INSTRUCTOR'S MANUAL TO ACCOMPANY

Database Systems: Design, Implementation and Management

Third Edition

PETER ROB

CARLOS CORONEL

COURSE TECHNOLOGY

Instructor's Manual to Accompany Database Systems: Design, Implementation, and Management, Third Edition is published by Course Technology.

Editorial Assistant	Lisa Ayers
Manufacturing	Sean Marr Elizabeth Martinez
Cover Designer	Susannah Lean

ISBN 0-7600-4905-X

Printed in the United States of America.
10 9 8 7 6 5 4 3 2

Please Read This Before You Begin

The Purpose Of This Manual

We realize that the third edition of ***Database Systems: Design, Implementation, and Management*** covers a lot of ground and includes a wealth of practical detail. Therefore, we suggested in the book's preface how you might the select topical coverage to fit different database course descriptions and/or different teaching interests. However, regardless of how you have decided to cover the book's material, we believe that using this manual can make your job easier and more exciting.

Database design and implementation is not something your students will learn by merely reading the text material and listening to lectures. Instead, success in this arena requires hands-on work and lots of practice. This manual is designed to let you help your students by suggesting a discussion focus for each chapter and by providing detailed answers to questions and problems. The five appendixes provide a map to the databases we have created for you, a procedure for exporting database tables, a guide to application building, some basic report formats, and an extensive transparency master set.

The Contents of this Manual

To meet the objectives we established for the Instructor's Manual, we present each chapter's material in this format:

1. Primary chapter objectives
2. Discussion focus
3. Answers to review questions
4. Answers to problems.

In addition to covering each chapter's questions and problems thoroughly, we also provide coverage of basic applications development and database export in the appendixes. Given the appendix coverage, teachers can use a microcomputer database such as Microsoft Access as a front-end to more sophisticated databases such as ORACLE. This arrangement can then be used to explore the client/server environment.

Throughout this manual, we have used **teacher's notes** to alert you to special problems, to suggest discussion topics, to suggest additional or modified topical coverage, and to suggest alternative problem solutions.

For more information regarding this Instructor's Manual, please call your Course Technology, Inc. sales representative or contact customer service at (800) 648 7450.

Special Features

You have probably noticed that this instructor's manual is rather hefty. Like most instructor's manuals, this one contains answers to the text's questions and problems. However, ***because we focus on the practical aspects of database design and implementation, our answers must contain much more detail than the answers commonly provided in instructor's manuals***. This wealth of detail means that the manual truly *augments* the text.

Because real world database design is based on very detailed descriptions of an organization's operations, each of the database designs shown in this manual may be used in two ways. First, examine each database design as a solution to the problem. Second, use each design as the basis for discussions about different ways to model the database requirements. Explore the semantics of the problem description. Keep reminding students that successful database design usually means having to satisfy conflicting goals: design elegance, information requirements, transaction speed, and so on. Also, examine how a simple change in wording can have a major effect on design. For example, can a particular part be ordered from just one vendor or from many vendors?

Quite aside from the fact that this manual contains very detailed question answers and problem solutions, we also offer the following features:

Databases Are Included

Because we focus on the practical aspects of design and implementation, we have made sure that all the databases, tables, relationships, and queries shown in the text, the end-of-chapter questions, and end-of-chapter problems are available to the teacher via the CD supplied to adopters.

Finding Each Chapter's Databases

To help the teacher find the appropriate database materials easily, we have organized them by storing each chapter's contents in its own directory. For example, Chapter 1's materials are stored in the CH1 directory, Chapter 2's materials are stored in the CH2 directory, and so on. Appendix 1 lists the available databases and the tables within them.

Database Naming Conventions

We have used database names that are, to the greatest extent possible, indicative of their function in each chapter. If the material is based on single tables and modifications of such tables, they are stored in databases whose names indicate their location in the chapter. For example, the database named CH1_TXT, located in the directory CH1, contains Chapter 1's general text illustrations. Similarly, the database named CH2_QUE, stored in the directory CH2, contains general material used in Chapter 2's question set. The database named CH3_RVEW, stored in the directory CH3, contains material pertaining to Chapter 3's end-of-chapter review section. Using the same convention, the CH1_PROB database, stored in the directory CH1, contains Chapter 1's problem material. If the

manual's discussion introduces new databases, they are stored as IM#_DISC. For example, the database introduced in the manual's Chapter 5 discussion is stored as IM5_DISC.

When the database material becomes more complex, including multiple tables, relationships, and queries, we have named them after their specific location in the text. For example, the database named CH1_P5, stored in the directory CH1, contains the material for Chapter 1's problem 5.

Finally, if complex database material is used in more than one section of the chapter or even in different chapters, the database name reflects its contents. Therefore, Chapter 2's invoice discussion material in found in a database named CH2INVOI. (Note: Database names are limited to eight characters.) Similarly, the extensive aviation database problem in Chapter 2 makes reference to the database named CH2_AVIA, Chapter 8's COMP_LAB database contains the extensive computer lab material, and the COLLEGE database contains college examples.

Tables within the databases are named to reflect the entities they represent. For example, the AGENT table contains the data for agents, the CUSTOMER table contains customer data, the AGENT_INI contains initial agent data, EMPLOYEE contains employee data, and so on.

Finally, the databases used in Chapter 13, The Data Warehouse, are to be distributed to students. The database containing the solution has an "S" extension. For example, DW-P3.MDB contains the student database for the data warehouse problem 3, while the DW-P3S.MDB contains the same data plus the dimension and fact tables required in the solution.

Database Format and Conversion

The databases and tables are available in the Microsoft Access 2.0 format. This format was selected for several reasons. First, Access 2.0 is easily and cheaply available. Second, Access 2.0 is widely used. Third, Access 2.0 is a Windows product that runs well under either Windows 3.x or Windows 95, so its use remains flexible. Finally, Access 2.0 structures are easily convertible. For example, if you use Access 7.0, the Access 2.0 structures will automatically be converted to the proper Access 7.0 format. If you use some other database software, the table structures and their contents are easily converted through the use of the Access Export facilities, or they may be converted while being imported through your own database software. We routinely export Access tables to Oracle, using the Access export function. See Appendix 2 to see how the export function is used.

Questions and Problems Are Shown With Their Answers

When using instructor's manuals in our courses, we have often been frustrated by having to flip back and forth between the book and the manual to keep track of discussion and problem segments. Many other instructors apparently share our frustration. For this reason, we decided to reproduce the book's questions and problems in this manual. So, in this manual you will be looking at each question and its answer and at each problem and its solution. This approach is especially useful when we present

solutions to problems whose technical details are referenced during the solution process. We hope that this simple service helps make your teaching life simpler and more rewarding.

Design Cases and Class Demos

Many of the problems presented in **DATABASE SYSTEMS** are sufficiently complex to serve as the basis for excellent design cases that may be handled by individual students or by small student design teams. Using the format and the procedures presented in Chapters 7 and 8 as a template, you can easily set up a real-world problem of special interest to you.

Depending on your students' background, you may let the students develop the

1. initial E-R diagram

2. final E-R diagram and the data dictionary

3. step 2 plus the implementation, using Structured Query Language (SQL) or your DBMS applications development software.

4. step 3 plus applications development, using the DBMS of your choice.

If you elect to carry the course all the way through item 4, you can let your students use application development software to prototype a simple menu system and some reports, or you can let your students use programming techniques to develop applications for some or all applications you deem appropriate. In our senior-level database course, we require students to produce a complete design, implementation, and applications development package, using the following components:

Cover Page

Main Body

1. Letter of Transmittal
2. Table of Contents
3. End User Manual

Technical Appendixes

1. Description of Operations
2. Business Rules
3. Initial E-R Model
4. Normalization
5. Final E-R Model
6. Relational Schema
7. System Organization
8. Table Structures and Contents
9. Query Documentation and Output
10. Forms and Form Documentation
11. Report Output and Report Documentation
12. Macros and macro Documentation

Video Demonstrations

We have prepared video demonstrations in which we examine various aspects of database design, implementation, and applications development. These videos are included on the CD that accompanies this manual:

- **Implementation of a Database Design and Basic Applications Development.** This video illustrates a database design from its E-R model to its implementation. It also examines the end user view of the database through some sample applications, using a menu-driven interface. Because the presentation is based on two quizzes, one dealing with database design and the other dealing with implementation and applications development, this video may be used as a template for the development of hands-on quizzes and tests.

- **Exporting MS Access 2.0 Tables To Oracle.** Because database prototyping is often done with the help of microcomputer RDBMSes such as Microsoft Access, we show how database tables produced in Access can be exported to an "industrial strength" RDBMS such as ORACLE. Because the relationships are not exported, we also demonstrate how such relationships can be re-established in Oracle.

- **Basic ORACLE SQL.** We show how Oracle SQL may be used to extract data from an Oracle database and how to do basic buffer and screen management. We then illustrate how SQL commands can be stored and edited with the help of a user-selected ASCII editor such as Notepad and how such stored SQL commands may be executed. We also demonstrate the use of a script file containing multiple SQL commands. A selection of Chapter 3's SQL problems is demonstrated. (All of Chapter 3's SQL problems are supplied as ASCII files, ready to be executed.)

- **Using Access 2.0 As a Front End To Oracle.** Because many applications are prototyped through the use of microcomputer RDBMSes such as Microsoft Access, we show how Oracle tables may be attached to an Access database object. We then demonstrate how Access may be used to easily develop the forms, including form/subform formats, queries, and reports may be generated. This approach ensures that the data are maintained and managed at the server level, while the end user applications are quickly and easily developed at the client level.

Extensive Test Bank

A comprehensive test bank is included on the CD that accompanies this manual.

Presentation Graphics

Presentation graphics are included on the CD that accompanies this manual.

Acknowledgments

We owe a debt of gratitude to Mr. R. Shaun McKee, who helped clean up some of the large database tables we use in our data warehouse exercises. He also used our aviation data as the template for generating additional records, thus improving another data warehouse exercise. In short, he helped us be more effective teachers by providing the necessary classroom and lab support.

Course Technology's Lisa Ayers transformed our manual's collection of loose pages into the Instructor's Manual you are about to use. Lisa made sure that our job was truly (and finally!) finished when we submitted these pages. Not having to worry about the final production details greatly simplified our lives. You have earned our gratitude and our thanks.

We especially give thanks to our better halves, Anne and Victoria. They believed they had us back after the completion of the **DATABASE SYSTEMS'** third edition, only to lose us again to the very time-consuming writing of this manual. We thank them for their grace and understanding.

Peter Rob
Carlos Coronel

Table of Contents

CHAPTER 1

File Systems and Databases

Primary Chapter Objectives

1. To develop the main database system concepts.

2. To show the evolution of database systems from computer file systems.

3. To show that database *design* is a crucial first step in the development and proper use of database systems.

4. To understand the main DBMS functions.

5. To describe the database environment.

6. To examine different types of database systems.

Suggested Class Discussion Focus

What is a computer file system, and what are its problems?

Emphasize the idea that a computer file system stores data in independent, unrelated files on disk. The sharing, security and integrity of the data can't be enforced efficiently because of data dependence and data redundancy problems.

What are the DBMS functions, and why are they important?

Direct the students' attention to the text's section 1.4, then discuss how the DBMS functions address each of the file system's problems. Emphasize that a proper database system eliminates data dependence and improves data integrity by minimizing data redundancy.

Describe the database environment and the different types of database systems.

Explain that the DBMS is an important component of the database environment, but that this environment also includes different types of hardware and software, people who perform different functions within the environment, procedures designed to accomplish desired activities, and data. The data constitute the database's central component through which information is generated. Emphasize

that the main purpose of the database environment is to help an organization to perform its mission and to achieve its goals.

Examine the different types of database systems in terms of the number of users, data location(s), and the type and expected use of the data. Discuss the historical path from file system to hierarchical DBMS to network DBMS to the Relational DBMS. This may also be a good opportunity to look ahead to a few chapters in order to at least mention the arrival of Client/Server systems, the impact of the Internet, Object Oriented DBMSes, and the Data Warehouse. We suggest that this preview be kept simple and that it be presented as a way to attract student interest.

What is a Database Model?

Explain what a database model is, what types of database models exist, and how database models represent data relationships.

Emphasize that a database model is a collection of logical constructs used to represent the database's data structure as well as the data relationship(s) found within that structure.

types of database models	***conceptual*** **database models: E-R, Object Oriented, and semantic**
	implementation **database models: hierarchical, network, and relational**

Show how database models represent data relationships: 1:1, 1:M, and M:N.

Why is database *design* so important?

Database design is a process that, in effect, yields a detailed database blueprint. This blueprint, usually in the form of an E-R diagram, is used as the basis for database implementation. A database created without the benefit of a detailed blueprint is unlikely to be satisfactory.

Pose this question: would you think it smart to build a house without the benefit of a blueprint? So why would you want to create a database without a blueprint? (Perhaps it would be OK to build a chicken coop without a blueprint, but would you want your house to be built the same way?)

Explain that modern database-and-applications development software is so easy to use that many people can quickly learn to implement a simple database and develop simple applications within a week or so, without giving design a thought. As data and reporting requirements become more complex, those same people will simply (and quickly!) produce the required add-ons. That's how data redundancies and all their attendant anomalies develop, thus reducing the "database" and its applications to a status worse than useless. Stress these points:

- Good applications can't overcome bad database designs.
- The existence of a DBMS does not guarantee good data management, nor does it ensure that the database will be able to generate correct and timely information.
- Ultimately, the end user decides what data will be stored in the database.

Discuss the main concepts of the three database implementation models; then discuss advantages and disadvantages of each.

Direct the students' attention to sections 1.5.1, 1.5.2, and 1.5.3. Since most current DBMSes are based on the relational model, focus on the relational model's conceptual simplicity compared to the hierarchical and network models.

Answers to Review Questions

1. Discuss each of the following terms:

a. Data

Raw facts from which the required information is derived. Data have little meaning unless they are grouped in a logical manner.

b. Field

A character or a group of characters (numeric or alphanumeric) that describes a specific characteristic. A field may define a telephone number, a date, or other specific characteristics that the end user wants to keep track of.

c. Record

A logically connected set of one or more fields that describes a person, place, event, or thing. For example, a CUSTOMER record may be composed of the fields L_NAME, F_NAME, INITIAL, ADDRESS, CITY, STATE, ZIPCODE, and PHONE.

d. File

Historically, a collection of file folders, properly tagged and kept in a filing cabinet. Although such manual files still exist, we more commonly think of a (computer) file as a *collection of related records* that contain information of interest to the end user. For example, a sales organization is likely to keep a file containing customer data.

Teacher's Note: Field, record, and file are computer terms, created to help describe how data are stored in computers. Emphasize that computer file data storage does not match the human perception of such data storage.

2. What is data redundancy and what characteristics of the file system can lead to it?

Data redundancy exists when unnecessarily duplicated data are found in the database. For example, a customer's telephone number may be found in the customer file, in the sales agent file, and in the invoice file. Data redundancy is symptomatic of a (computer) file system, given its inability to represent and manage data relationships. Data redundancy may also be the result of poorly-designed databases that allow the same data to be kept in different locations. (Here's another opportunity to emphasize the need for good database design!)

3. It is said that file systems lack data independence. Discuss.

File systems exhibit data dependence because file access is dependent on a file's data characteristics. Therefore, any time the file data characteristics are changed, the programs that access the data within those files must be modified. Data ***independence*** exists when changes in the data characteristics don't require changes in the programs that access those data.

4. What is a DBMS and what are its functions?

A DBMS is best described as a collection of programs that manage the database structure and that control shared access to the data in the database. Current DBMSes also store the relationships between the database components; they also take care of defining the required access paths to those components. The functions of a current- generation DBMS may be summarized as follows:

1. stores the definitions of data and their relationships (metadata) in a data dictionary; any changes made are automatically recorded in the data dictionary.

2. creates the complex structures required for data storage.

3. transforms entered data to conform to the data structures in item 2.

4. creates a security system and enforces security within that system.

5. creates complex structures that allow multiple-user access to the data.

6. performs backup and data recovery procedures to ensure data safety.

7. promotes and enforces integrity rules to eliminate data integrity problems.

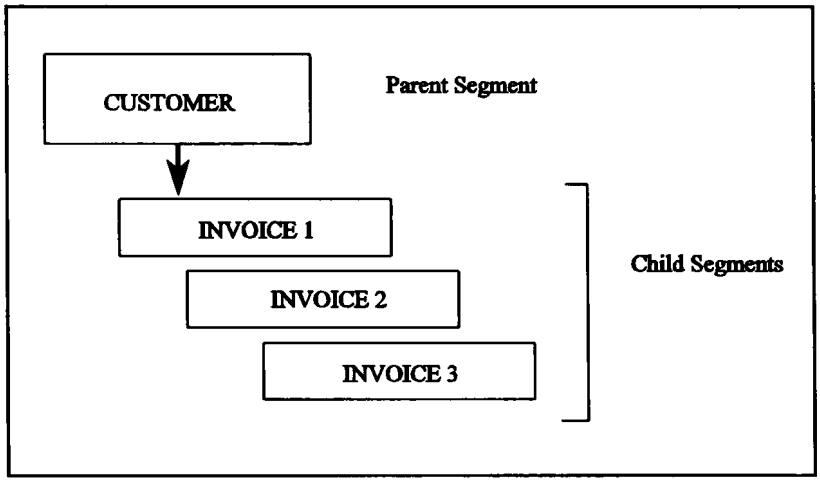
8. provides access to the data via utility programs and from programming languages interfaces.

9. provides end-user access to data within a computer network environment.

5. How does the Hierarchical Database Structure address the problem of data redundancy?

Data redundancy is avoided by using an upside-down tree structure composed of parents and their children. Since a parent can have many children, but each (hierarchical) child can have only one parent, the parent data is identical for all of its children and need not be repeated.

In a classical file system, a customer who has several invoices has his/her personal information re-entered for each invoice. In a the hierarchical database, the information relevant to a customer is found only in the CUSTOMER (parent) segment. For example, the hierarchical structure looks like this:



6. What do each of the following acronyms represent and how is each related to the birth of the Network Database Model?

a. CODASYL

Conference on Data Systems Languages. This group created a COBOL standard; furthermore, CODASYL created the network model specifications.

b. SPARC

Standards Planning and Requirements Committee. This committee augmented the database standards in 1975.

c. ANSI

American National Standards Institute. Adopted the CODASYL database specifications as a standard database model.

d. DBTG

Database Task Group. This group defined an environment to facilitate database creation and data manipulation.

7. What three languages were adopted by the DBTG to standardize the basic Network Database Model and why was such standardization important to users and designers?

The three languages were:

1. The **DDL (schema)** constitutes the Data Definition Language for the database schema. The DDL's use enabled the database administrator to define the database schema, i.e., its over-all blueprint.

2. The **DDL (subschema)** allows the definition of the specific database components that will be used by each application.

3. The **DML** is the Data Manipulation Language that allows us to manipulate the database contents.

Standardization is important to users and designers because it allows them to shift from one commercial application to another with little trouble when they operate at the logical level.

8. What is data independence and why is it important?

Data independence exists when data access programs are not subject to change when any of the file's data characteristics change. Data independence is important because it substantially decreases programming effort and program maintenance costs.

9. Describe the basic features of the Relational Database Model and discuss their importance to the user and the designer.

A relational database is a single data repository that provides both structural and data independence while maintaining conceptual simplicity.

The relational database model is perceived by the user to be a collection of tables in which data are stored. Each table resembles a matrix composed of row and columns. Tables are related to each other by sharing a common value in one of their columns.

The relational model represents a breakthrough for users and designers because it lets them operate in a simpler conceptual environment. End users find it easier to visualize their data as a collection of data organized as a matrix. Designers find it easier to deal with *conceptual* data representation, freeing them from the complexities associated with physical data representation.

10. Compare the file system with the three database systems discussed in this chapter in terms of data independence.

The file system exhibits both data and structural dependency. In contrast, the hierarchical and network models yield some data independence, but they still exhibit database structure dependency. For example, changes in segments or their location require data managers and programmers to perform complex system management tasks and extensive maintenance coding. Finally, the relational model yields both data and structural independence.

Answers to Problems

Given the following file structure, answer problems 1 through 6.

PROJECT	MANAGER	M_PHONE	M_ADDRESS	BID_PRICE
21-5Z	Holly B. Parker	904-338-3416	3334 Lee Rd., Gainesville, FL 37123	$16,833,460.00
25-2D	Jane D. Grant	615-898-9909	218 Clark Blvd., Nashville, TN 36362	$12,500,000.00
25-5A	George F. Dorts	615-227-1245	124 River Dr., Franklin, TN 29185	$32,512,420.00
25-9T	Holly B. Parker	904-338-3416	3334 Lee Rd., Gainesville, FL 37123	$21,563,234.00
27-4Q	George F. Dorts	615-227-1245	124 River Dr., Franklin, TN 29185	$10,314,545.00
29-2D	Holly B. Parker	904-338-3416	3334 Lee Rd., Gainesville, FL 37123	$25,559,999.00
31-7P	William K. Moor	904-445-2719	216 Morton Rd., Stetson, FL 30155	$56,850,000.00

1. How many records does the file contain, and how many fields are there per record?

The file contains seven records (21-5Z through 31-7P) and each of the records is composed of 5 fields (PROJECT through BID_PRICE.)

2. What problem would you encounter if you wanted to produce a listing by city? How would you solve this problem by altering the file structure?

The city names are contained within the M_ADDRESS attribute and decomposing this character (string) field is cumbersome at best. If the ability to produce city listings is important, it is best to store the city name as a separate attribute.

3. If you wanted to produce a listing of the file contents by last name, area code, city, state, or ZIPCODE, how would you alter the file structure?

The more we divide the address into its component parts, the greater its information capabilities. For example, by dividing M_ADDRESS into its component parts (M_STREET, M_CITY, M_STATE, and M_ZIPCODE), we gain the ability to easily select records on the basis of zip codes, city names, and states. Similarly, by subdividing the MANAGER name into its components M_LASTNAME, M_FIRSTNAME, and M_INITIAL, we gain the ability to produce more efficient searches and listings. For example, creating a phone directory is easy when you can sort by last name, first name, and initial. Finally, separating the area code and the phone number will yield the ability to efficiently group data by area codes. Thus M_PHONE might be decomposed into M_AREACODE and M_PHONE. The more you decompose the data into their component parts, the greater the search flexibility. Data that are decomposed into their most basic components are said to be ***atomic***.

4. What data redundancies do you detect and how could these redundancies lead to anomalies?

Note that the manager named Holly B. Parker occurs three times, indicating that she manages three projects coded 21-5Z, 25-9T, and 29-2D, respectively. (The occurrences indicate that there is a 1:M relationship between PROJECT and MANAGER: each project is managed by only one manager but, apparently, a manager may manage more than one project.) Ms. Parker's phone number and address also occur three times. If Ms. Parker moves and/or changes her phone number, these changes must be made more than once *and they must all be made correctly... without missing a single occurrence*. If any occurrence is missed during the change, the data are "different" for the same person. After some time, it may become difficult to determine what the correct data are. In addition, multiple occurrences invite misspellings and digit transpositions, thus producing the same anomalies. The same problems exist for the multiple occurrences of George F. Dorts.

5. Using two relational database tables, PROJECT and MANAGER, eliminate the redundancies discovered in problem 4. Make sure you use the naming conventions discussed in Section 1.3.3 and connect the two tables through the appropriate link. (*Hint:* Use Database Table 1.3 as an example.)

Start by splitting the manager attributes from the original table to produce the MANAGER table, then add an attribute named MGR_CODE that will uniquely identify each manager record. Use this same code in the PROJECT table to link the two tables. Note that the two tables meet the standards that were described in this chapter and that the solutions to problems 2 and 3 have also been incorporated in the MANAGER table. Note also that the redundancies discussed in the answer to problem 4 have been eliminated: each manager's attributes occur only once in the MANAGER table, thus eliminating the potential for anomalies. The only remaining redundancy is the manager's code ... which occurs more than once in the PROJECT table.