

THE EXPERT'S VOICE® IN ORACLE

OakTable  
PRESS

# Beginning Oracle SQL

*Build a solid foundation for success in Oracle*



Lex de Haan, Daniel Fink, Tim Gorman,  
Inger Jørgensen, Karen Morton

Apress®

# Beginning Oracle SQL



Lex de Haan  
Daniel Fink  
Tim Gorman  
Inger Jørgensen  
Karen Morton

Apress®

## **Beginning Oracle SQL**

Copyright © 2009 by Lex de Haan, Daniel Fink, Tim Gorman, Inger Jørgensen, Karen Morton

All rights reserved. No part of this work may be reproduced or transmitted in any form or by any means, electronic or mechanical, including photocopying, recording, or by any information storage or retrieval system, without the prior written permission of the copyright owner and the publisher.

ISBN-13 (pbk): 978-1-4302-7197-0

ISBN-13 (electronic): 978-1-4302-7196-3

Printed and bound in the United States of America 9 8 7 6 5 4 3 2 1

Trademarked names may appear in this book. Rather than use a trademark symbol with every occurrence of a trademarked name, we use the names only in an editorial fashion and to the benefit of the trademark owner, with no intention of infringement of the trademark.

President and Publisher: Paul Manning

Lead Editor: Jonathan Gennick

Technical Reviewers: Tim Gorman, Daniel Fink

Editorial Board: Clay Andres, Steve Anglin, Mark Beckner, Ewan Buckingham, Gary Cornell, Jonathan Gennick, Jonathan Hassell, Michelle Lowman, Matthew Moodie, Duncan Parkes, Jeffrey Pepper, Frank Pohlmann, Douglas Pundick, Ben Renow-Clarke, Dominic Shakeshaft, Matt Wade, Tom Welsh

Coordinating Editor: Jim Markham

Copy Editor: Seth Kline

Compositor: Bytheway Publishing Services

Indexer: Brenda Miller

Artist: April Milne

Cover Designer: Anna Ishchenko

Distributed to the book trade worldwide by Springer-Verlag New York, Inc., 233 Spring Street, 6th Floor, New York, NY 10013. Phone 1-800-SPRINGER, fax 201-348-4505, e-mail [orders-ny@springer-sbm.com](mailto:orders-ny@springer-sbm.com), or visit <http://www.springeronline.com>.

For information on translations, please contact Apress directly at 2855 Telegraph Avenue, Suite 600, Berkeley, CA 94705. Phone 510-549-5930, fax 510-549-5939, e-mail [info@apress.com](mailto:info@apress.com), or visit <http://www.apress.com>.

Apress and friends of ED books may be purchased in bulk for academic, corporate, or promotional use. eBook versions and licenses are also available for most titles. For more information, reference our Special Bulk Sales—eBook Licensing web page at <http://www.apress.com/info/bulksales>.

The information in this book is distributed on an “as is” basis, without warranty. Although every precaution has been taken in the preparation of this work, neither the author(s) nor Apress shall have any liability to any person or entity with respect to any loss or damage caused or alleged to be caused directly or indirectly by the information contained in this work.

The source code for this book is available to readers from this book’s catalog page at <http://www.apress.com>. The exact link as of this writing is: <http://apress.com/book/view/1430271970>.

# Contents at a Glance

■ Contents at a Glance .....	iii
■ Contents .....	iv
■ About the Authors .....	xvii
■ Acknowledgments .....	xix
■ Introduction .....	xxi
■ Chapter 1: Relational Database Systems and Oracle.....	1
■ Chapter 2: Introduction to SQL, AQL*Plus, and SQL Developer .....	25
■ Chapter 3: Data Definition, Part I .....	71
■ Chapter 4: Retrieval: The Basics.....	83
■ Chapter 5: Retrieval: Functions .....	117
■ Chapter 6: Data Manipulation .....	145
■ Chapter 7: Data Definition, Part II.....	163
■ Chapter 8: Retrieval: Multiple Tables and Aggregation .....	195
■ Chapter 9: Retrieval: Some Advanced Features .....	233
■ Chapter 10: Views.....	265
■ Chapter 11: Writing and Automating SQL*Plus Scripts .....	287
■ Chapter 12: Object-Relational Features.....	329
■ Appendix A: The Seven Case Tables .....	349
■ Appendix B: Answers to the Exercises .....	359
■ Index .....	405

# Contents

■ <b>Contents at a Glance</b> .....	<b>iii</b>
■ <b>Contents</b> .....	<b>iv</b>
■ <b>About the Authors</b> .....	<b>xvii</b>
■ <b>Acknowledgments</b> .....	<b>xix</b>
■ <b>Introduction</b> .....	<b>xxi</b>
■ <b>Chapter 1: Relational Database Systems and Oracle</b> .....	<b>1</b>
1.1 Information Needs and Information Systems .....	<b>1</b>
1.2 Database Design .....	<b>2</b>
Entities and Attributes .....	<b>2</b>
Generic vs. Specific .....	<b>3</b>
Redundancy .....	<b>4</b>
Consistency, Integrity, and Integrity Constraints .....	<b>5</b>
Data Modeling Approach, Methods, and Techniques .....	<b>6</b>
Semantics .....	<b>7</b>
Information Systems Terms Review .....	<b>7</b>
1.3 Database Management Systems .....	<b>7</b>
DBMS Components .....	<b>8</b>
Kernel .....	<b>8</b>
Data Dictionary .....	<b>8</b>
Query Languages .....	<b>8</b>
DBMS Tools .....	<b>9</b>

Database Applications .....	9
DBMS Terms Review .....	9
<b>1.4 Relational Database Management Systems .....</b>	<b>10</b>
<b>1.5 Relational Data Structures .....</b>	<b>10</b>
Tables, Columns, and Rows .....	11
The Information Principle .....	12
Datatypes.....	12
Keys.....	12
Missing Information and Null Values .....	13
Constraint Checking .....	14
Predicates and Propositions .....	14
Relational Data Structure Terms Review .....	14
<b>1.6 Relational Operators .....</b>	<b>15</b>
<b>1.7 How Relational Is My DBMS?.....</b>	<b>16</b>
<b>1.8 The Oracle Software Environment.....</b>	<b>17</b>
<b>1.9 Case Tables .....</b>	<b>19</b>
The ERM Diagram of the Case .....	19
Table Descriptions .....	21
<b>■ Chapter 2: Introduction to SQL, AQL*Plus, and SQL Developer .....</b>	<b>25</b>
<b>2.1 Overview of SQL .....</b>	<b>25</b>
Data Definition .....	26
Data Manipulation and Transactions .....	26
Retrieval .....	27
Security .....	29
Privileges and Roles .....	29
GRANT and REVOKE.....	31
<b>2.2 Basic SQL Concepts and Terminology .....</b>	<b>32</b>
Constants (Literals).....	32

Variables.....	34
Operators, Operands, Conditions, and Expressions.....	34
Arithmetic Operators .....	35
The Alphanumeric Operator: Concatenation.....	35
Comparison Operators.....	35
Logical Operators .....	36
Expressions .....	36
Functions.....	37
Database Object Naming .....	38
Comments .....	39
Reserved Words.....	39
<b>2.3 Introduction to SQL*Plus.....</b>	<b>39</b>
Entering Commands .....	40
Using the SQL Buffer .....	41
Using an External Editor .....	42
Using the SQL*Plus Editor .....	43
Using SQL Buffer Line Numbers .....	46
Using the Ellipsis .....	48
SQL*Plus Editor Command Review.....	48
Saving Commands.....	49
Running SQL*Plus Scripts .....	51
Specifying Directory Path Specifications.....	52
Adjusting SQL*Plus Settings.....	53
Spooling a SQL*Plus Session.....	56
Describing Database Objects.....	57
Executing Commands from the Operating System.....	57
Clearing the Buffer and the Screen .....	57
SQL*Plus Command Review .....	57

2.4 Introduction to SQL Developer .....	58
Installing and Configuring SQL Developer .....	58
Connecting to a Database.....	61
Exploring Objects.....	62
Entering Commands .....	63
Run Statement.....	64
Run Script.....	65
Saving Commands to a Script .....	66
Running a Script.....	67
<b>■ Chapter 3: Data Definition, Part I .....</b>	<b>71</b>
3.1 Schemas and Users .....	71
3.2 Table Creation.....	72
3.3 Datatypes.....	73
3.4 Commands for Creating the Case Tables .....	75
3.5 The Data Dictionary .....	77
<b>■ Chapter 4: Retrieval: The Basics .....</b>	<b>83</b>
4.1 Overview of the SELECT Command .....	83
4.2 The SELECT Clause .....	85
Column Aliases.....	86
The DISTINCT Keyword.....	87
Column Expressions .....	87
The DUAL Table .....	88
Null Values in Expressions.....	90
4.3 The WHERE Clause .....	90
4.4 The ORDER BY Clause.....	91
4.5 AND, OR, and NOT.....	94
The OR Operator .....	94
The AND Operator and Operator Precedence Issues .....	95



The NOT Operator .....	96
<b>4.6 BETWEEN, IN, and LIKE.....</b>	<b>98</b>
The BETWEEN Operator .....	98
The IN Operator .....	99
The LIKE Operator .....	100
<b>4.7 CASE Expressions .....</b>	<b>101</b>
<b>4.8 Subqueries.....</b>	<b>104</b>
The Joining Condition .....	105
When a Subquery Returns Too Many Values .....	106
Comparison Operators in the Joining Condition .....	107
When a Single-Row Subquery Returns More Than One Row .....	108
<b>4.9 Null Values .....</b>	<b>109</b>
Null Value Display .....	109
The Nature of Null Values .....	109
The IS NULL Operator .....	111
Null Values and the Equality Operator .....	112
Null Value Pitfalls.....	113
<b>4.10 Truth Tables.....</b>	<b>114</b>
<b>4.11 Exercises .....</b>	<b>116</b>
<b>■ Chapter 5: Retrieval: Functions .....</b>	<b>117</b>
<b>5.1 Overview of Functions .....</b>	<b>117</b>
<b>5.2 Arithmetic Functions.....</b>	<b>119</b>
<b>5.3 Text Functions .....</b>	<b>121</b>
<b>5.4 Regular Expressions .....</b>	<b>125</b>
Regular Expression Operators and Metasymbols .....	126
Regular Expression Function Syntax .....	127
Influencing Matching Behavior .....	127
REGEXP_INSTR Return Value.....	128

REGEXP_LIKE .....	128
REGEXP_INSTR .....	129
REGEXP_SUBSTR .....	130
REGEXP_REPLACE .....	130
<b>5.5 Date Functions .....</b>	<b>131</b>
EXTRACT .....	132
ROUND and TRUNC .....	133
MONTHS_BETWEEN and ADD_MONTHS .....	133
NEXT_DAY and LAST_DAY .....	134
<b>5.6 General Functions .....</b>	<b>134</b>
GREATEST and LEAST .....	135
NVL .....	136
DECODE .....	136
<b>5.7 Conversion Functions .....</b>	<b>137</b>
TO_NUMBER and TO_CHAR .....	138
Conversion Function Formats .....	139
Datatype Conversion .....	141
CAST .....	141
<b>5.8 Stored Functions .....</b>	<b>142</b>
<b>5.9 Exercises .....</b>	<b>143</b>
<b>■ Chapter 6: Data Manipulation .....</b>	<b>145</b>
<b>6.1 The INSERT Command .....</b>	<b>146</b>
Standard INSERT Commands .....	146
INSERT Using Subqueries .....	149
<b>6.2 The UPDATE Command .....</b>	<b>151</b>
<b>6.3 The DELETE Command .....</b>	<b>154</b>
<b>6.4 The MERGE Command .....</b>	<b>157</b>
<b>6.5 Transaction Processing .....</b>	<b>159</b>

6.6 Locking and Read Consistency .....	160
Locking .....	160
Read Consistency .....	161
<b>■ Chapter 7: Data Definition, Part II .....</b>	<b>163</b>
7.1 The CREATE TABLE Command .....	163
7.2 More on Datatypes.....	165
Character Datatypes .....	166
Comparison Semantics.....	167
Column Data Interpretation .....	167
Numbers Revisited .....	167
7.3 The ALTER TABLE and RENAME Commands.....	167
7.4 Constraints.....	170
Out-of-Line Constraints .....	170
Inline Constraints.....	172
Constraint Definitions in the Data Dictionary.....	173
Case Table Definitions with Constraints.....	174
A Solution for Foreign Key References: CREATE SCHEMA.....	176
Deferrable Constraints.....	177
7.5 Indexes .....	178
Index Creation.....	179
Unique Indexes .....	180
Bitmap Indexes .....	180
Function-Based Indexes .....	180
Index Management.....	181
7.6 Performance Monitoring with SQL Developer AUTOTRACE.....	182
7.7 Sequences .....	185
7.8 Synonyms .....	186
7.9 The CURRENT_SCHEMA Setting .....	188

7.10 The DROP TABLE Command .....	189
7.11 The TRUNCATE Command.....	191
7.12 The COMMENT Command.....	191
7.13 Exercises .....	193
<b>■ Chapter 8: Retrieval: Multiple Tables and Aggregation .....</b>	<b>195</b>
8.1 Tuple Variables .....	195
8.2 Joins .....	197
Cartesian Products .....	198
Equijoins .....	198
Non-equijoins .....	199
Joins of Three or More Tables .....	200
Self-Joins .....	201
8.3 The JOIN Clause.....	202
Natural Joins .....	203
Equijoins on Columns with the Same Name.....	204
8.4 Outer Joins .....	205
Old Oracle-Specific Outer Join Syntax.....	206
New Outer Join Syntax .....	207
Outer Joins and Performance.....	208
8.5 The GROUP BY Component .....	208
Multiple-Column Grouping.....	210
GROUP BY and Null Values .....	210
8.6 Group Functions.....	211
Group Functions and Duplicate Values .....	212
Group Functions and Null Values.....	213
Grouping the Results of a Join .....	214
The COUNT(*) Function .....	214
Valid SELECT and GROUP BY Clause Combinations.....	216

<b>8.7 The HAVING Clause .....</b>	<b>217</b>
The Difference Between WHERE and HAVING .....	218
HAVING Clauses Without Group Functions .....	218
A Classic SQL Mistake .....	219
Grouping on Additional Columns .....	220
<b>8.8 Advanced GROUP BY Features.....</b>	<b>222</b>
GROUP BY ROLLUP.....	222
GROUP BY CUBE.....	223
CUBE, ROLLUP, and Null Values.....	224
The GROUPING Function .....	224
The GROUPING_ID Function.....	225
<b>8.9 Partitioned Outer Joins .....</b>	<b>226</b>
<b>8.10 Set Operators.....</b>	<b>228</b>
<b>8.11 Exercises .....</b>	<b>231</b>
<b>■ Chapter 9: Retrieval: Some Advanced Features.....</b>	<b>233</b>
<b>9.1 Subqueries Continued .....</b>	<b>233</b>
The ANY and ALL Operators.....	234
Defining ANY and ALL.....	235
Rewriting SQL Statements Containing ANY and ALL .....	236
Correlated Subqueries.....	237
The EXISTS Operator .....	238
Subqueries Following an EXISTS Operator .....	239
EXISTS, IN, or JOIN? .....	239
NULLS with NOT EXISTS and NOT IN .....	242
<b>9.2 Subqueries in the SELECT Clause.....</b>	<b>243</b>
<b>9.3 Subqueries in the FROM Clause .....</b>	<b>244</b>
<b>9.4 The WITH Clause.....</b>	<b>245</b>

9.5 Hierarchical Queries .....	247
START WITH and CONNECT BY .....	248
LEVEL, CONNECT_BY_ISCYCLE, and CONNECT_BY_ISLEAF .....	249
CONNECT_BY_ROOT and SYS_CONNECT_BY_PATH .....	250
Hierarchical Query Result Sorting .....	251
9.6 Analytical Functions.....	252
Partitions .....	254
Function Processing .....	257
9.7 Flashback Features .....	259
AS OF .....	260
VERSIONS BETWEEN.....	262
FLASHBACK TABLE .....	262
9.8 Exercises .....	264
■ <b>Chapter 10: Views.....</b>	<b>265</b>
10.1 What Are Views?.....	265
10.2 View Creation.....	266
Creating a View from a Query.....	267
Getting Information About Views from the Data Dictionary .....	269
Replacing and Dropping Views.....	271
10.3 What Can You Do with Views? .....	271
Simplifying Data Retrieval .....	271
Maintaining Logical Data Independence .....	273
Implementing Data Security .....	274
10.4 Data Manipulation via Views .....	274
Updatable Join Views .....	276
Nonupdatable Views.....	277
The WITH CHECK OPTION Clause.....	278
Disappearing Updated Rows .....	278

Inserting Invisible Rows .....	279
Preventing These Two Scenarios .....	280
Constraint Checking .....	280
10.5 Data Manipulation via Inline Views .....	281
10.6 Views and Performance .....	282
10.7 Materialized Views .....	283
Properties of Materialized Views .....	284
Query Rewrite .....	284
10.8 Exercises .....	286
<b>■ Chapter 11: Writing and Automating SQL*Plus Scripts .....</b>	<b>287</b>
11.1 SQL*Plus Variables .....	288
SQL*Plus Substitution Variables .....	288
SQL*Plus User-Defined Variables .....	290
Implicit SQL*Plus User-Defined Variables .....	291
User-Friendly Prompting .....	292
SQL*Plus System Variables .....	293
11.2 Bind Variables .....	298
Bind Variable Declaration .....	299
Bind Variables in SQL Statements .....	300
11.3 SQL*Plus Scripts .....	301
Script Execution .....	301
Script Parameters .....	302
SQL*Plus Commands in Scripts .....	304
The login.sql Script .....	305
11.4 Report Generation with SQL*Plus .....	306
The SQL*Plus COLUMN Command .....	307
The SQL*Plus TTITLE and BTITLE Commands .....	311
The SQL*Plus BREAK Command .....	312

The SQL*Plus COMPUTE Command .....	315
The Finishing Touch: SPOOL.....	317
<b>11.5 HTML in SQL*Plus .....</b>	<b>318</b>
HTML in SQL*Plus.....	318
<b>11.6 Building SQL*Plus Scripts for Automation .....</b>	<b>321</b>
What Is a SQL*Plus Script?.....	321
Capturing and Using Input Parameter Values.....	322
Passing Data Values from One SQL Statement to Another .....	323
Mechanism 1: The NEW_VALUE Clause.....	323
Mechanism 2: Bind Variables .....	324
Handling Error Conditions.....	325
<b>11.7 Exercises .....</b>	<b>326</b>
<b>■ Chapter 12: Object-Relational Features .....</b>	<b>329</b>
<b>12.1 More Datatypes.....</b>	<b>329</b>
Collection Datatypes.....	330
Methods.....	330
<b>12.2 Varrays.....</b>	<b>331</b>
Creating the Array.....	331
Populating the Array with Values .....	333
Querying Array Columns.....	334
<b>12.3 Nested Tables .....</b>	<b>336</b>
Creating Table Types.....	336
Creating the Nested Table .....	336
Populating the Nested Table.....	337
Querying the Nested Table .....	338
<b>12.4 User-Defined Types .....</b>	<b>339</b>
Creating User-Defined Types.....	339
Showing More Information with DESCRIBE .....	340



12.5 Multiset Operators .....	341
Which SQL Multiset Operators Are Available? .....	341
Preparing for the Examples .....	342
Using IS NOT EMPTY and CARDINALITY .....	343
Using POWERMULTISET .....	344
Using MULTISET UNION .....	345
Converting Arrays into Nested Tables .....	346
12.6 Exercises .....	346
■ <b>Appendix A: The Seven Case Tables .....</b>	<b>349</b>
ERM Diagram .....	349
Table Structure Descriptions .....	350
Columns and Foreign Key Constraints .....	351
Contents of the Seven Tables .....	352
Hierarchical Employees Overview .....	357
Course Offerings Overview .....	357
■ <b>Appendix B: Answers to the Exercises .....</b>	<b>359</b>
Chapter 4 Exercises .....	359
Chapter 5 Exercises .....	369
Chapter 7 Exercises .....	374
Chapter 8 Exercises .....	376
Chapter 9 Exercises .....	386
Chapter 10 Exercises .....	395
Chapter 11 Exercises .....	397
Chapter 12 Exercises .....	401
■ <b>Index .....</b>	<b>405</b>

# About the Author



■ **Lex de Haan** studied applied mathematics at the Technical University in Delft, The Netherlands. His experience with Oracle goes back to the mid-1980s, version 4. He worked for Oracle Corporation from 1990 until 2004, in various education-related roles, ending up in Server Technologies (product development) as senior curriculum manager for the advanced DBA curriculum. In that role, he was involved in the development of Oracle9i Database and Oracle Database 10g. In March 2004, he decided to go independent and founded Natural Join B.V. In 1999, he became involved in the ISO SQL language standardization process, as a member of the Dutch national body. Lex passed away on February 1, 2006.



■ **Daniel Fink** has been working with Oracle since 1995, starting as a developer/dba on Oracle7 Parallel Server on OpenVMS, and then moving to database administration. Currently working as a consultant, he focuses on diagnosis, optimization, and data recovery. He is also a highly regarded trainer and presenter, speaking at user group conferences in the United States and Europe. When not working with technology, he enjoys the mountains of Colorado on foot, on skis, or from the seat of a bicycle.



■ **Tim Gorman** has worked in IT with relational databases since 1984, as an Oracle application developer since 1990, and as an Oracle database administrator since 1993. He is an independent consultant (<http://www.EvDBT.com>) specializing in data warehousing, performance tuning, database administration (particularly availability). He has been an active member of the Rocky Mountain Oracle Users Group (<http://www.rmoug.org>). He has co-authored three previous books and taught classes and presented at conferences all over the US, Canada, Latin America, Europe, and Asia. Tim lives in Colorado with his wife Lori and their four teenage children. He still can't believe that he gets paid for doing this and is officially one very happy guy.



■ After a Languages Master degree (English and French) **Inger Jørgensen** started teaching SQL and PL/SQL as well as database administration from Oracle version 6 onwards with a five-year period in between of teaching developers Forms, Reports, and Graphics. Inger spent 18 years at Oracle Corporation, and is presently at Oracle partner Miracle in Denmark.



■ **Karen Morton** is an Oracle performance optimization specialist with nearly 20 years experience working with the Oracle database. She works with companies around the world teaching application optimization in both classroom settings and shoulder-to-shoulder consulting engagements. She is a frequent speaker at conferences and user groups, an Oracle ACE, and a member of the OakTable network. She blogs at <http://karenmorton.blogspot.com>.

# Acknowledgments

I want to thank many friends who contributed to the quality of this book by reviewing it and providing their feedback. Cary Millsap and Jocke Treugut, two good friends and members of the OakTable network, were my main reviewers. Cary helped me with his constant focus on “doing things right” from the very beginning, and Jocke helped me find the right balance between theory and practice. Martin Jensen, one of my good old friends inside Oracle and an Oakie as well, provided precisely the feedback I needed from his impressive Oracle consulting background. Stephen Cannan, my colleague in the Dutch national body for the SQL Standardization and the convenor of the international ISO / IEC / JTC1 / SC32 / WG3 committee, commented on my draft chapters based on his vast experience in the SQL standardization area.

Kristina Youso, a former colleague and good friend from my years in Global Curriculum Development in Oracle and one of the best content editors I have ever worked with, was so kind to check and improve my English language.

Last, but not least, I must mention the professionalism and enthusiasm of all the Apress folks involved in the production of this book: Tony Davis, Beckie Stones, Marilyn Smith, and Kelly Winquist. Thanks folks . . .

My two daughters are too old to be mentioned here, the cat was not involved in any way, and I leave it up to Mogens Nørgaard to say something nice about my wife, Juliette.

*Lex de Haan from first edition*

I am honored to be part of the team to update this book and maintain Lex's legacy. Thank you Juliette for your support in this project and for my last visit with Lex. Lex was a professional colleague and friend. I cherish the all too brief times we spent together at conferences and our email conversations. I want to thank Jonathan Gennick and James Markham with Apress who worked very hard to make this book possible and were very tolerant and understanding of the trials and tribulations of a first time author. Your patience and perseverance were invaluable. This project would not have been possible without Inger, Karen and Tim. Thank you for your time and energy. The discussions, reviews, last minute emails have been so very important and are greatly appreciated. Over the years, many people have supported, enlightened, educated and challenged me. Thank you to Tim, Vincent, Kirti, Rachel, Mogens, Cary, Jonathan, Carol, Craig, Lex, Kurt, Robyn, and fellow members of the Oak Table. Thanks to BD for encouraging me to make the leap to Oracle.

Family and friends make everything possible. Thanks Mom and Dad for all you have done and continue to do. The constant support of E, Janet, Sujeeva, and Roberta is invaluable. Thank you Beverly for all your support, understanding, and love over these months.

*Daniel Fink*

I would like to acknowledge his gratitude to Gary Dodge my friend and mentor, Mogens Norgaard for opportunity and motivation, Jonathan Gennick for patience and wisdom, Abdul Ebadi for encouragement and inspiration, and Lori Shine for hope, love, spirit, and fun. There are so many more, family, friends and colleagues alike, and I love the world in which I live.

*Tim Gorman*

I would like to thank Lex for his friendship and his great ability to teach in a clear, awesome, pedagogical way.

*Inger Jørgensen*

The first, and perhaps most important, acknowledgement I make goes to Lex de Haan for creating the original version of this book. I am very honored to follow in his footsteps and to enliven his work.

There have been so many enthusiastic people in the Oracle community that I've met in classes, consulting engagements, and conferences over the last 20 years. These connections are my favorite part of what I do. It is my hope that this book makes a positive contribution back to the community that has given me so much.

Finally, thanks to my family for all the support and encouragement not only to complete this work, but every single day. It's coming home to you that makes everything complete.

*Karen Morton*

# Introduction

This book was born from a translation of a book originally written by Lex de Haan in Dutch. That book was first published in 1993, and went through several revisions in its native Dutch before Lex decided to produce an English version. Apress published that English version in 2005 under the title “Mastering Oracle SQL and SQL\*Plus”. The book has since earned respect as excellent, accurate, and concise tutorial on Oracle’s implementation of SQL.

While SQL is a fairly stable language, there have been changes to Oracle’s implementation of it over the years. The book you are holding now is a revision of Lex’s original, English-language work. The book has been revised to cover new developments in Oracle SQL since 2005, especially those in Oracle Database 11g Release 1 and Release 2. The book has also been given the title “Beginning Oracle SQL”. The new title better positions the book in Apress’s line, better reflects the content, fits better with branding and marketing efforts, and marks the book as a foundational title that Apress intends to continue revising and publishing in the long term.

## About this Book

This is *not* a book about advanced SQL. It is *not* a book about the Oracle optimizer and diagnostic tools. And it is *not* a book about relational calculus, predicate logic, or set theory. This book is a SQL primer. It is meant to help you learn Oracle SQL by yourself. It is ideal for self-study, but it can also be used as a guide for SQL workshops and instructor-led classroom training.

This is a practical book; therefore, you need access to an Oracle environment for hands-on exercises. All the software that you need to install Oracle Database on either Windows or Linux for learning purposes is available free of charge from the Oracle Technology Network (OTN). Begin your journey with a visit to the OTN website at:

<http://www.oracle.com/technology/index.html>

From the OTN home page, you can navigate to product information, to documentation and manual sets, and to free downloads that you can install on your own PC for learning purposes.

This edition of the book is current with Oracle Database 11g Release 2. However, Oracle SQL has been reasonably stable over the years. All the examples should also run under Release 1. And most will still run under Oracle Database 10g, under Oracle Database 9i, and even under Oracle Database 8i, if you’re running software that old. Of course, as you go further back in release-time, you will find more syntax that is not supported in each successively older release. Oracle Corporation does tend to add a few new SQL features with each new release of their database product.

Oracle Corporation has shown great respect for SQL standards over the past decade. We agree with supporting standards, and we follow the ANSI/ISO standard SQL syntax as much as possible in this book. Only in cases of useful, Oracle-specific SQL extensions do we deviate from the international standard. Therefore, most SQL examples given in this book are probably also valid for other database management system (DBMS) implementations supporting the SQL language.

SQL statements discussed in this book are explained with concrete examples. We focus on the main points, avoiding peripheral and arcane side-issues as much as possible. The examples are presented clearly in a listing format, as in the example shown here in Listing I-1.

**1. Listing I-1. A SQL SELECT Statement**

```
SELECT 'Hello world!'
FROM dual;
```

One difference between this edition and its predecessor is that we omit the “SQL>” prompt from most of our examples. That prompt comes from SQL\*Plus, the command-line interface that old-guard database administrators and developers have used for years. We now omit SQL\*Plus prompts from all examples that are not specific to SQL\*Plus. We do that out of respect for the growing use of graphical interfaces such as Oracle SQL Developer.

This book does not intend (nor pretend) to be complete; the SQL language is too voluminous and the Oracle environment is much too complex. Oracle’s SQL reference manual, named *Oracle SQL Reference*, comes in at just over 1500 pages for the Oracle Database 11g Release 2 edition. Moreover, the current ISO SQL standard documentation has grown to a size that is simply not feasible anymore to print on paper.

The main objective of this book is the combination of *usability* and *affordability*. The official Oracle documentation offers detailed information in case you need it. Therefore, it is a good idea to have the Oracle manuals available while working through the examples and exercises in this book. The Oracle documentation is available online from the OTN website mentioned earlier in this introduction. You can access that documentation in html form, or you can download PDF copies of selected manuals.

The focus of this book is using SQL for data *retrieval*. Data definition and data manipulation are covered in less detail. Security, authorization, and database administration are mentioned only for the sake of completeness in the “Overview of SQL” section of Chapter 2.

Throughout the book, we use a case consisting of seven tables. These seven tables contain information about employees, departments, and courses. As Chris Date, a well-known guru in the professional database world, said during one of his seminars, “There are only three databases: employees and departments, orders and line items, and suppliers and shipments.”

The amount of data (i.e., the cardinality) in the case tables is deliberately kept low. This enables you to check the results of your SQL commands manually, which is nice while you’re learning to master the SQL language. In general, checking your results manually is impossible in real information systems due to the volume of data in such systems.

It is not the data volume or query response time that matters in this book. What’s important is the database structure complexity and SQL statement correctness. After all, it does no good for a statement to be fast, or to perform well, if all it does in the end is produce incorrect results. Accuracy first! That’s true in many aspects of life, including in SQL.

## About the Chapters of this Book

Chapter 1 provides a concise introduction to the theoretical background of information systems and some popular database terminology, and then continues with a global overview of the Oracle software and an introduction to the seven case tables. It is an important, foundational chapter that will help you get the most from the rest of the book.

Chapter 2 starts with a high-level overview of the SQL language. We follow that with an introduction to SQL\*Plus and SQL Developer. The first – SQL\*Plus – is a command-line tool that you can use to send a SQL statement to the database and get results back. Many database administrators use SQL\*Plus routinely, and you can rely upon it to be present in any Oracle Database installation. SQL Developer is

also a tool for testing and executing SQL. It comes with a graphical user interface, and it is a tool that has gained much ground and momentum with developers.

Data definition is covered in two nonconsecutive chapters: Chapter 3 and Chapter 7. This is done to allow you to start with SQL retrieval as soon as possible. Therefore, Chapter 3 covers only the most basic data-definition concepts (tables, datatypes, and the data dictionary).

Retrieval is also spread over multiple chapters—four chapters, to be precise. Chapter 4 focuses on the **SELECT**, **WHERE**, and **ORDER BY** clauses of the **SELECT** statement. The most important SQL functions are covered in Chapter 5, which also covers null values and subqueries. In Chapter 8, we start accessing multiple tables at the same time (joining tables) and aggregating query results; in other words, the **FROM**, the **GROUP BY**, and the **HAVING** clauses get our attention in that chapter. To finish the coverage of data retrieval with SQL, Chapter 9 revisits subqueries to show some more advanced subquery constructs. That chapter also introduces windows and analytical functions, hierarchical queries, and flashback features.

Chapter 6 discusses data manipulation with SQL. The commands **INSERT**, **UPDATE**, **DELETE**, and **MERGE** are introduced. This chapter also pays attention to some topics related to data manipulation: transaction processing, read consistency, and locking.

In Chapter 7, we revisit data definition, to drill down into constraints, indexes, sequences, and performance. Synonyms are explained in the same chapter. Chapters 8 and 9 continue coverage of data retrieval with SQL.

Chapter 10 introduces views. What are views, when should you use them, and what are their restrictions? This chapter explores the possibilities of data manipulation via views, discusses views and performance, and introduces materialized views.

Chapter 11 is about automation. SQL statements can be long, and sometimes you want to execute several in succession. Chapter 11 shows you how to develop automated scripts that you can run via SQL\*Plus. Many, many Oracle databases are kept alive and healthy by automated SQL\*Plus scripts written by savvy database administrators.

Oracle is an object-relational database management system. Since Oracle Database 8, many object-oriented features have been added to the SQL language. As an introduction to these features, Chapter 12 provides a high-level overview of user-defined datatypes, arrays, nested tables, and multiset operators.

Finally, the book ends with two appendixes. Appendix A at the end of this book provides a detailed look into the example tables used in this book's examples. Appendix B gives the exercise solutions.

## About the Case Tables

Chapter 1 describes the case tables used in the book's examples. Appendix A goes into even more detail, should you want it. The book's catalog page on the Apress.com website contains a link to a SQL\*Plus script that you can use to create and populate the example tables. The direct link to that page is: <http://apress.com/book/view/1430271970>. When you get there, look on the left side of the page for a section entitled "Book Resources". You should find a "Source Code" link within that section. Click on that link to download the script.





# Relational Database Systems and Oracle

The focus of this book is writing SQL in Oracle, which is a relational database management system. This first chapter provides a brief introduction to relational database systems in general, followed by an introduction to the Oracle software environment. The main objective of this chapter is to help you find your way in the relational database jungle and to get acquainted with the most important database terminology.

The first three sections discuss the main reasons for automating information systems using databases, what needs to be done to design and build relational database systems, and the various components of a relational database management system. The following sections go into more depth about the theoretical foundation of relational database management systems.

This chapter also gives a brief overview of the Oracle software environment: the components of such an environment, the characteristics of those components, and what can you do with those components.

The last section of this chapter introduces seven sample tables, which are used in the examples and exercises throughout this book to help you develop your SQL skills. In order to be able to formulate and execute the correct SQL statements, you'll need to understand the structures and relationships of these tables.

This chapter does not cover any object-relational database features. Chapter 12 discusses the various Oracle features in that area.

## 1.1 Information Needs and Information Systems

Organizations have business objectives. In order to realize those business objectives, many decisions must be made on a daily basis. Typically, a lot of *information* is needed to make the right decisions; however, this information is not always available in the appropriate format. Therefore, organizations need formal systems that will allow them to produce the required information, in the right format, at the right time. Such systems are called *information systems*. An information system is a simplified reflection (a *model*) of the real world within the organization.

Information systems don't necessarily need to be automated—the data might reside in card files, cabinets, or other physical storage mechanisms. This data can be converted into the desired information using certain procedures or actions. In general, there are two main reasons to automate information systems:

- **Complexity:** The data structures or the data processing procedures become too complicated.
- **Volume:** The volume of the data to be administered becomes too large.

If an organization decides to automate an information system because of complexity or volume (or both), it typically will need to use some database technology.

The main advantages of using database technology are the following:

- **Accessibility:** Ad hoc data-retrieval functionality, data-entry and data-reporting facilities, and concurrency handling in a multiuser environment
- **Availability:** Recovery facilities in case of system crashes and human errors
- **Security:** Data access control, privileges, and auditing
- **Manageability:** Utilities to efficiently manage large volumes of data

When specifying or modeling information needs, it is a good idea to maintain a clear separation between *information* and *application*. In other words, we separate the following two aspects:

- **What:** The information *content* needed. This is the *logical* level.
- **How:** The desired *format* of the information, the way that the results can be derived from the data stored in the information system, the minimum performance requirements, and so on. This is the *physical* level.

Database systems such as Oracle enable us to maintain this separation between the “what” and the “how” aspects, allowing us to concentrate on the first one. This is because their implementation is based on the *relational model*. The relational model is explained later in this chapter, in Sections 1.4 through 1.7.

## 1.2 Database Design

One of the problems with using traditional third-generation programming languages (such as COBOL, Pascal, Fortran, and C) is the ongoing maintenance of existing code, because these languages don’t separate the “what” and the “how” aspects of information needs. That’s why programmers using those languages sometimes spend more than 75% of their precious time on maintenance of existing programs, leaving little time for them to build new programs.

When using database technology, organizations usually need many database applications to process the data residing in the database. These database applications are typically developed using fourth- or fifth-generation application development environments, which significantly enhance productivity by enabling users to develop database applications *faster* while producing applications with *lower maintenance* costs. However, in order to be successful using these fourth- and fifth-generation application development tools, developers must start thinking about the structure of their data first.

It is *very* important to spend enough time on designing the data model *before* you start coding your applications. Data model mistakes discovered in a later stage, when the system is already in production, are very difficult and expensive to fix.

### Entities and Attributes

In a database, we store facts about certain objects. In database jargon, such objects are commonly referred to as *entities*. For each entity, we are typically interested in a set of observable and relevant properties, commonly referred to as *attributes*.

When designing a data model for your information system, you begin with two questions:

1. Which entities are relevant for the information system?
2. Which attributes are relevant for each entity, and which values are allowed for those attributes?

We'll add a third question to this list before the end of this chapter, to make the list complete.

For example, consider a company in the information technology training business. Examples of relevant entities for the information system of this company could be course attendee, classroom, instructor, registration, confirmation, invoice, course, and so on. An example of a partial list of relevant attributes for the entity **ATTENDEE** could be the following:

- Registration number
- Name
- Address
- City
- Date of birth
- Blood group
- Age
- Gender

For the **COURSE** entity, the attribute list could look as follows:

- Title
- Duration (in days)
- Price
- Frequency
- Maximum number of attendees

---

■ **Note** There are many different terminology conventions for entities and attributes, such as *objects*, *object types*, *types*, *object occurrences*, and so on. The terminology itself is not important, but once you have made a choice, you should use it consistently.

---

## Generic vs. Specific

The difference between *generic* versus *specific* is very important in database design. For example, common words in natural languages such as *book* and *course* have both generic and specific meanings. In spoken language, the precise meaning of these words is normally obvious from the context in which they are used.

When designing data models, you must be very careful about the distinction between generic and specific meanings of the same word. For example, a course has a title and a duration (generic), while a specific course offering has a location, a certain number of attendees, and an instructor. A specific book on the shelf might have your name and purchase date on the cover page, and it might be full of your personal annotations. A generic book has a title, an author, a publisher, and an ISBN code. This means that you should be careful when using words like *course* and *book* for database entities, because they could be confusing and suggest the wrong meaning.

Moreover, we must maintain a clear separation between an entity itself at the generic level and a specific occurrence of that entity. Along the same lines, there is a difference between an entity *attribute* (at the generic level) and a specific *attribute value* for a particular entity occurrence.

## Redundancy

There are two types of data: base data and derivable data. *Base data* is data that cannot be derived in any way from other data residing in the information system. It is crucial that base data is stored in the database. *Derivable data* can be deduced (for example, with a formula) from other data. For example, if we store both the age and the date of birth of each course attendee in our database, these two attributes are mutually derivable—assuming that the current date is available at any moment.

Actually, every question issued against a database results in derived data. In other words, it is both undesirable and impossible to store all derivable data in an information system. Storage of derivable data is referred to as *redundancy*. Another way of defining redundancy is storage of the same data more than once.

Sometimes, it makes sense to store redundant data in a database; for example, in cases where response time is crucial and in cases where repeated computation or derivation of the desired data would be too time-consuming. But typically, storage of redundant data in a database should be avoided. First of all, it is a waste of storage capacity. However, that's not the biggest problem, since gigabytes of disk capacity can be bought for relatively low prices these days. The challenge with redundant data storage lies in its ongoing maintenance.

With redundant data in your database, it is difficult to process data manipulation correctly under all circumstances. In case something goes wrong, you could end up with an information system containing internal contradictions. In other words, you would have *inconsistent* data. Therefore, redundancy in an information system results in ongoing consistency problems.

When considering the storage of redundant data in an information system, it is important to distinguish two types of information systems:

- Online transaction processing (OLTP) systems, which typically have continuous data changes and high volume
- Decision support (DSS) systems, which are mainly, or even exclusively, used for data retrieval and reporting, and are loaded or refreshed at certain frequencies with data from OLTP systems

In DSS systems, it is common practice to store a lot of redundant data to improve system response times. Retrieval of stored data is typically faster than data derivation, and the risk of inconsistency, although present for load and update of data, is less likely because most DSS systems are often read-only from the end user's perspective.

## Consistency, Integrity, and Integrity Constraints

Obviously, consistency is a first requirement for any information system, ensuring that you can retrieve reliable information from that system. In other words, you don't want any *contradictions* in your information system.

For example, suppose we derive the following information from our training business information system:

- Attendee 6749 was born on February 13, 2093.
- The same attendee 6749 appears to have gender Z.
- There is another, different attendee with the same number 6749.
- We see a course registration for attendee 8462, but this number does not appear in the administration records where we maintain a list of all persons.

In none of the above four cases is the consistency at stake; the information system is unambiguous in its statements. Nevertheless, there is something wrong because these statements do not conform to common sense.

This brings us to the second requirement for an information system: *data integrity*. We would consider it more in accordance with our perception of reality if the following were true of our information system:

- For any course attendee, the date of birth does not lie in the future.
- The gender attribute for any person has the value M or F.
- Every course attendee (or person in general) has a unique number.
- We have registration information only for existing attendees—that is, attendees known to the information system.

These rules concerning database contents are called *constraints*. You should translate all your business rules into formal integrity constraints. The third example—a unique number for each person—is a primary key constraint, and it implements *entity integrity*. The fourth example—information for only persons known to the system—is a foreign key constraint, implementing *referential integrity*. We will revisit these concepts later in this chapter, in Section 1.5.

Constraints are often classified based on the lowest level at which they can be checked. The following are four constraint types, each illustrated with an example:

- **Attribute constraints:** Checks attributes; for example, “Gender must be M or F.”
- **Row constraints:** Checks at the row level; for example, “For salesmen, commission is a mandatory attribute.”
- **Table constraints:** Checks at the table level; for example, “Each employee has a unique e-mail address.”
- **Database constraints:** Checks at the database level; for example, “Each employee works for an existing department.”

In Chapter 7, we'll revisit integrity constraints to see how you can formally specify them in the SQL language.

At the beginning of this section, you learned that information needs can be formalized by identifying which entities are relevant for the information system, and then deciding which attributes are relevant for each entity. Now we can add a third step to the information analysis list of steps to produce a formal data model:

1. Which entities are relevant for the information system?
2. Which attributes are relevant for each entity?
3. Which integrity constraints should be enforced by the system?

## Data Modeling Approach, Methods, and Techniques

Designing appropriate data models is not a sinecure, and it is typically a task for IT specialists. On the other hand, it is almost impossible to design data models without the active participation of the future end users of the system. End users usually have the most expertise in their professional area, and they are also involved in the final system acceptance tests.

Over the years, many methods have been developed to support the system development process itself, to generate system documentation, to communicate with project participants, and to manage projects to control time and costs. Traditional methods typically show a strict phasing of the development process and a description of what needs to be done in which order. That's why these methods are also referred to as *waterfall* methods. Roughly formulated, these methods distinguish the following four phases in the system development process:

1. **Analysis:** Describing the information needs and determining the information system boundaries
2. **Logical design:** Getting answers to the three questions about entities, attributes, and constraints, which were presented in the previous section
3. **Physical design:** Translating the logical design into a real database structure
4. **Build phase:** Building database applications

Within the development methods, you can use various *techniques* to support your activities. For example, you can use diagram techniques to represent data models graphically. Some well-known examples of such diagram techniques are Entity Relationship Modeling (ERM) and Unified Modeling Language (UML). In the last section of this chapter, which introduces the sample tables used throughout this book, you will see an ERM diagram that corresponds with those tables.

Another example of a well-known technique is *normalization*, which allows you to remove redundancy from a database design by following some strict rules.

*Prototyping* is also a quite popular technique. Using prototyping, you produce “quick and dirty” pieces of functionality to simulate parts of a system, with the intention of evoking reactions from the end users. This might result in time-savings during the analysis phase of the development process, and more important, better-quality results, thus increasing the probability of system acceptance at the end of the development process.

*Rapid application development* (RAD) is also a well-known term associated with data modeling. Instead of the waterfall approach described earlier, you employ an iterative approach.

Some methods and techniques are supported by corresponding computer programs, which are referred to as computer-aided systems engineering (CASE) tools. Various vendors offer complete and integral support for system development, from analysis to system generation, while others provide basic support for database design even though their products are general-purpose drawing tools (Microsoft Visio is an example).