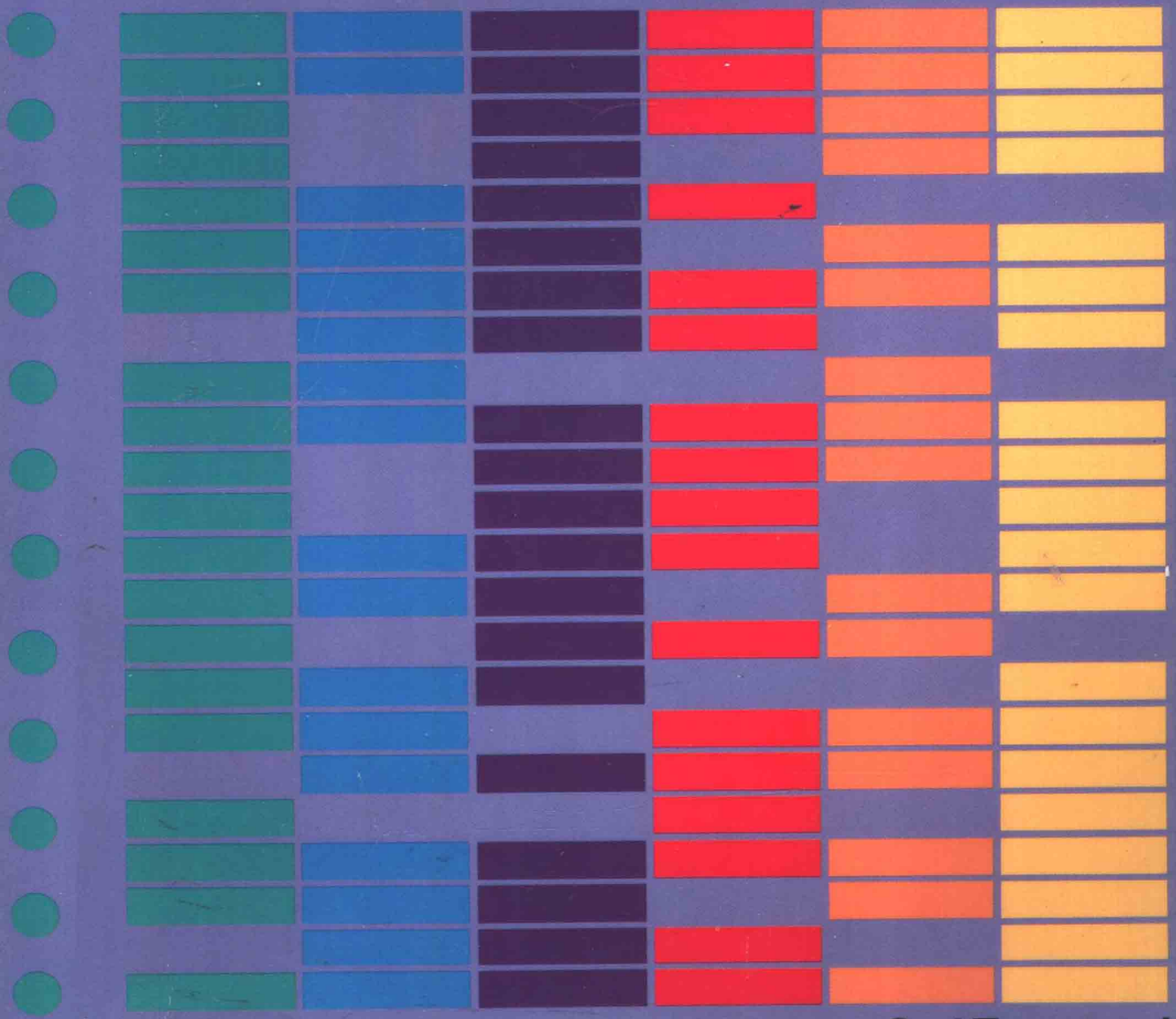
Osborne/McGraw-Hill

USING dBASE II®



Carl Townsend

USING dBASE II®

Carl Townsend

Osborne/McGraw-Hill
Berkeley, California

Published by
Osborne/McGraw-Hill
2600 Tenth Street
Berkeley, California 94710
U.S.A.

For information on translations and book distributors outside of the U.S.A., please write to Osborne/McGraw-Hill at the above address.

USING dBASE II®

34567890 DODO 8987654

ISBN 0-88134-108-8

Paul Hoffman, Technical Editor
Ted Gartner, Copy Editor
Richard Cash, Text Design
Yashi Okita, Cover Design
Jan Benes, Technical Illustrations

ACKNOWLEDGMENTS

Grateful acknowledgment is expressed to Ashton-Tate for their support and a beta-test copy of dBASE; to Climax Manufacturing; and Bob Harkema at Climax for his advice and development of the TeleVideo interface software in Chapter 11.

PREFACE

During the last few years there has been a rapid growth of a new type of consultant. This consultant works with microcomputer users and often has a background in mainframe and minicomputers, as well as some knowledge of systems analysis. Using new microcomputer tools like dBASE II, this consultant is able to design custom software packages at very nearly the cost of many existing commercial packages for a wide variety of applications. *Using dBASE II®* is designed to help such a consultant to begin to use one of the most popular of these program tools—dBASE II.

Using dBASE II® gives you an introduction to dBASE II as well as insights to the complete systems design process. Examples and listings give you the nucleus of programs that can be used in a wide variety of applications. Example systems used include programs for inventory and legal job costing. The book is an encyclopedia of my experiences with dBASE II over two years. The book stresses *structured design* and emphasizes approaching programming in a structured manner using top-down analysis.

C. T.

CONTENTS

ONE

Introduction to dBASE II

Fred's Friendly Automotive repaired cars. Everyone knew Fred was the best Volvo repairman in town, and most of his repair work involved Volvos and a few other foreign cars. He kept a large inventory of new and used parts. He knew that with labor costs at $30 per hour, he needed to be able to locate parts quickly and order parts as necessary to keep the inventory current. Fred knew he needed a computer with some type of inventory system, so he visited his local computer store.

"We do not have any programs specifically for automotive repair stores," the salesperson told him. "We have Superinventory, but it is really

a software package for a *manufacturer* with an inventory. For your application, you would need to get a BASIC language interpreter and write your own program. In fact, you would probably need to write several programs. As an alternative, you could purchase a program that would meet *some* of your needs."

Fred was not a computer expert; in fact, he had never used a computer. The idea of writing his own programs to process and report his automotive parts inventory sounded like an impossible job. In addition, Fred had very little time in his own schedule to learn to do programming. He left the computer store empty-handed.

A few weeks later Bill Braxton, a local computer programmer, drove his Volvo in to get a new alternator. Fred knew he had the alternator in stock, but he could not find it. "They might have put it on that Volvo they repaired a few days ago," said Fred as he began looking through the invoices to find the one that might indicate where the missing alternator had been installed. Bill told Fred he needed a microcomputer and Fred laughed.

"You know it, I know it, but who is going to program it?" he retorted.

"Look," Bill said, "I'll write a program to process your inventory and rent you my portable computer for a day for $100. I'll show you how to load the inventory into the computer, and later I'll pick it up and print it on my printer at home. If you like it, I can show you how to expand the program to meet your specific needs."

"You've got a deal," said Fred. "When do we start?"

Two days later Bill dropped off the portable computer and gave the secretary a lesson on how to enter the inventory information. Bill printed the listing on his home computer. A few days later Fred bought a computer and Bill showed him how to get information on the inventory status and how to update the quantities on hand as parts were added to or taken from the inventory. Soon Bill added a mailing list program, and shortly thereafter Fred was mailing advertising to all of his customers. Fred's total cost for his custom software was just under $1000, about the same cost as the inventory software packages he had seen that did not meet his needs. Bill wrote all of the programs in a new language developed specifically for microcomputers—dBASE II.

dBASE II is marketed as a relational database management system. A relational database system is one that stores information in two-dimensional tables in which data can be accessed by the relationship of information in the tables. More accurately, dBASE II is a file manage-

ment system with an advanced programming language. The language is very easy to learn and is "structured," making it possible for programmers to develop programs much faster than by using other high-level languages like BASIC or Fortran. Originally developed for the Viking Mars Lander Project, this programming language is now available for almost any CP/M system.

The Database Manager

A database management system, or DBMS, is essentially a group or collection of programs that connects the user to one or more collections of information. The collection of information, or pool, is called a database (see Figure 1-1). Program applications for a DBMS include general ledgers, inventory control, accounts receivable, inquiry and mailing list systems, cataloging, order entry systems, bibliographies—essentially any application in which data files are built, updated, analyzed, and reported.

With most database managers, the application programs are written in BASIC, PASCAL, Fortran or other high-level languages, and special calls are used to the database manager. dBASE II, however, uses an internal language so that no other language is needed.

Advantages of Database Management

Database managers prevent the duplication of data. As a system grows, information in one file soon appears in other files. Names, addresses, and other similar information begin to appear in several files even though each file contains identical information. Someone must enter this duplicate data into each file, creating a typing backlog and increasing the chance of typing errors. Extra disk storage is used, and changes are not always made to each file. Database managers, by storing everything in a single database, eliminate this problem.

Database managers also reduce program development time. Much of the routine programming work of file management, indexing, sorting, and

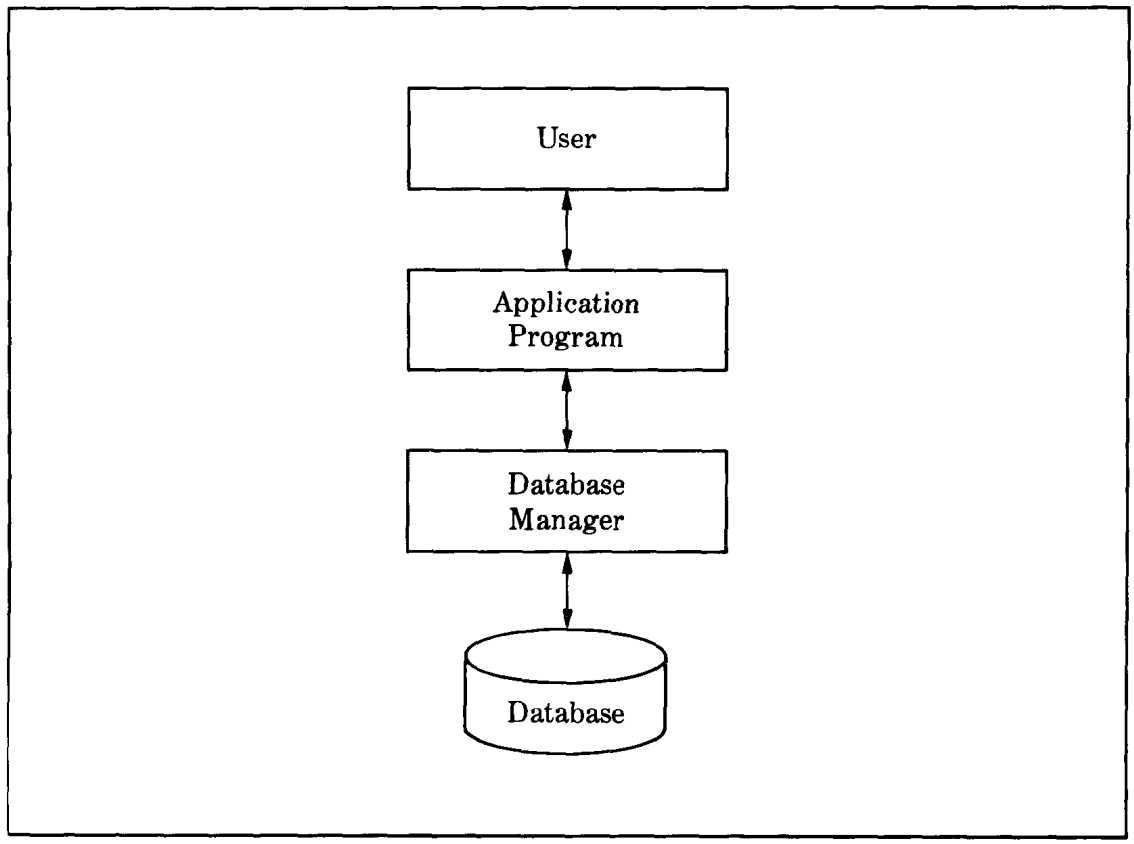


Figure 1-1. The Database manager

report generation is done automatically with simple calls in a database manager. Programs can easily evolve to meet the needs of the user.

Database managers improve data reliability. The integration of information and the relationship of information within a database is done automatically by the database manager. This relieves the programmer of the need to use pointers and pointer chains to locate needed information. Data reliability is improved. For instance, if sex is always "M" or "F," the database manager can prevent any other type of entry. If the ZIP code is always a number, any other entry could be rejected.

Types of Database Management Systems

If you are interested in purchasing a database management system for your microcomputer, you will find a very small number of true database management systems available. As shown in Figure 1-2, most of the tools available to the programmer in this area are, more accurately, indexing systems or file management systems.

The simplest (and least expensive) level of systems contains the index-

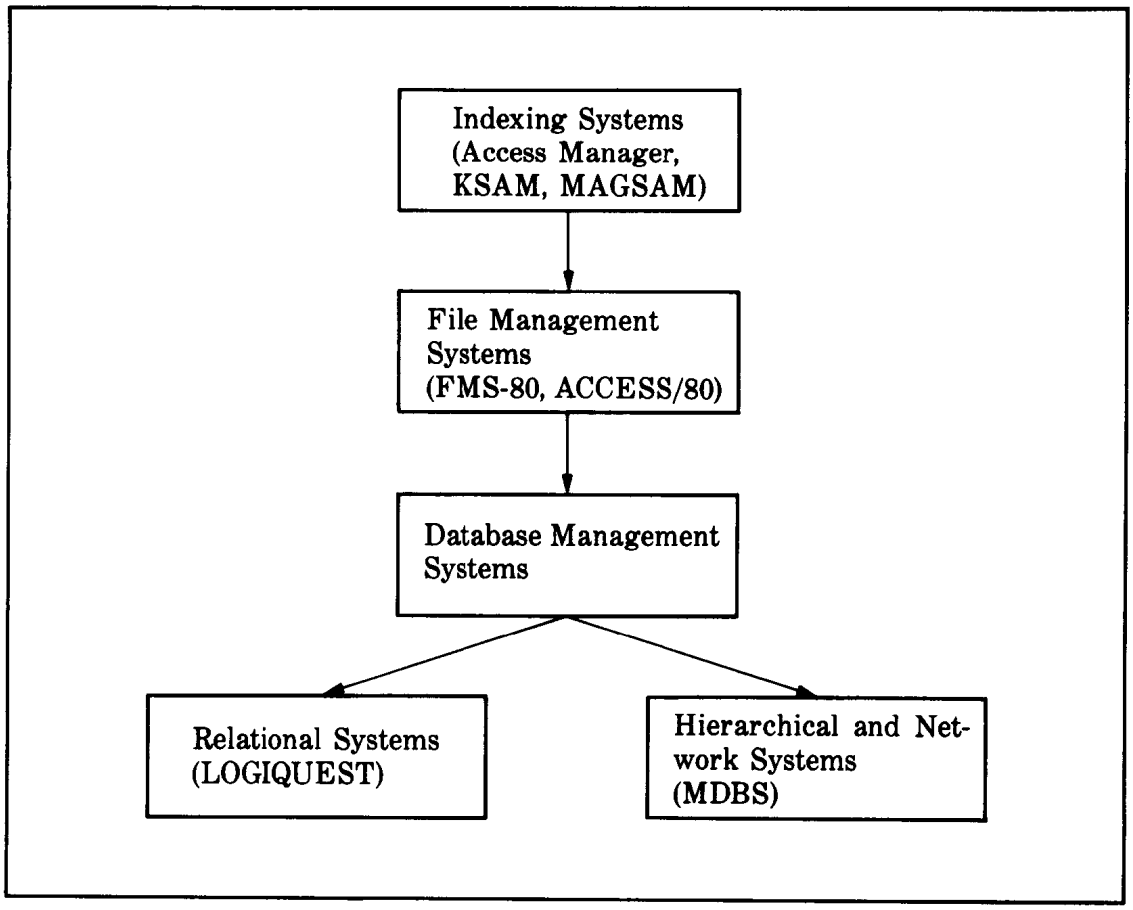


Figure 1-2. Types of information management systems

ing systems. These products are designed to work with existing high-level languages (like BASIC) to simplify the indexing process. These products make it possible for a programmer to add indexing capabilities to a program that formerly had only sequential or random-access capabilities (see Chapter 3 for more on this). However, programming must still be done in the higher-level language. Examples of these products include MAGSAM and ACCESS MANAGER. These are all inexpensive products, but they do require programming expertise to use.

At the next level are the multi-file managers. These are generally more expensive than indexing systems and often include some type of internal indexing system. Multi-file managers also include extra features that vary with the product. Examples of these extra features include report generators and routines to add or update data in the files. Many of these products can be used by someone with very little computer experience. Examples of these products include FMS-80 and ACCESS/80.

At the highest level are the true database management systems. In a "perfect" database management system the storage and updating of the data is invisible to the user. All the data is stored in a single "sink" or database. The user is only concerned about *information flow*. As the system design is altered, very little change is necessary for existing programs. Although this ideal is never reached, it is approximated to various degrees in the better systems. All the better database management systems fall into three categories: hierarchical, network, or relational. Since the hierarchical category is a special case of network systems, there are really only two types. Although a more complete discussion of the network and relational database system concepts are beyond the scope of this book, a brief overview is important in understanding dBASE II.

In a hierarchical or network system, information is stored in a structure that looks very much like an inverted tree (see Figure 1-3). This tree could represent an inventory system for Fred's Friendly Automotive. The car is composed of various assemblies, and each assembly is composed of various subassemblies. The subassemblies in turn are composed of parts. Each part is a "child" of the "parent" subassembly or assembly that "owns" it. In a hierarchical system no child can have more than one parent. A network system is similar, except that a child *can* have more than one parent (see Figure 1-4).

A relational system is quite different in organizational structure. Information is stored in two-dimensional tables that are more conveniently called *files*. Information in the tables is accessed by the user based on any

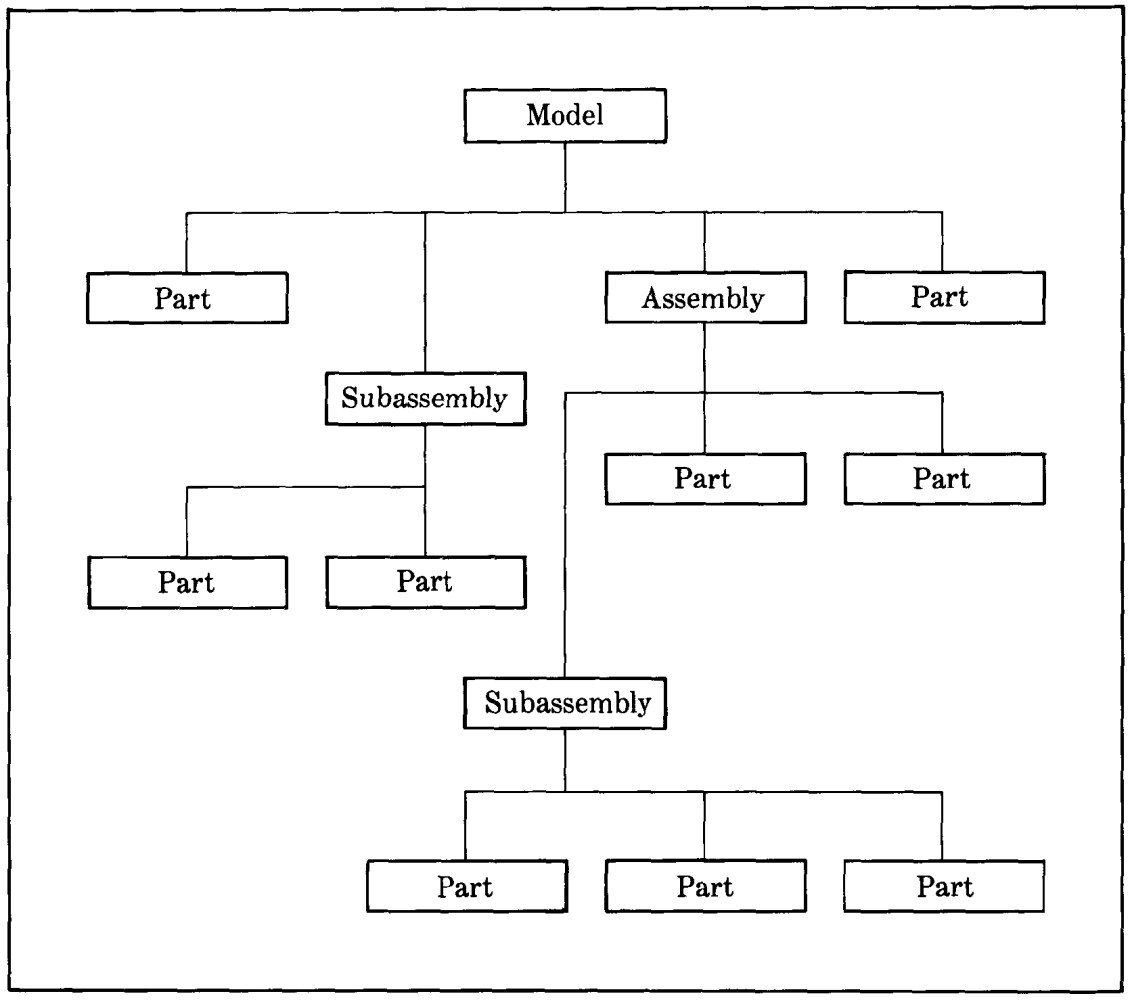


Figure 1-3. Hierarchical data structure

desired relationship. The differences between network and relational databases are shown in Table 1-1.

The inventory system Bill designed for Fred's Automotive consisted of three files for information storage. One file contained all the inventory with the part numbers, part descriptions, quantity on hand, cost, and price for each item. The second file contained the names, addresses, and an identifying number for each customer. A third file was the invoice file, which showed the invoice number and the labor and itemized part charges

Table 1-1. Differences Between Network and Relational Systems

Hierarchical and Network Database	Relational Database
Relationships between items in the database are stored physically in the database.	Relationships between items are not stored in the database and are created logically rather than physically.
Complex relationships of items that are a physical part of the database can be created.	Easier to understand and use.
Database files are not easily alterable to new physical relationships.	Database files can be easily altered to fit new situations.
Good machine performance if processor size and memory size are adequate.	Machine performance varies depending on how the application is implemented.
Large amounts of memory and secondary data storage space are needed.	Utilizes relatively little memory and secondary disk storage.
Extra space is necessary to store all the relational information.	Relational databases waste space by storing the maximum space for each field whether it is needed or not.

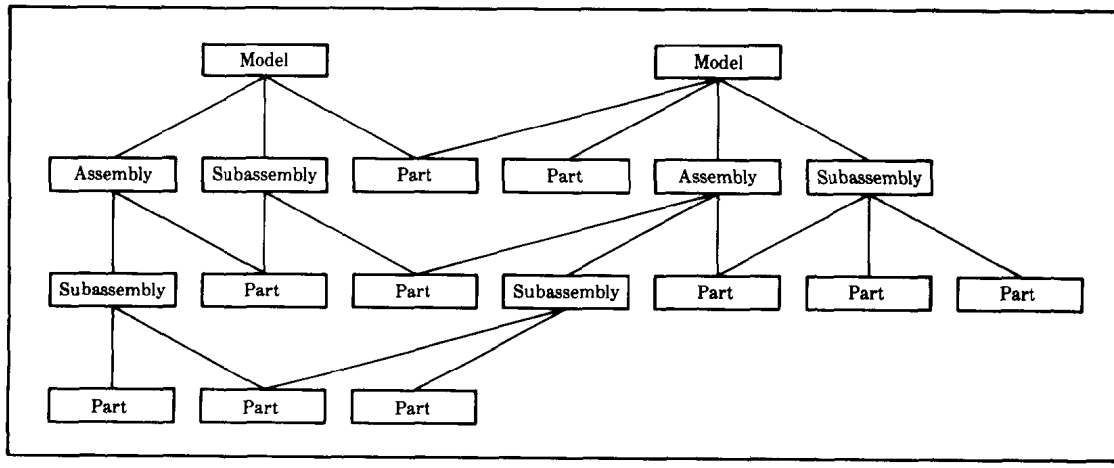


Figure 1-4. Example of network data structure

for each customer (these files are shown in Figure 1-5). Since dBASE II is a relational system, it was ideal for Fred's application. If Fred had wanted to keep the information about the relationship of parts to subassemblies and assemblies, he could still have used dBASE II. In Chapter 15 we show you how this is done.

A true DBMS has the following features:

- Program and data independence. The structure of the data files can be changed without changing the program. Programs can be changed without changing the data file structures.
- Information (and data) is stored (as seen by the user) as a single collection or file.
- The data can be used by a large collection of application programs with security control on the information to prevent unauthorized access by users who should not have access. Salaries of employees, for example, should not be accessible by someone interested only in employees' addresses.
- Tools for sorting, indexing, and reporting data in the database are provided to minimize programming development.

The History of dBASE II

The story of dBASE II is as fascinating as its capabilities. The story began in the space-age setting of the Jet Propulsion Laboratory (JPL) near Pasadena, California.

In 1974, scientists and engineers at JPL began using an advanced type of database management system to analyze information from many of JPL's unmanned space probes. The system, known as JPLDIS (JPL Data-Management and Information-Retrieval System) was written by Jeb Long.

Wayne Ratliff was a systems designer at the laboratory and wanted a similar system to run on his home microcomputer system. Ratliff was fascinated not only by the control of these very "intelligent" unmanned space probes, but also by the whole field of artificial intelligence. He began writing some artificial intelligence software using JPLDIS as a model. In 1980,

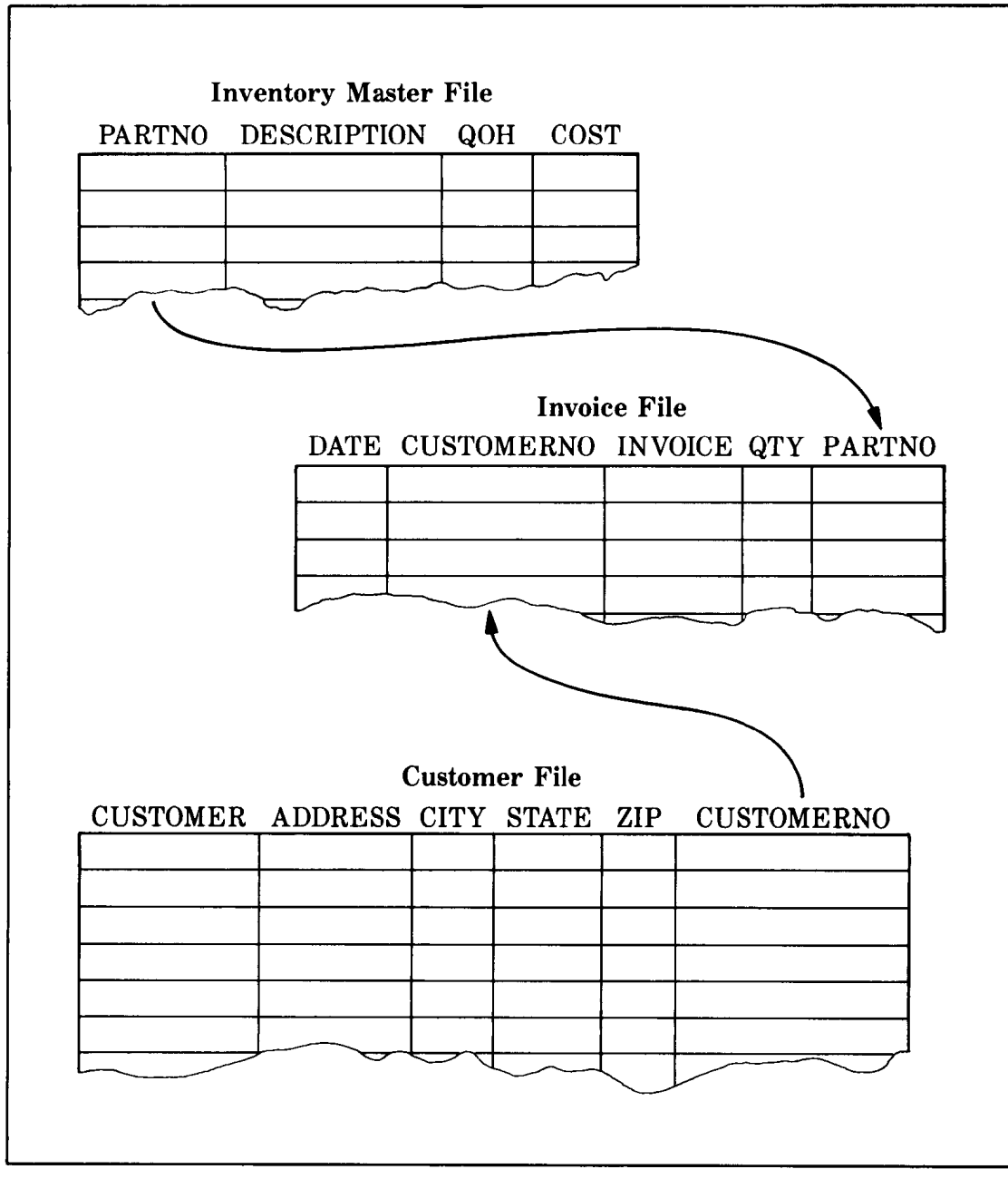


Figure 1-5. Relation between files in Fred's Automotive system

his program had become more powerful than JPLDIS. Jeb Long was impressed.

In 1979, Ratliff began advertising his program as VULCAN in the back pages of *BYTE* magazine. The early version lacked many of the features of its competition in 1980. There was no indexing feature, and records could only be located by scanning the entire file. Fancy menus, screen displays, and reports that are easy to create with dBASE II were not available in the early VULCAN version. In August of 1980, Ratliff released an advanced version of VULCAN with indexing and screen-generating features. The product now exceeded the capabilities of many of its competitors, but sales were slow. No more than fifty VULCANs were ever sold.

During the summer of 1980, George Tate began to hear about VULCAN. Tate had been a major software distributor for years, and he tried the VULCAN system. He too was impressed. When he learned that only fifty copies had been sold, he told Ratliff he could sell fifty a month. A contract was soon signed guaranteeing Ratliff generous royalties. The VULCAN name was dropped and the product was called dBASE II. There never was a dBASE I—this was simply a marketing ploy to imply a new, improved version. Several features were added to the documentation.

Then George Tate flooded the market with strange ads comparing dBASE II to a bilge pump. The database competitors did not like the advertisement and neither did the bilge pump manufacturer, but the advertisement worked. A separate company (Ashton-Tate) was formed to market the product. Over 2000 copies of dBASE II are sold a month now, and Ratliff is semi-retired from JPL.

dBASE II Concepts

dBASE II is marketed and sold as a relational database management system. Since several files are generally used in a given application and the relationship between information in different files is not stored in the system, dBASE II is not a true database management system in the strictest sense of the word. dBASE II is more like a file management system