Index

Symbols and Numerics

* (asterisk), 159
-- (double dash), 46, 207
/ (forward slash), 35
%FOUND variable, 144, 145, 146, 148, 156
%ISOPEN variable, 144, 146, 148, 156
<< >> (labeling blocks), 43
/* */ (multi-line comment), 46
%NOTFOUND variable, 144, 146, 148, 156
1NF (First Normal Form), 14
%ROWCOUNT cursor variable, 144, 146, 155, 156
%ROWTYPE declaration, 131–132, 143–144, 333–334
2NF (Second Normal Form), 14–15
; (semicolon), 35
' (single quote), 50–51
3NF (Third Normal Form), 15–16
%TYPE command, 212
_ (underscore), 191

A abbreviations, 190, 199
access to package, controlling, 172–173
accessing
object table, 271
same object repeatedly, 347
status of cursor, 144–148
account
unlocking or locking, 33
user, connecting to database and, 31–32
activity audit, 308–309
actual parameter, 63
Ada programming language, 1, 18
ADD_MONTHS function, 239–240
AFTER EVENT trigger, 176–177
Agile approach
development teams and, 349–350
pair programming and, 350
rapid delivery and, 350–351
test-first approach and, 351
algorithm
evaluating, 339
refactoring, 347
alias for column, 216
anonymous block
defining cursor in, 139
description of, 34–36, 42
naming, 56–57
nesting, 43
API (Application Programming Interface), 68, 373
Application Development Framework - Business Components (Oracle), 169, 182
application development software, 17
application server
placing code in, 183–185
setting up Oracle environment and, 24
software, setting up Oracle environment and, 24
architecture, system
checking while writing code, 339–340
understanding, 336–337, 340
architecture, testing, 345
array
associative, 198, 280–283
variable size (VARRAY), 197, 198, 272–275
ASCII function, 244–245
assertion, 344
assigning
code to user-defined exception, 115–116
value in record, 262–265
value to variable, 48–49
associative array, 198, 280–283
asterisk (*), 159
attribute
definition of, 13
dependency of, 14, 15
of exception, 106
of object, 267
auditing with autonomous transaction, 300–302, 308–310
autonomous transaction
activity audit and, 308–309
auditing, security, and, 300–302
data changes and, 303–305
description of, 291, 298
exceptions and, 307–308
functions and, 368–369
locks and, 306–307
nested transaction compared to, 302–303
query audit and, 309–310
self-mutating problems, 310–312
syntax for, 299–300
using with DDL commits, 368
avoiding conflicts of variable scope, 51–52

• B •

batch routine
database server and, 24
description of, 21
INSTEAD OF trigger view and, 181
BEFORE EVENT trigger, 176
BEGIN statement, 208, 210
Beginning Database Design (Powell), 16
best practices
Agile approach and, 349–351
coding, 336–338
importance of, 335–336
keeping up-to-date, 352–354
testing code, 343–349
writing code with best practices in mind, 338–343
BFILE column
description of, 254–255
loading data to BLOB using, 257–258
loading data to CLOB using, 256–257
populating, 255–256
binary large object (BLOB)
description of, 254
loading page to, 257–258
BINARY_DOUBLE datatype, 225–226
BINARY_FLOAT datatype, 225–226
BINARY_INTEGER datatype, 225–226
bind variable, 317–324
BLOB (binary large object)
description of, 254
loading page to, 257–258
Boolean datatypes, 241–242
Boolean expression, 380–382
breakpoint, setting, 343
building on the fly
DDL, 325–327, 366–367
SQL, 316–325
built-in functions
character datatypes and, 244–250
dates and, 237–241
numeric datatypes and, 228–229
using, 386–388
built-in packages
DBMS_RANDOM, 230
overview of, 83–84
BULK COLLECT command
dynamic SQL and, 329–330
NO_DATA_FOUND exception and, 360
using, 284–287
bulk operations, 283–287, 348, 352
business function for code, 336–337
business logic, placing, 185–186

• C •

C programming language, 1, 18, 348
caching calculated data, 365
call
overloading, 76–78
to subprogram, resolving, 78–80
calling cursor declared in different
package, 141
candidate key, 13
capitalization
coding standards for, 211
naming standards and, 190, 191
case sensitivity
identifiers and, 44
text literal and, 51
CASE statement, 89–91
casting collection to table, 278–279
CHAR datatype, 242–244
character datatypes
  built-in functions and, 244–250
  CHAR versus VARCHAR2 datatypes,
  242–244
description of, 221, 242
character large object (CLOB)
description of, 254
loading data to, 256–257
string operations and, 258–259
character literal, 50
character set, 43
CHR function, 244–245
client computer and software, setting up
  Oracle environment and, 24
CLOB (character large object)
description of, 254
loading data to, 256–257
string operations and, 258–259
closing explicit cursor, 382–383
code. See also coding; coding standards;
  storing code in database; testing code;
  writing code
  assigning to user-defined exception,
  115–116
  commenting, 342, 395–396
  compiling, in database package, 170–172
  compiling, while writing, 341–342
  cutting and pasting, 391–392
  having someone review, 205–206, 339
  managing, 166
  modular, 136
  placing in database, 165–166
  readability of, 393–395
  recursive, 80–82
  running, 56–59
  self-documenting, 395
  server-side, pros and cons of, 20–21
code library, 340
coding. See also code; coding standards
  built-in functions and, 386–388
  communicating effectively, 337
  creating specification, 337–338, 345
  thinking through program, 336
  understanding big picture, 336–337
coding standards
  for capitalization, 211
  for comments, 207–210
  for constant value, 202–205
  for data conversion for dates, 213
  for data element, 206
  for datatype declarations, 211–213
  for global variables, 210
  for indentation, 210–211
  for line length, 213
  overview of, 201–202
  for program units, 205–206
  for SQL, 214–217
  for synonyms, 213–214
collection
  associative arrays, 280–283
description of, 271–272
  naming standards for, 197–198
  nested table, 275–279
  V ARRAY, 272–275
column
  alias for, 216
  BFILE, 254–258
  order, and DML statement, 360–362
  prefixing name of, 215–216
  comma, placing, 206
  command line interface and SQL*Plus, 28
  comment
    coding standards for, 207–210
    description of, 45–46
    multi-line, 46
    single-line, 46, 207
    commenting out code, 342–343, 395–396
  COMMIT command, 291, 293–294, 325–327
    implicit, 366–369
  communicating effectively, 337
  compilation error
    interpreting and fixing, 73–76
    spotting, 57
  compiler hints and directives, 82–83
  compiling code
    in database package, 170–172
    while writing, 341–342
  composite datatypes, 221
  conditional statement
    CASE, 89–91
    description of, 85
    grouping, 380–382
    IF...ELSE, 87–89
    IF...THEN, 86–87
    NULL value, comparing with, 91–95
    working with, 95–97
conferences, Oracle, 353–354
constant keyword, 47
constant value, coding standards for, 202–205
control structure. See conditional statement; loop
controlling
access to package, 172–173
scope with variable declaration, 52
when trigger fires, 176–177
conventional wisdom, evaluating, 352–353
converting datatype implicitly, 388–390
cost of testing, 344
CREATE OR REPLACE PACKAGE
command, 70
CREATE OR REPLACE procedure/function
command, 68–69
cursor. See also explicit cursor; implicit cursor
accessing status info using variables, 144–148
declaring, 128–129, 137–141
to loop through multiple records, 132–133
new wisdom regarding, 352
passing parameter to, 134–137
placing in nested loop, 133–134
retrieving information with too many, 347–348
to return more than one piece of information, 129–132
SELECT * statement and, 214–215
updating records fetched from, 148–150
CURSOR FOR loop
comparing to cursor with LOOP
command, 150–152
description of, 150
difficulty using, 153–155
exception handling and, 152–153, 385
to know what record is processing, 155
cursor variable, 144–145, 324–325
cutting and pasting code, 391–392

Data Definition Language (DDL)
building on the fly, 325–327, 366–367
description of, 17
data element, coding standards for, 206
Data Manipulation Language (DML), 17
data model, 337
logical, 12
physical, 12
database. See also database connection, establishing; placing code in database; storing code in database
design terminology, 12–13
installing, 27
logic implemented in, 180
normalization of, 13–16
procedures and functions, 68
relational, 9–12
Relational Database Management Systems (DBMS), 16
setting up Oracle environment and, 24
storing code inside, 36
database administrator
DBMS and, 16
resources for, 2
database application developer, 16
database connection, establishing
operating services, checking, 32
password, resetting, 33–34
setting up server to communicate, 34
unlocking or locking account, 33
user accounts and, 31–32
username and, 32
database designer, 16
Database Development For Dummies (Taylor), 337
database server, setting up Oracle environment and, 24
database-level trigger, 19, 20
datatype
converting implicitly, 388–390
defining, 48
defining own, 260–271
description of, 221
functions and, 157
generic declarations of, 211–213
groups of, 221–222
internal, 236–237
large objects (LOBs), 253–259

• D •
data caching, 182
data collections, 271–280
data consistency, transactions and, 292–293
naming, 196–197
Native Dynamic SQL and, 328–334
numeric, 222–229
string, 242–250
user-defined subtypes, 259–260
of variable, 211–213
datatype family of parameters, 78–79
delete. See also date/time information
    explicit data conversion for, 213
    as number, 375–376
Date, Chris J. (author), 16
DATE datatype, 229–234
date/time information
    built-in functions and, 237–241
    data conversion, 213
    DATE datatype, 229–234, 375–376
    INTERVAL datatype, 236–237
    TIMESTMAP datatype, 234–235
    TIMESTAMP WITH TIME ZONE datatype, 235–236
DBMS (Relational Database Management Systems)
    description of, 16
    setting up Oracle environment and, 24
DBMS_JAVA package, 84
DBMS_JOB package, 84
DBMS_OUTPUT package, 83, 340, 343
DBMS_OUTPUT.PUT_LINE command, 35–36, 56
DBMS_RANDOM package, 84, 230
DBMSUTILITY package, 84
DDL (Data Definition Language)
    building on the fly, 325–327, 366–367
description of, 17
deadlock, 112, 298, 325
debugging
    adding information to debugging messages, 378
double dash (––), 46, 207
    ELSIF statement, 88–89
dependency of attribute, 14, 15
derogency and Agile approach, 350–351
deterministic behavior of function, 159–160
direct declaration, 46–47
directory, 255
discarding code, 341
dynamic, definition of, 313
dynamic SQL
    bind variables and, 317–324
    building DDL on the fly, 325–327
    building SQL on the fly, 316–317
cursor variables and, 324–325
datatypes and, 328–334
    EXECUTE IMMEDIATE command, 314–316
    overview of, 313–314
quoted strings and, 327–328

E

elements. See lexical set of elements
ELSIF statement, 88–89
Embarcadero, RapidSQL, 31
embedding code in database view, 21
END statement, 58, 208, 210
endless loop, 81, 384–386
enforcing naming standards, 199–200
Enterprise Edition, 27
entity, 13
error code prefix, 110

error message
  compilation and, 342–343
  including in user-defined exception, 116–117
  listing example, 58
  parsing and, 56–57

errors and coding standards, 201–202, 217

exception. See also exception handler
  autonomous transaction and, 307–308
  commands to support, 106
  description of, 105
  handling, 122–126
  naming standards for, 196
  predefined, 111–112
  propagation of, 118–126
  types of, 110–111
  unhandled, 108
  user-defined, 105, 113–118, 196
  WHEN OTHERS exception, 370–372

exception code, 110

exception handler
  adding, 107–110
  avoiding exception raised in, 124–126
  CURSOR FOR loop and, 152–153, 154–155
  description of, 106
  implicit cursor and, 143
  NOCOPY hint and, 363–364
  SELECT INTO command and, 357–360
  WHEN OTHERS THEN NULL and, 217, 377–378
  writing, 126

exception message, 111

exception name, 111

exception section of anonymous block, 35

EXECUTE IMMEDIATE command, 314–316

EXIT command, 98, 101–102, 104

EXIT WHEN statement, 98

exiting loop, 385

explicit column list, 214–215

explicit commit, 294

explicit cursor
  checking status of, 145–146
  closing, 382–383
  conventional wisdom and, 352
  description of, 128
  implicit compared to, 374–375

explicit data conversion for dates, 213

expression
  Boolean, 380–382
  description of, 53
  parentheses and, 216, 381–382
  regular, 250–251
  simple, example of, 54

external large object, 254–255

EXTRACT function, 237–238

• F •

Feuerstein, Steven (PL/SQL expert), 5, 211, 353

file, naming standards for, 198–199

A First Course in Database Systems (Ullman and Widom), 16

First Normal Form (1NF), 14

FOR loop, 102–104, 385

foreign key, 11, 12

formal parameter, 63–67

format mask, 231–234

forward slash (/), 35

%FOUND variable, 144, 145, 146, 148, 156

full rollback, 294–295

function. See also built-in functions
  autonomous transaction and, 368–369
  description of, 21, 59
  getter, 203, 347
  group, 380
  naming standards for, 193–194
  performance issues and, 158–161, 364–366
  referencing in SQL, 155–160
  RETURN statement and, 369–370
  returning value with, 61–62
  setter, 203, 347
  storing in database, 68–69

functional programming language, 41–42

• G •

generic programming language, 41

getter function, 203, 347

global constant, 202–203
global variable
coding standards and, 210
description of, 52–53
passing parameters compared to, 372–373
storing in database table, 168–169
storing in middle tier, 169–170
“good enough is best” expression, 348–349
grouping functions, 380
grouping logical conditions, 380–382
GUI tool
  SQL*Plus, 28–29
third-party, 30–31

• H •

handling exception. See exception handler
hard coding, 202–206
hardware requirements, 25–26
header
  commenting code and, 396
  of program unit, defining cursor in, 138
HR (Human Resources) schema, 38

• I •

IDE (Integrated Development Environment), 343
identifier, 44
IF . . . ELSE statement, 87–89
IF . . . THEN statement, 86–87
implicit commit
  DDL statements and, 325–327, 366–369
  firing, 294
implicit cursor
  checking status of, 146–148
  conventional wisdom and, 352
description of, 142
  handling exceptions in, 143
  retrieving single row using, 142
  returning, into record, 143–144
  using, 373–375
implicit rollback, 297–298
implicitly converting datatype, 388–390
IN OUT parameter, 65–67, 362–364
IN parameter, 63–64
indenting code, 210–211
Independent Oracle Users Group, 5, 353
INDEX BY BINARY_INTEGER associative array, 283
infinite loop, 81, 384–386
init-caps method, 191
INSERT statement
  DML statements and, 360–362
element of, 206
  records and, 265–267
installing
  database, 27
Oracle environment, 23
instance
  of entity, 13
  of object, 269
INSTEAD OF trigger, 72–73, 177–182
INSTR function, 245–248
integer literal, 49–50
Integrated Development Environment (IDE), 343
interface, 30. See also GUI tool
internal large object, 254
Internet Application Server, 24
interpreter and running code, 56
INTERVAL datatype, 236–237
INVALID_NUMBER exception, 111
%ISOPEN variable, 144, 146, 148, 156
iSQL*Plus, 29
iteration. See loop

• J •

Java, influence on naming standards, 188
Java (J2EE) environment, 20–21, 183
JavaServer Faces environment, 21
JSP/Struts environment, 21

• K •

Koletzke, Peter, Oracle Developer: Advanced Forms & Reports, 22

• L •

labeling block (<<< >>>), 43
large object (LOB)
description of, 222, 253
large object (LOB) (continued)
  external, 254–255
  internal, 254
LAST_DAY function, 240
lexical set of elements
  comments, 45–46
  delimiters, 44–45
  identifiers, 44
  literals, 45
  overview of, 43
line length, 213
listings
  ADD_MONTHS function, 239–240
  AFTER EVENT trigger, 176
  assigning Boolean variables, 241–242
  assigning error message for user-defined exception, 117
  assigning fraction to integer, 223
  assigning name to predefined exception code, 113
  assigning record variables, 265
  assigning value to variable, 48–49
  associative array, 198
  associative array, indexing, 282–283
  associative array, using, 281
  author comment block, 207
  autonomous transaction, 303
  autonomous transaction, code using, 311
  autonomous transaction syntax, 299
  autonomous transaction, with data changes, 304–305
  autonomous transaction, with DDL commits, using, 368
  backup routine procedure, 367
  bind variables, multiple, using, 319–320
  bind variables, reusing in dynamic PL/SQL, 321–322
  bind variables, reusing in dynamic SQL, 322
  bind variables, using in dynamic SQL, 319
  BULK COLLECT command, using, 285
  BULK COLLECT command, using with limit, 286–287
  bulk query into nested tables, 285–286
  caching calculated data, 365
  calling cursor declared in different package, 141
  CHAR datatype, 243
  CHR and ASCII functions, 245
  CLOB string operations, 259
  code assigned to user-defined exception, 115–116
  code, badly written, 393
  code, somewhat improved, 393–394
  code, well-written, 394–395
  code, with conditional control to avoid exception, 108
  code, with explicit handler for predefined exception, 108–110
  code, with no exception handler, 107
  code, with no repeated sections, 392
  combining ways of assigning variable values, 263
  commented code, correct, 395–396
  comparisons using NULL value, 91–92
  compiling stored procedures, 73–74
  condition using CASE statement, 90
  create type illustrated, 328–329
  cross datatype, stable example, 389
  cross datatype, unstable example, 389
  cursor variables and dynamic SQL, 324
  cursors in nested loops, 134
  DATE format mask, 233
  dates, finding difference between, 375
  declaring cursor, 128
  declaring cursor, in package body, 140
  declaring cursor, within procedure, 138
  declaring variable by reference, 47–48
  decreasing size of array, 274–275
  defining cursor in anonymous block, 139
  defining record type for cursor using $ROWTYPE declaration, 132
  difference between explicit and implicit cursors, 374
  displaying date and time, 230–231
  DML trigger example, 71–72
  DML using record variables, 266
  dynamic SQL, 315
  dynamic SQL, reviewing, 315–316
  dynamic SQL, using, 316–317
  electronic funds transfer, performing, 292
  ELSE, using in condition statement, 87–88
  ELIF statement, 88
  endless loop example, 384
error from COMMIT in trigger, 301
error messages, 58
exception handler, adding to CURSOR FOR loop, 154–155
exceptions raised in exception handler, 125–126
exiting from nested loop, 100
explicit commit, 294
explicit cursor, 145–146
explicit cursor, correctly written, 382–383
explicit date conversion, 213
explicit record type, 261–262
format mask failure, 233–234
format masks, 232
formatting address, correct code, 390
formatting address, incorrect code, 390
function example, 61–62
globals stored in package, 203–204
grouping conditions, 381
handling uncommitted changes using
p_rollback_test, 307
handling unnamed exception, 112
hard-coded data value, 202–203
hard-coded rule parameter, 203
implicit commits in DDL statements (bug fixed), 326–327
implicit commits in DDL statements (with bug), 326
implicit declaration, 262
INDEX BY BINARY_INTEGER, using, 283
insert by using RECORD type variable, 362
INSERT statement, improper, 361
INSERT statement, safe, 361
INSTEAD OF trigger view, 178–180
INSTR function, 247–248
integer and real literals, examples of, 50
LAST_DAY function, 240
limiting function returns, 369–370
loading data to CLOB, 256–257
loading page to BLOB, 258
locking example, 306
login function package, 167–168
loop, exiting, 385
loop, SQL replacement for regular, 385–386
looping through cursor using CURSOR FOR loop, 151
looping through cursor using LOOP command, 151
looping through records in cursor, 133
LPAD and LTRIM functions, 249–250
named notation, 66–67
nested loop, 99
nested table, creating, 276
nested table, to return employee list, 277–278
nested WHILE loop, 101
NULL bind variable, 322–323
NUMBER and BINARY_INTEGER datatypes, 226
NUMBER datatype, 223, 224–225
NVL function, 93–94
object type example, 269
OUT parameter, 64–65, 320–321
parameter naming example, 195–196
passing parameters in cursor, 135
passing variables without copying, 264
placing data elements on separate lines, 206
p_log_audit, as autonomous transaction, 301
p_log_audit, non-working, 300
pointer, creating, 256
procedure example, 59–60
propagating exceptions between program units, 118–120
proper naming standards usage, 197
query audit function, 310
raising exception, in declaration section, 124
raising exception, local PL/SQL block, 123
record datatype, using, 266–267
referencing package specifications, 171
referencing subtype, 260
replacing hard-coded data value with reference, 204–205
replacing rule parameter with reference, 205
retrieving cursor variables with record variable, 130–131
returning cursor variables quick and easy way, 130
rollback, code requiring, 297
rollback, full, explicit, 295
listings (continued)
rollback, partial, 295–296
row-level trigger, 175–176
savepoint, adding, 296
SELECT FOR UPDATE command, 149–150
SELECT * in cursor, 214–215
SELECT INTO, with correct result, 359–360
SELECT INTO, with wrong result, 359
setting precision and scale, 224
simple condition statement, 86, 87
simple expression, example of, 54
single quote character, using as part of
text string, 51
social security number, better routine to
test check, 387–388
social security number, routine to check, 387
SQL objects, using, 270
statement-level trigger, 175
stopping procedure execution, 61
storing exception message, 371
SUBSTR function, 246–247
subtypes example, 212–213
table prefixing of columns, 215–216
time zone variable, declaring, 235–236
TIMESTAMP datatype, 234–235
traditional condition statement, 89–90
TRUNC and ROUND functions, 238–239
%TYPE command illustrated, 212
user-defined exception, 114–115
validating rows, 363
variables, using in static SQL, 318
VARRAY, 198, 273–274
weak REF CURSOR construct, 332–333
well-commented code, 208–210
WHILE loop, 101
wrapper package, 173
literal
character, 50
description of, 45
string, 50–51
load testing, 345–346
LOB (large object)
description of, 222, 253
external, 254–255
internal, 254
local PL/SQL block, defining cursor in, 138–139
local variable, 52–53
locking
account, 33
autonomous transaction and, 306–307
INSTEAD OF trigger view, 181–182
record, 150
logic implemented in database, 180, 185–186
logical data model, 12
logical expression, parentheses and, 216
logical operators
conditions and, 95
description of, 55–56
updating record using, 148–150
Loney, Kevin, Oracle Database 10g: The Complete Reference, 19, 140
loop
CURSOR FOR loop, 150–155, 385
description of, 85
endless, 81, 384–386
FOR loop, 102–104, 385
nested, 99–100, 133–134
simple, 97–99
types of, 97
WHILE loop, 100–102
LOOP command, 150–151
looping through multiple records, 132–133
Lowe, Doug, Networking For Dummies, 7th Edition, 25
LPAD function, 249–250
LTRIM function, 249–250

M

maintaining code, 341, 392
managing code, 166
mathematical expression, parentheses and, 216
McCullough-Dieter, Carol, Oracle 9i For Dummies, 2
memory requirements of code, 339
method, of object, 267
middle tier, storing global variable in, 169–170
middle-tier approach
优点和缺点，183–185
description of, 183–184
混合 notation, 67
模块型代码, 136
MONTHS_BETWEEN function, 240–241
multi-line comment (/* */), 46

• N •
named block, 42
named notation, 66–67
naming standards
application development tools influence on, 189
for collections, 197–198
enforcing, 199–200
for exceptions, 196
for files, 198–199
Java influences on, 188
for objects, 190–191
Oracle database influences on, 188
overview of, 187–188
for parameters, 194–196
for program units, 193–194
setting for common code elements, 189
for user-defined datatypes, 196–197
for variables, 191–193
Native Dynamic SQL
bind variables and, 317–324
building DDL on the fly, 325–327
building SQL on the fly, 316–317
cursor variables and, 324–325
datatypes and, 328–334
EXECUTE IMMEDIATE command, 314–316
overview of, 313–314
quoted strings and, 327–328
nested loop
description of, 99–100
placing cursor in, 132–133
nested table, 275–279
nested transaction, 302–303
nesting anonymous blocks, 43
.NET environment, 20–21, 183

network, setting up Oracle environment and, 24–25
Networking For Dummies, 7th Edition (Lowe), 25
NOCOPY hint
passing parameter and, 82, 362–364
passing record variable and, 264
normalized database
description of, 10
First Normal Form (1NF), 14
Second Normal Form (2NF), 14–15
Third Normal Form (3NF), 15–16
usefulness of, 13
% NOTFOUND variable, 144, 146, 148, 156
NULL bind variable, 322–323
NULL value
comparing with, 91–95
DEFAULT clause and, 47
handling, 379–380
NUMBER datatype, 222–225
precision, 222
scale, 222
number, date as, 375–376
numeric datatypes
BINARY_FLOAT and BINARY_DOUBLE, 225–226
BINARY_INTEGER, 225–226
built-in functions and, 227–229
description of, 221, 222
NUMBER, 222–225
numeric literal, 49–50
NVL function, 93–94

• O •
object
creating, 328–329
naming standards for, 190–191
repeatedly accessing same, 347
object identifier, 270
object table, 270, 271
object type, 267–271
object type abbreviations, 199
object type body, 268
object type specification, 268
ODTUG-SQLPLUS-L list, 354
OE (Order Entry) schema, 38
OPEN/FETCH/CLOSE sequence, 133, 150, 152, 153–154
OPEN...FOR construct, 331–333
operand, 53
operating services, checking, 32
operating system requirements, 25
operators
description of, 53–54
logical, 55–56, 95, 148–150
precedence of, 54–55
ORA error code prefix, 110
Oracle. See also Oracle environment
Application Development Framework - Business Components, 169, 182
Application Server, 17
Applications, 17
Database 10g, 27
DBMS, 17
Developer, programming for, 21–22
OpenWorld conference, 353
SQL Developer, 30, 166, 189
staying up-to-date on, 352–354
Oracle Database 10g PL/SQL Programming (Urman), 353
Oracle Database 10g: The Complete Reference (Loney), 19, 140
Oracle Developer: Advanced Forms & Reports (Koletzke and Dorsey), 22
Oracle Development Tools User Group, 5, 353
Oracle environment
hardware and software requirements for, 25–26
installing, 23
overview of, 17
setting up, 23–25
SQL in, 18
Oracle 9i For Dummies (McCullough-Dieter), 2
Oracle Technology Network (OTN), 5, 26, 27
ORDER BY statement, and parameters, 136
Order Entry (OE) schema, 38
OUT parameter
bind variables and, 320–321
overview of, 64–65
overloading call, 76–78

• p •

package
avoiding size limitations with, 174
body, defining cursor in, 139–140
built-in, 83–84, 230
calling cursor declared in different, 141
code scope and visibility in, 167–168
coding standards for, 217
commenting out code and, 343
compiling code in, 170–172
controlling access to, 172–173
description of, 69–70
naming standards for, 193–194
pkg_global, 204
as place to put code, 166
placing, for optimal performance, 173–174
referencing, 173
size limitations and, 174
storing global values in database tables, 168–169
storing global values in middle tier, 169–170
storing globals in, 203–204
value, as session-specific, 168
wrapper, 172–173
package specification (spec)
defining cursor in, 140–141
description of, 70
recompiling, 170
referencing, 171–172
PAD function, 249–250
pair programming, 350
parameter
bind variables and, 320–321
datatype family of, 78–79
declaring different number of, 76–77
IN OUT parameter, 65–67, 362–364
IN parameter, 63–64
naming standards for, 194–196
OUT parameter, 64–65
passing, global variable compared to, 372–373
passing, to cursor, 134–137
passing, to procedure, 36–37, 61, 66
returning list based on, 277–279
subprograms and, 59, 63
using different names of, 77
parentheses
in complex mathematical and logical expressions, 216, 381–382
execution order and, 55
parsing error, 56–57
partial rollback, 295
Pascal, 1, 18
passing
bind variables, 319–320
IN OUT parameter, 362–364
parameter, global variable compared to, 372–373
parameter, to cursor, 134–137
parameter, to procedure, 36–37, 61, 66
variable, from program to subprogram, 82
variable, without copying, 264
password, resetting, 33–34
performance issues
bind variables and, 317–324
built-in functions and, 388
code and, 339
functions and, 158–161, 364–366
INSTEAD OF trigger view and, 181
large objects and, 254
placing packages and, 173–174
testing code and, 345–348
writing code and, 341
performance testing, 345–346
physical data model, 12
pkg_global package, 204
placing
code in application server, 183–185
comma, 206
placing code in database
advantages of, 182
disadvantages of, 182–183
overview of, 165–166
PL/SQL (Programming Language/Structured Query Language)
database triggers and, 19–20
description of, 1, 9, 18, 41
features of, 18–19
as functional language, 41
keeping code server-side and, 20–21
lexical elements, 43
limitations of, 42
Oracle Developer and, 21–22
scripting and, 20
SQL compared to, 10
pointer, 254–255
point-in-time value, 229
positional notation, 66
Powell, Gavin, Beginning Database Design, 16
pragma autonomous transaction, 299–300
PRAGMA command, 83, 113
precedence of operators, 54–55
precision, 222
predefined exception, 111–112
prefix
for column, 215
naming object and, 190
naming parameter and, 194–195
naming procedure or function and, 193
naming variable and, 191, 192, 193
primary key, 13
procedural block, 42–43
procedural section of anonymous block, 35
procedure
declaring cursor within, 138
description of, 21, 59
naming standards for, 193–194
passing parameter to, 36–37, 66
stored, 36, 68, 165–166
storing in database, 68–69
wrapping task into, 59–61
production environment, testing code for, 345–346
program, creating. See also subprogram;
thinking through program
Oracle SQL Developer, 30
SQL*Plus and, 28–29
third-party tools, 30–31
program fundamentals
anonymous block, 34–36
passing parameters to procedures, 36–37
stored procedures, 36
program unit. See also function; package;
procedure; trigger
coding standards for, 205–206
naming standards for, 193–194
packaging stored, 217
programming language, PL/SQL as, 41–42
Programming Language/Structured Query Language. See PL/SQL propagation of exceptions avoiding exceptions raised in declaration part and exception handler, 124–126 handling exceptions without halting program, 122–123 overview of, 118–122 proving code with test case, 340 purchase order table, example of, 11–12

• Q •

Quality Assurance team, 343 query audit, 309–310 Quest Software SQL Navigator, 31, 166 Toad, 31, 166 utPLSQL, 345 quoted string, 327–328

• R •

RETURNING INTO clause, 262–263
reusing
  bind variable, 321–322
  code, 59
ROLLBACK command, 291, 294–298
ROUND function, 238
routine
  batch, 21, 24, 181
  backup, 367
  limiting size of, 342–343
  tuning performance of, 346–348
row, validating, 363
%ROWCOUNT cursor variable, 144, 146, 155, 156
row-level trigger, 175–176
%ROWTYPE declaration, 131–132, 143–144, 333–334
rules of normalization
  description of, 10, 13
  First Normal Form (1NF), 14
  Second Normal Form (2NF), 14–15
  Third Normal Form (3NF), 15–16
running code
  compilation errors, 57
  identifying common mistakes, 56–57
  interpreter and, 56
  semicolon-related errors, 57–59
runtime, 78

• S •
sample data sets
  Human Resources and Order Entry
    schemas, 38
  overview of, 37
  Scott/Tiger schema, 37–38
savepoint, 295, 296
scalar datatypes. See also character
datatypes; date/time information;
numeric datatypes
  Boolean, 241–242
  description of, 221
scale, 222
schema, 27, 31
scientific notation, 50, 223
scope of variable, 51–53
Scott/Tiger schema, table descriptions,
  37–38
scripting, speed and, 20. See also code;
coding
  Second Normal Form (2NF), 14–15
  security and autonomous transaction,
    300–302
  security software, 25–26
SELECT FOR UPDATE command,
  149–150, 181
SELECT INTO command, 142–143, 357–360
SELECT * statement, 136
SELECT statement, parameters and,
  136
selector, 90
self-documenting code, 395
self-mutation issues, 310–312
semicolon (;), 35
semicolon-related error, 57–59
separating individual words, 191
server, setting up to communicate, 34
server-side code, pros and cons of, 20–21
server-side PL/SQL, 9
session-level trigger, 19–20
session-specific package value, 168
SET SERVEROUTPUT ON command, 35, 56
setter function, 203, 347
short-circuit evaluation, 95–96
simple loop, 97–99
simulating production environment,
  345–346
single quote (‘), 50–51
single-line comment, 46, 207
size limitations and package, 174
software
  application development, 17
  application server, 24
  assigning code to user-defined exception
    and, 116
  client, 24
downloading from Oracle Technology
  Network, 26
  requirements for, 25–26
  security, 25–26
testing, 344
spec (package specification)
  defining cursor in, 140–141
  description of, 70
  recompiling, 170
  referencing, 171–172
specification, code, 337–338, 345
SQL (Structured Query Language). See also Native Dynamic SQL
coding standards for, 214–217
description of, 17
features of, 19
integration with PL/SQL, 1, 18, 127–161
in Oracle environment, 18
objects, 270
PL/SQL compared to, 18
referencing functions, 155–157
stored procedure, 36, 68, 73, 165–166
SQL Developer (Oracle), 30, 166, 189
SQL For Dummies, 5th Edition (Taylor), 3, 18
SQL Navigator (Quest Software), 31, 166
SQLCODE function, 111–112
SQL*Plus, 28–29
standards, 187. See also coding standards;
naming standards
statement-level rollback, 297
statement-level trigger, 174–175
status of cursor, accessing
  explicit, 145–146
  implicit, 146–148
  variables for, 144–145
stopping execution
  of procedure, 61
  of program, handling exception without, 122–123
stored procedure, 36, 68, 165–166
storing
  exception message, 371
  large objects, 254
storing code in database
  compilation errors, 73–76
  overview of, 67–68
  packages, 69–70, 166–174
  procedures and functions, 68–69
  triggers, 71–73, 174–182
string
  datatype, 242–250
  quoted, 327–328
  string literal, 50–51
string operations and CLOB, 258–259
strong REF CURSOR construct, 331
Structured Query Language. See SQL
subprogram
  overview of, 59
  parameters of, 63–67
  resolving call to, 78–80
  returning value with function, 61–62
  wrapping task into procedure, 59–61
SUBSTR function, 245–248
subtype
  coding standards and, 212–213
  user-defined, 259–260
suffix
  naming collection and, 197
  naming object and, 190
  naming procedure or function and, 194
  naming user-defined datatype and, 196
  naming variable and, 191, 192, 193
synonyms, 213–214
syntax
  anonymous block, 34–36
  passing parameters to procedures, 36–37
  stored procedures, 36
SYSDATE, minimizing calls to, 348
system architecture
  checking while writing code, 339–340
  understanding, 336–337, 340
system trigger, 73

• T •
table
  casting collection to, 278–279
designing, 180
management of, 352
nested, 275–279
object, 270, 271
placing trigger on, 174–177
purchase order, example of, 11–12
storing global value in, 168–169
table alias, guidelines for, 215
table descriptions for Scott/Tiger schema, 37–38
table-level trigger, 19
Taylor, Allan G.
  Database Development For Dummies, 337
  SQL For Dummies, 5th Edition, 3, 18
team approach, 349–350
termination point of recursion, 81
terminology for database design, 12–13
test case, 340
test-first approach, 351
testing code
   architecture for, 345
   “good enough is best” expression and, 348–349
   overview of, 343–344
   performance and load, 345–346
   specifications and, 345
text literal, 51
thinking through program
   communicating effectively, 337
   creating code specification, 337–338
   overview of, 336
   understanding big picture, 336–337
Third Normal Form (3NF), 15–16
third-party tools for creating programs, 30–31
TIMESTAMP datatype, 234–235
TIMESTAMP WITH TIME ZONE datatype, 235–236
Toad (Quest Software), 31, 166
TO_CHAR function, 230–231, 237
TO_DATE function, 232
transaction. See also autonomous transaction
   COMMIT command and, 293–294
   description of, 291
   LOB pointers and, 259
   to maintain data consistency, 292–293
   nested, 302–303
   transaction-level rollback, 297–298
   transitive dependency, 15
TRANSLATE function, 248–249
trigger
   as autonomous transaction, 308–309
   database, 19
   DML, 71–72
   EVENT, 176–177
   INSTEAD OF, 72–73, 177–182
   naming standards for, 193–194
   placing on table, 174–177
   row-level, 175–176
   statement level, 174
   storing code in database using, 71
   system, 73
   types of, 19–20
TRIM function, 249–250
TRUNC function, 238
TRUNCATE TABLE command, 366
tuning. See performance issues
%TYPE command, 212

• U •

Ullman, Jeffrey D., A First Course in Database Systems, 16
underscore (_), 191
unlocking account, 33
unnamed exception, 112
UPDATE statement
   records and, 148, 150, 265–267
   transaction and, 293
   updating record fetched from cursor, 148–150
Urman, Scott, Oracle Database 10g PL/SQL Programming, 353
user account, connecting to database and, 31–32
user and Agile approach, 349
user groups, 5, 354
user interface application developer, 16
user-defined datatype, naming standards for, 196–197
user-defined exception
   assigning code to, 115–116
   example of, 105
   including error message in, 116–117
   naming standards for, 196
user-defined exception handling, 114–115
user-defined subtype
   description of, 259–260
   object, 267–271
   record datatype, 261–267
   username, connecting to database with, 32
   USER_SOURCE view, 74–75
   UTL_FILE package, 83–84
   utPLSQL (Quest Software), 345

• V •

validating row, 363
value. See also NULL value
   assigning in record, 262–265
   assigning to variable, 48–49
value. See also NULL value (continued)
constant, coding standards for, 202–205
getting with RETURNING clause, 323–324
global, 168–170
package, as session-specific, 168
passing variable from program into subprogram using, 82
point-in-time, 229
returning with function, 61–62
VALUE_ERROR exception, 111
VARCHAR2 datatype, 242–244
variable. See also global variable
assigning value to, 48–49
bind variable compared to, 318–319
cursor, 144–145, 156, 324–325
datatype of, 211–213
declaring, 46–48
global, 169–170, 210
listing separately, 129–130
literals and, 49–51
local, 52–53
naming standards for, 191–193
passing from program into subprogram using reference, 82
passing without copying, 264
record, 129, 130–131
scope of, 51–53
VARRAY (variable size array), 197, 198, 272–275
view
description of, 19
embedding code in database, 21
INSTEAD OF trigger, 177–182
USER_SOURCE, 74–75
view layer, placing code in, 185
view-level trigger, 19
visibility of code, 53, 103, 167

• W •

watch, setting, 343
weak REF CURSOR construct, 332
Web interface, SQL*Plus, 29
Web sites
Feuerstein, Steven, 5
Independent Oracle Users Group, 353

Oracle Development Tools User Group, 353, 354
Oracle Technology Network, 5, 26
third-party GUI tools, 31
user groups, 5
WHEN OTHERS exception, 370–372
WHEN OTHERS THEN NULL command, 113, 217, 377–378
WHEN OTHERS THEN RETURN NULL command, 113
WHERE clause and parameters, 135
WHILE loop, 100–102
white space, 217
Widom, Jennifer, *A First Course in Database Systems*, 16
words
reserved, 44, 211, 214
separating individual, 191
wrapper package, 172–173
wrapping task into procedure, 59–61
writing
API, 373
code specification, 337–338
exception, 126
useful comments, 208
writing code
checking architecture, 339–340
code libraries, 340
compiling and, 341–342
debugging and, 342–343
maintenance issues, 341
passing parameter to procedure, 36–37
performance issues, 341
proving with test cases, 340
simple program, 34–36
stored procedure, 36
stubbing out, 338–339

• Y •

Y2K crisis, 202