Introduction to Data Base Management in Business

JAMES BRADLEY
University of Calgary

Introduction to Data Base Management in Business

JAMES BRADLEY
University of Calgary

HOLT, RINEHART AND WINSTON
New York Chicago San Francisco Philadelphia
Montreal Toronto London Sydney
Tokyo Mexico City Rio de Janeiro Madrid
HOLT-SAUNDERS JAPAN

Library of Congress Cataloging in Publication Data

Bradley, James, 1942–
Introduction to data base management in business.

Bibliography: p.
Includes index.
1. Business—Data processing. 2. Data base management.
I. Title.
HF5548.2.B678 1983 658'.054 82–23302

ISBN 0-03-061693-X (US College Edition)
ISBN 4-8337-0157-X (Holt-Saunders International Edition)

Address correspondence to:
383 Madison Avenue
New York, New York 10017

Printed in Japan, 1983

3 4 5 039 9 8 7 6 5 4 3 2 1

CBS COLLEGE PUBLISHING
Holt, Rinehart & Winston
The Dryden Press
Saunders College Publishing

Preface

Information has always been important in business decisions. With advancing technology however, and with the resulting increase in the complexity of business organizations, the quality and variety of information needed for business and management decisions is steadily increasing. Since technology will undoubtedly continue to advance, the need for information at all levels in business organizations can only grow. We have thus entered an age where it is the superior use of information that will determine the success of corporations and the wealth of nations. This information is increasingly to be found in data bases. Data base management is thus important, and is becoming the foundation for all data processing and a great deal more. It is also exciting, and every business student graduating in the 1980s should take a course in it. This book is written for just such a course.

Inevitably a wide variety of business students will be required to study data base management, and so the book begins at an elementary level and makes few assumptions about the computer skills of the students. In later sections, however, there is some material that may well challenge the most gifted honors student.

The book is also designed for self-study, and so minimizes the burden on the instructor. It covers the full spectrum of the subject nevertheless, and its design permits it to be used both with short introductory courses and with more intensive two-semester courses. Thus the book is suitable both for those who are aiming only at a sound working knowledge of the subject, as well as for those aiming at developing considerable expertise in data base management.

Many years of experience with data base management have convinced me that the subject is more difficult to teach than most. Some reasons for this are the wide scope of the subject, the existence of different approaches to data base management, and a lack of any standard terminology. Indeed, it often appears that no vendor of a data base management system is satisfied unless his or her terminology is distinctly unique. Another problem for an instructor is the common lack of a good selection of data base systems on the campus or departmental computer.

Every effort has been made to remove these difficulties. Throughout the book I have used a terminology that is both consistent and as near to common usage as possible. However, the special terms used with specific data base systems are also introduced and explained. In addition, a common diagrammatic method is used with all approaches to data base management. Finally, in order to help students concentrate on the differences among the major approaches to data base management, I have restricted the number of data bases used to just two. One data base is about components used in manufacturing, while the other is about the banking business. Included as well are some minor variations on the theme of these two data bases. Examples of additional data bases can be found in the instructor manual.

The problem of availability of data base systems is difficult both for author and instructor. For obvious reasons the book is restricted to only the most commonly used systems, and I have tried to facilitate the instructor who has access to some of them. With each data base example in either the text or the instructor manual, I have therefore included sufficient detail to enable instructors or students to implement the data base on their computer with a minimum of effort. To give complete data base definitions would not be possible given the wide variety of operating systems and machines on which data base systems can be installed. Note that since IBM operating systems dominate in the commercial world, with many examples I have tacitly assumed an IBM operating system environment—although I have kept the accompanying details to a minimum.

Even if no data base system is available for student use, an effective course can still be given using this book. For each system described, the organization of the material and the examples of data base manipulation are carefully designed to permit students to learn in the absence of access to a working system. If access is possible, so much the better.

Most of the later chapters have an accompanying supplement or additional perspective. These additional perspectives contain supplementary material that is less important, more advanced, or simply just emerging. They can all be skipped without loss of continuity. They are included to widen the perspectives of both student and teacher, and to be used in more advanced courses. Note that in most cases the material in these additional perspectives is not more difficult than in other parts of the book, and should be used by instructors when it suits their purposes.

The accompanying instructor manual contains useful suggestions on how to design a course in data base management, and how to tackle specific topics. In addition it contains a test bank of some 500 questions with answers, together with answers to selected questions in the text. Furthermore, the manual contains most of the diagrams and tables used in the text, but in a form suitable for making transparencies. Additional data bases and data base definitions are also to be found in the manual.

I am grateful to the many people who helped in the preparation of this book. In particular, Ken Howard at the University of Victoria's Computer Center must be thanked for the long hours spent at a terminal helping me with the intricacies of ADABAS. I must also thank Professors David W. Chilson, F. Paul Fuhs, Patrick Lamont, and Daniel R. Rota for their painstaking reviews of the manuscript and their many excellent suggestions that were incorporated in the text. An acknowledgment is also warranted to MIT's Paul A. Samuelson, from whom I borrowed the idea of placing the additional perspectives at the ends of many of the chapters.

James Bradley

For the sake of persons of different types, scientific truth should be presented in different forms, and should be regarded as equally scientific, whether it appears in the robust form and the vivid coloring of a physical illustration, or in the tenuity and paleness of a symbolic expression.

James Clerk Maxwell
Address to the British Association for the Advancement of Science, 1870.

Contents

I

Foundations

This book is about the management of data in large enterprises, mostly by means of a fairly new tool called a *data base management system*. As such the book is about a fairly specialized branch of management, but one that nevertheless is crucial for the overall management of business and public organizations.

Although management methods exhibit wide variation in practice, it is easy to define what management is. Management may be defined as *the creation and maintenance of useful and productive order*. At the risk of oversimplification, we can conveniently separate the management activities related to the *creation* of useful and productive order from those related to the *maintenance* of that order once created. Thus it is one task of management to create a new factory, and a different task to maintain that factory as a useful and productive enterprise in the community.

As with the management of factories and other forms of human endeavor, the management of business data depends on an initial creation of useful and productive structures. Prior to the invention of the electronic computer, management of business data was largely restricted to the manual maintenance of accounting data. With the computer came the organization of data into *computer files* containing thousands and often hundreds of thousands of *records*. Here indeed was a useful and productive order, or so it appeared in the beginning. The maintenance of this orderly arrangement of data became the task of the data processing department in a large enterprise. The data in the computer files had to be updated at regular intervals to ensure currency; and because of the need for information by all kinds of people in a large organization, there was a continual increase in the number and kinds of computer files that had to be maintained. This method of managing data by means of computer files worked well in the beginning, but as the number of computer files increased, it became apparent that such *collections* of computer files no longer constituted a useful and productive arrangement of the enterprise's data. In fact, with many enterprises, it would be safe to state that a destructive degree of disorder had crept into the once orderly collections of files. The detailed reasons for this phenomenon are described in Chapter 4, but for the moment it is sufficient to state that by the late 1960s there was a need for a new arrangement of data.

Data bases

The new arrangement of data was called a *data base*. It is probably true to state that modern data bases are among the most highly ordered collections of data ever conceived by man. Complex software products known as data base management systems are used to create and maintain these data bases. But what is unique about these highly ordered collections of data is that the inherent order is flexible, that is, by means of the sophisticated data base management system, the data base can be relatively easily reordered to meet the needs of an enterprise. The needs of an enterprise are constantly changing, and we must never forget that a data base, while fascinating, is not an end in itself. It exists to help an enterprise to be productive and useful, and in practice this ultimately means to help the enterprise to sell a product or service.

Multiple data bases

The old files of conventional data processing were supposed to be independent of each other, but in reality they contained an increasing amount of duplicate data that was often inconsistent. They were also highly inflexible. *A data base is a highly integrated but flexible collection of computer files that cross-reference each other, with little or no duplication of data.*

When the data base systems necessary for managing data bases first appeared, it was thought that the ultimate goal was the creation of one large central data base for the enterprise that would replace all the inflexible computer files.

It was not long before it was painfully experienced that such a grandiose concept had little chance of success in practice, since an attempt to construct such a central data base is likely to bankrupt the enterprise. That expenditures are for acquiring the benefits of advanced technology is no guarantee that they are not also extravagant. A single all-embracing central data base to replace all the conventional files of an enterprise requires expenditures that are all too often an extravagance.

Instead of the large super data base, many different data bases are nowadays used within an enterprise along with many different conventional computer files. These data bases are sometimes called *subject data bases*, since a subject data base contains all the data relevant to a certain type of entity or subject that is important for the enterprise (see Ref. 4). Examples are electronic components, customers, finished products, and so on. But independent conventional computer files are still very much in evidence. Many of them are specialized files that are used in association with data bases. However, many are simply files that contain important data that could well be aggregated into data bases, if and when the cost can be justified. In many cases, for reasons of low return on investment, this will never happen, so that for most enterprises the management of data will include the creation and maintenance of both conventional files and data bases, at least for the foreseeable future.

Managing the data base environment

There is more to the management of data than just having computers for creation and manipulation of data bases. The data in a data base is normally shared between users in many parts of an organization and at many different organizational levels. This gives rise to benefits in the form of increased opportunity for the managers of the enterprise to improve the quality of their decisions. This is of fundamental importance, for in today's highly automated but complex business environment, it is more the quality of decisions that counts than how hard a manager works. But the sharing of data among different departments and even divisions in an enterprise has its administrative costs, partly incurred by the need to hire *data base administration personnel*. Thus the use of the data base tool has a marked effect on the way an enterprise is managed.

The design of this book is based on the premise that a manager can fully appreciate the opportunities and problems inherent in the data base environment only if he or she has a basic grasp of the underlying technology and its historical development. We therefore begin with a study of the technology and defer discussion of the data base environment and data base administration until Chapter 13.

The technology

Data base management has its roots in the technology of processing business data in conventional computer files. There are many different ways of organizing computer files depending on the application. When computer files are integrated under the control of a software system called a data base management system, we have a data base. Thus in order to fully appreciate data bases, we need to have an understanding of conventional computer files as they are used in business. As mentioned earlier, despite the trend to data bases, it is certain that nondata base files are going to be with us for a long time to come. For these reasons it is necessary to begin a study of data bases with a review of computer files, and this is the purpose of the first two chapters of this book. In the present chapter we begin right at the beginning and look at computer facilities for reading in and writing out data, and at data storage facilities. These topics are the very foundation for data base technology.

Most of the book is therefore devoted to giving the reader a basic grasp of data base technology. It is inevitable that it does contain some technical material. Our aim, however, is to impart an understanding of the basic ideas and methods and not to develop technical expertise. For this reason, only absolutely essential technical material is included. In addition, the subject is developed in a step-by-step fashion, starting with the most elementary concepts.

1.1 THE MANAGEMENT OF INPUT/OUTPUT

The *input/output facilities* of a computer are fundamental for business data processing. Data to be processed must be input from some external *storage medium*, and following processing, data must be output and stored. In most of this book we shall be dealing with large quantities of business data that are stored on magnetic disks and occasionally on magnetic tapes. It is therefore important that the reader acquire an elementary understanding of how input/output with these disks and tapes is managed. As we shall see, input/output is managed by both hardware and software facilities. Although these facilities are quite complex, the basic ideas involved are easily grasped by students with only an elementary knowledge of computing. Indeed, it turns out that input/output managment methods have a lot in common with methods used in business logistics.

It is computer programs that initiate the input and output of data, and so we begin our study with a careful examination of what goes on when a computer program is executed.

What does what in a computer

In Figure 1.1 we have a diagram showing the *central processing unit* (CPU) and *internal memory* of a computer. The CPU is the calculating device of a computer and the place where all arithmetic and other operations are carried out. The internal or *main memory* is made up of many storage cells, each of which can hold either an instruction or an item of data. The CPU can connect to any memory cell, but only to one at a time (color connection in the diagram).

In the diagram the memory contains some instructions from a program excerpt and some data items being manipulated by the program instructions. The instructions refer to the memory cells or locations containing the data items, and this collection of memory cells is called the *program data area*. The instructions of a program are carried out by the CPU in a regular sequence, in the order of ascending memory-cell address value, until either a STOP instruction is executed or a GOTO instruction is executed. A GOTO instruction causes a break in the regular execution sequence, a new execution sequence starting from the address specified in the GOTO instruction (see third instruction in diagram).

To actually execute or carry out an instruction, the CPU becomes directly connected to the memory location containing that instruction, and a copy of the instruction is transferred to the CPU. The CPU then accesses the memory locations referred to in the instruction and carries out the arithmetic or other operation specified in the instruction. In the diagram we have used symbolic

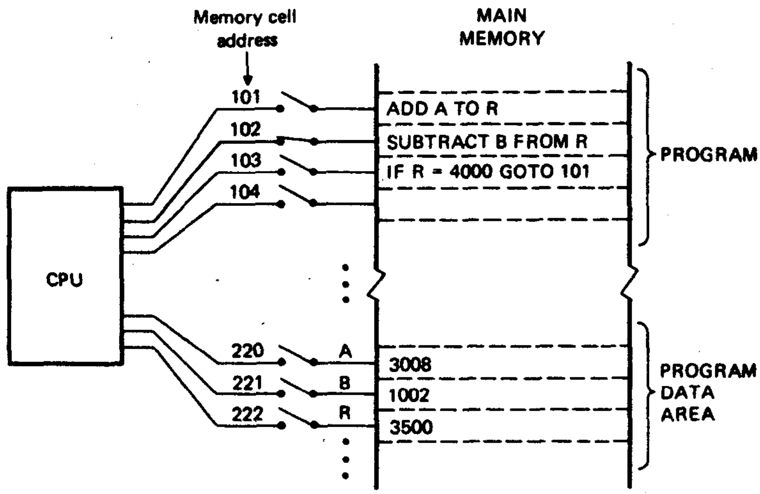

FIGURE 1.1 Shows how the central processing unit (CPU) executes a simple program excerpt residing in the main memory. When a program instruction is carried out, the memory location containing it is directly connected to the CPU, to which the instruction is transferred.

names such as B and R for the memory locations containing data to be manipulated. In an actual executing program the instruction

```
SUBTRACT B FROM R
```

would be replaced by the equivalent of

```
SUBTRACT CONTENTS OF LOCATION 221 FROM CONTENTS OF LOCATION 222
```

In a program written in FORTRAN or COBOL, we use symbolic names for the memory locations that contain data to be manipulated, it being the job of the FORTRAN or COBOL compiler to translate these symbolic names to memory addresses in its preparation of the program for execution.[1]

At this point the reader may well have checked back to the title to confirm that it concerned data bases and not computer machine language. However, our purpose in beginning this way is to make sure that the reader is quite clear about what does what in a computer. From the discussion above we see that it is the CPU that does all that is done, that is, it is the CPU that carries out or executes an instruction in a program by manipulating the data in the program's data area. An analogy may help. In the baking of a cake, we can regard the cook as corresponding to the CPU, the recipe to the program, and the program's data to the ingredients. Thus the cook bakes a cake by manipulating the ingredients according to

[1]The compiler does not complete the job however. It is completed by a loader program that places or loads the user program in main memory for execution.

the instructions in the recipe. Similarly, the CPU processes data by manipulating the data in the program data area in accordance with the instructions in the program. However, in common computer parlance we often say that a program carries out some processing function. For example, we might say that program X computes an employee's net pay, which is equivalent to saying that a certain recipe bakes a cake. Of course it is the CPU under the control of program X that computes the net pay, just as it is a cook guided by a recipe who bakes a cake. Thus stating that a program carries out a processing function is quite clearly a misuse of the English language, but this misuse is well established, and no great harm results from it provided we have only one processor that is executing programs. In a computer system, however, there are typically several processors, most of them involved in input/output, as we shall soon see, and we need to be careful about stating what does what in order to avoid misunderstandings.

We shall not make any further use of diagrams of the type shown in Figure 1.1, relying instead on the type shown in Figure 1.2, which essentially describes the same thing. Here we depict a CPU and a main memory containing a program accompanied by data in a program data area. We use the two-headed arrow to indicate that the CPU may connect to any memory location occupied by a program instruction or data in order to execute the program. A black arrow indicates that a program may be executed by the CPU, while a color arrow indicates that a program actually is being executed.

Input/output and external storage media

Our next consideration has to do with how data gets in and out of a program data area. Suppose the data to be processed is on a magnetic tape or disk. Since the CPU under program control manipulates only data in a program data area in main memory, there has to be a way to move data to be processed from the tape or disk to the program data area, and vice versa. The solution to this problem leads us to the *multiprogramming operating system*, a basic software component of every larger computer system and a component to which a data base management system must interface. A brief review of multiprogramming systems will therefore be relevant.

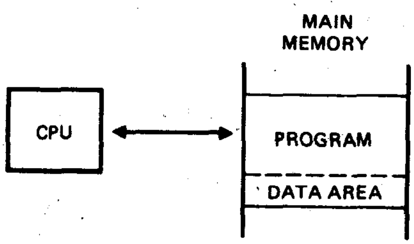

FIGURE 1.2 Simple representation of a program on main memory that may be executed by a CPU. The two-headed arrow indicates that the CPU may connect to a memory location, permitting transfer of data or an instruction to the CPU, or transfer of data from the CPU to the main memory.

Multiprogramming operating systems

In a primitive computer system it is the CPU that connects directly to an external device such as a tape drive or card-reader; in such a primitive system the CPU accesses data in the external medium and places it in main memory, doing this under the control of some input/output subprogram in main memory. Unfortunately, accessing external storage is a very slow process compared with accessing main memory; for example, a modern CPU could make about 1 million main memory accesses in the time it takes to make about 10 disk accesses. Put another way, the CPU can process a very large amount of data in a main memory program data area in the time taken to get additional data in from an external storage device. (The reason for this is that the external devices are electromechanical, while main memory is electronic.) Thus input/output is a bottleneck for computer systems, no matter whether the input/output is with a simple tape file, or with a complex data base.

The common solution to the problem is to use a multiprogramming operating system along with simple subsidiary or assistant processors called *input/output processors*, or *channels*. When the CPU can no longer continue execution of a program because additional data must be obtained from an external device, the job of procuring this data is handed over to an I/O processor, while the CPU executes some other program with sufficient data in its program data area. The situation as far as programs in main memory is concerned is depicted in Figure 1.3. Main memory can be regarded as divided into *partitions*, or *regions*. A typical partition holds a program that carries out some data processing function (an *application program*), together with the program's data area and some additional subprograms and data areas which we will come back to shortly. One

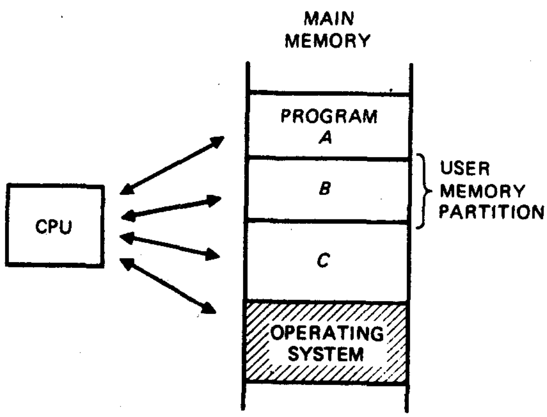

FIGURE 1.3 With most larger computers, main memory contains several user programs in different stages of execution, but at any instant only one program is being executed (grayed arrow). Each program (plus certain subsidiary programs and data areas) can be taken as occupying a memory partition. At least one memory partition is reserved for the operating system, a system program that determines (among other things) which program will execute next.

or more of these memory partitions are also allocated to the multiprogramming operating system, which is a large systems program that controls the execution of the application programs and the input/output processors (not shown in Figure 1.3). At any instant, the program in only one partition is being executed by the CPU. When the CPU executes an instruction requiring access to an external device (typically a READ or WRITE instruction), control passes to the operating system; i.e., operating system routines are executed, and the operating system then starts one or more I/O processors. Control then passes back to some other application program. Thus at any given instant, the programs in the various partitions of memory are all in various stages of execution, but only one partition is actually executing. At the same time one or more I/O processors are transmitting data.

Role of I/O processors

In Figure 1.4, we show in more detail the contents of a typical main memory partition containing an application program, and we also show a single I/O processor or channel connected to a *disk controller* which is in turn connected to one or more disk units. To understand exactly the role of the memory partition, I/O processor, and disk controller, let us suppose that user program *A* is executing, and that the program contains a READ instruction requiring that data be obtained from a disk. When the CPU executes the READ instruction, the CPU switches execution to the *access method*. The access method is really a large subprogram of the operating system, but it resides in the partition belonging to our user program *A*. The access method checks in an area of program *A*'s

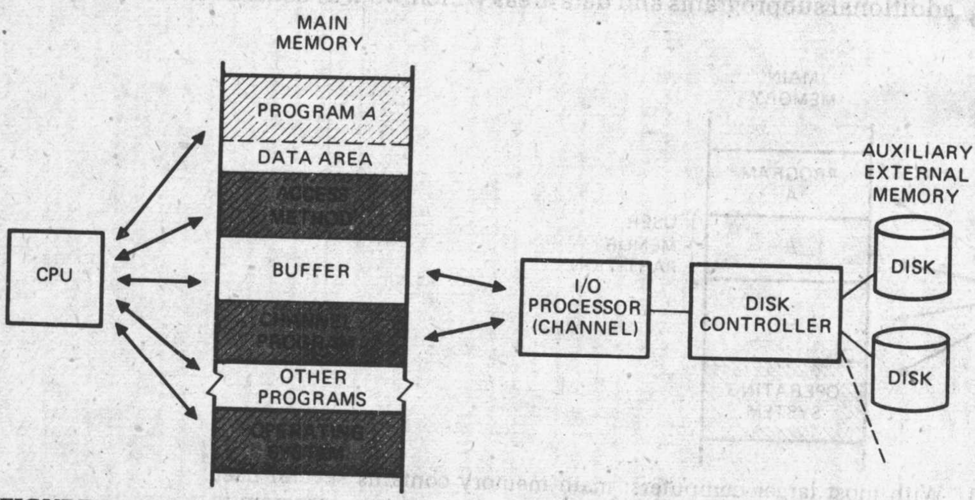

FIGURE 1.4 Shows how data (in the form of records) is input from and output to external memory (shown here as disk units). The heavy black lines in the main memory show the boundaries of memory partitions. The top partition contains a user program with its data area, plus a buffer area (which can hold one or more blocks of records), plus two subsidiary programs which are really part of the operating system, namely, an access method and a channel program.