MASTERING dBASE III PLUS™

A STRUCTURED APPROACH

Carl Townsend

MASTERING dBASE III PLUS™:

A STRUCTURED APPROACH

CARL TOWNSEND

SYBEX®

SAN FRANCISCO • PARIS • DÜSSELDORF • LONDON

Cover design by Thomas Ingalls + Associates
Book design by Ingrid Owen

Library of Congress Card Number: 86-61482
ISBN 0-89588-372-4
Manufactured in the United States of America
10 9 8 7 6 5 4 3

ACKNOWLEDGMENTS

Grateful acknowledgment is expressed to Ashton-Tate for supplying the copies of dBASE III PLUS and Framework used to prepare this book and to the many add-on manufacturers who supplied evaluation copies of their products for review. I would also like to thank the people at SYBEX who worked on this book: Carol Alden and Karl Ray for acquisitions and management; Susan Weisberg, Bonnie Gruen, and Jeremy Elliot for editing and technical support; Olivia Shinomoto and David Clark, word processing; Donna Scanlon, typesetting; and Jon Strickland, proofreading. A special thanks is due to Deborah Hastings and Nancy Morrice at The Columbia Group, my agent, for helping with the manuscript. I would also like to thank Mike Maurice for the bounce-bar menu listing in Chapter 9 and the many other developers who have helped me in learning both dBASE II and dBASE III PLUS. Thanks also to Marcos Cesar at Datalogica and my other friends in Sao Paulo for their help in correcting *Mastering dBASE III*. Finally, I would like to offer a special thanks to all of my friends who read *Using dBASE II* and patiently waited and encouraged me as *Mastering dBASE III PLUS* was written.

TABLE OF CONTENTS

MAKING THE DECISION

Bill Weigel ran a small business from his home doing computer consulting for local businesses. Bill could locate almost any type of software for a user and help a businessperson purchase the type of software or hardware needed for a particular application. Bill's wife, Sue, did the accounting for their business, and both enjoyed their cottage industry. Eventually, however, the books got out of hand. Sue found it a full-time job to keep the bookkeeping up-to-date, so Bill started looking for an accounting program he could run on his IBM personal computer.

TO PROGRAM OR NOT TO PROGRAM

The salesperson at Bill's local computer store showed him several general ledger programs. None of them were too expensive, so Bill was able to borrow a few of the manuals for a few days to study the products more closely. Although most of the products appeared to meet most of his needs, he did note several deficiencies:

1. Almost all the products did not provide the original code the program was written in; Bill would essentially be purchasing a "black box." If he wanted to add features or make changes later, he would not be able to do this without the original code. In addition, Bill did not like the idea of trusting his records to an unknown system.
2. Some of the features Bill wanted, such as being able to compare current account status with a budget, were not available on many of the systems.
3. Many of the systems could not be used with other software he was using, such as spreadsheets, word processors, and graphic utilities.
4. Recovery procedures from a bad posting or closing were often inadequate, complex, or time consuming.
5. Bill had serious questions about how long a closing or posting would take, as the manuals did not give many specifications about this.
6. Sue liked to do the accounting a specific way, and did not want to change her way of accounting to accommodate somebody else's program. All the programs he looked at used a fairly rigid system.

There were some advantages in purchasing one of the commercial products, and Bill identified these as well:

1. The cost of the commercial product would be less than the cost of his time in developing his own software.
2. He would be purchasing a proven product with a large user base.
3. He would be able to start his bookkeeping tomorrow. If he wrote his own software, it would be several weeks before he could start using it.

After Bill finished his research, he made the decision to write his own software. Sue had had considerable accounting experience, and Bill felt they had enough knowledge in this area to define what was needed. Bill decided to use dBASE III PLUS for his general ledger system. His development costs would not be much higher than with a commercial product, and he could design his reports exactly as he wished them for his own work and that of his CPA. It would not take very long to get a prototype of the system operational for Sue to use. The tremendous flexibility of dBASE III PLUS made it easy to create any special reports the CPA would need. He could also interface the data to spreadsheets and word processors, such as the Framework program he had already purchased. Bill solved his problem with a database manager called dBASE III PLUS.

WHY USE DATABASE MANAGEMENT?

A *database management system,* or DBMS, is a group or collection of programs that gives the user access to a collection of information stored as data. This collection, or pool, of information is called the *database.* Typical applications for a database management system might include:

Financial—payroll, general ledger, accounts receivable, accounts payable

Inventory Control—inventory, material requirement planning, purchase order systems, sales order systems

Business—real estate management, project management, medical data systems, sales prospect control, cost accounting, etc.

Home data management—cataloging of records, books and tapes, nutrition analysis, mailing lists

In some types of database managers, the user writes the application programs in BASIC, C, Pascal, FORTRAN, or another high-level language. Other database managers, such as dBASE III PLUS, use a language that is part of the database management system, and no additional programming language is needed.

Database managers help reduce the duplication of data that often occurs when a system grows to meet many needs. For example, the name and address that were saved in a payroll file occur again in a company's medical records file, and again in the credit union file. There is a chance of error each time the name and address are entered. In addition, the duplication of data wastes disk space. With a database management system the name and address are stored only once.

Database managers also minimize program development time. Without database management the programmer must keep track of how and where

data is stored. The programmer has the responsibility of writing programs that locate the desired record, often using complex index routines. Every time a report program is written, routines must be included to extract the desired records, sort them to a desired order, and then print the report. A database manager, on the other hand, keeps track of how and where the data is stored, permitting the programmer to concentrate on the application and reducing programming time. When Bill wrote his general ledger in dBASE III PLUS, he found it took him only 10-25 percent of the time it would have taken to write the same program in BASIC or Pascal.

Database managers mean improved program reliability. Programs are shorter and can be developed with fewer errors. Integration of the data flow is automatic. Bill found his programs often ran the first time correctly, and most programs were only one or two pages long. If he had used BASIC, C, or Pascal, he would have spent many hours chasing problems in programs that were many pages in length.

Database managers also mean improved data reliability since they can prevent an incorrect type of entry. For example, if the department code is always a number, any nonnumeric entry could be rejected.

TYPES OF DATABASE MANAGEMENT SYSTEMS

There are several types of tools available today for information management. Many of these could most accurately be called indexing systems or file management systems.

The simplest and most inexpensive tools are the indexing systems designed to work with existing high-level languages (such as C). These products make it possible for a programmer to add indexing capability to a program that formerly had only sequential or random access (see Chapter 8 for more on this). One popular example is the Access Manager sold by Digital Research. Indexing systems are all inexpensive products, but they require programming expertise to use.

On the next level are the multifile managers. These vary in cost, some less expensive than many indexing systems. Many file managers include some type of indexing system. They also include extra features, varying with the particular product, such as report generators and routines to add or update data in the files. Most of these products can generally be used by someone with very little computer experience. PFS:File is an example of this type of product.

Finally, there are the true database management systems. Most of the better database management systems are one of three types: hierarchical, network, or relational. The hierarchical is a special case of the network system,