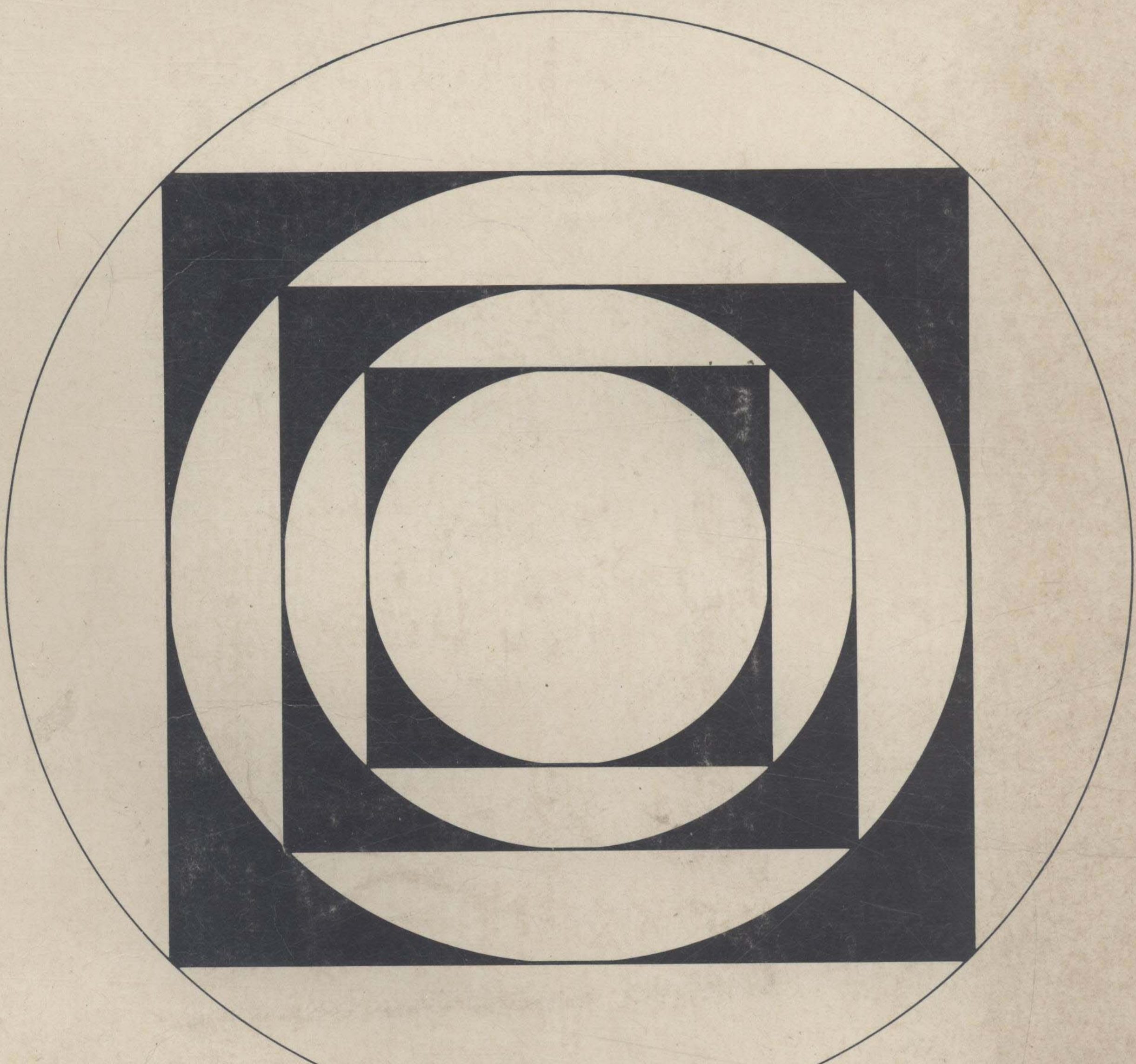
INSTRUCTOR'S GUIDE

DATABASE ANALYSIS AND DESIGN

I. T. HAWRYSZKIEWYCZ



SRA

INSTRUCTOR'S MANUAL

to accompany

DATABASE DESIGN

I.T. Hawryszkiewycz

Canberra College of Advanced Education

SRA®

SCIENCE RESEARCH ASSOCIATES, INC.
Chicago, Henley-on-Thames, Sydney, Toronto

A Subsidiary of IBM

Printed in the United States of America

ISBN 0-574-21486-0

10 9 8 7 6 5 4 3 2 1

PREFACE

The first section of this manual serves as a guide to how the text can be used in courses on computer science and information systems. For each chapter, each subsequent section then describes the objectives, provides a lecture outline, and presents, where appropriate, detailed solutions to the problems. The final section offers solutions to the design projects.

I.T. Hawryszkiewycz

CONTENTS

GENERAL USE OF THE TEXT

The material in the book covers the three major issues in database design:

(1) database theory

(2) the software used to implement databases

(3) the design process itself.

The text can be adapted to courses oriented either to computer science or to information systems. Possible use of the book in these courses is shown in Figures 1 and 2. The major difference is that the information systems course excludes relational theory (especially that in Chapter 3) and emphasizes design. The computer science course makes greater use both of this theory and of the material on software structures in Chapter 9.

Figures 1 and 2 are guidelines only, and variations are possible. For example:

- You can replace some of the material on semantic models in Figure 1 by putting more emphasis on file structures (Chapter 9). This change depends on the students' knowledge prior to the course. If students know little about file structures, the material on semantic modeling can be replaced by that on file structures, which allows models to come in later in the design overview.

- You can rearrange the presentation by first introducing the relational model and then going on with relational software and network and hierarchical systems. Then do relational design, followed by semantic models. (The chapter sequence is now 2, 9, 10, 11, 12, 13, 3, 4, 5, 8, 14, 15, 16).

Similar variations are possible for the information systems course. A further variation here is to present the semantic model before relational theory.

The two-semester course shown in Figure 3 suggested for those institutions that place considerable emphasis on database in their teaching. In this case the first semester concentrates on database theory and database design principles. The second semester covers database management software and the application of the design principles to designing databases that use this software.

Weeks 1-4

Chapters 1, 2, and 3--introduction to databases and discussion of relational theory.

Weeks 5-6

Chapters 4 and 5--discussion of the semantics of data and of the need for semantic models. Illustration of semantic modeling, using Chapter 5, and a discussion of the relationship between the E-R model and relational theory (Chapters 6 and 7 can serve as extra material here or as a graduate extension).

Weeks 7-8

Chapters 9 and 10--introduction to software used to support databases.

Weeks 9-12

Chapters 11, 12 and 13--data models and their implementation by database management systems.

Weeks 13-14

Chapters 8, 14, 15 and 16--an overview of the database design process.

Week 15

Summary.

Figure 1

A ONE-SEMESTER COMPUTER SCIENCE COURSE

Weeks 1-3

Chapters 1 and 2--introduction to databases and discussion of the relational model.

Weeks 4-5

Chapters 4 and 5--discussion of the semantics of data and of the need for semantic models. Illustration of semantic modeling and its relationship to relational theory.

Week 6

Chapter 8--development of a database specification.

Weeks 7-8

Chapters 9 and 10--introduction to software used to support database. Emphasis on Chapter 10 and data models.

Weeks 9-12

Chapters 11, 12 and 13--data models and their implementation by database management systems.

Weeks 13-14

Chapter 14, 15 and 16--database design.

Week 15

Summary.

Figure 2

A ONE-SEMESTER INFORMATION SYSTEMS COURSE

SEMESTER 1: DATA BASE THEORY AND PRINCIPLES

Weeks	Chapters	Goal
1-2	1-2	Introduction and relation theory
2-4	3	Relational design
6-7	4-6	Semantic modeling
8	8	Database specification and conversion to logical record structures
9-10	9	File structures
11-12	14, 16	Design principles and their application to file structure
13-14		An institution-dependent part describing an available DBMS and conversion of E-R model to the DBMS
15		Summary

SEMESTER 2: DATABASE SOFTWARE

Weeks	Chapters	Goal
1	10	Revison and introduction to DBMS
2-3	11	Relational DBMS
4-6	12	Network DBMS
7	14 (partial)	Network design
8-10	13	Hierarchical DBMS
11	14 (partial)	Hierarchical design
12	16	Performance analysis factors
13-14	10, 17 (partial)	Issues of integrity, security, and management
15		Summary

Figure 3

A TWO-SEMESTER COURSE

CHAPTER 1

INTRODUCTION

Objectives: (1) Outline the course philosophy.

(2) Describe the need for staged design.

(3) Outline the design stages.

(4) Introduce the criteria used to make design decisions.

LECTURE OUTLINE

(1) It is important that this chapter makes the student aware that database design proceeds in stages. Each stage is directed to one well-defined design problem. Explain the reasons for staged design. A good approach here is to outline criteria to judge good designs and to state that it is not possible to make decisions that consider all criteria at the same time. Design can be broadly subdivided into two kinds of stages:

(a) Data analysis stages make decisions about how the data are to be structured in the database.

(b) Technical design stages make decisions about the representation of the data on the computer. These decisions are again divided into two parts:

- logical design to arrange the data into files
- physical design to select the method of storing and accessing the files.

Stress the reason for separating data analysis and technical design. The direct approach here is to say that analysis first defines the ideal. This ideal is then implemented within the constraints of an available technical system.

(2) The important functions at the different kinds of stages should then be pointed out. The goal of data analysis in this context is to identify user data and represent them by a model that is easily understood by users and computer personnel. To do this, data analysis must provide

(a) a model for representing data

(b) rules and criteria for creating good representation of user data.

In technical design the major objective is to select the representation of user data on the computer system. Such representation must ensure that

(a) all user access requirements can be met

(b) the representation of user data makes minimal use of computer resources.

(3) Some of these objectives may then be elaborated in informal terms. Thus students should be made informally aware of the following important criteria for structuring data:

(a) Each fact is stored once; hence there is no redundancy in the system.

(b) Consistency is maintained after database operations.

(c) Change is supported through symmetry; that is, we can easily change the access key if all attributes are at the same level.

(d) The logical structure presented to the user is independent of its physical representation.

The good effects of these characteristics should be explained. For example, no redundancy means less storage; consistency means that conflicting information is not given to different users; change is necessary to support the continuous evolution of information systems; and independence from physical structure means that it is easier to make logical changes.

(4) The important point at the conclusion is to lead to the relational model and describe its role in the database field. It serves both as a theory to provide criteria for structuring data and as a data model that can be directly implemented on machines. These reasons should be used to initially motivate the study of relational theory. Point out that what we are doing is first explaining the fundamentals of database theory in the early chapters and then showing the application of this theory in later chapters.

(5) It may also be worthwhile to relate the activities in data analysis and technical design to the top-down elaboration inherent in problem solving. In this way students can be led to the notion of semantic models. It is possible here to briefly note that semantic models are used at the top level of design to identify essential features of any enterprise. Relational theory is then used to refine semantic model structures so that it satisfies all the good criteria.

CHAPTER 2

RELATIONAL MODEL

Objectives: (1) Describe the relational model.

(2) Define the elementary normal forms.

(3) Outline functional dependency theory.

(4) Introduce relational languages.

LECTURE OUTLINE

(1) Although many of the issues here appear conceptually simple (once you know them), they may take a while to sink in for the novice. So the best advice here is to proceed very slowly in the early stages. Make sure that the students understand the basics before proceeding to the more complex issues of relational theory. Particularly important here are the following ideas:

- Functional dependencies are a property of the information and not of the way that attributes are arranged into relations.

- Normal forms are necessary to preserve consistency after operations.

(2) It is most important that students become familiar with the relational structure before commencing the study of normal form. Some time spent on Problems 1 to 5 can prove very valuable.

(3) Perhaps the most difficult problem for students trying to find the highest normal form is to find the relation key. Some time spent on finding keys can be profitable. One way is to start with a key made of all the determinants of FDs in the relation. Then eliminate attributes that are determined by other attributes. Alternate elimination sequences lead to alternate relation keys.

(4) Properties of FDs are introduced now to further consolidate relational theory. The material is relatively straightforward. Redundancy can serve as the objective here. We wish to derive the minimal set of FDs so as not to introduce redundancy in the design process. Outline the inference rules to illustrate the concept of redundant FDs, and then describe the membership algorithm.

(5) Relational languages are used to complete the picture on the relational model. The aim at this stage should not be to give a detailed explanation of one relational language. On the other hand, you should describe the versatility of the relational model, showing the "kinds" of languages it supports. Specific implementations will come later. The natural interface and selective power should be used as the criteria to judge languages.

SOLUTIONS

Problem 1

This problem is intended primarily to familiarize students with the basic structures of relations and the concept of functional dependency. It is important to stress that the contents of the relation must be consistent with the FDs between the relation attributes and not vice versa. This issue is covered further in Problem 3. The students should examine each instance and determine whether that instance is consistent with the FDs. The solution is as follows.

- CASE 1

INSTANCE 1--valid

INSTANCE 2--not valid as NAME → ADDRESS, ADDRESS → NO-ROOMS } does not hold

INSTANCE 3--not valid as NAME → ADDRESS, ADDRESS → NO-ROOMS } does not hold

- CASE 2

  INSTANCE 1--valid

  INSTANCE 2--valid

  INSTANCE 3--not valid as ADDRESS → NO-ROOMS does not hold

- CASE 3

  INSTANCE 1--valid

  INSTANCE 2--not valid as NAME, DATE-MOVED-IN → ADDRESS does not hold

  INSTANCE 3--valid

Problem 2

This problem continues the one discussed in Problem 1. It further reinforces the requirements that the relation contents must be consistent with the FDs between attributes for all time. Thus the contents must remain consistent after each database operation. For the operations in this problem the following holds:

add (JOE, ADD4, 05081, 17)
add (JIM, ADD5, 04081, 30) } *R*1 is consistent with the FDs in CASE 2 after these operations

add (JILL, ADD1, 070781, 9) not valid as ADDRESS → NO-ROOMS will not hold

Problem 3

This problem further reinforces the principle that the FDs are derived from the nature of the enterprise and not from the current contents of a relation.

Thus it would appear from relation WORK that

TYPE-OF-ITEM-MADE → DATE (each item is made on one day only)

TYPE-OF-ITEM-MADE → OPERATOR (each item is made by a different operator)

This situation need not hold for all time in this enterprise. The conditions in the last part of the problem lead to the FDs

(a) OPERATOR, DATE → MACHINE

(b) OPERATOR, DATE → MACHINE
MACHINE, DATE → OPERATOR

(c) TYPE-OF-ITEM-MADE → MACHINE

Problem 4

This problem is designed to illustrate the anomalies that can arise in nonnormal relations following the execution of add, delete, or update operations. The following anomalies arise after the given operations:

(a) PROJECT-NO → PLANNED-END-DATE no longer holds.

(b) The end date is changed in only one of three tuples and hence PROJECT-NO → PLANNED-END-DATE no longer holds.

(c) Same as (b) because price is changed in only one tuple and hence ITEM-NO → COST no longer holds.

(d) The cost of X6 is also removed from the database.

(e) Adds dummy values for PROJECT-NO, QTY-USED, START-DATE, PLANNED-END-DATE.

PROJECT-USE can be normalized by decomposing it into the following three relations:

*R*1 (ITEM-NO, COST)

*R*2 (PROJECT-NO, ITEM-NO, QTY-USED)

*R*3 (PROJECT-NO, START-DATE, PLANNED-END-DATE)

The operations will not cause anomalies, given these relations.

Problem 5

Problem 5 is designed to give students experience in developing FDs, given an enterprise description. The FDs here are as follows:

(a) PERSON → DEPT

(b) ITEM-NAME, STORE → PRICE-OF-ITEM

(c) PERSON, POSITION → START-TIME, END-TIME
PERSON, START-TIME ↔ POSITION, END-TIME
PERSON, END-TIME ↔ POSITION, START-TIME

(d) PROJECT, TASK → TASK-START, TASK-COST

Problem 6

This is the first instance in which students are required to first find the relation keys and then determine the normal form of the relation. The derivation of the relation key, given the FDs, seems to cause the biggest problem. One way to approach this is to commence with a relation key, which consists of the union of all the determinants of FDs between the relational attributes. Once this is done, any dependencies between attributes in this initial set are eliminated and what remains becomes the relation key.

The solution to the four parts of this problem is as follows:

(a) *AB* is the key and *D* is dependent on part of the key.

(b) *AB* is one key and *E* is another key.

(c) *AB* is one key and *AD* is another key; and *D*, although a prime attribute, is partially dependent on a key of which it is not a part.

(d) *R*1 keys are *A* or *B*, relation is in BCNF
*R*2 key is *C*, relation is 2NF as dependency between nonkey attributes
*R*3 key is *AD*, relation is in 1NF as dependency on part of key
*R*4 key is *AD* and relation is in 1NF

Problem 7

This problem is similar to Problem 6 except that now all the FDs are given globally. The solution to the given relations is as follows:

(a) The relation key is {SALE-NO, ITEM-IN-SALE} and the relation is in BCNF.

(b) The relation key is {SALE-NO, ITEM-IN-SALE}, but the relation is now only in 1NF because ITEM-PRICE depends on part of this key, that is, ITEM-IN-SALE.

(c) The relation key is {SALE-NO, ITEM-IN-SALE}, but the relation is in 1NF because REGION depends (transitively) on part of this key, that is, ITEM-IN-SALE.

(d) This relation, although in 1NF, does not contain useful information.

(e) This relation is all key and hence in BCNF, but again it does not contain useful information.

(f) The relation key is SALE-NO and the relation is in 2NF because there is an FD between the nonkey items.

Problem 8

Here we continue with the problem presented in Problems 6 and 7. The solution here is as follows:

	Keys	Nonprime Attributes	Normal Form
*R*0	*XY*	*WZ*	1 because *W* is dependent on part of key
*R*1	*A*	*BCD*	2 because of dependency between nonprime attributes
*R*2	*KL*	*MN*	1 because *M* is dependent on part of key
*R*3	*P*	*QRS*	2 because of dependency between nonprime attributes
*R*4	*T*, *U*	*V*	BCNF
*R*5	*WY*, *YZ* (*note:* $YZ \rightarrow W$ is redundant)	*X*	3 because part of one key (*W*) is partially dependent on another key (*YZ*)
*R*6	*ABC*	*D*	1 because *D* depends on part of the key
*R*7	*KM*, *KL* (*note:* $KL \rightarrow N$ is redundant)	*N*	3 because *M* is dependent on part of key *KL*

Problem 9

This is another problem on normal form and finding relation keys. The answer:

*R*1 relation key is (PERSON-NO)

note PERSON-NO → MANAGER as
PERSON-NO → DEPT-OF-PERSON → MANAGER

Relation is in 2NF because there is a dependency between nonkey attributes.

*R*2 relation key is (PERSON-NO, PROJECT-NO, WEEK-NO); relation is in 1NF because NAME is functionally dependent on part of the key, that is, PERSON-NO

$R3$ relation key is (PROJ-NO, ITEM-NO, PERSON-NO, WEEK-NO); relation is in 1NF because there is dependency on part of the key

$R4$ relation key is (PROJ-NO, ITEM-NO, PERSON-NO, WEEK-NO); that is, the relation is all key and therefore is in BCNF.

Problem 10

This is the final problem on normal forms. The solution:

$R1$ is not BCNF but it is in 1NF because (PROJ-NO, MACHINE-NO) is key and PROJ-NAME is dependent on part of the key

$R2$ is not BCNF because TIME-USE-ON-PROJECT is dependent on part of (PROJ-NO, MACHINE-NO, PERSON-NO)

$R3$ is all key and hence BCNF

Problem 11

This is the first problem on properties on FDs. The goal here is to determine whether the FDs in column B can be derived from those in column A. The solution:

(a) Yes, because $X \to Z$ follows by transitivity: $X \to Y \to Z$ implies $X \to Z$.

(b) Yes, in column A, $Z \to A$ and $XY \to A$.
Hence by pseudotransitivity, $ZY \to A$.

(c) $ZY \to A$ cannot be derived from $XY \to A$ and $X \to Z$.

(d) Yes, in column A, $XY \to Z$ and $X \to Y$ because from $X \to Y$ and $X \to X$, $X \to XY$; and now, by transitivity, $X \to XY \to Z$.

(e) Some but not all FDs in column B can be derived from those in column A.

Thus $X \to B$ because $XY \to B$ and $X \to Y$ results in $X \to B$ in the same way as in (d). However, it is not true that $ZY \to A$.

(f) $ZY \to V$ cannot be derived from the FDs in column A. (As a further example, $ZX \to V$ can be so derived.)

Problem 12

The membership algorithm can be applied to the sets of FDs to detect redundant dependencies.

. SET 1

We apply the membership algorithm to $A \to B$:

Step 1: Start with $T = A$.

Step 2: There is no FD with its determinant in T.

Step 4: B is not in T, so $A \to B$ is not redundant.

We apply the membership algorithm to $AD \to C$:

Step 1: Start with $T = AD$.

Step 2: The only FD with its determinant in T is $A \to B$.

Step 3: This step is now repeated because T has been changed. The only remaining

FD with its determinant in T is $B \rightarrow C$. Hence C is added to T, which now becomes $ABCD$.

Step 4: C is now in T, so $AD \rightarrow C$ is redundant.

- SET 2

$ZW \rightarrow V$ is redundant because of the following steps:

Step 1: $T = ZW$.

Step 2: For $W \rightarrow Y$, add Y to T
for $Z \rightarrow X$, add X to T.
Now $T = ZWXY$
add V to T because $XY \rightarrow V$.

Step 4: Now V is in T and hence $ZW \rightarrow V$ is redundant.

- SET 3

$PS \rightarrow Q$ is redundant because of the following steps:

Step 1: Start with $J = PS$ (note that J is now used as the variable).

Step 2: Add RST to J because $P \rightarrow RST$ and P is in J
add T to J because $PS \rightarrow T$ and PS is in J.
Now $J = RPST$
add V to J because $SR \rightarrow V$ and SR is in J.
Now $J = VRPST$
add Q to J because $VRT \rightarrow SQP$ and VRT is in J.

Step 4: $PS \rightarrow Q$ is redundant because Q is in J.

Problem 13

This is another problem in detecting redundant FDs. The solutions:

SET 1 either $AC \rightarrow Z$ or $AB \rightarrow Z$ is redundant but not both.
Hence two minimal coverings exist.

SET 2 $PS \rightarrow T$ and $QS \rightarrow P$ are both redundant.

SET 3 No redundancy.

Problem 14

This is a straightforward application of relational algebra. The solutions:

(a) $R1$ = SELECT STUDENT WHERE STUDENT-ID = x
$R2$ = PROJECT $R1$ OVER UNIT

(b) $R1$ = SELECT UNIT-ENROLLMENT WHERE STUDENT-ID = x
$R2$ = JOIN $R1$, UNIT-LECTURER ON (UNIT, SEMESTER)
$R3$ = PROJECT $R2$ OVER LECTURER-IN-CHARGE

(c) $R1$ = JOIN UNIT-ENROLLMENT, STUDENT OVER STUDENT-ID
$R2$ = SELECT $R1$ WHERE UNIT = x AND SEMESTER = y
$R3$ = PROJECT $R2$ OVER COURSE

(d) $R1$ = SELECT UNIT-ENROLLMENT WHERE STUDENT-ID = x
$R2$ = JOIN $R1$, UNIT-OFFER OVER UNIT
$R3$ = PROJECT $R2$ OVER SCHOOL