Chapter 2: Database Design Fundamentals

Solutions

Answers to Review Questions

1. An entity is a person, place, thing, or event.

2. An attribute is a property of an entity.

3. A relationship is an association between tables (entities). A one-to-many relationship between two tables is a relationship in which each row in the first table can be associated with many rows in the second table, but each row in the second table is associated with only one row in the first table.

4. A repeating group is multiple entries in a single location in a table.

5. A relation is a two-dimensional table in which the entries in the table are single-valued (each location in the table contains a single entry), each column has a distinct name (or attribute name), all values in a column are values of the same attribute, the order of the rows and columns is immaterial, and each row contains unique values.

6. A relational database is a collection of relations.

7. For each table, you write the name of the table and then within parentheses list all of the columns in the table. Underline the primary keys.

```
CUSTOMER (CUSTOMER_NUM, LAST_NAME, FIRST_NAME, ADDRESS, CITY,
          STATE, POSTAL_CODE, PHONE)
TRIP (TRIP_ID, TRIP_NAME, START_LOCATION, STATE, DISTANCE,
      MAX_GRP_SIZE, TYPE, SEASON)
GUIDE (GUIDE_NUM, LAST_NAME, FIRST_NAME, ADDRESS, CITY,
        STATE, POSTAL_CODE, PHONE_NUM, HIRE_DATE)
RESERVATION (RESERVATION_ID, TRIP_ID, TRIP_DATE, NUM_PERSONS,
             TRIP_PRICE, OTHER_FEES, CUSTOMER_NUM)
TRIP_GUIDES (TRIP_ID, GUIDE_NUM)
```

8. To qualify the name of a field, indicate the table in which the field appears. You do this by preceding the name of the field with the name of the table and a period.

9. A column (attribute), B, is functionally dependent on another column, A (or possibly a collection of columns), if at any point in time a value for A determines a single value for B.

10. Column A (or a collection of columns) is the primary key for a table if (1) *All* columns in the table are functionally dependent on A and (2) No subcollection of the columns in A (assuming A is a collection of columns and not just a single column) also has property 1. The primary key of the CUSTOMER table is the CUSTOMER_NUM column. The primary key of the TRIP table is the TRIP_ID column. The primary key of the GUIDE table is the GUIDE_NUM column. The primary key of the

RESERVATION table is the RESERVATION_ID column. The primary key of the TRIP_GUIDES table is the combination of the TRIP_ID and GUIDE_NUM columns.

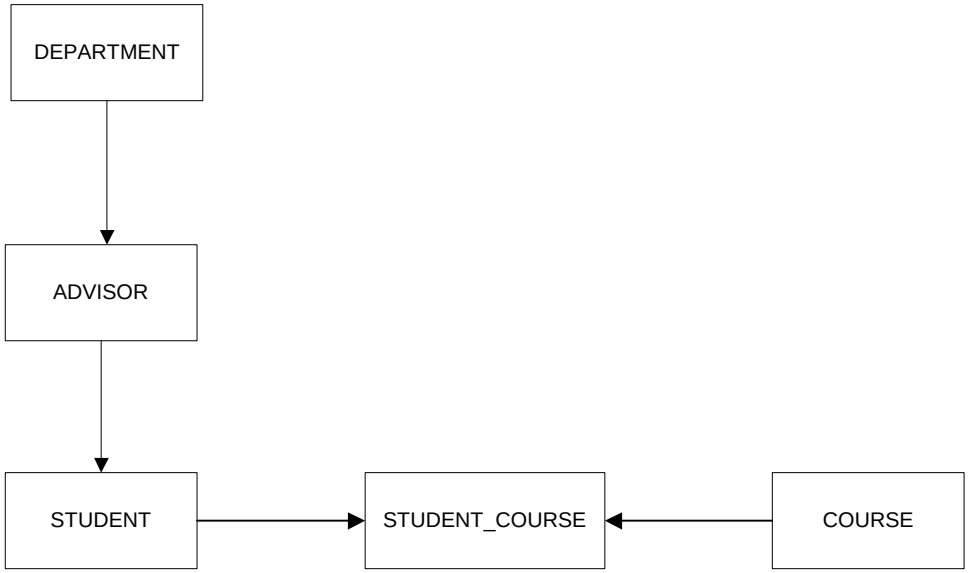
11. Functional dependencies:

```
DEPARTMENT_NUM → DEPARTMENT_NAME
ADVISOR_NUM → ADVISOR_LAST_NAME, ADVISOR_FIRST_NAME, DEPARTMENT_NUM
COURSE_CODE → DESCRIPTION
STUDENT_NUM → STUDENT_LAST_NAME, STUDENT_FIRST_NAME, ADVISOR_NUM
STUDENT_NUM, COURSE_CODE → GRADE
```

Relations:

```
DEPARTMENT (DEPARTMENT_NUM, DEPARTMENT_NAME)
ADVISOR (ADVISOR_NUM, ADVISOR_LAST_NAME, ADVISOR_FIRST_NAME,
      DEPARTMENT_NUM)
COURSE (COURSE_CODE, DESCRIPTION)
STUDENT (STUDENT_NUM, STUDENT_LAST_NAME, STUDENT_FIRST_NAME,
      ADVISOR_NUM
STUDENT_COURSE (STUDENT_NUM, COURSE_CODE, GRADE)
```

Entity-Relationship diagram: (**NOTE**: Your rectangles may be in different positions as long as they are connected by the same arrows.)



12. A table (relation) is in first normal form (1NF) if it does not contain repeating groups.

13. A table (relation) is in second normal form if it is in first normal form and no nonkey column is dependent on only a portion of the primary key. If a table is not in second normal form, the table contains redundancy, which leads to a variety of update anomalies. A change in a value can require not just one change, but several. There is the possibility of inconsistent data. Adding additional data to the database may not be possible without creating artificial values for part of the key. Finally, deletions of certain items can result in inadvertently deleting crucial information from the database.

14. A table is in third normal form if it is in second normal form and if the only determinants it contains are candidate keys. A change in a value can require not just one change, but several. There is the possibility of inconsistent data. Adding certain additional data to the database may not be possible without creating artificial rows in the table. Finally, deletions of certain items can result in inadvertently deleting crucial information from the database.

15.

```
STUDENT (STUDENT_NUM, STUDENT_LAST_NAME, STUDENT_FIRST_NAME,
      ADVISOR_NUM)
ADVISOR (ADVISOR_NUM, ADVISOR_LAST_NAME, ADVISOR_FIRST_NAME)
COURSE (COURSE_CODE, DESCRIPTION)
STUDENT_COURSE  (STUDENT_NUM, COURSE_CODE, GRADE)
```

16. [Critical Thinking] If a student can have more than one advisor, there is a many-to-many relatioship between students and advisors. Remove ADVISOR_NUM from the STUDENT relation and add a relation STUDENT_ADVISOR)

```
STUDENT (STUDENT_NUM, STUDENT_LAST_NAME, STUDENT_FIRST_NAME)
ADVISOR (ADVISOR_NUM, ADVISOR_LAST_NAME, ADVISOR_FIRST_NAME)
STUDENT_ADVISOR (ADVISOR_NUM, STUDENT_NUM)
COURSE (COURSE_CODE, DESCRIPTION)
STUDENT_COURSE (STUDENT_NUM, COURSE_CODE, GRADE)
```

17. [Critical Thinking] If students can repeat a course, then the STUDENT_NUM, COURSE_CODE, YEAR, and SEMESTER determine the grade.

```
STUDENT (STUDENT_NUM, STUDENT_LAST_NAME, STUDENT_FIRST_NAME,
      ADVISOR_NUM)
ADVISOR (ADVISOR_NUM, ADVISOR_LAST_NAME, ADVISOR_FIRST_NAME)
COURSE (COURSE_CODE, DESCRIPTION)
STUDENT_COURSE (STUDENT_NUM, COURSE_CODE, YEAR, SEMESTER, GRADE)
```

Answers to TAL Distributors Exercises

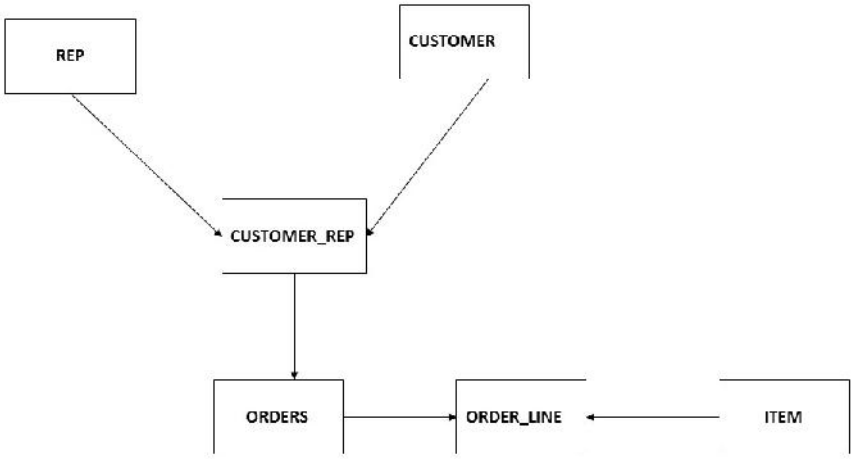
1. **NOTES**: The CUSTOMER_REP table in the following lists implements the relationship between customers and reps. If customer 126, for example, is represented by both rep 15 and rep 30, there would be a row in the table in which the customer number is 126 and the rep number is 15 as well as a row in which the customer number is 126 and the rep number is 30. A row would only be allowed in the ORDERS table if the combination of the customer number and the rep number match a row in the CUSTOMER_REP table.

```
REP (REP_NUM, LAST_NAME, FIRST_NAME, STREET,
      CITY, STATE, POSTAL_CODE, COMMISSION, RATE)
CUSTOMER (CUSTOMER_NUM, CUSTOMER_NAME, STREET,
      CITY, STATE, POSTAL_CODE, BALANCE, CREDIT_LIMIT)
CUSTOMER_REP (CUSTOMER_NUM, REP_NUM)
ORDERS (ORDER_NUM, ORDER_DATE, CUSTOMER_NUM, REP_NUM)
ORDER_LINE (ORDER_NUM, ITEM_NUM, NUM_ORDERED,
      QUOTED_PRICE)
```

```
ITEM (ITEM_NUM, DESCRIPTION, ON_HAND, CATEGORY,
      STOREHOUSE, PRICE)
```

Relationships: There are one-to-many relationships from REP to CUSTOMER_REP, CUSTOMER to CUSTOMER_REP, CUSTOMER_REP to ORDERS, ORDERS to ORDER_LINE, and ITEM to ORDER_LINE.

Entity-Relationship diagram: (**NOTE**: Your rectangles may be in different positions as long as they are connected by the same arrows.)

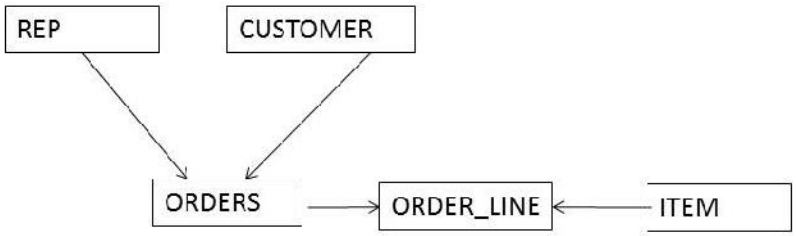


2. **NOTES**: There is no relationship between customers and reps, so there is no REP_NUM column in the CUSTOMER table nor is there an additional table like the CUSTOMER_REP table in Exercise 1. A row can only exist in the ORDERS table if the customer number matches a row in the CUSTOMER table and the rep number matches a row in the REP table.

```
REP (REP_NUM, LAST_NAME, FIRST_NAME, STREET, CITY, STATE, POSTAL_CODE,
      COMMISSION, RATE)
CUSTOMER (CUSTOMER_NUM, CUSTOMER_NAME, STREET, CITY, STATE,
      POSTAL_CODE, BALANCE, CREDIT_LIMIT)
ORDERS (ORDER_NUM, ORDER_DATE, CUSTOMER_NUM, REP_NUM)
ORDER_LINE (ORDER_NUM, ITEM_NUM, NUM_ORDERED, QUOTED_PRICE)
ITEM (ITEM_NUM, DESCRIPTION, ON_HAND, CATEGORY, STOREHOUSE, PRICE)
```

Relationships: There are one-to-many relationships from REP to ORDERS, CUSTOMER to ORDERS, ORDERS to ORDER_LINE, and ITEM to ORDER_LINE.

Entity-Relationship diagram: (**NOTE**: Your rectangles may be in different positions as long as they are connected by the same arrows.)

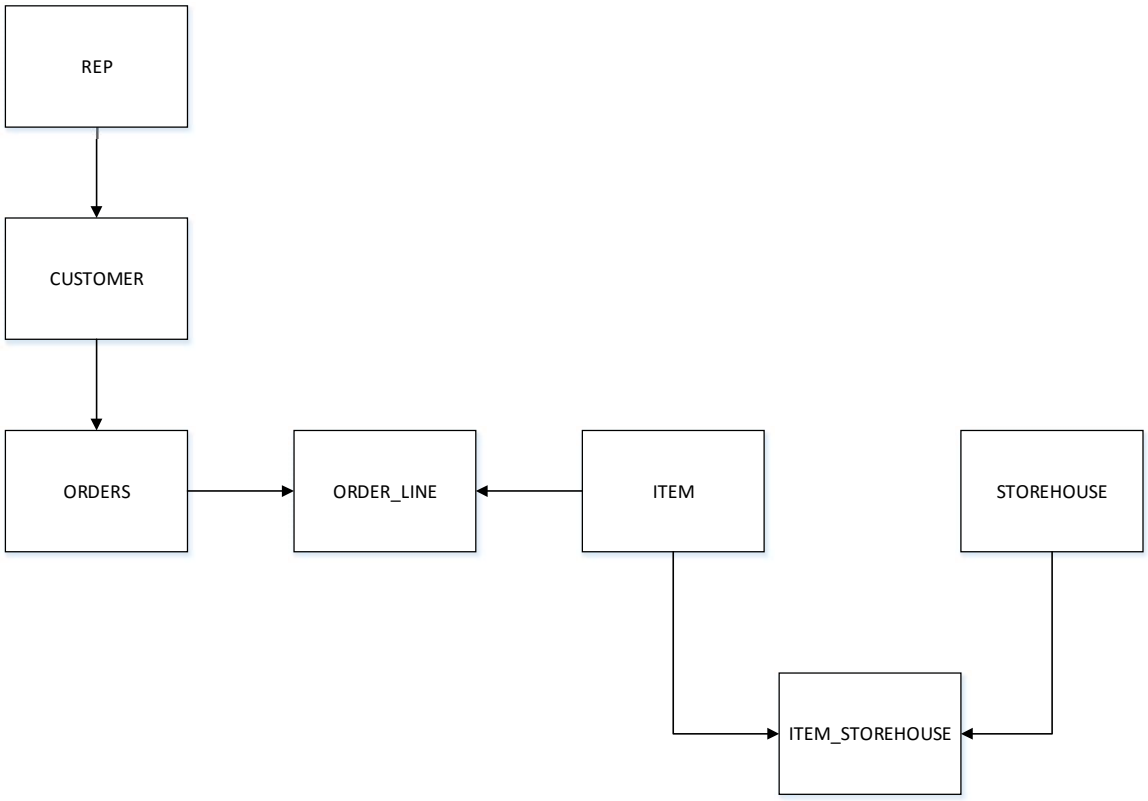


3. **NOTES**: The STOREHOUSE_NUM and ON_HAND columns do not appear in the ITEM table. There is a STOREHOUSE table, whose key is STOREHOUSE_NUM and which contains the STOREHOUSE description. Information about units on hand is stored in a new table, the ITEM_STOREHOUSE table, whose key is the combination of the ITEM number and STOREHOUSE number. If there are 10 units of ITEM BR23 on hand in STOREHOUSE 2, for example, there would be a row in ITEM_STOREHOUSE on which the ITEM number is BR23, the STOREHOUSE number is 2, and the number of units on hand is 10.

```
REP (REP_NUM, LAST_NAME, FIRST_NAME, STREET, CITY, STATE, POSTAL_CODE,
      COMMISSION, RATE)
CUSTOMER (CUSTOMER_NUM, CUSTOMER_NAME, STREET, CITY, STATE,
      POSTAL_CODE, BALANCE, CREDIT_LIMIT, REP_NUM)
ORDERS (ORDER_NUM, ORDER_DATE, CUSTOMER_NUM)
ORDER_LINE (ORDER_NUM, ITEM_NUM, NUM_ORDERED, QUOTED_PRICE)
ITEM (ITEM_NUM, DESCRIPTION, CATEGORY, PRICE)
STOREHOUSE (STOREHOUSE_NUM, STOREHOUSE_DESCRIPTION)
ITEM_STOREHOUSE (ITEM_NUM, STOREHOUSE_NUM, ON_HAND)
```

Relationships: There are one-to-many relationships from REP to CUSTOMER, CUSTOMER to ORDERS, ORDERS to ORDER_LINE, ITEM to ORDER_LINE, ITEM to ITEM_STOREHOUSE, and STOREHOUSE to ITEM_STOREHOUSE.

Entity-Relationship diagram: (**NOTE**: Your rectangles may be in different positions as long as they are connected by the same arrows.)



4. Functional Dependencies:

```
ITEM_NUM → DESCRIPTION, ON_HAND, CATEGORY, STOREHOUSE, PRICE
ORDER_NUM → ORDER_DATE, CUSTOMER_NUM
CUSTOMER_NUM → CUSTOMER_NAME
ITEM_NUM, ORDER_NUM → NUM_ORDERED, QUOTED_PRICE
```

Relations:

```
ITEM (ITEM_NUM, DESCRIPTION, ON_HAND, CATEGORY, STOREHOUSE, PRICE)
ORDERS (ORDER_NUM, ORDER_DATE, CUSTOMER_NUM)
CUSTOMER (CUSTOMER_NUM, CUSTOMER_NAME)
ORDER_LINE (ITEM_NUM, ORDER_NUM, NUM_ORDERED, QUOTED_PRICE)
```

NOTE: The keys for ORDER_LINE could also have been listed as ORDER_NUM, ITEM_NUM.

5. [Critical Thinking] One way to address this change is to add two tables to the database: STOREHOUSE and MANAGER.

```
STOREHOUSE (STOREHOUSE, MANAGER_NUM)
MANAGER (MANAGER_NUM, LAST_NAME, FIRST_NAME)
```

Answers to Colonial Adventure Tours Exercises

1. Many answers are possible. Here is one possible solution:

1NF but not 2NF:

```
TripGuides (TripID, GuideNum, TripName,)
```

Conversion to 2NF:

```
Trip (TripID, TripName)
TripGuides (TripID, GuideNum)
```

2NF but not 3NF:

```
Reservation (ReservationID, TripID, OwnerNum, LastName,
FirstName)
```

Conversion to 3NF:

```
Owner (OwnerNum, LastName, FirstName)
Reservation (ReservationID, TripID, OwnerNum)
```

2. Functional Dependencies:

```
TRIP_ID → TRIP_NAME, STATE_ABBREVIATION, STATE_NAME
GUIDE_NUM → GUIDE_LAST, GUIDE_FIRST
STATE_ABBREVIATION → STATE_NAME
```

Tables (Relations):

```
TRIP (TRIP_ID, TRIP_NAME, STATE_ABBREVIATION)
STATE (STATE_ABBREVIATION, STATE_NAME)
GUIDE (GUIDE_NUM, GUIDE_LAST, GUIDE_FIRST)
TRIP_GUIDE (TRIP_ID, GUIDE_NUM)
```

NOTE: The TRIP_GUIDE relation is necessary to relate trips and guides. (You could have assigned it any name you like.)

3. [Critical Thinking] 3NF:

```
Participant (ParticipantNum, LastName, FirstName, Address, City,
              State, PostalCode, Phone, BirthDate)
Class (ClassNum, Description, MaxPersons, ClassFee)
ClassParticipant (ClassNum, ParticipantNum, ClassDate, ActualNum)
FK    ClassNum → Class
FK    ParticipantNum → Participant
```

Diagram: The student's diagram should have the following boxes (rectangles):
Guide, Trip, Reservation, Customer, TripGuides, Participants, Class
The diagram should have the following connections (arrows):
Guide to TripGuides, Trip to TripGuides, Customer to Reservation,. Participant to ClassParticipant, Class to ClassParticipant

Answers to Solmaris Condominium Group Exercises

1. Functional Dependencies

```
LOCATION_NUM        LOCATION_NAME

LOCATION_NUM, UNIT_NUM   SQR_FT, BDRMS, BATHS, CONDO_FEE
```

3NF

```
LOCATION (LOCATION_NUM, LOCATION_NAME)

CONDO_UNIT (LOCATION_NUM, UNIT_NUM, SQR_FT, BDRMS, BATHS,
            CONDO_FEE)
```

2. Functional Dependencies:

```
CONDO_ID → LOCATION_NUM, UNIT_NUM, SQR_FT, BDRMS, BATHS,
      CONDO_FEE, OWNER_NUM, LAST_NAME, FIRST_NAME
OWNER_NUM → LAST_NAME, FIRST_NAME
```

Tables (Relations):

```
CONDO_UNIT (CONDO_ID, LOCATION_NUM, UNIT_NUM, SQR_FT, BDRMS,
      BATHS, CONDO_FEE, OWNER_NUM)
OWNER (OWNER_NUM, LAST_NAME, FIRST_NAME)
```

3. [Critical Thinking] Functional Dependencies
NOTE: The design assumes that the weekly rate can very with the rental agreement. If students assume that the weekly rate is always the same then the rate would be stored only in the CONDO_UNIT table. The design also assumes that both LOCATION_NUM and CONDO_UNIT_NUM uniquely identify a given condo. This is different than the way Solmaris database is designed for this text. As an alternative you can use the same design for the CONDO_UNIT table as that shown in the text.

```
RENTER_NUM → FIRST_NAME, MID_INITIAL, LAST_NAME, ADDRESS,
             CITY, STATE, POSTAL_CODE, PHONE_NUM, EMAIL

LOCATION_NUM → LOCATION_NAME, ADDRESS, CITY, STATE, POSTAL_CODE

LOCATION_NUM, CONDO_UNIT_NUM → SQR_FT, BEDRMS, BATHS,
                               MAX_PERSONS, WEEKLY_RATE

RENTER_NUM, LOCATION_NUM, CONDO_UNIT_NUM → START_DATE, END_DATE,
                                           RENTAL_RATE
```

3 NF

```
RENTER (RENTER_NUM, FIRST_NAME, MID_INITIAL, LAST_NAME, ADDRESS,
        CITY, STATE, POSTAL_CODE, PHONE_NUM, EMAIL)

LOCATION (LOCATION_NUM, LOCATION_NAME, ADDRESS, CITY, STATE,
          POSTAL_CODE)

CONDO_UNIT (CONDO_UNIT_NUM, LOCATION_NUM, SQR_FT, BEDRMS, BATHS,
            MAX_PERSONS, WEEKLY_RATE)

RENTAL_AGREEMENT (RENTER_NUM, LOCATION_NUM, CONDO_UNIT_NUM,
                  START_DATE, END_DATE, RENTAL_RATE)
```

Diagram: The student's diagram should have the following boxes (rectangles):
Renter, Location, Condo_Unit, Rental_Agreement
The diagram should have the following connections (arrows):
Renter to Rental_Agreement, Location to Condo_Unit, Location to Rental_Agreement, Condo_Unit to Rental Agreement

Chapter 2

Database Design Fundamentals

At a Glance

Instructor's Manual Table of Contents

- Overview
- Objectives
- Teaching Tips
- Quick Quizzes
- Class Discussion Topics
- Additional Projects
- Additional Resources
- Key Terms

Lecture Notes

Overview

In this chapter, students learn about database design. Students examine the important concepts related to databases. They learn how to identify tables and columns and how to identify the relationships between the tables. Students learn how to produce an appropriate database design for a given set of requirements. They examine the process of normalization, a process that identifies and fixes potential problems in a database design. Finally, students learn how to visually represent a database design.

Chapter Objectives

In this chapter, students learn about:

- What the terms *entity*, *attribute*, and *relationship* mean
- What the terms *relation* and *relational database* mean
- What functional dependencies are and how to identify when one column is functionally dependent of another
- What the term *primary key* means and how to identify primary keys in tables
- How to design a database to satisfy a set of requirements
- How to convert an unnormalized relation to first normal form
- How to convert tables from first normal form to second normal form
- How to convert tables from second normal form to third normal form
- How to create an entity-relationship diagram to represent the design of a database

Teaching Tips

Introduction

1. Define **database design.** Database design is the process of determining the particular tables and columns that will comprise a database.

Teaching Tip	This chapter does not need to be covered in sequence. It can be covered later in the course. If you are using a textbook such as Pratt and Last's *Concepts of Database Management, Eighth Edition*, you may want to skip this chapter entirely. Be prepared to spend considerable class time on this chapter. The material is complex, and it is important that students understand all of the concepts presented. The best way for students to learn the material is to work through lots of examples. Use the embedded questions that are included throughout the chapter to test students' understanding. Encourage students to bring their texts with them to class so that they can review the examples.

Database Concepts

1. An understanding of fundamental database concepts is essential to good database design.

Relational Databases

1. Define **relational database**. A relational database is a collection of tables. Formally, tables are called relations.
2. Use Figure 2-1 to emphasize that the TAL Distributors database is a collection of tables.
3. Review the Note on page 23.

Entities, Attributes, and Relationships

1. Define **entity.** An entity is a person, place, object, event, or idea for which you want to store and process data. The entities of interest to TAL Distributors are customers, orders, items, and sales reps.
2. Define **attribute**. An attribute is a characteristic or property of an entity. The terms column and field are used as synonyms in many database systems. For TAL Distributors, the attributes of interest for the entity "customer" are such things as customer name, street, city, and so on.
3. Define **relationship** and **one-to-many relationship**. A relationship is an association between entities. There is a one-to-many relationship between sales reps and customers in the TAL Distributors database. One sales rep represents many customers, but each customer is associated with only one sales rep.
4. In a relational database, each entity has its own tables, and the attributes of the entity are columns in the table. A one-to-many relationship is handled by using common columns in the two tables.
5. Use Figure 2-1 to illustrate the one-to-many relationship between sales reps and customers.

6. Use Figure 2-2 to illustrate **repeating groups** (multiple entries in an individual location in a table).
7. Define **relation.** A relation is a two-dimensional table with specific properties. These properties include:
   - Entries in the table are single-valued.
   - Each column has a distinct name.
   - All values in a column are values of the same attribute.
   - The order of the columns is immaterial.
   - Each row is distinct.
   - The order of the rows is immaterial.
8. Use Figure 2-3 to discuss the six properties of a relation.
9. See the Note on page 26. Mention that the formal term for a table is **relation**, and the formal term for a row is **tuple**. A row also is called a **record**. Columns in a table are also called **fields** or attributes.
10. DBDL (Database Definition Language) is a commonly accepted shorthand notation for showing the structure of a table. After the name of the table, all of the columns in the table are listed within a set of parentheses. While each column in a table has a distinct name, the same column name can be used in more than one table within the same database.
11. When two or more tables in a database use the same column name, **qualify** the column name; that is, combine the table name and the column name

Teaching Tip	Reinforce the material in this section by using either the Colonial Adventure Tours database or the Solmaris Condominium Group database and asking students to identify the entities, attributes, and relationships. Use review question 11 on page 55 as an in-class exercise to test students' understanding of entities attributes and relationships.

Quick Quiz 1

1. A(n) _____ is a person, place, object, event, or idea for which you want to store and process data.
   Answer: entity
2. A(n) _____ is a characteristic or property of an entity.
   Answer: attribute
3. A(n) _____ is the association between entities.
   Answer: relationship

Functional Dependence

1. Functional dependence is a formal name for what is basically a simple idea. In a relational database, column B is **functionally dependent** on another column A (or possibly a collection of columns) if a value for A determines a single value for B at any one time. Another way of defining functional dependence is to say that A **functionally determines** B.
2. Use Figure 2-4 to explain functional dependence. Make sure that students understand what functional dependence is before proceeding or they will be lost for the remainder of the chapter.
3. Review the embedded Questions and Answers on pages 28 and 29.
4. Use Figures 2-5 and 2-6 to point out that you cannot determine functional dependencies by looking at sample data. You must understand the users' policies

Teaching Tip	Use review question 11 as an in-class exercise to test students' understanding of functional dependencies.

Primary Keys

1. To make each row distinct, one or more columns must uniquely identify a given row in a table. This column or collection of columns is called the **primary key**.
2. A more precise definition for a primary key is the following:
   Column (attribute) A (or a collection of columns) is the **primary key** for a table (relation), R, if:
   Property 1: All columns in R are functionally dependent on A.
   Property 2: No subcollection of the columns in A (assuming that A is a collection of columns and not just a single column) also has Property 1.
3. Review the embedded Questions and Answers on pages 30 and 31 to make sure that students understand the concept of a primary key.
4. Explain that, when using the shorthand representation of a database, the primary key is underlined.
5. Discuss the three Notes on pages 31 and 32.
6. Point out that a **candidate key** is a column or collection of columns on which all columns in the table are functionally dependent. The definition for a primary key really defines a candidate key as well. If two or more columns in a table are identified as candidate keys, choose one to be the primary key. The decision is usually based on the specific application for which the database will be used.

Teaching Tip	Use Review Question 11 as an in-class exercise to test students' understanding of primary keys.

Quick Quiz 2

1. The _____ is the unique identifier for a table.
   Answer: primary key
2. If a table includes one or more columns that can be used as a primary key, both columns are referred to as _____.
   Answer: candidate keys
3. To indicate a table's primary key with a shorthand representation of a database, _____ the column or collection of columns that comprise the primary key.
   Answer: underline

Database Design

1. Point out that the determination of the database requirements is part of the process known as systems analysis.

Design Method

1. Review the design steps given in this section.
2. To design a database for a set of requirements:
   (1) Read the requirements, identify the entities (objects) involved, and name the entities.
   (2) Identify the unique identifiers for the entities identified in step 1.
   (3) Identify the attributes for all of the entities.
   (4) Identify the functional dependencies that exist among the attributes.
   (5) Use the functional dependencies to identify the tables by placing each attribute with the attribute or minimum combination of attributes on which it is functionally dependent.
   (6) Identify any relationships between tables.

Teaching Tip	Use Figure 2-1 as a visual aid as you explain each of the steps above and ask the students to identify the items listed in the steps.

Database Design Requirements

1. Review the requirements that the database for TAL Distributors must support. The database must store specific data about sales reps, customers, items, orders, and order lines.
2. Mention that there are certain constraints, such as, "there is only one customer per order" that the database must enforce.

Teaching	Use Figure 2-1 to illustrate the requirements.

Tip	Use Figure 1-1, which shows a sample order for TAL Distributors.

Database Design Process Example

1. Discuss the six steps to create a database design for TAL Distributors given a set of requirements. It is in this step that students often have trouble
2. Be sure to point out the functional dependencies discussed in step 4. Functional dependency is a difficult concept for some students to grasp.
3. Review the embedded Questions and Answers on pages 38 and 39.

Normalization

1. Stress that database design is an iterative process. Once you create an initial database design, you must analyze it for potential problems.
2. Define **Normalization**. Normalization is a process in which you identify the existence of potential problems, such as data duplication and redundancy, and implement ways to correct these problems. The goal of normalization is to convert unnormalized relations into various types of normal forms.
3. Define an **unnormalized relation**. An unnormalized relation is a relation (table) that contains a repeating group. A table in a particular **normal form** possesses a certain desirable collections of properties.
4. Point out that normalization is a process in which a table that is in first normal form is better than a table that is not in first normal form, a table in second normal form is better than a table in first normal form, and so on. The goal of normalization is to take an initial collection of tables and produce a new collection of tables that represents the same information but is free of problems.

First Normal Form

1. An unnormalized relation is a relation (table) that contains a repeating group. A table (relation) is in **first normal form (1NF)** if it does not contain repeating groups.
2. Use Figures 2-7 and 2-8 to explain converting an unnormalized table to 1NF. In general, when converting a non-first normal form table to first normal form, the primary key usually will include the original primary key concatenated with the key to the repeating group.

Teaching Tip	Students have problems understanding a concatenated primary key. In a relational database, every row must be unique. There are instances when the only way to make each row unique is to consider more than one column as the primary key.

Second Normal Form

1. Use Figure 2-9 to illustrate a relation that is in first normal form but not in second normal form.

2. Point out the **redundancy**; that is, duplication of data in Figure 2-9. This duplication can cause update anomalies.
3. **Update anomalies** occur when a column is dependent on only a portion of the primary key and fall into four categories:

Update	Instead of changing one row, it is necessary to update multiple rows.
Inconsistent data	If the same value appears in more than one row, for example, part description, an update could change one row without changing the other rows.
Additions	Cannot add a record correctly.
Deletions	Cannot delete a record correctly.

4. Emphasize the fact that much real-world data (including relational data) are not well structured and have update anomalies.
5. Define **second normal form (2NF)**. Second normal form eliminates update anomalies caused by partial dependencies. A table (relation) is in second normal form (2NF) if it is in first normal form and no nonkey column is dependent on only a portion of the primary key. A column is a **nonkey column** if it is not a part of the primary key.
6. Point out again that you cannot determine functional dependence by looking at sample data.
7. Mention the Note on page 44. If a relation has a single-column primary key , it automatically is in 2NF.
8. Use Figure 2-10 to explain converting to 2NF.

Third Normal Form

1. Use Figure 2-11 to illustrate update anomalies with a table in 2NF.
2. Define **determinant**. Any column or collection of columns that determines another column is called a determinant. A candidate key is a column or collection of columns that could function as the primary key. Update anomalies also can occur when one nonkey column determines another nonkey column.
3. Define **third normal form (3NF)**. A table (relation) is in third normal form (3NF) if it is in second normal form and the only determinants it contains are candidate keys.
4. Mention the Note on page 48. The definition used in this text for 3NF is really the definition for **Boyce-Codd normal form (BCNF)**.
5. Use Figure 2-12 to explain converting to 3NF. Show students how each progressive normal form solves update problems of the previous normal form.
6. Review the embedded Question and Answer on pages 50 and 51.

Teaching Tip	Point out that normalization is a technique that allows us to analyze the design of a relational database to see whether it is bad. It alerts us to update anomalies and provides a method for correcting those problems. The goal of normalization is to start with a table or collection of tables and produce a new collection of tables that is equivalent (represents the same information) but is free of problems. Emphasize to students that normalization does not add additional attributes or

	remove attributes from tables; it merely rearranges the attributes into an equivalent collection of tables. Many students seem to think that they need to add more attributes to normalize relations in a database. Normalization alerts us to problems with the database design and provides a method to correct those problems. Edgar Codd developed the process and gave it the name normalization.

Quick Quiz 3

1. A relation is in _____ normal form if no repeating groups exist.
   Answer: first

2. If the primary key of a relation contains only a single column, then the relation is automatically in _____ normal form.
   Answer: second

3. Any column (or collection of columns) that determines another column is called a(n) _____.
   Answer: determinant

Diagrams for Database Design

1. Remind students of the old adage that "a picture is worth a thousand words." For many people, a database design is easier to understand if it is depicted in graphical form.
2. Explain an **entity-relationship (E-R) diagram**. In an E-R diagram, a rectangle represents an entity or table. One-to-many relationships between entities are drawn as lines between the corresponding rectangles. There are several different styles of E-R diagrams.
3. Mention the Note on page 52.
4. Use Figures 2-13 through 2-15 to illustrate the different styles of E-R diagrams.

Class Discussion Topics

1. Ask students for other examples of relations (tables) that could have more than one candidate key. Some examples are: state data that contain both state abbreviation and state name and inventory data that contain both a tag number and a serial number.
2. Have students read the Note on page 31. Ask them how they feel about using Social Security number as a primary key. Also ask them for additional databases where Social Security number is being used as a primary key other than banks and places of employment. Some examples are: insurance company, doctor's office, dentist's office.

3. Ask students for examples of unnormalized relations in a student environment. One example is students and courses. Courses will be the repeating group. How would they convert to 1NF?

Additional Projects

1. Place students in teams. Have them design a database to meet the requirements for a student activity database. The database must keep track of information about the student as well as the campus activities he or she participates in. Attributes such as number of years in activity as well as any office held are important. A student may engage in more than one activity.
2. Divide the class into small groups and assign Review Questions 11 and 15 as a group exercise. Make sure that students state their assumptions.

Additional Resources

1. *Concepts of Database Management*, Eighth Edition by Philip Pratt and Mary Last Cengage Learning, 2015.
2. Database Concepts: www.service-architecture.com/database/articles/
3. Normalization: http://databases.about.com/od/specificproducts/a/normalization.htm

Key Terms

- **attribute:** A characteristic or property of an entity
- **Boyce-Codd normal form (BCNF):** A relation is in Boyce-Codd normal form if it is in second normal form and the only determinants it contains are candidate keys; also called third normal form
- **candidate key:** A minimal collection of columns in a table
- **concatenation:** A combination of columns
- **database design:** Process of determining the particular tables and columns that will comprise a database
- **determinant:** A column in a table that determines at least one other column
- **entity:** A person, place, object, event, or idea for which you want to store and process data
- **entity-relationship (E-R) diagram:** A graphical illustration for database design that uses rectangles for entities and arrows for relationships
- **field :** An attribute
- **first normal form (1NF):** A table that does not contain any repeating groups
- **functionally dependent:** Column B is functionally dependent on column A (or on a collection of columns) if a value for A determines a single value for B at any one time
- **functionally determine:** Column A functionally determines column B if B is functionally dependent on A
- **nonkey column:** A column that is not part of the primary key
- **normal form:** A progression that proceeds from first normal form to second normal form to third normal form. A table in a particular normal form possesses a certain desirable collection of properties.
- **normalization:** A process that analyzes a database design to identify the existence of potential problems and implements ways to correct these problems
- **one-to-many relationship:** A relationship in which one entity is associated with many other entities
- **primary key:** The column or collection of columns that uniquely identifies a given row in a table
- **qualify:** To combine a column name with a table name
- **record:** A row in a table
- **redundancy:** Duplication of data
- **relation:** A two-dimensional table in which the entries are single valued; each column has a distinct name (or attribute name); all values in a column are values of the same attribute; the order of the rows and columns is immaterial; and each row contains unique values
- **relational database:** A collection of relations
- **relationship:** The association between entities

- **repeating group:** Multiple entries in an individual location in a table
- **second normal form (2NF):** A table that is in first normal form and where no nonkey column is dependent on only a portion of the primary key
- **third normal form (3NF):** A table that is in second normal form and the only determinants are candidate keys
- **tuple:** A row in a table
- **unnormalized relation:** A table that satisfies the definition of a relation except that it might contain repeating groups
- **update anomaly:** An update problem that can occur in a database as a result of a faulty design