GLOBAL EDITION

Modern Database Management

THIRTEENTH EDITION

Jeffrey A. Hoffer
V. Ramesh
Heikki Topi

THIRTEENTH EDITION

GLOBAL EDITION

MODERN DATABASE MANAGEMENT

Jeffrey A. Hoffer
University of Dayton

V. Ramesh
Indiana University

Heikki Topi
Bentley University

Pearson

Harlow, England • London • New York • Boston • San Francisco • Toronto • Sydney • Dubai • Singapore • Hong Kong
Tokyo • Seoul • Taipei • New Delhi • Cape Town • Sao Paulo • Mexico City • Madrid • Amsterdam • Munich • Paris • Milan

Vice President, IT & Careers: Andrew Gilfillan
Senior Portfolio Manager: Samantha Lewis
Managing Producer: Laura Burgess
Associate Content Producer: Stephany Harrington
Content Producer, Global Edition: Sonam Arora
Assistant Acquisitions Editor, Global Edition: Rosemary Iles
Senior Project Editor, Global Edition: Daniel Luiz
Manager, Media Production, Global Edition: Gargi Banerjee
Manufacturing Controller, Production, Global Edition: Kay Holman
Portfolio Management Assistant: Madeline Houpt
Director of Product Marketing: Brad Parkins
Product Marketing Manager: Heather Taylor
Product Marketing Assistant: Jesika Bethea
Field Marketing Manager: Molly Schmidt
Field Marketing Assistant: Kelli Fisher
Cover Image: mistery/Shutterstock
Vice President, Product Model Management: Jason Fournier
Senior Product Model Manager: Eric Hakanson
Lead, Production and Digital Studio: Heather Darby
Digital Studio Course Producer: Jaimie Noy
Program Monitor: Danica Monzor, SPi Global
Full-Service Project Management: Neha Bhargava, Cenveo® Publisher Services
Composition: Cenveo Publisher Services

Credits and acknowledgments borrowed from other sources and reproduced, with permission, in this textbook appear on the appropriate page within text.

Pearson Education Limited

KAO Two
KAO Park
Harlow
CM17 9NA
United Kingdom

and Associated Companies throughout the world

Visit us on the World Wide Web at: www.pearsonglobaleditions.com

Authorized adaptation from the United States edition, entitled Modern Database Management, *13th edition, ISBN 978-0-13-477365-0, by Jeffrey A. Hoffer, V. Ramesh, and Heikki Topi, published by Pearson Education © 2019.*

ISBN 10: 1-292-26335-0

ISBN 13: 978-1-292-26335-9

eBook ISBN: 978-1-292-26341-0

British Library Cataloguing-in-Publication Data

A catalogue record for this book is available from the British Library.

10 9 8 7 6 5 4 3 2 1

Typeset in Palatino LT Pro by Cenveo® Publisher Services

To Patty, for her sacrifices, encouragement, and support for more than 35 years of being a textbook author widow. To my students and colleagues, for being receptive and critical and for challenging me to be a better teacher.

—J.A.H.

To Gayathri, for her sacrifices and patience these past 25 years. To my parents, for letting me make the journey abroad, and to my cat, Raju, who was a part of our family for more than 20 years.

—V.R.

To Anne-Louise, for her loving support, encouragement, and patience. To Leila and Saara, whose laughter and joy of life continue to teach me about what is truly important. To my teachers, colleagues, and students, from whom I continue to learn every day.

—H.T.

This page intentionally left blank

BRIEF CONTENTS

This page intentionally left blank

CONTENTS

ONLINE CHAPTERS

This page intentionally left blank

PREFACE

This text is designed for introductory courses in database management. Such a course is usually required as part of an information systems curriculum in business schools, computer technology programs, and applied computer science departments. The Association for Information Systems (AIS), the Association for Computing Machinery (ACM), and the International Federation of Information Processing Societies (IFIPS) curriculum guidelines (e.g., IS 2010 and MSIS 2016) all outline this type of database management course or the competencies a student completing the course is expected to have. Previous editions of this text have been used successfully for more than 35 years at both the undergraduate and graduate levels as well as in management and professional development programs.

WHAT'S NEW IN THIS EDITION?

This 13th edition of *Modern Database Management* updates and expands materials in areas undergoing rapid change as a result of improved managerial practices, database design tools and methodologies, and database technology. Later, we detail changes to each chapter. The themes of this 13th edition reflect the major trends in the information systems field and the skills required of modern information systems graduates. The most important changes are as follows:

- The book has been restructured in several important ways. Chapter 7 on databases in applications now also includes segments on transaction integrity, designing multi-user solutions, and application level security, bringing these important perspectives together with their context. The revised chapter on physical database design and database infrastructure (new Chapter 8) includes also coverage of database security, backup and recovery, cloud-based database solutions, and other essential database infrastructure topics. This new comprehensive structure on physical design and infrastructure is now placed after the SQL chapters. The new version of Chapter 9 integrates material on data warehousing and data integrity in a conceptually natural pairing. Recognizing the way in which analytics capabilities rely on all types of data management solutions, Chapter 11, on analytics and implications, is now separate from Chapter 10, on big data. Finally, Chapter 12 brings together data and database administration with data quality, emphasizing the essential connections between the three.
- The part structure of the book has been redesigned to be fully aligned with the new chapter structure.
- We have introduced a new overarching framework (Figure 1-5), which gives our readers a clearer overview of structure of the book and its core topic areas. The framework communicates clearly the increasing importance of informational systems (divided into Analytics–Data Warehousing and Analytics–Big Data) in addition to this book's traditional strength of transactional systems.
- Given the continued and still increasing interest in big data and analytics, we have continued to expand content in this area. The book has now separate chapters on big data technologies (Chapter 10) and analytics (Chapter 11). In addition to general coverage of NoSQL and Hadoop technologies, Chapter 10 provides also detailed examples of MongoDB, Pig, and Hive. Chapter 11 includes extended coverage of R, Python, and Apache Spark—all essential technologies for analytics professionals that allow a link between analytics and data management architectures.
- We emphasize the increasing importance of cloud-based database solutions, mobile technologies, and agile development throughout the book.
- Chapter 1 now better recognizes the broad range of enterprise level applications data management solutions enable and support, including enterprise systems, data warehouses, and data lakes.

- Chapter 7 on databases in applications now includes an extensive example demonstrating the use of Python in the context of database-driven applications.
- The instructor's manual will have more material to support the case Forondo Artist Management Excellence that was introduced in the 12th edition.

In addition to the new topics covered, specific improvements to the textbook have been made in the following areas:

- Every chapter went through significant edits to streamline coverage to ensure relevance with current technologies and eliminate redundancies.
- The entire book has been edited so that its language clearly reflects its focus on the readers as learners instead of authors as teachers
- End-of-chapter material (review questions, problems and exercises, and/or field exercises) in every chapter has been revised with new and modified questions and exercises.
- We continued to update the figures in several chapters to reflect the changing landscape of technologies that are being used in modern organizations.
- The Web Resources section in each chapter was updated to ensure that students have information on the latest database trends and expanded background details on important topics covered in the text.
- The book continues to be available through VitalSource, an innovative e-book delivery system, and as an electronic book in the Kindle format.

Also, we continue to provide on the student Companion Web site several custom-developed short videos that address key concepts and skills from different sections of the book. These videos, produced by the textbook authors, help students learn difficult material by using both the printed text and a mini-lecture or tutorial. Videos have been developed to support Chapters 1 (introduction to database), 2 and 3 (conceptual data modeling), 4 (normalization), and 6 and 7 (SQL). Look for special icons on the opening page of these chapters to call attention to these videos, and go to **www.pearsonglobaleditions.com** to find these videos.

FOR THOSE NEW TO *MODERN DATABASE MANAGEMENT*

Modern Database Management has been a leading text since its first edition in 1983. In spite of this market leadership position, some instructors have used other good database management texts. Why might you want to switch at this time? There are several good reasons:

- One of our goals, in every edition, has been to lead other books in coverage of the latest principles, concepts, and technologies. See what we have added for the 13th edition in "What's New in This Edition?" In the past, we have led in coverage of object-oriented data modeling and UML, Internet databases, data warehousing, and the use of CASE tools in support of data modeling. For the 13th edition, we continue this tradition by continuing to expand and improve coverage of big data and analytics, focusing on what every database student needs to understand about these topics.
- While remaining current, this text focuses on what leading practitioners say is most important for database developers. We work with many practitioners, including the professionals of the Data Management Association (DAMA) and The Data Warehousing Institute (TDWI), leading consultants, technology leaders, and authors of articles in the most widely read professional publications. We draw on these experts to ensure that what the book includes is important and covers not only important entry-level knowledge and skills but also those fundamentals and mind-sets that lead to long-term career success.
- In the 13th edition of this highly successful book, material is presented in a way that has been viewed as very accessible to students. Our methods have been refined through continuous market feedback for more than 35 years as well as through our own teaching. Overall, the pedagogy of the book is sound, and we believe that the new framework that we introduced in Chapter 1 will further strengthen our students'

understanding of the big picture of data management. We use many illustrations that help make important concepts and techniques clear. We use the most modern notations. The organization of the book is flexible, so you can use chapters in whatever sequence makes sense for your students. We supplement the book with data sets to facilitate hands-on, practical learning and with new media resources to make some of the more challenging topics more engaging.

- Our text can accommodate structural flexibility. For example, you may have particular interest in introducing SQL early in your course. Our text makes this possible. First, we cover SQL in depth, devoting two full chapters to this core technology of the database field. Second, we include many SQL examples in early chapters. Third, many instructors have successfully used the two SQL chapters early in their course. Although logically appearing in the life cycle of systems development as Chapters 5 and 6, part of the implementation section of the text, many instructors have used these chapters immediately after Chapter 1 or in parallel with other early chapters. Finally, we use SQL throughout the book, for example, to illustrate Web application connections to relational databases in Chapter 7 and online analytical processing in Chapter 11.
- We have the latest in supplements and Web site support for the text. See the supplement package for details on all the resources available to you and your students.
- This text is written to be part of a modern information systems curriculum with a strong business systems development focus. Topics are included and addressed so as to reinforce principles from other typical courses, such as systems analysis and design, networking, Web site design and development, MIS principles, and application development. Emphasis is on the development of the database component of modern information systems and on the management of the data resource. Thus, the text is practical, supports projects and other hands-on class activities, and encourages linking database concepts to concepts being learned throughout the curriculum the student is taking.

SUMMARY OF ENHANCEMENTS TO EACH CHAPTER

The following sections present a chapter-by-chapter description of the major changes in this edition. Each chapter description presents a statement of the purpose of that chapter, followed by a description of the changes and revisions that have been made for the 13th edition. Each paragraph concludes with a description of the strengths that have been retained from prior editions.

PART I: THE CONTEXT OF DATABASE MANAGEMENT

Chapter 1: The Database Environment and Development Process

This chapter discusses the role of databases in organizations and previews the major topics in the remainder of the text. The primary change to this chapter has been the introduction of a new integrated data management framework (Figure 1-5) and supporting text accompanying it. This framework recognizes the increasing importance of the *informational* systems in addition to the traditional focus of this book on *transactional* systems. After presenting a brief introduction to the basic terminology associated with storing and retrieving data, the chapter presents a well-organized comparison of traditional file processing systems and modern database technology. The chapter then introduces the core components of a database environment. It then goes on to explain the process of database development in the context of structured life cycle, prototyping, and agile methodologies. The chapter also discusses important issues in database development, including management of the diverse group of people involved in database development and frameworks for understanding database architectures and technologies (e.g., the three-schema architecture). Reviewers frequently note the compatibility of this chapter with what students learn in systems analysis and design classes. A brief history of the evolution of database technology, from pre-database files to modern object-relational technologies, is presented. The chapter also provides

an overview of the range of database applications that are currently in use within organizations—personal, multi-tier, and enterprise applications. The explanation of enterprise databases includes databases that are part of enterprise resource planning systems and data warehouses. The chapter concludes with a description of the process of developing a database in a fictitious company, Pine Valley Furniture. This description closely mirrors the steps in database development described earlier in the chapter. The first chapter provides an introduction to the FAME case, which then continues through the book until Chapter 8.

PART II: DATABASE ANALYSIS AND LOGICAL DESIGN

Chapter 2: Modeling Data in the Organization

This chapter presents a thorough introduction to conceptual data modeling with the entity-relationship (E-R) model. The chapter title emphasizes the reason for the E-R model: to unambiguously document the rules of the business that influence database design. Specific subsections explain in detail how to name and define elements of a data model, which are essential in developing an unambiguous E-R diagram. The chapter continues to proceed from simple to more complex examples, and it concludes with a comprehensive E-R diagram for the Pine Valley Furniture Company. In the 13th edition, we have provided six new problems and exercises; these new exercises present some more modern situations, such as Internet of Things applications for databases. A variety of other problems and exercises as well as review questions have been changed to emphasize important topics of the chapter. Appendix A provides information on different data modeling tools and notations.

Chapter 3: The Enhanced E-R Model

This chapter presents a discussion of several advanced E-R data model constructs, primarily supertype/subtype relationships. As in Chapter 2, problems and exercises have been revised, with three new exercises and several building on or extending the new exercises from Chapter 2. The third part of the new FAME case is presented in this chapter. The chapter continues to present thorough coverage of supertype/subtype relationships and includes a comprehensive example of an extended E-R data model for the Pine Valley Furniture Company.

Chapter 4: Logical Database Design and the Relational Model

This chapter describes the process of converting a conceptual data model to the relational data model, as well as how to merge new relations into an existing normalized database. It provides a conceptually sound and practically relevant introduction to normalization, emphasizing the importance of the use of functional dependencies and determinants as the basis for normalization. Concepts of normalization and normal forms are extended in Appendix B. The chapter features a discussion of the characteristics of foreign keys and introduces the important concept of a nonintelligent enterprise key. Enterprise keys (also called surrogate keys for data warehouses) are emphasized as some concepts of object-orientation have migrated into the relational technology world. New problems and exercises are included that draw upon the new problems and exercises from Chapters 2 and 3 for relational modeling and normalization. The chapter continues to emphasize the basic concepts of the relational data model and the role of the database designer in the logical design process.

PART III: DATABASE IMPLEMENTATION AND USE

Chapter 5: Introduction to SQL

This chapter (Chapter 6 in 12th edition) presents a thorough introduction to the SQL used by most DBMSs (SQL:1999) and introduces the changes that are included in the latest standards (SQL: 2011 and SQL:2016). This edition adds coverage of the new features of SQL:2016, including row pattern recognition, JSON support, and extended analytical

capabilities. The new edition also clarifies coverage of SQL data types and, overall, makes it easier to move from relational design in Chapter 4 directly to database implementation without the material on physical database design (now in Chapter 8). The coverage of SQL is extensive and divided between this chapter and Chapter 6. This chapter includes examples of SQL code, using mostly SQL:1999 and SQL:2016 syntax, as well as some Oracle 12c and Microsoft SQL Server syntax. Some unique features of MySQL are mentioned. In this edition, coverage of views has been moved to Chapter 6. Chapter 5 explains the SQL commands needed to create and maintain a database and to program single-table queries. Five review questions and 13 problems and exercises have been added to the chapter or modified extensively. The chapter continues to use the Pine Valley Furniture Company case to illustrate a wide variety of practical queries and query results.

Chapter 6: Advanced SQL

This chapter (Chapter 7 in 12th edition) continues the description of SQL, with a careful explanation of multiple-table queries, transaction integrity, data dictionaries, dynamic and materialized views, triggers and stored procedures (the differences between them are now more clearly explained), and embedding SQL in other programming language programs. All forms of the OUTER JOIN command are covered. Standard SQL (with an updated focus on SQL:2016) is also used. The revised version of the chapter includes now thorough coverage of views and the purposes for which they are used, including their role in enabling security and privacy solutions. This chapter illustrates how to store the results of a query in a derived table, the CAST command to convert data between different data types, and the CASE command for doing conditional processing in SQL. Emphasis continues on the set-processing style of SQL compared with the record processing of programming languages with which the student may be familiar. The section on routines has been revised to provide clarified, expanded, and more current coverage of this topic. The material of transaction integrity, has, however been moved to Chapter 7, where it most naturally belongs. The chapter continues to contain a clear explanation of subqueries and correlated subqueries, two of the most complex and powerful constructs in SQL. At the end, the chapter discusses material that is new to this chapter: data dictionary facilities (in practice, using SQL to understand the structure of the database) and recent extensions and enhancements to SQL. Chapter review material has been updated with 13 new problems and exercises and three new review questions.

Chapter 7: Databases in Applications

This chapter (Chapter 8 in 12th edition) provides a modern discussion of the concepts of client/server architecture and applications, middleware, and database access in contemporary database environments. The chapter has been structurally significantly modified to provide additional clarity, including the integration of material on a two-tiered architecture into the section on three-tiered architecture. In addition to a revised example of writing a Java web application, there is an entire new section—including an extensive and detailed example—on writing Web applications with Python, a widely used general purpose programming language that has become very popular in analytics. Sections on transaction integrity, concurrent access, and application level data security have been revised and moved to this chapter to provide additional conceptual clarity. Material on cloud computing has been moved to Chapter 8 on database infrastructure. Review questions and problems and exercises have been updated.

Chapter 8: Physical Database Design and Database Infrastructure

This chapter (Chapter 5 in the 12th edition) describes the steps that are essential in achieving an efficient database design, with a strong focus on those aspects of database design and implementation that are typically within the control of a database professional in a modern database environment. In addition, several new topics on database infrastructure have been integrated into this chapter to improve the structural clarity of the book, including data dictionaries and repositories, general database software security features, and database backup and recovery. A revised and extended section on cloud-based database infrastructure completes the chapter. Overall, the chapter emphasizes ways to

improve database performance, with references to specific techniques available in Oracle and other DBMSs to achieve this goal. The discussion of indexes includes descriptions of the types of indexes that are widely available in database technologies as techniques to improve query processing speed. Appendix C provides excellent background on fundamental data structures for programs of study that need coverage of this topic. The chapter continues to emphasize the physical design process and the goals of that process. Review questions and problems and exercises have been updated and extended based on the new structure and content of the chapter.

PART IV: ADVANCED DATABASE TOPICS

Chapter 9: Data Warehousing and Data Integration

This chapter describes the basic concepts of data warehousing, the reasons data warehousing is regarded as critical to competitive advantage in many organizations, and the database design activities and structures unique to data warehousing. The most important change of this chapter is the integration of material on data integration (formerly in Chapter 10 in the 12th edition) into it. This change strengthens the readers' ability to understand the essential role of data integration in data warehousing (particularly in ETL and other aspects of data preparation), and it clarifies the structure of the book. Topics covered in this chapter include alternative data warehouse architectures and the dimensional data model (or star schema) for data warehouses. In this edition, additional attention is given to cloud-based implementation of data warehouses. Throughout the chapter, several details have been updated to ensure technical correctness. Operational data store and independent, dependent, and logical data marts are defined. The chapter includes multiple new and revised review questions and problems and exercises.

Chapter 10: Big Data Technologies

This chapter incorporates big data infrastructure material from Chapter 11 in the 12th edition, significantly expanding it and making it more directly applicable with substantial detailed descriptive examples of MongoDB (the most popular NoSQL database) and Pig (scripting language and task automation environment for Hadoop) and Hive (an SQL-like declarative language for querying data stored in Hadoop). This new version of the material gives the students a much more practical, hands-on sense of the purposes for which these well-known tools can be used and how they can serve the goals of big data management. The chapter also includes several new problems and exercises based on these environments. Overall, the chapter helps the readers understand how big data technologies have expanded the possibilities for analytics-driven innovation through advanced informational systems that are pushing boundaries further in terms of volume, velocity, and variety of data while paying continuous attention to value and veracity of big data.

Chapter 11: Analytics and its Implications

Chapter 11 offers integrated coverage of analytics, including descriptive, predictive, and prescriptive analytics. It is based on material on analytics in the big data and analytics chapter in the 12th edition, expanding it with comprehensive new sections on R, Python, and Apache Spark and bringing in material on analytical functions in SQL. The discussion on analytics is linked not only to the coverage of big data but also the material on data warehousing in Chapter 9 and the general discussion on data management in Chapter 1 (as indicated in the new framework in Chapter 1). The chapter also covers approaches and technologies used by analytics professionals, such as on-line analytical processing, data visualization, business performance management and dashboards, data mining, and text mining. Finally, the chapter integrates the coverage of big data and analytics technologies to the individual, organizational, and societal implications of these capabilities. Review questions on the new material have been added.

Chapter 12: Data and Database Administration with Focus on Data Quality

This chapter presents a thorough discussion of the importance and roles of data and database administration and describes a number of the key issues that arise when these functions are performed. This chapter emphasizes the changing roles and approaches of data and database administration, with a renewed and strengthened emphasis on data quality. The chapter both discusses essential characteristics of high-quality data and the mechanisms that organizations need to put in place to enable data quality improvement. Data governance, data availability, and master data management are also covered. The chapter continues to emphasize the critical importance of data and database management in managing data as a corporate asset.

Chapter 13: Distributed Databases

This chapter—available on the book's Web site—reviews the role, technologies, and unique database design opportunities of distributed databases. The objectives and trade-offs for distributed databases, data replication alternatives, factors in selecting a data distribution strategy, and distributed database vendors and products are covered. This chapter provides thorough coverage of database concurrency access controls. Many reviewers have indicated that they are seldom able to cover this chapter in an introductory course, but having the material available is critical for advanced students or special topics.

Chapter 14: Object-Oriented Data Modeling

This chapter presents an introduction to object-oriented modeling using Object Management Group's Unified Modeling Language (UML). This chapter has been carefully reviewed to ensure consistency with the latest UML notation and best industry practices. UML provides an industry-standard notation for representing classes and objects. The chapter continues to emphasize basic object-oriented concepts, such as inheritance, encapsulation, composition, and polymorphism. As with Chapter 13, Chapter 14 is available on the textbook's Web site.

APPENDICES

In the 13th edition three appendices are available on the book's Web site and are intended for those who wish to explore certain topics in greater depth.

Appendix A: Data Modeling Tools and Notation

This appendix addresses a need raised by many readers—how to translate the E-R notation in the text into the form used by the CASE tool or the DBMS used in class. Specifically, this appendix compares the notations of CA ERwin Data Modeler r9.7, Oracle SQL Data Modeler 4.2, SAP Sybase PowerDesigner 16.6, and Microsoft Visio Professional 2016. Tables and illustrations show the notations used for the same constructs in each of these popular software packages.

Appendix B: Advanced Normal Forms

This appendix presents a description (with examples) of Boyce-Codd and fourth normal forms, including an example of BCNF to show how to handle overlapping candidate keys. Other normal forms are briefly introduced. The Web Resources section includes a reference for information on many advanced normal form topics.

Appendix C: Data Structures

This appendix describes several data structures that often underlie database implementations. Topics include the use of pointers, stacks, queues, sorted lists, inverted lists, and trees.

PEDAGOGY

A number of additions and improvements have been made to end-of-chapter materials to provide a wider and richer range of choices for the user. The most important of these improvements are the following:

1. ***Review Questions*** Questions have been updated to support new and enhanced chapter material.
2. ***Problems and Exercises*** This section has been reviewed in every chapter, and many chapters contain new problems and exercises to support updated chapter material. Of special interest are questions in many chapters that give students opportunities to use the data sets provided for the text. Problems and exercises are presented in roughly increasing order of difficulty, which should help instructors and students find exercises appropriate for what they want to accomplish.
3. ***Field Exercises*** This section provides a set of "hands-on" mini-cases that can be assigned to individual students or to small teams of students. Field exercises range from directed field trips to Internet searches and other types of research exercises.
4. ***Case*** The 13th edition of this book includes the same mini-case that was introduced in the 12th edition: Forondo Artist Management Excellence Inc. (FAME). In the first three chapters, the case begins with a description provided in the "voice" of one or more stakeholders, revealing a new dimension of requirements to the reader. Each chapter has project assignments intended to provide guidance on the types of deliverables instructors could expect from students, some of which tie together issues and activities across chapters. These project assignments can be completed by individual students or by small project teams. This case provides an excellent means for students to gain hands-on experience with the concepts and tools they have studied. The instructor's manual will include new materials to support the use of the case.
5. ***Web Resources*** Each chapter contains a list of updated and validated URLs for Web sites that contain information that supplements the chapter. These Web sites cover online publication archives, vendors, electronic publications, industry standards organizations, and many other sources. These sites allow students and instructors to find updated product information, innovations that have appeared since the printing of the book, background information to explore topics in greater depth, and resources for writing research papers.

We continue to provide several pedagogical features that help make the 13th edition widely accessible to instructors and students. These features include the following:

1. ***Learning objectives*** appear at the beginning of each chapter, as a preview of the major concepts and skills students will learn from that chapter. The learning objectives—carefully updated to be aligned with the new chapter structure—also provide a great study review aid for students as they prepare for assignments and examinations.
2. ***Chapter introductions and summaries*** both encapsulate the main concepts of each chapter and link material to related chapters, providing students with a comprehensive conceptual framework for the course.
3. ***The chapter review*** includes the Review Questions, Problems and Exercises, and Field Exercises discussed earlier and also contains a Key Terms list to test the student's grasp of important concepts, basic facts, and significant issues.
4. ***A running glossary*** defines key terms in the page margins as they are discussed in the text. These terms are also defined at the end of the text, in the Glossary of Terms. Also included is the end-of-book Glossary of Acronyms for abbreviations commonly used in database management.

ORGANIZATION

We encourage instructors to customize their use of this book to meet the needs of both their curriculum and student career paths. The modular nature of the text, its broad coverage, its extensive illustrations, and its inclusion of advanced topics and emerging issues make customization easy. The many references to current publications and Web sites

can help instructors develop supplemental reading lists or expand classroom discussion beyond material presented in the text. The use of appendices for several advanced topics allows instructors to easily include or omit these topics.

The modular nature of the text allows the instructor to omit certain chapters or to cover chapters in a different sequence. For example, an instructor who wishes to emphasize data modeling may cover Chapter 14 (available on the book's Web site) on object-oriented data modeling along with or instead of Chapters 2 and 3. An instructor who wishes to cover only basic entity-relationship concepts (but not the enhanced E-R model) may skip Chapter 3 or cover it after Chapter 4 on the relational model.

We have contacted many adopters of *Modern Database Management* and asked them to share with us their syllabi. Most adopters cover the chapters in sequence, but several alternative sequences have also been successful. These alternatives include the following:

- Some instructors cover Chapter 12 on data and database administration immediately after Chapter 8 on physical database design and the relational model.
- To introduce SQL as early as possible, many instructors have effectively covered 12th edition Chapters 6 and 7 (SQL) immediately after Chapter 4; therefore, we have now placed them as Chapters 5 and 6. Some have even covered the new Chapter 5 immediately after Chapter 1, which the book makes possible.
- Many instructors have students read appendices along with chapters, such as reading Appendix on data modeling notations with Chapter 2 or Chapter 3 on E-R modeling, Appendix B on advanced normal forms with Chapter 4 on the relational model, and Appendix C on data structures with Chapter 8.

THE SUPPLEMENT PACKAGE: WWW.PEARSONGLOBALEDITIONS.COM

A comprehensive and flexible technology support package is available to enhance the teaching and learning experience. All instructor and student supplements are available on the text Web site: **www.pearsonglobaleditions.com**.

For Students

The following online resources are available to students:

- *Complete chapters on distributed databases and object-oriented data modeling* as well as appendices focusing on data modeling notations, advanced normal forms, and data structures allow you to learn in depth about topics that are not covered in the textbook.
- *Accompanying databases* are also provided. Two versions of the Pine Valley Furniture Company case have been created and populated for the 13th edition. One version is scoped to match the textbook examples. A second version is fleshed out with more data and tables. This version is not complete, however, so that students can create missing tables and additional forms, reports, and modules. Databases are provided in several formats (ASCII tables, Oracle script, and Microsoft Access), but formats vary for the two versions. Some documentation of the databases is also provided. Both versions of the PVFC database are also provided on Teradata University Network.
- *Several custom-developed short videos that address key concepts and skills from different sections of the book* help students learn material that may be more difficult to understand by using both the printed text and a mini lecture.

For Instructors

The following online resources are available to instructors:

- The *Instructor's Resource Manual* by Heikki Topi, Bentley University, provides chapter-by-chapter instructor objectives, classroom ideas, and answers to Review Questions, Problems and Exercises, Field Exercises, and Project Case Questions.

The Instructor's Resource Manual is available for download on the instructor area of the text's Web site.

- The *Test Bank* and *TestGen*, by John Russo, Wentworth Institute of Technology, includes a comprehensive set of test questions in multiple-choice, true/false, and short-answer format, ranked according to level of difficulty and referenced with page numbers and topic headings from the text. The Test Bank is available in Microsoft Word and as the computerized TestGen. TestGen is a comprehensive suite of tools for testing and assessment. It allows instructors to easily create and distribute tests for their courses, either by printing and distributing through traditional methods or by online delivery via a local area network (LAN) server. Test Manager features Screen Wizards to assist you as you move through the program, and the software is backed with full technical support.
- *PowerPoint presentation slides*, by Michel Mitri, James Madison University, feature lecture notes that highlight key terms and concepts. Instructors can customize the presentation by adding their own slides or editing existing ones.
- The *Image Library* is a collection of the text art organized by chapter. It includes all figures, tables, and screenshots (as permission allows) and can be used to enhance class lectures and PowerPoint slides.
- *Accompanying databases* are also provided. Two versions of the Pine Valley Furniture Company case have been created and populated for the 13th edition. One version is scoped to match the textbook examples. A second version is fleshed out with more data and tables. This version is not complete, however, so that students can create missing tables and additional forms, reports, and modules. Databases are provided in several formats (ASCII tables, Oracle script, and Microsoft Access), but formats vary for the two versions. Some documentation of the databases is also provided. Both versions of the PVFC database are also available on Teradata University Network.

VITALSOURCE eTEXTBOOK

VitalSource eTextbooks were developed for students looking to save on required or recommended textbooks. Students simply select their eText by title or author and purchase immediate access to the content for the duration of the course using any major credit card. With a VitalSource eText, students can search for specific key words or page numbers, take notes online, print out reading assignments that incorporate lecture notes, and bookmark important passages for later review. For more information or to purchase a VitalSource eTextbook, visit **www.vitalsource.com**.

ACKNOWLEDGMENTS

We are grateful to numerous individuals who contributed to the preparation of *Modern Database Management*, 13th edition. First, we wish to thank our reviewers for their detailed suggestions and insights, characteristic of their thoughtful teaching style. As always, analysis of topics and depth of coverage provided by the reviewers were crucial. Our reviewers and others who gave us many useful comments to improve the text include Tamara Babaian, Bentley University; Subhajyoti Bandyopadhyay, University of Florida; Gary Baram, Temple University; Bijoy Bordoloi, Southern Illinois University, Edwardsville; Timothy Bridges, University of Central Oklahoma; Traci Carte, University of Oklahoma; Laurie Crawford, Franklin University; Wingyan Chung, Santa Clara University; Jagdish Gangolly, State University of New York at Albany; Jon Gant, Syracuse University; Jinzhu Gao, University of the Pacific; Monica Garfield, Bentley University; Rick Gibson, American University; Joy Godin, Georgia College & State University; Jian Guan, University of Louisville; Chengqi Guo, James Madison University; Connie Hecker, Missouri Western State University; William H. Hochstettler III, Franklin University; Dinakar Jayarajan, Illinois Institute of Technology; Michael Johnson, Christopher Newport University; Weiling Ke, Clarkson University; Dongwon Lee, Pennsylvania State University; Ingyu Lee, Troy University; Linda

LeSage, Davenport University; Chang-Yang Lin, Eastern Kentucky University; Brian Mennecke, Iowa State University; Kazuo Nakatani, Florida Gulf Coast University; Dat-Dao Nguyen, California State University, Northridge; Fred Niederman, Saint Louis University; Selwyn Piramuthu, University of Florida; Lara Preiser-Houy, California State Polytechnic University, Pomona; John Russo, Wentworth Institute of Technology; Becky Rutherfoord, Kennesaw State University; Ioulia Rytikova, George Mason University; Richard Segall, Arkansas State University; Sharlene Smith, Gaston College; John Snyder, Colorado Mesa University; Josephine Stanley-Brown, Norfolk State University; Chelley Vician, University of St. Thomas; Ruth Weldon, University of St. Francis; and Daniel S. Weaver, Messiah College; Zuopeng Zhang, State University of New York Plattsburgh; Dana Zhu, Iowa State University; Songhua Zu, New Jersey Institute of Technology.

We received excellent input from experts in industry, including Steve Williams (President, DecisionPath Consulting), Tom Victory (DecisionPath Consulting), Todd Walter, Carrie Ballinger, Rob Armstrong, and David Schoeff (all of Teradata Corp); Chad Gronbach and Philip DesAutels (Microsoft Corp.); Peter Gauvin (Ball Aerospace); and Michael Alexander (Open Access Technology, International).

We are very thankful to Ge Yan, Indiana University, for his contributions to some of the technical material in Chapter 7. We also want to thank Heikki Topi, Bentley University, for his role as author of the *Instructor's Resource Manual*. In addition to his duties as author, Heikki took on this additional task and has been diligent in preparing the *Instructor's Resource Manual*; in the process he has helped us clarify and fix various parts of the text. We also want to recognize the important role played by Chelley Vician of the University of St. Thomas, the author of several previous editions of the *Instructor's Resource Manual*; her work added great value to this book. We also thank Sven Aelterman, Troy University, for his many excellent suggestions for improvements and clarifications throughout the text.

We are also grateful to the staff and associates of Pearson for their support and guidance throughout this project. In particular, we wish to thank Senior Portfolio Manager Samantha Lewis for her support through this revision process; Program Monitor Danica Monzor (SPi Global), and Associate Project Manager Neha Bhargava (Cenveo), who kept us on track and made sure everything was complete; and Associate Content Producer Stephany Harrington.

While finalizing this edition of *Modern Database Management*, we pause to remember with deep gratitude the contributions of Dr. Fred McFadden and Dr. Mary Prescott, coauthors of previous editions of this text. Fred and Mary are not with us anymore, but their contributions to *MDBM*, both content and spirit, continue to be directly and indirectly included in this book.

Finally, we give immeasurable thanks to our spouses, who endured many evenings and weekends of solitude for the thrill of seeing a book cover hang on a den wall. In particular, we marvel at the commitment of Patty Hoffer, who has lived the lonely life of a textbook author's spouse through 13 editions over more than 35 years of late-night and weekend writing. We also want to sincerely thank Anne-Louise Klaus for being willing to continue her wholehearted support for Heikki's involvement in the project. Although the book project was no longer new for Gayathri Mani, her continued support and understanding are very much appreciated. Much of the value of this text is due to their patience, encouragement, and love, but we alone bear the responsibility for any errors or omissions between the covers.

Jeffrey A. Hoffer

V. Ramesh

Heikki Topi

GLOBAL EDITION ACKNOWLEDGMENTS

Pearson would like to thank the following people for their work on the Global Edition:

Contributors

Imran Medi, Asia Pacific University of Technology and Innovation
Sahil Raj, Punjabi University
Shamikh Siddiqui, Jumeira University, Dubai

Reviewers

Thomas Chesney, University of Nottingham
Kamran Munir, University of the West of England
Liyana Shuib, University of Malaya
Shaomin Wu, The University of Kent

PART I

The Context of Database Management

AN OVERVIEW OF PART I

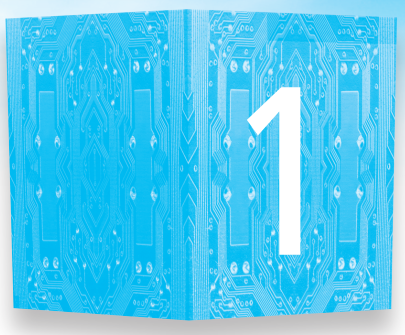
Chapter 1
The Database Environment and Development Process

In this chapter and opening part of the book, we set the context and provide basic database concepts and definitions used throughout the text. In this part, you will understand database management as an exciting, challenging, and growing field that provides numerous career opportunities for information systems students. Databases continue to become a more common part of everyday living and a more central component of business operations. From the database that stores contact information in your smartphone or tablet to the very large databases that support enterprise-wide information systems and provide important insights for organizational decision makers, databases have become the central points of data storage that were originally envisioned decades ago. Customer relationship management and Internet shopping are examples of two database-dependent activities that have developed in recent years. The development of data warehouses and "big data" repositories that provide managers the opportunity for deeper and broader historical analysis of data and specific guidance for future actions also continues to take on more importance.

We begin by providing basic definitions of *data*, *database*, *metadata*, *database management system*, *data warehouse*, and other terms associated with this environment. We compare databases with the older file management systems they replaced and describe several important advantages that are enabled by the carefully planned use of databases. You will see a framework that provides an integrated perspective to both transactional and analytic use of various data management technologies to be used throughout this book and in your career.

The chapter also describes the general steps followed in the analysis, design, implementation, and administration of databases. Further, this chapter also illustrates how the database development process fits into the overall information systems development process. Database development for both structured life cycle and prototyping methodologies is explained. We introduce enterprise data modeling, which sets the range and general contents of organizational databases. This is often the first step in database development. You will learn about the concept of schemas and the three-schema architecture, which is the dominant approach in modern database systems. We describe the major components of the database environment and the types of applications as well as multi-tier and enterprise databases. Enterprise databases include those that are used to support enterprise resource planning systems and data

warehouses. Finally, we describe the roles in which you might be involved as part of a database development project. The Pine Valley Furniture Company case is introduced and used to illustrate many of the principles and concepts of database management. This case is used throughout the text as a continuing example of the use of database management systems.



The Database Environment and Development Process

LEARNING OBJECTIVES

After studying this chapter, you should be able to:

- Concisely define each of the following key terms: **database, data, information, metadata, database application, data model, entity, relational database, database management system (DBMS), data independence, user view, constraint, data modeling and design tools, repository, enterprise data modeling, systems development life cycle (SDLC), conceptual schema, logical schema, physical schema, prototyping, agile software development, project, enterprise resource planning (ERP) system, data warehouse,** and **data lake**.
- Name several limitations of conventional file processing systems.
- Explain at least 10 advantages of the database approach compared to traditional file processing.
- Identify several costs and risks of the database approach.
- Distinguish between operational (transactional) and informational (data warehousing and big data) data management approaches and related technologies.
- List and briefly describe nine components of a typical database environment.
- Identify four categories of applications that use databases and their key characteristics.
- Describe the life cycle of a systems development project, with an emphasis on the purpose of database analysis, design, and implementation activities.
- Explain the prototyping and agile-development approaches to database and application development.
- Explain the roles of individuals who design, implement, use, and administer databases.
- Explain the differences between personal, multi-tiered, and enterprise-level data management solutions.
- Explain the differences among external, conceptual, and internal schemas and the reasons for the use of a three-schema architecture for databases.

Visit *www.pearsonglobaleditions.com* to view the accompanying video for this chapter.

DATA MATTER!

The amount of data being generated, stored, and processed is growing by leaps and bounds. According to a McKinsey Global Institute Report (Manyika et al., 2011), it is estimated that in 2010 alone, global enterprises stored more than 7 exabytes of data (an exabyte is a billion gigabytes), while consumers stored more than 6 exabytes of new data on devices such as personal computers, smartphones, tablets, and notebooks. That is a lot of data! As more and more of the world becomes digital and the products we use every day, such as watches, refrigerators, and so forth, become smarter, the amount of data that needs to be generated, stored, and processed will only continue to grow.

The availability of all of these data is also opening up unparalleled opportunities for companies to leverage data for various purposes. A recent study by IBM (IBM, 2011) shows that one of the top priorities for CEOs in the coming years is the ability to use insights and intelligence that can be gleaned from data for competitive advantage. The McKinsey Global Institute Report (Manyika et al., 2011) estimates that by appropriately leveraging the data available to them, the U.S. retail industry can see up to a 60 percent increase in net margin, and manufacturing can realize up to a 50 percent reduction in product development costs.

The availability of large amounts of data is also fueling innovation in companies and allowing them to think differently and creatively about various aspects of their businesses. Below you will find some examples from a variety of domains:

1. The Memorial Sloan-Kettering Cancer center is using IBM Watson (do you remember Watson beating Ken Jennings in *Jeopardy*?) to help analyze the information from medical literature, research, past case histories, and best practices to help provide oncologists with evidence-based recommendations (**www-935.ibm.com/services/multimedia/MSK_Case_Study_IMC14794.pdf**).
2. Continental Airlines (now United) invested in a real-time business intelligence capability and was able to dramatically improve its customer service and operations. For example, it can now track whether a high-value customer is experiencing a delay in a trip, where and when the customer will arrive at the airport, and the gate the customer must go to make the next connection (Anderson-Lehman et al., 2004).
3. A leading fast-food chain uses video information from its fast-food lane to determine what food products to display on its (digital) menu board. If the lines are long, the menu displays items that can be served quickly. If the lines are short, the menu displays higher-margin but slower-to-prepare items (Laskowski, 2014).
4. Nagoya Railroad analyzes data about its customers' travel habits along with their shopping and dining habits to better understand its customers. For example, it was able to identify that young women who used a particular train station for their commute also tended to eat at a particular type of restaurant and buy from certain types of stores. This information allows Nagoya Railroad to create a targeted marketing campaign (**http://public.dhe.ibm.com/common/ssi/ecm/en/ytc03707usen/YTC03707USEN.PDF**).
5. Kroger, a fast-growing grocery store chain, was able to increase the return rates of its direct mail campaigns to a very high level of 70 percent by personalizing the offers based on the data the company had collected regarding their customer's purchasing behavior (Groenfeldt, 2013).

At the heart of all the above examples is the ability to collect, organize, and manage data. This is precisely the focus of this textbook. This understanding will give you the power to support any business strategy and the deep satisfaction that comes from knowing how to organize data so that financial, marketing, or customer service questions can be answered almost as soon as they are asked. Enjoy!

INTRODUCTION

Over the past two decades, data have become strategic assets for most organizations. Databases store, manipulate, and retrieve data in nearly every type of organization, including business, health care, education, government, libraries, and many scientific fields. Individuals with various personal devices and employees using enterprise-wide distributed applications depend on database technology. Customers and other remote users access databases through diverse technologies, such as automated teller machines, Web browsers, smartphones, and intelligent living and office environments. Most Web-based applications depend on a database foundation.

Following this period of rapid growth, will the demand for databases and database technology level off? Very likely not! In the highly competitive environment of today, there is every indication that database technology will assume even greater importance. Managers seek to use knowledge derived from databases for competitive advantage. For example, detailed sales databases can be mined to determine customer buying patterns as a basis for advertising and marketing campaigns. Organizations embed procedures called *alerts* in databases to warn of unusual conditions, such as impending stock shortages or opportunities to sell additional products, and to trigger appropriate actions. Analytics in its various forms—including big data analytics—depends on databases and other data management technologies.

Although the future of databases is assured, much work remains to be done. Many organizations have a proliferation of incompatible databases that were developed to meet immediate needs rather than based on a planned strategy or a well-managed evolution. Enormous amounts of data are trapped in older, "legacy" systems, and the data are often of poor quality. New skills are required to design and manage data warehouses and other repositories of data and to fully leverage all the data that are being captured in the organization. There is a shortage of skills in areas such as database analysis, database design, database application development, and business analytics. You will learn about these and other important issues in this textbook to equip you for the jobs of the future.

A course in database management has emerged as one of the most important courses in the information systems curriculum today. Further, many schools have added additional elective courses in data warehousing, data mining, and other aspects of business analytics to provide in-depth coverage of these important topics. As information systems professionals, you must be prepared to analyze database requirements and design and to implement databases within the context of information systems development. You also must be prepared to consult with end users and show them how they can use databases (or data warehouses) to build decision models and systems for competitive advantage. The widespread use of databases attached to Web sites that return dynamic information to users of these sites requires that you understand not only how to link databases to the Web-based applications but also how to secure those databases so that their contents can be viewed but not compromised by outside users.

In this chapter, you will learn about the basic concepts of databases and database management systems (DBMSs). You will review traditional file management systems and some of their shortcomings that led to the database approach. Next, you will consider the benefits, costs, and risks of using the database approach. We review the range of technologies used to build, use, and manage databases; describe the types of applications that use databases (personal, multi-tier, and enterprise); and describe how databases have evolved over the past five decades. The chapter also presents a framework that will help you understand traditional and emerging data management approaches and technologies in a joint context.

Because a database is one part of an information system, this chapter also examines how the database development process fits into the overall information systems development process. The chapter emphasizes the need to coordinate database development with all the other activities in the development of a complete

information system. It includes highlights from a hypothetical database development process at Pine Valley Furniture Company. Using this example, the chapter introduces tools for developing databases on personal computers and the process of extracting data from enterprise databases for use in stand-alone applications.

There are several reasons for discussing database development at this point. First, although you may have used the basic capabilities of a database management system, such as Microsoft Access, you may not yet have developed an understanding of how these databases were developed. Using simple examples, this chapter briefly illustrates what you will be able to do after you complete a database course using this text. Thus, this chapter helps you develop a vision and context for each topic developed in detail in subsequent chapters.

Second, many students learn best from a text full of concrete examples. Although all of the chapters in this text contain numerous examples, illustrations, and actual database designs and code, each chapter concentrates on a specific aspect of database management. This chapter will help you understand, with minimal technical details, how all of these individual aspects of database management are related and how database development tasks and skills relate to what you are learning in other information systems courses.

Finally, many instructors want you to begin the initial steps of a database development group or individual project early in your database course. This chapter gives you an idea of how to structure a database development project sufficient to begin a course exercise. Obviously, because this is only the first chapter, many of the examples and notations you will encounter in this chapter are much simpler than those required for your project, for other course assignments, or in a real organization.

One note of caution: You will not learn how to design or develop databases just from this chapter. Sorry! You will discover that the content of this chapter is introductory and simplified. Many of the notations used in this chapter are not exactly like the ones you will learn in subsequent chapters. Our purpose in this chapter is to give you a general understanding of the key steps and types of skills, not to teach you specific techniques. You will, however, learn fundamental concepts and definitions and develop an intuition and motivation for the skills and knowledge presented in later chapters.

BASIC CONCEPTS AND DEFINITIONS

Database
An organized collection of logically related data.

A **database** is an organized collection of logically related data. Not many words in the definition, but have you looked at the size of this book? There is a lot to do to fulfill this definition.

A database may be of any size and complexity. For example, a salesperson may maintain a small database of customer contacts—consisting of a few megabytes of data—on her laptop computer. A large corporation may build a large database consisting of several terabytes of data (a *terabyte* is a trillion bytes) on a large mainframe computer that is used for decision support applications. Very large data warehouses contain more than a petabyte of data. (A *petabyte* is a quadrillion bytes.) The assumption throughout the text is that all databases are computer based.

Data

Historically, the term *data* referred to facts concerning objects and events that could be recorded and stored on computer media. For example, in a salesperson's database, the data would include facts such as customer name, address, and telephone number. This type of data is called *structured* data. The most important structured data types are numeric, character, and dates. Structured data are stored in tabular form (in tables, relations, arrays, spreadsheets, and so forth) and are most commonly found in traditional databases and data warehouses.

The traditional definition of data now needs to be expanded to reflect a new reality: Databases today are used to store objects such as documents, e-mails, tweets, Facebook

posts, GPS information, maps, photographic images, sound, and video segments in addition to structured data. For example, the salesperson's database might include a photo image of the customer contact. It might also include a sound recording or video clip about the most recent product. This type of data is referred to as *unstructured* data, or as multimedia data. Today, structured and unstructured data are often combined in the same database to create a true multimedia environment. For example, an automobile repair shop can combine structured data (describing customers and automobiles) with multimedia data (photo images of the damaged autos and scanned images of insurance claim forms). One of the defining elements of "big data" technologies, which will also be covered later in this book, is that they provide capabilities to deal with highly heterogeneous data.

An expanded definition of **data** that includes structured and unstructured types is "a stored representation of objects and events that have meaning and importance in the user's environment."

Data

Stored representations of objects and events that have meaning and importance in the user's environment.

Data versus Information

The terms *data* and *information* are closely related and in fact are often used interchangeably. However, it is useful to distinguish between data and information. **Information** is data that have been processed in such a way that the knowledge of the person who uses the data is increased. For example, consider the following list of facts:

Information

Data that have been processed in such a way as to increase the knowledge of the person who uses the data.

Baker, Kenneth D.	324917628
Doyle, Joan E.	476193248
Finkle, Clive R.	548429344
Lewis, John C.	551742186
McFerran, Debra R.	409723145

These facts satisfy our definition of data, but most people would agree that the data are useless in their present form. Even if you guessed that this is a list of people's names paired with their Social Security numbers, the data remain useless because you would have no idea what the entries mean. Notice what happens when you see the same data in a context, as shown in Figure 1-1a.

By adding a few additional data items and providing some structure, you are able to recognize a class roster for a particular course. This is useful information to some users, such as the course instructor and the registrar's office. Of course, as general awareness of the importance of strong data security has increased, few organizations use Social Security numbers as identifiers any longer. Instead, most organizations use an internally generated number for identification purposes.

Another way to convert data into information is to summarize them or otherwise process and present them for human interpretation. For example, Figure 1-1b shows

Class Roster

Course: MGT 500 Business Policy Semester: Spring 2018

Section: 2

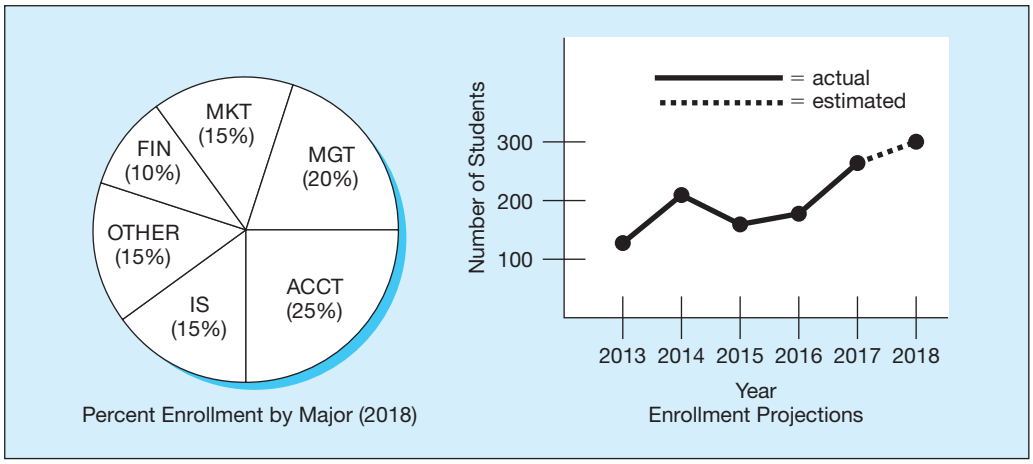
Name	ID	Major	GPA
Baker, Kenneth D.	324917628	MGT	2.9
Doyle, Joan E.	476193248	MKT	3.4
Finkle, Clive R.	548429344	PRM	2.8
Lewis, John C.	551742186	MGT	3.7
McFerran, Debra R.	409723145	IS	2.9
Sisneros, Michael	392416582	ACCT	3.3

(a) Data in context

FIGURE 1-1 Converting data to information

FIGURE 1-1 ***(continued)***

(b) Summarized data



summarized student enrollment data presented as graphical information. This information could be used as a basis for deciding whether to add new courses or to hire new faculty members.

In practice, according to our definitions, databases today may contain data, information, or both. For example, a database may contain an image of the class roster document shown in Figure 1-1a. Also, data are often preprocessed and stored in summarized form in databases that are used for decision support. Throughout this text, we use the term *database* without distinguishing its contents as data or information.

Metadata

Metadata

Data that describe the properties or characteristics of end-user data and the context of those data.

As discussed earlier, data become useful only when placed in some context. The primary mechanism for providing context for data is metadata. **Metadata** are data that describe the properties or characteristics of end-user data and the context of that data. Some of the properties that are typically described include data names, definitions, length (or size), and allowable values. Metadata describing data context include the source of the data, where the data are stored, ownership (or stewardship), and usage. Although it may seem circular, many people think of metadata as "data about data."

Some sample metadata for the Class Roster (Figure 1-1a) are listed in Table 1-1. For each data item that appears in the Class Roster, the metadata show the data item name, the data type, length, minimum and maximum allowable values (where appropriate), a brief description of each data item, and the source of the data (sometimes called the *system of record*). Notice the distinction between data and metadata. Metadata are once removed from data. That is, metadata describe the properties of data but are separate from that data. Thus, the metadata shown in Table 1-1 do not include any sample data from the Class Roster of Figure 1-1a. Metadata enable database designers and users to understand what data exist, what the data mean, and how to distinguish between data items

TABLE 1-1 Example Metadata for Class Roster

Data Item			Metadata			
Name	Type	Length	Min	Max	Description	Source
Course	Alphanumeric	30			Course ID and name	Academic Unit
Section	Integer	1	1	9	Section number	Registrar
Semester	Alphanumeric	10			Semester and year	Registrar
Name	Alphanumeric	30			Student name	Student IS
ID	Integer	9			Student ID (SSN)	Student IS
Major	Alphanumeric	4			Student major	Student IS
GPA	Decimal	3	0.0	4.0	Student grade point average	Academic Unit

that at first glance look similar. Managing metadata is at least as crucial as managing the associated data because data without clear meaning can be confusing, misinterpreted, or erroneous. Typically, much of the metadata are stored as part of the database and may be retrieved using the same approaches that are used to retrieve data or information.

Data can be stored in files (think Excel sheets) or in databases. In the following sections, you will learn about the progression from file processing systems to databases and the advantages and disadvantages of each.

TRADITIONAL FILE PROCESSING SYSTEMS

When computer-based data processing was first available, there were no databases. To be useful for business applications, computers had to store, manipulate, and retrieve large files of data. Computer file processing systems were developed for this purpose. Although these systems have evolved over time, their basic structure and purpose have changed little over several decades.

As business applications became more complex, it became evident that traditional file processing systems had a number of shortcomings and limitations (described next). As a result, these systems have been replaced by database processing systems in most business applications today. Nevertheless, you should have at least some familiarity with file processing systems since understanding the problems and limitations inherent in file processing systems can help you avoid these same problems when designing database systems. It should be noted that Excel files, in general, fall into the same category as file systems and suffer from the same drawbacks listed below. Informal use of Excel for management of data is believed to continue to be quite widespread, although valid research results regarding this are difficult to find.

File Processing Systems at Pine Valley Furniture Company

Early computer applications at Pine Valley Furniture used the traditional file processing approach. This approach to information systems design met the data processing needs of individual departments rather than the overall information needs of the organization. The information systems group typically responded to users' requests for new systems by developing (or acquiring) new computer programs for individual applications, such as inventory control, accounts receivable, or human resource management. No overall map, plan, or model guided application growth.

Three of the computer applications based on the file processing approach are shown in Figure 1-2. The systems illustrated are Order Filling, Invoicing, and Payroll.

FIGURE 1-2 Old file processing systems at Pine Valley Furniture Company

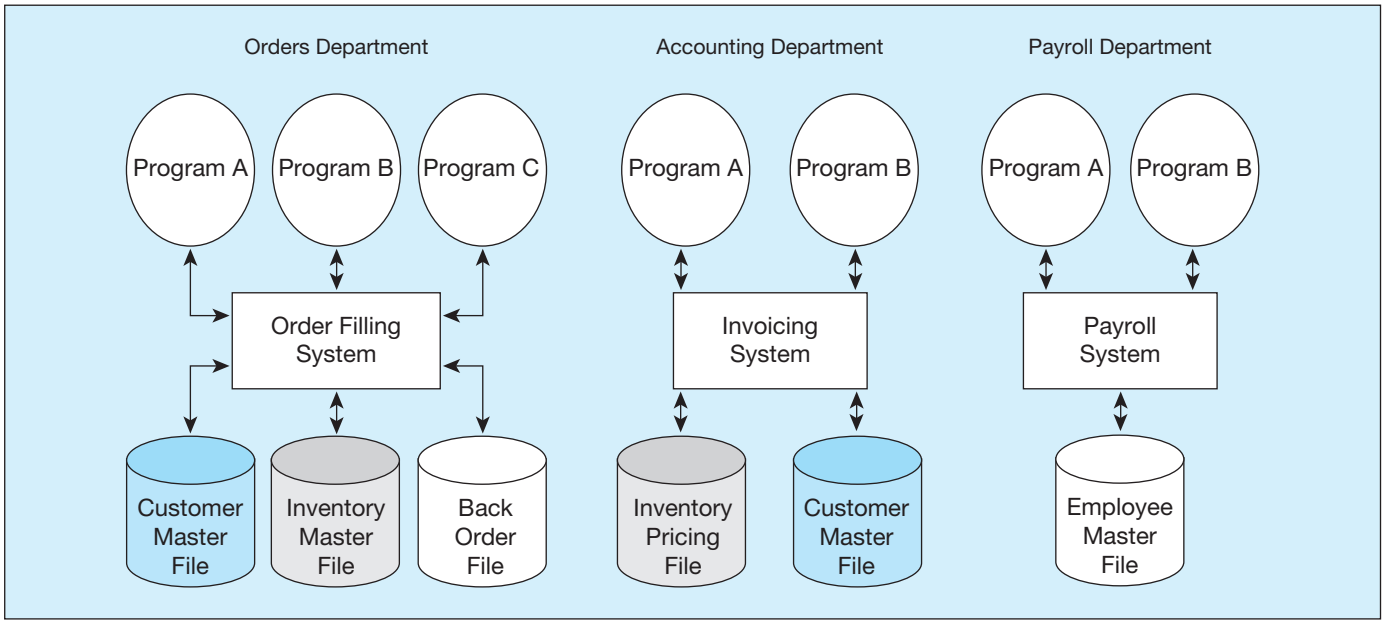


TABLE 1-2 Disadvantages of File Processing Systems
Program-data dependence
Duplication of data
Limited data sharing
Lengthy development times
Excessive program maintenance

The figure also shows the major data files associated with each application. A *file* is a collection of related records. For example, the Order Filling System has three files: Customer Master, Inventory Master, and Back Order. Notice that there is duplication of some of the files used by the three applications, which is typical of file processing systems.

Disadvantages of File Processing Systems

Several disadvantages associated with conventional file processing systems are listed in Table 1-2 and described briefly next. It is important to understand these issues because if you do not follow the database management practices described in this book, some of these disadvantages can also become issues for databases as well.

Database application
An application program (or set of related programs) that is used to perform a series of database activities (create, read, update, and delete) on behalf of database users.

PROGRAM-DATA DEPENDENCE File descriptions are stored within each **database application** program that accesses a given file. For example, in the Invoicing System in Figure 1-2, Program A accesses the Inventory Pricing File and the Customer Master File. Because the program contains a detailed file description for these files, any change to a file structure requires changes to the file descriptions for all programs that access the file.

Notice in Figure 1-2 that the Customer Master File is used in the Order Filling System and the Invoicing System. Suppose it is decided to change the customer address field length in the records in this file from 30 to 40 characters. The file descriptions in each program that is affected (up to five programs) would have to be modified. It is often difficult even to locate all programs affected by such changes. Worse, errors are often introduced when making such changes.

DUPLICATION OF DATA Because applications are often developed independently in file processing systems, unplanned duplicate data files are the rule rather than the exception. For example, in Figure 1-2, the Order Filling System contains an Inventory Master File, whereas the Invoicing System contains an Inventory Pricing File. These files contain data describing Pine Valley Furniture Company's products, such as product description, unit price, and quantity on hand. This duplication is wasteful because it requires additional storage space and increased effort to keep all files up to date. Data formats may be inconsistent, data values may not agree, or both. Reliable metadata are very difficult to establish in file processing systems. For example, the same data item may have different names in different files, or, conversely, the same name may be used for different data items in different files.

LIMITED DATA SHARING With the traditional file processing approach, each application has its own private files, and users have little opportunity to share data outside their own applications. Notice in Figure 1-2, for example, that users in the Accounting Department have access to the Invoicing System and its files, but they probably do not have access to the Order Filling System or to the Payroll System and their files. Managers often find that a requested report requires a major programming effort because data must be drawn from several incompatible files in separate systems. When different organizational units own these different files, additional management barriers must be overcome.

LENGTHY DEVELOPMENT TIMES With traditional file processing systems, each new application requires that the developer essentially start from scratch by designing

new file formats and descriptions and then writing the file access logic for each new program. The lengthy development times required are inconsistent with today's fast-paced business environment, in which time to market (or time to production for an information system) is a key business success factor.

EXCESSIVE PROGRAM MAINTENANCE The preceding factors all combined to create a heavy program maintenance load in organizations that relied on traditional file processing systems. In fact, as much as 80 percent of the total information system's development budget might be devoted to program maintenance in such organizations. This, in turn, means that resources (time, people, and money) are not being spent on developing new applications.

As discussed above, these disadvantages are true also in situations when individuals and organizational units maintain important organizational data in Excel spreadsheets. Further, it is important to note that many of the disadvantages of file processing you have learned about can also be limitations of databases if an organization does not properly apply the database approach. For example, if an organization develops many separately managed databases (say, one for each division or business function) with little or no coordination of the metadata, uncontrolled data duplication, limited data sharing, lengthy development time, and excessive program maintenance can occur. Thus, the database approach, which is explained in the next section, is as much a way to manage organizational data as it is a set of technologies for defining, creating, maintaining, and using these data.

THE DATABASE APPROACH

So, how do you overcome the flaws of file processing? No, you do not call Ghostbusters, but you can do something better: You should follow the database approach. You will first learn some core concepts that are fundamental in understanding the database approach to managing data. You will then discover how the database approach can overcome the limitations of the file processing approach.

Data Models

Designing a database properly is fundamental to establishing a database that meets the needs of the users. **Data models** capture the nature of and relationships among data and are used at different levels of abstraction as a database is conceptualized and designed. The effectiveness and efficiency of a database is directly associated with the structure of the database. Various graphical systems exist that convey this structure and are used to produce data models that can be understood by end users, systems analysts, and database designers. Chapters 2 and 3 are devoted to developing your understanding of data modeling, as is Chapter 14, on the book's Web site, which addresses a different approach using object-oriented data modeling. A typical data model is made up of entities, attributes, and relationships, and the most common data modeling representation is the entity-relationship model. A brief description is presented next. More details will be forthcoming in Chapters 2 and 3.

Data model
Graphical systems used to capture the nature and relationships among data.

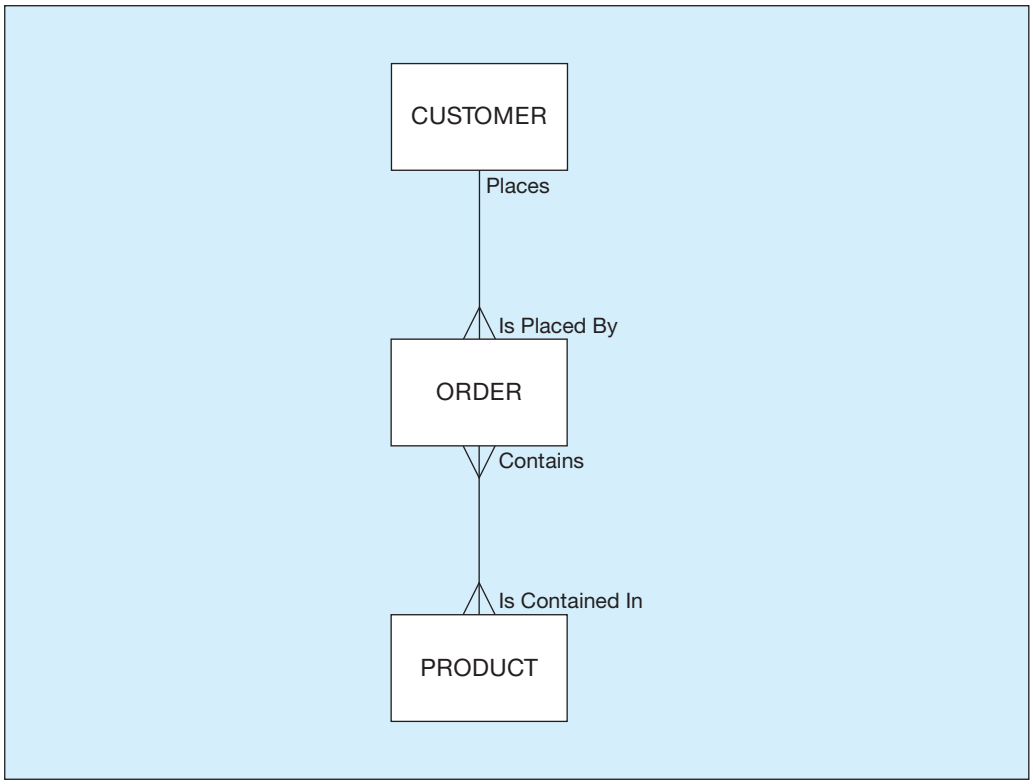
ENTITIES Customers and orders are objects about which a business maintains information. They are referred to as "entities." An **entity** is like a noun in that it describes a person, a place, an object, an event, or a concept in the business environment for which information must be recorded and retained. CUSTOMER and ORDER are entities in Figure 1-3a. The data you are interested in capturing about the entity (e.g., Customer Name) is called an *attribute*. Data are recorded for many customers. Each customer's information is referred to as an *instance* of CUSTOMER.

Entity
A person, a place, an object, an event, or a concept in the user environment about which the organization wishes to maintain data.

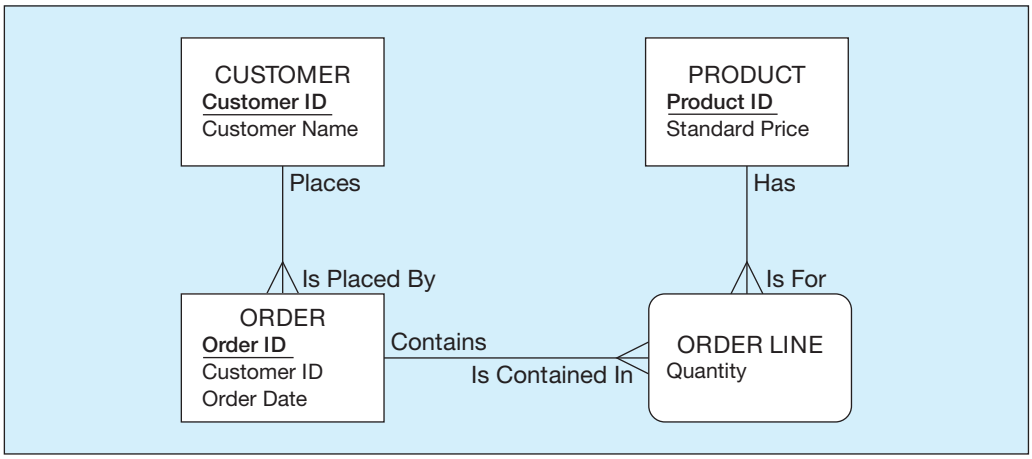
RELATIONSHIPS A well-structured database establishes the *relationships* between entities that exist in organizational data so that desired information can be retrieved. Most relationships are one-to-many (1:*M*) or many-to-many (*M*:*N*). A customer can place (the Places relationship) more than one order with a company. However, each

FIGURE 1-3 **Comparison of enterprise- and project-level data models**

(a) Segment of an enterprise data model



(b) Segment of a project data model



order is usually associated with (the Is Placed By relationship) a particular customer. Figure 1-3a shows the 1:*M* relationship of customers who may place one or more orders; the 1:*M* nature of the relationship is marked by the crow's foot attached to the rectangle (entity) labeled ORDER. This relationship appears to be the same in Figures 1-3a and 1-3b. However, the relationship between orders and products is *M*:*N*. An order may be for one or more products, and a product may be included on more than one order. It is worthwhile noting that Figure 1-3a is an enterprise-level model, where it is necessary to include only the higher-level relationships of customers, orders, and products. The project-level diagram shown in Figure 1-3b includes additional levels of details, such as the further details of an order.

Relational database

A database that represents data as a collection of tables in which all data relationships are represented by common values in related tables.

Relational Databases

Relational databases establish the relationships between entities by means of common fields included in a file, called a *relation*. The relationship between a customer and the customer's order depicted in the data models in Figure 1-3 is established by including the customer's number with the customer's order. Thus, a customer's identification number

is included in the file (or relation) that holds customer information such as name, address, and so forth. Every time the customer places an order, the customer identification number is also included in the relation that holds order information. Relational databases use the identification number to establish the relationship between customer and order.

Database Management Systems

A **database management system (DBMS)** is a software system that enables the use of a database approach. The primary purpose of a DBMS is to provide a systematic method of creating, updating, storing, and retrieving the data stored in a database. It enables end users and application programmers to share data, and it enables data to be shared among multiple applications rather than propagated and stored in new files for every new application (Mullins, 2002). A DBMS also provides facilities for controlling data access, enforcing data integrity, managing concurrency control, and restoring a database. You will learn about these DBMS features in detail in Chapters 7 and 8.

Database management system (DBMS)
A software system that is used to create, maintain, and provide controlled access to user databases.

Now that you understand the basic elements of a database approach, you are in a good position to try to understand the differences between a database approach and a file-based approach. Let us begin by comparing Figures 1-2 and 1-4. Figure 1-4 depicts a representation (entities) of how the data can be considered to be stored in the database. Notice that unlike Figure 1-2, in Figure 1-4, there is only one place where the CUSTOMER information is stored rather than the two Customer Master Files. Both the Order Filling System and the Invoicing System will access the data contained in the single CUSTOMER entity. Further, what CUSTOMER information is stored, how it is stored, and how it is accessed are likely not closely tied to either of the two systems. All of this enables you to achieve the advantages listed in the next section. Of course, it is important to note that a real-life database will likely include thousands of entities and relationships among them.

Advantages of the Database Approach

The primary advantages of a database approach, enabled by DBMSs, are summarized in Table 1-3 and described next.

PROGRAM-DATA INDEPENDENCE The separation of data descriptions (metadata) from the application programs that use the data is called **data independence.** With the database approach, data descriptions are stored in a central location called the *repository*.

Data independence
The separation of data descriptions from the application programs that use the data.

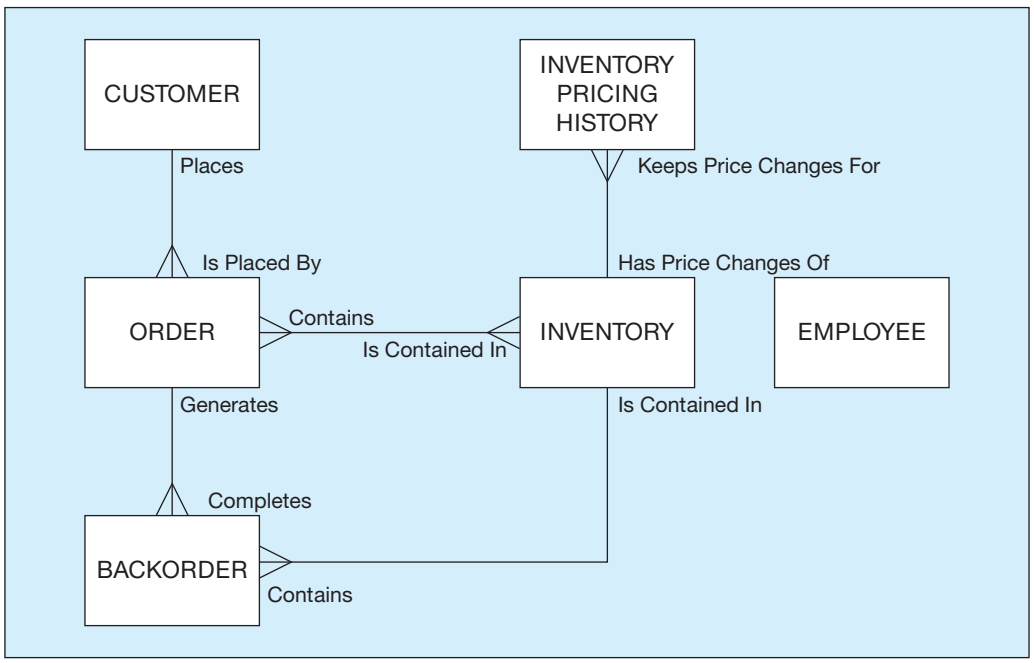


FIGURE 1-4 Enterprise model for Figure 1-3 segments

TABLE 1-3 Advantages of the Database Approach

Program-data independence
Planned data redundancy
Improved data consistency
Improved data sharing
Increased productivity of application development
Enforcement of standards
Improved data quality
Improved data accessibility and responsiveness
Reduced program maintenance
Improved decision support

This property of database systems allows an organization's data to change and evolve (within limits) without changing the application programs that process the data.

PLANNED DATA REDUNDANCY Good database design attempts to integrate previously separate (and redundant) data files into a single, logical structure. Ideally, each primary fact is recorded in only one place in the database. For example, facts about a product, such as the Pine Valley oak computer desk, its finish, price, and so forth, are recorded together in one place in the Product table, which contains data about each of Pine Valley's products. The database approach does not eliminate redundancy entirely, but it enables the designer to control the type and amount of redundancy. At other times, it may be desirable to include some limited redundancy to improve database performance, as you will see in later chapters.

IMPROVED DATA CONSISTENCY By eliminating or controlling data redundancy, you can greatly reduce the opportunities for inconsistency. For example, if a customer's address is stored only once, we cannot disagree about the customer's address. When the customer's address changes, recording the new address is greatly simplified because the address is stored in a single place. Finally, you avoid wasting storage space that results from redundant data storage.

IMPROVED DATA SHARING A database is designed as a shared corporate resource. Authorized internal and external users are granted permission to use the database, and each user (or group of users) is provided one or more user views into the database to facilitate this use. A **user view** is a logical description of some portion of the database that is required by a user to perform some task. A user view is often developed by identifying a form or report that the user needs on a regular basis. For example, an employee working in human resources will need access to confidential employee data; a customer needs access to the product catalog available on Pine Valley's Web site. The views for the human resources employee and the customer are drawn from completely different areas of one unified database.

User view
A logical description of some portion of the database that is required by a user to perform some task.

INCREASED PRODUCTIVITY OF APPLICATION DEVELOPMENT A major advantage of the database approach is that it greatly reduces the cost and time for developing new business applications. There are three important reasons that database applications can often be developed much more rapidly than conventional file applications:

1. Assuming that the database and the related data capture and maintenance applications have already been designed and implemented, the application developer can concentrate on the specific functions required for the new application without having to worry about file design or low-level implementation details.
2. The database management system provides a number of high-level productivity tools, such as forms and report generators, and high-level languages that automate

some of the activities of database design and implementation. You will learn about many of these tools in subsequent chapters.

3. Significant improvement in application developer productivity, estimated to be as high as 60 percent (Long, 2005), is currently being realized through the use of Web services based on the use of standard Internet protocols and a universally accepted data format (XML).

ENFORCEMENT OF STANDARDS When the database approach is implemented with full management support, the database administration function should be granted single-point authority and responsibility for establishing and enforcing data standards. These standards will include naming conventions, data quality standards, and uniform procedures for accessing, updating, and protecting data. The data repository provides database administrators with a powerful set of tools for developing and enforcing these standards. Unfortunately, the failure to implement a strong database administration function is perhaps the most common source of database failures in organizations. You will learn about the database administration (and related data administration) functions in Chapter 12.

IMPROVED DATA QUALITY Concern with poor quality data is a common theme in strategic planning and database administration today. In 2011 alone, poor data quality is estimated to have cost the U.S. economy almost $3 trillion, almost twice the size of the federal deficit (**http://hollistibbetts.sys-con.com/node/1975126**). The database approach provides a number of tools and processes to improve data quality. Two of the more important are the following:

1. Database designers can specify integrity constraints that are enforced by the DBMS. A **constraint** is a rule that cannot be violated by database users. We describe numerous types of constraints (also called "business rules") in Chapters 2 and 3. If a customer places an order, the constraint that ensures that the customer and the order remain associated is called a "relational integrity constraint," and it prevents an order from being entered without specifying who placed the order.
2. One of the objectives of a data warehouse environment is to clean up (or "scrub") operational data before they are placed in the data warehouse (Jordan, 1996). Do you ever receive multiple copies of a catalog? The company that sends you three copies of each of its mailings could recognize significant postage and printing savings if its data were scrubbed, and its understanding of its customers would also be enhanced if it could determine a more accurate count of existing customers. You will learn about data warehouses and data integration in Chapter 9 and the potential for improving data quality in Chapter 12.

Constraint
A rule that cannot be violated by database users.

IMPROVED DATA ACCESSIBILITY AND RESPONSIVENESS With a relational database, end users without programming experience can often retrieve and display data, even when they cross traditional departmental boundaries. For example, an employee can display information about computer desks at Pine Valley Furniture Company with the following query:

```
SELECT *
FROM Product_T
WHERE ProductDescription = "Computer Desk";
```

The language used in this query is called Structured Query Language, or SQL. (You will study this language in detail in Chapters 5 and 6.) Although the queries constructed can be *much* more complex, the basic structure of the query is easy for even novice, nonprogrammers to grasp. If they understand the structure and names of the data that fit within their view of the database, they soon gain the ability to retrieve answers to new questions without having to rely on a professional application developer. This can be dangerous; queries should be thoroughly tested to be sure they are returning accurate data before relying on their results, and novices may not understand that challenge.

REDUCED PROGRAM MAINTENANCE Stored data must be changed frequently for a variety of reasons: New data item types are added, data formats are changed, and so forth. A celebrated example of this problem was the well-known "year 2000" problem, in which common two-digit year fields were extended to four digits to accommodate the rollover from the year 1999 to the year 2000.

In a file processing environment, the data descriptions and the logic for accessing data are built into individual application programs (this is the program-data dependence issue described earlier). As a result, changes to data formats and access methods inevitably result in the need to modify application programs. In a database environment, data are more independent of the application programs that use them. Within limits, you can change either the data or the application programs that use the data without necessitating a change in the other factor. As a result, program maintenance can be significantly reduced in a modern database environment.

IMPROVED DECISION SUPPORT Some databases are designed expressly for decision support applications. For example, some databases are designed to support customer relationship management, whereas others are designed to support financial analysis or supply chain management. You will study how databases are tailored for different decision support applications and analytical styles in Chapters 9 through 11.

Cautions about Database Benefits

The previous section identified 10 major potential benefits of the database approach. However, we must caution you that many organizations have been frustrated in attempting to realize some of these benefits. For example, the goal of data independence (and, therefore, reduced program maintenance) has proven elusive due to the limitations of older data models and database management software. Fortunately, the relational model and the newer object-oriented model provide a significantly better environment for achieving these benefits. Another reason for failure to achieve the intended benefits is poor organizational planning and database implementation; even the best data management software cannot overcome such deficiencies. For this reason, you will learn about the importance of database planning and design throughout this text.

Costs and Risks of the Database Approach

A database is not a silver bullet, and it does not have the magic power of Harry Potter. As with any other business decision, the database approach entails some additional costs and risks that must be recognized and managed when it is implemented (see Table 1-4).

NEW, SPECIALIZED PERSONNEL Frequently, organizations that adopt the database approach need to hire or train individuals to design and implement databases, provide database administration services, and manage a staff of new people. Further, because of the rapid changes in technology, these new people will have to be retrained or upgraded on a regular basis. This personnel increase may be more than offset by other productivity gains, but an organization should recognize the need for these specialized skills, which are required to obtain the most from the potential benefits. You will learn about the staff requirements for database management in Chapter 12.

TABLE 1-4 Costs and Risks of the Database Approach

New, specialized personnel
Installation and management cost and complexity
Conversion costs
Need for explicit backup and recovery
Organizational conflict

INSTALLATION AND MANAGEMENT COST AND COMPLEXITY A multi-user database management system is a large and complex suite of software that has a high initial cost, requires a staff of trained personnel to install and operate, and has substantial annual maintenance and support costs. Installing such a system may also require upgrades to the hardware and data communications systems in the organization. Substantial training is normally required on an ongoing basis to keep up with new releases and upgrades. Additional or more sophisticated and costly database software may be needed to provide security and to ensure proper concurrent updating of shared data.

CONVERSION COSTS The term *legacy system* is widely used to refer to older applications in an organization that are based on file processing and/or older database technology. The cost of converting these older systems to modern database technology—measured in terms of dollars, time, and organizational commitment—may often seem prohibitive to an organization. The use of data warehouses is one strategy for continuing to use older systems while at the same time exploiting modern database technology and techniques (Ritter, 1999).

NEED FOR EXPLICIT BACKUP AND RECOVERY A shared corporate database must be accurate and available at all times. This requires that comprehensive procedures be developed and used for providing backup copies of data and for restoring a database when damage occurs. These considerations have acquired increased urgency in today's security-conscious environment. A modern database management system normally automates many more of the backup and recovery tasks than a file system. You will learn about procedures for security, backup, and recovery in Chapter 8.

ORGANIZATIONAL CONFLICT A shared database requires a consensus on data definitions and ownership as well as responsibilities for accurate data maintenance. Experience has shown that conflicts on data definitions, data formats and coding, rights to update shared data, and associated issues are frequent and often difficult to resolve. Handling these issues requires organizational commitment to the database approach, organizationally astute database administrators, and a sound evolutionary approach to database development.

If strong top management support of and commitment to the database approach are lacking, end-user development of stand-alone databases is likely to proliferate. These databases do not follow the general database approach that we have described, and they are unlikely to provide the benefits described earlier. In the extreme, they may lead to a pattern of inferior decision making that threatens the well-being or existence of an organization.

INTEGRATED DATA MANAGEMENT FRAMEWORK

The database approach described above is associated with relational database technologies and used primarily as a foundation for the design and implementation of transaction processing systems (*operational systems*). Data management technologies are also increasingly often used as *informational systems*, as a foundation for analytics, or the systematic analysis and interpretation of data to improve our understanding of a real-world domain. Transactional systems are still the core of this book with a focus on relational databases and the SQL language. These technologies continue, in practice, to be a fundamental source of data for all areas of data management, and no other technology is as widely used. They form the foundation on which business activities of modern organizations are built because they enable the way in which organizations interact and do business with their stakeholders.

This book does, however, also cover data management technologies intended primarily for enabling and supporting analytics. They can be divided into two major categories: data warehousing and big data. Data warehousing has existed as a concept since late 1980s, and, as you will learn in Chapter 9, both conceptual approaches and implementation technologies for data warehousing are already well developed and

FIGURE 1-5 Integrated data management framework

	Operational	Informational	
	Transactional	Analytical– Data Warehousing	Analytical– Big Data
Technology	Relational	Relational	Non-relational
Modeling	Conceptual data modeling with (E)ER (Chapters 2 and 3)	Data warehousing and data integration (Chapter 9)	Big data technologies, including Hadoop & NoSQL (Chapter 10)
Design	Logical data modeling with the relational model; Normalization (Chapter 4)		
Infrastructure	Physical design of relational databases; Security; Cloud computing (Chapter 8)		
Access	SQL (Chapters 5 and 6) Applications with SQL (Chapter 7)		
Data analysis	Analytics and its implications (Chapter 11)		
Governance and data management	Lifecycle (Chapter 1) Governance, data quality, and master data management (Chapter 12)		

mature. Indeed, most traditional data warehouses are implemented using the same relational technologies as transactional systems. Big data technologies have emerged as another category of informational systems since the early 2010s. They are characterized by their ability to deal with large *volumes* of data with a *variety* of data types arriving to the organizational systems with high *velocity* (the so-called three Vs of big data). A major difference between big data systems compared to both data warehousing and operational, transaction-focused systems is that structures of the latter are typically expected to be carefully designed before data are stored in them ("schema on write"), whereas many of the big data analytics technologies are intended to be used in the "schema on read" mode. In the latter approach, the structure of the data and the relationships between the data elements will be determined later, either right before or at the time of the use of the data.

Figure 1-5 presents a framework that illustrates the structure and the contents of this book based three key categories: Transactional (an operational category), Analytical–Data Warehousing, and Analytical–Big Data (informational categories). As the framework demonstrates, the book explores data management in the operational context at a more detailed level than the informational one, dedicating Chapters 2 through 8 to transactional systems and separating the coverage of modeling, design, access, and infrastructure into different chapters. The framework also shows the following:

- This book recognizes the growing importance of informational uses of data management processes and technologies and presents them in the same broader context with operational uses. You will learn about the informational use of data management from two different analytical perspectives: data warehousing (Chapter 9) and big data (Chapter 10). Both of these chapters deal with questions regarding modeling, design, access, and infrastructure in an integrated way.
- In many areas, such as analytics (Chapter 11), and data management, governance, and quality (Chapter 12), the concerns and key questions are shared between the operational and informational perspectives.

COMPONENTS OF THE DATABASE ENVIRONMENT

Now that you have seen the advantages and risks of using the database approach to managing data, let us examine the major components of a typical database environment and their relationships (see Figure 1-6). You have already been introduced to some (but not all) of these components in previous sections. Following is a brief description of the nine components shown in Figure 1-6:

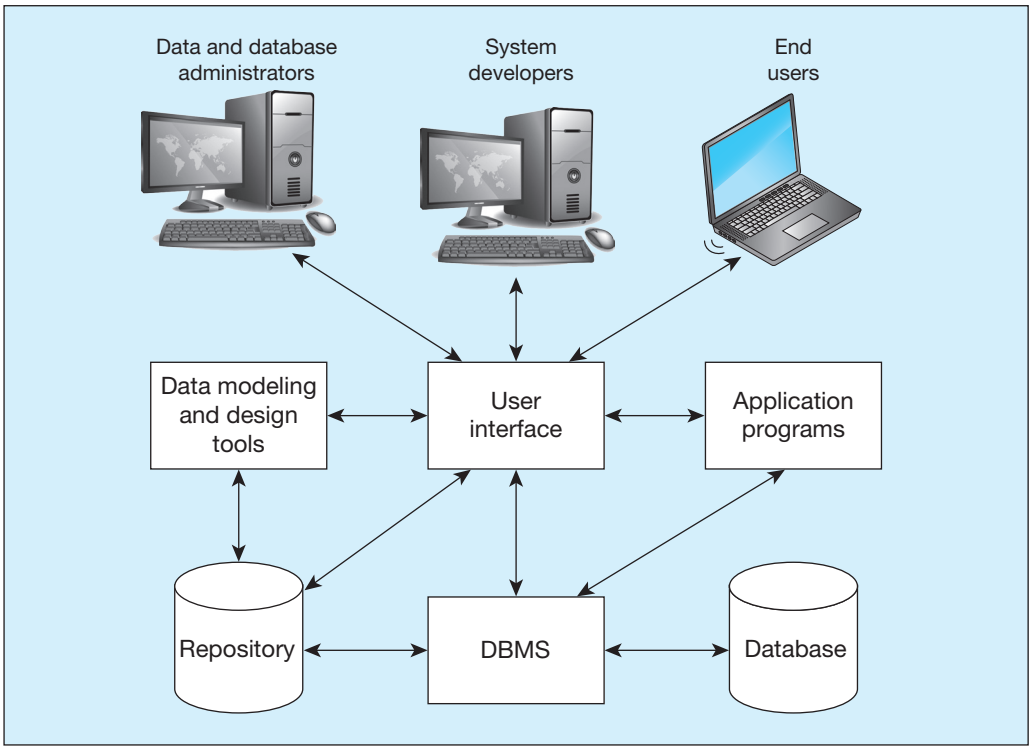


FIGURE 1-6 Components of the database environment

1. ***Data modeling and design tools*** **Data modeling and design tools** are automated tools used to design databases and application programs. These tools help with creation of data models and in some cases can also help automatically generate the "code" needed to create the database. You will learn more about the use of automated tools for database design and development throughout the text, particularly in Chapters 4 and 8.
2. ***Repository*** A **repository** is a centralized knowledge base for all data definitions, data relationships, screen and report formats, and other system components. A repository contains an extended set of metadata important for managing databases as well as other components of an information system. We describe the repository in Chapter 9.
3. ***DBMS*** A DBMS is a software system that is used to create, maintain, and provide controlled access to user databases. You will learn about many of the technical functions of a DBMS in Chapter 8.
4. ***Database*** A database is an organized collection of logically related data, usually designed to meet the information needs of multiple users in an organization. It is important to distinguish between the database and the repository. The repository contains definitions of data, whereas the database contains occurrences of data. You will explore the activities of database design and implementation in Chapters 4 through 8.
5. ***Application programs*** Computer-based application programs are used to create and maintain the database and provide information to users. Key database-related application programming skills are described in Chapters 5 through 10.
6. ***User interface*** The user interface includes languages, menus, and other facilities by which users interact with various system components, such as data modeling and design tools, application programs, the DBMS, and the repository. User interfaces are illustrated throughout this text, with a particular focus in Chapters 5, 6, 8, and 11.
7. ***Data and database administrators*** Data administrators are persons who are responsible for the overall management of data resources in an organization. Database administrators are responsible for physical database design and for managing technical issues in the database environment. You will learn about these functions in detail in Chapters 8 and 12.

Data modeling and design tools
Software tools that provide automated support for creating data models.

Repository
A centralized knowledge base of all data definitions, data relationships, screen and report formats, and other system components.

8. ***System developers*** System developers are persons, such as systems analysts and programmers, who design new application programs. The content of all chapters of this book is useful for system developers, but Chapters 2 through 8 are likely to be particularly valuable because of their focus on transactional systems.
9. ***End users*** End users are persons throughout the organization who add, delete, and modify data in the database and who request or receive information from it. All user interactions with the database must be routed through the DBMS. This text is targeted primarily to students striving to become data management and systems development professionals, but advanced end users can benefit from many of the skills covered in it (particularly conceptual data modeling in Chapters 2 to 3, SQL in Chapters 5 to 6, and Analytics in Chapter 11).

In summary, the database operational environment shown in Figure 1-6 is an integrated system of hardware, software, and people, designed to facilitate the storage, retrieval, and control of the information resource and to improve the productivity of the organization.

THE DATABASE DEVELOPMENT PROCESS

Enterprise data modeling
The first step in database development, in which the scope and general contents of organizational databases are specified.

How do organizations start developing a database? In many organizations, database development begins with **enterprise data modeling**, which establishes the range and general contents of organizational databases. Its purpose is to create an overall picture or explanation of organizational data, not the design for a particular database. A particular database provides the data for one or more information systems, whereas an enterprise data model, which may encompass many databases, describes the scope of data maintained by the organization. In enterprise data modeling, you review current systems, analyze the nature of the business areas to be supported, describe the data needed at a very high level of abstraction, and plan one or more database development projects.

Figure 1-3a showed a segment of an enterprise data model for Pine Valley Furniture Company, using a simplified version of the notation you will learn in Chapters 2 and 3. Besides such a graphical depiction of the entity types, a thorough enterprise data model would also include business-oriented descriptions of each entity type and a compendium of various statements about how the business operates, called *business rules*, which govern the validity of data. Relationships between business objects (business functions, units, applications, and so forth) and data are often captured using matrixes and complement the information captured in the enterprise data model. Figure 1-7 shows an example of such a matrix.

FIGURE 1-7 Example business function-to-data entity matrix

Business Functions \ Data Entity Types	Customer	Product	Raw Material	Order	Work Center	Work Order	Invoice	Equipment	Employee
Business Planning	X	X						X	X
Product Development		X	X		X			X	
Materials Management		X	X	X	X	X		X	
Order Fulfillment	X	X	X	X	X	X	X	X	X
Order Shipment	X	X		X	X		X		X
Sales Summarization	X	X		X			X		X
Production Operations		X	X	X	X	X		X	X
Finance and Accounting	X	X	X	X	X		X	X	X

X = data entity is used within business function

Enterprise data modeling as a component of a top-down approach to information systems planning and development represents one source of database projects. Such projects often develop new databases to meet strategic organizational goals, such as improved customer support, better production and inventory management, or more accurate sales forecasting. Many database projects arise, however, in a more bottom-up fashion. In this case, projects are requested by information systems users who need certain information to do their jobs or by other information systems professionals who see a need to improve data management in the organization.

A typical bottom-up database development project usually focuses on the creation of one database. Some database projects concentrate only on defining, designing, and implementing a database as a foundation for subsequent information systems development. In most cases, however, a database and the associated information processing functions are developed together as part of a comprehensive information systems development project.

Systems Development Life Cycle

As you may know from other information systems courses you've taken, a traditional process for conducting an information systems development project is called the **systems development life cycle (SDLC)**. The SDLC is a complete set of steps that a team of information systems professionals, including database designers and programmers, follow in an organization to specify, develop, maintain, and replace information systems. Textbooks and organizations use many variations on the life cycle and may identify anywhere from 3 to 20 different phases.

Systems development life cycle (SDLC)
The traditional methodology used to develop, maintain, and replace information systems.

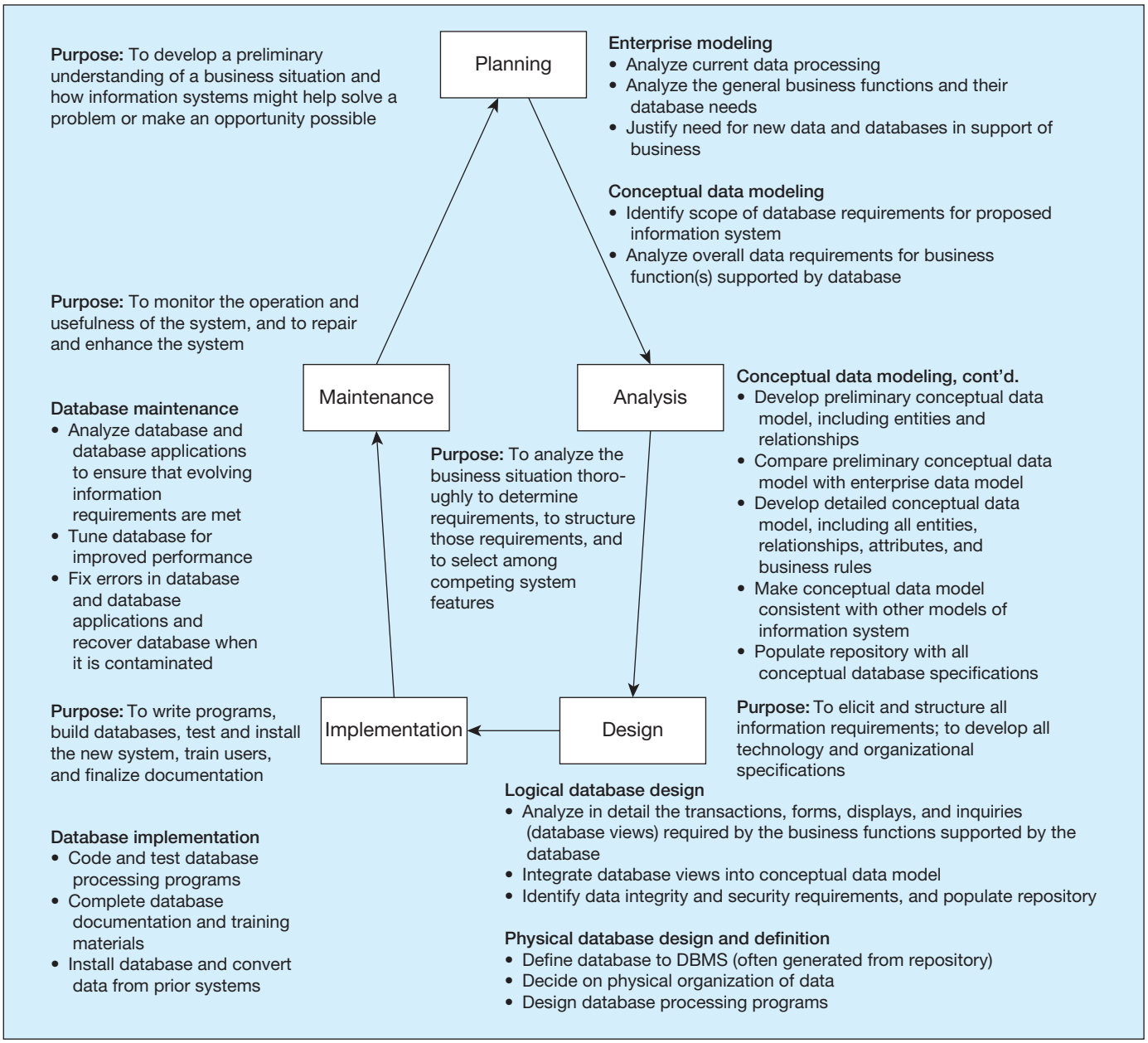
The various steps in the SDLC and their associated purpose are depicted in Figure 1-8 (Valacich and George, 2016). The process appears to be circular and is intended to convey the iterative nature of systems development projects. The steps may overlap in time, they may be conducted in parallel, and it is possible to backtrack to previous steps when prior decisions need to be reconsidered. Some believe that the most common path through the development process is to cycle through the steps depicted in Figure 1-8 but at more detailed levels on each pass as the requirements of the system become more concrete.

Figure 1-8 also provides an outline of the database development activities typically included in each phase of the SDLC. Note that there is not always a one-to-one correspondence between SDLC phases and database development steps. For example, conceptual data modeling occurs in both the Planning and the Analysis phase. We will briefly illustrate each of these database development steps for Pine Valley Furniture Company later in this chapter.

PLANNING—ENTERPRISE MODELING The database development process begins with a review of the enterprise modeling components that were developed during the information systems planning process. During this step, analysts review current databases and information systems, analyze the nature of the business area that is the subject of the development project, and describe, in general terms, the data needed for each information system under consideration for development. They determine what data are already available in existing databases and what new data will need to be added to support the proposed new project. Only selected projects move into the next phase based on the projected value of each project to the organization.

PLANNING—CONCEPTUAL DATA MODELING For an information systems project that is initiated, the overall data requirements of the proposed information system must be analyzed. This is done in two stages. First, during the Planning phase, the analyst develops a diagram similar to Figure 1-3a, as well as other documentation, to outline the scope of data involved in this particular development project without consideration of what databases already exist. Only high-level categories of data (entities) and major relationships are included at this point. This step in the SDLC is critical for improving the chances of a successful development process. The better the definition of the specific needs of the organization, the closer the conceptual model should come

FIGURE 1-8 Database development activities during the systems development life cycle (SDLC)



to meeting the needs of the organization and the less recycling back through the SDLC should be needed.

ANALYSIS—CONCEPTUAL DATA MODELING During the Analysis phase of the SDLC, the analyst produces a detailed data model that identifies all the organizational data that must be managed for this information system. Every data attribute is defined, all categories of data are listed, every business relationship between data entities is represented, and every rule that dictates the integrity of the data is specified. It is also during the Analysis phase that the conceptual data model is checked for consistency with other types of models developed to explain other dimensions of the target information system, such as processing steps, rules for handling data, and the timing of events. However, even this detailed conceptual data model is preliminary because subsequent SDLC activities may find missing elements or errors when designing specific transactions, reports, displays, and inquiries. With experience, the database developer gains mental models of common business functions, such as sales or financial record keeping,

but must always remain alert for the exceptions to common practices followed by an organization. The output of the conceptual modeling phase is a **conceptual schema**.

Conceptual schema
A detailed, technology-independent specification of the overall structure of organizational data.

DESIGN—LOGICAL DATABASE DESIGN Logical database design approaches database development from two perspectives. First, the conceptual schema must be transformed into a logical schema, which describes the data in terms of the data management technology that will be used to implement the database. For example, if relational technology will be used, the conceptual data model is transformed and represented using elements of the relational model, which include tables, columns, rows, primary keys, foreign keys, and constraints. (You will learn how to conduct this important process in Chapter 4.) This representation is referred to as the **logical schema**.

Logical schema
The representation of a database for a particular data management technology.

Then, as each application in the information system is designed, including the program's input and output formats, the analyst performs a detailed review of the transactions, reports, displays, and inquiries supported by the database. During this so-called bottom-up analysis, the analyst verifies exactly what data are to be maintained in the database and the nature of those data as needed for each transaction, report, and so forth. It may be necessary to refine the conceptual data model as each report, business transaction, and other user view is analyzed. In this case, one must combine, or integrate, the original conceptual data model along with these individual user views into a comprehensive design during logical database design. It is also possible that additional information processing requirements will be identified during logical information systems design, in which case these new requirements must be integrated into the previously identified logical database design.

The final step in logical database design is to transform the combined and reconciled data specifications into basic, or atomic, elements following well-established rules for well-structured data specifications. For most databases today, these rules come from relational database theory and a process called *normalization*, which you will learn about in detail in Chapter 4. The result is a complete picture of the database without any reference to a particular database management system for managing these data. With a final logical database design in place, the analyst begins to specify the logic of the particular computer programs and queries needed to maintain and report the database contents.

Physical schema
Specifications for how data from a logical schema are stored in a computer's secondary memory by a database management system.

DESIGN—PHYSICAL DATABASE DESIGN AND DEFINITION A **physical schema** is a set of specifications that describe how data from a logical schema are stored in a computer's secondary memory by a specific database management system. There is one physical schema for each logical schema. Physical database design requires knowledge of the specific DBMS that will be used to implement the database. In physical database design and definition, an analyst decides on the organization of physical records, the choice of file organizations, the use of indexes, and so forth. To do this, a database designer needs to outline the programs to process transactions and to generate anticipated management information and decision support reports. The goal is to design a database that will efficiently and securely handle all data processing against it. Thus, physical database design is done in close coordination with the design of all other aspects of the physical information system: programs, computer hardware, operating systems, and data communications networks.

IMPLEMENTATION—DATABASE IMPLEMENTATION In database implementation, a designer writes, tests, and installs the programs/scripts that access, create, or modify the database. The designer might do this using standard programming languages (e.g., Java, C#, or Visual Basic.NET) or in special database processing languages (e.g., SQL) or use special-purpose nonprocedural languages to produce stylized reports and displays, possibly including graphs. Also, during implementation, the designer will finalize all database documentation, train users, and put procedures into place for the ongoing support of the information system (and database) users. The last step is to load data from existing information sources (files and databases from legacy applications plus new data now needed). Loading is often done by first unloading data from existing files and databases into a neutral format (such as binary or text files) and then loading these data into the new database. Finally, the database and its associated applications

are put into production for data maintenance and retrieval by the actual users. During production, the database should be periodically backed up and recovered in case of contamination or destruction.

MAINTENANCE—DATABASE MAINTENANCE The database evolves during database maintenance. In this step, the designer adds, deletes, or changes characteristics of the structure of a database in order to meet changing business conditions, to correct errors in database design, or to improve the processing speed of database applications. The designer might also need to rebuild a database if it becomes contaminated or destroyed due to a program or computer system malfunction. This is typically the longest step of database development because it lasts throughout the life of the database and its associated applications. Each time the database evolves, view it as an abbreviated database development process in which conceptual data modeling, logical and physical database design, and database implementation occur to deal with proposed changes.

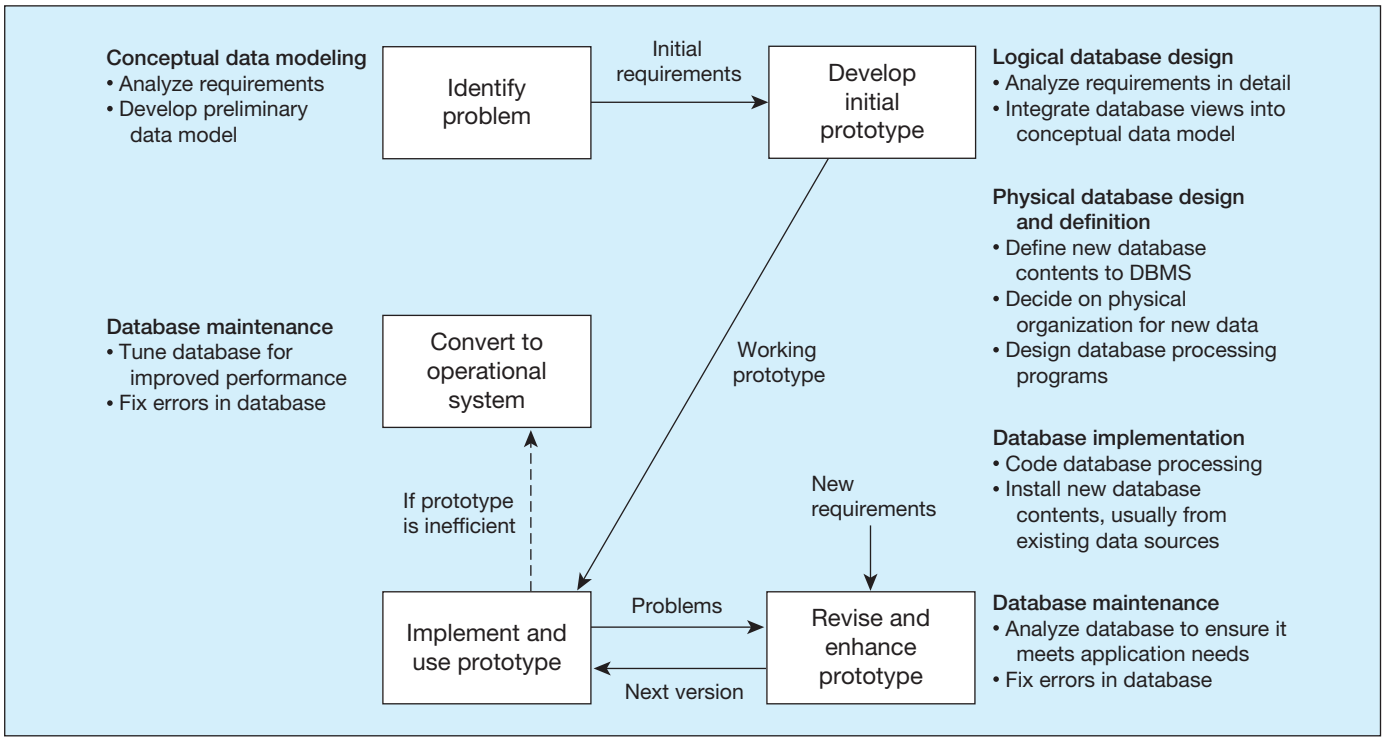
Alternative Information Systems Development Approaches

The systems development life cycle or slight variations on it are often used to guide the development of information systems and databases. The SDLC is a methodical, highly structured approach that includes many checks and balances to ensure that each step produces accurate results and that the new or replacement information system is consistent with existing systems with which it must communicate or for which there needs to be consistent data definitions. Whew! That's a lot of work! Consequently, the SDLC is often criticized for the length of time needed until a working system is produced, which occurs only at the end of the process. Instead, organizations increasingly use rapid application development (RAD) methods, which follow an iterative process of rapidly repeating analysis, design, and implementation steps until they converge on the system the user wants. These RAD methods work best when most of the necessary database structures already exist and hence for systems that primarily retrieve data rather than for those that populate and revise databases.

Prototyping
An iterative process of systems development in which requirements are converted to a working system that is continually revised through close work between analysts and users.

One of the most popular RAD methods is **prototyping**, which is an iterative process of systems development in which requirements are converted to a working system that is continually revised through close work between analysts and users. Figure 1-9 shows the

FIGURE 1-9 The prototyping methodology and database development process



prototyping process. This figure includes annotations to indicate roughly which database development activities occur in each prototyping phase. Typically, you make only a very cursory attempt at conceptual data modeling when the information system problem is identified. During the development of the initial prototype, you simultaneously design the displays and reports the user wants while understanding any new database requirements and defining a database to be used by the prototype. This is typically a new database, which is a copy of portions of existing databases, possibly with new content. If new content is required, it will usually come from external data sources, such as market research data, general economic indicators, or industry standards.

Database implementation and maintenance activities are repeated as new versions of the prototype are produced. Often, security and integrity controls are minimal because the emphasis is on getting working prototype versions ready as quickly as possible. Also, documentation tends to be delayed until the end of the project, and user training occurs from hands-on use. Finally, after an accepted prototype is created, the developer and the user decide whether the final prototype and its database can be put into production as is. If the system, including the database, is too inefficient, the system and database might need to be reprogrammed and reorganized to meet performance expectations. Inefficiencies, however, have to be weighed against violating the core principles behind sound database design.

With the increasing popularity of visual programming tools (such as Visual Basic, Java, or C#) that make it easy to modify the interface between user and system, prototyping is becoming the systems development methodology of choice to develop new applications internally. With prototyping, it is relatively easy to change the content and layout of user reports and displays.

The benefits from iterative approaches to systems development demonstrated by RAD and prototyping approaches have resulted in further efforts to create ever more responsive development approaches. In February 2001, a group of 17 individuals interested in supporting these approaches created "The Manifesto for Agile Software Development." For them, **agile software development** practices include valuing (www.agilemanifesto.org) the following:

Individuals and interactions over processes and tools

Working software over comprehensive documentation

Customer collaboration over contract negotiation

Responding to change over following a plan

Agile software development

An approach to database and software development that emphasizes "*individuals and interactions* over processes and tools, *working software* over comprehensive documentation, *customer collaboration* over contract negotiation, and *response to change* over following a plan."

Emphasis on the importance of people, both software developers and customers, is evident in their phrasing. This is in response to the turbulent environment within which software development occurs as compared to the more staid environment of most engineering development projects from which the earlier software development methodologies came. The importance of the practices established in the SDLC continues to be recognized and accepted by software developers, including the creators of "The Manifesto for Agile Software Development." However, it is impractical to allow these practices to stifle quick reactions to changes in the environment that change project requirements.

The use of agile or adaptive processes should be considered when a project involves unpredictable and/or changing requirements, responsible and collaborative developers, and involved customers who understand and can contribute to the process (Fowler, 2005). If you are interested in learning more about agile software development, investigate agile methodologies, such as eXtreme Programming, Scrum, the DSDM Consortium, and feature-driven development.

Three-Schema Architecture for Database Development

The explanation earlier in this chapter of the database development process referred to several different but related models of databases developed on a systems development project. These data models and the primary phase of the SDLC in which they are developed are summarized here:

- Enterprise data model (during the Information Systems Planning phase).
- External schema or user view (during the Analysis and Logical Design phases).