

Ben Forta

Sams **Teach Yourself**

Oracle[®] PL/SQL

in **10**
Minutes

SAMS

FREE SAMPLE CHAPTER



SHARE WITH OTHERS

Ben Forta

Sams **Teach Yourself**

Oracle[®] PL/SQL

in **10 Minutes**

SAMS

800 East 96th Street, Indianapolis, Indiana 46240

Sams Teach Yourself Oracle® PL/SQL in 10 Minutes

Copyright © 2016 by Pearson Education, Inc.

All rights reserved. No part of this book shall be reproduced, stored in a retrieval system, or transmitted by any means, electronic, mechanical, photocopying, recording, or otherwise, without written permission from the publisher. 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: 978-0-672-32866-4

ISBN-10: 0-672-32866-6

Library of Congress Control Number: 2015910491

Printed in the United States of America

First Printing September 2015

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 or from the use of the programs accompanying it.

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 international@pearsoned.com.

Acquisitions Editor

Mark Taber

Managing Editor

Kristy Hart

Senior Project Editor

Betsy Gratner

Copy Editor

Paula Lowell

Indexer

Lisa Stumpf

Proofreader

Sarah Kearns

Editorial Assistant

Vanessa Evans

Cover Designer

Mark Shirar

Senior Compositor

Gloria Schurick

Contents

| | |
|--|-----------|
| Introduction | 1 |
| What Is This Book? | 1 |
| Who Is This Book For?..... | 2 |
| Companion Website | 3 |
| Conventions Used in This Book..... | 3 |
| 1 Understanding SQL | 5 |
| Database Basics..... | 5 |
| What Is SQL?..... | 10 |
| Try It Yourself..... | 11 |
| Summary | 12 |
| 2 Getting Started with Oracle and PL/SQL | 13 |
| What Is Oracle? | 13 |
| Getting Set Up | 16 |
| Summary | 20 |
| 3 Working with Oracle | 21 |
| Creating a Working Environment..... | 21 |
| Making the Connection..... | 25 |
| A Quick Introduction to Oracle SQL Developer | 27 |
| Creating and Populating the Example Tables..... | 28 |
| One More Look at Oracle SQL Developer | 32 |
| Summary..... | 32 |

| | | |
|----------|--|-----------|
| 4 | Retrieving Data | 33 |
| | The <code>SELECT</code> Statement | 33 |
| | Retrieving Individual Columns..... | 33 |
| | Retrieving Multiple Columns..... | 36 |
| | Retrieving All Columns | 37 |
| | Retrieving Distinct Rows | 38 |
| | Using Fully Qualified Table Names..... | 40 |
| | Using Comments..... | 40 |
| | Summary | 42 |
| 5 | Sorting Retrieved Data | 43 |
| | Sorting Data | 43 |
| | Sorting by Multiple Columns..... | 45 |
| | Specifying Sort Direction..... | 47 |
| | Summary | 49 |
| 6 | Filtering Data | 51 |
| | Using the <code>WHERE</code> Clause..... | 51 |
| | The <code>WHERE</code> Clause Operators..... | 53 |
| | Summary | 59 |
| 7 | Advanced Data Filtering | 61 |
| | Combining <code>WHERE</code> Clauses..... | 61 |
| | Using the <code>IN</code> Operator | 65 |
| | Using the <code>NOT</code> Operator..... | 67 |
| | Summary | 68 |

| | | |
|-----------|---|------------|
| 8 | Using Wildcard Filtering | 69 |
| | Using the <code>LIKE</code> Operator..... | 69 |
| | Tips for Using Wildcards | 74 |
| | Summary | 74 |
| 9 | Searching Using Regular Expressions | 75 |
| | Understanding Regular Expressions..... | 75 |
| | Using Oracle PL/SQL Regular Expressions | 76 |
| | Summary | 89 |
| 10 | Creating Calculated Fields | 91 |
| | Understanding Calculated Fields..... | 91 |
| | Concatenating Fields | 92 |
| | Performing Mathematical Calculations | 96 |
| | Summary | 98 |
| 11 | Using Data Manipulation Functions | 99 |
| | Understanding Functions | 99 |
| | Using Functions | 100 |
| | Summary | 107 |
| 12 | Summarizing Data | 109 |
| | Using Aggregate Functions..... | 109 |
| | Aggregates on Distinct Values..... | 117 |
| | Combining Aggregate Functions..... | 118 |
| | Summary | 119 |

| | |
|---|------------|
| 13 Grouping Data | 121 |
| Understanding Data Grouping..... | 121 |
| Creating Groups | 122 |
| Filtering Groups..... | 123 |
| Grouping and Sorting..... | 126 |
| SELECT Clause Ordering..... | 129 |
| Summary | 129 |
| 14 Working with Subqueries | 131 |
| Understanding Subqueries | 131 |
| Filtering by Subquery | 131 |
| Using Subqueries as Calculated Fields | 136 |
| Summary | 140 |
| 15 Joining Tables | 141 |
| Understanding Joins | 141 |
| Creating a Join | 144 |
| Summary | 152 |
| 16 Creating Advanced Joins | 153 |
| Using Table Aliases | 153 |
| Using Different Join Types..... | 154 |
| Using Joins with Aggregate Functions | 160 |
| Using Joins and Join Conditions | 161 |
| Summary | 162 |

| | | |
|-----------|--|------------|
| 17 | Combining Queries | 163 |
| | Understanding Combined Queries..... | 163 |
| | Creating Combined Queries..... | 164 |
| | Summary | 169 |
| 18 | Inserting Data | 171 |
| | Understanding Data Insertion..... | 171 |
| | Inserting Complete Rows | 172 |
| | Inserting Retrieved Data | 176 |
| | Summary | 178 |
| 19 | Updating and Deleting Data | 179 |
| | Updating Data..... | 179 |
| | Deleting Data..... | 181 |
| | Guidelines for Updating and Deleting Data..... | 183 |
| | Summary | 184 |
| 20 | Creating and Manipulating Tables | 185 |
| | Creating Tables..... | 185 |
| | Updating Tables | 190 |
| | Deleting Tables | 193 |
| | Renaming Tables | 193 |
| | Summary | 194 |

| | | |
|-----------|---|------------|
| 21 | Using Views | 195 |
| | Understanding Views | 195 |
| | Using Views | 197 |
| | Summary | 204 |
| 22 | Working with Stored Procedures | 205 |
| | Understanding Stored Procedures | 205 |
| | Why Use Stored Procedures | 206 |
| | Using Stored Procedures | 207 |
| | Summary | 214 |
| 23 | Using Cursors | 215 |
| | Understanding Cursors | 215 |
| | Working with Cursors | 216 |
| | Summary | 222 |
| 24 | Using Triggers | 223 |
| | Understanding Triggers | 223 |
| | Creating Triggers | 224 |
| | Dropping Triggers | 225 |
| | Using Triggers | 225 |
| | Summary | 232 |
| 25 | Managing Transaction Processing | 233 |
| | Understanding Transaction Processing..... | 233 |
| | Controlling Transactions | 235 |
| | Summary | 238 |

| | | |
|-----------|--|------------|
| 26 | Managing Security | 239 |
| | Understanding Access Control..... | 239 |
| | Managing Users | 240 |
| | Summary | 244 |
| A | The Example Tables | 245 |
| | Understanding the Sample Tables | 245 |
| B | Oracle PL/SQL Datatypes | 251 |
| | String Datatypes | 252 |
| | Numeric Datatypes..... | 253 |
| C | Oracle PL/SQL Reserved Words and Keywords | 255 |
| | Index | 259 |

About the Author

Ben Forta has three decades of experience in the computer industry in product design and development, support, training, and marketing. As Adobe Inc.'s Senior Director of Education Initiatives, he spends a considerable amount of time teaching, talking, and writing about Adobe products, coding and application development, creativity, and digital literacy and provides feedback to help shape the future direction of Adobe products.

Ben is the author of more than 40 books, including the world's best-selling title on SQL, as well as titles on topics as diverse as Regular Expressions, mobile development, and Adobe ColdFusion. More than 750,000 copies of his books are in print in English, and titles have been translated into fifteen languages. Many of these titles are used as textbooks in colleges and universities worldwide.

Education is Ben's passion. Between writing, lecturing, and in-classroom experience, Ben has dedicated his professional and personal lives to teaching, inspiring, and sharing his love for technology and creativity. He is immensely grateful to have had the opportunity to share with millions worldwide.

Ben is also a successful entrepreneur with experience creating, building, and selling start-ups. He is a sought-after public speaker, a writer, and a blogger, and he presents on education and development topics worldwide.

Acknowledgments

It's been sixteen years since the publication of my first book on SQL, *Sams Teach Yourself SQL in 10 Minutes*. That book was met with such positive feedback that it has been updated three times, has spawned four spin-off titles (the most recent being the book you are reading right now), and has been translated more than a dozen times. In all of its various flavors and iterations, this little book has helped hundreds of thousands learn the basics of SQL. So, first and foremost, thanks to all of you who have trusted me and this book over the years; your support is both incredibly humbling and a source of great pride.

I am blessed with some very vocal and opinionated readers who regularly share ideas, comments, suggestions, and occasionally criticism. These books continue to improve directly in response to that feedback, so thanks, and please keep it coming.

Thanks to the numerous schools and colleges the world over who have made this series part of their curriculum. Seeing students use my writing as part of their studies never ceases to thrill.

And finally, thanks to my partners at Pearson with whom I've now published more than 40 titles, and without whose support none would have seen the light of day. In particular, thanks to Betsy Gratner for shepherding this book through the process, Paula Lowell for her editing help, and Mark Taber for once again patiently and encouragingly supporting whatever I toss his way.

Ben Forta

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: feedback@sampublishing.com

Mail: Sams Publishing
ATTN: Reader Feedback
800 East 96th Street
Indianapolis, IN 46240 USA

Reader Services

Visit our website and register this book at informit.com/register for convenient access to any updates, downloads, or errata that might be available for this book.

Introduction

Oracle Database (or Oracle RDBMS) is so prevalent and well established that most users simply refer to it as “Oracle” (ignoring the fact that Oracle, the company, produces other software, and even hardware). Oracle Database (I’ll do what most do and just call it “Oracle” to simplify things) has been around since the 1970s, making it one of the earliest database management systems. Oracle is one of the most used database management systems (DBMS) in the world. In fact, most surveys rank it as #1 in database use and popularity worldwide, especially among corporate users, and over the years it has proven itself to be a solid, reliable, fast, and trusted solution to all sorts of data storage needs.

That’s the good news. The not-so-good news is that getting started with Oracle can be tricky, especially when compared to some of the alternative DBMSs. Oracle’s power, capabilities, security, and more are an important part of why it is so trusted. But that makes installation, configuration, and even the tooling a little more complex, too. On top of that, Oracle’s implementation of the SQL language, called PL/SQL, tends to differ subtly from other SQL implementations, and this can make using Oracle just a bit trickier.

What Is This Book?

This book is based on my best-selling *Sams Teach Yourself SQL in 10 Minutes*. That book has become one of the most used SQL tutorials in the world, with an emphasis on teaching what you really need to know—methodically, systematically, and simply. However, as popular and as successful as that book is, it does have some limitations:

- ▶ In covering all the major DBMSs, coverage of DBMS-specific features and functionality had to be kept to a minimum.
- ▶ To simplify the SQL taught, the lowest common denominator had to be found—SQL statements that would (as much as possible) work with all major DBMSs. This requirement necessitated that better DBMS-specific solutions not be covered.

- ▶ Although basic SQL tends to be rather portable between DBMSs, more advanced SQL most definitely is not. As such, that book could not cover advanced topics, such as triggers, cursors, stored procedures, access control, transactions, and more, in any real detail.

And that is where this book comes in. *Sams Teach Yourself Oracle PL/SQL in 10 Minutes* builds on the proven tutorials and structure of *Sams Teach Yourself SQL in Ten Minutes*, without getting bogged down with anything but Oracle and PL/SQL. Starting with simple data retrieval and working toward more complex topics, including the use of joins, sub-queries, regular expressions, full text-based searches, stored procedures, cursors, triggers, table constraints, and much more. You'll learn what you need to know methodically, systematically, and simply—in highly focused lessons designed to make you immediately and effortlessly productive.

Who Is This Book For?

This book is for you if

- ▶ You are new to SQL.
- ▶ You are just getting started with Oracle PL/SQL and want to hit the ground running.
- ▶ You want to quickly learn how to get the most out of Oracle and PL/SQL.
- ▶ You want to learn how to use Oracle in your own application development.
- ▶ You want to be productive quickly and easily using Oracle without having to call someone for help.

It is worth noting that this book is not intended for all readers. If you are an experienced SQL user, then you might find the content in this book to be too elementary. However, if the preceding list describes you and your needs relative to Oracle, you'll find *Sams Teach Yourself Oracle PL/SQL in 10 Minutes* to be the fastest and easiest way to get up to speed with Oracle.

Companion Website

This book has a companion website at forta.com/books/0672328666.

Visit the site to

- ▶ Access table creation and population scripts for creating the example tables used throughout this book
- ▶ Visit the online support forum
- ▶ Access online errata (if one might be required)
- ▶ Find other books that might be of interest to you

Conventions Used in This Book

This book uses different typefaces to differentiate between code and regular English, and also to help you identify important concepts.

Text that you type and text that should appear on your screen appears in `monospace type`. It looks like this to mimic the way text looks on your screen.

Placeholders for variables and expressions appear in *monospace italic* font. You should replace the placeholder with the specific value it represents.

This arrow (➡) at the beginning of a line of code means that a single line of code is too long to fit on the printed page. Continue typing all the characters after the ➡ as if they were part of the preceding line.

NOTE

A Note presents interesting pieces of information related to the surrounding discussion.

TIP

A Tip offers advice or teaches an easier way to do something.

CAUTION

A Caution advises you about potential problems and helps you steer clear of disaster.

New Term sidebars provide clear definitions of new, essential terms.

Input ▼

The Input icon identifies code that you can type in yourself. It usually appears by a listing.

Output ▼

The Output icon highlights the output produced by running Oracle PL/SQL code. It usually appears after a listing.

Analysis ▼

The Analysis icon alerts you to the author's line-by-line analysis of input or output.

LESSON 4

Retrieving Data

In this lesson, you'll learn how to use the `SELECT` statement to retrieve one or more columns of data from a table.

The `SELECT` Statement

NOTE: Sample Tables Required

From this point on, all lessons use the sample database tables. If you have yet to install these, please refer to Lesson 3, “Working with Oracle,” before proceeding.

As explained in Lesson 1, “Understanding SQL,” SQL statements are made up of plain English terms. These terms are called *keywords*, and every SQL statement is made up of one or more keywords. The SQL statement you’ll probably use most frequently is the `SELECT` statement. Its purpose is to retrieve information from one or more tables.

To use `SELECT` to retrieve table data, you must, at a minimum, specify two pieces of information—what you want to select, and from where you want to select it.

Retrieving Individual Columns

We’ll start with a simple SQL `SELECT` statement, as follows:

Input ▼

```
SELECT prod_name  
FROM products;
```

TIP: Type Then Execute

By now it should be obvious, but I'll remind you one last time. Type the SQL code in the Oracle SQL Developer Worksheet screen, and then click the Run Script button to execute it. Results appear in a screen below the Worksheet. If you need more room, you can drag and resize all the screens.

Analysis ▼

The previous statement uses the `SELECT` statement to retrieve a single column called `prod_name` from the `products` table. The desired column name is specified right after the `SELECT` keyword, and the `FROM` keyword specifies the name of the table from which to retrieve the data. The following shows the output from this statement:

Output ▼

```
+-----+
| prod_name |
+-----+
| .5 ton anvil |
| 1 ton anvil |
| 2 ton anvil |
| Oil can |
| Fuses |
| Sling |
| TNT (1 stick) |
| TNT (5 sticks) |
| Bird seed |
| Carrots |
| Safe |
| Detonator |
| JetPack 1000 |
| JetPack 2000 |
+-----+
```

NOTE: Unsorted Data

If you tried this query yourself, you might have discovered that the data displayed in a different order than shown here. If this is the case, don't worry—it is working exactly as it is supposed to. If

query results are not explicitly sorted (we'll get to that in the next lesson), data will be returned in no order of any significance. It might be the order in which the data was added to the table, but it might not. As long as your query returned the same number of rows, then it is working.

A simple `SELECT` statement like the one just shown returns all the rows in a table. Data is not filtered (so as to retrieve a subset of the results), nor is it sorted. We'll discuss these topics in the next few lessons.

NOTE: Terminating Statements

Multiple SQL statements must be separated by semicolons (the `;` character). Oracle (like most DBMSs) does not require that a semicolon be specified after single statements. That said, most SQL developers get in the habit of always terminating their SQL statements with semicolons, even when they are not needed.

NOTE: SQL Statements and Case

Note that SQL statements are not case sensitive, so `SELECT` is the same as `select`, which is the same as `Select`. Many SQL developers find that using uppercase for all SQL keywords and lowercase for column and table names makes code easier to read and debug.

However, be aware that while the SQL language is not case sensitive, identifiers (the names of databases, tables, and columns) might be. As a best practice, pick a case convention, and use it consistently.

TIP: Use of White Space

All extra white space within a SQL statement is ignored when that statement is processed. You can specify SQL statements on one long line or break them up over many lines. Most SQL developers find that breaking up statements over multiple lines makes them easier to read and debug.

Retrieving Multiple Columns

To retrieve multiple columns from a table, you use the same `SELECT` statement. The only difference is that you must specify multiple column names after the `SELECT` keyword, and separate each column by a comma.

TIP: Take Care with Commas

When selecting multiple columns, be sure to specify a comma between each column name, but not after the last column name. Doing so generates an error.

The following `SELECT` statement retrieves three columns from the `products` table:

Input ▼

```
SELECT prod_id, prod_name, prod_price
FROM products;
```

Analysis ▼

Just as in the prior example, this statement uses the `SELECT` statement to retrieve data from the `products` table. In this example, three column names are specified, each separated by a comma. The output from this statement is as follows:

Output ▼

| prod_id | prod_name | prod_price |
|---------|----------------|------------|
| ANV01 | .5 ton anvil | 5.99 |
| ANV02 | 1 ton anvil | 9.99 |
| ANV03 | 2 ton anvil | 14.99 |
| OL1 | Oil can | 8.99 |
| FU1 | Fuses | 3.42 |
| SLING | Sling | 4.49 |
| TNT1 | TNT (1 stick) | 2.5 |
| TNT2 | TNT (5 sticks) | 10 |
| FB | Bird seed | 10 |

| | | | |
|---------|--------------|---------|---------|
| FC | Carrots | 2.5 | |
| SAFE | Safe | 50 | |
| DTNTR | Detonator | 13 | |
| JP1000 | JetPack 1000 | 35 | |
| JP2000 | JetPack 2000 | 55 | |
| +-----+ | +-----+ | +-----+ | +-----+ |

NOTE: Presentation of Data

SQL statements typically return raw, unformatted data. Data formatting is a presentation issue, not a retrieval issue. Therefore, presentation (for example, alignment and displaying the price values as currency amounts with the currency symbol and commas) is typically specified in the application that displays the data. Actual raw retrieved data (without application-provided formatting) is rarely displayed as is.

Retrieving All Columns

In addition to being able to specify desired columns (one or more, as shown previously), you can also use `SELECT` statements to request all columns without having to list them individually. This is done using the asterisk (*) wildcard character in lieu of actual column names, as follows:

Input ▼

```
SELECT *  
FROM products;
```

Analysis ▼

When you specify a wildcard (*), all the columns in the table are returned. The columns are in the order in which the columns appear in the table definition. However, you cannot rely on this because changes to table schemas (adding and removing columns, for example) could cause ordering changes.

CAUTION: Using Wildcards

As a rule, you are better off not using the * wildcard unless you really do need every column in the table. Even though use of wildcards might save you the time and effort needed to list the desired columns explicitly, retrieving unnecessary columns usually slows down the performance of your retrieval and your application.

TIP: Retrieving Unknown Columns

There is one big advantage to using wildcards. As you do not explicitly specify column names (because the asterisk retrieves every column), it is possible to retrieve columns whose names are unknown.

Retrieving Distinct Rows

As you have seen, `SELECT` returns all matched rows. But what if you did not want every occurrence of every value? For example, suppose you wanted the vendor ID of all vendors with products in your `products` table:

Input ▼

```
SELECT vend_id
FROM products;
```

Output ▼

```
+-----+
| vend_id |
+-----+
| 1001 |
| 1001 |
| 1001 |
| 1002 |
| 1002 |
| 1003 |
| 1003 |
| 1003 |
| 1003 |
```

```

|      1003 |
|      1003 |
|      1003 |
|      1005 |
|      1005 |
+-----+

```

The `SELECT` statement returned 14 rows (even though only 4 vendors are in that list) because 14 products are listed in the `products` table. So how could you retrieve a list of distinct values?

The solution is to use the `DISTINCT` keyword which, as its name implies, instructs Oracle to only return distinct values:

Input ▼

```

SELECT DISTINCT vend_id
FROM products;

```

Analysis ▼

`SELECT DISTINCT vend_id` tells Oracle to only return distinct (unique) `vend_id` rows, and so only 4 rows are returned, as shown in the following output. If you use it, you must place the `DISTINCT` keyword directly in front of the column names:

Output ▼

```

+-----+
| vend_id |
+-----+
|      1001 |
|      1002 |
|      1003 |
|      1005 |
+-----+

```

CAUTION: **Can't Be Partially** `DISTINCT`

The `DISTINCT` keyword applies to all columns, not just the one it precedes. If you were to specify `SELECT DISTINCT vend_id, prod_price`, all rows would be retrieved unless *both* of the specified columns were distinct.

Using Fully Qualified Table Names

The SQL examples used thus far have referred to columns by just the column names. Referring to columns using fully qualified names (using both the table and column names) is also possible. Look at this example:

Input ▼

```
SELECT products.prod_name
FROM products;
```

This SQL statement is functionally identical to the first one used in this lesson, but here a fully qualified column name is specified.

Table names, too, may be fully qualified, as shown here:

Input ▼

```
SELECT products.prod_name
FROM crashcourse.products;
```

Once again, this statement is functionally identical to the one just used (assuming, of course, that the `products` table is indeed in the `crashcourse` database).

Situations exist where fully qualified names are required, as we will see in later lessons. For now, it is worth noting this syntax so you'll know what it is if you run across it.

Using Comments

As you have seen, SQL statements are instructions that Oracle processes. But what if you wanted to include text that you do not want processed and executed? Why would you ever want to do this? Here are a few reasons:

- ▶ The SQL statements we've been using here are all very short and very simple. But, as your SQL statement grows (in length and complexity), you'll want to include descriptive comments (for your own future reference or for whoever has to work on the project next). You need to embed these comments in the SQL scripts, but they are obviously not intended for Oracle

processing. (For an example of this, see the `create.sql` and `populate.sql` files you used in Lesson 3.)

- ▶ The same is true for headers at the top of a SQL file, perhaps containing the programmer contact information and a description and notes. (You also see this use case in the `create.sql` and `populate.sql` files.)
- ▶ Another important use for comments is to temporarily stop SQL code from being executed. If you were working with a long SQL statement and wanted to test just part of it, you could *comment out* some of the code so that Oracle saw it as comments and ignored it.

Oracle supports two forms of comment syntax. We'll start with inline comments:

Input ▼

```
SELECT prod_name  -- this is a comment
FROM products;
```

Analysis ▼

You may embed comments inline using `--` (two hyphens). Anything after the `--` is considered comment text, making this a good option for describing columns in a `CREATE TABLE` statement, for example.

Here is another form of inline comment:

Input ▼

```
-- This is a comment
SELECT prod_name
FROM products;
```

Analysis ▼

A `--` at the start of a line makes the entire line a comment. You can see this format comment used in the accompanying `create.sql` and `populate.sql` scripts.

You can also create multi-line comments, and comments that stop and start anywhere within the script:

Input ▼

```
/* SELECT prod_name, vend_id  
FROM products; */
```

```
SELECT prod_name  
FROM products;
```

Analysis ▼

`/*` starts a comment, and `*/` ends it. Anything between `/*` and `*/` is comment text. This type of comment is often used to *comment out* code, as shown in this example. Here, two `SELECT` statements are defined, but the first won't execute because it has been commented out.

TIP: Oracle SQL Developer Color Coding

You might have noticed that Oracle SQL Developer color codes your PL/SQL. SQL statements are usually displayed in blue, identifiers (like table and column names) are in black, and so on. Color coding makes it easier to read your code and to find mistakes; if you've mistyped a PL/SQL statement, it'll probably appear in the wrong color. Oracle SQL Developer also color codes any comments (inline or multi-line) and displays them in a light gray. This makes it easy to locate comments and commented-out code (and can also help you find code that you no longer want commented out).

Summary

In this lesson, you learned how to use the SQL `SELECT` statement to retrieve a single table column, multiple table columns, and all table columns. You also learned about commenting and saw various ways that you can use comments. In the next lesson, you'll learn how to sort the retrieved data.

This page intentionally left blank

Index

Symbols

|| operator, 93–94
% (percent sign), 70
; (semicolon), 35
_ (underscore), 72–73
* (wildcard), 37

A

ABS() function, 107
access control, 239–240
 access rights, setting, 242–243
 passwords, 244
 user accounts, 241
 users, 240–241
access rights, setting, 242–243
Add_Month(), 103
advantages
 of IN operator, 67
 of SQL, 11
AFTER trigger, 225, 228
aggregate functions, 109–110
 ALL argument, 117
 AVG(), 110–111
 combining, 118–119
 COUNT(), 112–113
 defined, 109
 DISTINCT argument, 117
 distinct values, 117–118
 joins, 160–161
 MAX(), 113–114
 MIN(), 114–115
 naming aliases, 119
 overview, 109
 SUM(), 115–116
aliases
 alternative uses, 96
 concatenating fields, 95–96
 creating, 153

 fields, concatenating, 95–96
 table names, 153–154
alphabetical sort order, 47–49
ALTER TABLE, 190–193
ALTER USER, 244
anchors, regular expressions
 (PL/SQL), 87–88
AND keyword, 58
AND operator, combining WHERE
 clause, 61–62
Application Express, 25
applications, filtering query results,
 52
ASC keyword, query results sort
 order, 49
AS keyword, 95–96
auto increment, 174
AVG() function, 110–111
 DISTINCT argument, 117

B

basic character matching, regular
 expressions (PL/SQL), 76–79
basic syntax, stored procedures,
 208–209
BEFORE triggers, 225, 228
best practices
 joins, 161
 primary keys, 10
BETWEEN keyword, 58
BETWEEN operator (WHERE
 clause), 57
breaking up, data, 8

C

calculated fields, 91–92
 concatenating fields, 92–94
 column aliases, 95–96

- mathematical calculations, 96–98
- overview, 91–92
- subqueries, 136–139
- views, 202–203
- calculated values, totaling, 116
- calculations, testing, 98
- cartesian product, 145
- case insensitive equality comparisons, 55
- case sensitivity, 71
 - query result sort order, 49
 - SQL statements, 35
- character classes, matching, 84–85
- characters
 - % (percent sign) wildcard, 70–71
 - _ (underscore) wildcard, 72
- clauses, 44
 - GROUP BY clause, 122–123, 126–128
 - HAVING clause, 124–125
 - ORDER BY clause, 45–46, 52, 126–128
 - positioning, 53, 59
 - SELECT clause, ordering, 129
 - WHERE clause. *See* WHERE clause
- client-based results formatting, 92
- clients, 14
- client-server software, Oracle, 13–15
- client tools, Oracle, 15–16
- CLOSE statement, 217
- closing cursors, 217–218
- color coding, Oracle SQL Developer, 42
- column aliases
 - alternative uses, 96
 - concatenating fields, 95–96
- columns, 7–8, 92. *See also* fields
 - concepts, 7–8
 - derived columns, 96
 - fully qualified names, 145
 - GROUP BY clause, 123
 - individual columns, 111
 - INSERT SELECT statements, 177
 - INSERT statement and, 173
 - INSERT statement, omitting columns, 175
 - multiple, sorting query results by, 45–46
 - NULL value columns, 187–189
 - omitting, 175
 - padded spaces, RTrim() function, 94–95
 - primary keys, 9–10
 - retrieving, 33–37
 - all columns*, 37–38
 - separating names in queries, 36
 - sorting data
 - descending on multiple columns*, 49
 - multiple columns*, 45–47
 - subquery result restrictions, 135
 - updating multiple, 181
 - values, deleting, 181
- combined queries, 163
 - creating, 164–166
 - duplicate rows and, 167–168
 - including/eliminating duplicate rows, 167
 - overview, 163
 - rules, 166
 - sorting results, 168–169
- combining
 - aggregate functions, 118–119
 - WHERE clause, 61
 - AND operator*, 61–62
 - order of evaluation*, 63–65
 - OR operator*, 62–63
- comments, 40–42
- COMMIT statement (transaction processing), 236–237
- commits (transaction processing), defined, 235
- complex joins, views, 198–199

concatenating, 93
 fields, 92–95
 aliases, 95–96
 column aliases, 95–96
 mathematical calculations, 96–98

connecting Oracle SQL Developer to Oracle servers, 25–26

constructs, programming constructs (stored procedures), 209–210

controlling transactions, 235
 COMMIT, 236–237
 ROLLBACK, 236
 SAVEPOINT, 237–238

correlated subqueries, 138

COS() function, 107

COUNT() function, 110–113
 DISTINCT argument, 118
 joins and, 160

COUNT* subquery, 136

CREATE OR REPLACE TRIGGER, 227

CREATE PROCEDURE, 208

create.sql, 30

CREATE TABLE, 185–187
 DEFAULT keyword, 189–190

CREATE TRIGGER, 224

CREATE USER statement, 241

CREATE VIEW statement, 197

cursor data, 220–222
 fetching, 218–220

cursors, 215
 closing, 217–218
 creating, 216–217
 data, fetching, 218–220
 explicit cursors, 216
 implementing, 216
 implicit cursors, 216
 opening, 217–218
 overview, 215

customers table, 247

custom workspaces, creating, 24–25

D

data

breaking correctly (columns), 8

breaking up, 8

cursor data, 220–222
fetching, 218–220

deleting, 181–182
guidelines, 183

filtering
IN operator, 65–67
NOT operator, 67–68
WHERE clause, 51–53

inserting, 171
complete rows, 172–176
retrieved data, 176–177

retrieving
all columns, 37–38
distinct rows, 38–39
individual columns, 33–35
multiple columns, 36–37

sorting, 43–45
by multiple columns, 45–47
by nonselected columns, 45
specifying sort direction, 47–49

updating, 179
guidelines, 183

database management systems. *See* DBMS (Database Management System)

databases, 5–8. *See also* tables
 columns, 7–8
 concepts, 5–6
 datatypes, 7–8
 defined, 6
 primary keys, 9–10
 rows, 8–9
 schemas, 7

database servers, 14

data grouping, 121–122

data insertion, 171
 inserting
complete rows, 172–176
retrieved data, 176–177

- views, 204
- WHERE clause, 202
- date and time functions, 100, 103–107
- dates, 104
- DBMS (Database Management System), 1, 6
 - interactive tools, 143
 - Oracle, 13
 - query sort order, 44
- decimal rounding, 52
- DECLARE statement, 219
 - cursors, creating, 216–217
- dedicated Oracle instances, 22–24
- default system instance, 22
- default values, tables, 189–190
- DEFAULT values, 190
- DELETE FROM, 182
- DELETE statement, 181–182
 - guidelines, 183
 - WHERE clause, 182
- DELETE triggers, 228–230
- deleting
 - column values, 181
 - data, 181–183
 - tables, 193
 - user accounts, 241
- derived columns, 96
- DESC keyword, 47–49
 - query results sort order, 47–49
- dictionary sort order (query results), 49
- DISTINCT argument, AVG() function, 117
- DISTINCT keyword, 39
- distinct rows, retrieving, 38–39
- distinct values, aggregate functions, 117–118
- DROP command, 213
- dropping
 - stored procedures, 213
 - triggers, 225
- DROP TABLE statement, 193
- DROP TRIGGER, 225
- DROP USER statement, 241
- duplicate rows, including/eliminating, 167

E

- empty strings, 189
- equality operator (WHERE clause), 53, 243
- equijoins, 148
- escaping, 72
- ETrim() function, 95
- example tables, 28–29
 - creating, 30–31
 - obtaining table scripts, 28–30
- execute, 28
- executing stored procedures, 209
- EXP() function, 107
- explicit commits, 236
- explicit cursors, 216
- Extract(), 103

F

- FETCH, 218–220
- fetching cursor data, 218–220
- fields, 92. *See also* columns
 - calculated fields, 91–92
 - concatenating fields*, 92–96
 - mathematical calculations*, 96–98
 - overview*, 91–92
 - performing mathematical calculations*, 96–98
 - subqueries*, 136–139
 - views*, 202–203
 - concatenating, 92–95
 - aliases*, 95–96
- filter condition, 51
- filtering
 - AND operator, 61–62
 - application level, 52
 - data
 - IN operator*, 65–67
 - NOT operator*, 67–68
 - WHERE clause*, 51–53
 - by date, 104
 - groups, 123–126

- IN operator, 65–67
 - multiple criteria, 61
 - NOT operator, 67–68
 - order of evaluation, 63–65
 - OR operator, 62–63
 - by subqueries, 131–135
 - views, unwanted data, 201–202
 - filters
 - % (percent sign) wildcard, 70–71
 - _ (underscore) wildcard, 72–73
 - foreign keys, 142
 - ALTER TABLE, 192
 - formatting
 - retrieved data with views, 199–200
 - server-based compared to client-based, 92
 - statements, 187
 - subqueries, 134
 - four-digit years, 104
 - FROM clause
 - creating joins, 144
 - subqueries, 139
 - FROM keyword, 34
 - fully qualified column names, 145
 - fully qualified table names, 40
 - functions, 99, 207
 - aggregate functions, 109–110
 - AVG()* function, 110–111
 - combining*, 118–119
 - COUNT()* function, 112–113
 - distinct values*, 117–118
 - joins*, 160–161
 - MAX()* function, 113–114
 - MIN()* function, 114–115
 - SUM()* function, 115–116
 - date and time functions, 100, 103–107
 - defined, 99
 - numeric functions, 100, 107
 - RTrim(), 94–95
 - system functions, 100
 - text functions, 100
 - text manipulation functions, 100–102
 - types of, 100
-
- ## G
-
- GRANT, 242–243
 - greater than operator (WHERE clause), 53
 - greater than or equal to operator (WHERE clause), 53
 - GROUP BY clause, 122–123, 126–128
 - grouping versus sorting, 126–128
 - grouping data, 121–122
 - filtering groups, 123–126
 - GROUP BY clause, 122–123
 - nested groups, 123
 - groups
 - creating, 122–123
 - filtering, 123–126
 - nested groups, 123
 - guidelines for updating/deleting data, 183
-
- ## H
-
- HAVING clause, 124–125
 - grouping data, 124
-
- ## I
-
- implicit commit, 236
 - implicit cursors, 216
 - IN keyword, 67
 - inner joins, 148–149
 - IN operator, 65–67
 - INSERT, 171–175
 - inserting data, 171
 - complete rows, 172–176
 - retrieved data, 176–177
 - INSERT SELECT, 176–177
 - INSERT statement
 - columns lists, 175
 - completing rows, 172–174

- omitting columns, 175
- overview, 171
- query data, 176–177
- security privileges, 171
- VALUES, 175

INSERT trigger, 225–228

installing Oracle, 19–20

instances

- dedicated Oracle instances,
 - creating, 22
- default system instance, 22
- matching multiple instances,
 - regular expressions, 85–87

J

joining multiple tables, 149–151

joins, 141

- aggregate functions, 160–161
- best practices, 161
- complex joins, views, 198–199
- creating, 144–145
- cross joins, 148
- equijoins, 148
- inner joins, 148–149
- left outer join, 160
- natural joins, 157–158
- outer joins, 158–160
- performance, 150
- reasons for using, 143
- right outer join, 160
- self joins, 154–157
 - versus subqueries*, 157
- views, 198–199
- WHERE clause, 145–148

K

keys

- foreign keys, 142
 - ALTER TABLE*, 192
- primary keys, 9–10, 142

keywords, 33, 255–257

- AND, 62
- AS, 95–96

- ASC, query results sort order, 49

- BETWEEN, 58

- DEFAULT, table values,
 - 189–190

- DESC, 47–49

- DISTINCT, 39

- FROM, 34

- IN, 67

- NOT, 67

- OR, 63

L

languages, SQL, 11

Last_Day(), 103

left outer join, 160

Length(), 101

less than operator (WHERE clause), 53

less than or equal to operator (WHERE clause), 53

LIKE operator, 69, 88

searching with

- percent sign (%)*, 70

- underscore (_)*, 72–73

LOOP statement, 222

Lower() function, 101

LTrim() function, 95, 101

M

Mac users, Oracle, 17

matching character classes, regular expressions (PL/SQL), 84–85

matching multiple instances, regular expressions (PL/SQL), 85–87

matching one of several characters, regular expressions (PL/SQL), 80–81

matching ranges, regular expressions (PL/SQL), 82–83

matching special characters, regular expressions (PL/SQL), 83–84

mathematical calculations, performing in fields, 96–98

mathematical operators, 98
 MAX() function, 110, 113–114
 non-numeric data, 114
 NULL values, 114
 MIN() function, 110, 114–115
 DISTINCT argument, 118
 NULL values, 115
 Months_Between(), 103
 multi-event triggers, 231
 multiple columns
 retrieving, 36–37
 sorting data, 45–47
 multiple instances, matching regular
 expressions, 85–87
 multiple tables, joining, 149–151
 multiple worksheets, 28

N

names
 fully qualified column
 names, 145
 fully qualified table names, 40
 natural joins, 157–158
 navigating tables, cursors, 215
 nested groups, 123
 Next_Day(), 103
 non-equality operator (WHERE
 clause), 53, 243
 NOT NULL, 189
 NOT operator, 67–68
 NULL, 59
 NULL keyword, updating columns,
 181
 NULL values, 187–189
 AVG() function, 110–111
 compared to empty strings, 189
 COUNT() function, 113
 empty strings, 189
 table columns, 187–189
 numeric datatypes, 253–254
 numeric functions, 100, 107

O

omitting columns, 175
 opening cursors, 217–218
 OPEN statement, 217
 operators
 defined, 61
 grouping related, 64
 HAVING clause, 124
 IN operator, 65–67
 LIKE operator, 69
 mathematical operators, 98
 NOT operator, 67–68
 || operator, 93–94
 WHERE clause, 53
 *checking against a single
 value, 54–56*
 *checking for nonmatches,
 56–57*
 *checking for no value,
 58–59*
 *checking for range of
 values, 57–58*
 Oracle, 13
 client-server software, 13–15
 client tools, 15–16
 installing, 19–20
 Mac users, 17
 PL/SQL, 15
 savepoints, 237
 setting up
 installing software, 19–20
 obtaining software, 18–19
 required software, 16–18
 Oracle Database Express Edition, 18
 Oracle Express Edition, creating
 custom workspaces, 24–25
 Oracle servers connecting to Oracle
 SQL Developer, 25–26
 Oracle SQL Developer, 19, 32
 color coding, 42
 connecting to Oracle servers,
 25–26
 overview, 27–28

ORDER BY clause, 45–46, 52, 126–128

- ascending/descending sort order, 47–49
- compared to GROUP BY clause, 126–128
- SELECT statement, 44
- sorting by multiple columns, 45–46
- views, 197

ordering

- SELECT clause, 129
- sequence number, 47

orderitems table, 248–249

order of evaluation, combining (WHERE clause), 63–65

orders table, 248

OR matches, regular expressions (PL/SQL), 79–80

OR operator, combining (WHERE clause), 62–63

outer joins, 158–160

P

parentheses, WHERE clause, 65

passwords, access control, 244

percent sign (%), wildcard searches, 70

performance, 151

- joins, 150
- subqueries, 136
- views, 197

performing mathematical calculations, calculated fields, 96–98

PI() function, 107

placeholders, 235–238

PL/SQL (Procedural Language/Structured Query Language), 15

- regular expressions, 76
 - anchors*, 87–88
 - basic character matching*, 76–79
 - matching character classes*, 84–85

- matching multiple instances*, 85–87
- matching one of several characters*, 80–81
- matching ranges*, 82–83
- matching special characters*, 83–84
- OR matches*, 79–80

populate.sql, 30

populating tables, 31–32

portability, 99

- INSERT statements and, 175

predicates (operators), 70

primary keys, 9–10, 142

- best practices, 10
- concepts, 9–10
- Customer example table, 248
- importance, 9
- OrderItems example table, 249
- Orders example table, 248
- Products example table, 247, 250
- updating tables, 191–192
- Vendors example table, 247

privileges, 242–243

Procedural Language/Structured Query Language. *See* PL/SQL

processing

- subqueries, 133
- transactions. *See* transaction processing

productnotes table, 249–250

products table, 247

programming constructs, stored procedures, 209–210

Q

queries, 131

- calculated fields, 136–139
 - concatenating fields*, 92–96
 - mathematical calculations*, 96–98
 - overview*, 91–92

- combined queries, 163
 - creating*, 164–166
 - including/eliminating duplicate rows*, 167
 - sorting results*, 168–169
- combining, 133
- data formatting, 37
- defined, 131
- filtering by, 131–134
- INSERT statement and, 176–177
- multiple WHERE clauses, 166
- overview, 131
- sorting results, 43–44
 - ascending/descending order*, 47–49
 - case sensitivity*, 49
 - by multiple columns*, 45–46
 - nonselected columns and*, 45
- subqueries, 139
- table aliases, 154
- views, 195
- wildcards (*), 37–38
- quotes, WHERE clause, 57

R

- records, compared to rows, 9
- referential integrity, 143
- reformatting retrieved data with views, 199–201
- REGEXP_INSTR(), 76
- REGEXP_LIKE(), 76, 88
- REGEXP_REPLACE(), 76
- REGEXP_SUBSTR(), 76
- regular expressions, 75
 - PL/SQL, 76
 - anchors*, 87–88
 - basic character matching*, 76–79
 - matching character classes*, 84–85
 - matching multiple instances*, 85–87

- matching one of several characters*, 80–81
- matching ranges*, 82–83
- matching special characters*, 83–84
- OR matches*, 79–80
- relational databases, sort order and, 44
- relational tables, 141–143
- removing views, 197
- renaming tables, 193
- REPEAT UNTIL statement, 222
- repetition metacharacters, 85
- REPLACE PROCEDURE, 208
- reserved words, 255
- restrictions, views, 197
- result sets, 215
- retrieved data
 - inserting, 176–177
 - reformatting with views, 199–201
- retrieving data
 - all columns, 37–38
 - distinct rows, 38–39
 - individual columns, 33–35
 - multiple columns, 36–37
- reusable views, 199
- REVOKE statement, 243–244
- RIGHT keyword (outer joins), 159
- right outer join, 160
- ROLLBACK command (transaction processing), 236
- rollbacks (transaction processing)
 - COMMIT statement, 237
 - defined, 235
 - ROLLBACK command, 236
 - savepoints and, 237–238
 - statements, 238
- rows, 8–9
 - cursors, 215
 - duplicate rows, 167
 - inserting complete rows, 172–176
 - INSERT statement, 172–174
 - retrieving distinct rows, 38–39
- RTrim() function, 94–95, 99–101

rules

UNION, 166

views, 197

run scripts versus run statements, 28

run statements, 28

S

sample tables, 245

customers table, 247

orderitems table, 248–249

orders table, 248

productnotes table, 249–250

products table, 247

vendors table, 246

savepoints, transaction

processing, 235–238

scalability, 143

scale, 143

schemas, 7

schemata, 7

search criteria, 51

search patterns, 69

security

access control, 239–240

*deleting user accounts, 241**passwords, 244**setting access rights,
242–243**user accounts, 241**users, 240–241*

UPDATE statement, 179, 182

SELECT clause, ordering, 129

SELECT statement, 33

AS keyword, 95–96

AVG() function, 111

combined queries, 163

combining, 61

concatenating fields, 94

COUNT() function, 113

IS NULL clause, 58

ORDER BY clause, 44

retrieving

*all columns, 37–38**distinct rows, 38**individual columns, 33–35**multiple columns, 36–37**unknown, 38*

subqueries, 133–134

WHERE clause, 51

self joins, 154–157

semicolons (;), 35

sequence number, ordering by, 47

sequence (SELECT statement
clauses), 129

server-based results formatting

compared to client-based, 92

servers, 14

SET command, updating tables, 180

SIN() function, 107

software, obtaining for Oracle setup,
18–19

sort direction, specifying, 47–49

sorting

combined query results,

168–169

data, 43–45

*by multiple columns, 45–47**by nonselected columns, 45**specifying sort direction,
47–49*

versus grouping, 126–128

query results, 43–44

*ascending/descending
order, 47–49**case sensitivity, 49**by multiple columns, 45–46**nonselected columns and,
45*

SOUNDEX() function 101–102

spaces, removing (RTrim function),
94–95

special characters, 69

matching with regular expres-
sions, 83–84

SQL, 10–11, 15

advantages of, 11

deleting/updating data, 183

overview, 10

PL/SQL, 15

- SQL statements, 30, 33
 - case sensitivity, 35
 - comments, 40–42
 - terminating, 35
 - white space, 35
 - SQRT() function, 107
 - statements
 - ALTER TABLE, 190
 - clauses, 44
 - COMMIT, 237
 - CREATE TABLE, 185–186
 - CREATE VIEW, 197
 - DELETE, 181–183
 - DROP TABLE, 193
 - formatting, 187
 - grouping related operators, 64
 - INSERT. *See* INSERT statement
 - rollbacks, 238
 - defined*, 235
 - SELECT. *See* SELECT statement
 - stored procedures
 - disadvantages of*, 207
 - overview*, 205–206
 - usefulness of*, 206
 - UPDATE, 179–183
 - stored procedures, 205–208
 - basic syntax, 208–209
 - building intelligent stored procedures, 210–213
 - creating, 208
 - disadvantages of, 207
 - dropping, 213
 - executing, 209
 - overview, 205–206
 - programming constructs, 209–210
 - reasons for using, 206–207
 - usefulness of, 206
 - strings. *See also* text functions
 - datatypes, 252–253
 - empty, compared to NULL values, 189
 - wildcard searching and, 70
 - subqueries, 131
 - as calculated fields, 136–139
 - building queries, 139
 - correlated subqueries, 138
 - filtering by, 131–135
 - FROM clause, 139
 - maximum amount of, 135
 - SELECT statements, 133–134
 - self joins and, 155–157
 - UPDATE statement, 181
 - WHERE clause, 135
 - SUM() function, 110, 115–116
 - multiple columns, 116
 - NULL values, 116
 - syntax, stored procedures, 208–209
 - Sysdate(), 103
 - SYS logins, 240
 - system functions, 100
 - SYSTEM login, 240
- ## T
-
- table aliases, 153–154
 - table rights, 243
 - tables, 6–7
 - calculated fields
 - concatenating fields*, 92–96
 - mathematical calculations*, 96–98
 - overview*, 91–92
 - Cartesian Product, 145
 - concepts, 6–7
 - creating, 144, 185–187
 - CREATE TABLE, 186
 - default values, 189–190
 - NULL values, 187–189
 - overview, 185
 - customers table, 247
 - default values, 189–190
 - deleting, 193
 - example tables, 28–29
 - creating*, 30–31
 - obtaining table scripts*, 28–30
 - populating*, 31–32

- functions of, 28, 245
- inner joins, 148
- inserting data, 172–174
 - from queries, 176–177*
- joining multiple tables, 149–151
- multiple tables, 149–151
- naming, 7
- NULL value, checking for, 58
- NULL value columns, 187–189
- orderitems table, 248
- orders table, 248
- overview, 141
- performance considerations, 150
- products table, 247
- relational tables, 141–143
- renaming, 193
- rows, 8
- sample tables, 245
- updating, 179–181, 190–191
 - deleting data, 181–182*
 - primary keys, 191–192*
- usefulness of, 143
- vendors table, 246
- virtual. *See* views
- WHERE clause, 145, 148

table scripts, obtaining, 29–30

Tan() function, 107

terminating SQL statements, 35

testing calculations, 98

text functions, 100–101

- list of common, 101–103

text manipulation functions, 100–102

To_Date(), 103

tools, client tools (Oracle), 15–16

trailing spaces, 72

transaction processing, 233–235, 237–238

- COMMIT command, 237
- explicit commits, 236
- managing, 235
- overview, 233–234
- ROLLBACK command, 236
- terminology, 235

transactions

- controlling, 235
 - COMMIT, 236–237
 - ROLLBACK, 236
 - SAVEPOINT, 237–238
- defined, 235

triggers, 223–224

- AFTER trigger, 228
- BEFORE triggers, 228
- creating, 224–225
- DELETE triggers, 228–230
- dropping, 225
- INSERT trigger, 225–228
- multi-event triggers, 231
- overview, 232
- UPDATE triggers, 230–231

Trim() function, 95

trimming padded spaces, 94–95

U

underscore (_), 72–73

UNION

- creating combined queries, 164–166
- duplicate rows, 167
- rules for, 166
- sorting combined query results, 168–169
- versus WHERE clause, 168

unions, 163. *See also* combined queries

unknown, 59

unsorted data, query results, 34

UPDATE, 179–181

- guidelines, 183
- security privileges, 179, 182
- subqueries, 181

UPDATE triggers, 230–231

updating

- data, 179
 - guidelines, 183*
- table data, 179–181
 - deleting data, 181–182*

tables, 190–191
 primary keys, 191–192
 views, 198, 203–204
 Upper() function, 101
 user accounts, 241
 users, access control, 240–241

V

values
 concatenation, 93
 trimming padded space, 95
 VALUES, 175
 vendors table, 246
 views, 195–196
 calculated fields, 202–203
 complex joins, 198–199
 creating, 197
 data retrieval, 204
 filtering data, 201–202
 joins, simplifying, 198–199
 ORDER BY clause, 197
 overview, 195
 performance, 197
 reasons for using, 196
 reformatting retrieved data,
 199–201
 removing, 197
 restrictions, 197
 reusable, 199
 rules, 197
 updating, 198, 203–204
 usefulness of, 196
 virtual tables. *See* views

W-X-Y-Z

websites, example table download
 site, 29

WHERE clause, 51–53. *See also*

HAVING clause

 checking against single value, 54
 checking for nonmatches, 56–57
 checking for NULL value, 58

 checking for range of values,
 57–58
 combining, 61
 AND operator, 61–62
 order of evaluation, 63–65
 OR operator, 62–63
 in queries, 163

 data retrieval, 202

 DELETE statements, 182

 filtering data, 124

 filtering groups, 125

 joins, 145–148

 operators, 53

checking against a single
 value, 54–56

checking for nonmatches,
 56–57

checking for no value,
 58–59

checking for range of val-
 ues, 57–58

 parentheses, 65

 quotes, 57

 Soundex() function, 102

 subqueries, 134–135

 UPDATE statements, 179–180

 versus UNION, 168

 wildcards, 69

white space, SQL statements, 35

wildcard (*), 37

wildcards, 37–38, 69

 natural joins, 158

wildcard searches

 LIKE operators, 69

 percent sign (%), 70

 tips for, 74

 trailing spaces, 72

 underscore (_), 72–73

working environments, 21

 creating

custom workspaces, 24–25

dedicated Oracle instances,
 22–24

worksheets, multiple worksheets, 28

writing stored procedures, 207