Database Design

Gio Wiederhold

DATABASE DESIGN

Gio Wiederhold
Assistant Professor of Computer Science
Stanford University
Stanford, California

McGRAW-HILL BOOK COMPANY
New York St. Louis San Francisco Auckland Bogotá Düsseldorf
Johannesburg London Madrid Mexico Montreal New Delhi
Panama Paris São Paulo Singapore Sydney Tokyo Toronto

Library of Congress Cataloging in Publication Data

Wiederhold, Gio.
Database design.

(McGraw-Hill computer science series)
1. Data base management. 2. File organization (Computer science) 3. Data structures (Computer science) I. Title.
QA76.9.D3W53 001.6'425 76-54320
ISBN 0-07-070130-X

DATABASE DESIGN

1 2 3 4 5 6 7 8 9 0 K P K P 7 8 3 2 1 0 9 8 7

This book was set in Times Roman by Progressive Typographers. The editors were Peter D. Nalle and Madelaine Eichberg; the production supervisor was Charles Hess. The drawings were done by Long Island Technical Illustrators.
Kingsport Press, Inc., was printer and binder.

Preface

ORIGIN

The outline for the material in this book is the result of a course given at Stanford University on Database Structures in 1971. Little coherent published material was available, even though a large number of references could be cited. In particular, no clear definitions of the concept of a *schema* were available. There was clearly a need to begin to bring together the knowledge in the area of database management in a structured form suitable for teaching, using both commercial and scholarly sources. Subsequently, courses based on the developing notes have been taught at the University of California, in both San Francisco and Berkeley, as professional seminars sponsored by the local ACM chapters, and at the Institute of Information Technology, Japan.

I have been advised that "anyone who has attempted to teach the subject of Information Management Systems has damaged his career." This may be due to the one-sided experience presented in many of the available publications. A mere collection of important papers does not provide an adequate framework for a general approach. An effort has been made here to develop concepts from the diversity of material and to present the subject in such a way that the concepts which evolve can be applied in practice. An engineering attitude to the problems of database organization has been used in order to combine formality with applicability.

I hope that this text fills the void that was found and that it provides material to extend and improve the teaching of the data-processing aspects of computer science.

OBJECTIVE

This book is intended to present the methods, the criteria for choices between alternatives, and the principles and concepts that are relevant to the practice of database software design. No actual systems are described completely, nor are systems surveyed and compared, although specific examples are used to illustrate points to be made. The material provides the basis to allow the reader to understand, recognize the implications, and evaluate database approaches.

This book includes two major sections:

1 The description and analysis of file systems (Chaps. 2 to 6)
2 The description and analysis of database systems (Chaps. 7 to 10)

The first section is intended to provide a solid foundation for the latter section, since the issues arising in database design are difficult to discuss if file-design concepts are not available to draw upon. A number of subjects which pertain to both files and databases, namely, reliability, protection of privacy, integrity, and coding, are presented in Chaps. 11 to 14. If the material is taught in two distinct courses, these chapters should not be ignored in either course.

The audience may range from students of computing, who have finished a reasonably complete course in programming, to applications and systems programmers, who wish to synthesize their experiences into a more formal structure. The material covered should be known by system designers or system analysts faced with implementation choices. This book probably contains too much detail to be of interest to management outside of the database management area itself.

DESIGN METHODOLOGY

This book presents a comprehensive collection of database design tools. In order to employ them, a strategy of problem decomposition, followed by a structured design process is advised. The conceptual categorization of database approaches given in Chapter 10 will help with the initial objective definition. Chapter 7 provides the means to construct a model which integrates the requirements of multiple applications which share the database. The schemas in Chap. 8 provide data description methods, and existing database systems, described in Chap. 9 and Appendix B, suggest available implementation alternatives.

If the database is to be directly supported by a file system, then the basic file choices in Chap. 3 and their combinations shown in Chap. 4 provide the alternatives. The data representation can be chosen using the material of Chap. 14.

The performance of the chosen approach can be predicted as shown in Chap. 5. The model defined earlier provides the framework for the translation of application loads to the load to be handled by the database. An optimal file design may be selected after application of the load parameters to the performance formulas from Chaps. 3 and 4. The formulas also require the hardware description parameters of Chap. 2.

An illustration of the database design process is given in Wiederhold.[76] Problems of reliability, protection, and integrity (Chaps. 11, 12, and 13) require a close scrutiny of the available operating system. The long-term maintenance is guided by considerations presented in Chap. 15.

CURRICULA

In terms of published curricula the material covers some of the topics for Course A8 specified in the report by the ACM Curriculum Committee on Computer Science (ACM[68]) and includes some material for Course A5. It also covers some topics not discussed in the report. The material is presented so that it would be particularly appropriate as part of Course 4 of the Computer Science Program for Small Colleges presented in Austing.[73] It provides all the material for the file and database subjects of Courses C1, C2, C3, C4, and D2 specified in the Curriculum Recommendations for Graduate Professional Programs in Information Systems (Ashenhurst[72]). The author feels, however, that these courses are easier to teach using a depth-first approach to the subjects versus the breadth-first approach advocated in the curriculum proposal.

Current courses and curricula tend to neglect files and databases, possibly because of a shortage of teaching material. This weakness is especially evident in students who enter industry or commerce with a bachelor's level education in computing or data processing. It is reasonable to expect that students majoring in computing and computer applications should be familiar with this subject area (Teichrow,[71] Sprowls[75]). Projections regarding the future use of computers give a considerable weight to the importance of the database area (Steel in Jardine[74]), so that we can expect an increasing demand for educational services in this area.

TERMINOLOGY

The terminology in the area of database and file management is yet quite inconsistent. Within this book a major effort has been made to use a consistent vocabulary and to define all terms. These terms are listed in the index. In order to aid both experienced readers and users of the references, Appendix A cites common alternate terminology and compares it with the terminology used in this text. The introductory chapter is mainly devoted to definitions. It is assumed that all the subjects touched upon are familiar to the reader.

Most of the program examples that appear throughout the text use a simple subset of PL/1. The variable names are chosen so that they will aid in the comprehension of the programs; they are printed in lowercase. Keywords which are to be recognized by the translating programs appear in uppercase. The programs are designed to be obvious to readers familiar with a procedure-oriented programming language. A number of introductory PL/1 texts (Hume,[75] Richardson,[75] Mott[72]) can be used to explain features that are not recognized. Some PL/1 texts, unfortunately, omit the statements required for the manipulation of data files.

The sample evaluations of database-design problems are of necessity incomplete, and frequently are based on simplistic assumptions. An effort has been made to note such assumptions, so that the design process itself will not be compromised.

EXERCISES

The exercises listed in each chapter have been kept relatively simple. It is strongly suggested that an analysis of some of the systems described in the referenced literature be used for major assignments. Many of the problem statements in fact require such material. The analysis of comparison of actual systems may seem to be an excessively complex task, but the effort to be expended has been shown to be manageable by students when the material of this book has been assimilated. Appendix B provides references to a large number of system implementations. A knowledge of calculus will be helpful when doing some of the problems, but purely graphical methods will provide adequate results.

REFERENCES

Source material for this book came from many places and experiences. References are not cited throughout the text, since the intent is to produce primarily a textbook which integrates the many concepts and ideas in database design. A background section at the end of every chapter provides references to the sources used and indicates further study material.

The bibliography has been selected to include some material for each of the subject areas introduced. Whenever many references could be cited, those sources which are easier to obtain, such as books and journals, have been chosen over those which are more difficult to locate, such as trade publications, research reports, and theses. Computer manuals are not cited; up-to-date information is best obtained from manufacturers. Preference is also given to references which are directly related to large-scale database-oriented problems. In order to keep the size of the bibliography in this book within bounds, those papers which appear in books or proceedings containing much relevant material are cited only by author and source publication. The references cited will provide a generous foothold for students intending to pursue a specific topic in depth. The list may be able to inhibit some of the efforts being expended toward previously solved problems. Redundant efforts are seen frequently when a discipline discovers databases. An extensive annotated bibliography is being maintained by the author and may be requested through him, if more reference material is desired.

I wish to apologize to those authors of work I failed to reference, either because of application of these rules or because of lack of awareness on my part.

ACKNOWLEDGMENTS

I wish to thank John Bolstead, Frank Germano, Jerry Miller, Richard Moore, Bernard Pagurek, Gerry Purdy, Diane Ramsey-Klee, Justine Roberts, Diane Rode, Hank Swan, Steve Weyl, and Anthony Wasserman for their assistance in reviewing and correcting the manuscript; the members of the SHARE Database Committee for many ideas engendered during mutual discussions; and the students in the courses in which I covered this material for their support and ideas.

Specifically, I wish to recognize Donn Parker's and Lance Hoffman's reviews of Chap. 12 and the discussions on relational databases with Steve Weyl, which have helped me greatly. Dick Karpinski, Jean Porte, John Rhodes, and Eugene Lowenthal reviewed sections of particular interest to them.

Many colleagues have provided ideas and material. Thomas Martin and, again, Jerry Miller and Steve Weyl deserve special recognition, since without their interactions during the implementation of file and database systems, I would not have been able to develop the insights on which this material is based. Miller and my wife went through the galleys and proofs with great care, the errors which remain are mine. I received support from the National Library of Medicine at the University of California, San Francisco, during the final year of this work, and have also benefited from the computer services at Stanford University.

This book would not have been written without the inspiration, support, and just plain hard work of my wife Voy. The appreciation she has received from her students and from users of computer manuals she has written has encouraged me to attempt to present this material in as straightforward a fashion as she has been able to present PL/1 programming.

GIO WIEDERHOLD

SYMBOLS USED IN PERFORMANCE FORMULAS

A average space required for attribute name
a number of different attributes in a file
a' average number of attributes occurring in a record
B block size
b block count
btt block-transfer time = *B/t*
C cost factors
c computational overhead per record—used where the effect may not be negligible
D space required for goal attribute
d number of records that have been invalidated
e = 2.718281828459 . . .
F subscript denoting a fetch for a specific record
f other cost factors, sample size
G space required for an interblock gap
h classification variable
I subscript denoting insertion of a record
i counting variable
j number of cylinders
K kilo or thousand (1024) times
k number of tracks per cylinder
L load-frequency factors
M multiprogramming factor
M mega or million (1,048,576) times
m number of available slots for records
ms millisecond
N subscript denoting getting the next serial record
n number of records in a file
o number of records that overflow the primary file
P space required for a pointer
p probability, also subscript denoting primary
q production demand by a file computation
R space required for a complete record
RW subscript indicating rewriting of a record
r rotational latency time
SI storage space for index
s average seek time
s' effective seek time
s second
T the time required for various operations
t transfer rate from a storage unit to processing memory
t' bulk transfer rate
U subscript denoting an update of a record
u utilization
V average space for value part of an attribute
v number of records updated
W wasted space due to gaps per record
w wait time in queues
X subscript denoting reading the entire file
x number of levels in an index structure, also level number of master index
Y subscript denoting a reorganization of a file
y fanout ratio
Z subscript denoting an update by replacement with a larger record

Contents

Chapter 1

Definitions and Introduction to the Subject

The order and connection of ideas is the same as the order and connection of things.

Baruch (Benedict de) Spinoza
Prop. VII from Ethics, Part Two

When we talk informally about a *database*, we refer to a collection of mutually related data, to the computer hardware that is used to store it, and to the programs that are used to manipulate it.

This chapter begins by defining the concept of a file, since files are the prime component of database concerns. The second section discusses operations or tasks performed when using a database.

In Sec. 1-3 we develop a classification of data management which will provide the structure for all subsequent material. We then take familiar concepts from programming and relate them to the presented classification. This sequence is intended to provide a link between our programming experience and the approach taken in this presentation of database-management methodology. In Sec. 1-8 a list of application areas is given as an aid for the selection of a topic for analysis to be followed throughout the text.

You may have learned nothing new after you have read this chapter, but the framework established will allow us to proceed through the book in an orderly fashion.

1-1 FILES

A database is a collection of related data. The data storage for a database is accomplished by the use of one or more files. A *file* is defined to be a collection of similar records kept on secondary computer-storage devices. Typical of *secondary storage* devices are disk drives with magnetic disks, although a number of alternate possibilities will be described in Chap. 2. A *record* is defined at this point to be a collection of related *fields* containing elemental data items. A more formal and detailed definition will be developed in Chap. 3. A data item typically represents a value which is part of a description of an object or an event. Computational processes can manipulate such values.

1-1-1 Size

To warrant the attention and the approaches discussed in this book, the database should be reasonably *large*. We will be discussing only processes that are applicable to large external files. Collections of data that can be processed in their entirety in the directly addressable memory of a computer, its primary storage, allow techniques that will not be covered here. The use of the term database also implies that a number of people are involved. Not only may data entry be done by people far removed from the users, but the data may contain information suitable for a variety of purposes. The quantity of data to be handled may range from moderately large to very large. These measures of size depend on the hardware and on operational constraints which may apply in a given environment.

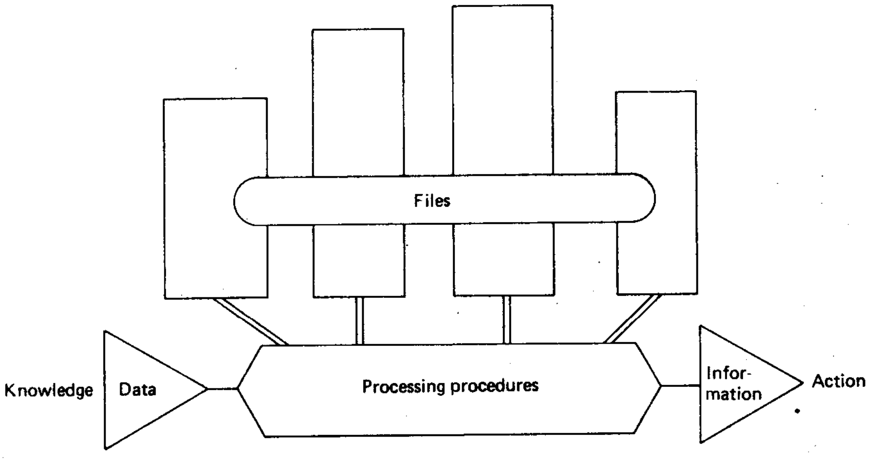

Figure 1-1 A database.

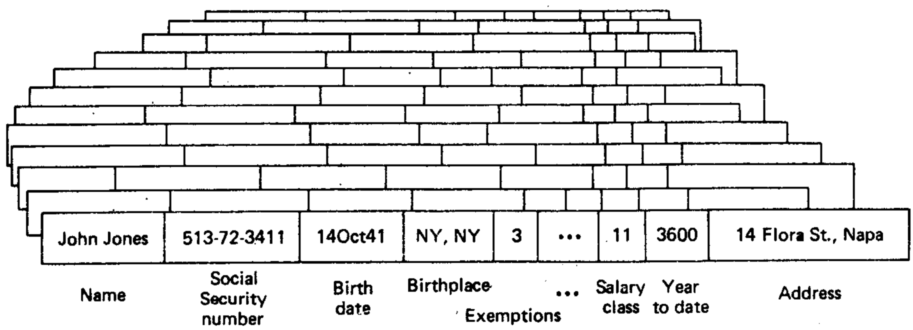

Figure 1-2 A payroll file.

Large implies a quantity of data which is greater than a single person can handle wholly by himself, even when he or she has access to a computer system. The actual quantity will vary depending on the complexity of the data and applications. An example of a large database (Emerson in Jardine[74])* is the integrated personnel and product data system in a manufacturing company of about 6000 employees, with more than 300,000 records of 21 types.

A *very large* database is an essential part of an enterprise and will be in continuous use by many people. At the same time it will extend over many storage units. An example of a very large database is found at a telephone company with 5 million subscribers (Karsner in Kerr[75]). Much larger yet are databases at the social security administration and other national systems.

To have a copy of the contents of a file frozen at a certain instant in time is important for many purposes, including periodic analysis of data, backup for reliability purposes, and auditing. To avoid problems, it is best not to permit a file to be used and modified while it is being copied. In a very large database these two considerations conflict. This is a useful functional definition which imposes certain design constraints on the systems which we will be discussing.

1-1-2 File Organization

Files not only are characterized by their size but are further distinguished by their organization. Differences in the organizations of files lead to differences in performance when storing and retrieving records. The evaluation and comparison of file organizations is an important aspect of this book.

Six basic file-organization types will be analyzed in detail in Chap. 3, and Chap. 4 will show some of the possible combinations and permutations of these basic file types. A database often requires more than one type of file.

1-1-3 Input-Output

When reading or writing files, data is transferred between storage units of the computer system. When reading input or writing output, data enters or leaves

* Superior number next to reference indicates the year of publication.

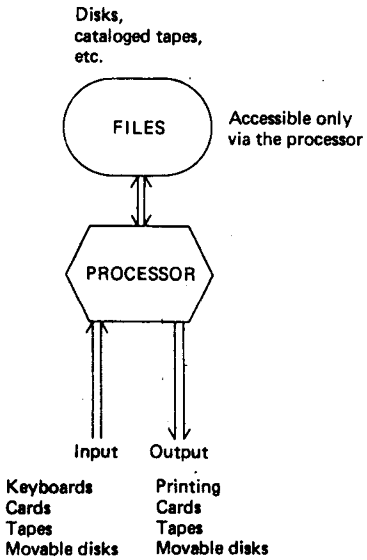

Figure 1-3 Files versus input-output.

the computer system. A database is concerned with the data which remains within the scope of the system. Data which is written on tape, stored, and later mounted and read again can be part of the database. Data which is taken out, modified externally, and reentered has to be considered new input.

Examples of devices used for input and output are keypunched cards, printed reports, tapes shipped to other computer system, and computer-gererated microfilm output.

Examples of devices used for files are fixed disks and drums, master tapes or disks kept at the computer site, archival tapes kept in remote vaults for protection, and card decks containing lists of customers or personnel.

In many computer systems the categories of file versus input and output are not well delineated. The subject of input and output is at best as complex as the database area and will not be covered in this text. We will assume that adequate input and output capabilities are available, including on-line terminals where appropriate, when we talk about database systems.

We will not discuss file organizations that are based on input and output facilities. These tend to regard data as a continuous stream of characters. *Stream files*, as defined by PL/1, and their equivalents in other systems, are based on the reading and writing of continuous lines of text. Continuous text streams are important for communication but are not suitable for data manipulation. The word ''file'' will also not be used to refer to the hardware employed to store the data comprising the files.