

Index

Symbols & Numerics

- & (ampersand), 495
- ' (apostrophe), XML entity reference for, 495
- * (asterisk) as `SELECT` wildcard, 79, 227–228, 299
- = (equals symbol) in comparison operators, 230–231
- > (greater than symbol) in comparison operators, 231 XML entity reference for, 495
- < (less than symbol) in comparison operators, 230–231 XML entity reference for, 495
- <> (not equal symbol) in comparison operators, 230
- % (percent sign) as `LIKE` wildcard, 234 using literally in `LIKE` predicate, 234–235
- “ (quotation marks), XML entity reference for, 495
- ; (semicolon) ending XML entity references, 495
- _ (underscore) as `LIKE` wildcard, 234 using literally in `LIKE` predicate, 234–235
- 0 (zero) minimum cardinality, 146–147 null values not same as, 69, 110 RAID level, 332, 335, 584 `SQLSTATE` class value (00), 384, 386 1 RAID level, 334, 335, 584 01 `SQLSTATE` class value, 384, 386 1:1 relationships. *See* one-to-one relationships
- 1:N relationships. *See* one-to-many relationships
- 1NF (First Normal Form), 104, 151, 152–153
- 1-tier driver system (ODBC), 480 02 `SQLSTATE` class value, 384, 386
- 2NF (Second Normal Form), 152, 153–154
- 2-tier driver system (ODBC), 481
- 3NF (Third Normal Form), 152, 154
- 3-option proposal, 139
- 4NF (Fourth Normal Form), 152, 155
- 5 RAID level, 334, 335, 584–585
- 5NF (Fifth Normal Form), 152, 155
- 10 RAID level, 334, 335, 585

A

- aborting transactions, 566
- ABS function, 214–215
- absolute fetches, 323
- .accdb format, 401
- Access. *See* Microsoft Access
- access modes for transactions, 348–349, 350
- access plans. *See* execution plans or access plans
- access privileges. *See* privileges
- ACID (atomicity, consistency, isolation, and durability), 347–348
- ACM Transactions on Database Systems* (Chen), 29
- action time for triggers, 471
- actions for triggers, 471
- ActiveX controls, 629
- actual cardinality of arrays, 220
- ADD COLUMN clause (`ALTER TABLE`), 78, 198, 440

- ADD CONSTRAINT clause (ALTER TABLE), 394
- ADO (ActiveX Data Objects), 400
- ADODB library, 404–405
- ADO.NET
 - embedding SQL in, 418
 - SQL Server support for, 401
- ADOX (ADO Extensions for DDL and Security) library, 405
- aggregate or set functions
 - AVG, 208
 - COUNT, 207–208, 240
 - defined, 207, 629, 635
 - MAX, 208
 - MIN, 209
 - overview, 207–209
 - SUM, 209
 - for XML documents, 510
- aliases
 - correlated subquery using, 281–282
 - defined, 306, 629
 - in equi-joins, 306
 - mixing with long form, avoiding, 306
- ALL keyword
 - with INTERSECT operations, 301
 - with UNION CORRESPONDING operations, 300
 - with UNION operations, 299
- ALL predicate (WHERE)
 - overview, 238–239
 - for quantified comparison operators, 272, 275, 276–277
 - syllogisms illustrating, 236–237
- ALTER TABLE statements
 - adding columns, 78, 198, 440
 - adding constraints, 394
 - concurrent access with, avoiding, 365
 - overview, 78, 198
 - removing columns, 78, 198, 440
- ampersand (&), 495
- AND logical connective, 245–246
- anomalies
 - cascading deletes to prevent, 174, 284
 - dangers of, 151
 - defined, 47
 - deletion, 151, 174, 284, 631
 - eliminating, 152–156, 345
 - insertion, 151, 633
 - modification, defined, 17, 151, 284, 633
 - from transitive dependencies, 154
 - update, 636
- ANSI/ISO SQL standard
 - development of, 51
 - indexes in, 178
 - for Persistent Stored Modules (SQL/PSM), 459
 - reserved words, 621–627
 - this book's compliance with, 2
 - variations in compliance with, 2, 52
- antivirus software, 338, 340–341
- ANY predicate (WHERE)
 - ambiguity with, 237
 - overview, 238–239
 - for quantified comparison operators, 272, 275
 - syllogisms illustrating, 236–237
- APIs (Application Programming Interfaces). *See also specific APIs*
 - Access support for, 400
 - DB2 support for, 401–402
 - defined, 629
 - MySQL support for, 402
 - native drivers with, 476
 - Oracle support for, 402
 - SQL Server support for, 401
- apostrophe ('), XML entity reference for, 495
- APPENDCHILDXML function (Oracle), 518–519
- applets, 629
- application layer firewalls, 340
- application (ODBC), 478

approximate numeric types, 97–98
 ARRAY type
 mapping to XML, 504–505
 INF violated by, 105
 overview, 105
 array value expressions, 220
 ASC keyword, 251, 318
 assertions
 defined, 71, 115, 629
 example, 71–72
 support generally lacking for, 72, 115
 assignment statements (SQL/PSM), 463
 asterisk (*) as *SELECT* wildcard,
 79, 227–228, 299
 ATOMIC keyword, 388, 461
 atomicity
 as ACID characteristic, 347
 atomic, defined, 347, 629
 for compound statements, 461
 in context of transactions, 65
 atomicity, consistency, isolation, and
 durability (ACID), 347–348
 attribute-centric mapping (SQL Server),
 531–532
 attributes (database). *See also* columns
 defined, 30–31, 141, 629
 as fields, 148
 functional dependencies, 60–61,
 149–150
 other terms for, 148
 attributes (XML), 494
 AUTHORIZATION clause (MODULE),
 94, 420
 AVG function, 208

B

B+ trees for indexes
 hash indexes versus, 559
 overview, 181, 589
 performance issues, 558

back end, 629
 backing up
 frequency for, 336
 full versus incremental backup, 336
 number of backups to save, 337
 NYBOT example, 335–336
 preparing for the worst, 335–336
 testing restore process, 337
 Basic language compared to SQL,
 410–411
 batch transactions, optimizing, 575
 BCNF (Boyce-Codd Normal Form),
 152, 155
 BEGIN keyword for compound
 statements, 460
 beta testing, 456
 BETWEEN predicate (*WHERE*), 231–232
 BIGINT type, 96
 binary relationships. *See also* many-to-
 many relationships; one-to-many
 relationships; one-to-one
 relationships
 converting E-R model to relational
 model, 164–168
 defined, 32, 142, 143
 bit data types, *BETWEEN* predicate with,
 232
 blank spaces versus null values, 69, 110
 BLOB (BINARY LARGE OBJECT) type, 97
 BOOLEAN type, 101, 110
 Boolean value expressions, 219–220
 Borland InterBase, 57, 72
 bottlenecks, 587. *See also* database
 tuning; performance; query tuning;
 tuning
 bounded priority inversion, 578
 Boyce-Codd Normal Form (BCNF),
 152, 155
 buffer manager, 617
 buffers, page. *See* page buffer for
 database; page buffer for
 transactions

- building database applications
 - adding constraints, 448–449
 - creating building blocks, 454–455
 - creating tables, 448
 - creating the database, 448–449
 - developing reports, 455
 - developing screen forms, 454
 - filling tables with sample data, 449–454
 - integrating the structure, 455
 - overview, 130
 - preparatory work for, 447–454
 - testing, 455–457
- Bush, Vannevar (database theorist), 9
- business rules
 - Oregon Lunar Society database, 431–432
 - overview, 40–41, 147
- Buxton, Stephen (*Querying XML*), 103, 535

C

- C language
 - embedded SQL example, 91–93
 - with MySQL, 406
 - ODBC connection example, 486–487
 - Pro*C, embedding SQL in, 414–416
 - SQL compared to, 410–411
 - with SQL for Turing-completeness, 51
- C# language
 - challenges using SQL with, 413
 - with MySQL, 406
 - SQL compared to, 411
- C++ language
 - challenges using SQL with, 413
 - SQL compared to, 411
- calculated columns, `DECLARE CURSOR` statement with, 318
- `CALL` statement (SQL/PSM)
 - for stored functions, 472–473
 - for stored procedures, 469

- candidate keys, 191
- capitalization, conversions for strings, 211
- cardinality
 - actual, 220
 - of arrays, 220
 - defined, 34
 - E-R diagram indications for, 35–36, 145, 146
 - mandatory, 35, 36, 37, 145–146
 - maximum, 34–35, 145, 220, 430–431
 - minimum, 35–37, 145–147, 431
 - optional, 35–36, 37, 145–146
 - zero versus one, 146–147
- `CARDINALITY` function
 - with arrays, 220
 - overview, 214
- Cartesian product
 - limitations of, 305
 - overview, 303–305
 - virtual tables from, 228–229
- `CASCADE` keyword (`REVOKE`), 380, 382
- cascading deletes
 - defined, 284
 - deletion anomalies prevented by, 174, 284
 - `ON DELETE CASCADE` clause, 174
- case, conversions for strings, 211
- `CASE` expression, 221–222
- `CASE . . . END CASE` statements (SQL/PSM), 464–465
- `CAST` expression
 - casting between SQL types, 224
 - for data type incompatibilities, 224–225, 413, 416
 - overview, 223–224
 - `XMLCAST` function versus, 512
- casting between XQuery and SQL types, 549
- `CATALOG_NAME` field (diagnostics area), 391, 395

- catalogs. *See also* schemas
 - clusters, 629
 - in database hierarchy, 47, 67–68
 - defined, 629
 - overview, 64
- CEIL or CEILING function, 216
- CHAR or CHARACTER type, 99
- CHARACTER LARGE OBJECT (CLOB)
 - type, 99–100
- character sets
 - collation rules for, 318
 - converting strings between, 211
 - data types for, 100
 - for different languages, 100
 - mapping to XML, 499
 - in module declaration, 94, 420
 - octets required by, 214
 - privileges applying to, 375
 - UPPER and LOWER functions with, 211
- character string types
 - BETWEEN predicate with, 232
 - CHAR or CHARACTER, 99
 - CHARACTER VARYING, 99
 - CLOB (CHARACTER LARGE OBJECT), 99–100
 - NATIONAL CHARACTER, 100
 - NATIONAL CHARACTER LARGE OBJECT, 100
 - NATIONAL CHARACTER VARYING, 100
 - overview, 99
- CHARACTER VARYING or VARCHAR
 - type, 99
- CHARACTER_LENGTH function, 213
- CHECK constraints
 - CREATE DOMAIN statement for, 77–78, 194
 - for domain integrity, 172
 - for ensuring data validity, 194
 - example, 70–71
 - expressions for, 70
 - overview, 112–113
- checkpoints
 - flushing the page buffer, 354, 571, 574–575
 - overview, 574–575
 - performance issues, 369
 - for ROLLBACK operation, 355
- Chen, Peter (*ACM Transactions on Database Systems*), 29
- class codes (SQLSTATE)
 - for compound statements, 463
 - for implementors, 385
 - indicating error conditions, 385, 386, 395
 - overview, 384–385
 - table summarizing values, 384, 463
- classes
 - entity, 30, 140
 - relationship, 142
- classic procedural languages
 - challenges using SQL with, 412–413
 - overview, 410
 - SQL compared to, 410–411
- CLASS_ORIGIN field (diagnostics area), 391, 392, 393
- Clear Creek Medical Center (CMCC)
 - E-R model, 42–46
- CLI, DB2 support for, 402
- client
 - defined, 629
 - locking and performance for, 592
- client for project. *See also* stakeholders
 - discovering the problem, 423–424
 - getting approval from, 122, 137–138, 139
 - keeping informed, 130
 - proposal for, 425–426
 - requirements phase dependant on, 123
 - Statement of Requirements as contract with, 125, 425
 - three-option proposal for, 139
 - upper management, 137–138
 - your immediate supervisor, 136

- client/server system, 629
- CLOB (CHARACTER LARGE OBJECT)
 - type, 99–100
- CLOSE statement, 324
- closing cursors, 324
- clustered indexes
 - defined, 181, 557
 - join conditions with, 314
 - for multipoint queries, 314
 - one allowed per table, 182, 557
 - ORDER BY clause eliminated by, 314
 - overview, 181–182
 - performance issues, 557–559
 - performance of, 558–559
 - “tired,” rebuilding, 314
- clusters, defined, 629
- CMCC (Clear Creek Medical Center)
 - E-R model, 42–46
- C++.NET language with MySQL, 406
- COALESCE expression, 223
- COBOL, 410
- CODASYL DBTG database model, 630
- Codd, E. F. (database expert)
 - as IBM employee, 49, 147
 - normal forms devised by, 152
 - relational model proposed by, 15, 147
 - rules of, 20, 22–23
- COLLATE BY clause, 317, 318
- collation
 - with cursors, 317, 318
 - privileges applying to, 375
 - rules for character sets, 318
 - sequence, defined, 630
- collection types
 - ARRAY, 105, 504–505
 - defined, 630
 - introduction of, 105
 - MULTISET, 105, 505
 - 1NF violated by, 105
- COLUMN_NAME field (diagnostics area), 391, 395
- column-name joins, 308–309
- columns. *See also* attributes (database)
 - adding, 78, 198
 - averaging values in, 208
 - calculated, 318
 - CHAR type for, 99
 - choosing for indexes, 177–178, 557, 558
 - constraints, 70–71, 111–113, 193
 - in database hierarchy, 47, 67–68
 - defined, 630
 - deleting, 198
 - dropping, 78
 - as fields, 148
 - finding maximum value in, 208
 - finding minimum value in, 209
 - fully qualified names for, 73
 - identifying values in, 204
 - INSERT statement for selected, 451
 - minimizing in retrievals, 594
 - other terms for, 148
 - qualifying names in equi-joins, 306
 - specifying, 68–69
 - transferring between tables, 453–454
 - in union-compatible tables, 297
 - VARCHAR type for, 99
 - XML types for, 103–104
- COMMAND_FUNCTION field (diagnostics area), 390, 395
- COMMAND_FUNCTION_CODE field (diagnostics area), 390
- comments
 - with APPENDCHILDXML function (Oracle), 519
 - creating in XML, 510
 - with DELETEXML function (Oracle), 521–522
 - with INSERTCHILDXML function (Oracle), 519–520
 - with INSERTXMLBEFORE function (Oracle), 520–521
 - with UPDATEXML function (Oracle), 522–523

- COMMIT operation
 - described, 88
 - ROLLBACK operation versus, 352
- committing transactions, 352, 566
- comparison operators
 - correlated subqueries with, 279–281
 - list of, 230–231
 - for quantified subqueries, 272, 275–277
 - tuning subqueries using, 285–289
- comparison predicates (WHERE)
 - ALL, 236–239, 272, 275, 276–277
 - ANY, 236–239, 272, 275
 - comparison operators for, 230–231
 - for correlated subqueries, 278–281
 - described, 230
 - DISTINCT, 240–241
 - EXISTS, 239–240, 278
 - IN, 232–233, 279
 - LIKE, 234–235
 - list of, 230
 - MATCH, 241–245
 - NOT EXISTS, 278–279
 - NOT IN, 232
 - NOT LIKE, 234
 - NOT NULL, 236
 - NULL, 235–236
 - OVERLAPS, 241
 - quantified comparison operators, 272, 275
 - SIMILAR, 235
 - SOME, 236–239, 272, 275
 - for subqueries returning a single value, 272–274
 - UNIQUE, 240, 243
- compiling
 - embedded SQL, 91
 - precompiling queries, 563, 595
- composite identifiers
 - defined, 31, 141
 - unique, 142
- composite indexes
 - defined, 559
 - overview, 182
 - performance issues, 559–560
 - query optimizers with, 182–183
- composite keys, 191, 630
- compound statements (SQL/PSM)
 - assignment, 463
 - conditions, 462–463
 - cursors within, 462
 - ensuring atomicity, 461
 - overview, 460–461
 - variables in, 462
- concatenating XML arguments, 509
- concatenation operator, 218
- conceptual view, 630. *See also* schemas
- concurrent access. *See also*
 - serializability
 - conflicts from, 346
 - corruption from, 346–347
 - deadlocks, 361–362, 579
 - defined, 630
 - READ UNCOMMITTED isolation allowing, 350–351
 - scheduling threads, 575–579
- condition joins, 307–308
- conditional value expressions
 - CASE, 221–222
 - COALESCE, 223
 - defined, 637
 - NULLIF, 223
 - overview, 220
- CONDITION_IDENTIFIER field
 - (diagnostics area), 392
- CONDITION_NUMBER field (diagnostics area), 391, 392, 395
- conditions of compound statements, 462–463
- CONNECT statement, 65
- connecting to remote database
 - JDBC for, 591
 - native drivers for, 475–476, 591
 - ODBC for, 477–487, 591–592
 - overview, 64–65
- connecting user interface to database, 445–447

- connection handle (ODBC), 482, 483
- CONNECTION_NAME field (diagnostics area), 391, 394
- consistency, as ACID characteristic, 348
- CONSTRAINT_CATALOG field (diagnostics area), 391, 393, 394
- CONSTRAINT_NAME field (diagnostics area), 391, 393, 394
- constraints. *See also specific kinds*
 - adding to existing table, 394
 - assertions, 71–72, 115, 629
 - column, 70–71, 111–113, 193
 - CREATE DOMAIN statement for, 77–78
 - data entry errors prevented by, 344
 - DEFERRABLE, 355–359
 - DEFERRED, 356, 358, 359, 630
 - defined, 155, 630
 - diagnostics area information, 393–394
 - expressions for, 70
 - foreign key, 114–115
 - implicit names for, 359
 - naming, 70
 - table, 71, 113–114, 194
 - uses for, 69, 193
- CONSTRAINT_SCHEMA field (diagnostics area), 391, 393, 394
- constructors for UDTs, 107–108
- containment hierarchy (DDL), 67–68
- CONTENT predicate for XML, 513
- CONTINUE action, 387, 389
- CONVERT function, 211
- converting. *See also mapping SQL to XML*
 - case of strings, 211
 - character set of string, 211
 - data types with CAST expression, 223–225, 413, 416
 - data types with XMLCAST function, 512
- converting E-R model to relational model
 - eliminating many-to-many relationships, 433–435
 - handling binary relationships, 164–168
 - normalization, 162–163, 435–437
 - overview, 161–162
 - sample conversion, 168–170
- correctness. *See* integrity
- correlated subqueries
 - with comparison operators, 279–281
 - defined, 277
 - with EXISTS and NOT EXISTS, 278–279
 - with HAVING clause, 279–281
 - with IN, 279
 - overview, 277–282
 - relational queries compared to, 290–295
 - tuning, 290–295
- correlation names. *See* aliases
- CORRESPONDING keyword
 - with INTERSECT operations, 301–302
 - with UNION operations, 300
- corruption. *See also* anomalies; integrity
 - avoiding, 174–175
 - from concurrent access, 346–347
 - sources of, 174
- COUNT function
 - EXISTS predicate equivalent to, 240
 - overview, 207–208
- CPU or processor
 - multiprocessor environments, 585
 - performance issues, 583, 597
 - performance monitors, 608–611
 - registers, 176, 567, 568
 - upgrading, 583
- CREATE ASSERTION statement, 71, 629
- CREATE DISTINCT TYPE statement, 107
- CREATE DOMAIN statements
 - for CHECK constraints, 77–78
 - for ensuring data validity, 194
- CREATE MODULE statement, 473–474
- CREATE ROLE statement, 373
- CREATE SCHEMA statements, 77
- CREATE TABLE statements
 - CHECK constraint in, 70–71
 - concurrent access with, avoiding, 365
 - distinct types with, 107

for multi-table views example, 75
 NOT NULL constraint in, 69–70, 112
 Oregon Lunar Society database,
 437–440
 overview, 68–69
 PRIMARY KEY constraint in, 71
 syntax, 68–69
 for tables holding XML data, 517
 using, 192–193
 CREATE TRIGGER statement, 470–472
 CREATE VIEW statements
 fully qualified column names for, 73
 for multi-table views, 76–77
 for single-table views, 73
 cropping strings, 211
 cross joins. *See* Cartesian product
 current user identifier, 372
 CURRENT_DATE function, 216–217
 CURRENT_TIME function, 216–217
 CURRENT_TIMESTAMP function, 216–217
 CURRENT_USER variable, 206, 372
 CURSOR_NAME field (diagnostics area),
 391, 395
 cursors
 absolute versus relative fetches, 323
 closing, 324
 within compound statements, 462
 datetime values fixed by opening,
 321, 322
 declaring, 316–320
 defined, 315, 630
 deleting a row, 323
 fetching data from single row, 322–324
 holdability, 317
 for modules in Oracle, 421
 naming, 316–317
 opening, 320–322
 ordering the query result set, 317–318
 performance issues, 594
 query expression for, 317
 returnability, 317
 scrollability, 317, 320

scrolling, 320
 sensitivity, 317, 319–320
 specifying updatability of rows, 319
 updating a row, 323
 uses for, 315–316
 variable values fixed by opening, 322

D

DAL, DB2 support for, 402
 DAO (Data Access Objects), 400
 Data Control Language. *See* DCL
 Data Definition Language. *See* DDL
 data entry
 automatic, 81, 452
 copying from foreign data file, 452
 errors from, 330–331, 343–345
 filling tables with sample data, 449–454
 forms for, 81, 450
 INSERT statements for, 80–81, 450–451
 reducing errors from, 331, 343–345
 transferring all rows between tables,
 452–453
 transferring selected columns and rows
 between tables, 453–454
 data integrity. *See* integrity
 Data Manipulation Language. *See* DML
 data redundancy, 17, 630
 data security. *See* integrity; security
 issues
 data sources, 482, 630
 data structure diagrams
 Honest Abe's database, 170
 many-to-many relationship, 167
 one-to-many relationship, 166
 one-to-one relationship, 164–165
 data sublanguages
 defined, 631
 limitations of, 315
 overview, 411–412
 SQL as, 89, 315, 411, 631

- data types. *See also* UDTs (user-defined types); *specific types*
 - booleans, 101
 - character strings, 99–100
 - collection types, 105
 - converting with `CAST` expression, 223–225
 - data entry errors prevented by, 344
 - datetimes, 101–103
 - defined, 631
 - exact numerics, 95–97
 - host language versus SQL, 90–91
 - implementation-specific, 94–95
 - incompatibilities, `CAST` for, 224–225, 413, 416
 - intervals, 103
 - literal value examples for, 204–205
 - mapping to XML, 500–505
 - overview, 94–95
 - `REF` types, 106
 - row types, 104, 503–504
 - SQL implementation support for, 104, 110
 - strong typing, 344, 345
 - table summarizing, 109–110
 - XML, 103–104, 497–499, 513–514
 - XQuery compared to SQL, 547–549
- data validity
 - determining for XML values, 513–514
 - domains for ensuring, 194–195
 - valid, defined, 194
- database administrator (DBA), 373–374, 631
- database application design
 - building an E-R model, 429–432
 - connecting user interface to database, 445–447
 - in design phase of SDLC, 129
 - determining deliverables, 426–429, 444
 - determining project scope, 428–429
 - discovering the client’s problem, 423–424
 - interviewing stakeholders, 424–425
 - planning for organization growth, 427–428
 - proposal for, 425–426
 - Statement of Requirements, 425
 - top-down approach, 443–447
 - user interface design, 441, 444–445
- database applications. *See also*
 - embedded SQL; module language building, 130, 447–455
 - complexity of, 327
 - defined, 52, 120
 - designing, 129
 - documenting, 131
 - interaction with database, 594
 - minimizing traffic between server, 563
 - overview, 120–121
 - suboptimal decisions by development tools, 592
 - temporal locality, 581
 - testing, 455–457
- database design
 - as basis for database tuning, 553
 - converting E-R model to relational model, 161–170
 - design phase of SDLC, 128
 - example conversion, 168–170
 - handling binary relationships, 164–168
 - normalizing a relational model, 162–163
 - physical design considerations, 555
 - security issues from flaws, 330, 345
- Database Development For Dummies* (Taylor), 330
- database dumps, 369, 573–574
- database engine, 120, 631
- Database Engine Tuning Advisor, 185–187, 602–607
- database locks, 361. *See also* locks
- database management system. *See* DBMS

- database object owners, 374
- database owner, privileges granted by, 473
- database publishing, 631
- database server
 - defined, 631
 - minimizing traffic between applications, 563
- database tuning. *See also* performance; query tuning; tuning
 - determining query complexity, 554
 - index considerations, 556–560
 - index tuning, 560–561
 - minimizing traffic between application and server, 563
 - optimal design as basis for, 553
 - physical design considerations, 555
 - precompiling queries, 563, 595
 - separating user interactions from transactions, 562
 - transaction tuning, 562
 - update operation considerations, 554
 - workload analysis, 554
- databases
 - defined, 52, 120, 631
 - DML for retrieving data, 79–80
 - documenting, 130
 - dumps, 369, 573–574
 - enterprise, 631
 - page buffer for, 579–580, 581, 617
 - personal, 631
 - terminology for elements, 147–148
 - workgroup, 631
- DATE type, 101
- datetime types
 - BETWEEN predicate with, 232
 - DATE, 101
 - extracting fields from, 213
 - overview, 101
 - TIME WITH TIME ZONE, 102
 - TIME WITHOUT TIME ZONE, 101–102
 - TIMESTAMP WITH TIME ZONE, 102–103
 - TIMESTAMP WITHOUT TIME ZONE, 102
- datetime value expressions, 218, 637
- datetime value functions
 - fixed by OPEN statement, 321, 322
 - overview, 216–217
- DBA (database administrator), 373–374, 631
- DBMS (database management system)
 - advantages, 12
 - defined, 52, 631
 - development of, 13, 14–15
 - disadvantages, 12
 - flat file systems versus, 12–13
- DBMS-based drivers (ODBC), 480–481
- DB2 (IBM)
 - assertions not supported by, 72
 - defined, 631
 - as development environment, 401–402
 - interfaces supported, 401–402
 - Java with, 407–408
 - overview, 56
 - platforms supported, 56
- DCL (Data Control Language). *See also specific statements*
 - defined, 86, 630
 - granting privileges, 86–87, 371
 - preserving database integrity, 87–88
 - revoking privileges, 87, 371
 - SQL*Module (Oracle) support for, 421
- DDL (Data Definition Language). *See also specific statements*
 - ADOX library, 405
 - containment hierarchy, 67–68
 - creating domains, 77–78
 - creating schemas, 77
 - creating tables, 68
 - creating views, 72–77

- DDL (Data Definition Language)
 - (*continued*)
 - defined, 67, 630
 - modifying tables and objects, 78
 - not supported by SQL*Module (Oracle), 421
 - removing tables and objects, 78
 - running concurrently with transactions,
 - avoiding, 365
 - specifying columns, 68–69
 - specifying constraints, 69–72
- deadlocks, 361–362, 579
- debugging
 - database application, 456
 - embedded SQL challenges for, 91
 - in maintenance phase of SDLC, 133
- DECIMAL type, 96
- DECLARE CURSOR statement
 - COLLATE BY clause with, 317, 318
 - FOR READ ONLY clause, 319
 - FOR UPDATE clause, 319
 - holdability keywords, 320
 - ORDER BY clause with, 317–318
 - query expression for, 317
 - returnability keywords, 320
 - scrollability keywords, 317, 320
 - sensitivity keywords, 319–320
 - specifying calculated columns, 318
 - syntax, 316
- declaring
 - condition handlers, 387
 - cursors, 316–320
 - exception handlers, 396
 - host variables in Pro*C, 416
 - modules, 93–94, 419–420
 - WHENEVER directive, 388–389
- decomposing many-to-many relationships, 167–168, 169, 433–435
- defaults
 - FETCH statement option, 322
 - ORDER BY clause sort order, 251, 318
 - schema, 77
 - transaction, 348
- DEFERRABLE constraints
 - implementing, 355–359
 - payroll examples, 356–359
 - setting to DEFERRED, 356, 358
 - setting to IMMEDIATE, 356, 359
 - used only within transactions, 359
- DEFERRED constraints
 - defined, 630
 - setting DEFERRABLE constraints to, 356, 358
 - setting to IMMEDIATE, 356, 359
- definition phase of SDLC, 122
- defragmenting hard disks, 615–616
- degree-three relationships
 - E-R diagram, 34, 143
 - overview, 33, 142
- degree-two relationships. *See* binary relationships
- delegating privileges, 379
- DELETE statements
 - with cursors, 323
 - deleting a row, 323
 - granting privileges for, 375, 473
 - for old transactions, 85
 - subqueries in, 284
 - WHERE clause, 284
- DELETXML function (Oracle), 521–522, 523
- deleting or removing
 - columns, 78, 198, 440
 - data from table, 84–85
 - index performance degraded by, 589
 - privileges, granting, 375
 - privileges, revoking, 87
 - record using updategram, 526
 - roles, 373
 - row that cursor points to, 323
 - tables, 78, 199
 - views, 78
 - XML nodes (Oracle), 521–522

- deletion anomalies
 - cascading deletes to prevent, 174, 284
 - defined, 151, 631
- deliverables
 - defined, 122
 - determining for project, 426–429, 444
 - planning for organization growth, 427–428
- denial-of-service attacks, 339
- dense indexes, 181
- dependencies
 - functional, 60–61, 149–150, 632
 - transitive, 154, 636
- DESC keyword, 251, 318
- descriptor, 631
- design phase of SDLC
 - database application design, 129
 - database design, 128
 - documenting, 129
 - keeping client informed, 130
 - overview, 127–128
- detail diagnostics area
 - described, 389
 - fields, 391–393
 - retrieving information, 395
- determinants, 149–150
- diagnostics areas
 - components, 389
 - constraint violation information, 393–394
 - defined, 631
 - detail area, 389, 391–393, 395
 - GET DIAGNOSTICS statement for, 392, 393–394, 395
 - header area, 389, 390–391, 395
 - LIFO order for information, 389
 - retrieving information, 395
 - size settings for detail area, 349, 390
- dirty reads, 350
- disk controller cache, 581–582, 597
- disk controller delay, 570
- disk mirroring (RAID 1), 334, 335
- disk subsystem management, 615–616
- DISTINCT keyword
 - not needed for primary key selects, 252
 - not needed with EXCEPT, 302
 - not needed with INTERSECT, 301
 - not needed with UNION, 299
 - query tuning for selects using, 252–254
- DISTINCT predicate (WHERE), 240–241
- distinct types
 - creating tables using, 107
 - creating types, 107
 - mapping to XML, 502–503
 - overview, 106–107
- distributed data processing, 632
- DKNF (Domain/Key Normal Form)
 - all anomalies eliminated by, 155
 - defined, 152, 155
 - determining if entity is in, 163
 - examples, 155–156, 163
 - normalizing relation into, 163
 - performance issues, 156–158, 162, 163
- DLL (Dynamic-Link Library), 479
- DML (Data Manipulation Language).
 - See also specific statements*
 - adding data to table, 80–81
 - defined, 78, 630
 - deleting data from table, 84–85
 - overview, 78–79
 - retrieving data, 79–80
 - SQL*Module (Oracle) support for, 421
 - updating data in table, 81–84
 - updating views, 85–86
- document element (XML), 494
- DOCUMENT predicate, 512–513
- Document Type Definition (DTD)
 - creating XML entity references, 495
 - XML Schema versus, 496
- documentation. *See also* comments
 - of database, 130
 - of database application, 131
 - in definition phase of SDLC, 122
 - in design phase of SDLC, 129

documentation (*continued*)
 in evaluation phase of SDLC, 125, 127
 finalizing in SDLC, 130–131, 132
 in implementation phase of SDLC, 130

DOM (Document Object Model), 518

domain integrity, 172, 632

Domain/Key Normal Form. *See* DKNF

domains
 creating, 77–78
 defined, 155, 172, 632
 for ensuring data validity, 194–195
 mapping to XML, 501–502
 privileges applying to, 375

DOUBLE PRECISION type, 98

DRDA, DB2 support for, 402

driver manager
 defined, 479, 632
 ODBC, 478, 479

drivers
 defined, 632
 native, 475–476, 481
 ODBC, 479–481

DROP COLUMN clause (ALTER TABLE), 78, 198, 440

DROP INDEX statements, avoiding concurrent access with, 365

DROP ROLE statement, 373

DROP TABLE statements
 dangers of, 199, 441
 examples, 78, 199, 440

DROP VIEW statements, 78

dropping. *See* deleting or removing

DTD (Document Type Definition)
 creating XML entity references, 495
 XML Schema versus, 496

dumps, 469, 573–574

duplicate rows, preserving with UNION operations, 299

durability, as ACID characteristic, 348

dynamic SQL, 635

DYNAMIC_FUNCTION field (diagnostics area), 390, 395

DYNAMIC_FUNCTION_CODE field (diagnostics area), 390

Dynamic-Link Library (DLL), 479

E

EDA/SQL, DB2 support for, 402

element-centric mapping (SQL Server)
 mixed with attribute-centric mapping, 532
 overview, 531

elements (XML)
 attributes of, 494
 document, 494
 empty, 494
 nested, 494
 overview, 493–494
 translating relational values to, 508–509
 updating record from (SQL Server), 526

e-mail
 phishing scams, 339
 viruses, 337–338
 zombie spambots, 339–340

embedded SQL. *See also* cursors;
 SQL/PSM (Persistent Stored Modules)
 challenges for, 90–91, 412–413
 common flow of execution, 316
 contrasting operating modes with, 412
 cursors for, 316
 data type incompatibilities, 224–225, 413
 debugging challenges, 91
 defined, 89, 91, 636
 example, 91–93
 impedance mismatch, 413
 in Java application, 417
 module language versus, 93, 419
 in Oracle Pro*C application, 414–416
 with other .NET applications, 418
 in Perl application, 417
 in PHP application, 417

- pre-compiler for, 91
- UDTs for matching host language types, 106
- in Visual Basic.NET application, 418
- empty elements (XML), 494
- END keyword, 388, 460
- enterprise databases, 631
- entities. *See also* E-R (Entity-Relationship) model; relations
 - classes, 30, 140
 - constructing E-R diagram for, 429–432
 - defined, 30, 140
 - determining if in DKNF, 163
 - in E-R model context, 30, 140–141
 - existence-dependent, 38
 - ID-dependent, 39
 - identifying, 429
 - instances, 30, 140
 - relations comparable to, 148, 432
 - strong versus weak, 37–38
 - supertype and subtype, 39–40
 - translating into relations, 161–162, 432–433
- entity integrity
 - defined, 171, 632
 - NOT NULL constraint for, 171
 - PRIMARY KEY constraint for, 171–172
 - UNIQUE constraint for, 171
- entity references (XML), 495
- Entity-Relationship model. *See* E-R model
- environment handle (ODBC), 482, 483
- environment tuning
 - adding hardware, 582–585
 - importance of, 565–566
 - maximizing hardware, 580–582
 - multiprocessor environments, 585
 - operating system, 575–580
 - recovery system, 567–575
 - surviving failures with minimum data loss, 566–567
- ENVIRONMENT_NAME field (diagnostics area), 394
- equality conditions, 178–179
- equals symbol (=) in comparison operators, 230–231
- equi-join queries
 - on multiple tables, 306–307
 - natural joins, 307, 308
 - overview, 180, 305–307
 - qualifying column names in, 306
 - on two tables, 305–306
- equipment failure. *See also* recovery system
 - backups as protection against, 335–337
 - data integrity dangers from, 87
 - fault tolerance, 331, 334, 335
 - hard failures, 574
 - inevitability of, 565, 568
 - likely for hard disks, 331, 568
 - minimizing, 329
 - RAID as protection against, 331–335
 - soft failures, 574
 - sources of, 328
 - surviving with minimum data loss, 566–567
 - before transaction written to disk, 567
- E-R diagrams
 - building for database application, 429–432
 - CMCC system, 45
 - degree-three relationship, 34, 142–143
 - degree-two relationship, 142
 - determining implications of, 45–46
 - determining relationships, 430
 - Gentoo Joyce system, 42
 - Honest Abe's database, 160, 168, 175
 - ID-dependent entities, 39
 - many-to-many relationship, 33, 144, 166
 - maximum cardinality, 34–35, 430–431
 - minimum cardinality, 35–37, 38, 145, 146, 431
 - one-to-many relationship, 33, 144
 - one-to-one relationship, 32, 143–144, 164
 - for order entry system, 74

- E-R diagrams (*continued*)
 - supertype/subtype relationship, 39–40
 - weak entities in, 38
- E-R (Entity-Relationship) model. *See also* E-R diagrams; *specific elements*
 - anomalies, defined, 47
 - attributes (overview), 30–31, 141
 - basic elements, 30, 140
 - building for database application, 429–432
 - business rules, 40–41, 147, 431–432
 - complex example, 42–46
 - complex relationship problems, 46–47
 - converting to relational model, 161–170, 432–437
 - in database design, 128
 - determining entities for, 429
 - drawing diagrams, 34–37
 - eliminating many-to-many relationships, 433–435
 - entities (overview), 30, 140–141
 - ID-dependent entities, 39
 - identifiers (overview), 31, 141–142
 - importance of, 29
 - maximum cardinality, 34–35, 145
 - minimum cardinality, 35–37, 145–147
 - normalization, 435–437
 - normalization for, 47
 - relationships (overview), 31–34, 142–144
 - scalability of, 29–30
 - simple example, 41–42
 - strong versus weak entities, 37–38
 - supertype and subtype entities, 39–40
 - terminology for elements, 148, 161–162, 432–433
- error conditions. *See also* exceptions
 - diagnostics area for, 389–395
 - handler actions for, 387
 - handler declarations for, 386–387
 - handler effects for, 387–388
 - identifying, 383–384
 - inevitability of, 383
 - requiring handling, 386
 - SQLSTATE class codes indicating, 385, 386, 395
 - WHENEVER directive for, 383, 388–389
- escape characters, LIKE predicate with, 234–235
- evaluation phase of SDLC
 - determining project scope, 125, 126–127
 - documenting, 125, 127
 - overview, 125
 - reassessing feasibility, 125, 127
- events for triggers, 471
- exact numeric types
 - BIGINT, 96
 - BLOB (BINARY LARGE OBJECT), 97
 - DECIMAL, 96
 - INTEGER, 95
 - limitations of, 95
 - NUMERIC, 96, 97
 - precision for, 95–97
 - SMALLINT, 96
- EXCEPT operations, 297, 302
- exceptions. *See also* error conditions
 - handling, 395–396
 - SQLSTATE class codes indicating, 385, 386, 395
 - WHENEVER directive for, 383, 388–389
- exclusive locks, 360. *See also* locks
- EXEC SQL directive, 93, 416
- EXECUTE privilege, 473
- executing statements. *See also* embedded SQL; module language
 - interactive SQL for, 89, 90, 636
 - overview, 89
 - privileges for, 375, 378–379, 473
- execution plans or access plans. *See also* query tuning
 - analyzing, 611–614
 - checking the access path, 612–613
 - choosing the best join type, 614
 - DBMS development of, 183

defined, 183–184
 filtering selectively, 613–614
 Northwinds example, 184–185
 robust, 183–184
 role of statistics in, 612
 execution profiles, 615
 existence-dependent entities, 38
 existential quantifiers
 ANY predicate (WHERE), 236–239, 272, 275
 SOME predicate (WHERE), 236–239, 272, 275
 EXISTS predicate (WHERE)
 for correlated subqueries, 278
 equivalent to COUNT comparison, 240
 overview, 239–240
 EXIT action, 387
 EXP function, 215
 explicit mapping (SQL Server), 526–531
 exponent for floating-point numbers, 98
 expressions
 array value, 220
 Boolean value, 219–220
 CAST, 223–225, 413, 416
 conditional value, 220–223, 637
 for constraints, 70
 datetime value, 218, 637
 defined value, 637
 described, 217
 FLWOR (XQuery), 542–547
 interval value, 219
 numeric value, 217, 637
 query optimizer limitations for, 561
 row value, 225, 635
 string value, 218, 511, 637
 XMLQUERY function for (XQuery), 511–512, 518
 XPath, 519–521
 Extensible Markup Language. *See* XML
 external routines, 65
 externally invoked routines, 65

EXTRACT function, 213
 extremal queries, 179

F

failure of equipment. *See* equipment failure
 fault tolerance, 331, 334–335
 feasibility
 analysis of, 122
 reassessing, 125, 127
 FETCH statements, 322–323
 fields. *See also* attributes; columns
 defined, 203–204
 other terms for, 148
 Fifth Normal Form (5NF), 152, 155
 file server, 632
 file-based drivers (ODBC), 480
 files, other terms for, 148
 filter ratio of tables, 184
 final documentation and testing phase
 of SDLC
 delivering results, 132
 documentation, 130–131, 132
 testing with sample data, 131–132
 firewalls, 340, 632
 First Normal Form (1NF), 104, 151, 152–153
 5 RAID level, 334, 335, 584–585
 5NF (Fifth Normal Form), 152, 155
 flat files
 advantages, 10–11
 DBMS versus, 12–13
 defined, 632
 disadvantages, 11
 overview, 10
 terminology for elements, 148
 FLOAT type, 98
 floating-point numbers
 defined, 98
 DOUBLE PRECISION type for, 98

floating-point numbers (*continued*)

FLOAT type for, 98

REAL type for, 97–98

FLOOR function, 216

flow of control statements (SQL/PSM)

CASE...END CASE, 464–465

FOR...DO...END FOR, 468

IF...THEN...ELSE...END IF, 464

ITERATE, 468

LEAVE, 466–467

LOOP...END LOOP, 466

need for, 463–464

REPEAT...UNTIL...END REPEAT,
467

WHILE...DO...END WHILE, 467

flushing the page buffer, 354, 571,
574–575. *See also* checkpoints

FLWOR expressions (XQuery)

conventions, 542

for clause, 543–544

let clause, 544–545

order by clause, 545–546

return clause, 543, 546–547

SELECT statements compared to, 547

SQL table corresponding to expression,
542–543

syntax, 542

where clause, 545

fonts in this book, 2

for clause (FLWOR), 543–544

FOR READ ONLY clause (DECLARE
CURSOR), 319

FOR UPDATE clause (DECLARE
CURSOR), 319

FOR...DO...END FOR statements
(SQL/PSM), 468

FOREIGN KEY constraint, 114–115

foreign keys

CLOB type not allowed for, 100

constraints, 114–115

defined, 195, 632

establishing relationships between
tables, 195–198

for referential integrity, 198

forest, 509, 632

forms

for data entry, 81, 450

screen, developing, 454

Fortran, 410

4NF (Fourth Normal Form), 152, 155

fragmented hard disks, 615–616

FROM clause (SELECT)

for Cartesian product, 228–229

for equi-joins, 180

overview, 228–229

specifying single table, 228

front end, 120, 632

full backup, 336. *See also* backing up

FULL keyword (MATCH), 244–245

full outer joins, 313

full table scans

indexes versus, 183, 590–591

overview, 177, 590

functional dependencies

defined, 60, 149, 632

determinants, 149–150

overview, 60–61, 149–150

functions. *See also specific functions*

mutator, 108, 634

observer, 108

for operating on XML, 508–512

Oracle, for updating XML data in tables,
518–523

as routines, 65

set or aggregate, 207–209, 510, 629, 635

value, 207, 209–217, 637

G

GET DIAGNOSTICS statement, 392,
393–394, 395

GMT (Greenwich Mean Time), 102

GO TO action, 389

GRANT OPTION FOR clause (REVOKE),
380

GRANT statements

- GRANT ALL PRIVILEGES, 379
- GRANT DELETE, 375
- GRANT EXECUTE, 375, 378–379, 473
- GRANT INSERT, 375, 376
- GRANT REFERENCES, 375, 377–378, 473
- GRANT SELECT, 375
- GRANT TRIGGER, 375, 378
- GRANT UNDER, 375, 378
- GRANT UPDATE, 375, 376
- GRANT USAGE, 375, 378, 473

identifying authorized users for, 371,
372–373

objects applied to, 375

overview, 86–87

PUBLIC keyword, 87

for roles, 381

syntax, 374–375

WITH ADMIN OPTION clause, 381

WITH GRANT OPTION clause,
379, 380–381

GRANTED BY clause (REVOKE),
380, 381–382

grantees, defined, 374

granularity of locks, 361, 365

greater than symbol (>)

- in comparison operators, 231
- XML entity reference for, 495

Greenwich Mean Time (GMT), 102

GROUP BY clause (SELECT)

- filtering first, 262–266
- for grouping queries, 180

ORDER BY clause with, 249

overview, 228, 247–249

grouping queries, 180

H

handler actions, 387

handler declarations, 386–387, 396

handler effects, 387–388

handles (ODBC), 482–483

hard disks

- adding more, 583–584
- construction of, 569–570
- controller cache, 581–582, 597
- controller delay, 570
- defragmenting, 615–616
- failure likely for, 331, 568
- hard failures, 574
- indexes' location on, 562
- managing, 615–616
- optimizing placement of code and data,
580–581
- partitioning insertions, 365
- performance considerations, 570–571,
583–585, 596–597
- performance monitors, 608–611
- RAID technology, 331–335, 584–585
- read/write time, 570
- rotational latency, 570, 571
- seek time, 570, 571
- settling time, 570, 571
- in storage hierarchy, 176, 568
- tuning write operations, 572–573
- write-back protocol, 582
- write-through protocol, 582

hard failures, 574

hardware. *See also* equipment failure;
specific hardware

- adding, 582–585
- managing resources, 615–618
- maximizing existing, 580–582
- performance issues, 580–585, 593,
596–597
- redundant, for fault tolerance, 335
- single precision as dependent on, 97–98

hash joins, 614

hash structures for indexes, 181, 559, 562

HAVING clause (SELECT)

- correlated subqueries in, 281–282
- GROUP BY clause with, 262–266

HAVING clause (SELECT) (*continued*)
 overview, 228, 249
 query tuning for, 262–266

header diagnostics area
 described, 389
 fields, 390–391
 retrieving information, 395

hierarchical database model
 defined, 632
 development of, 14
 diagrams, 16, 18
 IMS as, 15
 overview, 15–17
 relational model versus, 20, 23, 24–25
 uses for, 17

hierarchical storage, 176–177

hierarchies
 containment (DDL), 67–68
 memory, 176–177, 567–568
 in relational databases, 47, 67–68

holdability of cursors, 317

Honest Abe's Fleet Auto Repair
 database
 avoiding corruption, 174–175
 business overview, 160
 converting E-R model to relational
 model, 168–170
 CREATE TABLE statements for, 192–193
 data structure diagram, 170
 E-R diagrams, 160, 168, 175
 establishing relationships between
 tables, 195–198
 handling binary relationships, 164–168
 maintaining integrity, 170–174
 normalizing the relational model,
 162–163
 relationships, 161
 tables and attributes, 190–191

host language. *See* embedded SQL

host variables
 declaring in Pro*C, 416
 defined, 632

hot spots, 365–366, 598
hot tables, 187
HTML (HyperText Markup Language),
 491, 633

I

IBM. *See also* DB2 (IBM)
 Codd employed by, 49, 147
 IMS (first DBMS), 13, 14
 SQL developed by, 1, 50
 SQL/DS RDBMS product, 50

ID-dependent entities, 39

identifiers
 composite, 31, 141
 described, 31, 141
 mapping to XML, 500
 non-unique, 31, 141
 unique, 31, 141, 142

IF . . . THEN . . . ELSE . . . END IF
 statements, 464

impedance mismatch, 413

implementation, defined, 633

implementation phase of SDLC, 130

implementations of SQL. *See also specific implementations*
 ANSI/ISO compliance of, 2, 52
 assertions lacking in, 72, 115
 new data types in, 94–95
 overview, 52–57
 row types lacking in, 104

implicit mapping (SQL Server), 525–526

IMS (Information Management System)
 of IBM, 13, 14–15

IN predicate (WHERE)
 for correlated subqueries, 279
 overview, 232
 performance issues, 233
 with subqueries, 233

incremental backup, 336. *See also*
 backing up

- indexes
 - in ANSI/ISO SQL standard, 178
 - B+ trees for, 181, 558, 559, 589
 - choosing columns for, 177–178, 557, 558
 - choosing type for, 559
 - clustered, 181–182, 557–559
 - composite, 182–183, 559–560
 - data structures for, 180–181
 - Database Engine Tuning Advisor for, 185–187
 - defined, 177, 633
 - deletes' impact on, 589
 - disk location for, 562
 - effect on joins, 183
 - excessive, costs of, 178
 - full table scans versus, 183, 590–591
 - hash structures for, 181, 559, 562
 - for hot tables, 187
 - index-only queries, 590
 - ISAM, 562
 - load balancing for, 187
 - maintenance costs, 559
 - multi-column, 557
 - with OR logical connective, 266–267
 - overview, 177
 - performance impact of, 555
 - primary key for, 194
 - pros and cons of, 589–590
 - query types for, 178–180
 - rebuilding “tired,” 314, 560–561, 589
 - sparse versus dense, 181
 - table size issues, 183
 - tuning, 560–561
 - unnecessary, avoiding, 556
 - updates' impact on, 590
- Information Management System (IMS)
 - of IBM, 13, 14–15
- information schema, 633
- inheritance in supertype/subtype relationships, 40
- inner joins, 309
- INSERT statements
 - for data entry, 80–81, 450–451
 - granting privileges for, 376, 473
 - for incomplete data, 81
 - more efficient methods, 81
 - partitioning insertions, 365
 - for selected columns, 451
 - for single record, 80
 - subqueries in, 284–285
 - syntax, 450–451
 - VALUES clause, 80
 - WHERE clause, 284–285
- INSERTCHILDXML function (Oracle), 519–520, 523
- insertion anomalies, 151, 633
- INSERTXMLBEFORE function (Oracle), 520–521, 523
- instability of platforms, 329–330
- instances
 - entity, 30, 140
 - relationship, 142
- INTEGER type, 95
- integrity. *See also* corruption; referential integrity
 - domain, 172, 632
 - enforcing serializability with timestamps, 366–369
 - entity, 171–172, 632
 - hierarchical model issues, 17
 - importance of, 170–171
 - locking for, 360–366
 - performance tradeoff with, 156–158, 159
 - preventing transaction mix-ups, 348–359
 - recovery system tuning issues, 369
 - system failures damaging, 87
 - transactions for protecting, 87–88
 - user interactions damaging, 87
- interactive SQL, 89, 90, 636

InterBase (Borland), 57, 72
Internet, defined, 633
Internet threats
 denial-of-service attacks, 339
 exploits, 341
 phishing scams, 339
 protecting against, 340–341
 viruses, 337–338
 vulnerabilities, 341
 worms, 338–339
 zombie spambots, 339–340
INTERSECT operations
 ALL keyword with, 301
 CORRESPONDING keyword with,
 301–302
 described, 300–301
 DISTINCT keyword not needed with,
 301
 modeled on relational algebra, 297,
 300–301
INTERVAL type, 103
interval value expressions, 219
intervals
 defined, 103, 219
 extracting fields from, 213
 OVERLAPS predicate for, 241
 range queries using, 179
intranet, 633
IPX/SPX, 633
ISAM indexes, 562
isolation
 as ACID characteristic, 348
 levels compared, 352
 READ COMMITTED level, 351
 READ UNCOMMITTED level, 350–351
 REPEATABLE READ level, 351
 SERIALIZABLE level, 352
 SET TRANSACTION statement for, 349
 weakening levels for locking, 364–365
italics in this book, 2
ITERATE statements (SQL/PSM), 468

J

Java
 defined, 633
 embedding SQL in, 417
 Oracle support for, 402
 SQL compared to, 411
Java-based Embedded SQL (SQLJ),
 402, 407
JavaScript, 633
JDBC (Java DataBase Connectivity)
 DB2 support for, 402, 407
 defined, 591, 633
 MySQL support for, 402
 Oracle support for, 402, 407
Jet database engine, 399–400
JOIN operations
 Cartesian product, 228–229, 303–305
 choosing the best join type, 614
 clustered indexes with, 314
 column-name joins, 308–309
 condition joins, 307–308
 defined, 633
 equi-joins, 180, 305–307
 filter ratio and construction of, 184
 flexibility of, 297
 hash joins, 614
 index effect on, 183
 inner joins, 309
 left and right tables, defined, 310
 natural joins, 307
 nested-loop joins, 614
 ON versus WHERE clause with, 313
 outer joins, 310–313
 sort-merge joins, 614

K

keys. *See also* foreign keys; primary key candidate, 191
 composite, 191, 630

defined, 61, 150, 155
 as determinants, 150
 for indexes, 194
 locating rows with, 191
 overview, 61–62, 150
 required for relations, 62

L

L1 cache, 176, 568
 L2 cache, 176, 568
 LANGUAGE clause (MODULE), 94, 420
 LANs (local area networks). *See* networks
 leaf nodes (B+ tree), 589
 leaf structured types, 108
 LEAVE statements (SQL/PSM), 466–467
 left outer joins, 310–312
 less than symbol (<)
 in comparison operators, 230–231
 XML entity reference for, 495
 let clause (FLWOR), 544–545
 Level 1 cache, 176, 568
 Level 2 cache, 176, 568
 LIKE predicate (WHERE), 234–235
 literal values, 204–205
 livelocks, 367–369
 LN function, 215
 load balancing for hot table indexes, 187
 local area networks (LANs). *See* networks
 locks
 client performance degraded by, 592
 database, 361
 deadlocks, 361–362
 eliminating unneeded, 363
 exclusive, 360
 granularity, 361, 365
 locking subsystem management, 617–618
 minimizing time held, 562

 page, 361
 row, 361
 shared, 360
 short transactions for, 363–364
 table, 361
 tuning, 362–366, 562
 two-phase locking protocol, 360
 weakening isolation levels for, 364–365
 log file for transactions
 checkpoints, 355, 369, 574–575
 flushing the page buffer, 354, 571, 574–575
 information in, 354
 logging subsystem management, 617
 overview, 353–354, 569
 page buffer for, 353–354, 569, 571
 putting on different disk than transactions, 569–571
 tuning write operations, 572–573
 write-ahead log protocol, 354–355
 logarithms, 215
 logical connectives
 AND, 245–246
 defined, 633
 NOT, 247
 OR, 246, 266–267, 561
 uses for, 245
 with WHERE clause, 245–247
 LOOP . . . END LOOP statements (SQL/PSM), 466
 LOWER function, 211
 lowercase, converting strings to, 211

M

magnetic tape, 568
 maintenance
 costs for indexes, 559
 SDLC phase, 132–133
 Management Studio (SQL Server), 599–601, 610–611

- mandatory cardinality
 - defined, 35, 145
 - examples, 36–37, 145–146
 - importance of, 37
- mantissa for floating-point numbers, 98
- many-to-many relationships
 - in CMCC system, 46
 - converting E-R model to relational model, 166–168, 433–435
 - data structure diagram, 167
 - decomposing, 167–168, 169, 433–435
 - defined, 15, 32, 143
 - E-R diagram, 33, 144, 166
 - in Gentoo Joyce system, 41
 - maximum cardinality, 167
- mapping, defined, 499, 633
- mapping schemas (SQL Server)
 - allowing null values, 533–534
 - attribute-centric mapping, 531–532
 - creating for tables with parent-child relationship, 529–531
 - creating updategram with XDR schema, 528–529
 - creating updategram with XSD schema, 527–528
 - defined, 525
 - deleting a record using a datagram, 526
 - element-centric mapping, 531
 - explicit mapping, 526–531
 - implicit mapping, 525–526
 - inserting an XML element into a record, 525
 - mixed element-centric and attribute-centric mapping, 532
 - updating a record from an XML element, 526
- mapping SQL to XML. *See also* mapping schemas (SQL Server)
 - character sets, 499
 - creating an XML schema for an SQL table, 507–508
 - data types, 500–505
 - distinct UDTs, 502–503
 - domains, 501–502
 - identifiers, 500
 - null values, 506–507
 - tables, 505–506
- MATCH predicate (WHERE)
 - FULL keyword, 244–245
 - general form, 243
 - overview, 241–243
 - PARTIAL keyword, 244, 245
 - referential integrity protected by, 242–245
 - SIMPLE keyword, 244, 245
 - UNIQUE keyword, 244–245
- MAX function, 208
- maxima, extremal queries for, 179
- maximal structured types, 108
- maximum cardinality
 - of arrays, 220
 - in many-to-many relationship, 167
 - in one-to-many relationship, 164
 - in one-to-one relationship, 164
 - Oregon Lunar Society database, 430–431
 - overview, 34–35, 145
- maxInclusive facet (XML), 500–501
- McAfee antivirus software, 341
- .mdb files, 401
- Melton, Jim (*Querying XML*), 103, 535
- memory
 - adding RAM, 583
 - descriptor, 631
 - hierarchy of, 176–177, 567–568
 - non-volatile, 567, 568–569
 - volatile, 567, 568–569
- MESSAGE_LENGTH field (diagnostics area), 391, 395
- MESSAGE_OCTET_LENGTH field (diagnostics area), 391, 395
- MESSAGE_TEXT field (diagnostics area), 391, 395
- metadata, 633

- methods
 - modify() (SQL Server), 523
 - as routines, 65
- Microsoft Access
 - as development environment, 399–401
 - Jet engine, 399–400
 - libraries, 404–405
 - platforms supported, 53
 - QBE (query-by-example) interface, 53
 - SQL editor in, 53–56
 - SQL implementation, 52–53
 - VBA with, 399, 404–405
- Microsoft SQL Server
 - assertions not supported by, 72
 - Database Engine Tuning Advisor, 185–187, 602–607
 - as development environment, 401
 - Express Edition, 56
 - further information, 56
 - Management Studio, 599–601, 610–611
 - mapping schemas for XML, 525–534
 - modify() method, 523
 - .NET classes, 523
 - .NET languages with, 405–406
 - overview, 56
 - Performance Monitor, 610–611
 - SQL Server Profiler, 607–608
 - tools for updating XML data in tables, 523–534
 - updategrams, 523–524, 526, 527–529, 534
- Microsoft SQL Server 2005 Express Edition For Dummies* (Schneider), 56
- MIN function, 209
- minima, extremal queries for, 179
- minimum cardinality
 - data model's effect on, 36–37
 - Oregon Lunar Society database, 431
 - overview, 35–37, 145–147
 - ways of modeling, 38
 - zero versus one, 146–147
- minInclusive facet (XML), 500–501
- MOD function, 215
- modification anomalies
 - cascading deletes to prevent, 174, 284
 - dangers of, 151
 - defined, 17, 151, 284, 633
 - deletion, 151, 174, 284, 631
 - insertion, 151, 633
 - from transitive dependencies, 154
 - update, 636
- modify() method (SQL Server), 523
- MODULE declaration
 - AUTHORIZATION clause, 94, 420
 - LANGUAGE clause, 94, 420
 - NAMES ARE clause, 94, 420
 - SCHEMA clause, 94, 420
 - syntax, 93, 419
- module language
 - declaring modules, 93–94, 419–420
 - defined, 89, 93, 634
 - embedded SQL versus, 93, 419
 - module procedures, 320–321
 - modules in Oracle, 421
 - overview, 93–94, 418–419
 - SQL procedures in, 418–419
 - using SQLSTATE with, 385–386
- Moore's Law (Gordon Moore), 24
- MORE field (diagnostics area), 390, 391, 395
- multi-column indexes, 557
- multipoint queries
 - clustered indexes for, 314
 - described, 179
- multiprocessor environments, 585
- MULTISET type
 - mapping to XML, 505
 - 1NF violated by, 105
 - overview, 105
- multisets, 105
- multi-table views
 - creating tables for, 75
 - creating views, 75–77
 - uses for, 73–74

multi-valued fields, ROW type for, 104
mutator functions, 108, 634

MySQL

assertions not supported by, 72
as development environment, 402
overview, 57
platforms supported, 57
procedural languages supported,
406–407

N

NAMES ARE clause (MODULE), 94, 420

namespace for updategrams (SQL
Server), 523

naming or renaming. *See also* aliases
avoiding reserved words, 94
columns, fully qualified names for, 73
columns in equi-joins, 306
constraints, 70
cursors, 316–317
implicit constraint names, 359
mapping identifiers to XML, 500
UPDATE statement for, 83–84

NATIONAL CHARACTER LARGE OBJECT
type, 100

NATIONAL CHARACTER type, 100

NATIONAL CHARACTER VARYING type,
100

native drivers

ODBC versus, 481, 591–592
overview, 475–476

natural joins, 307, 308

nested elements (XML), 494

nested queries, 269–270, 634. *See also*
subqueries

nested-loop joins, 614

.NET languages

embedding SQL in, 418
Oracle support for, 402
SQL Server classes, 523
SQL Server support for, 405–406

NetBEUI, 634

Net.Data, DB2 support for, 402

Netscape plug-in, 634

network database model

defined, 634

development of, 14, 15

diagram, 19

overview, 17

relational model versus, 20, 23, 24–25

networks

connecting to databases over, 64–65,
475–487

firewalls for, 340, 632

performance issues, 597

protecting against worms, 339

worms spread over, 338

New York Board of Trade (NYBOT),
335–336

N:M relationships. *See* many-to-many
relationships

non-procedural languages

overview, 411–412

SQL as, 411

XQuery as, 536

nonrepeatable reads, 351

non-unique identifiers, 31, 141

non-volatile memory, 567, 568–569

normalization

1NF (First Normal Form), 104, 151,
152–153

2NF (Second Normal Form), 152, 153–154

3NF (Third Normal Form), 152, 154

4NF (Fourth Normal Form), 152, 155

5NF (Fifth Normal Form), 152, 155

BCNF (Boyce-Codd Normal Form),
152, 155

defined, 634

DKNF (Domain/Key Normal Form),
152, 155–158, 162, 163

for eliminating anomalies, 152–156, 345
of E-R model, 435–437

overnormalization for performance, 555

- performance tradeoff with, 156–158, 162, 163
 - of relational model, 162–163
 - for simplifying relationships, 47
 - Norton antivirus software, 341
 - NOT DEFERRABLE constraints, 356
 - not equal symbol (<>) in comparison operators, 230
 - NOT EXISTS predicate (WHERE), 278–279
 - NOT IN predicate (WHERE), 232
 - NOT LIKE predicate (WHERE), 234
 - NOT logical connective, 247
 - NOT NULL constraint
 - as column constraint, 70
 - in CREATE TABLE statement, 69, 112
 - DEFERRABLE, 356
 - for entity integrity, 171
 - for ID columns, 70
 - overview, 111–112
 - for primary key, 112
 - NOT NULL predicate (WHERE), 236
 - NULL predicate (WHERE), 235–236
 - null values
 - blank spaces not same as, 69, 110
 - with BOOLEAN type, 101, 110
 - COALESCE expression for bypassing, 223
 - defined, 69, 110
 - DISTINCT versus UNIQUE predicate with, 240–241
 - FALSE value not same as, 110
 - mapping schemas allowing (SQL Server), 533–534
 - meaningless in comparisons, 236
 - migrating from databases not supporting, 222–223
 - NOT NULL constraint for, 69–70, 111–112
 - overview, 110–111
 - query optimizer limitations for, 561
 - reasons for, 111
 - XML representation for, 506–507
 - XSD schema allowing, 533
 - zero not same as, 69, 110
 - NULLIF expression, 223
 - NUMBER field (diagnostics area), 390, 395
 - numeric character references (XML), 496
 - numeric data types. *See also specific types*
 - approximate, 97–98
 - exact, 95–97
 - NUMERIC type, 96, 97
 - numeric value expressions, 217, 637
 - numeric value functions
 - ABS, 214–215
 - CARDINALITY, 214, 220
 - CEIL or CEILING, 216
 - CHARACTER_LENGTH, 213
 - EXP, 215
 - EXTRACT, 213
 - FLOOR, 216
 - list of, 212
 - LN, 215
 - MOD, 215
 - OCTET_LENGTH, 213–214
 - POSITION, 212–213
 - POWER, 215
 - SQRT, 215
 - WIDTH_BUCKET, 216
 - NYBOT (New York Board of Trade), 335–336
-
- 0
-
- Object Linking and Embedding Database (OLE DB)
 - Access support for, 400
 - DB2 support for, 401
 - MySQL support for, 402
 - SQL Server support for, 401
 - object-oriented database management systems (OODBMS), 15, 23
 - object-oriented procedural languages, 411, 413

- object-relational databases, 15, 23
- objects
 - defined, 634
 - owners of, 374
 - privileges applying to, 375
- observer functions, 108
- OCTET_LENGTH function, 213–214
- ODBC (Open Database Connectivity)
 - Access support for, 400
 - application, 478
 - C code example, 486–487
 - data sources, 482
 - DBMS-based drivers, 480–481
 - DB2 support for, 401
 - defined, 477, 634
 - driver manager, 477–478, 479
 - drivers, 479–481
 - file-based drivers, 480
 - handles, 482–483
 - MySQL support for, 402
 - native drivers versus, 481, 591–592
 - one-tier driver system, 480
 - Oracle support for, 402
 - overview, 477–478
 - requests from applications, 482–487
 - six stages of operation, 484–486
 - SQL Server support for, 401
 - two-tier driver system, 481
- ODBC.NET, Oracle support for, 402
- offline storage, 177
- OLAP (online application processing), 216
- OLE DB (Object Linking and Embedding Database)
 - Access support for, 400
 - DB2 support for, 401
 - MySQL support for, 402
 - SQL Server support for, 401
- OLE.NET, Oracle support for, 402
- ON clause
 - for condition joins, 307–308
 - WHERE clause with joins versus, 313
- ON DELETE CASCADE clause, 174. *See also* cascading deletes
- 1 RAID level, 334, 335, 584
- 01 SQLSTATE class value, 384, 386
- 1NF (First Normal Form), 104, 151, 152–153
- one-tier driver system (ODBC), 480
- one-to-many relationships
 - converting E-R model to relational model, 165–166
 - data structure diagram, 166
 - defined, 15, 32, 143
 - E-R diagram, 33, 143–144, 165
 - in Gentoo Joyce system, 41
 - maximum cardinality, 164
- one-to-one relationships
 - converting E-R model to relational model, 164–165
 - data structure diagram, 164–165
 - defined, 15, 32, 143
 - E-R diagram, 32, 144, 164
 - maximum cardinality, 164
- online application processing (OLAP), 216
- OODBMS (object-oriented database management systems), 15, 23
- Open Database Connectivity. *See* ODBC
- OPEN statement
 - datetime values fixed by, 321, 322
 - examples, 321
 - syntax, 320
 - variable values fixed by, 322
- opening cursors, 320–322
- open-source DBMS products, 57
- OPENXML function (SQL Server), 523
- operating system tuning
 - determining page buffer size, 579–580
 - scheduling threads, 575–579
 - tuning page usage factor, 580
- operating systems. *See* platforms
- optimization. *See* database tuning; performance; query tuning; tuning

- optional cardinality
 - defined, 35–36
 - example, 36
 - importance of, 37
- OR logical connective
 - indexes with, 266–267
 - overview, 246
 - query optimizer limitations for, 561
- Oracle
 - defined, 634
 - as development environment, 402
 - embedding SQL in Pro*C application, 414–416
 - functions for updating XML data in tables, 518–523
 - initial release of, 49
 - Java with, 407
 - modules in, 421
 - native driver for 10g, 475–476
 - overview, 56–57
 - platforms supported, 56
 - version 10g, 402
- order by clause (FLWOR), 545–546
- ORDER BY clause (SELECT)
 - ASC versus DESC keyword for, 251, 318
 - clustered indexes eliminating need for, 314
 - COLLATE BY clause with, 317, 318
 - cursors with, 317–318
 - default sort order, 251, 318
 - GROUP BY clause with, 249
 - with left outer joins, 311
 - for ordering queries, 180
 - overview, 228, 249–251
 - performance issues, 259
 - query tuning, 259–262
- ordering queries, 180
- Oregon Lunar Society database
 - adding constraints, 448–449
 - building E-R model, 429–432
 - business rules, 431–432
 - client’s problem, 423–424
 - connecting user interface to database, 445–447
 - converting E-R model to relational model, 432–437
 - creating tables, 437–440, 448
 - creating the database, 448–449
 - determining deliverables, 426–429, 444
 - developing reports, 455
 - developing screen forms, 454
 - eliminating many-to-many relationships, 433–435
 - filling tables with sample data, 449–454
 - integrating the structure, 455
 - interviewing stakeholders, 424–425
 - maximum cardinality, 430–431
 - minimum cardinality, 431
 - normalizing the E-R model, 435–437
 - planning for organization growth, 427–428
 - proposal, 425–426
 - scope of project, 428–429
 - Statement of Requirements, 425
 - testing, 455–457
 - top-down approach, 443–447
 - updategram using nillable mapping schema, 534
 - user interface design, 441, 444–445
- organization growth, planning for, 427–428
- outer joins
 - described, 310
 - full, 313
 - inner joins versus, 309
 - left, 310–312
 - left and right tables, defined, 310
 - right, 312
- OVERLAPS predicate (WHERE), 241
- OVERLAY function, 211–212
- overnormalization for performance, 555
- owners of objects, 374

p

- page buffer for database
 - buffer manager, 617
 - page replacement algorithm, 581
 - sizing, 579–580
 - temporal locality, 581
 - usage factor, 580
- page buffer for transactions. *See also* checkpoints
 - flushing, 354, 571, 574–575
 - overview, 353–354, 569
 - tuning write operations, 572–573
- page locks, 361. *See also* locks
- parameter, defined, 634
- PARAMETER_MODE field (diagnostics area), 392
- PARAMETER_NAME field (diagnostics area), 392
- PARAMETER_ORDINAL_POSITION field (diagnostics area), 392
- PARTIAL keyword (MATCH), 244, 245
- partitioning transactions
 - accessing hot spots, 366
 - insertions, 365
 - locating hot spots, 598
 - for performance, 597–598
- patches, security, 341
- PCP (Priority Ceiling Protocol), 578
- percent sign (%)
 - as LIKE wildcard, 234
 - using literally in LIKE predicate, 234–235
- performance. *See also* database tuning; query tuning; tuning
 - analyzing query efficiency, 598–615
 - application/database interaction issues, 594
 - avoiding direct user interaction for, 593
 - bottlenecks, defined, 587
 - client, locking, 592
 - communication issues, 591–592
 - of correlated subqueries versus relational queries, 290–295
 - cursor issues, 594
 - database dump’s impact on, 369
 - Database Engine Tuning Advisor, 185–187, 602–607
 - DBMS issues, 12
 - decisions affecting, 565–566
 - DISTINCT keyword’s impact on, 252–254
 - DKNF issues, 156–158, 162, 163
 - eliminating GROUP BY clause for, 264–266
 - filter ratio’s impact on, 184
 - flat file advantages, 11
 - general ways of improving, 593–595
 - gradual degradation of, 587–588
 - hard disk considerations, 570–571, 583–585, 596–597
 - hardware issues, 580–585, 593, 596–597
 - of hierarchical model, 17
 - importance of, 565
 - IN predicate issues, 233
 - increased computer speeds for, 12
 - index issues, 589–591
 - indexes’ impact on, 555
 - integrity tradeoff with, 156–158, 159
 - isolating problems, 595
 - keys to, 553
 - lock tuning for, 362–366
 - managing resources for, 615–618
 - Microsoft SQL Server Management Studio, 610–611
 - monitors for, 608–611
 - Moore’s Law for, 24
 - of nested versus relational queries, 285–289
 - network issues, 597
 - ORDER BY clause’s impact on, 259–262
 - ordering HAVING and GROUP BY clauses for, 262–264
 - overnormalization for, 555

- partitioning for, 597–598
- physical design considerations, 555
- pinpointing problems, 588
- precompiling queries for, 563, 595
- query speed as measure of, 553
- recovery system tuning for, 369
- serializability issues for, 348
- SERIALIZABLE isolation costs, 352
- slow query issues, 588
- slow update issues, 588
- storage types' effect on, 176–177
- temporary tables' impact on, 255–262
- throughput, 363, 576–577
- top-down analysis for, 595–597
- tracking down problems, 595–598
- volatile memory used for, 568
- Performance Monitor (SQL Server), 610–611
- Perl scripts
 - embedding SQL in, 417
 - with MySQL, 406
- persistent storage, 567
- Persistent Stored Modules. *See* SQL/PSM
- personal databases, 631
- phishing scams, 339
- PHP
 - embedding SQL in, 417
 - with MySQL, 407
- physical design considerations, 555
- PIP (Priority Inheritance Protocol), 578
- planning your database, 189
- platforms
 - Access support for, 53
 - DB2 support for, 56
 - defined, 329
 - instability issues, 329–330
 - InterBase support for, 57
 - MySQL support for, 57
 - Oracle support for, 56
 - PostgreSQL support for, 57
 - upgrading or changing, 329–330
- PL/SQL, Oracle support for, 402
- point queries, 178–179
- portability
 - DBMS advantages, 12
 - flat file issues, 11
 - NUMERIC type for, 97
- POSITION function, 212–213
- PostgreSQL, 57, 72
- POWER function, 215
- precision
 - for approximate numeric types, 97–98
 - defined, 634
 - double, 98
 - for exact numeric types, 95–97
 - single, 97–98
- pre-compiler for embedded SQL, 91
- precompiling queries, 563, 595
- predicates. *See also specific predicates*
 - comparison, for WHERE clause, 229–245
 - defined, 512, 634
 - for XML, 512–514
- prefix match queries, 179
- primary key
 - CLOB type not allowed for, 100
 - defined, 634
 - DISTINCT keyword not needed for selects, 252
 - for indexes, 194
 - NOT NULL constraint for, 112
- PRIMARY KEY constraint
 - for entity integrity, 171–172
 - for ID columns, 114
 - for multiple columns, 113
 - overview, 71, 113–114
 - with single-column key, 113
 - as table constraint, 71
- Priority Ceiling Protocol (PCP), 578
- Priority Inheritance Protocol (PIP), 578
- priority inversion with threads, 578
- priority-based thread scheduling, 577–578

privileges. *See also* GRANT statements
for adding data, 376
for all actions, dangers of, 379
for changing data, 376
classes of users for, 373–374
DCL statements for, 86–87, 371
for defining UDTs, 378
delegating, 379
for deleting data, 375
for executing SQL statements, 375,
378–379, 473
granted by database owner, 473
granting, 86–87, 371, 374–379, 381
identifying authorized users for,
372–373
for looking at data, 375
in module declaration, 94, 420
objects applied to, 375
overview, 63–64
for referencing data in another table,
376–378, 473
referential integrity issues, 376–378
for responding to events, 378
revoking, 87, 371, 380–382
roles for, 64, 372–373, 381–382
for using database facilities, 378
Pro*C language (Oracle), 414–416
procedural capabilities of SQL. *See* SQL/
PSM (Persistent Stored Modules)
procedural languages. *See also* embedded
SQL; module language; *specific
languages*
challenges using SQL with, 412–413
with DB2, 407–408
defined, 315, 634
with Microsoft Access, 404–405
in module declaration, 94, 420
with MySQL, 406–407
with Oracle, 407
SQL compared to, 409–412
with SQL Server, 405–406
SQL with, 315
using SQLSTATE with, 385–386

PROCEDURE statement, 420–421
procedures, as routines, 65
processing instructions for XML, 511
processor or CPU
multiprocessor environments, 585
performance issues, 583, 597
performance monitors, 608–611
registers, 176, 567, 568
upgrading, 583
programming errors, reducing, 345–346
project team, forming, 122
proposal for database application,
425–426
PUBLIC keyword (GRANT), 87
public users, 374
Python, SQL compared to, 411

Q

quantified subqueries, 272, 275–277
QUEL data sublanguage, 411
queries. *See also* query tuning;
subqueries; XQuery
defined, 635
determining complexity of, 554
DML for, 79–80
importance of optimizing, 14
index-only, 590
nested, 269–270, 634
precompiling, 563, 595
query analyzers, 599–608
storage and performance of, 176–177
types of, 178–180
query analyzers
Database Engine Tuning Advisor,
602–607
overview, 599
SQL Server Management Studio,
599–601, 610–611
SQL Server Profiler, 607–608
query optimizers
composite indexes with, 182–183
limitations of, 251, 561

- query tuning
- analyzing access plans, 611–614
 - analyzing query efficiency, 598–615
 - for correlated subqueries, 290–295
 - Database Engine Tuning Advisor for, 602–607
 - execution profiles for, 615
 - finding problem queries, 611–615
 - for GROUP BY clause, 262–266
 - for HAVING clause, 262–264
 - importance of, 14
 - Microsoft Server Management Studio for, 599–601
 - need for, 251
 - for OR logical connective, 266–267
 - for ORDER BY clause, 259–262
 - overview, 251–252, 561–562
 - Performance Monitor for, 610–611
 - query analyzers for, 599–608
 - for SELECT DISTINCT, 252–254
 - SQL Server Profiler for, 607–608
 - for statements containing subqueries, 285–290
 - for temporary tables, 255–262
 - unnecessary columns, avoiding, 594
- Querying XML* (Melton and Buxton), 103, 535
- quotation marks (“”), XML entity reference for, 495
-
- R**
-
- RAD (rapid application development) tool, 635
- RAID (Redundant Array of Inexpensive Disks)
- advantages, 331
 - comparison of levels, 335
 - fault tolerance with, 331, 334–335
 - logical disk drives with, 331
 - overview, 331, 584–585
 - performance considerations, 584–585
 - RAID 0 level, 332, 335, 584
 - RAID 1 level, 334, 335, 584
 - RAID 5 level, 334, 335, 584–585
 - RAID 10 level, 334, 335, 585
 - striping, 332, 333
- RAM, adding, 583
- range queries, 179
- rapid application development (RAD) tool, 635
- RDML data sublanguage, 411
- READ COMMITTED isolation level
- compared to other levels, 352
 - for locking, 364–365
 - overview, 351
- READ UNCOMMITTED isolation level
- compared to other levels, 352
 - dirty reads allowed by, 350
 - uses for, 350–351
 - warning for, 350
- read/write time for hard disk, 570
- REAL type, 97–98
- rebuilding “tired” indexes, 314, 560–561, 589
- records. *See also* rows; tuples
- defined, 635
 - inserting XML element into, 525
 - other terms for, 148
 - updategram for deleting, 526
 - updating from XML element, 526
- recovery system. *See also* ROLLBACK operation
- backups, 335–337
 - checkpoints, 355, 369, 574–575
 - dumps, 369, 573–574
 - logging subsystem management, 617
 - optimizing batch transactions, 575
 - putting logs and transactions on different disks, 569–571
 - testing the restore process, 337
 - tuning the system, 369, 567–575
 - tuning write operations, 572–573
- redundancy, data, 17, 630
- Redundant Array of Inexpensive Disks. *See* RAID

- REF types, 106
- reference types, 635
- referential integrity
 - defined, 172–173, 198, 376, 635
 - losing, 241–242
 - maintaining, 172–174
 - MATCH predicate for protecting, 242–245
 - privilege issues for, 376–378
- registers
 - as fast storage, 176, 567
 - overview, 176
 - in storage hierarchy, 568
- relational algebra, 297
- relational database model
 - building, 147–150
 - Codd's rules for, 20, 22–23
 - converting E-R model to, 161–170, 432–437
 - defined, 20
 - diagram, 21
 - dominance of, 24–25
 - flexibility of, 23
 - normalizing, 162–163
 - overview, 20–23
 - set theory as basis of, 59–60
 - slow to catch on, 24
 - SQL concepts, 59–66
 - terminology for elements, 147–149, 161–162, 432–433
 - translating users' data model to, 29–47
- relational databases
 - ANSI/ISO compliance, 2
 - development of, 49–50
 - hierarchy in, 47, 67–68
 - as object-relational, 23
- relational operators, 297. *See also specific operations*
- Relational Software, first RDBMS
 - released by, 49
- relations. *See also* entities; tables
 - bad, defined, 150
 - characteristics of, 60
 - as collections of tuples, 59
 - criteria for, 149
 - defined, 149, 635
 - entities comparable to, 148, 432
 - keys required for, 62
 - other terms for, 148
 - tables compared to, 60, 149
 - translating entities into, 161–162, 432–433
- relationships. *See also* E-R (Entity-Relationship) model; many-to-many relationships; one-to-many relationships; one-to-one relationships
 - binary (degree-two), 32–33, 142, 143–144
 - classes, 142
 - complex (degree-three), 33–34, 142–143
 - described, 31
 - determining, 430
 - establishing between tables, 195–198
 - functional dependencies, 60–61, 149–150
 - Honest Abe's database, 161
 - instances, 142
 - normalizing into DKNF, 163
 - supertype/subtype, 39–40
- relative fetches, 323
- Remember icon, 5
- remote connections
 - JDBC for, 591
 - native drivers for, 475–476, 591
 - ODBC for, 477–487, 591–592
 - overview, 64–65
- removing. *See* deleting or removing
- renaming. *See* naming or renaming
- REPEATABLE READ isolation level
 - compared to other levels, 352
 - for locking, 364
 - overview, 351
- REPEAT...UNTIL...END REPEAT statements (SQL/PSM), 467

- reports, developing, 455
- requirements phase of SDLC
 - constructing users' data model, 124, 139
 - establishing requirements, 123–124, 135–138
 - stakeholders, 123, 135–138
 - Statement of Requirements, 124–125
 - summary of tasks, 125
 - three-option proposal, 139
- reserved words
 - defined, 635
 - list of (SQL:2003), 621–627
 - using correctly, 94
- restoring from backups, testing, 337
- RESTRICT keyword (REVOKE), 380, 382
- return clause (FLWOR), 543, 546–547
- returnability of cursors, 317
- RETURNED_SQLSTATE field (diagnostics area), 391, 392, 396
- REVOKE statements
 - CASCADE keyword, 380, 382
 - GRANT OPTION FOR clause, 380
 - GRANTED BY clause, 380, 381–382
 - overview, 87, 380–381
 - RESTRICT keyword, 380, 382
 - for roles, 381–382
 - syntax, 87, 380
- right outer joins, 312
- robust execution plans, 183–184
- roles
 - creating, 373
 - destroying, 373
 - granting privileges for, 381
 - overview, 64, 372–373
 - revoking privileges for, 381–382
- ROLLBACK operation
 - checkpoints for, 355
 - COMMIT operation versus, 352
 - described, 87–88, 352–353
 - log file for, 353–354
 - reasons for, 353
 - write-ahead log protocol for, 354–355
- root node, 519, 589
- rotational latency of hard disk, 570, 571
- rounding, functions for, 216
- round-robin thread scheduling, 577
- ROUTINE_CATALOG field (diagnostics area), 392
- ROUTINE_NAME field (diagnostics area), 392
- routines. *See also* functions
 - defined, 65
 - external, 65
 - externally invoked, 65
 - overview, 65–66
 - SQL, 65
 - SQL-invoked, 65
- ROUTINE_SCHEMA field (diagnostics area), 392
- row locks, 361. *See also* locks
- ROW type, 104, 503–504
- row value expressions, 225, 635
- row values, 203–204
- ROW_COUNT field (diagnostics area), 390, 395
- row-level triggers, 471
- rows. *See also* records; tuples
 - defined, 635
 - deleting row cursor points to, 323
 - fetching data with cursors, 322–323
 - finding number in table, 207–208
 - locating with keys, 191
 - not satisfying a condition, queries for, 271–272
 - other terms for, 148
 - satisfying a condition, queries for, 270–271
 - transferring between tables, all, 452–453
 - transferring between tables, selected, 453–454
 - updating row cursor points to, 324

S

- scale, 96, 635
- scheduling threads
 - context switching, 577
 - deadlocks, 579
 - overview, 575–576
 - priority inversion, 578
 - priority-based scheduling, 577–578
 - round-robin scheduling, 577
 - throughput improved by, 576–577
- SCHEMA clause (MODULE), 94, 420
- schema owner, 635
- SCHEMA_NAME field (diagnostics area), 391, 395
- schemas. *See also* catalogs; XML schemas
 - creating, 77
 - in database hierarchy, 47, 67–68
 - default, 77
 - defined, 635
 - in module declaration, 94, 420
 - overview, 64
- Schneider, Robert (*Microsoft SQL Server 2005 Express Edition For Dummies*), 56
- scope of project, determining, 122, 125, 126–127, 428–429
- screen forms, developing, 454
- scrollability of cursors, 317, 320
- SDLC (System Development Life Cycle). *See also specific phases*
 - definition phase, 122
 - design phase, 127–130
 - evaluation phase, 125–127
 - final documentation and testing phase, 130–132
 - implementation phase, 130
 - maintenance phase, 132–133
 - phases of, 121
 - requirements phase, 123–125
- searched CASE statements (SQL/PSM), 465
- Second Normal Form (2NF), 152, 153–154
- security issues. *See also* anomalies; integrity; privileges
 - ACID as protection against, 347–348
 - backups as protection against, 335–337
 - corruption, 174–175, 346–347
 - data entry errors, 330–331, 343–345
 - database design flaws, 330, 345
 - equipment failure, 328–329, 331
 - fault tolerance, 331, 334–335
 - for flat files, 11
 - GRANT ALL PRIVILEGES statement, 379
 - Internet threats, 337–341
 - operator error, 331
 - platform instability, 329–330
 - programming errors, 345–346
 - RAID as protection against, 331–335
 - referential integrity, 376–378
 - sources of problems, 327–328
 - tuning the recovery system, 369, 567–575
 - WITH GRANT OPTION clause, 379, 380–381
- seek time for hard disk, 570, 571
- SELECT keyword, 87
- SELECT statements. *See also* queries; *specific clauses*
 - FLWOR expressions (XQuery) compared to, 547
 - granting privileges for, 375
 - overview, 227–228
 - simple form, 79
 - wildcard with, 79, 227–228, 299
- semicolon (;) ending XML entity
 - references, 495
- sensitivity of cursors, 317, 319–320
- September 11, 2001, 335–336
- SEQUEL, 50, 635

- serializability
 - integrity provided by, 360
 - locking for, 360–366
 - performance issues, 348
 - timestamps for, 366–369
- SERIALIZABLE isolation level, 352
- SERVER_NAME field (diagnostics area), 391
- sessions, 65, 372
- SESSION_USER variable, 206
- SET clause (UPDATE), 82
- SET CONSTRAINTS ALL DEFERRED statement, 356
- SET CONSTRAINTS ALL IMMEDIATE statement, 356
- SET CONSTRAINTS DEFERRED statement, 358
- SET CONSTRAINTS IMMEDIATE statement, 359
- set or aggregate functions
 - AVG, 208
 - COUNT, 207–208, 240
 - defined, 207, 629, 635
 - MAX, 208
 - MIN, 209
 - overview, 207–209
 - SUM, 209
 - for XML documents, 510
- SET statement (SQL/PSM), 463
- set theory, 59–60
- SET TRANSACTION statement
 - access mode settings, 348–349, 350
 - applied to next transaction, 349
 - diagnostic size settings, 349, 390
 - isolation level settings, 349, 350–352
 - LOCAL keyword, 349
 - overview, 348–349
 - START TRANSACTION statement
 - versus, 349–350
 - syntax, 348–349
- settling time for hard disk, 570, 571
- SGML (Standard Generalized Markup Language), 491
- shared locks, 360. *See also* locks
- shared-disk architecture, 585
- shared-nothing architecture, 585
- SIMILAR predicate (WHERE), 235
- simple CASE statements (SQL/PSM), 464–465
- SIMPLE keyword (MATCH), 244, 245
- single precision, 97–98
- single-table views, creating, 72–73
- SMALLINT type, 96
- soft failures, 574
- SOME predicate (WHERE)
 - overview, 238–239
 - for quantified comparison operators, 272, 275
 - syllogisms illustrating, 236–237
- sorting. *See also* ORDER BY clause (SELECT)
 - by clustered indexes, 314
 - COLLATE BY clause for, 317, 318
- sort-merge joins, 614
- sparse indexes, 181
- special variables for logging users, 206
- SPECIFIC_NAME field (diagnostics area), 392
- SQL. *See also* embedded SQL; module language
 - casting between XQuery and SQL types, 549
 - as data sublanguage, 89, 315, 411, 631
 - defined, 635
 - development of, 1, 49–50
 - dynamic, 635
 - executing statements, 89
 - implementations of, 52–57
 - interactive, 89, 90, 636
 - limitations of, 51
 - mapping to XML, 499–508
 - as non-procedural language, 90

- SQL (*continued*)
 - privileges for executing statements, 375, 378–379, 473
 - procedural languages compared to, 409–412
 - as Structured Query Language, 50
 - uses for, 50
 - XML compared to, 497
 - XQuery compared to, 547–549
- SQL routines, 65
- SQL Server. *See* Microsoft SQL Server
- SQL Server Profiler, 607–608
- SQLAllocHandle function (ODBC), 484, 485
- SQLConnect function (ODBC), 484, 485
- SQLDriverConnect function (ODBC), 484, 485
- SQL/DS RDBMS product, 50, 636
- SQLExecDirect function (ODBC), 485
- SQL-invoked routines, 65
- SQLJ (Java-based Embedded SQL), 402, 407
- SQL*Module language (Oracle), 421
- SQL/PSM (Persistent Stored Modules)
 - ANSI/ISO SQL standard for, 459
 - compound statements, 460–463
 - embedding SQL in code, 459–460
 - flow of control statements, 463–468
 - granting privileges, 473
 - stored functions, 472
 - stored modules, 473–474
 - stored procedures, 469
 - triggers, 469–472
- SQL-session user identifier, 372
- SQLSetConnectOption function (ODBC), 485
- SQLSTATE status parameter
 - checking after statement execution, 385
 - class codes, 384–385, 386, 463
 - for compound statements, 463
 - diagnostics area information, 394–395
 - in module language programs, 385–386, 420–421
 - overview, 384–386
- WHENEVER directive for, 383, 388–389
- SQRT (square root) function, 215
- stakeholders. *See also* client for project building consensus, 138–139
 - identifying, 27–28
 - interviewing, 28, 136–137, 138, 424–425
 - lines of authority among, 135
 - obtaining buy-in from, 29
 - reconciling conflicting requirements, 28–29, 138
 - in requirements phase of SDLC, 123
 - standards organization, 137
 - upper management, 137–138
 - users, 136–137
 - variety of, 135
 - your immediate supervisor, 136
- standard for SQL. *See* ANSI/ISO SQL standard
- Standard Generalized Markup Language (SGML), 491
- standards, company, 137
- START TRANSACTION statement, 349–350
- statement handle (ODBC), 482, 483
- Statement of Requirements, 29, 124–125, 425
- statement-level triggers, 471
- storage. *See also* hard disks
 - hierarchical, 176–177
 - persistent, 567
 - query performance and types of, 176–177
- storage requirements
 - cost of storage, 12
 - DBMS issues, 12
 - flat file advantages, 10
 - in hierarchical model, 17
- stored functions (SQL/PSM)
 - overview, 472
 - in stored modules, 473–474
- stored modules (SQL/PSM), 473–474

- stored procedures (SQL/PSM)
 - overview, 469
 - in stored modules, 473–474
 - stored routines (SQL/PSM), 472
 - string value expressions
 - defined, 637
 - overview, 218
 - parsing to produce XML value, 511
 - string value functions
 - CONVERT, 211
 - departures from ANSI/ISO standard, 210
 - list of, 209–210
 - LOWER, 211
 - OVERLAY, 211–212
 - SUBSTRING (FROM), 210
 - SUBSTRING (SIMILAR), 210
 - TRANSLATE, 211
 - TRIM, 211
 - UPPER, 211
 - striping (RAID), 332, 333
 - strong entities, 37–38
 - strong typing, 344, 345
 - Structured Query Language, 50
 - structured types
 - defined, 107, 636
 - example, 108–109
 - leaf, 108
 - maximal, 108
 - mutator functions for, 108
 - observer functions for, 108
 - subtypes and supertypes, 108
 - SUBCLASS_ORIGIN field (diagnostics area), 391, 392, 393
 - subqueries. *See also* WHERE clause (SELECT)
 - comparison operators with, 272–274
 - correlated, 277–282, 290–295
 - defined, 269, 636
 - in DELETE statements, 284
 - DISTINCT predicate with, 240–241
 - EXISTS predicate with, 239–240
 - IN predicate with, 233
 - in INSERT statements, 284–285
 - quantified, 275–277
 - retrieving rows not satisfying a condition, 271–272
 - retrieving rows satisfying a condition, 270–271
 - returning multiple values, 270–272
 - returning single value, 272–277
 - tuning statements containing, 285–290
 - UNIQUE predicate with, 240
 - in UPDATE statements, 282–284
 - SUBSTRING (FROM) function, 210
 - SUBSTRING (SIMILAR) function, 210
 - subtypes, 108, 636
 - SUM function, 209
 - supertype and subtype entities, 39–40
 - supertypes, 108, 636
 - System Development Life Cycle. *See* SDLC
 - system failures. *See* equipment failure
 - SYSTEM_USER variable, 206
-
- T
-
- table locks, 361. *See also* locks
 - TABLE_NAME field (diagnostics area), 391, 395
 - tables. *See also* CREATE TABLE statements; relations; views or virtual tables
 - adding constraints to existing, 394
 - adding data, 80–81
 - altering structure of, 78, 198, 440
 - constraints, 71, 113–114, 194
 - creating an XML schema for, 507–508
 - creating for XML data, 517
 - in database hierarchy, 47, 67–68
 - defined, 636
 - deleting, 199
 - deleting data, 84–85
 - distinct types with, 107
 - dropping, 78

tables *(continued)*

- establishing relationships between, 195–198
 - filling with sample data, 449–454
 - filter ratio, 184
 - finding number of rows in, 207–208
 - full table scans, 177, 183, 590–591
 - hot, 187
 - indexes and size of, 183
 - inserting XML data into SQL pseudo-table, 515–517
 - left versus right, in joins, 310
 - locating rows with keys, 191
 - mapping to XML, 505–506
 - modifying, 78
 - other terms for, 148
 - privileges applying to, 375
 - relations compared to, 60, 149
 - temporary, query tuning for, 255–262
 - transferring all rows between, 452–453
 - transferring selected columns and rows between, 453–454
 - union-compatible, 297, 303
 - updating data, 81–84
 - usage in this book, 60
- Taylor, Allen G. (*Database Development For Dummies*), 330
- TCP/IP (Transmission Control Protocol/Internet Protocol), 636
- Technical Stuff icon, 5
- teleprocessing system, 636
- temporal locality, 581
- temporal partitioning, 598
- temporary tables
- for multiple selection conditions, 255–256
 - ORDER BY clause with, 259–262
 - query tuning for, 255–262
- 10 RAID level, 334, 335, 585
- testing
- beta testing, 456
 - database applications, 131–132, 455–457
 - restore process, 337
 - testers for, 456
- threads
- context switching, 577
 - deadlocks, 579
 - overview, 575–576
 - priority inversion, 578
 - priority-based scheduling, 577–578
 - round-robin scheduling, 577
 - throughput improved by, 576–577
- 3NF (Third Normal Form), 152, 154
- three-option proposal, 139
- throughput
- defined, 363, 576
 - improved by threads, 576–577
- tightly coupled architecture, 585
- TIME WITH TIME ZONE type, 102
- TIME WITHOUT TIME ZONE type, 101–102
- TIMESTAMP WITH TIME ZONE type, 102–103
- TIMESTAMP WITHOUT TIME ZONE type, 102
- timestamps
- datetime types for, 102–103
 - defined, 366
 - enforcing serializability with, 366–369
 - livelocks with, 367–369
 - update example, 366–367
- Tip icon, 5
- “tired” indexes, rebuilding, 314, 560–561, 589
- TRANSACTION_ACTIVE field (diagnostics area), 390, 391
- transactions
- aborting, 566
 - access modes, 348–349, 350
 - ACID characteristics for, 347–348
 - batch, optimizing, 575
 - COMMIT operation, 88
 - committing, 352, 566
 - for data integrity protection, 87–88

- default, 348
 DEFERRABLE constraints for, 355–359
 defined, 65, 87, 347
 enforcing serializability with
 timestamps, 366–369
 hot spots, 365–366, 598
 isolation levels, 350–352
 locking, 360–366
 log file for, 353–355
 page buffer for, 353–354, 569, 571
 partitioning, 365, 366
 putting on different disk than logs,
 569–571
 ROLLBACK operation, 87–88, 352–355
 running DDL concurrently with,
 avoiding, 365
 separating user interactions from, 562
 SET TRANSACTION statement for,
 348–349
 starting, 349–352
 timestamps for, 366–369
 tuning the recovery system, 369
 tuning transactions, 562
 TRANSACTIONS_COMMITTED field
 (diagnosics area), 390, 391
 TRANSACTIONS_ROLLED_BACK field
 (diagnosics area), 390, 391
 transitive dependencies, 154, 636
 TRANSLATE function, 211
 translation, privileges applying to, 375
 translation tables, 636
 Transmission Control Protocol/Internet
 Protocol (TCP/IP), 636
 triage, 27
 TRIGGER_CATALOG field (diagnosics
 area), 392
 TRIGGER_NAME field (diagnosics area),
 392
 triggers
 action time, 471
 actions, 471
 creating, 470–472
 defined, 636
 events, 471
 overview, 469–470
 privileges for, 375, 378
 row-level, 471
 statement-level, 471
 triggered SQL statement, 472
 TRIGGER_SCHEMA field (diagnosics
 area), 392
 TRIM function, 211
 tuning. *See also* database tuning;
 performance; query tuning
 defined, 553
 hardware considerations, 580–585
 indexes, 560–561
 locks, 362–366
 multiprocessor environments, 585
 operating system, 575–580
 page usage factor, 580
 recovery system, 369, 567–575
 transactions, 562
 write operations, 572–573
 tuples. *See also* records; rows
 identified by keys, 61, 150
 other terms for, 148
 relations as collections of, 59
 Turing, Alan (computer expert), 51
 Turing-complete languages, 50, 51
 02 SQLSTATE class value, 384, 386
 2NF (Second Normal Form), 152, 153–154
 two-phase locking, 360
 two-tier driver system (ODBC), 481
 types. *See* data types; UDTs (user-
 defined types)
-
- ## U
-
- UDTs (user-defined types)
 constructors for, 107–108
 defined, 637
 distinct, mapping to XML, 502–503
 distinct types, 106–107

- UDTs (user-defined types) (*continued*)
 - for matching host language types, 106
 - overview, 106
 - privileges for, 375, 378
 - structured types, 107–109, 636
- _ (underscore)
 - as LIKE wildcard, 234
 - using literally in LIKE predicate, 234–235
- UNDO action, 387
- Unicode, 492, 499
- UNION ALL CORRESPONDING operations, 300
- UNION ALL operations, 299
- UNION CORRESPONDING operations, 300
- UNION operations
 - DISTINCT keyword not needed with, 299
 - modeled on relational algebra, 297
 - overview, 297–299
 - preserving duplicate rows, 299
 - for tables not union-compatible, 300
 - union-compatible tables, 297
 - wildcard issues with, 299
- UNIQUE constraint
 - for entity integrity, 171
 - overview, 112
- unique identifiers
 - composite, 142
 - defined, 31, 141
 - single-attribute, 142
- UNIQUE keyword (MATCH), 244–245
- UNIQUE predicate (WHERE)
 - overview, 240
 - referential integrity protected by, 243
- universal quantifier (ALL predicate of WHERE)
 - overview, 238–239
 - for quantified comparison operators, 272, 275, 276–277
 - sylogisms illustrating, 236–237
- Universal Time (UTC), 102
- Universal Turing Machine, 51
- update anomalies, 636
- UPDATE statements
 - with cursors, 324
 - granting privileges for, 376, 473
 - for merging categories, 83–84
 - need for, 81–82
 - for renaming a category, 83
 - for renaming all categories, 84
 - SET clause, 82
 - subqueries in, 282–284
 - syntax, 82
 - updating a row, 324
 - for updating data, 82–83
 - for views, problems with, 85–86
 - WHERE clause, 82, 282–284
- updategrams (SQL Server)
 - creating with XDR schema, 528–529
 - creating with XSD schema, 527–528
 - deleting record using, 526
 - keywords, 523, 524
 - namespace, 523
 - overview, 523
 - template example, 524
 - using nillable mapping schema, 534
- UPDATEXML function (Oracle), 522–523
- updating data structure
 - DBMS advantages, 12
 - flat file issues, 11
- updating XML documents
 - Oracle functions for, 518–523
 - overview, 517–518
 - SQL Server tools for, 523–534
- upgrading platforms, 329–330
- UPPER function, 211
- uppercase, converting strings to, 211
- user interface
 - connecting to database, 445–447
 - designing, 441, 444–445
- user-defined types. *See* UDTs

users
 building rapport with, 137
 classifying, 373–374
 defined, 65, 121
 direct interaction, avoiding, 593
 identifiers for, 372
 importance to database systems,
 63, 136–137
 interviewing, 27–28, 136–137
 overview, 121
 roles for, 372–373
 separating interactions from
 transactions, 562
 special variables for logging, 206
 for testing, 456
 users' data model. *See also* E-R (Entity-
 Relationship) model
 capturing, 27–29
 constructing, 124, 139
 in database design, 128
 interviewing stakeholders, 27–28,
 136–137, 138
 obtaining stakeholder buy-in, 29
 political issues for, 29
 reconciling conflicting requirements,
 28–29
 translating to relational model, 29–47
 vague nature of, 27
 UTC (Universal Time), 102

U

VALID predicate for XML, 513–514
 validity
 determining for XML values, 513–514
 domains for ensuring, 194–195
 valid, defined, 194
 value expressions
 array, 220
 Boolean, 219–220
 conditional, 220–223, 637
 datetime, 218, 637

defined, 637
 interval, 219
 numeric, 217, 637
 row, 225, 635
 string, 218, 511, 637
 value functions. *See also specific kinds*
 datetime, 216–217
 defined, 207, 637
 numeric, 212–216
 string, 209–212
 values. *See also specific kinds*
 identifying in columns, 204
 kinds of, 203
 translating to XML elements, 508–509
 VALUES clause (INSERT), 80
 VARCHAR or CHARACTER VARYING
 type, 99
 variables
 in compound statements, 462
 example, 205–206
 host, 416, 632
 special, for logging users, 206
 for SQLSTATE values, 386
 uses for, 205
 VBA (Visual Basic for Applications),
 399, 404–405
 views or virtual tables
 Cartesian product for, 228–229
 defined, 62–63, 72, 637
 dropping, 78
 fully qualified column names for, 73
 multi-table, 73–77
 privileges applying to, 375
 single-table, 72–73
 updating, problems with, 85–86
 as virtual tables, 62
 viruses
 antivirus software, 338, 340–341
 defined, 337
 options for protection, 338
 overview, 337–338

viruses (*continued*)

signs of infection, 338

worms versus, 338

Visual Basic.NET

challenges using SQL with, 413

embedding SQL in, 418

SQL compared to, 411

volatile memory, 567, 568–569

W

WANs (wide area networks). *See*
networks

Warning! icon, 5

weak entities, 39

WebDAV, Oracle support for, 402

WHENEVER directive, 383, 388–389

WHERE clause (DELETE), subqueries in,
284

where clause (FLWOR), 545

WHERE clause (INSERT), subqueries in,
284–285

WHERE clause (SELECT). *See also*
subqueries

ALL predicate, 236–239, 272, 275,
276–277

AND logical connective, 245–246

ANY predicate, 236–239, 272, 275

BETWEEN predicate, 231–232

comparison predicates with, 229–245

described, 228, 229

DISTINCT predicate, 240–241

for equi-joins, 180, 305, 307

EXISTS predicate, 239–240, 278

for extremal queries, 179

filtering selectively, 613–614

IN predicate, 232–233, 279

LIKE predicate, 234–235

logical connectives with, 245–247

MATCH predicate, 241–245

for multipoint queries, 179

for natural joins, 307

NOT EXISTS predicate, 278–279

NOT IN predicate, 232

NOT LIKE predicate, 234

NOT logical connective, 247

NOT NULL predicate, 236

NULL predicate, 235–236

ON clause with joins versus, 313

OR logical connective, 246, 266–267, 561

OVERLAPS predicate, 241

for point queries, 178–179

for range queries, 179

restricting rows returned, 79–80

SIMILAR predicate, 235

SOME predicate, 236–239, 272, 275

symbols for comparison operators,
230–231

syntax, 229

typical examples, 229

UNIQUE predicate, 240

WHERE clause (UPDATE)

specifying rows updated, 82

subqueries in, 282–284

WHILE...DO...END WHILE statements
(SQL/PSM), 467

wide area networks (WANs). *See*
networks

WIDTH_BUCKET function, 216

wildcards

issues for UNION operations, 299

for LIKE predicate, 234

for SELECT statement, 79, 227–228, 299

using literally with LIKE predicate,
234–235

Windows Disk Defragmenter, 616

WITH ADMIN OPTION clause (GRANT),
381

WITH GRANT OPTION clause (GRANT),
379, 380–381

workgroup databases, 631

workload analysis, 554

workload descriptions, 554

World Trade Center attack, 335–336

World Wide Web Consortium
 DOM developed by, 518
 XML Query Use Cases document, 538

worms
 denial-of-service attacks, 339
 options for protection, 339
 overview, 338
 viruses versus, 338
 zombie spambots, 339–340

write operations, tuning, 572–573
 write-ahead log protocol, 354–355
 write-back disk protocol, 582
 write-through disk protocol, 582
 WWW (World Wide Web), 637

X

XDR schema for updategram, 528–529
 XML (Extensible Markup Language). *See also* XQuery
 adding new node to tree (Oracle), 518–519
 attributes, 494
 characteristics of, 492
 creating tables for data, 517
 data type, 103–104, 497–499
 declaration, 493
 defined, 491, 637
 deleting nodes (Oracle), 521–522
 derivatives of, 491
 determining if value exists, 513
 determining if value is a document, 512–513
 determining if value is an instance, 513
 determining validity of values, 513–514
 document element, 494
 elements, 493–494
 empty elements, 494
 entity references, 495
 flexibility of, 492
 inserting data into SQL pseudo-table, 515–517
 inserting new value at node (Oracle), 519–520
 inserting new value before node (Oracle), 520–521
 mapping schemas (SQL Server), 525–534
 mapping SQL to, 499–508
 nested elements, 494
 numeric character references, 496
 Oracle support for, 402
 overview, 491–492
 root node, 519
 as SGML subset, 491
 SQL compared to, 497
 SQL functions for, 508–512
 SQL predicates for, 512–514
 syntax examples, 492–493
 treelike structure, 519–520
 updategrams (SQL Server), 523–524, 526, 527–529, 534
 updating data in tables, Oracle functions for, 518–523
 updating data in tables, SQL Server tools for, 523–534
 updating documents, 517–518
 updating values (Oracle), 522–523

XML Names, 500
 XML Query Use Cases document, 538
 XML Schema, 103, 496–497
 XML Schema Definition. *See* XSD
 XML schemas
 creating for SQL table, 507–508
 defined, 496
 determining validity of values, 513–514
 mapping ARRAY type to XML, 504–505
 mapping distinct UDT to XML, 502–503
 mapping domain to XML, 501–502
 mapping MULTISET type to XML, 505
 mapping ROW type to XML, 503–504

XML types
 association with XML Schema, 103
 for columns, 103–104
 determining validity of values, 513–514

- XML types (*continued*)
 - further information, 103
 - overview, 103–104, 497–498
 - subtypes, 103
 - uses for, 498–499
 - uses to avoid, 499
 - XMLAGG function, 510
 - XMLCAST function, 512
 - XMLCOMMENT function, 510
 - XMLCONCAT function, 509
 - XMLELEMENT function, 508
 - XML EXISTS predicate, 513
 - XMLFOREST function, 509
 - XMLPARSE function, 511
 - XMLPI function, 511
 - XMLQUERY function, 511–512, 518
 - XMLTABLE pseudo-function, 515–517
 - X/Open standards, DB2 support for, 402
 - XPath expressions
 - with APPENDCHILDXML function (Oracle), 519
 - with DELETXML function (Oracle), 521
 - with INSERTCHILDXML function (Oracle), 519
 - with INSERTXMLBEFORE function (Oracle), 520
 - XQuery
 - casting between XQuery and SQL types, 549
 - data types, SQL correspondences to, 547–549
 - defined, 536
 - development of, 536
 - FLWOR expressions, 542–547
 - functionality, 537
 - further information, 535
 - need for, 535
 - as non-procedural language, 536
 - Oracle support for, 402
 - requirements for implementations, 536–537
 - SQL compared to, 547–549
 - usage scenarios, 538–541
 - XML Query Use Cases document, 538
 - XMLQUERY function for expressions, 511–512, 518
 - XQuery 1.0 Language Specification, 536–537
 - XQuery Requirements document, 537
 - XSD (XML Schema Definition)
 - allowing null values in schemas, 533
 - creating updategram with schema, 527–528
 - defined, 496
 - overview, 496–497
-

Z

- zero (0)
 - minimum cardinality, 146–147
 - null values not same as, 69, 110
 - RAID level, 332, 335, 584
 - SQLSTATE class value (00), 384, 386
 - 01 SQLSTATE class value, 384, 386
 - 02 SQLSTATE class value, 384, 386
- zombie spambots, 339–340