CONTINUE.

A **maintenance word** is used to protect the data base from purging or restructuring by unauthorized people. It is similar to a password, except that passwords enable only certain people to update the the data in a set; where as a maintenance word helps prevent purging or restructuring of the whole data base. If the field is left blank, no maintenance word exists so anyone can purge or restructure the data base.

60. Press EXIT SUBSYSTEM to get out of the DEFINE subsystem.

Now your data base is ready for you to add data. If you were to get a graphic structure of the PART data base, it would look like this:



Chapter 9

Linking Other Subprograms to QUERY/45

Run User Program is a feature of QUERY/45 which enables you to write, load and execute one or more subprograms, using them as extensions to the main QUERY subsystem. The subprograms can perform various operations, such as mathematical calculations, report generation, etc. They have access to all COMMON variables defined by QUERY/45 and may call various utility subprograms, which are described later in this chapter. These extensions should be written by experienced programmers. Anyone can use the extension once it is written.

This chapter contains the information necessary for a programmer to write subprograms for use as extensions to QUERY/45. The following information is included:

- How Run User Program Works
- Restrictions on the "Extend" Subprogram
- Using the Search Result
- Using COMMON Variables
- Descriptions of the Utility Subprograms
- An Example

How Run User Program Works

The Run User Program feature can be accessed either immediately after opening a data base of after a search has been performed. It is accessed by pressing the RUN USER PROGRAM softkey. When this is done, you are asked to enter the file name and the volume label of the file which contains the subprograms you have written.

When Run User Program is accessed, QUERY/45 deletes all subprograms from memory except the utility subprograms described later. This maximizes the amount of memory available for your subprograms and variables. The exact amount of memory available depends on the size of the data base currently open and the configuration of your System 45. As an estimate, a 9845B with 187K bytes of memory has approximately 100K bytes of memory available for extension subprograms.

The Extend Subprogram

The first subprogram in the file which you load must be called "Extend". QUERY/45 begins running your extension subprograms by executing CALL "Extend" followed by the proper parameters. These parameters are described later in this chapter. The remainder of the subprograms you want to use should follow the "Extend" subprogram. "Extend" can call them as necessary.

Here is an example of how a file of extension subprograms is structured:

File Name: Report

```
SUB Extend ...
SUB Compute
SUB Print
```

Restrictions on the "Extend" Subprogram

The file loaded by Run User Program must contain a subprogram named "Extend", which in turn can call the other subprograms. The first few lines of this subprogram must appear as follows:

```
SUB Extend(INTEGER Item(*),Syn$(*),Code$(*),REAL Range(*))
OPTION BASE 1
COM INTEGER Err,Sts(*),Switch(*),Storage$,Seqvol$,Key_buf$
COM INTEGER Key,Keysw,Printer(*),Device(*),Dim(*),Base$(*)
COM INTEGER Token,Lineptr1,Lineptr2,Command$,Token$,Recall$
COM INTEGER Result(*),Thread(*),Screen$,Vars(*),Buf1$,Buf2$
ON ERROR CALL Global_error
DISP
```

The ON ERROR statement stops the program for any error from 81 – 99 (mass storage errors). Error 80 sets the variable Sts(1) to 80. Error 2 displays an "insufficient memory" comment and sets the variable Err to – 2. All other errors cause an error message to be displayed. In this case, you may press CONTINUE, which causes Err to be set to – 1; the program may or may not recover from the error.

The DISP statement clears the "Loading Extension" message that is displayed when Run User Program is begun.

The calling parameters and the COMMON variables are explained in later sections of this chapter. The parameters should not be altered, and most of the COMMON variables should be preserved. This is described in further detail in the next two sections.

No binary routines are allowed in the loaded subprograms, even if they are not referenced. If any binary routines are present, Error 207 occurs when the file is loaded.

Programming Hint

At first it may be difficult to debug the extensions and match parameters properly. The TRACE PAUSE statement can be useful in debugging extensions.

Note

Some of the important features of QUERY/45 include the use of softkeys and the use of "fields" to enter information. Use of the INPUT statement is discouraged in favor of the resident screen-input utility routines. Additionally, the subprograms you write should NOT execute an ON KBD or OFF KBD statement.

Using the Search Result

There are several variables in COMMON which pertain to the search result. They are:

Variable	Meaning
Result(7560,1)	the in-memory portion of the search result.
Dim(5)	the number of entries in the search result.
Dim(6)	the number of sets in the current thread (1 to 10)
Thread(10,2)	the first dimension contains the set numbers of the sets in the thread. The second dimension contains path information and should not be referenced by extension subprograms.
Vars(10)	an index into overflow files, used by Get_rec.

There are three basic types of searches which affect the contents of these variables.

Case 1:	FIND ALL for a single set.
Case 2:	FIND<SEARCH EXPRESSION>for a single set.
Case 3:	FIND ALL|<SEARCH EXPRESSION>for a thread.

Case 1

The first case is the simplest. A DBINFO statement is executed to determine the number of entries in the specified set. Dim(5) is set to the negative value of that number. This is the only time that Dim(5) is negative. A negative number indicates that the Result(*) array is invalid and that serial reads must be performed to obtain entries from the current search result. Dim(6) is set to 1, indicating that only one set exists in the thread. Thread(1,1) is set to the set number.

Case 2

The second case causes a search to be executed; the Result array is filled with the record numbers of the entries which meet the search criteria. Dim(6) and Thread(1,1) are set to 1 and the actual set number. Dim(5) is positive and indicates the number of entries in the search result. The Result(*) array is dimensioned to (7560,1). The first entry in the search result is in Result(1,1). The second entry is in Result(2,1), and so on, up to Result(7560,1).

If Dim(5) is larger than 7560, the search result overflows to disc files. Each file contains another 7560 record numbers. The overflow files are named "INOV1", "INOV2"... and are protected with the word "SEQUOIA". These files are created to hold one integer per record with the operation "CREATE FILE$,7560,4". They are always created on the data base root file volume.

The following example illustrates this type of search. Assume that the search was to FIND SUBJECT= PHOTOGRAPHY in a library data base and that only 3 entries were found. The variables are set up as follows (the set number and record numbers are arbitrary):

Dim(5)=3
Dim(6)=1
Thread(1,1)=2
Result(1,1)=5
Result(2,1)=57
Result(3,1)=3021

Case 3

The third case is similar to the second, except that several sets are simultaneously involved in the search process. Assume that a thread was set up from BOOK to CALL_NUMBER to INVENTORY in a library data base. After performing a search, only four entries qualify for the search result. The variables are set up as follows:

Dim(5)=4		
Dim(6)=3		
Thread(1,1)=2	Thread(2,1)=5	Thread(3,1)=7
Result(1,1)=10	Result(1,2)=10	Result(1,3)=4
Result(2,1)=10	Result(2,2)=10	Result(2,3)=5
Result(3,1)=27	Result(3,2)=5	Result(3,3)=98
Result(4,1)=45	Result(4,2)=18	Result(4,3)=11

Note that the RESULT array has 3 in the second subscript. It has been redimensioned to Result(2520,3). The second subscript corresponds to the number of sets in the thread. The first subscript is equal to 7560 divided by that number. In the worst case (a 10-set thread), only 756 entries are in the "in-memory" portion of the search result. Overflow files contain the remaining entries, if any. Again, these overflow files contain only 7560 integers, one per record. The program must take these in groups of 10 at a time, however. In a sense, overflow files can be viewed as "pages" of the search result. The first page is always in memory. Subsequent pages are on a disc.

It is recommended that extension subprograms use the resident utility subprogram Get_rec to read entries from the search result. A subprogram may use DBGETs, but Get_rec provides automatic selection of serial or directed mode, handles status errors, requests the user to mount a needed volume if it is not on-line, concatenates records from several DBGETs in the case of a thread, and reads overflow files to get the record numbers when necessary. The variable Vars(10) must originally be set to 0 to indicate that the search result starts in memory (page "0"). Get_rec increments Vars(10) as necessary as it proceeds from one overflow file to the next in its traversal of the search result.

Using COMMON Variables

Extensions to QUERY/45 have all of the following variables available in COMMON. The most important of these are Dim, Thread and Result, which have already been described. Other variables that may prove useful are those used by the Scan subprogram, the Sts(*) array, and the Key and Keysw variables. Most of the other variables are used internally by QUERY/45 and are of no use to extension subprograms.

Extensions must preserve the contents of COMMON. If any of the following variables are changed, they must be restored before returning to QUERY/45: Switch(*), Storage$, Seqvol$, Device(*), Dim(*), Base$(*), Recall$, Result(*), Thread(*), and Vars(*) (except for Vars(10)).

Advanced applications, such as sorting the search result, must guarantee that all variables governing the search result are consistent before returning to QUERY/45.

The following list describes the variables in COMMON:

Variable	Meaning
Err	a flag indicating a locally detected error.
Sts(10)	status array for DBOPEN, DBINFO, DBGET, etc.
Switch(10)	various system switches indicate the specified condition when set to 1: Switch(1) indicates re-entry information to MAIN. Switch(2) indicates formal command mode if true. Switch(3) is the SELECT FOR depth counter. Switch(4) indicates UPDATE called from SEARCH. Switch(5) indicates non-existence of INFORMATION file Switch(6) indicates typewriter mode to READFIELD. Switch(7) indicates the type of CRT on the 9845. Switch(8) indicates AUTOCLEAR to UPDATE. Switch(9) indicates LINEAR or COLUMNAR to LIST. Switch(10) counts the number of recursions in SEARCH.
Storage$	configured mass storage devices for VOLUME DEVICES ARE
Seqvol$	the name of the volume containing QUERY/45 programs.
Key_buf$	the KBD$ buffer used by READFIELD.
Key	the numeric value of the globally detected ON KBD key set automatically when Keysw = 1 or −1 and an appropriate key is pressed.
Keysw	−2: ignore all keys and beep −1: throw away all keys except k7 and k15 (the ABORT softkey). The variable Key can only be 0 or 8. This is shown in lines 1240-1280 of the sample program at the end of this chapter. 0: don't even read KBD$. 1: read KBD$, and return one of the following in Key (Note: if Key is already set, subsequent reads of KBD$ are thrown away and cause a beep. Keys not shown below are also thrown away.) 1-8:Softkeys 9:STOP 10:CONTINUE 11:UP-ARROW 12:DOWN-ARROW

Printer(6)	Parameters for the current output device: Printer(1) = select code, Printer(4) = page length, Printer(2) = HP-IB address, Printer(5) = line number, Printer(3) = print width, Printer(6) = page number.
Device(2,4)	Parameters for optional printer and other configuration information. Device(1,1)...(1,4) contain the same information) as in Printer(1...4) above. The second dimension contains the following information: Device(2,1) = 1 for perforated thermal printer paper, 0 for continuous thermal printer paper. Device(2,2) = 0 for STANDARD lexical order. 1 for FRENCH 2 for SPANISH 3 for SWEDISH 4 for GERMAN 5 for KATAKANA Device(2,3) = 1 for M-D-Y date interpretation 5 for D-M-Y date interpretation Device(2,4) = 0 for CHECKREAD OFF = 1 for CHECKREAD ON
Dim(10)	dimensions of various dynamic arrays: Dim(1) = size of ITEM table. Dim(2) = size of SYN table. Dim(3) = size of CODE table. Dim(4) = size of RANGE table. Dim(5) = number of entries in search result. (this is negative for FIND ALL) Dim(6) = number of sets in thread. Dim(7) = number of entries in SORT$ array. Dim(8) = length of SORT$ array. Dim(9) = total number of sets in data base. Dim(10) = number of accessible sets in data base.
Base$(3)	Base$(1) = the name of the currently open data base. Base$(2) = the password used to open the base. Base$(3) = the volume containing the root file.
Token	token number found by the Scan subprogram.
Lineptr1	points to beginning of token in Command$.
Lineptr2	points to end of token in Command$.
Command$	the command line being parsed by the Scan subprogram.
Token$	the token found by the Scan subprogram.
Recall$	the previous formal command, which may be recalled.
Result	the search result. Refer to the previous description.
Thread	the current set or thread being searched.
Screen$	an array for dumping screen contents to the printer. SEARCH subsystem builds the DBGET program in this string variable.
Vars(11)	various reserved variables: Vars(1) = subsystem number currently loaded Vars(2) = length of Buffer$ needed by Get_rec to perform DBGETs to concatenate all sets in the thread. Vars(3) = Number of updates since last backup. Vars(4) = used by SEARCH as a thread index. Vars(5) = used by SEARCH as pointer into Screen$ as a program is built.

Vars(6) = used by SEARCH as pointer into Screen$ for beginning of program.
Vars(7) = number of keys to sort/optimization switch
Vars(8) = predefined thread number being searched.
Vars(9) = number of overflow files created.
Vars(10) = overflow file index used by Get_rec.
Vars(11) = used by SEARCH for optimizing program.
Bf1$[256] a large primary buffer for DBINFOs.
Bf2$[100] a small secondary buffer for DBINFOs.

Parameters of the "Extend" Subprogram

There are four parameters passed by reference to the "Extend" subprogram. They are the four dynamically dimensioned arrays created when a data base is opened: the item table, the synonym table, the code table, and the range table. The table sizes are declared in the Dim(*) variable which is in COMMON.

While an extension subprogram might not use these variables directly, they are necessary as parameters to several of the resident utility subprograms which an extension may call. These parameters are described below:

INTEGER Item(Dim(1))

This array has one element per item in the data base. Each element indicates three things for the corresponding data item: whether or not it is required, its type and its offset in the code or range array.

The sign indicates a required item. Negative = required; positive = not required.

The type is found by performing: ABS(Item(I)) MOD 16. Possible values are:

0: code
1: string
2: integer
3: short real
4: long real
5: name
6: date
7 – 15: reserved

The code or range array (table) offset is found by performing ABS (Item(I)) DIV 16.

Syn$(Dim(2))[16]

The Syn$ (synonym) array contains information about all the synonyms. The value for each element is:

CHR$(item number) & "Synonym"

Code$(Dim(3))[16]

The Code$ array contains all the code values for the code-type data items. The value for each element is:

CHR$(number of codes for this item)&"code value" – first code value for this item

or

CHR$(0)&"code value" – subsequent codes for this item

When an item is a code-type item, its code values are found by using Item(I) to find the offset (element) on the code table, as described above.

REAL Range(Dim(4),2)

The Range array contains all default and defined ranges for numeric and date type data items. Range(1,1) contains the lower bound of the first range; Range (1,2) contains its upper bound, etc. To find the range of an item, Item(I) is used to find the offset (element) in the range table, as described above.

Utility Subprograms

The utility subprograms that remain in memory and are available to an extension subprogram are summarized below. A description of each of these routines follows this summary.

Reading from the screen:	Readfield: reads the value from a field on the screen
	FNString_input: calls Readfield to input a string
	FN Numbers_check: parses a string to see if it is a valid number
Softkey handling:	Print_label: labels softkeys
	FNKey: a function which waits for a softkey to be pressed
Parsing routines:	Scan: a scanner to find tokens in Command$
	Name_parse: parses Command$ for a name
	Date_parse: converts a date into a short-precision number
	Get_item: parses Command$ for a data item name
	F_error: displays an error comment
Date routines:	FNVerify_date: determines if a Month/Day/Year combination is valid
	FNFormat_date$: formats Month/Day/Year into a string
	FNEncode_date: encodes Month/Day/Year into a short-precision number
	Decode_date: decodes a short-precision number into a date
Disk volume routine:	FNAssign: a function which finds or assigns a file or volume
Read the search result:	Get_rec: reads a record from the search result
Utilities:	Printer: sets the OUTPUT TO device
	Lprt: prints to the OUTPUT TO device
	Dump_items: prints the item names in the current thread
	Dump_codes: prints the code values for an item

The following subprogram descriptions have sample calls which require that the passed parameters already be set up.

The READFIELD Subprogram

```
SUB Readfield(INTEGER Row,First_col,Len,Start_col,Hilight,Field$,Ret_code,Status)
```

Sample Call:

```
Field$ = “  ”
Sc = 1
Status = 0
CALL Readfield(5,30, – 36,Sc,129,Field$,Ret_code,Status)
```

This subprogram inputs the value from a field located anywhere on the screen. The field must be a contiguous string from 1 to 1600 characters in length. The contents entered into the field, a return code describing how the field was terminated, and a status word describing the state of the field are returned to the calling program. Invalid keys cause an audible beep and are ignored. A cursor is provided and can be moved with the arrow keys and the HOME and SHIFT-HOME keys. The CLEAR and CLEAR LINE keys clear the entire field, while the CLR → END key clears the field from the cursor position on. CONT, STORE, TAB, SHIFT-TAB, the special function keys, and the ↑ and ↓ keys terminate the input and return to the calling program. PRINTER IS 16 should be specified before the call if a different system printer has been specified.

Parameters

Parameter	Explanation
Row	screen row of input field
First_col	starting column of input field
Len	length of input field
Start_col	starting column for the cursor within Len
Hilight	numeric value of the video highlight for the field
Field$	string containing the field contents
Ret_code	code for the key that was pressed
Status	field status code

Argument Values for the Subprogram Call

Parameter	Value
Row	INTEGER variable. 1 to 20: echo typing – 20 to – 1: do not echo typing
First_col	INTEGER variable 1 to 80: actual first column
Len	INTEGER variable – 1600 to – 1: display the field 1 to 1600: field already displayed
Start_col	INTEGER variable 1 to ABS(Len)
Hilight	INTEGER variable 128 to 135
Field$	String variable or expression Field$ = “ ”: display highlighted blanks Field$ = string value: display highlighted string value
Ret_code	REAL variable value ignored
Status	REAL variable Status<0: display field and return without accepting input Status> = 0: accept input

Argument Values at Subprogram Exit

Parameters	Value
Row	ABS(Row)
First_col	First_col −1
Len	ABS(Len)
Start_col	current cursor position within field (between 1 and ABS(Len))
Hilight	unchanged
Field$	contains a new string value if one was entered.
Ret_code	−9: STOP was pressed −8 to −1: corresponding softkey was pressed. Ret_code = −1− (Key MOD 8) 0: CONT, STORE, EXECUTE or TAB was pressed 1: shift TAB was pressed 2: DOWN-ARROW was pressed 3: UP-ARROW was pressed
Status	0: field changed and is not blank 1: field did not change and is not blank 2: field changed and is now blank 3: field did not change and is blank

COMMON variables modified by Readfield

Keys$: contains new keyboard buffer
Switch(6): changed if user changes typewriter mode

Documented subprograms that call Readfield: FNString_input

The STRING INPUT Function

```
DEF FNString_input(INTEGER R,C,L,Sc,Field$,0,INTEGER Hilight)
```

Sample Call:

```
Sc=1
Field$=" "
ON FNString_input(5,30,36,Sc,Field$,0,129) GOTO K0,K1,K2,K3,K4,K5,K6,K7,
Stop, Tab_backward, Tab_forward
```

This function provides an alternate way of inputting the value of a field located anywhere on the screen. It calls the Readfield subprogram to get the input. FNString_input acts as a buffer between the Readfield subprogram and your main program, identifying fewer keystrokes and providing standard responses to status and return code information. Numbers input can be verified with FNNumber_check.

Parameters

Parameter	Explanation
R	screen row of input field
C	starting column of input field
L	length of input field
Sc	starting column for the cursor within L
Field$	string containing the field's contents
0	a QUERY/45-dependent variable; it **must** remain a zero
Hilight	numeric value of the video highlight for the field

Argument Values for the Subprogram Call

Parameter	Value
R	INTEGER variable 1 to 20: echo typing −20 to −1: do not echo typing
C	INTEGER variable 1 to 80: actual first column
L	INTEGER variable 1 to 1600
Sc	INTEGER variable 1 to ABS (Len)
Field$	String variable Field$ = " ": display highlighted blanks before entry allowed Field$ = string expression: display highlighted string expression before entry allowed
Hilight	INTEGER variable 128 to 135

Argument Values at Subprogram Exit

Parameter	Value
R	unchanged
C	unchanged
L	unchanged
Sc	no error: current cursor position error: 1
Field$	contains a new string value if one was entered
Hilight	unchanged

Subprograms called by String_input: Readfield

Possible Return Values for the function

Value	Meaning
1-8	softkey pressed
9	STOP key pressed
10	TAB backward
11	TAB forward

The NUMBER_CHECK Function

DEF FNNumber_check (N$,V,INTEGER C)

Sample Call:
N$="12345"
ON FNNumber_check (N$,V,C) GOTO Not_number

This function parses a character string to verify that it is a valid number. A zero is returned for the function if the number is valid, otherwise a message printed and a 1 is returned.

Parameters

Parameter	Explanation
N$	the string to be tested to see if it is a number
V	the numeric value of the number
C	the position of the first invalid character found

Argument Values for the Subprogram Call

Parameter	Value
N$	string variable character string that may contain a number
V	REAL variable value does not matter
C	INTEGER variable value does not matter

Argument Values at Subprogram Exit

Parameter	Value
N$	unchanged
V	value of the number in the string, if any
C	C>1: position of the first invalid character found, if any C=1: VAL(N$) failed

Possible values for the function:

0: success
1: failure

Error or status conditions checked:
String contains an invalid number (wrong character) or an out-of-range number.

The PRINT LABEL Subprogram

SUB Print_label(A$,B$)

Sample Calls:

CALL Print_label ("| | | | | DUMP | HELP | EXIT "," | | | | | SCREEN")

k0	k1	k2	k3	k4	k5	k6	k7
					DUMP SCREEN	HELP	EXIT

CALL Print_label (" | | | | | DUMP | | EXIT ","| | | | | SCREEN")

k0	k1	k2	k3	k4	k5	k6	k7
					DUMP SCREEN		EXIT

This subprogram displays the eight softkey labels on the CRT. Each label can have two lines, corresponding to the A$ and B$ parameters. The labels are automatically centered in their fields.

Parameters

Parameters	Explanation
A$	the top text line for all the labels
B$	the bottom text line for all the labels

Argument Values for the Subprogram Call

A$ and B$ contain characters for the softkey labels. The maximum number of characters per softkey is nine per row (top or bottom). Use vertical bars OR decimal points to separate the individual labels, even when a softkey isn't labeled. A$ and B$ must contain labels and either vertical bars or decimal points for all the labels up to the right-most label to be displayed; the remaining vertical bars or decimal points may be omitted.

Argument Values at Subprogram Exit

A$ and B$ are unchanged.

Documented subprograms that call Print_label: FNAssign

The KEY Function

DEF FNKey(Rc$)

Sample Call:

This call shows three labeled softkeys and the appropriate call to FNKey.

```
CALL Print_label("| | | | | DUMP | HELP | EXIT"," | | | | | SCREEN")
ON FNKey("000001230000") GOTO Dump,Help,Exit
```

This function returns the single-digit value which is the Nth digit in the string argument. N will be from 1 to 12 and is determined by the function using the mapping in the following table. This function waits for an enabled softkey to be pressed. (Softkeys are enabled with the Keysw variable = 1.) This function is useful when no screen input from Readfield is being done. FNKey recognizes eight softkeys, STOP, CONTINUE, UP-ARROW and DOWN ARROW.

This table shows how keystrokes determine the value returned by FNKey. The key in the left column is related to the value in the right column. This value indicates a digit position in the string. For example, when DOWN-ARROW is pressed, FNKey gets the value of the digit that is in the 12th position in the string argument. If the key pressed maps to a zero in the string, the keystroke is ignored, the computer beeps and FNKey is not exited.

Key	Position in the String
softkey k0 to k7	1 to 8 respectively
STOP	9
CONTINUE	10
UP-ARROW	11
DOWN-ARROW	12

Note that k0 to k7 = k8 to k15 = k16 to k23 = k24 to k31

Parameters

Parameter	Explanation
Rc$	determines the return value for the function based on the key pressed

Argument Values for the Subprogram Call

Rc$ is a string expression containing twelve digits. These digits map, by position, to the desired function value for each possible keystroke, as shown in the table above.

Argument Values at Subprogram Exit

Rc$ is unchanged.

COMMON values modified by FNKey:

Key: modified depending on the key pressed.
Keysw: temporarily set to 1 to enable softkeys; old value restored on exit

Documented Subprograms that call FNKey: Lprt, FNAssign

Error or status conditions checked:

If an unqualified key is pressed, the computer beeps and continues to wait.

The SCAN Subprogram

SUB Scan(Cmd)

Sample Calls:

```
Command$=Field$    ! Assume FNString_input just read something into Field$
CALL Scan(0)       ! get first token
! process token
CALL Scan(1)       ! get second token
! process next token
```

This subprogram scans Command$ for the presence of tokens. The numerical values returned in "Token" are:

Value	Token	Value	Token	Value	Token
0	END-OF-LINE	10	"	20	<ALPHA>
1	,	11	:	21	<ALPHA> >15 bytes
2	.	12	=	30	<NUMBER>
3	;	13	# or <>	31	<NUMBER> >15 bytes
4	+	14	<		
5	−	15	>		
6	*	16	<=		
7	/	17	>=		
8	(	18	$NULL		
9	)	19	INVALID CHARACTER		

"ALPHA" tokens must be 15 or less characters. Examples include "A", "A12", "A14B_12X". Case conversion is not done. Alpha tokens longer than 15 characters are given a token number of 21 but are not returned in Token$. They are delimited by Lineptrl and Lineptr2.

"NUMBER" tokens are integers with 15 or less characters. Longer numbers are given a token of 31 and not returned in Token$. The number +1.5E6 is broken up into tokens for the "+", "1", ".", "5", "E", and "6" substrings.

Parameters:

Parameter	Explanation
Cmd	zero: reset line pointers and get first token. non-zero: get next token.

COMMON:

Command$ – the command line to be scanned
Lineptrl – pointer into Command$ for beginning of token
Lineptr2 – pointer into Command$ for end of token
Token – the token number returned
Token$ – the token itself

The NAME_PARSE Subprogram

SUB Name_parse

Sample Call:

```
Command$ = "Sam Smith"
CALL Name_parse    ! When the program returns, Command$ contains "Smith, Sam"
```

The NAME_PARSE subprogram attempts to convert the string in Command$ to the format:

LAST,FIRST [MIDDLE][,SUFFIX]

If the string is already in this standard format, no changes are made. The string must contain at least two words for it to be a valid name. If this is not the case and the conversion cannot made, the subprogram returns with Err set to 18.

The DATE_PARSE Subprogram

```
SUB Date_parse(Cmd,SHORT Date)
```

Sample Call:

```
Command$ = "1-1-80"
CALL Date_parse(0,Date)   ! If successful, numerically encoded dates are stored in
                          ! the variable Date
```

This subprogram parses the string in Command$ to determine if it is in any of the accepted date formats. If so, FNEncode_date is called to return a numeric value for the date. If the string cannot be parsed correctly, Date_parse returns to its caller with Err set to 1. Device(2,3) determines if M-D-Y (= 1)or D-M-Y (= 5) format is used for parsing.

Parameters:

Parameters	Explanation
Cmd	Non-zero indicates continuation of parsing Command$. The token delimited by the variables Lineptr1 and Lineptr2 is examined first. Zero indicates that Command$ should be parsed from the beginning to the end of the line.
Date	Short real variable returning the encoded date.

The GET_ITEM Subprogram

```
SUB Get_item(Syn$(*),INTEGER Item(*),Set,Item,Itype,Ilen,Ioff,Idim, Isub,Tindex)
```

Sample Call:

```
Command$ = Field$   ! assume FNString_input just read a possible item name into
                    ! Field$
CALL Scan(0) ! set up scanner to point to first token
CALL Get_item(Syn$(*),Item(*),Set,Item,Itype,Ilen,Ioff,Idim, Isub,Tindex)
```

This subprogram parses Command$ for an item name in the format

[SET.] ITEM [(SUBCOUNT)]

If a thread was not set up on entry, a check is made to see if the item exists in a single set or if it is in several sets, a unique manual master or detail set. If so, the thread is set up to that set. Otherwise a "non-unique" error comment is generated. If the thread is set up on entry, the item must occur in one of the sets of the thread. If no set name is explicitly present, the left-most set in the thread in which the item occurs is used. On entry, Token$ must be set up to the first token to be used in the parse. On exit, Token$ points to the next token past the item name.

Parameters

Parameter	Explanation
Syn$(*)	Synonyms from information file
Item(*)	Item table from information file
Set	The set number item occurs in
Item	The item number
Itype	The item type from 1 to 7
Ilen	The item length
Ioff	The offset within a set (0-based). It does not account for multiple sets, but does account for subitem index
Idim	subitem count for that item (1 to N)
Isub	the specific subitem index
Tindex	how far into the thread

Errors

The following are values which may be returned in Err.

Value	Meaning
3	Non-existent item or item not in set.
4	Subitem index required.
5	Subitem index out of range or invalid.
11	Parenthesis expected.
14	Invalid set name or set not in thread.
16	Non-unique item. Set must be specified.

The F_ERROR Subprogram

```
SUB F_error(INTEGER I)
```

Sample Call:

```
CALL F_error(Err)
```

This subprogram displays one of the error comments below:

1. Invalid character.
2. Improper expression.
3. Non-existent item or item not in set.
4. Subitem index required.
5. Subitem index out of range or invalid.
6. Invalid number.
7. Type incompatibility.
8. Invalid date.
9. Improper string.
10. Invalid code value.
11. Parenthesis expected.
12. Comma expected.
13. String too long.
14. Invalid set name or set not in thread.

15. Value entered is not within allowable range.
16. Item is not unique; a set must be specified.
17. Maximum number of items exceeded.
18. Invalid NAME format.
19. An A or D or blank must be entered.
20. There is nothing to sort.
21. Invalid thread name.
22. Command ignored; search result is empty.
23. Quotes must enclose entire string due to reserved keyword.
24. SELECT FOR is not allowed when the search result overflows to disc.
25. There is nothing to restore.
26. Width of item list exceeds width of printer.
27. Null value not allowed for required item.
28. End of line expected.
29. No more than 10 nested parentheses are allowed.
30. The maximum number of multiple values or OR's has been exceeded.
31. An item name must occur on the left-hand side of the relational operator.
32. Ambiguous request: FIND ALL ENTRIES requires a blank search expression.

The VERIFY_DATE Function

```
DEF FNVerify_date(INTEGER M,D,Y)
```

Sample Call:

```
IF FNVerify_date(M,D,Y) THEN Ok
```

This action verifies date inputs of month, day, and year to determine if the date is a valid calendar date. The value "1" is returned if the date is valid; otherwise, the value "0" is returned.

Parameters:

Parameters	Explanation
M	Integer (1-12) for month
D	Integer (1-31) for day
Y	Integer (0-5475) for year

The FORMAT_DATE$ Function

```
DEF FNFormat_date$(INTEGER Format,SHORT Dcode)
```

Sample Call:

```
Field$=FNFormat_date$(1,Date)
```

The number in Dcode is decoded and then formatted as indicated by the format option. This string is returned to the calling program. If Dcode contains a number which is not in the valid range (−999998 through 999999), Format_date returns a null string. The formats are as follows:

Format code	Formatted example
1	February 16, 1979
2	February 16, 79
3	Feb 16, 1979
4	Feb 16, 79
5	16 February 1979
6	16 February 79
7	16 Feb 1979
8	16 Feb 79
9	2/16/1979
10	2/16/79
11	2-16-1979
12	2-16-79
13	16/2/1979
14	16/2/79
15	16-2-1979
16	16-2-79

Parameters:

Parameters	Explanation
Format	Code describing the format to be used.
Dcode	Encoded date which is to be formatted.

The ENCODE_DATE Function

```
DEF FNEncode_date(INTEGER M,D,Y)
```

Sample Call:

```
Date=FNEncode_date(M,D,Y)
```

Encode_date receives numbers for month, day, and year, and calculates the unique integer between −999998 and 999999 corresponding to this date. If the month-day-year combination does not represent a valid calendar date in the legal date range January 1, 0000 through September 11, 5475 the null value −999999 is returned.

Parameters:

Parameters	Explanation
M	Month
D	Day
Y	Year

The DECODE_DATE Subprogram

```
SUB Decode_date(SHORT Dcode,INTEGER M,D,Y)
```

Sample Call:

```
CALL Decode_date(Dcode,M,D,Y)
```

Decode_date receives an encoded date value from −999998 to 999999 and determines the month, day, and year represented by this code. If the encoded value is −999999 (the null value) then month, day, and year are set to 0.

Parameters:

Parameters	Explanation
D	Day
Dcode	Encoded date
M	Month
Y	Year

The ASSIGN Function

```
DEF FNAssign(#1,F$,V$,P$,A,Sc$)
```

Sample Call:

```
ON FNAssign(#2,File$,Base$(3),"SEQUOIA",791,Sc$) GOTO OK,No,No,No
```

The ASSIGN function locates the mass storage device in which the disc labeled "V$" (volume name) is mounted. If found, it examines the variable "A" to determine if it should assign a file or just return the select code of the mass storage device. If more than one-disc is mounted with the same volume name, the first volume containing the file name will be returned. If the volume isn't mounted, the program requests that the user mount the volume or abort the request.

Note

Help message number 43 is a general-purpose explanation for mounting any volume. Help message 79 explains mounting the root file volume. The screen, display line, and softkey labels are altered by this subprogram. The softkeys may change to ABORT only. Global_error is assumed to be in effect.

Parameters:

Parameters	Explanation
#1	the file number that will be assigned
F$	the file name
V$	the name of the volume the file should be on
P$	the protect code or null
A	"ASSIGNABLE" flag: 0 = not assignable 1 = assignable 2 = mount volume only; don't assign a file. If A>2, A DIV 10 is the number of the HELP message to be called if the HELP softkey is pressed; the "ASSIGNABLE" flag is then (A MOD 10). If A< =2, the QUERY disc is the requested volume and the HELP message is hard-coded in this routine.
Sc$	the select code of the volume containing the file

Possible Return Values for the function:

If the "ASSIGNABLE" flag is 0 or 1:

Value	Meaning
1	File has been assigned
2	Found the file and it is not type "DATA"
3	Can't assign the file since it is not type "DATA"
4	User aborted the request

If the "ASSIGNABLE" flag is 2:

Value	Meaning
1	Volume is mounted
4	User aborted the request

The GET_REC Subprogram

```
SUB Get_rec(#1,INTEGER Ptr,Buffer$)
```

Sample Call:

```
INTEGER Ptr
Ptr=Vars(10)=0
FOR I=1 TO ABS(Dim(5))
  CALL Get_rec(#1,Ptr,Buffer$)        ! Process the record
NEXT I
```

The GET_REC subprogram reads a record from the SEARCH RESULT into Buffer$. Buffer$ contains the concatenated records of all sets in the thread that are positive. For example, if the thread contains +1, +3, −5, +6, then records from sets 1, 3, and 6 are concatenated. The first call to Get_rec should have Vars(10) = Ptr = 0. Ptr is incremented on return. Vars(10) is incremented as overflow files are used. GLOBAL_ERROR should be in effect. Since FNAssign is called, the softkeys may be set to ABORT, the DISP line may be blanked, the CRT may be left intact or blanked, and PRINTER IS may be reset to 16. On exit, Err is set to 1 if any error or abort occurs.

Parameters:

Parameters	Explanation
#1	for overflow file assignment.
Ptr	pointer into the search result.
Buffer$	to return result.

The PRINTER Subprogram

```
SUB Printer(I)
```

Sample Call:

```
CALL Printer(0)
```

This subprogram sets the current "OUTPUT TO" device. The passed parameter should be 0 for the CRT, 1 for the thermal printer, and 2 for the configured external printer. This subprogram sets the values for the Printer(*) array in COMMON. The Lprt subprogram then uses this array to determine where to send the output.

The LPRT Subprogram

```
SUB Lprt(Buffer$,Cctl)
```

Sample Call:

```
DIM Buffer$[100]
CALL Printer(0)   ! Set CRT as OUTPUT device
CALL Lprt ("",0)   ! initializing the printer
FOR I=1 TO 100
  Buffer$="This is line number" & VAL$(I)&"."
  CALL Lprt (Buffer$,3) ! Print each line
  IF Err THEN Abort
NEXT I
Abort: CALL Lprt(" ",2) ! Print the last page number, do a page eject and shut off
                        ! the printer
```

This subprogram prints the Buffer$ to the current OUTPUT TO device. A blank line followed by a line with the page number is printed at the bottom of each page. For example, if the page length is 65 lines, 63 text lines are printed. At each invocation, PRINTER IS is set to the current output device specified in COMMON.

If the current output device is the CRT, the Lprt subprogram automatically pauses between pages enabling you to list the next page, dump the screen, or abort the subprogram. You can abort the subprogram at any time while printing by pressing the ABORT softkey which causes Err to be set to non-zero.

While using the Lprt subprogram, each line printed must be a string and therefore all numbers must be converted using VAL$.

Parameters:

Parameter	Explanation
Buffer$	output line buffer
Cctl	Carriage Control: 0 = RESET (Page #1, PRINTER IS, SOFTKEYS, Set KEYSW to -1.) 1 = PAGE EJECT 2 = EXIT (Page eject. Set KEYSW to −2) 3 = If necessary, do a page eject. Then print the line in LBF$.

COMMON:

Printer(1) = Select Code
Printer(2) = HP-IB address
Printer(3) = Page width
Printer(4) = Page length
Printer(5) = Current line number
Printer(6) = Current page number
Err = ABORT flag.

The DUMP_ITEMS Subprogram

```
SUB Dump_items
```

Sample Call:

```
CALL Dump_items
```

This subprogram prints item names in current thread to the thermal printer.

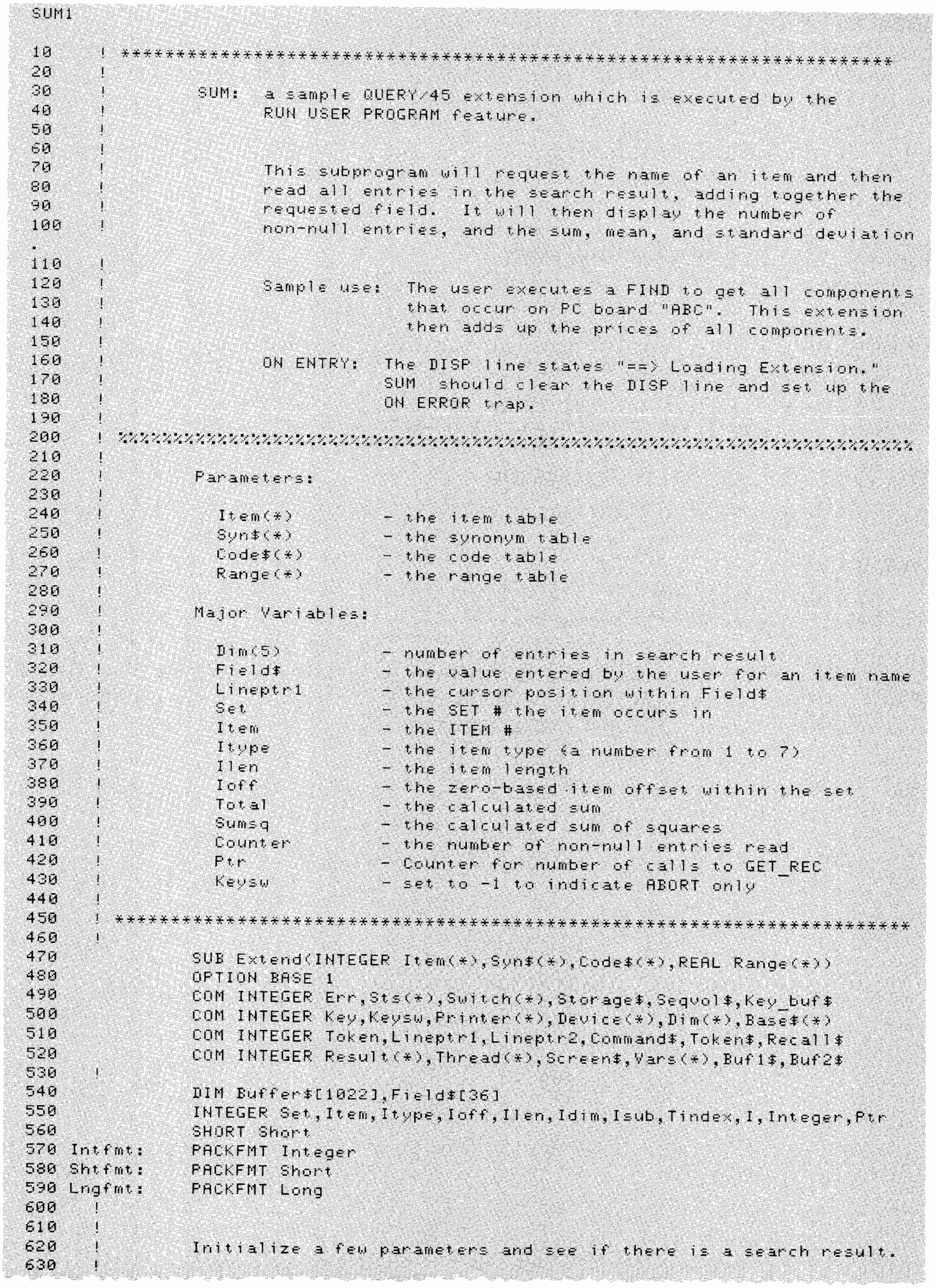
The DUMP_CODES Subprogram

```
SUB Dump_codes(INTEGER Ptr,Code$(*))
```

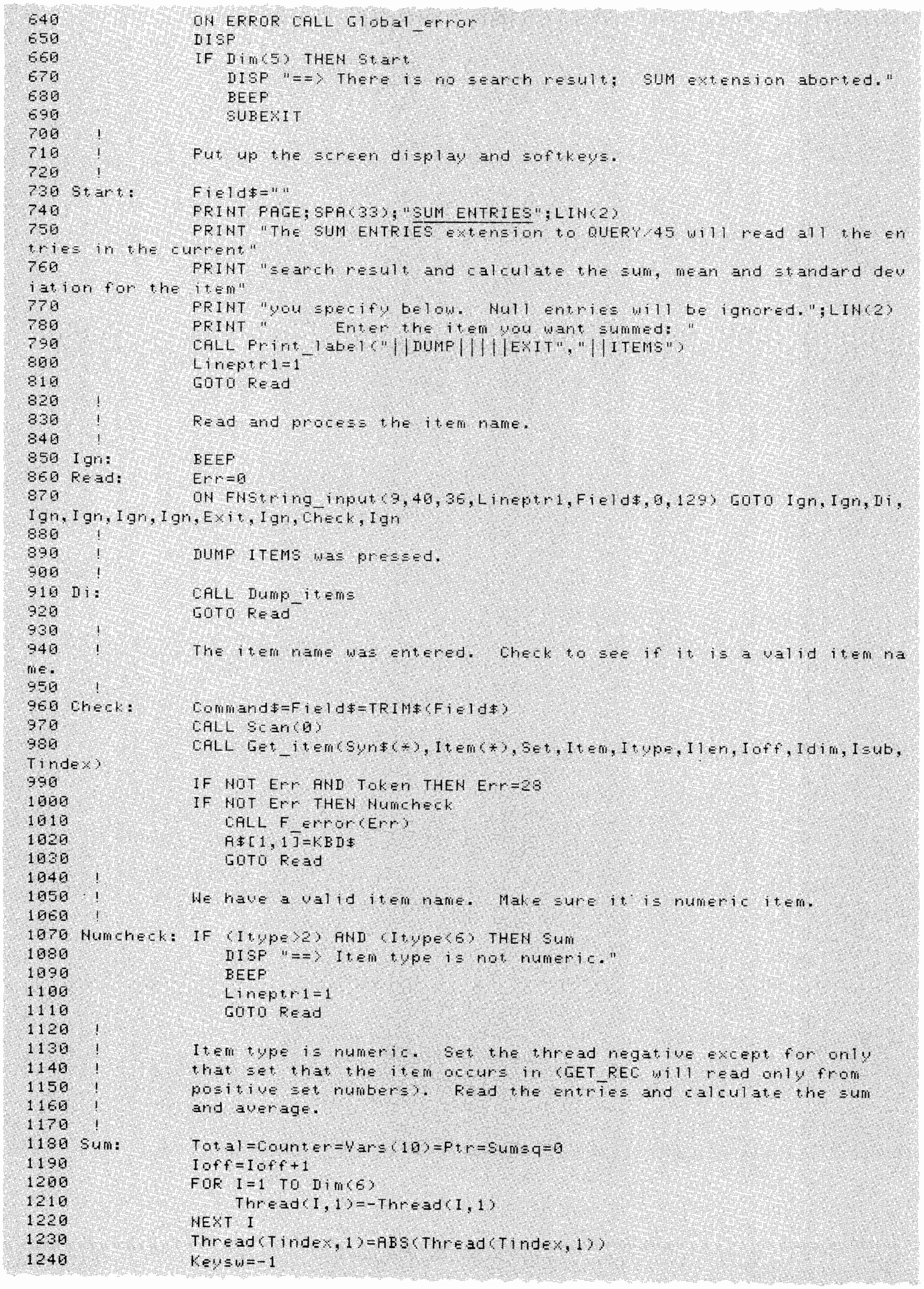
Sample Call:

```
Ptr=ABS(Item(I)) DIV 16
CALL Dump_codes(Ptr,Code$(*))
```

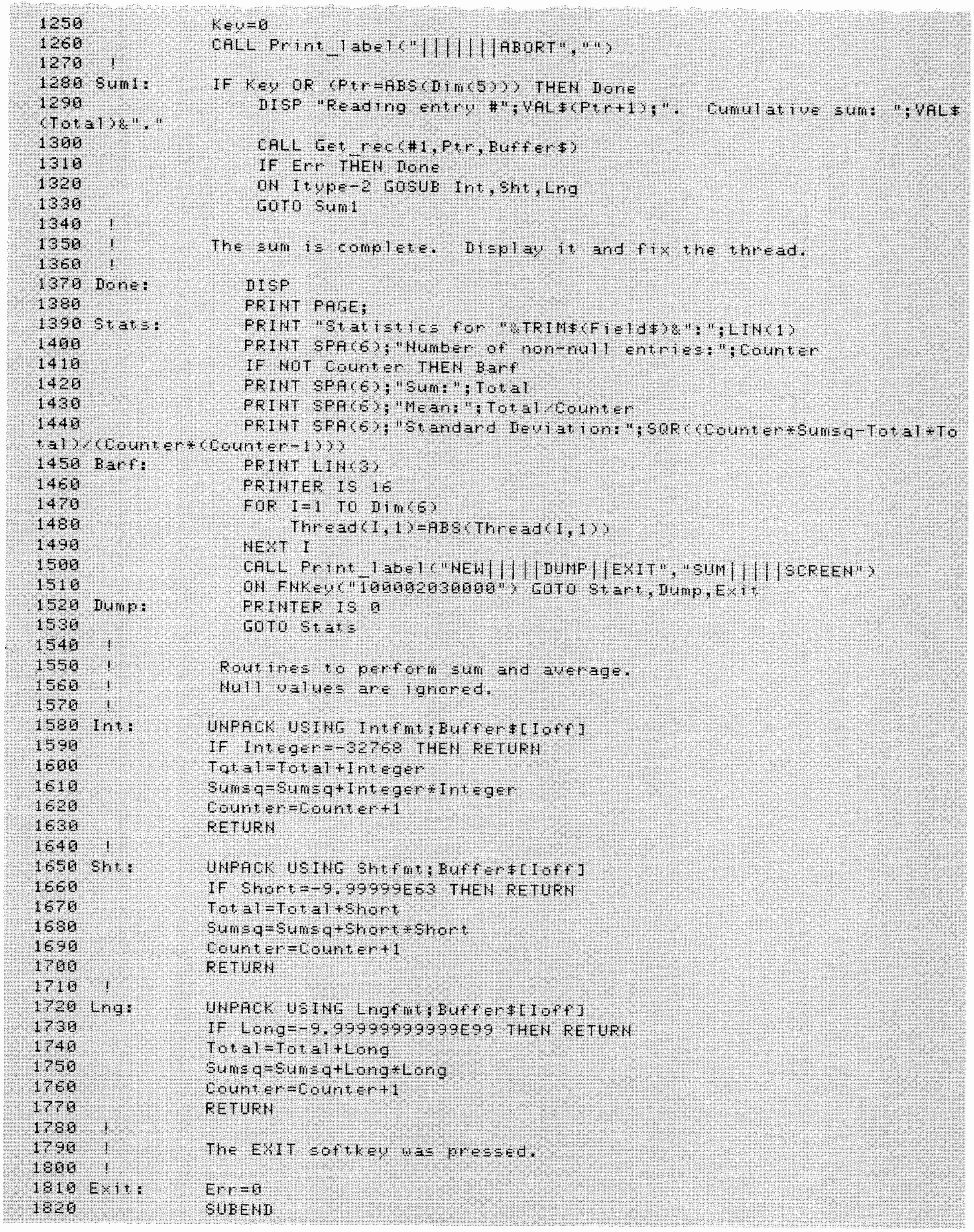
The DUMP_CODES subprogram prints the codes for an item on the thermal printer. Ptr is a pointer into the code table for the first code defined for that item. The Code$ array is the code table passed to the extension as a calling parameter.



```
SUM1

10     ! ******************************************************************
20     !
30     !          SUM:  a sample QUERY/45 extension which is executed by the
40     !                RUN USER PROGRAM feature.
50     !
60     !
70     !                This subprogram will request the name of an item and then
80     !                read all entries in the search result, adding together the
90     !                requested field.  It will then display the number of
100    !                non-null entries, and the sum, mean, and standard deviation
.
110    !
120    !                Sample use:  The user executes a FIND to get all components
130    !                               that occur on PC board "ABC".  This extension
140    !                               then adds up the prices of all components.
150    !
160    !                ON ENTRY:   The DISP line states "==> Loading Extension."
170    !                            SUM  should clear the DISP line and set up the
180    !                            ON ERROR trap.
190    !
200    ! %%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
210    !
220    !         Parameters:
230    !
240    !           Item(*)          - the item table
250    !           Syn$(*)          - the synonym table
260    !           Code$(*)         - the code table
270    !           Range(*)         - the range table
280    !
290    !         Major Variables:
300    !
310    !           Dim(5)           - number of entries in search result
320    !           Field$           - the value entered by the user for an item name
330    !           Lineptr1         - the cursor position within Field$
340    !           Set              - the SET # the item occurs in
350    !           Item             - the ITEM #
360    !           Itype            - the item type (a number from 1 to 7)
370    !           Ilen             - the item length
380    !           Ioff             - the zero-based item offset within the set
390    !           Total            - the calculated sum
400    !           Sumsq            - the calculated sum of squares
410    !           Counter          - the number of non-null entries read
420    !           Ptr              - Counter for number of calls to GET_REC
430    !           Keysw            - set to -1 to indicate ABORT only
440    !
450    ! ******************************************************************
460    !
470             SUB Extend(INTEGER Item(*),Syn$(*),Code$(*),REAL Range(*))
480             OPTION BASE 1
490             COM INTEGER Err,Sts(*),Switch(*),Storage$,Seqvol$,Key_buf$
500             COM INTEGER Key,Keysw,Printer(*),Device(*),Dim(*),Base$(*)
510             COM INTEGER Token,Lineptr1,Lineptr2,Command$,Token$,Recall$
520             COM INTEGER Result(*),Thread(*),Screen$,Vars(*),Buf1$,Buf2$
530    !
540             DIM Buffer$[1022],Field$[36]
550             INTEGER Set,Item,Itype,Ioff,Ilen,Idim,Isub,Tindex,I,Integer,Ptr
560             SHORT Short
570 Intfmt:     PACKFMT Integer
580 Shtfmt:     PACKFMT Short
590 Lngfmt:     PACKFMT Long
600    !
610    !
620    !        Initialize a few parameters and see if there is a search result.
630    !
```



```
640            ON ERROR CALL Global_error
650            DISP
660            IF Dim(5) THEN Start
670               DISP "==> There is no search result;  SUM extension aborted."
680               BEEP
690               SUBEXIT
700   !
710   !        Put up the screen display and softkeys.
720   !
730 Start:     Field$=""
740            PRINT PAGE;SPA(33);"SUM ENTRIES";LIN(2)
750            PRINT "The SUM ENTRIES extension to QUERY/45 will read all the en
tries in the current"
760            PRINT "search result and calculate the sum, mean and standard dev
iation for the item"
770            PRINT "you specify below.  Null entries will be ignored.";LIN(2)
780            PRINT "       Enter the item you want summed: "
790            CALL Print_label("||DUMP||||EXIT","||ITEMS")
800            Lineptr1=1
810            GOTO Read
820   !
830   !        Read and process the item name.
840   !
850 Ign:       BEEP
860 Read:      Err=0
870            ON FNString_input(9,40,36,Lineptr1,Field$,0,129) GOTO Ign,Ign,Di,
Ign,Ign,Ign,Ign,Exit,Ign,Check,Ign
880   !
890   !        DUMP ITEMS was pressed.
900   !
910 Di:        CALL Dump_items
920            GOTO Read
930   !
940   !        The item name was entered.  Check to see if it is a valid item na
me.
950   !
960 Check:     Command$=Field$=TRIM$(Field$)
970            CALL Scan(0)
980            CALL Get_item(Syn$(*),Item(*),Set,Item,Itype,Ilen,Ioff,Idim,Isub,
Tindex)
990            IF NOT Err AND Token THEN Err=28
1000           IF NOT Err THEN Numcheck
1010              CALL F_error(Err)
1020              A$[1,1]=KBD$
1030              GOTO Read
1040  !
1050  !        We have a valid item name.  Make sure it is numeric item.
1060  !
1070 Numcheck: IF (Itype>2) AND (Itype<6) THEN Sum
1080              DISP "==> Item type is not numeric."
1090              BEEP
1100              Lineptr1=1
1110              GOTO Read
1120  !
1130  !        Item type is numeric.  Set the thread negative except for only
1140  !        that set that the item occurs in (GET_REC will read only from
1150  !        positive set numbers).  Read the entries and calculate the sum
1160  !        and average.
1170  !
1180 Sum:      Total=Counter=Vars(10)=Ptr=Sumsq=0
1190           Ioff=Ioff+1
1200           FOR I=1 TO Dim(6)
1210               Thread(I,1)=-Thread(I,1)
1220           NEXT I
1230           Thread(Tindex,1)=ABS(Thread(Tindex,1))
1240           Keysw=-1
```



```
1250              Key=0
1260              CALL Print_label("|||||||ABORT","")
1270   !
1280 Sum1:        IF Key OR (Ptr=ABS(Dim(5))) THEN Done
1290                  DISP "Reading entry #";VAL$(Ptr+1);".  Cumulative sum: ";VAL$
(Total)&"."
1300                  CALL Get_rec(#1,Ptr,Buffer$)
1310                  IF Err THEN Done
1320                  ON Itype-2 GOSUB Int,Sht,Lng
1330                  GOTO Sum1
1340   !
1350   !          The sum is complete.  Display it and fix the thread.
1360   !
1370 Done:          DISP
1380                PRINT PAGE;
1390 Stats:         PRINT "Statistics for "&TRIM$(Field$)&":";LIN(1)
1400                PRINT SPA(6);"Number of non-null entries:";Counter
1410                IF NOT Counter THEN Barf
1420                PRINT SPA(6);"Sum:";Total
1430                PRINT SPA(6);"Mean:";Total/Counter
1440                PRINT SPA(6);"Standard Deviation:";SQR((Counter*Sumsq-Total*To
tal)/(Counter*(Counter-1)))
1450 Barf:          PRINT LIN(3)
1460                PRINTER IS 16
1470                FOR I=1 TO Dim(6)
1480                    Thread(I,1)=ABS(Thread(I,1))
1490                NEXT I
1500                CALL Print_label("NEW|||||DUMP||EXIT","SUM|||||SCREEN")
1510                ON FNKey("100002030000") GOTO Start,Dump,Exit
1520 Dump:          PRINTER IS 0
1530                GOTO Stats
1540   !
1550   !           Routines to perform sum and average.
1560   !           Null values are ignored.
1570   !
1580 Int:         UNPACK USING Intfmt;Buffer$[Ioff]
1590              IF Integer=-32768 THEN RETURN
1600              Total=Total+Integer
1610              Sumsq=Sumsq+Integer*Integer
1620              Counter=Counter+1
1630              RETURN
1640   !
1650 Sht:         UNPACK USING Shtfmt;Buffer$[Ioff]
1660              IF Short=-9.99999E63 THEN RETURN
1670              Total=Total+Short
1680              Sumsq=Sumsq+Short*Short
1690              Counter=Counter+1
1700              RETURN
1710   !
1720 Lng:         UNPACK USING Lngfmt;Buffer$[Ioff]
1730              IF Long=-9.99999999999E99 THEN RETURN
1740              Total=Total+Long
1750              Sumsq=Sumsq+Long*Long
1760              Counter=Counter+1
1770              RETURN
1780   !
1790   !          The EXIT softkey was pressed.
1800   !
1810 Exit:        Err=0
1820              SUBEND
```

Chapter 10

Using the Formal Command Language

When you are familiar with various features of QUERY/45, you can save time by using the formal command language rather than the softkeys. To use the formal commands, press the FORMAL COMMAND softkey at the beginning of QUERY/45. When you have accessed the formal command, press CONTINUE to perform the function.

Another feature that can save you time is the RECALL softkey. This softkey is used to re-display any command that has been previously entered, enabling you to view, modify and reuse previous commands. The commands are stored into a 321-character buffer on a last in, first out basis. Each time the RECALL softkey is pressed, a previous command is displayed in the command area on the CRT.

When the recall buffer becomes full, each new command causes one or more of the oldest commands, depending on size, to be lost.

This chapter lists the formal command syntax for each of the subsystems.

Syntax Guidelines

The following guidelines are used in these formal command descriptions.

[ ]	All items enclosed in brackets are optional.
|	A vertical line between two parameters means "or"; only one of the parameters can be included.
DOT MATRIX	All items in dox matrix must appear exactly as shown.
...	Three dots indicate that the previous item can be repeated.

SHOW SUBSYSTEM

```
OUTPUT TO    CRT | THERMAL | PRINTER
SHOW    BASES | STRUCTURE | SCHEMA | SETS |THREADS
HELP
STOP
DEFINE    FORM | THREAD
RUN file name VOLUME = volume name (Run User Program)
```

UPDATE SUBSYSTEM

```
ADD TO set name [USING FORM]
[PAUSE] DELETE IN set FOR search expression
MODIFY IN set [USING FORM] FOR search expression
[PAUSE] REPLACE IN set item = value FOR search expression
```

SEARCH SUBSYSTEM

```
BROWSE set
FIND [set | thread FOR] search expression
SELECT FOR search expression
RESTORE
[LINEAR] LIST item, item, item
SORT BY item [A | D], item [A | D], ...
```

Appendix A
System Configuration

To run your System 45 Data Base Management System, you need:

- 9845B with at least 187K bytes of Read/Write Memory.
- an internal printer.
- the Mass Storage ROM.
- the Advanced Programming ROM.
- the two IMAGE/45 ROMs.

You also need a mass storage device which is used to store your QUERY/45 program. If you are using an HP 9885 Flexible Disc Drive, then three flexible discs (P/N 9230-0420) are needed to store QUERY/45 and the sample data base.

Data Base Management can be ordered in several different ways. The following table shows the various options. An asterisk (*) indicates which item goes with which package.

			Options Numbers		
Qty.	**Description**	**HP Part Number**	**98430A**	**98429A**	**98428A**
2	IMAGE/45 ROMs	09845-65526 (right) 09845-65527 (left)	*	*	
3	QUERY/45 (tape cartridges)	09845-14754 09845-14755 09845-14756	*		*
1	IMAGE/45 Utilities (tape cartridge)	09845-14757	*	*	
1	Sample Data Base (tape cartridge)	11141-10666	*	*	*
1	IMAGE/45 Programming Manual	09845-91055	*	*	
1	QUERY/45 User's Guide	09845-91056	*		*
1	System 45 Quick Reference	09845-92015	*	*	
1	Data Base Design Kit	09845-91057	*	*	*
1	Binder	9282-0868	*	*	*
2	Error Message Stickers	7121-0234 (hard) 7121-8890 (status)	*	*	
1	Registration Reply Card		*	*	*
1	Configuration Letter	09845-91058	*	*	
1	Certificate of Copyrights		*	*	*

Please contact your Sales and Service Office if any listed items are missing.

Appendix B
Information File

Structure of the Information File

An **Information File** is a file created by QUERY/45 which contains information about a particular data base which is unique to QUERY/45, such as name synonyms, threads and code values. This file can be accessed by those of you who want to use IMAGE/45 to write your own programs, but still take advantage of the additional features provided by QUERY/45.

The Information File for a data base is created automatically by QUERY/45; its name is the name of the root file followed by 00. The Information File is made up of a number of 256-byte records. Each record contains a series of values for integers, strings and real numbers. These values can be retrieved using the READ# statement (explained in the Operating and Programming manual).

The format of the Information File is given below. The length of each string value is specified within square brackets [ ]. When there are a number of the same type of values, the total number is specified within parentheses; this number may be in terms such as "(number of items in data base)". Here is the general format of an Information File.

RECORD 1:	INTEGER	Number_sets, Number_items, Number_synonyms, Number_codes, Number_ranges, pointer to Record X, pointer to Record Y
	STRING	Base_description$[60]
RECORD 2,3:	INTEGER	Form_flag (one per set)
	STRING	Form_volumes$[8] (one per set)

At the beginning of Record #4 is an integer array:

INTEGER Item(Number_items)

This array has one element per item in the data base. Each element indicates three things for the corresponding data item:

- Required or not required
- Type
- Offset in code or range array

The sign indicates a required item:

negative = required
positive = not required

The type is found by performing: ABS(Item(I)) MOD 16.

Possible values are:

0: code
1: string
2: integer
3: short real
4: long real
5: name
6: date
7-15: reserved

The code or range table offset is found by performing ABS (Item(I)) DIV 16.

Following the integer array is a synonym array:

Syn$(Number_synonyms)[16]

The Syn$ (synonym) array contains information about all the synonyms. The value for each element is:

CHR$(item number) & "Synonym"

Following the synonym array is a code array:

Code$(Number_codes)[16]

The Code$ array contains all the code values for the code-type data items. The value for each element is:

CHR$(number of codes for this item)&"code value" — first code value for this item

or

CHR$(0)&"code value" — subsequent codes for this item

When an item is a code-type item, its code values are found by using Item(I) to find the offset (element) on the code table, as described above.

The last information found in these records is:

REAL Range(Dim(4),2)

The Range array contains all default and defined ranges for numeric and date type data items. Range(1,1) contains the lower bound of the first range; Range (1,2) contains its upper bound, etc. To find the range of an item, Item(I) is used to find the offset (element) in the range table, as described previously.

The following record (RECORD X), which is found by the record X pointer, contains:

RECORD X: STRING Set_description$(Number_sets)[60]

This record contains the set description for each data set.

The next four records, which are found by the record Y pointer, are defined as:

RECORD Y..Y+3: STRING Thread$ [15]
INTEGER Thread (10,2)

The last records in the Information File contain the thread information. A string array, Thread$, contains the thread names. The integer array, thread, contains the linked data set number followed by the data item number. Each thread defined is stored in a different record.

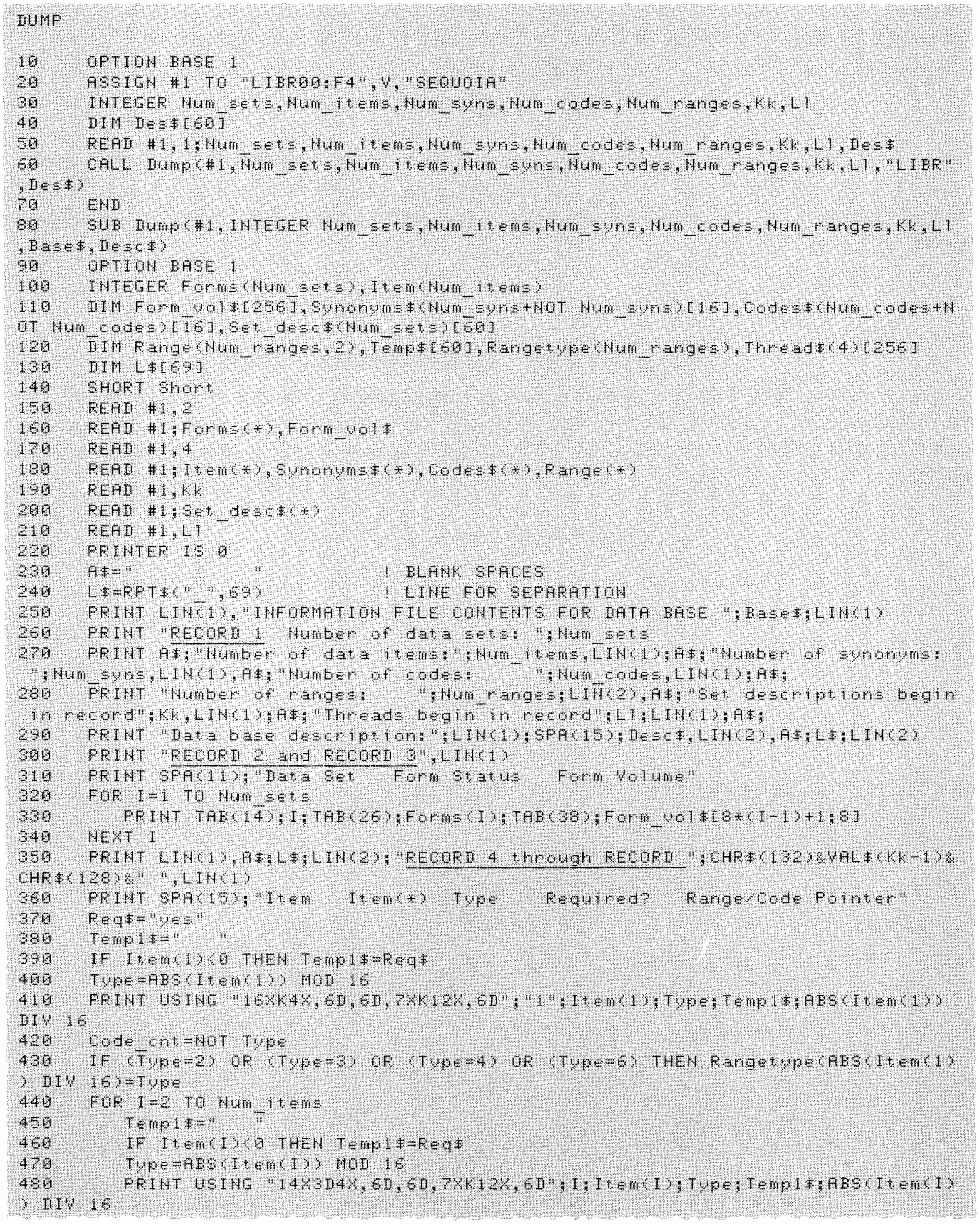
Purging the Information File

When you purge a data base from your disc, you need to also purge the information file. The information file has a protect word assigned by QUERY/45 when it is created. This protect word is "SEQUOIA" and to purge the information file type the following and press EXECUTE.

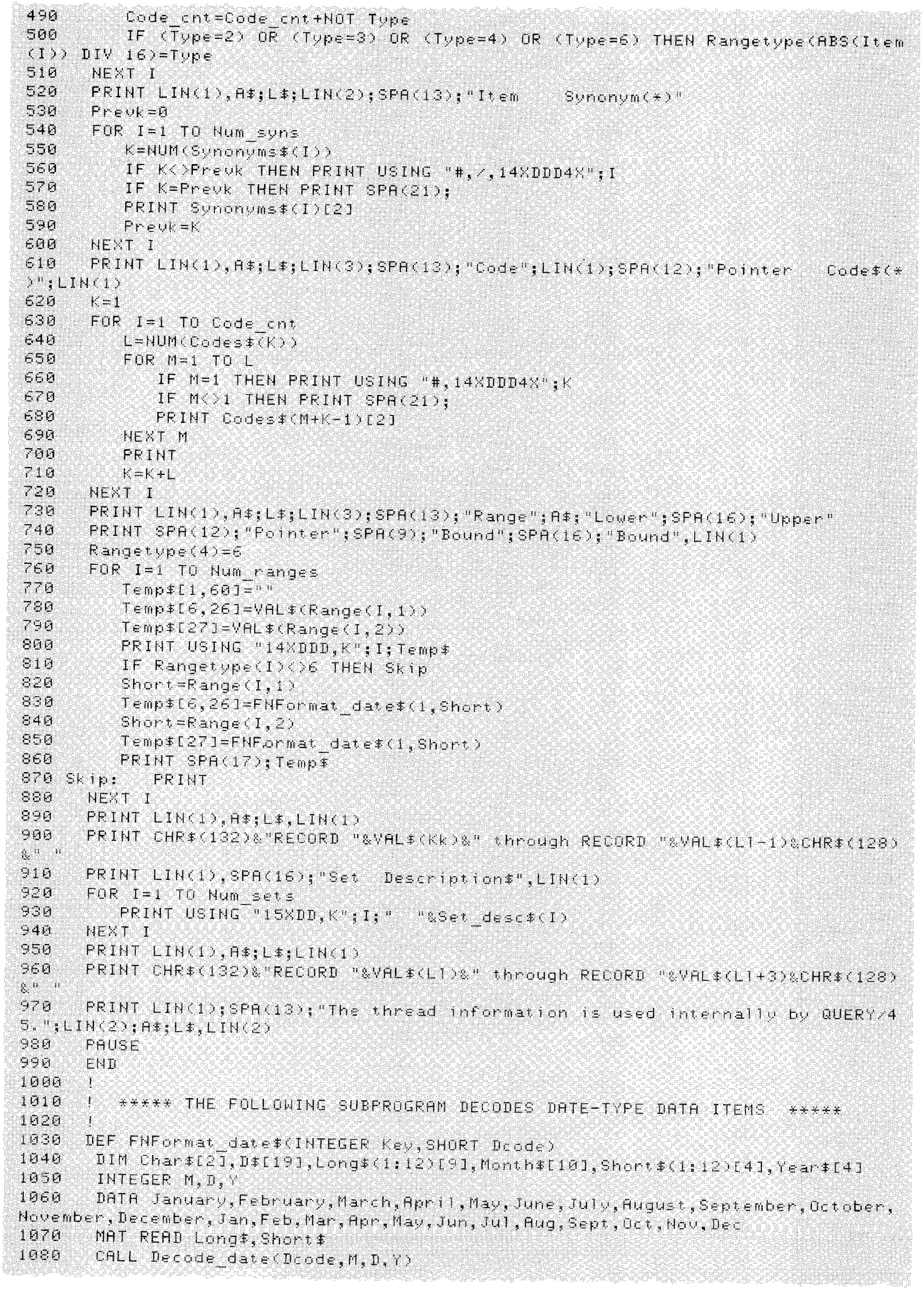
```
PURGE "file name00:msus","SEQUOIA"
```

Listing the Information File

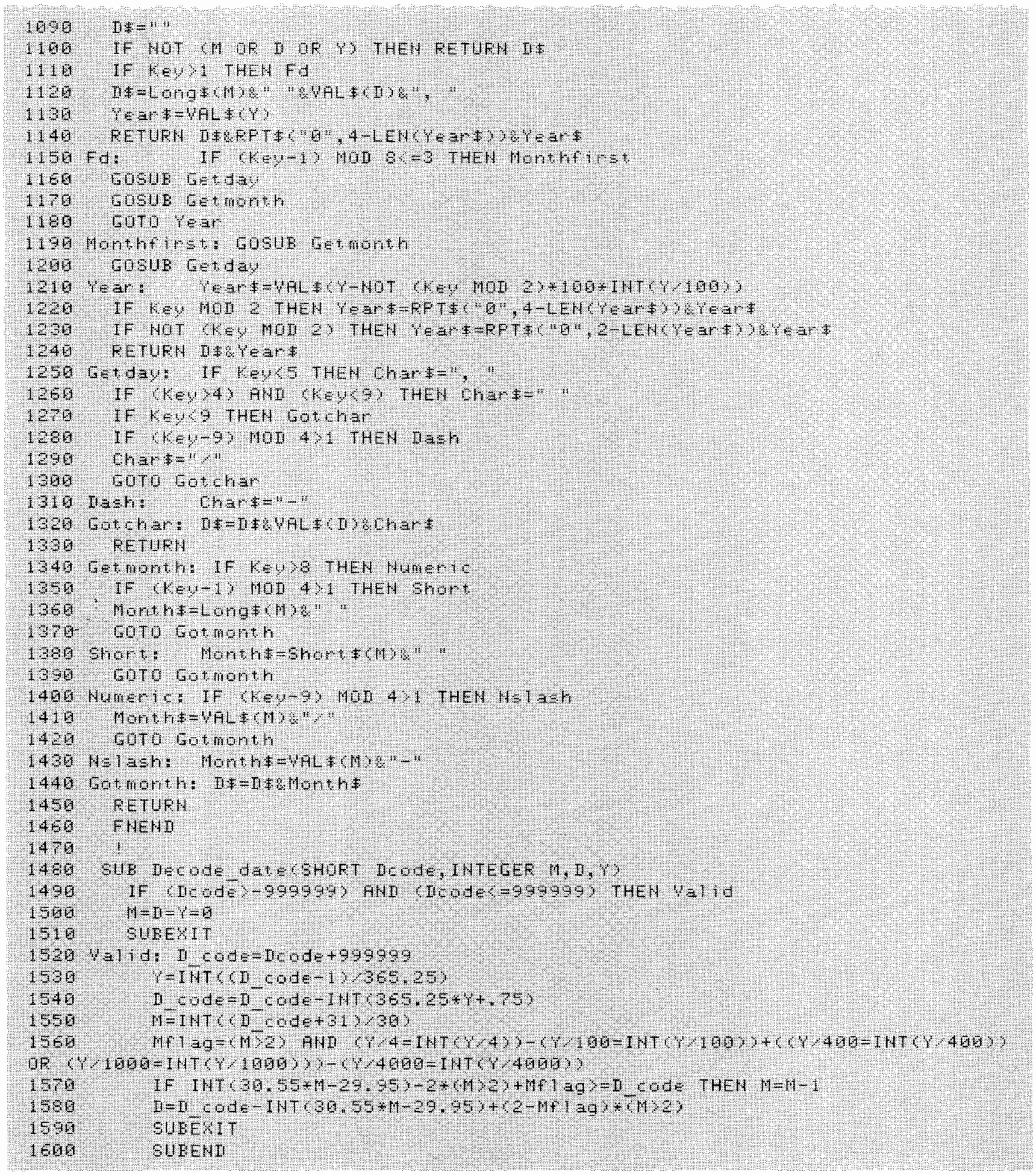
Here is a program that dumps the LIBR data base Information File. To dump another Information File, change line 20 from LIBR00:F8 to your Information File name and in line 60, at the end of the line, "LIBR" needs to be replaced with your data base name.



```
DUMP

10      OPTION BASE 1
20      ASSIGN #1 TO "LIBR00:F4",V,"SEQUOIA"
30      INTEGER Num_sets,Num_items,Num_syns,Num_codes,Num_ranges,Kk,L1
40      DIM Des$[60]
50      READ #1,1;Num_sets,Num_items,Num_syns,Num_codes,Num_ranges,Kk,L1,Des$
60      CALL Dump(#1,Num_sets,Num_items,Num_syns,Num_codes,Num_ranges,Kk,L1,"LIBR"
,Des$)
70      END
80      SUB Dump(#1,INTEGER Num_sets,Num_items,Num_syns,Num_codes,Num_ranges,Kk,L1
,Base$,Desc$)
90      OPTION BASE 1
100     INTEGER Forms(Num_sets),Item(Num_items)
110     DIM Form_vol$[256],Synonyms$(Num_syns+NOT Num_syns)[16],Codes$(Num_codes+N
OT Num_codes)[16],Set_desc$(Num_sets)[60]
120     DIM Range(Num_ranges,2),Temp$[60],Rangetype(Num_ranges),Thread$(4)[256]
130     DIM L$[69]
140     SHORT Short
150     READ #1,2
160     READ #1;Forms(*),Form_vol$
170     READ #1,4
180     READ #1;Item(*),Synonyms$(*),Codes$(*),Range(*)
190     READ #1,Kk
200     READ #1;Set_desc$(*)
210     READ #1,L1
220     PRINTER IS 0
230     A$="             "              ! BLANK SPACES
240     L$=RPT$("_",69)                 ! LINE FOR SEPARATION
250     PRINT LIN(1),"INFORMATION FILE CONTENTS FOR DATA BASE ";Base$;LIN(1)
260     PRINT "RECORD 1  Number of data sets: ";Num_sets
270     PRINT A$;"Number of data items:";Num_items,LIN(1);A$;"Number of synonyms:
 ";Num_syns,LIN(1),A$;"Number of codes:       ";Num_codes,LIN(1);A$;
280     PRINT "Number of ranges:      ";Num_ranges;LIN(2),A$;"Set descriptions begin
 in record";Kk,LIN(1);A$;"Threads begin in record";L1;LIN(1);A$;
290     PRINT "Data base description:";LIN(1);SPA(15);Desc$,LIN(2),A$;L$;LIN(2)
300     PRINT "RECORD 2 and RECORD 3",LIN(1)
310     PRINT SPA(11);"Data Set   Form Status   Form Volume"
320     FOR I=1 TO Num_sets
330        PRINT TAB(14);I;TAB(26);Forms(I);TAB(38);Form_vol$[8*(I-1)+1;8]
340     NEXT I
350     PRINT LIN(1),A$;L$;LIN(2);"RECORD 4 through RECORD ";CHR$(132)&VAL$(Kk-1)&
CHR$(128)&" ",LIN(1)
360     PRINT SPA(15);"Item   Item(*)  Type    Required?   Range/Code Pointer"
370     Req$="yes"
380     Temp1$="   "
390     IF Item(1)<0 THEN Temp1$=Req$
400     Type=ABS(Item(1)) MOD 16
410     PRINT USING "16XK4X,6D,6D,7XK12X,6D";"1";Item(1);Type;Temp1$;ABS(Item(1))
DIV 16
420     Code_cnt=NOT Type
430     IF (Type=2) OR (Type=3) OR (Type=4) OR (Type=6) THEN Rangetype(ABS(Item(1)
) DIV 16)=Type
440     FOR I=2 TO Num_items
450        Temp1$="   "
460        IF Item(I)<0 THEN Temp1$=Req$
470        Type=ABS(Item(I)) MOD 16
480        PRINT USING "14X3D4X,6D,6D,7XK12X,6D";I;Item(I);Type;Temp1$;ABS(Item(I)
) DIV 16
```



```
490       Code_cnt=Code_cnt+NOT Type
500       IF (Type=2) OR (Type=3) OR (Type=4) OR (Type=6) THEN Rangetype(ABS(Item
(I)) DIV 16)=Type
510    NEXT I
520    PRINT LIN(1),A$;L$;LIN(2);SPA(13);"Item     Synonym(*)"
530    Prevk=0
540    FOR I=1 TO Num_syns
550       K=NUM(Synonyms$(I))
560       IF K<>Prevk THEN PRINT USING "#,/,14XDDD4X";I
570       IF K=Prevk THEN PRINT SPA(21);
580       PRINT Synonyms$(I)[2]
590       Prevk=K
600    NEXT I
610    PRINT LIN(1),A$;L$;LIN(3);SPA(13);"Code";LIN(1);SPA(12);"Pointer    Code$(*
)";LIN(1)
620    K=1
630    FOR I=1 TO Code_cnt
640       L=NUM(Codes$(K))
650       FOR M=1 TO L
660          IF M=1 THEN PRINT USING "#,14XDDD4X";K
670          IF M<>1 THEN PRINT SPA(21);
680          PRINT Codes$(M+K-1)[2]
690       NEXT M
700       PRINT
710       K=K+L
720    NEXT I
730    PRINT LIN(1),A$;L$;LIN(3);SPA(13);"Range";A$;"Lower";SPA(16);"Upper"
740    PRINT SPA(12);"Pointer";SPA(9);"Bound";SPA(16);"Bound",LIN(1)
750    Rangetype(4)=6
760    FOR I=1 TO Num_ranges
770       Temp$[1,60]=""
780       Temp$[6,26]=VAL$(Range(I,1))
790       Temp$[27]=VAL$(Range(I,2))
800       PRINT USING "14XDDD,K";I;Temp$
810       IF Rangetype(I)<>6 THEN Skip
820       Short=Range(I,1)
830       Temp$[6,26]=FNFormat_date$(1,Short)
840       Short=Range(I,2)
850       Temp$[27]=FNFormat_date$(1,Short)
860       PRINT SPA(17);Temp$
870 Skip:    PRINT
880    NEXT I
890    PRINT LIN(1),A$;L$,LIN(1)
900    PRINT CHR$(132)&"RECORD "&VAL$(Kk)&" through RECORD "&VAL$(L1-1)&CHR$(128)
&" "
910    PRINT LIN(1),SPA(16);"Set  Description$",LIN(1)
920    FOR I=1 TO Num_sets
930       PRINT USING "15XDD,K";I;"  "&Set_desc$(I)
940    NEXT I
950    PRINT LIN(1),A$;L$;LIN(1)
960    PRINT CHR$(132)&"RECORD "&VAL$(L1)&" through RECORD "&VAL$(L1+3)&CHR$(128)
&" "
970    PRINT LIN(1);SPA(13);"The thread information is used internally by QUERY/4
5.";LIN(2);A$;L$,LIN(2)
980    PAUSE
990    END
1000   !
1010   !   ***** THE FOLLOWING SUBPROGRAM DECODES DATE-TYPE DATA ITEMS  *****
1020   !
1030   DEF FNFormat_date$(INTEGER Key,SHORT Dcode)
1040    DIM Char$[2],D$[19],Long$(1:12)[9],Month$[10],Short$(1:12)[4],Year$[4]
1050    INTEGER M,D,Y
1060    DATA January,February,March,April,May,June,July,August,September,October,
November,December,Jan,Feb,Mar,Apr,May,Jun,Jul,Aug,Sept,Oct,Nov,Dec
1070    MAT READ Long$,Short$
1080    CALL Decode_date(Dcode,M,D,Y)
```

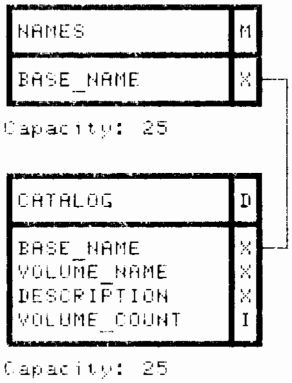


```
1090    D$=""
1100    IF NOT (M OR D OR Y) THEN RETURN D$
1110    IF Key>1 THEN Fd
1120    D$=Long$(M)&" "&VAL$(D)&", "
1130    Year$=VAL$(Y)
1140    RETURN D$&RPT$("0",4-LEN(Year$))&Year$
1150 Fd:        IF (Key-1) MOD 8<=3 THEN Monthfirst
1160    GOSUB Getday
1170    GOSUB Getmonth
1180    GOTO Year
1190 Monthfirst: GOSUB Getmonth
1200    GOSUB Getday
1210 Year:      Year$=VAL$(Y-NOT (Key MOD 2)*100*INT(Y/100))
1220    IF Key MOD 2 THEN Year$=RPT$("0",4-LEN(Year$))&Year$
1230    IF NOT (Key MOD 2) THEN Year$=RPT$("0",2-LEN(Year$))&Year$
1240    RETURN D$&Year$
1250 Getday:   IF Key<5 THEN Char$=", "
1260    IF (Key>4) AND (Key<9) THEN Char$=" "
1270    IF Key<9 THEN Gotchar
1280    IF (Key-9) MOD 4>1 THEN Dash
1290    Char$="/"
1300    GOTO Gotchar
1310 Dash:      Char$="-"
1320 Gotchar: D$=D$&VAL$(D)&Char$
1330    RETURN
1340 Getmonth: IF Key>8 THEN Numeric
1350    IF (Key-1) MOD 4>1 THEN Short
1360    Month$=Long$(M)&" "
1370    GOTO Gotmonth
1380 Short:     Month$=Short$(M)&" "
1390    GOTO Gotmonth
1400 Numeric: IF (Key-9) MOD 4>1 THEN Nslash
1410    Month$=VAL$(M)&"/"
1420    GOTO Gotmonth
1430 Nslash:   Month$=VAL$(M)&"-"
1440 Gotmonth: D$=D$&Month$
1450    RETURN
1460    FNEND
1470    !
1480   SUB Decode_date(SHORT Dcode,INTEGER M,D,Y)
1490      IF (Dcode>-999999) AND (Dcode<=999999) THEN Valid
1500      M=D=Y=0
1510      SUBEXIT
1520 Valid: D_code=Dcode+999999
1530         Y=INT((D_code-1)/365.25)
1540         D_code=D_code-INT(365.25*Y+.75)
1550         M=INT((D_code+31)/30)
1560         Mflag=(M>2) AND (Y/4=INT(Y/4))-(Y/100=INT(Y/100))+((Y/400=INT(Y/400))
OR (Y/1000=INT(Y/1000)))-(Y/4000=INT(Y/4000))
1570         IF INT(30.55*M-29.95)-2*(M>2)+Mflag>=D_code THEN M=M-1
1580         D=D_code-INT(30.55*M-29.95)+(2-Mflag)*(M>2)
1590         SUBEXIT
1600         SUBEND
```

Appendix C
The "ICAT" Catalog Data Base

Associated with QUERY/45 is a small data base, called ICAT, that contains information about the data bases that QUERY/45 can access. This data base is maintained by QUERY/45 and is used when the SHOW BASES softkey is pressed at the beginning of the program. For each data base, ICAT contains the data base name, the data base description, the volume name, how many volumes the data base spans. A total of 25 data bases can be stored in ICAT.

This data base stores each data base name in a automatic master data set and stores the rest of the data in a detail data set. Following is the diagram for the ICAT data base:



ICAT is stored on the same volume as QUERY/45 so that it is always available. It is automatically updated by the DEFINE subsystem whenever a new data base is created. You can also manually add, delete, or modify entries.

Why Update ICAT Manually?

If you purge a data base, you would want to delete the entry stored in ICAT.

Another reason that you might need to manually update ICAT is when you have created a data base using IMAGE/45 and now you want to do searches using QUERY/45. You can add the new data base to the ICAT data base so that QUERY/45 automatically recognizes that data base and knows which volume it is mounted on.

Manually Updating the ICAT Data Base

To update the ICAT data base, press the EDIT BASE CATALOG softkey at the beginning of QUERY/45. This causes ICAT to be opened and the UPDATE subsystem to be loaded. You now have the options of adding, deleting, modifying, or replacing entries containing information about the data bases. The updating procedure is the same as when using the UPDATE subsystem with one of your data bases.

Press the EXIT softkey to return to the beginning of QUERY/45.

Appendix D
Tables

ASCII Table

ASCII Char.	EQUIVALENT FORMS			ASCII Char.	EQUIVALENT FORMS			ASCII Char.	EQUIVALENT FORMS			ASCII Char.	EQUIVALENT FORMS		
	Binary	Octal	Dec		Binary	Octal	Dec		Binary	Octal	Oec		Binary	Octal	Dec
NUL	00000000	000	0	space	00100000	040	32	@	01000000	100	64	`	01100000	140	96
SOH	00000001	001	1	!	00100001	041	33	A	01000001	101	65	a	01100001	141	97
STX	00000010	002	2	"	00100010	042	34	B	01000010	102	66	b	01100010	142	98
ETX	00000011	003	3	#	00100011	043	35	C	01000011	103	67	c	01100011	143	99
EOT	00000100	004	4	$	00100100	044	36	D	01000100	104	68	d	01100100	144	100
ENO	00000101	005	5	%	00100101	045	37	E	01000101	105	69	e	01100101	145	101
ACK	00000110	006	6	&	00100110	046	38	F	01000110	106	70	f	01100110	146	102
BEL	00000111	007	7	'	00100111	047	39	G	01000111	107	71	g	01100111	147	103
BS	00001000	010	8	(	00101000	050	40	H	01001000	110	72	h	01101000	150	104
HT	00001001	011	9	)	00101001	051	41	I	01001001	111	73	i	01101001	151	105
LF	00001010	012	10	*	00101010	052	42	J	01001010	112	74	j	01101010	152	106
VT	00001011	013	11	+	00101011	053	43	K	01001011	113	75	k	01101011	153	107
FF	00001100	014	12	,	00101100	054	44	L	01001100	114	76	l	01101100	154	108
CR	00001101	015	13	–	00101101	055	45	M	01001101	115	77	m	01101101	155	109
SO	00001110	016	14	.	00101110	056	46	N	01001110	116	78	n	01101110	156	110
SI	00001111	017	15	/	00101111	057	47	O	01001111	117	79	o	01101111	157	111
DLE	00010000	020	16	ø	00110000	060	48	P	01010000	120	80	p	01110000	160	112
DC$_1$	00010001	021	17	1	00110001	061	49	Q	01010001	121	81	q	01110001	161	113
DC$_2$	00010010	022	18	2	00110010	062	50	R	01010010	122	82	r	01110010	162	114
DC$_3$	00010011	023	19	3	00110011	063	51	S	01010011	123	83	s	01110011	163	115
DC$_4$	00010100	024	20	4	00110100	064	52	T	01010100	124	84	t	01110100	164	116
NAK	00010101	025	21	5	00110101	065	53	U	01010101	125	85	u	01110101	165	117
SYN	00010110	026	22	6	00110110	066	54	V	01010110	126	86	v	01110110	166	118
ETB	00010111	027	23	7	00110111	067	55	W	01010111	127	87	w	01110111	167	119
CAN	00011000	030	24	8	00111000	070	56	X	01011000	130	88	x	01111000	170	120
EM	00011001	031	25	9	00111001	071	57	Y	01011001	131	89	y	01111001	171	121
SUB	00011010	032	26	:	00111010	072	58	Z	01011010	132	90	z	01111010	172	122
ESC	00011011	033	27	;	00111011	073	59	[	01011011	133	91	{	01111011	173	123
FS	00011100	034	28	<	00111100	074	60	\	01011100	134	92	|	01111100	174	124
GS	00011101	035	29	=	00111101	075	61	]	01011101	135	93	}	01111101	175	125
RS	00011110	036	30	>	00111110	076	62	^	01011110	136	94	~	01111110	176	126
US	00011111	037	31	?	00111111	077	63	_	01011111	137	95	DEL	01111111	177	127

Prime Number Table

Here are the prime numbers up to 1259. The following program can be used to generate all prime numbers up to 32 767.

2 3 5 7 11 13 17 19 23 29 31 37 41 43 47 53 59 61

67 71 73 79 83 89 97 101 103 107 109 113 127 131 137 139 149

151 157 163 167 173 179 181 191 193 197 199 211 223 227 229 233 239

241 251 257 263 269 271 277 281 283 293 307 311 313 317 331 337 347

349 353 359 367 373 379 383 389 397 401 409 419 421 431 433 439 443

449 457 461 463 467 479 487 491 499 503 509 521 523 541 547 557 563

569 571 577 587 593 599 601 607 613 617 619 631 641 643 647 653 659

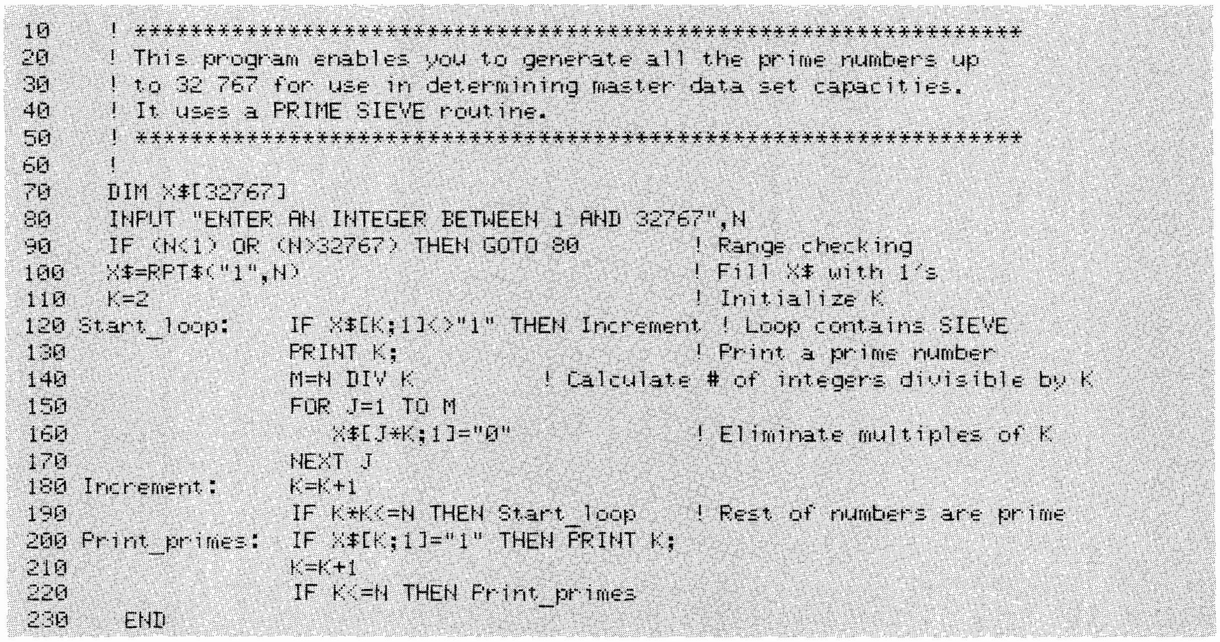
661 673 677 683 691 701 709 719 727 733 739 743 751 757 761 769 773

787 797 809 811 821 823 827 829 839 853 857 859 863 877 881 883 887

907 911 919 929 937 941 947 953 967 971 977 983 991 997 1009 1013 1019

1021 1031 1033 1039 1049 1051 1061 1063 1069 1087 1091 1093 1097 1103 1109 1117 1123

1129 1151 1153 1163 1171 1181 1187 1193 1201 1213 1217 1223 1229 1231 1237 1249 1259



```
10    ! ****************************************************************
20    ! This program enables you to generate all the prime numbers up
30    ! to 32 767 for use in determining master data set capacities.
40    ! It uses a PRIME SIEVE routine.
50    ! ****************************************************************
60    !
70    DIM X$[32767]
80    INPUT "ENTER AN INTEGER BETWEEN 1 AND 32767",N
90    IF (N<1) OR (N>32767) THEN GOTO 80          ! Range checking
100   X$=RPT$("1",N)                              ! Fill X$ with 1´s
110   K=2                                         ! Initialize K
120 Start_loop:     IF X$[K;1]<>"1" THEN Increment ! Loop contains SIEVE
130                 PRINT K;                      ! Print a prime number
140                 M=N DIV K          ! Calculate # of integers divisible by K
150                 FOR J=1 TO M
160                    X$[J*K;1]="0"              ! Eliminate multiples of K
170                 NEXT J
180 Increment:      K=K+1
190                 IF K*K<=N THEN Start_loop     ! Rest of numbers are prime
200 Print_primes:   IF X$[K;1]="1" THEN PRINT K;
210                 K=K+1
220                 IF K<=N THEN Print_primes
230   END
```

Subject Index

QUERY / 45 User's Guide

HEWLETT
PACKARD

Part No. 09845-91056
Microfiche No. 09845-98056

Printed in U.S.A.
Second Edition, Mar. 1981