

Ryan Stephens

**SEVENTH
EDITION**

New sample
database + standard
SQL examples:
Use these skills
anywhere

Sams **Teach Yourself**

SQL

in **24**
Hours

 Pearson

FREE SAMPLE CHAPTER

SHARE WITH OTHERS



Ryan Stephens

Sams **Teach Yourself**
SQL

Seventh Edition

in **24**
Hours

SAMS

Sams Teach Yourself SQL in 24 Hours, Seventh Edition

Copyright © 2022 by Pearson Education, Inc.

All rights reserved. Printed in the United States of America. This publication is protected by copyright, and permission must be obtained from the publisher prior to any prohibited reproduction, storage in a retrieval system, or transmission in any form or by any means, electronic, mechanical, photocopying, recording, or likewise. For information regarding permissions, request forms, and the appropriate contacts within the Pearson Education Global Rights & Permissions Department, please visit www.pearson.com/permissions. No patent liability is assumed with respect to the use of the information contained herein. Although every precaution has been taken in the preparation of this book, the publisher and author assume no responsibility for errors or omissions. Nor is any liability assumed for damages resulting from the use of the information contained herein.

ISBN-13: 978-0-13-754312-0

ISBN-10: 0-13-754312-3

Library of Congress Control Number: 2021948117

ScoutAutomatedPrintCode

Trademarks

All terms mentioned in this book that are known to be trademarks or service marks have been appropriately capitalized. Sams Publishing cannot attest to the accuracy of this information. Use of a term in this book should not be regarded as affecting the validity of any trademark or service mark.

Warning and Disclaimer

Every effort has been made to make this book as complete and as accurate as possible, but no warranty or fitness is implied. The information provided is on an “as is” basis. The author and the publisher shall have neither liability nor responsibility to any person or entity with respect to any loss or damages arising from the information contained in this book.

Special Sales

For information about buying this title in bulk quantities, or for special sales opportunities (which may include electronic versions; custom cover designs; and content particular to your business, training goals, marketing focus, or branding interests), please contact our corporate sales department at corpsales@pearsoned.com or (800) 382-3419.

For government sales inquiries, please contact governmentsales@pearsoned.com.

For questions about sales outside the U.S., please contact intlcs@pearson.com.

Editor-in-Chief

Mark Taub

Acquisitions Editor

Malobika

Chakraborty

Development Editor

Mark Renfrow

Managing Editor

Sandra Schroeder

Senior Project Editor

Tonya Simpson

Copy Editor

Krista Hansing

Editorial Services

Indexer

Cheryl Lenser

Proofreader

Betty Pessagno

Cover Designer

Chuti Prasertsith

Compositor

codemantra

Pearson's Commitment to Diversity, Equity, and Inclusion

Pearson is dedicated to creating bias-free content that reflects the diversity of all learners. We embrace the many dimensions of diversity, including but not limited to race, ethnicity, gender, socioeconomic status, ability, age, sexual orientation, and religious or political beliefs.

Education is a powerful force for equity and change in our world. It has the potential to deliver opportunities that improve lives and enable economic mobility. As we work with authors to create content for every product and service, we acknowledge our responsibility to demonstrate inclusivity and incorporate diverse scholarship so that everyone can achieve their potential through learning. As the world's leading learning company, we have a duty to help drive change and live up to our purpose to help more people create a better life for themselves and to create a better world.

Our ambition is to purposefully contribute to a world where:

- Everyone has an equitable and lifelong opportunity to succeed through learning.
- Our educational products and services are inclusive and represent the rich diversity of learners.
- Our educational content accurately reflects the histories and experiences of the learners we serve.
- Our educational content prompts deeper discussions with learners and motivates them to expand their own learning (and worldview).

While we work hard to present unbiased content, we want to hear from you about any concerns or needs with this Pearson product so that we can investigate and address them.

- Please contact us with concerns about any potential bias at <https://www.pearson.com/report-bias.html>.

Contents at a Glance

HOUR 1	Understanding the Relational Database and SQL	1
HOUR 2	Exploring the Components of the SQL Language	17
HOUR 3	Getting to Know Your Data	29
HOUR 4	Setting Up Your Database	39
HOUR 5	Understanding the Basics of Relational (SQL) Database Design	61
HOUR 6	Defining Entities and Relationships	77
HOUR 7	Normalizing Your Database	93
HOUR 8	Defining Data Structures	113
HOUR 9	Creating and Managing Database Objects	129
HOUR 10	Manipulating Data	153
HOUR 11	Managing Database Transactions	167
HOUR 12	Introduction to Database Queries	187
HOUR 13	Using Operators to Categorize Data	209
HOUR 14	Joining Tables in Queries	237
HOUR 15	Restructuring the Appearance of Data	259
HOUR 16	Understanding Dates and Times	279
HOUR 17	Summarizing Data Results from a Query	299
HOUR 18	Using Subqueries to Define Unknown Data	321
HOUR 19	Combining Multiple Queries into One	337
HOUR 20	Creating and Using Views and Synonyms	359
HOUR 21	Managing Database Users and Security	385
HOUR 22	Using Indexes to Improve Performance	413
HOUR 23	Improving Database Performance	427
HOUR 24	Working with the System Catalog	445
HOUR 25	Bonus Workshop for the Road	459

Appendixes

A	Common SQL Commands	499
B	Popular Vendor RDBMS Implementations	505
C	Answers to Quizzes and Exercises	507

Index	577
-------------	-----

Table of Contents

HOURL 1: Understanding the Relational Database and SQL	1
Thriving in a Data-Driven World	2
Understanding the Relational Database	7
The Relational Database Continues to Lead the Way	13
Examples and Exercises	13
Summary	13
Q&A	15
Workshop	15
HOURL 2: Exploring the Components of the SQL Language	17
SQL Definition and History	17
SQL: The Standard Language	18
SQL Sessions	21
Types of SQL Commands	22
Summary	25
Q&A	26
Workshop	26
HOURL 3: Getting to Know Your Data	29
The BIRD Database: Examples and Exercises in This Book	29
How to Talk About the Data	30
Entity Relationship Diagrams	32
Examples and Exercises	36
Summary	36
Q&A	37
Workshop	37
HOURL 4: Setting Up Your Database	39
Locating the Files You Need	39
Getting Set Up for Hands-on Exercises	40
List of Data by Table	49

Summary	57
Q&A	58
Workshop	58
HOUR 5: Understanding the Basics of Relational (SQL) Database Design	61
Understanding What Database Design Has to Do with SQL	61
The Database Design Process	62
Choosing a Database Design Methodology	63
Using a Simple Process to Think Through the Design of the BIRDS Database	64
Logical Model vs. Physical Design	71
Database Life Cycle	72
Summary	74
Q&A	75
Workshop	75
HOUR 6: Defining Entities and Relationships	77
Creating a Data Model Based on Your Data	77
Defining Relationships	81
Employing Referential Integrity	84
Creating an Entity Relationship	87
Summary	89
Q&A	90
Workshop	90
HOUR 7: Normalizing Your Database	93
Defining Normalization	94
Exploring the Most Common Normal Forms of the Normalization Process	96
Denormalizing a Database	106
Applying Normalization to Your Database	107
Summary	109
Q&A	111
Workshop	111

HOOR 8: Defining Data Structures	113
Defining Data	114
Understanding Basic Data Types	114
Using Data Types in the BIRDS Database	123
Summary	124
Q&A	126
Workshop	126
HOOR 9: Creating and Managing Database Objects	129
Database Objects and Schemas	129
Tables: The Primary Storage for Data	131
Integrity Constraints	146
Summary	150
Q&A	151
Workshop	151
HOOR 10: Manipulating Data	153
Getting an Overview of Data Manipulation	153
Populating Tables with New Data	154
Updating Existing Data	158
Deleting Data from Tables	162
Summary	164
Q&A	165
Workshop	165
HOOR 11: Managing Database Transactions	167
Defining Transactions	167
Controlling Transactions	168
Dealing with Poor Transactional Control	181
Summary	182
Q&A	183
Workshop	183
HOOR 12: Introduction to Database Queries	187
Using the SELECT Statement	187
Case Sensitivity	197

Fundamentals of Query Writing	199
Summary	206
Q&A	207
Workshop	207
HOUR 13: Using Operators to Categorize Data	209
Defining an Operator in SQL	209
Using Comparison Operators	210
Using Logical Operators	214
Using Conjunctive Operators	221
Using Negative Operators	225
Using Arithmetic Operators	228
Summary	233
Q&A	234
Workshop	234
HOUR 14: Joining Tables in Queries	237
Selecting Data from Multiple Tables	237
Understanding Joins	238
Join Considerations	250
Summary	255
Q&A	256
Workshop	256
HOUR 15: Restructuring the Appearance of Data	259
ANSI Character Functions	259
Common Character Functions	260
Miscellaneous Character Functions	268
Mathematical Functions	272
Conversion Functions	272
Combined Character Functions	275
Summary	275
Q&A	276
Workshop	276

HOUR 16: Understanding Dates and Times	279
Understanding How a Date Is Stored	279
Using Date Functions	281
Converting Dates	290
Summary	295
Q&A	296
Workshop	296
HOUR 17: Summarizing Data Results from a Query	299
Using Aggregate Functions	299
Grouping Data	306
Using the GROUP BY Clause	306
Understanding the Difference Between GROUP BY and ORDER BY	309
Using CUBE and ROLLUP Expressions	312
Using the HAVING Clause	315
Summary	316
Q&A	317
Workshop	317
HOUR 18: Using Subqueries to Define Unknown Data	321
Defining Subqueries	321
Embedded Subqueries	330
Using Correlated Subqueries	333
Summary	333
Q&A	335
Workshop	335
HOUR 19: Combining Multiple Queries into One	337
Differentiating Single Queries and Compound Queries	337
Using Compound Query Operators	338
Using ORDER BY with a Compound Query	348
Using GROUP BY with a Compound Query	350
Retrieving Accurate Data	353
Summary	353
Q&A	354
Workshop	354

HOURL 20: Creating and Using Views and Synonyms	359
Defining Views	359
Creating Views	361
Updating Data Through a View	374
Dropping a View	379
Understanding the Performance Impact of Nested Views	379
Defining Synonyms	380
Summary	381
Q&A	382
Workshop	382
HOURL 21: Managing Database Users and Security	385
Managing Users in the Database	386
Understanding the Management Process	388
Maximizing Tools Utilized by Database Users	398
Understanding Database Security	398
Assigning Privileges	399
Controlling User Access	402
Controlling Privileges Through Roles	406
Summary	408
Q&A	410
Workshop	411
HOURL 22: Using Indexes to Improve Performance	413
Defining an Index	413
Understanding How Indexes Work	414
Using the CREATE INDEX Command	415
Identifying Types of Indexes	415
Knowing When to Consider Using an Index	420
Knowing When to Avoid Indexes	421
Altering an Index	422
Dropping an Index	423
Summary	423
Q&A	424
Workshop	424

HOUR 23: Improving Database Performance	427
Defining SQL Statement Tuning	427
Comparing Database Tuning and SQL Statement Tuning	428
Formatting Your SQL Statement	428
Running Full Table Scans	434
Identifying Other Performance Considerations	435
Using Cost-Based Optimization	439
Summary	440
Q&A	441
Workshop	441
HOUR 24: Working with the System Catalog	445
Defining the System Catalog	445
Creating the System Catalog	446
Determining What Is Contained in the System Catalog	447
Identifying System Catalog Tables by Implementation	448
Querying the System Catalog	449
Updating System Catalog Objects	454
Summary	455
Q&A	456
Workshop	456
HOUR 25: Bonus Workshop for the Road	459
The BIRDS Database	459
Predators of Birds	461
Photographers of Birds	463
Creating the New Tables	468
Workshop: Describing Your Tables	470
Workshop: Basic Queries	471
Workshop: Adding Tables	471
Workshop: Manipulating Data	472
Workshop: Joining Tables	474
Workshop: Comparison Operators	477
Workshop: Logical Operators	478
Workshop: Conjunctive Operators	480

Workshop: Arithmetic Operators	480
Workshop: Character Functions	481
Workshop: Aggregating Data	484
Workshop: GROUP BY and HAVING	486
Workshop: Composite Queries	487
Workshop: Creating Tables from Existing Tables	490
Workshop: Inserting Data into a Table from Another Table	491
Workshop: Creating Views	491
Workshop: Embedding Subqueries	493
Workshop: Creating Views from Subqueries	493
Workshop: Generating SQL Code from a SQL Statement	494
Summary	495
Workshop	496

APPENDIXES

APPENDIX A: Common SQL Commands	499
SQL Statements	499
SQL Query Clauses	503
APPENDIX B: Popular Vendor RDBMS Implementations	505
Installing the Oracle Database Software Used for Examples and Hands-On Exercises	506
APPENDIX C: Answers to Quizzes and Exercises	507
Hour 1, "Understanding the Relational Database and SQL"	507
Hour 2, "Exploring the Components of the SQL Language"	508
Hour 3, "Getting to Know Your Data"	510
Hour 4, "Setting Up Your Database"	511
Hour 5, "Understanding the Basics of Relational (SQL) Database Design"	512
Hour 6, "Defining Entities and Relationships"	514
Hour 7, "Normalizing Your Database"	517
Hour 8, "Defining Data Structures"	521
Hour 9, "Creating and Managing Database Objects"	523
Hour 10, "Manipulating Data"	524

Hour 11, "Managing Database Transactions"	527
Hour 12, "Introduction to Database Queries"	531
Hour 13, "Using Operators to Categorize Data"	534
Hour 14, "Joining Tables in Queries"	537
Hour 15, "Restructuring the Appearance of Data"	540
Hour 16, "Understanding Dates and Times"	542
Hour 17, "Summarizing Data Results from a Query"	545
Hour 18, "Using Subqueries to Define Unknown Data"	549
Hour 19, "Combining Multiple Queries into One"	551
Hour 20, "Creating and Using Views and Synonyms"	554
Hour 21, "Managing Database Users and Security"	556
Hour 22, "Using Indexes to Improve Performance"	559
Hour 23, "Improving Database Performance"	560
Hour 24, "Working with the System Catalog"	565
Hour 25, "Bonus Workshop for the Road"	567
Index	577

About the Author

Ryan Stephens is an entrepreneur who has built his career and multiple IT companies around SQL, data, and relational databases. He has shared his knowledge and experience with organizations, students, and IT professionals all over the world. Two of the companies he has co-founded, Perpetual Technologies, Inc. (PTI), and Indy Data Partners, have provided expert database and IT services to large-scale government and commercial clients for more than 25 years. Ryan has authored several books for Pearson, including *Sams Teach Yourself SQL in 24 Hours*, 6th Edition; some of his books have been translated and published internationally. Additionally, Ryan has worked for large organizations and has consulted within the areas of SQL, database design, database management, and project management. He designed and taught a database management program for Indiana University–Purdue University in Indianapolis and currently teaches online SQL and database classes for Pearson Education.

Dedication

*This book is dedicated to my daughter,
Charlie Marie, named after my late adoptive mom,
Charlotte Anne Pritchett Stephens, who is the first person I remember
teaching me to write. I love you both.*

Acknowledgments

First of all, I would like to thank the contributing authors to previous editions of this book, Ronald Plew and Arie D. Jones.

Ronald Plew is my retired co-founder of Perpetual Technologies and is a great friend. I could not have started my first successful business without him. Ron studied and consulted in the field of relational database technology for more than 20 years, is a published author, and helped me build a database training program at Indiana University–Purdue University (IUPUI) in Indianapolis. Ron taught SQL and database classes at IUPUI for 5 years.

Arie D. Jones is the vice president for Emerging Technologies for Indy Data Partners, Inc. (IDP), in Indianapolis and helped me get that company off to a successful start. Arie leads IDP's team of experts in planning, design, development, deployment, and management of database environments and applications to achieve the best combination of tools and services for each client. He is a regular speaker at technical events and has authored several books and articles on database-related topics.

Thank you also to Angie Gleim, Amy Reeves, Terri Klein, and all of my team members for doing such a professional job running our companies so that I have the freedom to take on projects like this. And as always, thanks to the staff at Pearson and Sams Publishing for your attention to detail and patience. It is always a pleasure working with you.

We Want to Hear from You!

As the reader of this book, *you* are our most important critic and commentator. We value your opinion and want to know what we're doing right, what we could do better, what areas you'd like to see us publish in, and any other words of wisdom you're willing to pass our way.

We welcome your comments. You can email or write to let us know what you did or didn't like about this book—as well as what we can do to make our books better.

Please note that we cannot help you with technical problems related to the topic of this book.

When you write, please be sure to include this book's title and author as well as your name and email address. We will carefully review your comments and share them with the author and editors who worked on the book.

Email: community@informit.com

What This Book Is About

This book is about the Structured Query Language (SQL), the standard language used to communicate with any relational database. Relational databases are one of the most popular data-management solutions, and many vendor software implementations of SQL are available for organizations and individuals. The text does not focus on a specific implementation of SQL, but Oracle is used for most of the examples because it is a leading database-management software product that adheres closely to the SQL standard. In the upcoming hours, you learn how to apply the SQL standard from a practical standpoint, using real-world examples and hands-on exercises. You can use almost any SQL implementation with this book, which attests to the portability of SQL and its popularity.

What You Need for This Book

This book has no prerequisites, other than a desire to learn about relational databases and SQL. The material is adapted from an approach that the author has used for more than 25 years in the consulting and information technology world to teach thousands of students both locally and globally. It starts with a basic approach to understanding databases and both defining and managing data. It also introduces SQL as the standard language for communicating with any relational database. The book is set up to show all the basic concepts and syntax of SQL while also providing real-world examples. Beginners can easily learn from simplified examples, while experienced information technology professionals can peruse the more advanced exercises.

For software, access to a relational database management system is recommended. This book uses Oracle, but many other options also adhere to the SQL standard (including Microsoft SQL Server, MySQL, IBM, and PostgreSQL). If you do not currently have access to a relational database management system, many options are available for free, either as online access or as a download for learning and development purposes. The author provides guidance on software options in the book as well. All the database creation scripts and data are provided to the reader for hands-on exercises and continued learning after completing this book.

Thank you, and enjoy.

Author's Website

Please visit the author's website at www.ryanSQL.com for additional SQL resources, hands-on exercises, and links to other Pearson Education materials created by the author.

HOUR 1

Understanding the Relational Database and SQL

What You'll Learn in This Hour:

- ▶ Understand information, data, and databases in organizations
- ▶ Examine common database environments
- ▶ Learn about the key components of the relational database
- ▶ Understand how SQL relates to the relational database

Welcome to the world of Structured Query Language (SQL) and the vast, growing database technologies of today's global organizations. By reading this book, you have taken your first step in building the knowledge base needed to navigate today's world of relational databases and data management. Before you dive into SQL, its role in relational databases, and the specifics of the language, you learn the basic concepts of the relational database in this hour so that you have a solid foundation to build upon for increased knowledge and sustainability.

NOTE

SQL Is the Language to Know in IT

SQL is the standard language used to communicate with any relational database, so this first hour concentrates on the relational database itself. This concept is critical in learning SQL and understanding how to most effectively take advantage of your data. In the long run, truly understanding the relational database puts you in a position to excel beyond your peers and make your organization more competitive in a world that heavily relies on data.

The best approach to fully understanding any topic, including SQL, is to start with the basics and build from there. In this hour, you learn the basics of the relational database, get a little SQL history, and look at the key elements that comprise the relational database. From there, you will have a solid foundation to build upon.

SQL is a database language. It is the standard language used to communicate with any relational database, which is one of the most popular databases in use today (and, in fact, for decades). As you begin to understand the nuts and bolts of a relational database—and even database design, which later chapters cover—you can more easily grasp the concepts of SQL and more quickly apply SQL concepts and syntax to real-world situations in real databases. Hour 2, “Exploring the Components of the SQL Language,” focuses on the standard components of the SQL language to provide a preview. Then Hours 3, “Getting to Know Your Data,” and 4, “Setting Up Your Database,” get you involved with the database used in this book so that you deeply understand both the data and how that data is related in the database. When you have this information, you will understand SQL more easily, the knowledge you gain will stick, and you will have the basic foundation to start applying SQL.

Thriving in a Data-Driven World

Data is all around you: You can see it every day in your personal life and the organizations you work for. Data is on your phone, on your computer, in online stores, inside the walls of physical stores, and in complex databases in organizations all over the world. Truly, the world cannot survive without data.

Data, or information, has been around since the beginning of time and people have used that data to make decisions. The modern world has evolved to better understand how data can be used in everyday life and how to make organizations more competitive.

As the world becomes more competitive, it is your responsibility to learn how to use data effectively. You must be able to understand the information and data you’re working with and how to apply it to everyday situations. In today’s world, you have access to mountains of information, especially on the Internet. However, information must become usable data: If the data is not usable, it is useless. If it is inaccurate, it is useless. If it contains inconsistencies, it is useless. Databases and other technologies have helped people better take advantage of data, but this has also created a new problem: The volume of data is growing quickly and must be managed carefully. The relational database and SQL, in particular, facilitate easy data management, if done right.

NOTE

Information Must Become Usable Data

Information is all around us. However, a lot of information in this world is incorrect or not consistent with other sources. Even in databases within some of the most respected organizations in the world, data can be inaccurate or difficult to understand. This can create business problems. You must understand how to turn information into useful data for you and your organization. Fortunately, you can use the relational database and SQL to protect data and present it to end users in a relevant manner. Data must be kept safe, clean, and consistent within the database, and SQL offers a primary way to control that.

Organizations, Data, and User Communities

You know by now that data is critical to the success of any organization or individual. Within an organization, many types of individuals depend on data—some of them every day:

- ▶ Organizational leaders
- ▶ Managers
- ▶ Technical users
- ▶ Administrators
- ▶ Functional and end users
- ▶ Stakeholders
- ▶ Customers

Figure 1.1 illustrates these individuals and also shows how data sits at the core of any organization. All sorts of users access data that is stored in a database of some sort. Applications are available to end users to manage data in a database and query the data in the database. This data is available to customers and other users to access the database in a limited manner, without jeopardizing the integrity of the data itself. Reports and other useful information can be generated from a database so that key personnel in an organization can use them to make high-level decisions; *business intelligence* is another term for this. Additionally, the administrators of the database are included in the general information technology environment, along with technical users that design, develop, and code the elements within a database and the applications surrounding the data. By looking at this simple diagram, you can easily see how important data is in an organization. SQL is the standard language that brings everything together so that all individuals, whether directly or indirectly, can interact with the database and get both accurate and available data to successfully perform their everyday duties.

Databases Defined

In simple terms, a *database* is a collection of data. You might like to think of a database as an organized mechanism that has the capability to store information that a user can retrieve in an effective and efficient manner.

As previously mentioned, people use databases every day without realizing it. For example, a contact list is a database of names, addresses, emails, telephone numbers, and other important information. The listings are alphabetized or indexed, which enables the user to easily look up a particular contact and quickly get in touch. Ultimately, this data is stored in a database somewhere on a computer.

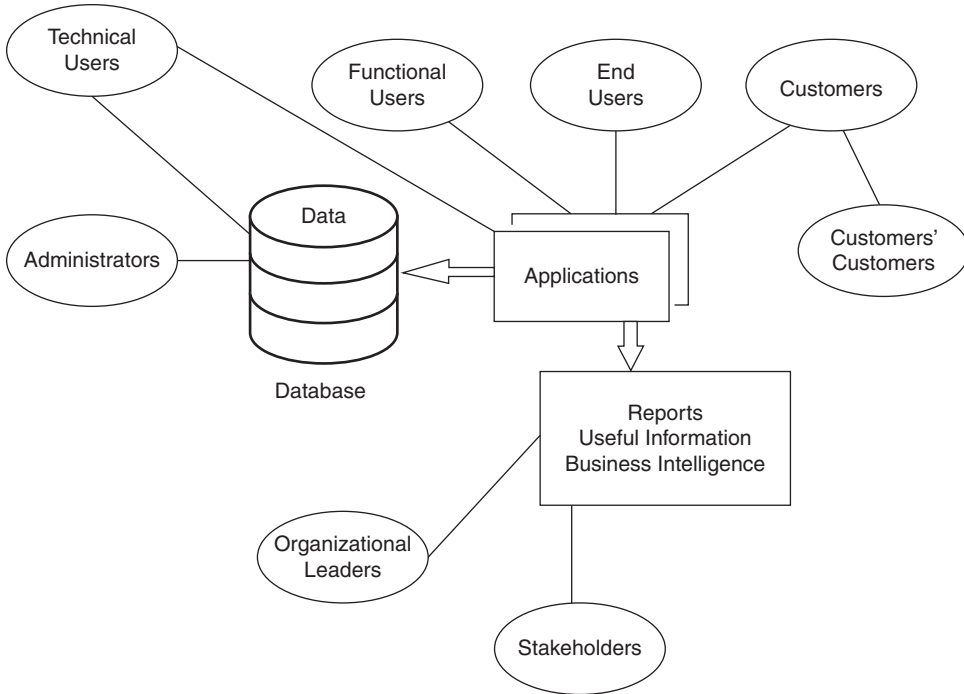


FIGURE 1.1
Data and user communities

Of course, databases must be maintained. As people move to different cities or states, entries have to be added or removed from the contact list. Likewise, entries need to be modified when people change names, addresses, telephone numbers, and so on. Figure 1.2 illustrates a simple database.

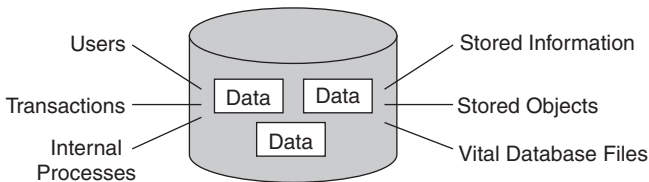


FIGURE 1.2
A typical database

Think of your mobile phone or any mobile device: It contains data and probably has many apps installed that access various databases. Additionally, your phone contains data in the form of contact lists and pictures.

Common Database Environments

In this section, you explore a couple of the most common database environments. A database environment contains all the key components to house a database and store data effectively, from the database itself, to the operating system, the network, and any applications that access the database via these key components. The following two major environments are briefly discussed in this hour:

- ▶ Client/server environments
- ▶ Web-based environments

Client/Server Environments

In the past, the computer industry was predominately ruled by mainframe computers—large, powerful systems capable of high storage capacity and high data processing capabilities. Users communicated with the mainframe through dumb terminals—terminals that did not think on their own but relied solely on the mainframe’s CPU, storage, and memory. Each terminal had a data line attached to the mainframe. The mainframe environment definitely served its purpose and still does in many businesses, but a greater technology was soon introduced: the client/server model.

In the *client/server system*, the main computer, called the server, is accessible from a network—typically, a *local area network (LAN)* or a *wide area network (WAN)*. The server is normally accessed by personal computers (PCs) or other servers instead of dumb terminals. Each PC, called a *client*, is granted access to the network, allowing communication between the client and the server. The main difference between client/server and mainframe environments is that the user’s PC in a client/server environment can think on its own and run its own processes using its own CPU and memory, but it is readily accessible to a server computer through a network. In most cases, a client/server system is much more flexible for today’s overall business needs.

Modern database systems reside on various types of computer systems with various operating systems. The most common types of operating systems are Windows-based systems, Linux, and command-line systems such as UNIX. Databases reside mainly in client/server and web environments.

Lack of training and experience is the main reason for failed implementations of database systems. Today’s businesses need personnel who can work within the client/server model and web-based systems (explained in the next section), can address the rising (and sometimes unreasonable) demands placed on modern organizations, and understand Internet technologies and network computing. Figure 1.3 illustrates the concept of client/server technology.

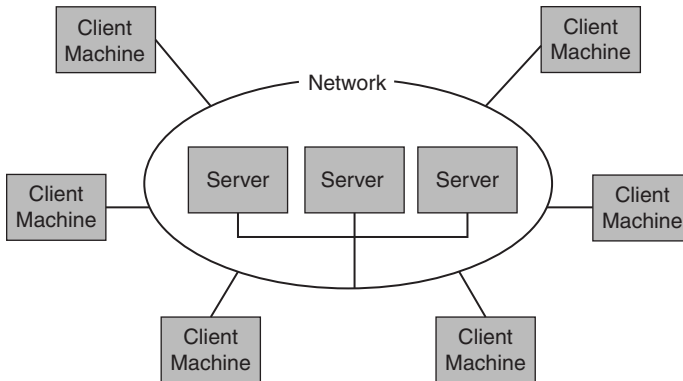


FIGURE 1.3
The client/server model

Web-Based Environments

Business information systems have largely moved toward web integration. Databases have been accessible through the Internet for years, meaning that customers access an organization's information through an Internet browser such as Chrome, Microsoft Edge, or Firefox. Customers (users of data) can order merchandise, check inventories, check the status of their orders, make administrative changes to accounts, transfer money from one account to another, and so on.

A customer simply invokes an Internet browser, goes to the organization's website, logs in (if required by the organization), and uses an application built into the organization's web page to access data. Most organizations first require users to register and then issue them a login and password.

Of course, many actions happen behind the scenes when a database is accessed through a web browser. For instance, the web application can execute SQL that accesses the organization's database, returns data to the web server, and then returns that data to the customer's Internet browser.

From the user's standpoint, the basic structure of a web-based database system is similar to that of a client/server system (refer to Figure 1.3). Each user has a client machine that is connected to the Internet and contains a web browser. The network in Figure 1.3 (for a web-based database) just happens to be the Internet instead of a local network. For the most part, a client is still accessing a server for information; it doesn't matter that the server might exist in another state or even another country. The main point of web-based database systems is to expand the potential customer base of a database system that knows no physical location bounds, thus increasing data availability and an organization's customer base.

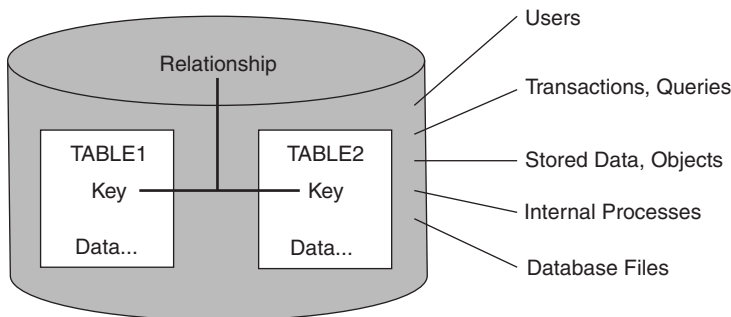
NOTE

Data Integration Across Multiple Environments

These days, organizations and individuals alike use a variety of databases and applications to access data that is often spread among multiple environments. With modern technology, data located in a variety of environments, vendor implementations, and even types of databases can be seamlessly integrated.

Understanding the Relational Database

A *relational database* is a database that is divided into logical units called tables. These tables are related to one another within the database. A relational database allows data to be broken down into logical, smaller, manageable units, facilitating easier maintenance and providing more optimal database performance according to the level of organization. In Figure 1.4, you can see that tables are related to one another through a common key (data value) in a relational database.

**FIGURE 1.4**

The relational database

Again, tables are related in a relational database so that adequate data can be retrieved in a single query (although the desired data might exist in more than one table). Having common *keys*, or *fields*, among relational database tables allows data from multiple tables to be joined to form one large set of data. As you venture deeper into this book, you see more of a relational database's advantages, including overall performance and easy data access.

NOTE

Relationships Within a Relational Database

The following sections provide an overview of relationships between data in a RDBMS. Sample data illustrates how data can be related to one another and how easy it is to bring common data together. Future hours expand upon this through numerous examples and hands-on exercises. You will even get the opportunity to design a portion of the sample database in this book and define relationships of your own.

Taking a Glimpse into a Sample Database

All databases have a simple reason for existing: They store and maintain valuable data. In this section, you look at a simplified example data set that illustrates how data might look in a relational database and also shows how data is related through key relationships. These relationships, which are also rules for how data is stored, hold the key to the value of a relational database.

Figure 1.5 shows a table called `EMPLOYEES`. A table is the most basic type of object where data is stored within a relational database. A database object is a defined structure that physically contains data or has an association to data stored in the database.

Table: `EMPLOYEES`

ID	LAST_NAME	FIRST_NAME
1	Smith	Mary
2	Jones	Bob
3	Williams	Steve
4	Mitchell	Kelly
5	Burk	Ron

FIGURE 1.5
Table structure

Fields

Every table is divided into smaller entities called fields. A *field* is a column in a table that is designed to maintain specific information about every record in the table. The fields in the `EMPLOYEES` table consist of `ID`, `LAST_NAME`, and `FIRST_NAME`. They categorize the specific information that is maintained in a given table. Obviously, this is a simplistic example of the data that might be stored in a table such as `EMPLOYEES`.

Records, or Rows of Data

A *record*, also called a *row* of data, is a horizontal entry in a table. Looking at the last table, `EMPLOYEES`, consider the following first record in that table:

```
1      Smith      Mary
```

The record consists of an employee identification, employee last name, and employee first name. For every distinct employee, there should be a corresponding record in the `EMPLOYEES` table.

A *row of data* is an entire record in a relational database table.

Columns

A *column* is a vertical entity in a table that contains all the information associated with a specific field in a table. For example, a column in the `EMPLOYEES` table for the employee's last name consists of the following:

```
Smith
Jones
William
Mitchell
Burk
```

This column is based on the field `LAST_NAME`, the employee's last name. A column pulls information about a certain field from every record within a table.

Referential Integrity

Referential integrity is the hallmark of any relational database. Figure 1.6 shows two tables, `EMPLOYEES` and `DEPENDENTS`, that are related to one another in our imaginary database. The `DEPENDENTS` table is simply a table that contains information about dependents of each employee in the database, such as spouses and children. As in the `EMPLOYEES` table, the `DEPENDENTS` table has an `ID`. The `ID` in the `DEPENDENTS` table references, or is related to, the `ID` in the `EMPLOYEES` table. Again, this is a simplistic example to show how relationships work in a database and to help you understand referential integrity.

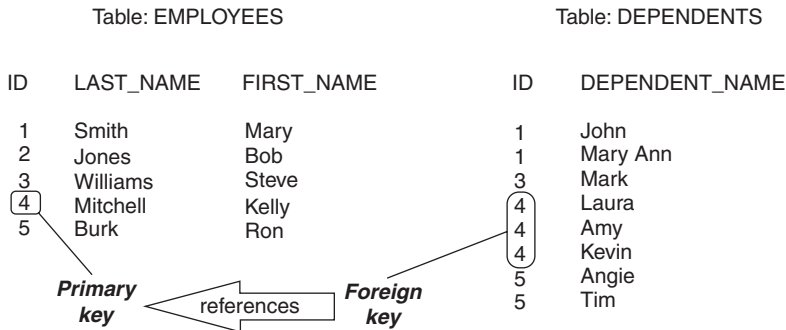


FIGURE 1.6
Table relationships

The key point to note in Figure 1.6 is that there is a relationship between the two tables through the `ID` field. The `ID` field in `EMPLOYEES` is related to the `ID` field in `DEPENDENTS`. The `ID` field in `EMPLOYEES` is a primary key, whereas the `ID` field in `DEPENDENTS` is a foreign key. These types of keys are critical to any relational database structure and to referential integrity.

Primary Keys

A *primary key* is a column that makes each row of data in the table unique in a relational database. In Figure 1.6, the primary key in the `EMPLOYEES` table is `ID`, which is typically initialized during the table creation process. The primary key ensures that all employee identifications are unique, so each record in the `EMPLOYEES` table has its own `ID`. Primary keys alleviate the possibility of a duplicate record in a table and are used in other ways, which you learn more about as you progress throughout the book.

Primary keys are typically initialized during the table creation process, although a primary key can be added later as long as duplicate data does not already exist in the table.

Foreign Keys

A *foreign key* is a column in a table that references a column in another table. Primary and foreign keys establish relationships between tables in a relational database.

Columns identified as foreign keys have these characteristics:

- ▶ Do not have to have unique values
- ▶ Ensure that each entry in the column has a corresponding entry in the referenced table
- ▶ Ensure that column data (child records) are never deleted if corresponding data (parent records) is found in the table referenced (the primary key column)

In Figure 1.6, the foreign key in the `DEPENDENTS` table is `ID`, which is the column that contains the employee `ID` of the corresponding table in the `EMPLOYEES` table. Again, this is a simplistic example: Ideally, the `DEPENDENTS` table would have an `ID` for each dependent and an `ID` for each employee referenced. So for learning purposes, the `ID` in the `DEPENDENTS` table is a foreign key that references the `ID` in the `EMPLOYEES` table. The record, or row of data, in the `EMPLOYEES` table, is a parent record because of the primary key, which might have child records within the database. Likewise, the `ID` in the `DEPENDENTS` table is a foreign key, or child record, which requires a relationship to a parent record, or primary key, somewhere in the database.

NOTE

Relationships Within a Relational Database

Primary and foreign key relationships, or parent/child relationships, are the most basic relationships in a relational database. This is the most basic rule that can be set and is the only way to facilitate the ease and efficiency that a relational database provides for effective data management.

When manipulating data in the database, keep the following points in mind:

- ▶ You add data into a column identified as a foreign key unless corresponding data already exists as a primary key in another table.

- ▶ You cannot delete data from a column identified as a foreign key unless you first remove any corresponding data from columns in tables with primary keys that are referenced by the foreign key.

Getting Data from Multiple Tables

In Figure 1.6, you should understand the primary and foreign keys that have been defined at this point. When these keys are defined upon table creation or modification, the database knows how to help maintain the integrity of data within and also knows how that data is referenced between tables.

Let's say you want to know the dependents of Kelly Mitchell. You have likely already used a common-sense process while looking at Figure 1.6. However, this was probably your thought process:

- ▶ You see that Kelly Mitchell's ID is 4.
- ▶ You look for Kelly Mitchell's ID of 4 in the `DEPENDENTS` table.
- ▶ You see that Kelly Mitchell's ID of 4 corresponds to three records in the `DEPENDENTS` table.
- ▶ Therefore, Kelly Mitchell's dependents are Laura, Amy, and Kevin.

This common-sense process of finding information is almost exactly the process that occurs behind the scenes when you use SQL to “ask” a relational database about the data it contains.

NULL Values

`NULL` is the term used to represent a missing value. A *NULL value* in a table is a value in a field that appears to be blank; that field has no value. It is important to understand that a `NULL` value is different from a zero value or a field that contains spaces: A field with a `NULL` value has intentionally been left blank during record creation. For example, a table that contains a column called `MIDDLE_NAME` might allow null or missing values because not every person has a middle name. Records in tables that do not have an entry for a particular column signify a `NULL` value.

Additional table elements are discussed in detail during the next two hours.

Logical vs. Physical Database Elements

Within a relational database, both logical and physical elements exist throughout the database lifecycle. Logical elements are typically conceived as database structures during the planning and design phase. Physical structures are objects that are created later; they comprise the database itself that stores data that SQL and various applications access.

For example, logical elements might include the following elements during conception:

- ▶ Entities
- ▶ Attributes
- ▶ Relationships
- ▶ Information/data

Those logical elements later become the following physical elements during database creation:

- ▶ Tables
- ▶ Fields/columns
- ▶ Primary and foreign key constraints (built-in database rules)
- ▶ Usable data

Database Schemas

A *schema* is a group of related objects in a database. A schema is owned by a single database user, and objects in the schema can be shared with other database users. Multiple schemas can exist in a database. Figure 1.7 illustrates a database schema.

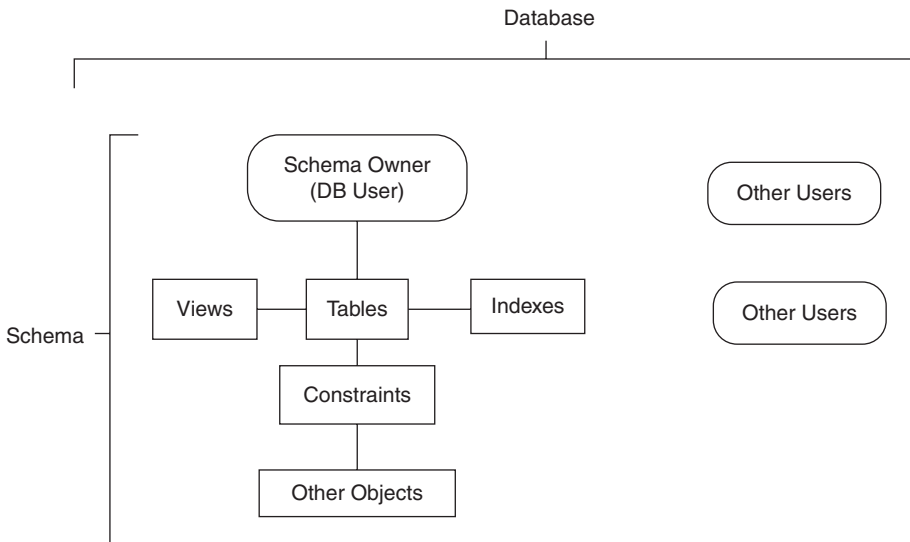


FIGURE 1.7
A schema

The Relational Database Continues to Lead the Way

The relational database has been the superior database choice for effectively managing data for several decades, and it continues to dominate the market share. This is true for many reasons:

- ▶ A well-designed relational database provides a simple, organized, easy-to-manage mechanism for data.
- ▶ A relational database is scalable as data grows and objects are added to the database.
- ▶ Linking data in multiple tables is easy.
- ▶ Maintaining the integrity of data is simple using built-in features such as referential integrity (primary and foreign key constraints).
- ▶ The overall management of data and use of SQL to communicate with the database is simplified.
- ▶ Relevant and useful data is easy to retrieve.

SQL and the relational database management system go hand in hand; you cannot have one without the other. SQL is an English-like language that enables you to create and manage a relational database and to then easily and effectively retrieve data from the database for a specific purpose.

Examples and Exercises

Most of the exercises that you encounter in this book use Oracle, a dominant leader in the market that adheres closely to the SQL standard but also offers many enhancements.

Oracle (as well as many implementations of SQL, or the relational database), make freely distributed versions of its database available. You can then select an implementation of your choice, install it, and follow along with the exercises in the book. Note that because these databases are not 100% compliant with SQL:2016, the exercises might have slight variations of the exact syntax suggested in the ANSI standard. However, if you learn the basics of the ANSI standard, you can generally translate your skills between different database implementations.

Summary

In this hour, you got a brief introduction to the SQL language and the relational database. SQL is the standard language for communicating with any relational database. It is important to first understand what comprises the relational database, identify the key elements, know what data

looks like, and understand how the data in the relational database is related to other data within the database. When you fully grasp these concepts, your journey into the SQL language will take flight sooner and the concepts in this book will make more sense. The goal of this hour was to introduce some basic database concepts and give you a foundation for thinking about data at a higher level.

Simply put, a relational database is a database that is logically organized into multiple tables that might or might not be related to one another. A table consists of one or more columns, or fields, and one or more rows of data. A table can have just a few rows of data or millions upon millions of rows of data. Tables are related through common columns, or fields, that are defined in a relational database using constraints such as primary keys and foreign keys. These keys are built-in features of a relational database that maintain referential integrity within the database. This is the key to the relational database and one of the main components of SQL. Subsequent chapters cover all these concepts in examples and hands-on exercises.

Q&A

Q. If I learn SQL, can I use any of the implementations that use SQL?

A. Yes, you can communicate with a database whose implementation is compliant with ANSI SQL. If an implementation is not completely compliant, you likely will pick it up quickly with some adjustments.

Q. In a client/server environment, is the personal computer the client or the server?

A. The personal computer is the client in a client/server environment, although a server can also serve as a client.

Q. In the context of a database, how do information and usable data differ?

A. Information and data are conceptually the same. However, when you are creating a database, it is important to gather all required information and work to form that information into usable data. Usable data comes from reliable sources and is accurate, consistent, clean, and up-to-date. Usable data is critical for business operations and when making critical decisions. Data that is derived from inaccurate sources or that has not properly been entered or evolved with the database is not usable.

Q. Can SQL be used to integrate databases among multiple environments, platforms, or types of databases?

A. When working with a relational database, SQL is the primary language and tool for working with data, including integrating data among multiple sources. SQL is also necessary when migrating data between sources. Medium to large organizations commonly have many different sources of data, only some of which are SQL based. SQL is an important component for pulling everything together.

Workshop

The following workshop consists of a series of quiz questions and practical exercises. The quiz questions are designed to test your overall understanding of the current material. The practical exercises give you the opportunity to apply the concepts discussed during the current hour, as well as build on the knowledge you acquired in previous hours of study. Be sure to complete the quiz questions and exercises before continuing to the next hour. Refer to Appendix C, “Answers to Quizzes and Exercises,” for answers.

Quiz

1. What does the acronym SQL stand for?
2. What is a schema? Give an example.
3. How do logical and physical components within a relational database differ? How are they related?
4. What keys are used to define and enforce referential integrity in a relational database?
5. What is the most basic type of object in a relational database?
6. What elements comprise this object?
7. Must primary key column values be unique?
8. Must foreign key column values be unique?

Exercises

For the following exercises, refer to Figure 1.6, earlier in this hour.

1. Who are Mary Smith's dependents?
2. How many employees do not have dependents?
3. How many duplicate foreign key values exist in the `DEPENDENTS` table?
4. Who is Tim's parent or guardian?
5. Which employee can be deleted without first having to delete any dependent records?

Index

SYMBOLS

- * (asterisk)
 - in COUNT function, 301
 - as multiplication operator, 230–231
 - in SELECT command, 144
- || (double pipe), concatenation, 261
- = (equality) operator, 210
- ! (exclamation mark), negating equality operator, 225
- > (greater than) operator, 212–213
- < (less than) operator, 212–213
- (minus sign) operator, 229–230
- <> (non-equality) operator, 211–212
- () (parentheses), grouping queries, 224
- + (plus sign)
 - addition operator, 229
 - concatenation, 261
 - outer joins, 243
- ' (quotation marks)
 - concatenation, 262

- inserting data, 155
- for numeric data types, 234
- ; (semicolon) in SQL statements, 135
- / (slash) division operator, 231–232

A

- abandoned privileges, 404
- ABOVE_AVG_BIRDS table
 - creating, 325–326
 - deleting data, 329–330
 - inserting data, 327
- ABS function, 272
- access control. *See* privileges
- accuracy of data, compound queries and, 353
- ADD_MONTH function, 287
- adding
 - columns, 138
 - autoincrementing columns, 140–141
 - mandatory columns, 139–140

- data to databases, 67–68
- entities, 37
- time to dates, 286–288
- addition operator, 229**
- ADMIN OPTION command, 403**
- aggregate functions, 484–486**
 - AVG, 303–304
 - COUNT, 300–302
 - DISTINCT keyword and, 305
 - GROUP BY clause and, 308–309
 - input for, 317
 - MAX, 304–305
 - MIN, 305–306
 - purpose of, 299
 - SUM, 302–303
- aliases, 204–206, 207, 241**
- ALL operator, 220**
- ALL_TABLES table, accessing, 451–452**
- alphanumeric, fields as, 126**
- ALTER INDEX command, 422–423**
- ALTER TABLE command, 137–141, 499**
 - dropping constraints, 150
 - vendor implementations, 141, 149
- ALTER USER command, 396, 499**
- ALTER VIEW command, 362**
- altering**
 - indexes, 422–423
 - users, 396
- American National Standards Institute. See ANSI standard**
- AND operator, 221–222**
- ANSI standard, 13, 18–19, 36, 259, 276**

- ANY operator, 220–221**
- arguments, 189**
- arithmetic operators, 228–229, 480–481**
 - addition, 229
 - combinations of, 232
 - division, 231–232
 - multiplication, 230–231
 - subtraction, 230
- ascending order, 195, 200–201**
- ASCII function, 271**
- asterisk (*)**
 - in COUNT function, 301
 - as multiplication operator, 230–231
 - in SELECT command, 144
- attributes. See also columns**
 - in BIRDS example database, 34–35, 66–67
 - entities and, 33
 - modifying, 138–139
- autoincrementing columns, adding, 140–141**
- automated database design, 75**
- automated population of data, 154**
- AVG function, 303–304**

B

- base tables, joins with, 250–252, 256**
- batch loads, disabling indexes during, 438**
- BETWEEN operator, 216–217, 322, 324–325**

- BIG_BIRDS table**
 - converting characters to numbers, 274–275
 - converting numbers to characters, 273
 - creating, 172, 189
 - creating synonyms, 380–381
 - IS NOT NULL operator, 227–228
 - LENGTH function, 269
 - NOT EXISTS operator, 228
 - NOT LIKE operator, 227
 - rolling back to savepoint, 177–180
 - selecting all records, 190–191
 - TRANSLATE function, 266
- bird rescue information, 67–68**
 - ERDs, creating, 87–89
 - first normal form (FNF), 107–108
 - groups of data, creating, 69
 - lists of fields, creating, 69–71
 - second normal form (SNF), 108
 - third normal form (TNF), 108–109
- BIRDS example database, 29–30, 459–461. See also names of specific tables in database**
 - adding data to, 67–68
 - aggregate functions, 300, 484–486
 - ANSI standard and, 36
 - arithmetic operators, 480–481
 - base table joins, 250–252
 - Cartesian products, 252–254
 - case sensitivity, 198–199

- character functions, 481–484
 - comparison operators, 477–478
 - compound queries, 487–490
 - conjunctive operators, 480
 - creating
 - schemas, 393–394
 - tables from existing tables, 490
 - tables/data, 46–48, 468–470, 471–474
 - users, 392–393
 - views, 491–493
 - views from subqueries, 493–494
 - data included, 30–31
 - data types in, 123–124
 - database design process, 64–71
 - gathering data, 68
 - knowing data, 65–68
 - modeling data, 68–71
 - describing tables, 470–471
 - embedded subqueries, 493
 - entities and attributes in, 34–35
 - ERD for, 32, 66–67, 87–89
 - expansion opportunities, 31–32
 - file locations, 39
 - generating SQL code, 494–495
 - GROUP BY clause, 486
 - HAVING clause, 486
 - inner joins, 239–240
 - inserting data into tables, 491
 - joins, 474–477
 - list of tables/data, 49–56
 - logical operators, 478–480
 - manipulating data, 472–474
 - naming conventions, 105
 - non-equijoins, 242–243
 - normalization
 - first normal form (FNF), 98, 107–108
 - second normal form (SNF), 100–101, 108
 - third normal form (TNF), 102–104, 108–109
 - outer joins, 244–245
 - referential integrity, 79–81, 84–87
 - foreign keys, 86–87
 - primary keys, 85–86
 - relationships, 78–79
 - many-to-many, 83
 - one-to-many, 82–83
 - one-to-one, 82
 - resetting to original, 58
 - SELECT command, 471
 - selecting data based on conditions, 193–194
 - users of, 31
- BIRDS table, 460–461**
- ALL operator, 220
 - AND operator, 221–222
 - ANY operator, 220–221
 - AVG function, 304
 - base table joins, 250–252
 - BETWEEN operator, 216–217
 - Cartesian products, 252–254
 - check constraints, 149
 - COALESCE function, 269–270
 - column aliases, 205
 - columns/data types, 123
 - combinations of comparison operators, 213–214
 - compound queries
 - INTERSECT operator, 346–347
 - MINUS operator, 347–348
 - ORDER BY clause, 348–350
 - OR operator, 342–343
 - single queries versus, 337–338
 - UNION ALL operator, 344–346
 - UNION operator, 340–342
 - correlated subqueries, 333
 - COUNT function, 301
 - counting records, 203–204
 - creating, 135–136
 - tables from existing tables, 141–143, 189
 - views, 362–363, 364–367, 373–374
 - data in, 49–50
 - DECODE function, 264–265
 - distinct values, 191–192
 - dropping, 145
 - embedded subqueries, 330–332
 - equality operator, 210
 - EXISTS operator, 219
 - greater than operator, 212
 - GROUP BY versus ORDER BY clauses, 309–312
 - grouping data, 308–309
 - HAVING clause, 315–316
 - inner joins, 239–240

- inserting data
 - into columns, 156
 - from queries, 157–158
 - into tables, 154–155
 - IS NULL operator, 214–216
 - less than operator, 212–213
 - LTRIM function, 267
 - MAX function, 304–305
 - MIN function, 305–306
 - modifying, 137–140
 - non-equality operator, 211–212
 - non-equijoins, 242–243
 - NOT BETWEEN operator, 226
 - OR operator, 222–224
 - outer joins, 244–245
 - primary key constraints, 146–147
 - roles, creating, 453–454
 - RTRIM function, 268
 - sorting data, 200–202
 - subqueries
 - with CREATE TABLE command, 325–326
 - with INSERT command, 327
 - with SELECT command, 323–325
 - with UPDATE command, 328–329
 - SUM function, 303
 - unique constraints, 147–148
- BIRDS_FOOD table**
- columns/data types, 124
 - creating views, 365–367
 - data in, 51–52
 - HAVING clause, 315–316
- BIRDS_MIGRATION table**
- columns/data types, 124
 - creating views, 370–371
 - data in, 53–54
- BIRDS_NESTS table**
- columns/data types, 124
 - data in, 55
- BIRDS_PREDATORS table**
- creating, 468–470
 - data in, 461–463
- BIRDS_TRAVEL table**
- composite indexes, 419
 - single-column indexes, 416–418
 - unique indexes, 418–419
- BLOB data type, 117**
- Boolean values, 121**
- business intelligence, 3**
- C**
- cardinality, 422
 - Cartesian products, 252–255
 - case sensitivity
 - of data, 154, 197–199
 - of SQL commands, 136
 - CAST command, 114
 - CAST operator, 290
 - CEIL function, 272
 - Center for Internet Security, 399
 - CHAR data type, 116
 - character functions, 481–484
 - ASCII, 271
 - COALESCE, 269–270
 - combinations of, 275
 - CONCAT, 261–263
 - DECODE, 264–265
 - LENGTH, 268–269
 - LOWER, 263–264
 - LPAD, 270
 - LTRIM, 267
 - purpose of, 259–260
 - REPLACE, 266–267
 - RPAD, 271
 - RTRIM, 268
 - SUBSTR, 265
 - TRANSLATE, 266
 - UPPER, 263
 - character values
 - converting numbers to, 273
 - converting to numbers, 273–275
 - character-based sorts, 196
 - check constraints, 149
 - child entities, 79–81
 - choosing database design methodology, 63–64
 - clients, 5
 - client/server environments, 5–6, 15
 - COALESCE function, 269–270
 - Codd, E. F., 18
 - collation, 198
 - columns, 9. *See also* attributes
 - adding, 138
 - autoincrementing columns, 140–141
 - mandatory columns, 139–140
 - aliases, 204–206, 207
 - attributes, modifying, 138–139

- in BIRDS example database, 123–124
 - cardinality, 422
 - in compound queries, 354
 - data types, 114, 132–133
 - distinct values, 191–192
 - dropping, 137
 - in GROUP BY clause, 317
 - inserting data, 156
 - joins on multiple, 256
 - modifying, factors in, 141
 - naming standards, 132
 - privileges on, 404
 - qualifying, 240
 - restricting access, 361
 - sorting, 195–197, 200–202
 - in system catalog, 456
 - unique constraints, 147–148
 - updating
 - multiple columns, 161–162
 - single column, 158–161
- command-line prompt, 45**
- commands**
- ADMIN OPTION, 403
 - ALTER INDEX, 422–423
 - ALTER TABLE, 137–141, 499
 - dropping constraints, 150
 - vendor implementations, 141, 149
 - ALTER USER, 396, 499
 - ALTER VIEW, 362
 - case sensitivity, 136
 - CAST, 114
 - COMMIT, 169–171, 183, 499
 - CONNECT, 21–22, 397
 - CONVERT, 114
 - CREATE DATABASE, 395
 - CREATE INDEX, 415, 499
 - CREATE ROLE, 407, 500
 - CREATE SCHEMA, 393–395
 - CREATE SYNONYM, 380–381
 - CREATE TABLE, 134–136, 189–191, 471–474, 500
 - with SELECT command, 141–144
 - subqueries and, 325–326
 - CREATE TABLE AS, 372–373, 500
 - CREATE TYPE, 122, 500
 - CREATE USER, 389–390, 500
 - CREATE VIEW, 361–362, 500–501
 - from multiple tables, 364–368
 - ORDER BY clause, 373–374
 - from single table, 362–363
 - from views, 368–372
 - WITH CHECK option, 376–379
 - DELETE, 162–163, 329–330, 501
 - DESC, 470–471
 - DISCONNECT, 22, 397
 - DROP INDEX, 423, 501
 - DROP ROLE, 407–408
 - DROP SCHEMA, 395
 - DROP SYNONYM, 381
 - DROP TABLE, 145–146, 501
 - DROP USER, 398, 501
 - DROP VIEW, 379, 501
 - EXIT, 22
 - EXPLAIN PLAN, 440
 - formatting for readability, 429–431
 - generating SQL code from, 494–495
 - GRANT, 389–390, 402–403, 501
 - GRANT OPTION, 403
 - INSERT, 165, 502
 - into columns, 156
 - committing, 183
 - NULL values, 158
 - with queries, 156–158, 502
 - subqueries and, 326–327
 - for tables, 154–155
 - RELEASE SAVEPOINT, 180–181
 - REVOKE, 398, 404, 502
 - ROLLBACK, 171–176, 183, 502
 - ROLLBACK TO SAVEPOINT, 177–180
 - SAVE TRANSACTION, 177
 - SAVEPOINT, 176–177, 502
 - SELECT, 23–24, 115, 471. 502, *See also* joins
 - asterisk (*) in, 144
 - case sensitivity of data, 197–199
 - FROM clause, 192, 207, 503
 - column aliases, 204–206, 207
 - COUNT function, 202–204
 - with CREATE TABLE command, 141–144

- CUBE expression, 313–314
- DISTINCT keyword, 191–192, 207
- with DML commands, 153–154
- examples of queries, 199–202
- GROUP BY clause, 306–312, 503
- HAVING clause, 315–316, 351–353, 437, 504
- with INSERT command, 156–158
- multiple. *See* compound queries
- ORDER BY clause, 195–197, 207, 309–312, 317, 504
- order of operation for clauses, 207
- on other users' tables, 204
- purpose of, 187–188
- ROLLUP expression, 312–313
- SELECT clause, 188–191, 503
- subqueries and, 323–325
- WHERE clause, 192–194, 503. *See also* operators
- SET CONNECTION, 397
- SET IMPLICIT_TRANSACTIONS, 170–171
- SET ROLE, 408
- SET TRANSACTION, 181
- SHOW USER, 449–450
- SP_ADDLOGIN, 391
- SP_ADDUSER, 391
- START, 46
- TKPROF, 440
- tuning. *See* database performance
- types of, 22
 - data administration, 24
 - DCL (Data Control Language), 24
 - DDL (Data Definition Language), 22
 - DML (Data Manipulation Language), 23, 153–154, 472–474
 - DQL (Data Query Language), 23–24. *See also* SELECT command
 - transactional control, 24–25, 165, 168
- UPDATE, 158, 503
 - after COMMIT command, 183
 - in multiple columns, 161–162
 - in single column, 158–161
 - subqueries and, 328–329
 - with views, 374–376
- COMMIT command, 169–171, 183, 499**
- committing transactions, 169–171, 183**
- comparison operators, 210, 477–478**
 - combinations of, 213–214, 223
 - equality, 210
 - greater than, 212–213
 - less than, 212–213
 - non-equality, 211–212
- composite indexes, 419**
- composite primary keys, 86**
- compound queries, 487–490**
 - data accuracy, 353
 - data types and column lengths, 354
 - EXCEPT/MINUS operators, 347–348, 354
 - GROUP BY clause, 350–353, 354
 - INTERSECT operator, 346–347
 - ORDER BY clause, 348–350
 - single queries versus, 337–338
 - UNION ALL operator, 343–346
 - UNION operator, 338–343, 354
- CONCAT function, 261–263**
- concatenation, 260, 261–263**
- conditions, 157, 192, 194, 431–432**
- conjunctive operators, 221, 480**
 - AND, 221–222
 - OR, 222–224, 342–343, 436–437
- CONNECT command, 21–22, 397**
- connections to databases, 21–22, 397**
- constant characters, 115**
- constraints**
 - check, 149
 - dropping, 150
 - foreign keys, 148
 - implicit indexes and, 420
 - NOT NULL value, 149
 - primary keys, 146–147
 - on queries, 202

- referential integrity and, 90, 146
 - unique, 147–148
 - on unique indexes, 419
 - violating, 138
- conversion functions**
- purpose of, 272
 - TO_CHAR, 115, 273
 - TO_DATE, 294–295
 - TO_NUMBER, 273–275
- CONVERT command, 114**
- converting**
- data types, 114, 120, 126
 - date and time, 290
 - date pictures, 290–293
 - to strings, 294
 - from strings, 294–295
 - values
 - characters to numbers, 273–275
 - numbers to characters, 273
- correlated subqueries, 333**
- COS function, 272**
- cost-based optimization, 439**
- COUNT function, 202–204, 300–302, 452**
- counting records, 202–204**
- CREATE DATABASE command, 395**
- CREATE INDEX command, 415, 499**
- CREATE ROLE command, 407, 500**
- CREATE SCHEMA command, 393–395**
- CREATE SYNONYM command, 380–381**
- CREATE TABLE AS command, 372–373, 500**
- CREATE TABLE command, 134–136, 189–191, 471–474, 500**
- with SELECT command, 141–144
 - subqueries and, 325–326
- CREATE TYPE command, 122, 500**
- CREATE USER command, 389–390, 500**
- CREATE VIEW command, 361–362, 500–501**
- from multiple tables, 364–368
 - ORDER BY clause, 373–374
 - from single table, 362–363
 - from views, 368–372
 - WITH CHECK option, 376–379
- cross joins, 252–255**
- crosstab reports, 313–314**
- CUBE expression, 313–314**
- current date, retrieving, 282–285**
- D**
- data. See also attributes; entities**
- adding new, 67–68
 - in BIRDS example database, 30–31, 49–56
 - case sensitivity, 154, 197–199
 - creating in tables, 46–48
 - deleting in tables, 47–48, 162–163
 - explained, 114
 - gathering (for database design), 68
 - grouping, 306
 - with aggregate functions, 307, 308–309
 - with GROUP BY clause, 306–307
 - information and, 2, 15
 - inserting, 165
 - into columns, 156
 - NULL values, 158
 - from queries, 156–158
 - into tables, 154–155, 491
 - justification, 273
 - knowing (for database design), 65–68
 - modeling (for database design), 68–71
 - groups of data, 69
 - lists of fields, 69–71
 - populating, 154
 - redundancy
 - in denormalization, 106–107
 - reducing, 95–96
 - retrieving from multiple tables, 11, 237–238. *See also* joins
 - selecting. *See* SELECT command
 - sorting, 195–197, 200–202, 207, 437
 - in system catalog, 447
 - database design, 448
 - performance statistics, 448
 - security information, 448
 - user data, 448
 - updating, 158
 - in multiple columns, 161–162

- in single column, 158–161
 - through views, 374–376
 - user communities and, 3–4
- data accuracy, compound queries and, 353**
- data administration commands, 24**
- Data Control Language. See DCL (Data Control Language)**
- Data Definition Language. See DDL (Data Definition Language)**
- data dictionary. See system catalog**
- data integrity, 105–106**
- Data Manipulation Language. See DML (Data Manipulation Language)**
- Data Query Language. See DQL (Data Query Language)**
- data types**
 - AVG function and, 304, 317
 - in BIRDS example database, 123–124
 - Boolean values, 121
 - in columns/fields, 132–133
 - in compound queries, 354
 - COUNT function and, 302, 317
 - date and time, 119–120, 280–281
 - decimal, 118
 - domains, 122
 - explained, 114–115
 - fixed-length strings, 115–116, 126
 - floating-point decimal, 119
 - identifying by justification, 273
 - integer, 118–119
 - large object types, 117
 - length limits, 126
 - literal strings, 120
 - NULL values, 121
 - numeric, 117–118, 126
 - precision, 304
 - purpose of, 114
 - SUM function and, 303, 317
 - user-defined, 122
 - varying-length strings, 116, 126
 - vendor implementations, 121
- database administration, purpose of, 24**
- database administrators (DBAs), 46, 105–106, 387**
- database design**
 - for BIRDS example database, 64–71
 - creating ERDs, 87–89
 - gathering data, 68
 - knowing data, 65–68
 - modeling data, 68–71
 - referential integrity, 79–81, 84–87
 - relationships, 78–79, 82–84
 - choosing methodology, 63–64
 - deploying, 75
 - end-user needs, 111
 - logical versus physical design, 71–72
 - manual versus automated, 75
 - normalization, 237–238
 - benefits of, 105–106
 - denormalization versus, 111
 - drawbacks of, 106
 - naming conventions, 104–105
 - normal forms, 96–104, 107–109, 111
 - purpose of, 94
 - raw databases, 94
 - redundancy, 95–96
 - process of, 62–63
 - purpose of, 61
 - relationships
 - many-to-many, 83
 - one-to-many, 82–83
 - one-to-one, 82
 - purpose of, 78–79
 - recursive, 84
 - referential integrity, 79–81, 84–87
 - types of, 81–82
 - requirements gathering, 68
 - in system catalog, 448
- database life cycle, 72–74**
 - development environment, 72
 - production environment, 74
 - test environment, 73
- database management systems (DBMSs), 17**
- database objects, 8, 129. See also names of specific objects (tables, views, clusters, etc.)**
 - naming standards, 136–137
 - schemas and, 129–131, 388
 - system catalog and, 445–446
- database performance**
 - cost-based optimization, 439
 - expected gains, 441
 - full table scans, 434–435

- indexes
 - altering, 422–423
 - composite, 419
 - creating, 415
 - disabling during batch loads, 438
 - dropping, 423
 - how they work, 414–415
 - implicit, 420
 - planning, 420
 - purpose of, 413–414
 - re-creating, 424
 - single-column, 416–418
 - storage considerations, 424
 - unique, 418–419, 424
 - when to avoid, 421–422
 - when to use, 420
- joins, 250, 431–432
- large sort operations, 437
- SQL statement tuning
 - database tuning versus, 428
 - formatting for readability, 429–431
 - HAVING clause, 437
 - LIKE operator and wildcards, 435–436
 - most restrictive condition, 432–434
 - OR operator, 436–437
 - purpose of, 427
 - table order in FROM clause, 431
- statistics in system catalog, 448
- stored procedures, 437
- subqueries and, 332
- tools for, 440, 441
- transactions and, 181
- views and, 379, 438
- database tuning, 428**
- databases. See also attributes; entities**
 - BIRDS example database, 29–30, 459–461. *See also names of specific tables in database*
 - adding data to, 67–68
 - aggregate functions, 300, 484–486
 - ANSI standard and, 36
 - arithmetic operators, 480–481
 - base table joins, 250–252
 - Cartesian products, 252–254
 - case sensitivity, 198–199
 - character functions, 481–484
 - comparison operators, 477–478
 - compound queries, 487–490
 - conjunctive operators, 480
 - creating schemas, 393–394
 - creating tables from existing tables, 490
 - creating tables/data, 46–48, 468–470, 471–474
 - creating users, 392–393
 - creating views, 491–493
 - creating views from subqueries, 493–494
 - data included, 30–31
 - data types in, 123–124
 - database design process, 64–71
 - describing tables, 470–471
 - embedded subqueries, 493
 - entities and attributes in, 34–35
 - ERD for, 32, 66–67
 - expansion opportunities, 31–32
 - file locations, 39
 - first normal form (FNF), 98, 107–108
 - generating SQL code, 494–495
 - GROUP BY clause, 486
 - HAVING clause, 486
 - inner joins, 239–240
 - inserting data into tables, 491
 - joins, 474–477
 - list of tables/data, 49–56
 - logical operators, 478–480
 - manipulating data, 472–474
 - naming conventions, 105
 - non-equijoins, 242–243
 - outer joins, 244–245
 - resetting to original, 58
 - second normal form (SNF), 100–101, 108
 - SELECT command, 471

- third normal form (TNF), 102–104, 108–109
 - users of, 31
 - connections to, 21–22, 397
 - denormalization, 106–107, 111
 - designing. *See* database design
 - environments, 5
 - client/server, 5–6, 15
 - integration across, 7
 - web-based, 6
 - naming standards, 35–36
 - purpose of, 3–4
 - raw, 94
 - relational
 - advantages of, 13
 - logical versus physical elements, 11–12
 - purpose of, 7
 - retrieving data from multiple tables, 11
 - schemas, 12–13
 - structure of, 8–11
 - reversing changes. *See* transactional control commands
 - security. *See* security
 - software
 - downloading/installing, 40–43
 - list of, 43–44
 - users. *See* users
- date and time**
- converting, 290
 - date pictures, 290–293
 - to strings, 294
 - from strings, 294–295
 - data types, 119–120, 280–281
 - DATETIME elements, 280
 - leap years, 281
 - default year format, 282
 - functions
 - adding time, 286–288
 - current date, 282–285
 - list of, 289–290
 - purpose of, 281–282
 - subtracting time, 288–289
 - time zones, 285–286
 - vendor implementations, 279, 281, 291–293, 296
- DATE data type, 119, 280**
- date parts, 291–293**
- date pictures, 290–293**
- DATEDIFF function, 289**
- DATENAME function, 289**
- DATEPART function, 289**
- DATETIME data type, 119**
- elements of, 280
 - leap years, 281
 - vendor implementations, 281
- DAYNAME function, 290**
- DAYOFMONTH function, 290**
- DAYOFWEEK function, 290**
- DAYOFYEAR function, 290**
- DBAs (database administrators), 46, 105–106, 387**
- DBMSs (database management systems), 17**
- DCL (Data Control Language), 24**
- DDL (Data Definition Language), 22**
- debugging subqueries, 335**
- decimal data types, 118**
- DECODE function, 264–265**
- default schemas, 394**
- default year format, 282**
- DELETE command, 162–163, 329–330, 501**
- deleting. *See also* dropping**
- data, 47–48
 - records, 162–163
 - savepoints, 180–181
- denormalization, 106–107, 111**
- deploying database design, 75**
- DESC command, 470–471**
- descending order, 195, 201, 207**
- describing tables, 470–471**
- designing databases. *See* database design**
- development environment (in database life cycle), 72**
- disabling**
- constraints, 150
 - indexes during batch loads, 438
- DISCONNECT command, 22, 397**
- DISTINCT keyword, 191–192, 207**
- aggregate functions and, 305
 - in AVG function, 303
 - in COUNT function, 300–301
 - in MAX function, 304
 - in MIN function, 305
 - in SUM function, 302–303
- division operator, 231–232**
- DML (Data Manipulation Language), 23, 153–154, 472–474**
- DELETE command, 162–163, 329–330
 - INSERT command, 165

- for columns, 156
- committing, 183
- NULL values, 158
- with queries, 156–158
- subqueries and, 326–327
- for tables, 154–155
- as transactions, 167
- UPDATE command, 158
 - after COMMIT command, 183
 - in multiple columns, 161–162
 - in single column, 158–161
 - subqueries and, 328–329
- domains, 122
- double pipe (| |), concatenation, 261
- DOUBLE PRECISION data type, 119
- downloading Oracle, 40–43
- DQL (Data Query Language), 23–24. *See also* SELECT command
- DROP INDEX command, 423, 501
- DROP ROLE command, 407–408
- DROP SCHEMA command, 395
- DROP SYNONYM command, 381
- DROP TABLE command, 145–146, 501
- DROP USER command, 398, 501
- DROP VIEW command, 379, 501
- dropping
 - columns, 137
 - constraints, 150
 - indexes, 423
 - roles, 407–408
 - schemas, 395

- synonyms, 381
- tables, 47, 145–146, 151
 - in views, 360, 382
- users, 404

E

- embedded functions, resolving, 275
- embedded subqueries, 330–332, 335, 493
- enabling constraints, 150
- end-user needs in database design, 111
- entities. *See also* tables
 - adding, 37
 - attributes and, 33
 - in BIRDS example database, 34–35, 66–67
 - groups of data, creating, 69
 - lists of fields, creating, 69–71
 - relationships and, 32–33, 37
 - creating ERDs, 87–89
 - first normal form (FNF), 97–98, 107–108
 - purpose of, 78–79
 - referential integrity, 79–81, 84–87
 - second normal form (SNF), 98–101, 108
 - third normal form (TNF), 102–104, 108–109
- entity relationship diagrams. *See* ERDs (entity relationship diagrams)

- environments (database), 5
 - client/server, 5–6, 15
 - integration across, 7
 - web-based, 6
- equality operator, 210
- equijoins, 239–240
- ERDs (entity relationship diagrams)
 - attributes and entities, 33
 - for BIRDS example database, 32, 66–67, 87–89
 - purpose of, 32
 - types of relationships, 81–82
- EXCEPT operator, 347–348, 354
- exclamation mark (!), negating equality operator, 225
- existing tables, creating new tables from, 141–144, 490
- EXISTS operator, 219
- EXIT command, 22
- EXP function, 272
- expansion opportunities for BIRDS example database, 31–32
- EXPLAIN PLAN command, 440
- explicit conversion, 114

F

- field definition, 114
- fields, 7, 8
 - as alphanumeric, 126
 - creating lists of, 69–71
 - data types, 114, 132–133
 - NOT NULL in, 121

file locations for BIRDS example database, 39

first normal form (FNF), 97–98, 107–108

fixed-length strings, 115–116, 126

floating-point decimal data types, 119

FLOOR function, 272

FOOD table

- base table joins, 250–252
- Cartesian products, 252–254
- columns/data types, 123
- creating views, 365–367
- data in, 50–51
- HAVING clause, 315–316
- selecting all records, 199–200

foreign keys, 10–11, 237

- as constraints, 148
- identifying, 86–87
- multiple associated with primary keys, 90

forgotten password recovery, 456

formatting commands for readability, 429–431

fractions of seconds, 120

free database software, 21

FROM clause, 192, 207, 431, 503

front-end tools, 95

full outer joins, 243

full table scans, 434–435

functions

- aggregate, 484–486
 - AVG, 303–304
 - COUNT, 202–204, 300–302, 452
 - DISTINCT keyword and, 305

- GROUP BY clause and, 308–309
- input for, 317
- MAX, 304–305
- MIN, 305–306
- purpose of, 299
- SUM, 302–303

ANSI standard, 276

character, 481–484

- ASCII, 271
- COALESCE, 269–270
- combinations of, 275
- CONCAT, 261–263
- DECODE, 264–265
- LENGTH, 268–269
- LOWER, 263–264
- LPAD, 270
- LTRIM, 267
- purpose of, 259–260
- REPLACE, 266–267
- RPAD, 271
- RTRIM, 268
- SUBSTR, 265
- TRANSLATE, 266
- UPPER, 263

conversion

- purpose of, 272
- TO_CHAR, 115, 273
- TO_DATE, 294–295
- TO_NUMBER, 273–275

date and time

- adding time, 286–288
- current date, 282–285
- list of, 289–290
- purpose of, 281–282
- subtracting time, 288–289

- embedded, resolving, 275
- mathematical, 272
- purpose of, 276, 299

G

gathering data (database design process), 68

generating SQL code from commands, 494–495

GETDATE function, 282, 289

grand totals, 312–313

GRANT command, 389–390, 402–403, 501

GRANT OPTION command, 403

granting privileges, 402–404

- ADMIN OPTION command, 403
- authority for, 402
- automatically, 401
- GRANT command, 402–403
- GRANT OPTION command, 403
- in MySQL, 391
- in Oracle, 390
- partial, 410
- in SQL Server, 391

graphical user interfaces (GUIs), 389, 398

greater than operator, 212–213

GROUP BY clause, 306–312, 486, 503

- with aggregate functions, 308–309
- columns in, 317
- compound queries, 350–353, 354

creating views, 374
 CUBE expression, 313–314
 ORDER BY clause versus,
 309–312, 317
 ROLLUP expression, 312–313

grouping

data, 306
 with aggregate functions,
 307, 308–309
 with GROUP BY clause,
 306–307
 privileges, 405–406. *See also*
 roles
 queries, 224

**groups of data (in database
 design process), creating, 69**

**GUIs (graphical user interfaces),
 389, 398**

H

**HAVING clause, 315–316,
 351–353, 437, 486, 504**

history of SQL, 18

I

IBM, 18

implementations of SQL

for book, 21
 choosing, 21
 list of, 20
 portability between, 26

**implicit conversion, 114, 120,
 126, 274**

implicit indexes, 420

IN operator, 217–218

database performance,
 436–437
 with subqueries, 325

indexes

altering, 422–423
 creating, 415
 disabling during batch loads,
 438
 dropping, 423
 full table scans versus,
 434–435
 how they work, 414–415
 most restrictive condition, 434
 planning, 420
 purpose of, 413–414
 re-creating, 424
 storage considerations, 424
 types of

composite, 419
 implicit, 420
 single-column, 416–418
 unique, 418–419, 424
 when to avoid, 421–422
 when to use, 420

**inequality, testing for, 211–212,
 225**

information, data and, 2, 15

initiating transactions, 181

inner joins, 239–240

INSERT command, 165, 502

for columns, 156
 committing, 183
 NULL values, 158
 with queries, 156–158, 502

subqueries and, 326–327
 for tables, 154–155

inserting data, 165

into columns, 156
 NULL values, 158
 from queries, 156–158
 into tables, 154–155, 491

installing Oracle, 40–43

integer data types, 118–119

**integers in ORDER BY clause,
 196, 201–202**

**integrity constraints. *See*
 constraints**

INTERSECT operator, 346–347

IS NOT NULL operator, 227–228

IS NULL operator, 214–216

J

joins, 238–239, 474–477

with base tables, 250–252,
 256
 Cartesian products, 252–255
 database performance, 250,
 431–432
 inner, 239–240
 on multiple columns, 256
 on multiple keys, 249–250
 non-equi joins, 241–243
 order of, 256
 outer, 243–245, 476–477
 self, 246–249, 463, 476
 table aliases, 241

justification, 273

K**keys, 7**

- foreign, 10–11, 237
 - as constraints, 148
 - identifying, 86–87
 - multiple associated with primary keys, 90
- joins on multiple, 249–250
- natural, 97
- primary, 10, 237
 - composite, 86
 - as constraints, 146–147
 - first normal form (FNF), 97–98, 107–108
 - identifying, 85–86
 - multiple foreign keys with, 90
 - second normal form (SNF), 98–101, 108
 - third normal form (TNF), 102–104, 108–109

knowing data (database design process), 65–68**L**

- LAN (local area network), 5**
- large object types, 117**
- LCASE function, 264**
- leap seconds, 120, 280**
- leap years, 281**
- left outer joins, 243**

LEN function, 269**LENGTH function, 268–269****less than operator, 212–213****LIKE operator, 218–219, 435–436****lists**

- of fields, creating, 69–71
- in SQL statements, 189

literal strings, 120, 260**local area network (LAN), 5****LOCATIONS table**

- columns/data types, 123
- data in, 56
- SUBSTR function, 265

LOCATIONS2 table

- creating, 173
- rolling back transactions, 173–176

logical database design.**See database design****logical database elements, 11–12, 71–72, 113****logical operators, 214, 478–480**

- ALL, 220
- ANY, 220–221
- BETWEEN, 216–217
- combinations of, 223
- EXISTS, 219
- IN, 217–218, 436–437
- IS NULL, 214–216
- LIKE, 218–219, 435–436
- negating, 225

LOWER function, 263–264**LPAD function, 270****LTRIM function, 267****M****managing users. See users****mandatory columns, adding, 139–140****manual database design, 75****manual population of data, 154****many-to-many relationships, 83****mathematical functions, 272****MAX function, 304–305****methodology for database design, choosing, 63–64****Microsoft SQL Server. See SQL Server****MIGRATION table**

- columns/data types, 124
- committing transactions, 169–170
- compound queries
 - GROUP BY clause, 351–353
 - UNION operator, 339–340
- creating tables from existing tables, 144
- creating views, 367–368, 370–371
- CUBE expression, 314
- data in, 53
- IN operator, 217–218
- LIKE operator, 218
- LOWER function, 264
- NOT IN operator, 226
- ROLLUP expression, 312–313
- updating single column, 159
- UPPER function, 263

MIGRATION_LOCATIONS table
 GROUP BY versus ORDER BY clauses, 309–312
 grouping data, 308
 literal strings, 260
 LPAD function, 270
 REPLACE function, 267
 RPAD function, 271

MIN function, 305–306

MINUS operator, 347–348, 354

minus sign (-) operator, 229–230

modeling data (database design process), 68–71
 groups of data, creating, 69
 lists of fields, creating, 69–71

modifying tables, 137–141
 adding columns, 138
 attributes, 138–139
 autoincrementing columns, 140–141
 dropping columns, 137
 factors in, 141
 mandatory columns, 139–140

monitoring user sessions, 397

MONTHS_BETWEEN function, 290

most restrictive condition, 432–434

multiple columns
 joins on, 256
 updating, 161–162

multiple foreign keys, associated with primary keys, 90

multiple keys, joins on, 249–250

multiple operators, 234

multiple queries, combining. See compound queries

multiple tables
 creating views from, 364–368
 selecting data from, 11, 237–238. *See also* joins

multiplication operator, 230–231

MySQL
 CREATE DATABASE command, 395
 date and time, 281, 290
 date parts, 293
 LCASE function, 264
 NOW function, 282
 roles, lack of, 407
 UCASE function, 263
 users
 altering, 396
 creating, 390, 391

N

naming standards
 for columns, 132
 for database objects, 136–137
 for savepoints, 178
 for synonyms, 382
 for system catalog, 446
 for tables, 35–36, 104–105, 151
 vendor implementations, 131, 134

natural keys, 97

negative operators, 225
 IS NOT NULL, 227–228
 NOT BETWEEN, 226
 NOT EQUAL, 225
 NOT EXISTS, 228
 NOT IN, 226
 NOT LIKE, 227

nested queries. See subqueries

nested views, 379

NESTS table
 columns/data types, 124
 data in, 54

NEXT_DAY function, 290

NICKNAMES table
 columns/data types, 123
 concatenation, 262–263
 correlated subqueries, 333
 data in, 55–56
 dropping, 145
 foreign key constraints, 148
 inner joins, 239–240
 LIKE operator, 218–219
 non-equijoins, 242–243
 outer joins, 244–245
 primary key constraints, 147

non-equality operator, 211–212, 225

non-equijoins, 241–243

normal forms, 111
 first normal form (FNF), 97–98, 107–108
 purpose of, 96

- second normal form (SNF), 98–101, 108
 - third normal form (TNF), 102–104, 107, 108–109
 - normalization, 237–238**
 - benefits of, 105–106
 - denormalization versus, 111
 - drawbacks of, 106
 - naming conventions, 104–105
 - normal forms, 111
 - first normal form (FNF), 97–98, 107–108
 - purpose of, 96
 - second normal form (SNF), 98–101, 108
 - third normal form (TNF), 102–104, 107, 108–109
 - purpose of, 94
 - raw databases, 94
 - redundancy, 95–96
 - NOT BETWEEN operator, 226**
 - NOT EQUAL operator, 225**
 - NOT EXISTS operator, 228**
 - NOT IN operator, 226**
 - NOT LIKE operator, 227**
 - NOT NULL value**
 - adding mandatory columns, 139–140
 - in columns/fields, 121, 133
 - as constraint, 149
 - creating tables, 140
 - IS NOT NULL operator, 227–228
 - NOT operator, 225**
 - NOW function, 282**
 - NULL values, 11**
 - COALESCE function, 269–270
 - in columns/fields, 133
 - creating tables, 140
 - with data types, 121
 - inserting, 158
 - IS NULL operator, 214–216
 - MAX function and, 317
 - MIN function and, 317
 - NUMERIC data type, 117**
 - numeric data types, 117–118, 126, 234**
 - numeric sorts, 196**
 - numeric values**
 - converting characters from, 273–275
 - converting to characters, 273
- O**
- object privileges, 401**
 - one-to-many relationships, 82–83**
 - one-to-one relationships, 82**
 - operators**
 - arithmetic, 228–229, 480–481
 - addition, 229
 - combinations of, 232
 - division, 231–232
 - multiplication, 230–231
 - subtraction, 230
 - comparison, 210, 477–478
 - combinations of, 213–214, 223
 - equality, 210
 - greater than, 212–213
 - less than, 212–213
 - non-equality, 211–212
 - conjunctive, 221, 480
 - AND, 221–222
 - OR, 222–224, 342–343, 436–437
 - logical, 214, 478–480
 - ALL, 220
 - ANY, 220–221
 - BETWEEN, 216–217
 - combinations of, 223
 - EXISTS, 219
 - IN, 217–218, 436–437
 - IS NULL, 214–216
 - LIKE, 218–219, 435–436
 - negating, 225
 - multiple, 234
 - negative, 225
 - IS NOT NULL, 227–228
 - NOT BETWEEN, 226
 - NOT EQUAL, 225
 - NOT EXISTS, 228
 - NOT IN, 226
 - NOT LIKE, 227
 - purpose of, 192, 209
 - set
 - EXCEPT/MINUS, 347–348, 354
 - INTERSECT, 346–347
 - UNION, 337–343, 354
 - UNION ALL, 343–346
 - optimization. See database performance**

OR operator, 222–224, 342–343, 436–437

Oracle

- ADD_MONTH function, 287
 - for BIRDS example database, 13, 21, 36
 - CONCAT function, 261–262
 - date and time, 281, 290
 - date parts, 291–293
 - downloading/installing, 40–43
 - history of SQL, 18
 - MINUS operator, 347–348
 - outer joins, 243
 - performance tools, 440, 441
 - recommendation for, 44, 505
 - requirement for book exercises, 58
 - roles, 405–406
 - rolling back to savepoint, 178
 - SQL*Loader utility, 165
 - SUBSTR function, 265
 - SYSDATE function, 282
 - system catalog tables, 448–449
 - system privileges, 400
 - users
 - altering, 396
 - creating, 46, 389–390
- ORDER BY clause, 195–197, 207, 504**
- compound queries, 348–350
 - creating views, 373–374
 - descending order, 207
 - GROUP BY clause versus, 309–312, 317

order of precedence, 232
organizations, data and, 3–4
outer joins, 243–245, 476–477

P

parent entities, 79–81

parentheses, grouping queries, 224

parsing, 437

partial privileges, granting, 410

passwords

- changing, 410
- recovering, 456

performance. See database performance

performance statistics in system catalog, 448

permissions. See privileges

PHOTOGRAPHERS table

- adding time to dates, 286–288
- converting dates to strings, 294
- COUNT function, 301–302
- creating, 468–470
- current date, 283–285
- data in, 463–467
- outer joins, 476–477
- self joins, 246–249, 463, 476
- subtracting time, 288–289

PHOTOS table

- columns/data types, 123
- data in, 56

physical database elements, 11–12, 71–72, 113

planning indexes, 420

plus sign (+)

- addition operator, 229
- concatenation, 261
- outer joins, 243

populating tables, 154

POWER function, 272

precedence, 232

precision

- in decimal data types, 118
- truncated data, 304

PREDATORS table

- creating, 468–470
- data in, 461–463

predefined queries. See views

primary keys, 10, 237

- composite, 86
- as constraints, 146–147
- first normal form (FNF), 97–98, 107–108
- identifying, 85–86
- multiple foreign keys with, 90
- second normal form (SNF), 98–101, 108
- third normal form (TNF), 102–104, 108–109

PRIVATE synonyms, 380

privileges

- on columns, 404
- granting, 402–404
 - ADMIN OPTION command, 403
 - authority for, 402

- automatically, 401
- GRANT command, 402–403
- GRANT OPTION command, 403
- in MySQL, 391
- in Oracle, 390
- partial, 410
- in SQL Server, 391
- grouping, 405–406
- object, 401
- PUBLIC database account, 405
- purpose of, 388, 399, 402
- revoking, 397–398, 402, 404
- system, 400
- production environment (in database life cycle), 74**
- PUBLIC database account, 405**
- PUBLIC synonyms, 380**

Q

- qualifying columns, 240**
- queries. See also SELECT command; subqueries**
 - compound, 487–490
 - data accuracy, 353
 - data types and column lengths, 354
 - EXCEPT/MINUS operators, 347–348, 354
 - GROUP BY clause, 350–353, 354
 - INTERSECT operator, 346–347

- ORDER BY clause, 348–350
 - single queries versus, 337–338
- UNION ALL operator, 343–346
- UNION operator, 338–343, 354
- conditions in, 194
- constraining, 202
- examples of, 199–202
- grouping, 224
- inserting data from, 156–158, 491, 502
- literal strings in, 260
- predefined. *See* views
- purpose of, 23, 156–157, 187–188
- to system catalog, 449–454
- tuning. *See* database performance

Query Analyzer, 440

- quotation marks (')**
 - concatenation, 262
 - inserting data, 155
 - for numeric data types, 234

R

- raw databases, 94**
- RDMSs (relational database management systems), 17**
- readability, formatting commands for, 429–431**
- REAL data type, 119**

records, 8, 133–134. *See also* data

- counting, 202–204
- deleting, 162–163
- distinct values, 191–192
- relationships
 - many-to-many, 83
 - one-to-many, 82–83
 - one-to-one, 82
 - recursive, 84
 - selecting all, 190–191, 199–200
- recovering passwords, 456**
- re-creating indexes, 424**
- recursive relationships, 84**
- redundancy**
 - in denormalization, 106–107
 - reducing, 95–96
- referential integrity, 9, 79–81, 84–87**
 - check constraints, 149
 - constraints and, 90, 146
 - dropping constraints, 150
 - foreign keys
 - as constraints, 148
 - identifying, 86–87
 - in logical database design, 85
 - NOT NULL value, 149
 - primary keys
 - as constraints, 146–147
 - identifying, 85–86
 - unique constraints, 147–148
- relational database management systems (RDMSs), 17**
- relational databases**
 - advantages of, 13

- connections to, 21–22
 - designing. *See* database design
 - logical versus physical elements, 11–12
 - purpose of, 7
 - retrieving data from multiple tables, 11
 - schemas, 12–13
 - structure of, 8
 - columns, 9
 - fields, 8
 - foreign keys, 10–11
 - NULL values, 11
 - primary keys, 10
 - records, 8
 - referential integrity, 9
 - relationships. *See also* relational databases**
 - in BIRDS example database, 66–67
 - entities and, 32–33, 37
 - creating ERDs, 87–89
 - first normal form (FNF), 97–98, 107–108
 - purpose of, 78–79
 - referential integrity, 79–81, 84–87
 - second normal form (SNF), 98–101, 108
 - third normal form (TNF), 102–104, 108–109
 - types of, 81–82
 - many-to-many, 83
 - one-to-many, 82–83
 - one-to-one, 82
 - recursive, 84
 - RELEASE SAVEPOINT command, 180–181**
 - removing. *See* dropping; revoking
 - renaming columns temporarily, 204–206, 207
 - REPLACE function, 266–267**
 - requirements gathering (database design process), 68
 - rescues. *See* bird rescue information
 - resetting to original database, 58
 - resolving embedded functions, 275
 - restricting column access, 361
 - retrieving
 - current date, 282–285
 - data from multiple tables, 11, 237–238. *See also* joins
 - REVOKE command, 398, 404, 502**
 - revoking privileges, 397–398, 402, 404
 - right outer joins, 243
 - ROLE_TAB_PRIVS table, accessing, 453–454**
 - roles
 - creating, 407, 453–454
 - dropping, 407–408
 - in Oracle, 405–406
 - purpose of, 406–407
 - setting per session, 408
 - ROLLBACK command, 171–176, 183, 502**
 - ROLLBACK TO SAVEPOINT command, 177–180**
 - rolling back transactions, 171–176, 183
 - to savepoint, 177–180
 - ROLLUP expression, 312–313**
 - ROUND function, 272**
 - rows of data. *See* records
 - RPAD function, 271**
 - RTRIM function, 268**
- ## S
- SAVE TRANSACTION command, 177**
 - SAVEPOINT command, 176–177, 502**
 - savepoints
 - deleting, 180–181
 - naming standards, 178
 - purpose of, 176–177
 - rolling back to, 177–180
 - scale in decimal data types, 118
 - schema owners, 129–130, 388
 - schemas, 12–13
 - creating, 393–395
 - database objects and, 129–131
 - default, 394
 - dropping, 395
 - dropping tables, 151
 - users versus, 388
 - second normal form (SNF), 98–101, 108**
 - seconds, fractions of, 120**
 - security. *See also* users
 - passwords, 410
 - privileges
 - authority for granting/revoking, 402
 - on columns, 404

- granting, 390, 391, 402–404
- granting automatically, 401
- granting partial, 410
- grouping, 405–406
- object privileges, 401
- PUBLIC database account, 405
- purpose of, 388, 399, 402
- revoking, 397–398, 404
- system privileges, 400
- roles
 - creating, 407
 - dropping, 407–408
 - purpose of, 406–407
 - setting per session, 408
- user management and, 398–399
- views for, 361
- security information in system catalog, 448**
- security officers, 387**
- SELECT clause, 188–191, 503**
- SELECT command, 23–24, 115, 471. 502. See also joins**
 - asterisk (*) in, 144
 - case sensitivity of data, 197–199
 - FROM clause, 192, 207, 503
 - column aliases, 204–206, 207
 - COUNT function, 202–204
 - with CREATE TABLE command, 141–144
 - DISTINCT keyword, 191–192, 207
 - with DML commands, 153–154
 - examples of queries, 199–202
 - GROUP BY clause, 306–312, 503
 - with aggregate functions, 308–309
 - columns in, 317
 - compound queries, 350–353, 354
 - CUBE expression, 313–314
 - ORDER BY clause versus, 309–312, 317
 - ROLLUP expression, 312–313
 - HAVING clause, 315–316, 351–353, 437, 504
 - with INSERT command, 156–158
 - multiple. See compound queries
 - ORDER BY clause, 195–197, 207, 504
 - compound queries, 348–350
 - descending order, 207
 - GROUP BY clause versus, 309–312, 317
 - order of operation for clauses, 207
 - on other users' tables, 204
 - purpose of, 187–188
 - SELECT clause, 188–191, 503
 - subqueries and, 323–325
 - WHERE clause, 192–194, 503. See also operators
- selecting. See choosing; SELECT command**
- self joins, 246–249, 463, 476**
- semicolon (;) in SQL statements, 135**
- SERIAL method, 140–141**
- sessions of SQL, 21–22, 397**
- SET CONNECTION command, 397**
- SET IMPLICIT_TRANSACTIONS command, 170–171**
- SET keyword, 162**
- set operators**
 - EXCEPT/MINUS, 347–348, 354
 - INTERSECT, 346–347
 - UNION, 337–343, 354
 - UNION ALL, 343–346
- SET ROLE command, 408**
- SET TRANSACTION command, 181**
- SHORT_BIRDS table**
 - creating from existing table, 157–158
 - deleting records, 162–163
 - updating multiple columns, 161–162
 - updating single column, 160
- SHOW USER command, 449–450**
- SIGN function, 272**
- SIN function, 272**
- single queries, compound queries versus, 337–338**
- single quotation marks ('), inserting data, 155**
- single-column indexes, 416–418**
- slash (/) division operator, 231–232**
- SMALL_BIRDS table**
 - addition operator, 229
 - creating, 189

- creating views, 369–372
 - division operator, 231–232
 - dropping views, 379
 - multiplication operator, 230–231
 - selecting all records, 190–191
 - subtraction operator, 229–230
 - updating, 374–376
 - WITH CHECK option, 376–378
 - SMALLEST_BIRDS table, creating, 373**
 - software**
 - downloading/installing, 40–43
 - list of, 43–44
 - SOME operator, 220**
 - sorting**
 - data, 195–197, 200–202, 207, 437
 - rules for, 196
 - SP_ADDLOGIN command, 391**
 - SP_ADDUSER command, 391**
 - SQL (Structured Query Language)**
 - command-line prompt, 45
 - commands. *See also* commands
 - case sensitivity, 136
 - data administration, 24
 - DCL (Data Control Language), 24
 - DDL (Data Definition Language), 22–23
 - DML (Data Manipulation Language), 23, 153–154, 472–474
 - DQL (Data Query Language), 23–24. *See also* SELECT command
 - formatting for readability, 429–431
 - generating SQL code from, 494–495
 - transactional control, 24–25, 165, 168
 - tuning. *See* database performance
 - types of, 22
 - data types
 - in BIRDS example database, 123–124
 - Boolean values, 121
 - date and time, 119–120, 280–281
 - decimal, 118
 - domains, 122
 - explained, 114–115
 - fixed-length strings, 115–116, 126
 - floating-point decimal, 119
 - integer, 118–119
 - large object types, 117
 - length limits, 126
 - literal strings, 120
 - NULL values, 121
 - numeric, 117–118, 126
 - purpose of, 114
 - user-defined, 122
 - varying-length strings, 116, 126
 - vendor implementations, 121
 - database design and, 61, 63
 - functions. *See* functions
 - history of, 18
 - as nonprocedural language, 18
 - purpose of, 2, 13, 15
 - referential integrity, 80
 - sessions, 21–22, 397
 - standards
 - ANSI standard, 18–19, 36
 - importance of, 20
 - vendor implementations, 20–21
 - users. *See* users
- SQL Server**
 - committing transactions, 170–171
 - CONCAT function, 261–262
 - date and time, 281, 289
 - date parts, 291
 - deleting savepoints, 181
 - GETDATE function, 282
 - LEN function, 269
 - permission groups, 406
 - Query Analyzer, 440
 - rolling back to savepoint, 179
 - saving transactions, 177
 - SUBSTR function, 265
 - system privileges, 400
 - users, creating, 390–391
 - SQL*Loader utility, 165**
 - SQRT function, 272**
 - SSIS (SQL Server Integration Services) utility, 165**
 - standards for SQL**
 - ANSI standard, 18–19, 36
 - importance of, 20
 - vendor implementations
 - for book, 21
 - choosing, 21
 - list of, 20
 - portability between, 26

START command, 46

statements. See commands

stored procedures, 437

strings. See also character

functions

- in columns/fields, 132–133
- concatenation, 260
- converting dates to, 294
- converting numbers to, 273
- converting to dates, 294–295
- converting to numbers, 273–275
- fixed-length, 115–116, 126
- large object types, 117
- literal, 120, 260
- numeric values in, 126
- substrings, 260
- varying-length, 116, 126

Structured Query Language.

See SQL (Structured Query Language)

subqueries

- correlated, 333
- with CREATE TABLE command, 325–326
- creating views from, 493–494
- database performance, 332
- debugging, 335
- with DELETE command, 329–330
- embedded, 330–332, 335, 493
- with INSERT command, 326–327
- purpose of, 321
- rules for, 321–322

- with SELECT command, 323–325
- syntax, 322–323
- for unknown values, 324
- with UPDATE command, 328–329

SUBSTR function, 265

substrings, 260, 265

subtotals, 312–313

subtracting time, 288–289

subtraction operator, 230

SUM function, 302–303

summarized data, views for, 361

super-aggregate rows, 312–313

suspending users, 410

synonyms

- creating, 380–381
- dropping, 381
- naming standards, 382
- purpose of, 380
- vendor implementations, 380

SYSDATE function, 282

system analysts, 387

system catalog

- columns in, 456
- contents of, 447
 - database design, 448
 - performance statistics, 448
 - security information, 448
 - user data, 448
- creating, 446–447
- direct manipulation, 452
- forgotten password recovery, 456
- naming standards, 446

purpose of, 445–446

queries to, 449–454

tables by vendor

implementation, 448–449

updating, 454–455

user access information, 456

system date, retrieving, 282–285

system privileges, 400

T

tables, 131. See also entities

- aliases, 241
- in BIRDS example database, 49–56
- in FROM clause, 431
- creating, 46–48, 134–136, 468–470, 471–474
 - from existing tables, 141–144, 189–191, 490
 - from views, 372–373
- deleting records, 162–163
- describing, 470–471
- distinct values, 191–192
- dropping, 47, 145–146, 151
 - in views, 360, 382
- full table scans, 434–435
- indexes
 - altering, 422–423
 - composite, 419
 - creating, 415
 - dropping, 423
 - how they work, 414–415
 - implicit, 420
 - planning, 420

- purpose of, 413–414
- re-creating, 424
- single-column, 416–418
- storage considerations, 424
- unique, 418–419, 424
- when to avoid, 421–422
- when to use, 420
- inserting data, 154–155, 165
 - into columns, 156
 - from queries, 156–158, 491
- joins, 238–239
 - with base tables, 250–252, 256
 - Cartesian products, 252–255
 - database performance, 250
 - inner, 239–240
 - on multiple columns, 256
 - on multiple keys, 249–250
 - non-equijoins, 241–243
 - order of, 256
 - outer, 243–245
 - self, 246–249
- modifying, 137–141
 - adding columns, 138
 - attributes, 138–139
 - autoincrementing columns, 140–141
 - dropping columns, 137
 - factors in, 141
 - mandatory columns, 139–140
- naming standards, 35–36, 104–105, 151
- populating, 154
- purpose of, 7
- relationships
 - many-to-many, 83
 - one-to-many, 82–83
 - one-to-one, 82
 - recursive, 84
- schemas and, 129–131
- selecting data from multiple, 11, 237–238. *See also* joins
- selecting from other users, 204
- structure of, 8
 - columns, 9, 132–133
 - fields, 8
 - foreign keys, 10–11
 - NULL values, 11
 - primary keys, 10
 - records, 8, 133–134
 - referential integrity, 9
- in system catalog, 448–449
- updating, 158
 - in multiple columns, 161–162
 - in single column, 158–161
- views versus, 373
- virtual. *See* views
- tablespaces, 390**
- TAN function, 272**
- test environment (in database life cycle), 73**
- TEXT data type, 117**
- third normal form (TNF), 102–104, 107, 108–109, 111**
- time. *See* date and time**
- TIME data type, 119, 280**
- time zones, 285–286**
- TIMESTAMP data type, 119, 280**
- TKPROF command, 440**
- TO_CHAR function, 115, 273**
- TO_DATE function, 294–295**
- TO_NUMBER function, 273–275**
- transactional control commands, 24–25, 165, 168**
 - COMMIT, 169–171, 183
 - database performance and, 181
 - RELEASE SAVEPOINT, 180–181
 - ROLLBACK, 171–176, 183
 - ROLLBACK TO SAVEPOINT, 177–180
 - SAVEPOINT, 176–177
 - SET TRANSACTION, 181
- transactions**
 - committing, 169–171, 183
 - database performance and, 181
 - initiating, 181
 - purpose of, 167–168
 - rolling back, 171–176, 183
 - savepoints
 - deleting, 180–181
 - purpose of, 176–177
 - rolling back to, 177–180
 - vendor implementations, 168, 170–171
- TRANSLATE function, 266**
- truncated data, 304**
- tuning commands. *See* database performance**
- types. *See* data types**

U**UCASE function, 263****undoing transactions. See rolling back transactions****UNION ALL operator, 343–346****UNION operator, 337–343, 354****unique constraints, 147–148****unique indexes, 418–419****units of work, 171****UPDATE command, 158, 503**

- after COMMIT command, 183
- in multiple columns, 161–162
- in single column, 158–161
- subqueries and, 328–329
- with views, 374–376

updating

- data, 158
 - in multiple columns, 161–162
 - in single column, 158–161
 - through views, 374–376
- system catalog, 454–455

UPPER function, 263**user accounts. See users****user communities, data and, 3–4****user data in system catalog, 448****USER_TAB_COLUMNS table, accessing, 452–453****USER_TABLES table, accessing, 450–451****user-defined data types, 122****users**

- altering, 396
- of BIRDS example database, 31

creating, 46, 410

- in BIRDS example database, 392–393

in MySQL, 390, 391

in Oracle, 389–390

in SQL Server, 390–391

vendor implementations, 389

dropping, 404

GUI tools, 398

management

- importance of, 385–386
- process of, 388–389
- responsibilities in, 387
- security and, 398–399

monitoring sessions, 397

passwords, 410

privileges

- authority for granting/revoking, 402
- on columns, 404
- granting, 390, 391, 402–404
- granting automatically, 401
- granting partial, 410
- grouping, 405–406
- object privileges, 401
- PUBLIC database account, 405
- purpose of, 388, 399, 402
- revoking, 397–398, 404
- system privileges, 400

roles

- creating, 407
- dropping, 407–408

purpose of, 406–407

setting per session, 408

schemas versus, 388

suspending, 410

types of, 386–387

V**VARBINARY data type, 116****VARCHAR data type, 116****VARCHAR2 data type, 116****varying-length strings, 116, 126****vendor implementations of SQL**

- ALTER TABLE command, 141, 149
- ALTER VIEW command, 362
- for book, 21, 505
- choosing, 21
- connections to databases, 397
- CREATE SCHEMA command, 395
- creating users, 389
 - in MySQL, 390, 391
 - in Oracle, 389–390
 - in SQL Server, 390–391
- data types and, 121
- date and time, 279, 281, 289–290, 291–293, 296
- grouping privileges, 405–406
- implicit conversion, 274
- for indexes, 415
- inequality, testing for, 225
- list of, 20, 505

- naming standards, 131, 134
- portability between, 26
- revoking privileges, 397–398
- setting roles, 408
- synonyms, 380
- system catalog tables, 448–449
- system privileges, 400
- time zones, 286
- for transactions, 168, 170–171

views

- creating, 361–362, 491–493
 - from multiple tables, 364–368
- ORDER BY clause, 373–374
- from single table, 362–363

- from subqueries, 493–494
- tables from, 372–373
- from views, 368–372
 - WITH CHECK option, 376–379
- database performance and, 379, 438
- dropping, 379
- dropping tables in, 360, 382
- purpose of, 359–360
 - maintaining summarized data, 361
 - security, 361
 - simplifying data access, 360–361
- storage of, 382
- tables versus, 373
- updating data through, 374–376

violating constraints, 138

virtual tables. See views

W

WAN (wide area network), 5

Waterbird Photography. See BIRDS example database

web-based environments, 6

WHERE clause. See also operators

- importance of, 161, 163

- most restrictive condition, 432–434

- in SELECT command, 192–194, 503

wide area network (WAN), 5

wildcards, 435–436

WITH CHECK option (views), 376–379

Y

years, default format, 282