¡le

DESIGN CONCEPTS AND IMPLEMENTATION OF DATABASES ,
Gio Wiederhold, INTRODUCTION, Chapter 1, file D1CID.TEX April 4, 1996

Chapter 1: current Chapter 1

1-0 Introduction

add references to DBD:File Organization volume in 1–3.

1-1 Databases

1-2 Database Management

1-3 Database Design and Models

1-4 Size ???

1-4 Databases and the World

1-5 A Hierarchical View of Data

1-6 Binding ¡¡ move to Chapter 2?? ¿¿

1-7 Applications

remove file issues from 1–4, 1–5.

redo 1–6 to reflect databases only.

insert material from 7–0 as map to book

Background and Refernces

An extensive history

First moving pictures:

1834, William George Horner: 'Wundertrommel', a cab with 5 slits and inside pictures of an acrobat turning a ball with her feet.

Chapter 1

Introduction and Definitions

Most of what follows is true.

Paul Newman ¡Robert Redford¿ as The Sundance Kid
Butch Cassidy and the Sundance Kid, 20th Century Fox, 1969

1-0 INTRODUCTION

¡insert A here¿ ¡¡AD¿¿

This book deals with databases and the systems used to manage them. We define a database to be a collection of mutually related data, which is managed so that information can be obtained from it.

A database management system (DBMS) is a collection of programs which uses computers to store, maintain, and interrogate the database. To support the DBMS we require not only hardware, by operating systems and file system services. Figure 1-1 shows the components. ¡how does this compare with Fig. 2-1?.¿

¡¡¡¡¡

DEFINITION ¡¡move into body¿¿ An integrated collection of computer programs which provides the services required to manage a database is now refered to as a database management system (DBMS). ¿¿¿¿¿¿

0-1 Layout of this Chapter

In this chapter we will survey the major concepts associated with databases and DBMS.

¡more¿

It is important to understand where database technology is applicable. Section 1-9 provides a number of examples, and the project exercises are geared to an application selected by the reader. Since databases are used in most activities performed with computers a classification is provided which will assign priorities to the ¡design criteria¿.

, and in the last section, Sec. 1-9, provide a road map to the remainder of this book. The background section provides a short history of the field.

1-1 DATABASES

Data are recorded observations of facts and events. Data describing an object or event is placed together to form a record, and collections of records of similar events are filed together.

¡¡A2¿¿

A database is composed of multiple related files.

A single database may serve several applications. For instance, a film production company may used the data both for scheduling performances and for financial statements. It may also provide an inquiry function, that queries as `Where will ''Gone with the Wind'' be performed next?` can be rapidly answered.

¡¡A3¿¿

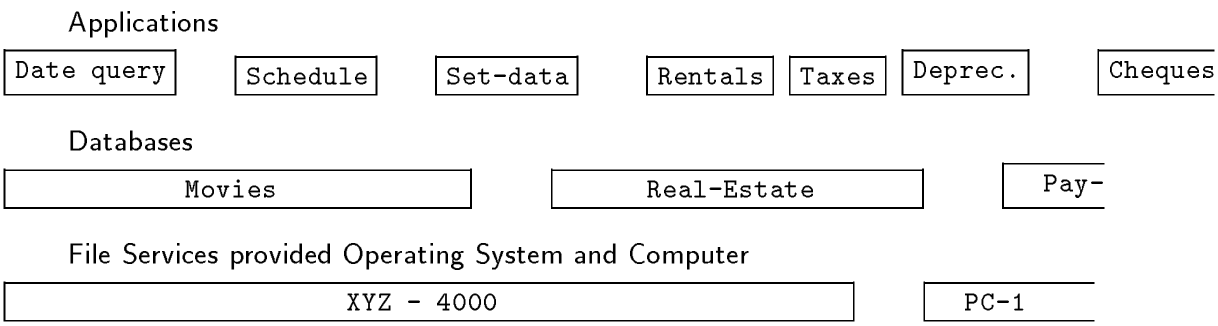

Figure 1-1 Databases and their friends.

Unique Characteristics ¡¡Where¿¿ The concept of users who are sharing a common, long-term information resource leads to several features which are unique to database systems: schemas, logging, and non-procedural access. Not part of a database itself, but an important aspect of data management is the concept of database modeling. Modeling provides conceptual tools to abstract the complexity of a database, and plays an important role in the interaction of users, designers, and implementors of databases. We will present models in a later section of this ¡¡paper¿¿.

Schemas An important concept which distinguishes database management from traditional files usage is the use of a schema. A schema contains a machine-processable description of the contents of the database. This description is available to the users so that they can learn the names and the description of the data elements, but its primary function is to control the input-output operations to the files which are used to hold the database.

Using a schema, retrieval of data from the database is carried out by giving the name of the data element. All accesses to the database are indirect and mediated using the information kept in the schema. Invariably, after some time, additions to the database are required. The changes may be needed to satisfy one of the users or for some aggregate system enhancement. These changes are reflected in the schema, and often most applications should continue to operate undisturbed.

The information given in the schema is made available in a variety of ways which relate to the organization of the DBMS.

I A simple and flexible approach is to interpret schema tables at execution time. A schema translator will compile the tables into a suitable format.

C Faster execution is possible by compiling the schema information into a suite of routines which execute the basic append, read, and update operations on the data elements. To avoid an excessive number of routines related elements may be grouped into segments.

T The schema information can also be bound into transaction programs. Then the schema information is referenced during their compilation and incorporated into the application code. Changes to the schema require recompilation of progrmans submitted by the users. These programs will be kept on a library associated with the DBMS to assure that any required changes are incorporated.

In either case the the symbolic names given by the user are translated to addresses within the appropriate file. The schema also includes other specifications, such as the format of the data, its type, its relationships to data in other files of the database, and information about access structure, which are needed to locate the data rapidly. A schema may also contain information on access privileges.

The use of a schema simplifies the sharing of the database, since it provides a single interaction point for each user, although many distinct users use the database. The schema interface increases the processor load.

Logging Logging of user interactions with the database provides a historical record. This record serves a variety of functions. Most important is that it protects the common resource. Recovery from a failure of the computer system is made possible by combining a backup copy with recent log entries.

The log also provides an audit trail and the ability to undo the effect of previous erroneous actions to the database. Without a systematic ability to repair errors remote users would be frustrated and soon cease to use the system.

The use of logging helps the sharing of the database since it provides operational isolation from errors and failures. Logging places additional demands on the storage resources of the system.

Non-procedural Access Capabilities Since databases may be used directly by clerical and management personnel there also has been a strong trend to develop higher level, problem-oriented languages in the database area. Today database management has joined word-processing and spreadsheet programs in the micro-computer market, an arena where most people who interact with their computers have learned little or no programming.

The use of high-level languages also increases processing requirements. Since most of the interaction uses display terminals, demands are also made on high bandwidth communications and effective graphics interfaces.

¡¡B1¿¿

1-2 DATABASE MANAGEMENT

In order to maintain a database many programs are needed. Some of the programs are directly concerned with the applications, others provide general support functions. The application programs for modern DBMS's are organized as *transactions*, the remainder make up the *database management system*. We will introduce these two components in term.

```
    |                        Users                       |
    ------------------------------------------------------
    |   direct    |                 |  indirect   |
    | interaction |  Graphics       | interaction |
    -------------------              -----------------
    | User Interfaces |              |  Programming  |
    -------------------              |   Languages   |
\
    /             -----------------
    Artificial   >    <       Formal      \       /
    Intelligence /           Languages    /      \
    ---------------------------------------|
    |           Database        ---------------------------
    |          Management       |        Databases        |
    |           Systems         ---------------------------
    ---------------------------------------|
    File    \                   /  Reliability
    Support  /                  \     Support
    -------------------------
    Processors   |      Operating       |  Communication
    general/   |       Systems        |    Systems
    special    -------------------------
    |                 |
    ------------------------------------------------------
    |         Large Capacity Storage Devices             |
    |                                                    |
    |     VLSI     \     Magnetic      /     Optical     |
    |  Technology   \   Technology    /    Technology    |
    ------------------------------------------------------
```

Figure 1-2 The place of databases within computer systems.

1-2-1 Transactions

The operations that applications performed are organized into *transactions*. A transaction may be a simple query or a complex computation. There will also be update transactions to the database, and these have to be performed with care, to assure that the integrity of the database is maintained. Much cross checking may be necessary during an update transaction.

An important concept is that transaction are always to be either executed completely, or not at all. This means that if a transaction fails, all partial effects on the database must be undone.

---

1 A simple database query:

"`Where will ''Gone with the Wind'' play next?`"

2 A transaction involving some processing:

"`Compute monthly net profits for "Humor" class in "Urban", "Sub-urban" theaters.`"

3 A simple update of an existing record:

"`CHANGE release_date("Star Wars IX") TO 10Dec87.`"

4 a transaction involving processing, reporting, and database update:

```
“Execute SCHEDULING_PROGRAM(Theatre"Galaxy 4",
        period 1Q87, category 17)''
```

Figure 1-2 Transactions on a database.

Transaction may require the use of multiple files, and these files may be distributed over multiple computers. The concept of a transaction will be further expanded in Sec. 2-3.

¡¡B4¿¿

1-2-1 Database management support

To manage the voluminous and diverse data there must be a well integrated set of programs which carry out the actions specified by the transactions. These programs must provide:

1 Persistent storage - data must be lost due to accidents or errors
2 Access to the database by a variety of uses and applications, often active at the same time
3 Maintenance of database integrity, while multiple activities take place
4 Limitation of access to the database, or unauthorized subsets of it, to authorized users only.
5 Adaptability of the database to changing requirements and resources.
6 Efficiency of execution of these functions

¡¡B5¿¿

The actual management of the resources: allocation of storage, initiation of computation processes, handling of data transfer between computers and programs, and input and output is performed with the help of file management for such systems are summarized in Chapter 2, but detailed knowledge is not conveyed in this book. One database management system will be able to support many databases. Sometimes there will even be more than one database using the same database management system on the same computer, because the two databases were not, or not sufficiently so, related. We also find some computer sites with multiple database management systems in use. Sometimes there is a good reason for having multiple systems, sometimes it is due to history.

1-2-3 Database Systems and Information

1-3 DATABASE DESIGN AND MODELS

The process of designing a database to serve a family of applications involves much detail. The implementation of a database management ¡¡¿¿.

¡¡B6¿¿ We can refer to the combination of applications, database management system, and their supporting software and hardware as a *database system*. The objective of any such system is to provide information to the user so that they can combine this information with their, or with stored knowledge. ¡copy F10D Knowledge defs, Information def¿

KNowledge can be represented in a variety of ways.

1 As English —or other —language text to be intepreted by the reader
2 As a computer program, to be intepreted by computer hardware
3 As logical rules, to be unified and interpreted by logic program interpreter
4 In knolwdge -representation schemes as frames and semantic nets, for interpretation by one or more interpreters, adapted to specific types of tasks

Preface

This text distinguishes itself in that it considers the maintenance of the database throughout. The cost and effort to maintain databases is high, and a database system cannot provide useful information unless it is well maintained. WE also assume throughout that the database may be distributed over multiple computers.

A unique aspect of this text is found in Chapter 7 where a specifications for the modules of a database management system is given. This material conveys both an insight why a DBMS is comples, and how it may be organized to make the complexity manageable and maintainable.

¡¡ remove 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 used to manipulate it. ¿¿

> By mutually related data we mean that the data represents knowledge about a specific enterprise, say a company, a university, or a governmental institution. The data may also be related because it deals with a certain problem area, perhaps about a disease which is of concern to the staff of a number of hospitals. The data should be organized so that it can be processed to yield information.

The organization of the data in a database has to represent the underlying meaning or semantics of the data correctly and efficiently. In conventional programs the structure of data is arranged for the convenience of the program. A database contains data to be used by many and diverse programs. The organization of a database can hence not be solely determined by decisions made while programming specific functions.

1-4 Size

To warrant the attention and the approaches discussed in this book, the database should be reasonably *large*. We will concentrate on topics that are applicable to large external files. This outlook limits the choice of algorithms that can be applied. 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 variety of people are involved. Data entry may be done by people far removed from the users and the data may contain information suitable for several 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.

> *Large* implies a quantity of data which is greater than a single person can handle alone, even when aided by a computer system. The actual quantity will vary depending on the complexity of the data and applications.

> 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 require many storage devices. A very large database presents particular problems in its management, since it cannot be turned off without affecting the operation and well-being of the enterprise into which it is integrated. A formal definition of a very large database is one where the time required for making a copy is greater than the permissible interval between update transactions performed to keep the database up to date.

An example of a large database 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 [Emerson in Jardine[74]]

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.

Figure 1-25 Large and vary large databases.

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 modified while it is being copied. In a very large database these two considerations conflict. This definition imposes certain design constraints on the systems which we will be discussing.

1-4 DATABASES AND THE WORLD

The contents of a database is intended to be a representation of the real world. It is desirable to have an accurate and complete representation of our knowledge about reality so that decisions made on the basis of extracted information are applicable to the real world. It is, however, soon obvious that the model of the real world, as entered into a database, is both crude and incomplete.

While searching for a representation there is a reliance on abstraction: one speaks of a specific individual as an employee, a welder, a manager; and then this abstraction is encoded into numbers or characters, which are subsequently transformed into binary computer codes. Statements that *the database content represents reality* may indicate the goal of a system designer, but will be misleading to the user who has to reconstruct useful information from these codes.

During encoding and decoding a number of distortions and biases are introduced. ¡...¿

This material is presented to provide guidance to applications designers when faced with the initial stages of a database problem. It is not unusual that a programmer is asked to *computerize* an information problem. Since a traditional computer science or engineering education has been concerned only with data-processing rather than with the transformation of knowledge to data, this facet of the database design effort may be neglected. Decisions implemented at the level of representation will bind the database to a specific content for many years of its operation. A designer of a database systems should be aware of alternatives of representation so that the system will support diversity and only restrict the unreasonable.

1-5 A HIERARCHICAL VIEW OF DATA

Data can be approached from two points of view. There is the comprehensive view, in which the database is seen as a component of an overall information system involving people, organizations, and communication. The database should describe all of this world. In contrast there is a microscopic view, where the database is a collection of the hardware bits and bytes provided by computer technology. This book present file organization according to a viewpoint between these two extremes.

For us a database is a structured representation of data which describes a defined and quite limited subset of the real world. This representation is implemented using files, records, and references between these records. Figure 1-12 presents the structure in the form of a layered pyramid. The breadth of the base symbolizes the volume of data required to obtain information.

To be expanded to 193 pt for Figure space
was Fig.1-12

Figure 1-12 Database layers.

By not focusing on the comprehensive aspects of databases we avoid having to consider the many types of applications of files. The technical aspects of stored data can be discussed independently of the informational significance of the data value stored. This focus is not intended to imply that people who actually implement information processing systems should ignore these considerations in their particular applications. It is clearly irresponsible if computer professionals ignore the ramifications of the task at hand or their contribution to the social and human environment. A number of textbooks deal with the proper use of information for business and governmental operations and decision making [Davis[74], DeGreene[73], Gotlieb[73], Westin[71]].

¡¡ Also database systems are not discussed in this text ... more, citations ¿¿

This text also avoids subjects on the engineering side of the file technology, namely the design and construction of data storage devices. Although the capabilities and performance of computer hardware are central to the evaluation of file systems, very little discussion of how the performance is achieved will take place. An actual system developer must understand the architecture of the hardware that provides the building blocks of the computer systems. Textbooks are available for the computer engineer at many levels of sophistication [Hill[73], Siewiorek[82]].

This, first book of a pair, concentrates on the layer immediatly above the hardware, where the resources are organized into files to provide rapid and reliable storage of data. The second book, or any of many database system texts, will take files as building blocks for larger, multi-file systems.

Table 1-2 presents the four layers and their functions. The table also lists the topics which may be studied to provide more background for each of the layers, The reader is not expected now to be knowledgeable in all these areas; they are listed merely as navigational aids.

The term *naming* in the table is used to mean the creation of symbolic references to data with minimal concern as to location and form. The term *addressing* is used to denote the exact specification of position and representation. We see in Table 1-2 that as we descend to more primitive layers, naming is replaced more and more by addressing.

A comparison A comparison can be made here between the layers presented in Table 1-2 and the operation of a railroad.

1 The *conceptual layer* concerns itself with the idea of value added due to the transportation of the goods and their relevance to the economic structure.

2 The *descriptive layer* concerns itself with scheduling, train sizes, and equipment combinations required to move goods at a a reasonable cost and with adequate flexibility.

3 The *organizational layer* concerns itself with the building of trackage, the operation of marshaling yards, and the operational efficiency of the complex.

4 The *material layer* concerns itself with the design of locomotives, freight cars, signaling mechanisms, rails, and ties, and all the myriad of pieces of equipment required to actually run a railroad.

DESCRIPTION AND ORGANIZATION This text considers the third layer shown in the diagram: the organizational layer.

Within the *organizational layer* are the processes which locate and move data according to demands of correctness and efficiency. It is not the content but the size and the position of the data element which is of concern. File systems provide tools to fetch records containing data according to relative position or name. This layer is analyzed using quantitative tools in order to provide good performance estimates.

The two central layers are highly dependent on each other. The descriptive layer can easily specify requirements that the organizational layer cannot reasonably supply using any given set of hardware. On the other hand, a file organization that expands the basic hardware capabilities only minimally may impose restrictions that prevent the descriptive layer from being effective. The separation of the two inner layers assures that logical requirements are kept distinct from the performance issues.

Table 1-2a Layers of Database Structure: conceptual and descriptive

Layer 1 In the *conceptual layer* we deal with *information*, data with sufficient significance to make us act. Information has the essence of newness and nonredundancy so that its acquisition increases our knowledge.

A: Some processes used to obtain information:
- Cross tabulation and exception reporting
- Mathematical and statistical data analysis
- Linear and dynamic programming
- Inference using logic and artificial intelligence

B: Intellectual tools for managing information:
- Intuitive and formal models of the application area
- Semantic data models
- Information theory

C: Required for the production of information are:
- Named and described data
- Means to collect needed data comprehensively or as representative samples
- Means to analyze such data, with intelligent interaction

Layer 2 In the *descriptive layer* we deal with *data*, text and numbers describing items and events.

A: Processes used to obtain and manipulate data:
- Data collection
- Data organization
- Data selection and retrieval

B: Intellectual tools for data manipulation:
- Logic
- Data modeling
- Computer science concepts of syntax and semantics
- Knowledge of formal languages and meta-languages
- Propositional and relational calculus

C: Required for large-scale data manipulation are:
- Named files
- Named or addressable records
- Addressable data elements
- Catalogs and descriptions of available facilities
- Facilities to obtain processing resources from the computer system
- Procedure libraries

Table 1-2*b* Layers of Database Structure: organizational and material

---

Layer 3 In the *organizational layer* we deal with *data representation,* data removed from context, but named and processable.

A: Processes used to encode data and manipulate the encoded data:
- Encoding and decoding
- Storage layout definition
- File management
- Space management for files
- Moving and comparing of fields containing coded data
- Control of errors

B: Intellectual tools for the design of data representation and manipulation:
- Algorithms to estimate performance
- Programming principles
- Knowledge of computer architecture and component performance

C: Required for the representation and manipulation of data are:
- Storage devices for encoded data
- Operating system services to move blocks of data between memory and storage
- Processor capability to compare and manipulate encoded data

Layer 4 In the *material layer* we find *hardware,* the equipment used to provide unformatted addressable storage.

A: Processes used to produce hardware:
- Engineering design
- Manufacturing capability
- Maintenance procedures
- Market research

B: Tools required for the engineer who is designing hardware:
- Knowledge of electronics, physics, and logic
- Knowledge of manufacturing practice

C: What the engineer in turn has to rely on:
- Availability of components, devices, and measurement tools
- Availability of production facilities

The programs which define the applications of a database will be a reflection of the decision maker's concept of the information problems being faced. They will vary over time as the needs, the analysis tools, and the insight of the users of the database mature. During this time the operating systems and computers which support the files may evolve and change. We therefore want a clean separation or interface between the layers.

To ensure that the separation of layers is not evaded, systems which support the concept of layering hide the internals of lower layers from users at the higher layers. This idea of *information hiding* is valid only when the functions provided by the supporting layer are reliable, complete, and reasonably efficient. The independence of the descriptive and organizational layers is also referred to as *data independence.*

1-6 BINDING

¡¡ better in chapter 2? ¿¿ Many people in data-processing still accept that the contents and the structure of the database have to be fully determined before procedures that operate on it can be written. This assumption defeats one of the major advantages that can be accomplished when we consign processes to computer technology. When designing and implementing a database system, it is desirable that methods be employed that allow development and change, so that we are not wholly dependent on perfect foresight. The fact that all our data and procedures can be manipulated by computation should make it possible to add data and relationships to the database at a later time. The operations involved in such a change may not be trivial, but a system that takes advantage of such growth possibilities can avoid obsolescence.

Whenever a decision is implemented which defines a fixed relationship between data elements, the number of choices available from that point on is reduced. Some knowledge has been used to make processing feasible. The process of fixing data or processing relationships is referred to as *binding*. At the final stage of data processing all relationships needed to generate information have to be bound.

BINDING TIME The concept of binding is in common use to distinguish compiling, *early binding*, versus interpretive, *late binding*, processes. The range of binding choices in the database area is considerably greater than in the area of programming. Binding frequently takes place years before a new database system produces its first results.

Binding choices can be illustrated by expanding Example 1-1. Let us assume that an employees classification is not found in a single file (`job_classification`), but that there is a file for each department, and the names of these files are in an array `dep_class_filename`. Statements `s3` to `s5` then become a loop which appends the lists together.

¡¡check for equal blank spacing ¿¿

```
      ...
      no_employees = 0;
      DECLARE dep_emp_list(1000) INTEGER, dep INTEGER,
        dep_class_filename(100) CHAR(8) EXTERNAL, filename CHAR(8);
 s3:  DO dep = 1 TO no_depts;
         dep_emp_list = 0;
         filename = dep_class_filename(dep);
         OPEN FILE(job_classification) TITLE(filename);
 s4:     READ FILE(job_classification) KEY(class) INTO(dep_emp_list);
         CLOSE FILE(job_classification);
 s4a:    DO i = 1 TO 1000 WHILE dep_emp_list(i) ¬= 0;
            no_employees = no_employees + 1;
            emp_list(no_employees) = dep_emp_list(i);
 s4b:    END;
      END;
 s5:  PUT SKIP ...
```

Figure 1-3 Using departmental files for the "raise" transaction.

This program is bound to the assumption that an employee, including a manager, is listed in only one department file. This constraint will have to be enforced when the departmental employee lists are created. Otherwise, if one manager becomes responsible for two departments, raise will give this manager a 21 percent raise instead of the 10 percent raise. Inserting after s4a the statements shown in Fig. 1-13 releases the binding and resolves the problem at the time that raise is executed. An analysis of the program's execution will show that now retrieval processing takes much more computational effort. Late binding can be especially costly because it is done at a critical time, when the answer is needed, and because it has to be done on an element-by-element basis, rather than once and for all when the structure is defined.

```
/* check for duplicated employees */
DECLARE id INTEGER;
DO id = 1 TO no_employees;
   IF dep_emp_list(i) = emp_list(id) THEN GO TO s4b;
END;
```

Figure 1-13 Verification of database state in a transaction.

Early binding may be chosen because it can provide considerable initial efficiencies. In a database a significant amount of binding occurs when the designer specifies the position of data elements within the files. Later modifications of bound structures will be extremely awkward and often cause inefficient processing when the opposite was intended. When new data types or relationships are to be added, a major reorganization of the file structure and modification of the associated programs may be required.

¡This situation is analogous to the making of machine-language changes in the output of compilers. Such situations are no longer acceptable in programming because of the low cost of recompilation; in the area of file management, the patching of file structures is still common. An example of such a patch is given in Fig. 1-4.¿

To avoid the cost of a major change a programmer may decide to put additional data elements into positions originally intended for other purposes.

Given that the records of a personnel file have fields as follows:

```
name,address,sex,salary,military_service,date_employed;
```

and a field for maternity_leave is required. The records are used by many programs, even though only a few of them use the military_service field. The patch adopted is to use this field for maternity_leave whenever the sex field indicates a female employee.

```
IF sex='F' THEN maternity_leave = military_service;
```

Figure 1-4 A database patch

In this example the decisions made at an earlier binding time determine the implementation of later system features. The patch will cause further problems as progress in sexual equality invalidates the assumptions made.

The lack of proper programming methodology and the pressures to patch rather than to do changes properly are a cause of frustration in many computer shops. The consequences are often cynicism and low productivity of professional personnel. Much of the design guidance presented in this book has as objective that binding decisions will be made in a knowledgeable manner. Both the users and the system maintainers have to be satisfied. The choice of binding time made by a database designer reflects a compromise between efficiency and flexibility.

1-7 APPLICATIONS

Most readers will have encountered databases in use, at least indirectly: when buying an airplane ticket, when consulting a library catalog, or when receiving a bank statement. Whether a database management system or a specific program provides the service is rarely clear to the user. In most applications the functions are provided jointly. This book should help define the interface as well.

Application areas that strongly depend on database systems include:

- Manufacturing with inventory management, Bills-of-Material processing, and production equipment scheduling
- Government at all levels with records on taxpaying individuals and property
- Financial institutions with lists of individual accounts, assets, and convertibility of funds
- Service industries with lists of service capabilities and allocation schedules
- Medical services with patient records, disease histories, problem classification, treatment effectiveness data
- Economic models with production and consumption data for allocation and planning
- Scientific research with collections of previously gathered data used to determine future research directions
- Offices which are automating their document management
- Libraries cataloging abstracts and indexes of their holdings

The background and references section cites some examples from each category. It will be useful to gain some understanding of the uses to which these systems are put, in order to obtain a perspective of the problems discussed. We note that information systems based on data in text form, often used for automated library support, tend not to use structured data but depend greatly on searches for keywords in titles, abstracts, and author lists. This book deals mainly with structured data, so that it is less relevant to applications that are based on content analysis.

1-7-2 A Classification of Database Application Types

While all databases have many functions in common, the relative emphasis can differ a great deal. Table 2-1 provides a categorization of database applications, and the expected emphasis for the types recognized. For these types different database management systems have been developed, although there is also a strong ¡tendency¿ to have database management systems that can serve a wide variety of tasks. Although the underlying functions remain the same, these applications differ in terms of criteria, so that the database systems that can serve them best will differ as well. ¡clarify the above¿

Table 1-2 Database application types.

Application type	Characteristics
Inventory Management (Bill-of-materials)	Large files, many small transactions Moderate number of on-line users High integrity requirements
Financial management	Large files, many small transactions Large number of inexpert users Very high integrity requirements
Airline reservations	Large, simple files, very many small transactions Very large number of expert and inexpert users Modest integrity, but high availability
Engineering support	Complex files, long transactions Few, demanding expert users High reliability and backup required
Information retrieval	Large textual files, variable transactions Mainly inexpert users Centralized update by experts
Personal record Keeping	Simple files, variable users Single, inexpert users Robustness required
Medical records	Complex files, variable transactions Demanding inexpert users Robustness and availability requirements
Plant and instrument control	Simple files, continuous Usage by monitoring programs Fail safe requirements

1-7-3 Bill-of-Materials

The management of inventories in manufacturing, particularly where products consist of many parts and subassemblies, has been one of the earliest and most productive areas of database systems. A particular objective of these *Bills-of-Material* systems is the proper accounting and order scheduling of parts that are components of diverse end products and hence particularly critical. We will draw many examples from personnel files, since their contents and function are frequently self-evident. More interesting and potentially more significant applications are in food and energy resource planning.

DEFINING OF AN APPLICATION In any application of computers, the purpose of the effort should be spelled out in detail. Only then can one identify and judge the relevance of the data elements contained in the files, the processes required to manipulate the data, and the file structures which will enable the processes to be effective. Knowledge of the environment is needed to determine the value of the produced information. We also have to know the response time intervals required to ensure timely delivery of the system outputs. All these items provide the data for system analysis and the subsequent selection of the hardware. The dynamics of the application environment will provide useful information to decide the degree of formalization of the data organization and the corresponding binding times.

A statement of objectives and constraints, which specifies the goal and environment of the application, is an essential part of any project carried out in practice. Such a statement should also be presented with any project carried out as an exercise within the scope of this book. A frequent error in the formulation of objectives is the inclusion of the method or technology to be used as part of the objective. A proper place for such an account is the section which describes the results of the system analysis.

Objective statements which imply that the desire to *computerize* was the reason for development of a database should send a shiver down the spine of any computing professional. If the choice of methods or technology has been intentionally limited, a separate section of the objective statement should clarify such conditions.

1-9-1 Overview of the Material to Be Presented Chapter 2 surveys the infrastructure needed for databases. A summary of hardware and its parameters is followed by a software system summary which stresses transaction systems.

Chapter 3 presents the structural concepts of databases. The relevant abstractions are relations and connections among the relations. In Chap. 4 we cover the operations neccessary to manipulate such a structure and in Chap. 5 we apply these concepts to the design of databases. In order to implement a database on a DBMS we must descibe the design decisions; this is done using a schema, as presented in Chap. 6.

In Chap. 7 we switch to DBMS software, and describe the components which make up a DBMS. With this background we can begin to discuss implementations. In Chap. 8 we present implementations of teh relational models, and in Chap. 9 we present implementations which also implement connections. Current trends in implementation are covered in Chap. 10. The discussion in theses three chapters proceeds in the direction of increased binding, which causes loss of flexibility but increases the performance of systems.

In Chaps. 12, 13, 14 some important implementation issues such as reliability, access protection, and integrity are covered.

We concentrate on concepts, and do not present complete system descriptions, although many actual systems will be referenced to allow further study. Appendix B can be consulted for references about the systems named in this chapter. The exact syntax of examples based on these systems has been modified at times in order to provide continuity. The fact that a certain system is cited here as a commendable example does not imply an endorsement of this implementation for a given application but only reflects on the values of the concepts being discussed.

BACKGROUND AND REFERENCES

The background and reference sections of each chapter provide both historical material as will as citations to further work. The history is provides to help the reader gain insight into why developments took place the way they did, to complement the text, which focuses on what databases are about. Some of the material is based on a summary of the history and status of databases which appeared as part of an IEEE centennial issue [Wiederhold^{84}].

A Short History Most of the elements required for today's databases were already envisaged by Vannevar Bush in a 1945 paper in describing MEMEX, a desk which provided storage of information, tools to process the information, means to anotate findings, and the capability to store the results Bush[45] ¡¡, reprinted in Kochen[67] and later ¿¿.

We have not yet achieved the flexibility and user-friendly interaction that Bush envisaged. There are good reasons for the delay. Database concepts could not be implemented until a number of developments in computer technology had achieved an adequate state. The infrastructure is composed of hardware, operating systems, and file services. These requirements started being satisfied around in the eraly sixties, and some systems originally conceived at that time, are still available today. Background for the infrastructure is provided with Chap. 2.

The development of magnetic disks led to reliability, econmomy, and random access storage. Theses factors enabled database technology by providing persistence of data, and persistence and on-line access permitted users to share the data.

Databases and Communication The shared database can best be viewed as a communication medium, which operates asynchronously. Each participant can enter data as it is generated and obtain information when needed. Having data distributed over multiple computers is only a logical extension of the database concept.

Handling and sharing of voluminous data requires an understanding of the meaning and structure of the data. This understanding must be shared among users, designers, and implementors. The capabilities provided by the hardware encouraged information modeling. Models became the prime focus of database research and development in the seventies. The model concept is so pervasive that people will talk about a *relational model* when they refer to an actual implementation. We will carefully distinguish models and implementations throughout, because modeling concepts have validity that go beyond particular styles of implementation, and implementors must face many issues not addressed by the models.

THE EMERGENCE OF DATABASE TECHNOLOGY Database development began with applications and was triggered by advances in storage techniques. This sequence, where technology triggers development of applications, and universities follow with generalization and education, has dominated computer science advances Hammer[82].

Some early information system applications addressed medical documentation (MEDLARS, Katter[75]), airline reservations (SABRE Desmonde[64]), and governmental planning (Clark[67]). Riley[81] has collected a number of early management applications. Expectations of effective use were tempered by disappointments and reassessments (Aron[69], Lucas[75])

Inventory applications, while demanding, were well understood. Here data about parts and subassemblies have to be accessed for a variety of products and those data have also to be managed by part location and by supplier.

¡¡¡¡¡¡¡¡ end edit BG ¿¿¿¿¿¿¿¿.

Database Management Systems The infrastructure for the implementation of a DBMS was in place in the early sixties and the unique elements defining database technology were incorporated into systems developed at that time. Several DBMS's which are still in use today had their origin in that period, although a substantial body of new knowledge and techniques has emerged since then. Certain older systems assimilated some of these advances, but became quite complex in the process. In settings where the staff is capable of dealing with the complexity these systems now provide a high level of performance and reliability.

The first DBMS, the General Electric Integrated Data Store (IDS), was used for an application in 1961 Bachman[74]. The system was further developed and saw wide distribution by 1964. Initially IDS provided mainly a programming interface to assure correct manipulation of the complex structures. Graphic techniques, later called Bachman diagrams, were developed to explain the underlying structure at an abstract level. When the General Electric computer operations were taken over by Honeywell this line of work was continued.

In the last ten years many newly developed DBMS's appeared which exhibit a simpler structure. The problem of acceptance of their new technology appears to follow the trend seen in programming languages. Simpler and more powerful approaches emerge and are used in newly established settings. The the experience obtained with older systems leads to a fear of disruption, discouraging the investment needed to replace systems which are established.

```
year disk              Events                         —
cap. —  [studies]  (fundamentals)   ¡applications¿        —
—-——————————————————————————————————
1945     —             [MEMEX]                         —
1946     —                                             —
1947     —                                             —
1948     —                [Information Theory]            —
...    —          (digital magnetic recording)          —
...    —                                            —
1956 128KB —                                           —
1957     —                                             —
1958¡RAMAC¿—     (hashing)                                —
1959     —(multi-lists)  (schema)  ¡inventories¿  (indexed retrieval)—
1960     —               ¡Airline reservation SABRE¿         —
1961     — ¡Bill-of-materials¿   [Inf.Retrieval studies]          —
1962 ¡1301¿—           [Information Algebra]                —
1963     —                  ¡Medlars¿                     —
1964   7MB —      GE IDS          NorthAm.Rockwell IMS           —
1965     —          (semaphores)        [SDC survey symp.]      —
1966     —       GM APL                                   —
1967     —          DM-1/SC-1              SDC TDMS           —
1968     —   [first courses]      IBM IMS        [Set Model]    —
1969     —  [CODASYL DBTG]                                     —
1970  29MB — [SHARE DB committee]      (B-trees)     [Relational Model]
—
1971     —                    (transactions)               —
1972     —    [privacy]                  [Courant Conference] —
1973  58MB —          System 2000  [SHARE conf.]                —
1974     —    Adabas   [paper collection]   [IBM System R project]   —
1975       — [New DBTG report]  QBE   [VLDB started] [ANSI-SPARC
report]—
1976 100MB — [ACM TODS begun]  [first textbooks]  [database machines]
—
1977     —(intense discussions:network vs.hierarchical vs.relational)—
1978 200MB —   [CODASYL internal schema]     [privacy legislation]    —
1979     —    [concurrency control           several relational      —
1980 2.5GB —         activity]   (universal     database systems      —
1981     —         DAPLEX          model)      IBM SQL/DS          —
1982 floppy—¡—micro-computer          [Relational Standard]          —
1983  1.2MB—      databases          micro access to large DBMS        —
1984     —         local communicating databases                 —
1985     —                      public access networks           —
1986     —    workstations                                 —
1987 optical discs                                        —
1988     —         local communicating databases                 —
....—......—..........................................................—
cap. —  [studies]  (fundamentals)   ¡applications¿         —

¡¡¡ use distinct colors for [studies], (fundamentals), ¡application¿ entries ¿¿¿
```

Figure 1-99 Events in the database world.

Parallel work on a system called the Associative Programming Language by George Dodd at the General Motors Research Laboratory provided similar facilities on IBM equipment Dodd[66].

The Seventies: Rapid Growth The term Management Information System (MIS) had become popular, and publications appeared which promised much greater capabilites than the underlying technology could deliver. The conflicts raised concern in the data-processing community. Eventually, in 1970, the IBM user groups, SHARE and GUIDE, published requirements documents which included notions of interfaces, program independence, and privacy protection.

Research Contributions As stated in the introduction, the interaction of industrial and research development has followed a typical path in Computer Science. Initial contributions are made in industry, often by people with very respectable academic backgrounds. When the scope and the problems become significant, researchers in academia and a few industrial research laboratories begin to formalize the field. The formalization leads to generalization and to better and cleaner solutions to the problems. These solutions are published as research papers and eventually there is a sufficient body of material so the subject can be taught. Teaching is first done with collections of papers and subsequently books written specifically as course texts appear on the scene. The field gains respectability and is integrated into the curriculum.

In databases the milestone formalization was provided by Ted Codd of IBM research Codd[70]. ¡¡refer to model chapter ¿¿

The relational model rapidly generated attention. Conference series were expanded and established that dealt exclusively with databases. Noteworthy was the ACM special interest group on File Definition and Translation, now renamed Management of Data (SIGMOD) Beginning in 1974, the database Working Group 2.6, a part of the IFIP TC-2 committee. Problems specific to *very large databases* were to be the focus of a conference series by that name intiated in 1975. By that time many practioners had became aware of relational concepts and began incorporating them in their thinking and implementations.

The concepts provided by the relational model also made it easier to teach courses on databases. Notes were developed from those courses and the first database textbooks were published in 1976. Authors of most early books were associated with IBM, but later authors came from academia as well as from industry. Database research projects were started at several universities. These projects often stressed generality and simple query interfaces.

¡¡¡ where ¿¿¿ Some systems stressed simplicity to obtain ease of use and fairly high performance. An example is the TOD system and its descendants, which provide a simple relational view oriented towards statistical data analysis Wiederhold[75].

Models and Modeling The relational model describes the contents of a database as a collection of independent relations. The clarity of the model motivated a widespread interest in relational implementations.

¡¡¡modeling taken out here¿¿

The Current State Databases have become ubiquitous.

Database Interfaces Early databases required a high level of expertise from their users. The users were programmers who had to understand data management concepts, they had to understand programming, and they had to understand the application that had to be placed on the DBMS. The only services provided by the early DBMS consisted of a suite of input- output programs and the management of multiple user interactions.

Today the users can deal with a very abstract conceptual view of the database that they are accessing, using the model provided by the DBMS. A request for information can be submitted using some matching formalism. Most modern database languages accept statements that describe WHAT information should be obtained, avoiding instructions which state HOW it should be done. DBMS's have satisfied the conditions required to bring computing capabilities to the public. Database queries are one of the few areas where non-procedural languages are in regular use. Their success is due to the ability of simple models to describe what is relevant and hide underlying details.

Complementary Books Many textbooks are now available which describe (Sanders[74], Kroenke[78], Atre[80]) or analyze (Davis[74], Everest[82]) management-oriented applications.

Application References environmental control (Ouelette[75]), and to the study of history (Schneider[75]). The use of computers in information systems oriented toward bibliographic data is described in books by Salton[75] and Lancaster[79]. Chemical data (Anzelmo[71]) have long been cataloged, and applications in clinical medicine have a long history (Allen[66], Davis[70], Yourdon[72], Weyl[75], Barnett[79], Wiederhold[81]). Motivation for scientific databases is provided by Lide[81]. This list can be extended further by use of references in the works cited.

PERIODICALS The activity in the database area has led to the publication of several journals devoted to database subjects. The rapid development in this field makes bibliographies and descriptive books rapidly obsolete. We will use this space to indicate sources for current material to complement the background sections of this and the earlier chapters.

Major journals in the database area are *Informations Systems* (since 1974) and the ACM *Transactions on Database Systems* (TODS, since 1975). Other database-oriented publications include the *Database Journal* (since 1975) and *Management Datamatics*, (since 1971). IEEE Software Eng, IEEE trans. General TOOIS, British Comp journal, ¡¡update from FODD¿¿

Newsletters in the area are *Data Engineering*, produced since 1977 by the IEEE Computer Society technical committee of that name and the *ACM-SIGMOD Record* of the ACM Special Interest Group on Management of Data. From 1968 to 1976 it was named SIGFDT, for File Definition and Translation.

ACM-SIGMOD sponsors annual conferences since 1975. The proceedings contain many important papers, as seen in the bibliography.

¡¡remove duplicate citations here ¿¿

(i.e., Codd[71], Dean[72], Rustin[74], King[75], Rothnie[76], Lowenthal[78], Bernstein[79], Chen[80]).

An international series of conferences, starting in 1975, on *Very Large Data Bases* (VLDB) cover all database topics. Since 198¡x¿ the Data Engineering TC of the IEEE Computer Society has sponsored conferences which intend to cover data-oriented systems from an engineering point of view, although much overlap is unavoidable. The British Computer Society has sponsored database conferences since 198¡x¿, and a European series has started with a meeting in Venice in 1988. In addition, a number of more specialized conferences take place. some of them deal with applications, other with technical areas where the overlap with databases is great. These can best be located through individual citations in the chapters dealing with such topics.

The introduction to the bibliography provides publication information for these periodicals. ¡verify¿

EXERCISES

1^p Write a statement of objective for one of the applications mentioned in Sec. ¡1-7.¿ Select a subject area with which you are familiar and in which you are interested. Many other exercises in this book can be based on the same application.

> This is the initial step of a project carried throughout this book, and also used in its predecessor. All exercises relating to the project are marked with a ^{p} symbol. At the end of the course the project is submitted in the form of a proposal to an enterprise manager to define a database and write the required file programs for this application.

2 Provide a list of data elements which you think would be included in the application presented above, and justify their value with a short sentence for each.

3 Identify which elements are likely to have a dynamic relationship or function and hence will cause definitional problems.

4 Inspect a file-oriented program that you have access to and select statements which typify some of the classifications given in ¡¡Table 1-2.¿¿

5 Rewrite the control-language statement presented in ¡¡ Example 1-2¿¿ in the language of the computer system you have available.

6 Write a query to the database with the patch shown in Fig. 1-4 which would produce an error.

7 Your are to design a non-trivial, non-grandiose database for any application of your choice, real or imaginary. If you are unsure of the appropriateness of your idea, or if you need some help with ideas, or if you need help with anything, please contact the Professor or the T.A.'s.

You may work alone, or in pairs; if you work with someone do steps (2.1), (3.1), and (5.1) as well.

By "design", we mean, write up (clearly) these parts:

(1) The general description of the application, including objectives, expected data sources, uses (query types) and desired performance on query types.... All of this should be made at the "management-type-user" level.

(2) Describe what entities you cover and their relationships at the management level.

(2.1) Do this from more than one point of view.

(3) Give examples of 2 or 3 important queries and their performance requirement.

(3.1) Do this for each view.

4 What is the distinction between a database and a database management system?

9 Give an example of a very large database as defined in Sec. 1-1-1.

SOLUTIONS

1-1

1-3 xxxxx