

# Index

## Symbols and Numerics

**\*** (asterisk) in selection criteria, avoiding, 38

**@** (at symbol)

with column names using FOR XML PATH clause, 489–490, 492

variable scope with one versus two symbols, 293

**[ ]** (brackets) for names, 16

**,** (commas) as column placeholders for BCP import, 533

**@@CONNECTIONS** function, 762

**@@CPU\_BUSY** function, 762

**@@CURSOR\_ROWS** function, 762–763

**@@DATEFIRST** function, 763

**@@DBTS** function, 763–764

**"** (double quotes) for names, 16

**=** (equals sign) in comparison operators, 47

**@@ERROR** function

error 547 trapped by, 293–294

error 2714 not trapped by, 295

@Error variable versus, 293

ERROR\_NUMBER() function compared to, 292

INSERT example using, 292–293

overview, 251, 764

saving the value from, 292–293

TRY/CATCH blocks versus, 295

using in sprocs, 293–295

**!** (exclamation mark)

in comparison operators, 47

separating groups in universal table column names, 475

**@@FETCH\_STATUS** function

overview, 251, 764

possible values, 424

simple cursor example, 424

**/** (forward slash) with FOR XML PATH clause, 491–492

**>** (greater than sign) in comparison operators, 47

**@@IDENTITY** function

moving value to local variable, 252

overview, 251, 765

using in scripts, 252

**@@IDLE** function, 765

**@@IO\_BUSY** function, 765

**@@LANGID** function, 765

**@@LANGUAGE** function, 765

**<** (less than sign) in comparison operators, 47

**@@LOCK\_TIMEOUT** function, 765–766

**@@MAX\_CONNECTIONS** function, 766

**@@MAX\_PRECISION** function, 766

@mp:id metaproperty, 495

@mp:localname metaproperty, 495

@mp:namespacerui metaproperty, 495

@mp:parentid metaproperty, 495

@mp:parentlocalname metaproperty, 495

@mp:parentnamespacerui metaproperty, 495

@mp:parentprefix metaproperty, 495

@mp:prefix metaproperty, 495

@mp:prev metaproperty, 495

@mp:xmltext metaproperty, 495

**@@NETLEVEL** function, 766

**@@OPTIONS** function, 766–767

**@@PACKET\_ERRORS** function, 768

**@@PACK\_RECEIVED** function, 767

**@@PACK\_SENT** function, 767

**%** (percent sign)

as modulus operator, 271

as wildcard, 48

**@@PROCID** function, 768

**@@REMSERVER** function, 768

**@@ROWCOUNT** function

EXEC scope example, 264

moving value to local variable, 253

overview, 251, 768

using in scripts, 252–253

**@@SERVERNAME function, 252, 768–769**

**@@SERVICENAME function, 769**

**' (single quotes) within strings, 263**

**@@SPID function, 769**

**[ ] (square brackets) for names, 16**

**@@TEXTSIZE function, 769**

**@@TIMETICKS function, 769**

**@@TOTAL\_ERRORS function, 769**

**@@TOTAL\_READ function, 769**

**@@TOTAL\_WRITE function, 769**

**@@TRANCOUNT function, 252, 770**

**\_ (underscore)**

in table names, avoiding, 83

as wildcard, 48

**@@VERSION function, 770**

**1NF (first normal form), 157**

**2NF (second normal form), 157**

**3NF (third normal form), 155, 157**

**32-level limit for recursion, 306–307**

**547 error**

CHECK constraints for monitoring, 344

from foreign key violations, 291

for inline errors in sprocs, 291, 293–294

trapped by @@ERROR, 293–294

**1205 error. See deadlocks**

**2714 error, 295**

## A

**abbreviations**

for dateparts, 775

in table names, 83

**ABS function, 777**

**accent sensitivity. See also case and case sensitivity**

COLLATE parameter for setting, 80

enabling for full-text catalogs, 612

in SQL Server indexes, 194–195

**Access (Microsoft), 85**

**ACID test for transactions, 359**

**ACOS function, 778**

**ADD parameter (ALTER FULLTEXT INDEX), 618**

**administration tasks. See also maintenance; specific tasks**

archiving data, 736

backup and recovery, 722–733

index maintenance, 733–736

scheduling jobs, 700–722

usefulness for developers, 699

**advanced query design**

casting and converting data types, 148–150

correlated subqueries, 139–144

derived tables, 144–146

EXISTS operator, 146–148

external calls, 150–151

importance for professionals, 134

nested subqueries, 135–139

performance optimization, 151–153

procedural versus set-based thinking for, 133

subqueries, defined, 134

**AdventureWorks sample database**

as core sample database for this book, 4

diagram of, 7, 8

overview, 4

uspLogError sproc, 285–288

viewing columns in Management Studio, 46

**AdventureWorksDW sample database, 5**

**AFTER clause (CREATE TRIGGER), 364**

**aggregate functions. See also specific functions**

Accumulate method, 407, 409–410

analysis supporting, 407

creating from assemblies, 407–412

with GROUP BY clause, 53–56

Init method, 407, 409

Merge method, 407, 410

NULLs ignored by, 56

overview, 54–56

system functions, 771–773

Terminate method, 407, 410

**aggregating data, subcategories for, 171–177**

**aliases**

for columns returned by function call, 55

for computed columns, 87

Configuration Manager Aliases list, 24

in correlated subqueries, 142

sa role as dbo alias, 76

for self-referencing INNER JOINS, 40, 41

for table names, 37

using, 37

**ALL keyword**

ALTER INDEX statement, 216, 736

GRANT statement, 650

**ALL operator, 48, 139**

**ALTER DATABASE statement**

for increasing database size, 91

options, 92

overview, 89–92

setting cursor default to local, 429

syntax, 91

termination options for users, 92

TRUSTWORTHY parameter, 406

viewing existing database settings, 89–90

**ALTER FULLTEXT CATALOG statement, 613–614**

**ALTER FULLTEXT INDEX statement**

ADD parameter, 618

differences to other ALTER statements, 617

DROP parameter, 618

ENABLE/DISABLE parameters, 618

START FULL POPULATION parameter, 618

- START INCREMENTAL POPULATION parameter, 618
- START UPDATE POPULATION parameter, 619
- STOP parameter, 619
- syntax, 617
- ALTER FUNCTION statement, 317**
- ALTER INDEX statement**
  - acting on ALL indexes, 216, 736
  - DISABLE option, 217, 735
  - for index maintenance, 227, 734–736
  - naming the index for, 216, 734
  - naming the table or view for, 216, 734
  - overview, 215, 734
  - REBUILD option, 216–217, 425–427, 734–735
  - REORGANIZE option, 217, 735–736
  - syntax, 215–216, 734
  - TableCursor for rebuilding all indexes in database, 423–427
- ALTER LOGIN statement, 642–643**
- ALTER PROC statement**
  - accepting a parameter, 284–285
  - CREATE PROCEDURE compared to, 283
  - overview, 283
  - with RETURN statement, 290
  - for spEmployee sproc, 284–285
- ALTER statement, 89. See also specific forms**
- ALTER TABLE statement**
  - adding CHECK constraints, 119
  - adding DEFAULT constraints, 121
  - adding foreign key, 110
  - adding primary key, 108
  - adding UNIQUE constraints, 118
  - batch for establishing precedence, 258
  - enabling/disable triggers, 378
  - example using, 94
  - NOCHECK option for constraints, 124–126
  - overview, 92–94
  - for self-referencing foreign keys, 111
  - syntax, 92–93
  - viewing existing table settings, 93
- ALTER TRIGGER statement, 363**
- ALTER VIEW statement**
  - CREATE VIEW compared to, 237
  - permissions retained by, 237
  - for rebuilding views to included added columns, 94
  - WITH ENCRYPTION option, 240
  - WITH SCHEMABINDING option, 242–243
- ALTER XML SCHEMA COLLECTION statement, 462–463**
- Analysis Services**
  - administrative considerations, 838
  - building a project, process of, 834
  - Configuration Manager for service management, 19
  - cubes, 829–830, 835–837
  - data mining, 834, 838
  - data warehouses, 831–833
  - described, 825
  - MDX with, 838
  - OLAP versus OLTP, 826–829
  - storage types, 830–831
  - uses for, 834
- Analysis Services Tasks (SSIS), 549**
- AND Boolean operator**
  - with FTS, 629
  - overview, 48
- ANSI SQL 92 standard**
  - CAST versus CONVERT and compliance with, 149
  - portability insured by, 36
  - “sys” functions not compliant with, 37
  - T-SQL entry-level compliance, 27, 36
- ANSI\_NULLS option**
  - not allowed for indexed views, 242
  - setting to OFF (recommended), 266
- ANY operator, 48, 139**
- application roles**
  - adding permissions to, 662
  - creating, 662
  - described, 656, 661
  - dropping, 663–664
  - other roles versus, 656, 661
  - process of, 661–662
  - uses for, 661
  - using, 662–663
- applications, partition-aware, 181**
- APP\_NAME function, 804**
- archiving data, 736**
- AS DEFAULT parameter (CREATE FULLTEXT CATALOG), 612**
- AS keyword**
  - CREATE TRIGGER statement, 367
  - GRANT statement, 650
- ASC sort order options (CREATE INDEX), 209–210**
- ASCII function, 799**
- ASIN function, 778**
- assemblies. See .NET assemblies**
- asterisk (\*) in selection criteria, avoiding, 38**
- asymmetric keys**
  - certificates versus, 667
  - CREATE LOGIN . . . FROM option, 642
  - CREATE USER option, 648
  - defined, 642
  - limiting access to, 666
  - overview, 666
- at symbol (@)**
  - with column names using FOR XML PATH clause, 489–490, 492
  - variable scope with one versus two symbols, 293
- ATAN function, 778**
- ATN2 function, 778**

## **atomicity**

- defined, 157
- first normal form for guaranteeing, 157
- of transactions, 329–330, 359

## **attributes**

- of documents, 171
- in entity boxes, 161

## **audit trails**

- defined, 362
- triggers for, 362

## **authentication**

- SQL Server Authentication, 26, 27, 639–646
- Windows authentication, 26–27, 639, 646–647

## **AUTHORIZATION parameter**

- CREATE ASSEMBLY statement, 398
- CREATE FULLTEXT CATALOG statement, 612

## **AUTO option (FOR XML clause), 470, 473–474**

## **autonomy in replication**

- conflict management trade-off with, 570
- defined, 566
- merge replication, 574
- overview, 566–567
- snapshot replication, 571

## **autonomy of batches, 255**

## **AVG function, 54, 771**

# B

## **backing up. See also recovery**

- backup and restore for creating database copies, 302
- backup set for, 724–725
- creating a device using T-SQL, 725
- database using SMO, 750–751
- DCM information for, 193
- destination for, 725
- differential backup, 724
- early and often, 722
- full backup, 724
- including mechanism in applications, 722
- Management Studio for, 722–725
- more important than RAID for data safety, 683
- scheduling, 725
- transaction log, 724, 728
- T-SQL for, 726–728

## **BACKUP DATABASE statement**

- example using, 727
- parameters, 726–727
- syntax, 726
- user rights for, 654–655

## **BACKUP LOG statement, 654–655**

## **Balanced Trees. See B-Trees**

## **bandwidth, 818–819**

## **batches. See also scripts**

- autonomous nature of, 255
- defined, 247, 253

establishing precedence with, 256–258

- purpose of, 256
- runtime errors in, 253, 255
- sent to server separately, 254–255
- separating scripts into, 253–254
- statements requiring their own batch, 256
- syntax (parse-time) errors in, 253, 255

## **BCM (Bulk Changed Map), 193**

## **BCP (Bulk Copy Program)**

- advantages of, 525–526
- case-sensitivity of switches, 526
- command for importing data to remote system, 533
- commands for importing with format file, 536, 537
- commas in source file as column placeholders, 533
- data import example, 531–533
- for exporting bulk data, 534–535
- fast versus slow mode, 534
- format files for, 531, 536–540
- for importing bulk data, 531–534
- interactive mode, 530
- logged versus nonlogged mode, 534
- overview, 33, 525–526, 543–544
- parallel loads with, 541
- source files usable for export, 535
- source files usable for import, 531
- switches, 527–530
- syntax, 526
- TABLOCK hint with, 534, 541
- transaction log size issues, 534

## **BCP .fmt file, 536**

## **BEGIN TRAN statement**

- described, 331
- example using, 332, 333–334
- missing, implicit transactions with, 340–341
- overview, 331
- syntax, 331
- @@TRANCOUNT incremented by 1 with, 252

## **BEGIN . . . END blocks**

- with IF . . . ELSE statement, 268–270
- with WHILE statement, 276

## **Beginning SQL Server 2005 Programming (Vieira), 17, 25, 44, 329, 457**

## **Beginning XML (Hunter), 458**

## **BETWEEN operator, 48**

## **bigint data type**

- for identity columns, 84
- overview, 11

## **binary data type, 13**

## **binary order for indexes, 194**

## **binding**

- defaults, 128
- rules, 127

## **bit data type, 11**

**BLOBs (Binary Large Objects)**

- backward compatibility issues for, 168
- data types for, 168
- Full-Text Search against, 170
- pages for, 192
- performance issues for, 168, 170
- row size limit extended by, 194
- using file storage instead of, 168–170

**blocking, 177. See also locking****BOL (Books Online), 18–19****Boolean operators**

- AND, 48, 629
- with FTS, 629
- NOT, 48, 629
- OR, 48, 629
- overview, 48
- in searched CASE statements, 270–271, 272–275
- table summarizing, 48
- unknown results returned as FALSE, 268

**bound connections, for avoiding deadlocks, 358****Boyce-Codd normal form, 158****brackets ( [ ] ) for names, 16****BREAK statement**

- controversy about, 276
- with WHILE statement, 276

**B-Trees (Balanced Trees)**

- defined, 195
- leaf-level nodes, 196, 199, 200, 201
- navigation by clustered indexes, 201–203
- non-leaf level nodes, 196–197, 200
- page splits with, 197–199
- root node, 195–196, 198
- as self-balancing, 197

**BU (bulk update) locks, 349****Bulk Changed Map (BCM), 193****Bulk Copy Program. See BCP****BULK INSERT command, 541–542, 657****Bulk Insert Task (SSIS), 549, 558–559****bulkadmin role, 657****Bulk-Logged recovery model, 728****Business Intelligence Studio**

- Connection Manager, 506–507
- Data Source Wizard, 505, 507
- deploying report models, 512
- Management Studio versus, 504
- opening SSIS from, 546
- overview, 32
- Report Model Wizard, 509–511
- Report Server Projects, 517–523
- Report Wizard, 518–519
- setting up Visual Studio for, 505
- starting a Report Model Project, 504
- starting a Report Server Project, 517

**C****calling**

- aliases for columns returned by functions, 55
- EXEC command, concatenating strings before, 262, 265
- external calls, 150–151
- GetFiles method, 405
- recursion, 305–307
- sprocs, assembly-based, 399–400
- sprocs, EXEC keyword for, 288, 399
- sprocs using OUTPUT keyword, 287
- sprocs using WITH RECOMPILE option, 304
- sprocs within sprocs, 301
- top-level function for ExampleTVF assembly, 404
- TRY/CATCH example with uspLogError, 286–288

**Callstack window (Debugger)**

- overview, 322
- using, 323–327

**cardinality, 162****cascading**

- defined, 112
- deletes, 112–116
- ON clause of foreign key for, 112–113
- SET NULL and SET DEFAULT, 112, 116
- updates, 112

**case and case sensitivity. See also accent sensitivity**

- BCP switches, 526
- COLLATE parameter for setting, 80
- index collation, 194–195
- SQLCMD flags, 259
- table naming standards, 83

**CASE statement**

- overview, 804–805
- searched, 270–271, 272–275, 805
- similar statements in other languages, 266, 270
- simple, 270, 271–272, 805
- syntax with Boolean expression (searched CASE), 270–271
- syntax with input expression (simple CASE), 270
- for trigger event types, 413–414
- using inline with SELECT, 271–275

**CAST function**

- CONVERT versus, 148–149
- converting dates, 150
- converting number to string, 149
- in EXEC procedures, 265
- overview, 805
- syntax, 149

**catalogs for FTS. See full-text catalogs****CEILING function, 778****certificates**

- asymmetric keys versus, 667
- certificate authority (CA) with SQL Server, 667
- CREATE LOGIN...FROM option, 642

## certificates (continued)

- CREATE USER option, 648
- described, 642
- limiting access to, 666
- char data type, 12**
- CHAR function, 799**
- CHARINDEX function, 799–800**
- CHECK constraints**
  - adding to existing table, 119
  - adding with diagramming tools, 187
  - applying assembly-based UDF as, 403
  - BCP with, 531
  - as column constraint, 88
  - criteria examples for, 118
  - deleting with diagramming tools, 187
  - disabling on replication subscribers, 588
  - as domain constraints, 103
  - editing with diagramming tools, 187
  - in logical model, 167
  - monitoring for error 547, 344
  - naming, 5
  - overview, 117–118
  - preventing negative numbers, 13
  - for preventing non-repeatable reads, 344
  - rules versus, 126
  - specifying in CREATE TABLE statement, 87
  - system-generated names for, avoiding, 5
  - as table constraint, 88
  - triggers versus, 362, 367–371
  - violating, example of, 119
- checkpoints**
  - at change of database options, 338
  - Checkpoint on Recovery option, 337–338
  - circumstances issuing, 337
  - defined, 5, 336
  - failure and recovery, 339–340
  - issuing manually with CHECKPOINT command, 337
  - at normal server shutdown, 338
  - Truncate On Checkpoint option, 338
  - when recovery interval option is exceeded, 339
- client statistics, 692**
- client-side cursors**
  - advantages of, 435
  - views supporting, 241
- client-side versus server-side processing, 672–673**
- closing**
  - connections, 816–817
  - cursors, 424, 430–431
- CLR (Common Language Runtime)**
  - for external calls, 151
  - file storage implementation using, 170
  - required for .NET assemblies, 394
  - T-SQL compliance with, 35
- cluster key, 201**

## clustered indexes

- advantages of, 220
- changing columns for, 219–221
- choosing a column for, 220, 221
- defined, 6
- disadvantages of, 220–221
- for indexed views, 243
- indicated by sp\_helpconstraint, 109
- leaf level as actual data, 201
- navigating the B-Tree, 201–203
- one per table allowed, 6
- overview, 201–203
- page split exceptions with, 193, 220–221
- performance benefits of, 203
- primary key as default for, 219, 220
- for ranged queries, 220–221
- selectivity within, 218–219
- as unique, 201
- clustered tables**
  - cluster key for, 201
  - defined, 201
  - non-clustered indexes on, 205–208
- COALESCE function, 805**
- COLLATE parameter**
  - CREATE DATABASE statement, 80
  - CREATE TABLE statement, 86
- COLLATIONPROPERTY function, 805–806**
- COL\_LENGTH function, 782**
- COL\_NAME function, 782–783**
- COLUMNPROPERTY function, 783**
- columns. See also identity columns**
  - available for ORDER BY clause, 50
  - changing for clustered indexes, 219–221
  - choosing for clustered indexes, 220, 221
  - COLUMNS\_UPDATED() function for checking trigger outcomes, 386–388
  - computed, 87
  - constraints, specifying in CREATE TABLE statement, 87
  - defining as XML data type, 458–459
  - as domain data, 5
  - editable by cursors, specifying, 452
  - listing explicitly with INSERT statement, 59
  - maximum per row, 193
  - moving using Management Studio, 99
  - moving using T-SQL, problems with, 94
  - order in indexes, 222
  - returned by function call, name lacking for, 55
  - UPDATE() function for checking trigger outcomes, 386
- COLUMNS\_UPDATED() function, 386–388**
- command object, 816**
- command-line console. See SQLCMD**
- commas (,) as column placeholders for BCP import, 533**

**COMMIT TRAN statement**

- example using, 333, 335
- overview, 331
- syntax, 331
- @@TRANCOUNT decremented by 1 with, 252

**Common Language Runtime. See CLR****comparison operators**

- ALL, 48, 139
- ANY, 48, 139
- BETWEEN, 48
- IN, 48
- LIKE, 48, 607
- SOME, 48, 139
- standard, 47, 138, 139, 266
- tests against NULLs, 138, 266

**compiling assemblies**

- ComplexNumber, 417
- ExampleAggregate, 410
- ExampleProc, 397–398
- ExampleTrigger, 414
- ExampleTVF, 406
- ExampleUDF, 402
- overview, 394

**ComplexNumber assembly, 417–418****computed columns, 87, 531****concatenation before calling EXEC, 262, 265****concurrency. See also isolation levels; locking**

- cursor options for, 447–450
- defined, 341
- not an issue between triggers and process firing, 389
- in OLAP environment, 342
- in OLTP environment, 342
- trade-off with consistency, 355

**Configuration Manager**

- Aliases list, 24
- client configuration, 23–24
- main areas of, 19
- NetLibs configuration, 20–23
- Network Configuration area, 19–24
- protocols in, 21
- Service Management area, 19

**Configure Distribution Wizard, 589–592, 593****Connection Manager, 506–507****connections**

- adding for Debugger, 318–319
- bandwidth decisions, 818–819
- bound, for avoiding deadlocks, 358
- C# example, 819–821
- closing properly, 816–817
- command object, defined, 816
- connection context of EXEC, 262
- connection object, defined, 815
- data set, defined, 816
- dependent data issues, 818
- hierarchy issues, 817–818

managing, 816–817

MARS for, 817

for .NET assemblies, 396

obtaining context for trigger, 412–413

performance considerations, 816–819

pooled, 816, 817

for replication, 568, 570, 581

for report models, 506–507

settings, caveats for, 817

sharing within processes, 816

for SMO, 745

for SSIS packages, 556–557, 559

VB.NET example, 821–823

**@@CONNECTIONS function, 762****consistency**

naming standards for, 84, 105

replication concerns, 567–568

for schema names, 74

SERIALIZABLE isolation level for, 355

trade-off with concurrency, 355

of transactions, 359

**constraints. See also specific kinds**

BCP with, 531

defined, 6, 87, 101

disabling, reasons for, 121

disabling temporarily, 124–126

disabling, to ignore existing bad data, 122–124

domain, 103

enforcing for XML, 458, 469

entity, 103–104

implied indexes created with, 215

key constraints, 105–118

in logical model, 167

as metadata, 10

mismatched for BCP import, format files for, 531, 536–540

with mutually exclusive conditions, avoiding, 121

naming, 4–5

NOCHECK option, 124–126

other data integrity methods compared to, 129–131

referential integrity, 104

rules versus, 10

specifying in CREATE TABLE statement, 87–88

sp\_helpconstraint for viewing information on, 109

system-generated names for, avoiding, 4–5

types of, 102–104

WITH NOCHECK option, 122–124

**CONTAINS statement, 624–625, 628****CONTAINSTABLE function, 626–627, 794****CONTINUE statement with WHILE statement, 276****control-of-flow statements. See also specific statements**

CASE, 266, 270–275, 413–414, 804–805

IF . . . ELSE, 266–270

TRY/CATCH blocks, 277–280

T-SQL choices for, 266

### control-of-flow statements (continued)

- WAITFOR, 277
- WHILE, 275–277
- conversion of cursors to other types**
  - from FAST\_FORWARD cursor, 445, 446
  - warning for, 446, 450–452
- CONVERT function**
  - CAST versus, 148–149
  - converting dates, 150
  - date formatting codes, 150
  - overview, 805
  - split point for two-digit to four-digit year conversion, 150
  - syntax, 149
- converting data types**
  - CAST function for, 148–150, 805
  - CONVERT function for, 148–150, 805
  - table summarizing, 14
- correlated subqueries**
  - aliases in, 142
  - importance of understanding, 139
  - ISNULL () function with, 143–144
  - nested subqueries versus, 140
  - overview, 140
  - in select list, 142–144
  - two-query approach versus, 140–141, 142
  - in WHERE clause, 140–142
- COS function, 778–779**
- COT function, 779**
- COUNT function**
  - DISTINCT clause with, 57
  - expressions with, 55–56
  - overview, 54–56, 771–772
- COUNT\_BIG function, 772**
- covered queries, 210**
- CPU intensive tasks**
  - CPU time statistics, 692
  - increasing memory for, 683–684
  - I/O intensive tasks versus, 679–680
  - multiprocessors for, 683
  - query types, 680
- @@CPU\_BUSY function, 762**
- CREATE ASSEMBLY statement**
  - for aggregate functions, 410
  - assemblies uploaded by, 397, 398
  - AUTHORIZATION parameter, 398
  - for custom data types, 417
  - FROM clause, 398
  - for scalar UDFs, 402
  - for sprocs, 398
  - syntax, 397
  - for table-valued functions, 406
  - for triggers, 414
  - WITH PERMISSION\_SET options, 398

- CREATE authority, 75**
- CREATE DATABASE statement**
  - basic syntax, 77
  - batch for establishing precedence, 256–258
  - COLLATE parameter, 80
  - example using, 80–82
  - FILEGROWTH parameter, 79
  - FILENAME parameter, 78
  - FOR ATTACH parameter, 80
  - full syntax, 77
  - LOG ON parameter, 79
  - MAXSIZE parameter, 79
  - model database as default for, 77, 78
  - NAME parameter, 78
  - ON clause, 78
  - overview, 77–82
  - PRIMARY keyword with, 78
  - SIZE parameter, 78
  - testing database existence before using, 148
  - TRUSTWORTHY parameter, 80
  - user rights for, 654–655
  - WITH DB CHAINING parameter, 80
- CREATE DEFAULT statement, 128, 654–655**
- CREATE FULLTEXT CATALOG statement**
  - AS DEFAULT parameter, 612
  - AUTHORIZATION parameter, 612
  - example using, 613
  - IN PATH parameter, 612
  - ON FILEGROUP parameter, 612
  - syntax, 611
  - WITH ACCENT\_SENSITIVITY parameter, 612
- CREATE FULLTEXT INDEX statement**
  - column list with, 615
  - example using, 614–615
  - KEY INDEX parameter, 616
  - LANGUAGE parameter, 615
  - ON parameter, 616
  - syntax, 614
  - TYPE COLUMN parameter, 615–616
  - WITH CHANGE\_TRACKING parameter, 616–617
  - WITH NO POPULATION parameter, 616–617
- CREATE FUNCTION statement**
  - for assembly-based scalar UDFs, 400, 402
  - for DayOnly () UDF, 310
  - for fnContactList () UDF, 311
  - for fnContactSearch () UDF, 311
  - for fnGetReports () UDF, 313–314
  - syntax, 308, 400
- CREATE INDEX statement**
  - ALLOW\_PAGE\_LOCKS option, 213
  - ALLOW\_ROW\_LOCKS option, 213
  - ASC/DESC sort order options, 209–210
  - DROP\_EXISTING option, 211–214
  - FILLFACTOR option, 211

- IGNORE\_DUP\_KEY option, 211
- INCLUDE columns option, 210
- for indexed views, 243
- legacy syntax, 209
- MAXDOP option, 213
- ON clause, 209, 214
- ONLINE option, 213
- PAD\_INDEX option, 210
- SORT\_IN\_TEMPDB option, 212–213
- STATISTICS\_NORECOMPUTE option, 212
- syntax, 208–209
- WITH keyword, 210
- CREATE LOGIN statement**
  - FROM clause, 641–642
  - sp\_addlogin deprecated by, 640
  - syntax, 640
  - WITH clause, 640–641
- CREATE PROCEDURE statement**
  - ALTER PROC compared to, 283
  - for assembly-based sprocs, 398–399
  - basic example, 282–283
  - with RETURN statement, 288–289
  - for spEmployee sproc, 282–283
  - for spFactorial sproc, 305–306
  - for spInsertValidatedStoreContact sproc, 293–294
  - for spTestReturns sproc, 289–290
  - for spTriangular sproc, 306–307
  - syntax, 282
  - user rights for, 654–655
- CREATE RULE statement, 126–127, 654–655**
- CREATE statement. See also specific forms**
  - basic syntax, 77
  - limited to local server, 77
  - testing object existence before using, 147–148
- CREATE TABLE statement**
  - basic syntax, 82
  - for cascading actions, 112–113
  - COLLATE parameter, 86
  - column constraints option, 87
  - computed columns option, 87
  - data types specification, 84
  - for DEFAULT constraints, 84, 120–121
  - defining column as XML data type, 459
  - example using, 88–89
  - with foreign key, 109
  - full syntax, 82
  - IDENTITY parameter, 84–85
  - model database as default for, 82
  - NONCLUSTERED option, 219
  - NOT FOR REPLICATION parameter, 85
  - NULL/NOT NULL parameter, 86
  - ON clause, 88, 112–113
  - overview, 82–89
  - with primary key, 87, 107
  - ROWGUIDCOL parameter, 85–86
  - script for creating if not already existing, 267, 268, 269–270
  - for self-referencing foreign keys, 111–112
  - table constraints option, 87–88
  - testing table existence before using, 147–148
  - TEXTIMAGE\_ON clause, 88
  - for UNIQUE constraints, 117–118
  - user rights for, 654–655
- CREATE TRIGGER statement**
  - AFTER clause, 364
  - AS keyword, 367
  - DELETE parameter, 366
  - FOR clause, 364
  - INSERT parameter, 366
  - INSTEAD OF clause, 364
  - NOT FOR REPLICATION parameter, 367
  - ON clause, 363
  - syntax, 363
  - UPDATE parameter, 366
  - WITH APPEND parameter, 366
  - WITH ENCRYPTION option, 363
- CREATE TYPE statement, 418**
- CREATE USER statement, 647–648**
- CREATE VIEW statement**
  - ALTER VIEW compared to, 237
  - basic syntax, 232
  - extended syntax, 232
  - simple example, 232
  - user rights for, 654–655
  - WITH SCHEMABINDING option, 241
- CREATE XML SCHEMA COLLECTION statement, 461–462**
- CreateImportText.vbs file, 553**
- CROSS JOINS**
  - alternative (legacy) syntax for, 68
  - as Cartesian product, 44
  - overview, 44–45
  - uses for, 45
- C# connectivity example, 819–821**
- cubes, 829–830, 835–837**
- CURRENT\_TIMESTAMP function, 806**
- CURRENT\_USER function, 806**
- cursor data type**
  - declaring for sproc parameter, 284
  - overview, 12
- @@CURSOR\_ROWS function, 762–763**
- cursors. See also DECLARE statement for cursors; FETCH statement; specific kinds**
  - altering data within, 453–456
  - avoiding when possible, 133, 456, 676–677
  - client-side, advantages of, 435, 673
  - client-side, views supporting, 241

### **cursors (continued)**

- closing, 424, 430–431
- concurrency options, 447–450
- CURSOR\_STATUS function, 773–774
- deallocating, 424, 430, 431
- declaring, 423
- default scope, 428, 429
- default type, 427
- described, 421
- detecting conversion of types, 450–452
- dynamic, 442–445
  - extended syntax, 427
  - FAST\_FORWARD, 445–446
  - @FETCH\_STATUS variable for, 424, 425
  - forward-only, 432
  - keyset-driven, 438–441
  - lifespan, 422–427
  - LOCAL versus GLOBAL scope, 428–432
  - object libraries for creating, 422
  - opening, 423–424
  - performance considerations, 434
  - for rebuilding all indexes in database, 423–427
  - result set features of, 422
  - scrollability of, 432–434
  - SELECT statements valid for, 452
  - sensitivity, 435, 441, 442, 445
  - simple example, 424–425
  - specifying editable columns for, 452
  - sproc for creating, 428–432
  - static, 435–437
  - stating options explicitly, 429
  - types of, 434–435
  - utilizing and navigating, 424
  - warning for conversion to other types, 446, 450–452

### **CURSOR\_STATUS function, 773–774**

#### **CursorTest cursor**

- dynamic cursor example, 443–444
- global scope example, 429, 430
- keyset-driven cursor example, 438–440
- local scope example, 431
- scrollable cursor example, 433
- SCROLL\_LOCKS example, 448–449
- static cursor example, 436–437
- TYPE\_WARNING example, 451

**custom data types. See user-defined data types**

## **D**

### **data access. See also user rights**

- direct, for users, 302
- impersonation contexts for, 80
- indexes for, 200
- metadata access paths, 37
- ownership for, 77
- table scans for, 199

### **data convergence, 567**

#### **Data Flow Task (SSIS), 549**

#### **data integrity. See also constraints**

- checking the delta of an update, 369–371
- choosing mechanisms for, 129–131
- custom error messages, 371
- as database responsibility, 101–102
- DRI (declarative referential integrity), 362, 367–369
- referential integrity, 104, 362
- for requirements sourced from other tables, 367–369
- triggers for implementing, 129, 367–371

#### **Data Manipulation Language (DML). See also T-SQL (Transact-SQL)**

- basic statements, 35
- triggers, 362

#### **data marts, 833**

#### **data mining, 834, 838**

#### **Data Mining Query Task (SSIS), 549**

#### **data pages, 192**

#### **data sets**

- defined, 816
- returning using C#, 819–820
- returning using VB.NET, 821–822

#### **Data Source View Wizard, 508**

#### **Data Source Views, 507–508**

#### **Data Source Wizard, 505, 507**

#### **Data Transformation Services (DTS), 31, 545, 546.**

**See also SSIS (SQL Server Integration Services)**

#### **data types**

- for columns in UNIONS, compatibility required, 70
- conversions (table summarizing), 14
- CREATE TABLE statement specification of, 84
- custom, creating from assemblies, 417–419
- declaring for variables, 249
- in entity boxes, 161
- equivalents in other programming languages, 13
- for identity columns, 84
- in logical model, 167
- overview, 13–14
- for sproc parameters, 284
- table summarizing, 11–13
- user-defined, 10, 417–419, 587
- using rule or default, determining, 129

#### **data warehouses, 831–833**

#### **Database Consistency Checker (DBCC)**

- DBCC DBREINDEX command, 227–228, 733
- DBCC SHOWCONTIG command, 224–227, 228
- “sys” functions versus, 224
- for troubleshooting performance, 692

#### **Database Engine Tuning Advisor, 222, 672**

#### **Database Master Key, 666**

#### **database object**

- attaching to current server when creating, 80
- creating using Management Studio, 96–97
- creating using SMO, 745–746

- current as default, 76
- database-qualified names, 76
- dropping, 95, 750
- editing using Management Studio, 97
- enabling full-text indexing for, 610–611
- files, 190
- getting structure information for, 81–82
- granting user access to, 647–648
- increasing size explicitly, 91
- locking, 346
- other objects as children of, 2
- overview, 2–5
- sample databases, 3, 4–5
- specifying current database, 37
- in storage hierarchy, 189
- system databases, 3–4
- testing existence before creating, 148
- database owner (dbo), 75, 76**
- database roles. See also specific roles**
  - fixed, 656, 658–659
  - limited scope of, 658
  - user-defined, 656, 659–661
- DATABASEPROPERTY function, 783–784**
- DATABASEPROPERTYEX function, 610, 785**
- DATALLENGTH function, 806**
- DATEADD function, 775**
- DATEDIFF function, 775**
- @@DATEFIRST function, 763**
- DATENAME function, 775**
- DATEPART function, 776**
- dates**
  - converting with CAST, 150
  - converting with CONVERT, 150
  - dateparts and abbreviations, 775
  - formatting for Report Server Projects, 520–521
  - system functions, 774–777
- datetime data type**
  - overview, 12
  - UDF for, 309–310
- DAY function, 776**
- DayOnly() UDF, 310, 316–317**
- db\_accessadmin role, 658**
- db\_backupoperator role, 658**
- DBCC (Database Consistency Checker)**
  - DBCC DBREINDEX command, 227–228, 733
  - DBCC SHOWCONTIG command, 224–227, 228
  - “sys” functions versus, 224
  - for troubleshooting performance, 692
- DBCC SHOWCONTIG command**
  - for checking DBCC DBREINDEX command effects, 228
  - example using, 225
  - OLAP versus OLTP systems and results desired, 226–227
  - output from, 225–226
  - syntax, 224–225
- dbcreator role, 657**
- db\_datareader role, 658**
- db\_datawriter role, 658**
- db\_ddladmin role, 75, 658**
- db\_denydatareader role, 658**
- db\_denydatawriter role, 658**
- DB\_ID function, 785**
- DB\_NAME function, 785**
- dbo (database owner), 75, 76**
- db\_owner role**
  - CREATE authority as default for, 75
  - overview, 658
  - ownership of objects created by, 76
- db\_securityadmin role, 658**
- @@DBTS function, 763–764**
- DCM (Differential Changed Map), 193**
- DDL triggers, 362. See also triggers**
- deadlocks**
  - defined, 347–348, 355
  - determination by server, 356
  - error 1205 for, 355
  - prevented by update locks, 347–348
  - rules for avoiding, 356–358
  - with subcategories, 177
  - victim, 355, 356
- deallocating cursors, 424, 430, 431**
- Debugger**
  - Callstack window, 322
  - challenges for remote debugging, 317
  - icons at top, 321
  - Locals window, 321–322
  - Output window, 322
  - setting up SQL Server for, 318
  - starting, 318–321
  - for triggers, 376–377, 390–392
  - using, 322–327
  - Watch window, 322
  - yellow arrow on left, 321
- decimal data type, 11**
- declarative referential integrity (DRI), 362, 367–369**
- DECLARE statement for cursors**
  - basic syntax, 423
  - DYNAMIC option, 442–445
  - extended syntax, 427
  - FAST\_FORWARD option, 445–446
  - FOR clause, 452
  - FOR UPDATE option, 452
  - FORWARD\_ONLY option, 432
  - GLOBAL option, 428–431
  - KEYSET option, 438–441
  - LOCAL option, 428, 431–432
  - OPTIMISTIC option, 450
  - READ\_ONLY option, 447
  - SCROLL option, 432–434
  - SCROLL\_LOCKS option, 447–450

## DECLARE statement for cursors (continued)

---

### **DECLARE statement for cursors (continued)**

STATIC option, 435–437  
TableCursor example, 423, 424–425, 426  
TYPE\_WARNING option, 446, 450–452

### **DECLARE statement in scripts, 249**

#### **declaring**

cursors, 423  
data types for variables, 249  
member variables in ExampleAggregate assembly, 408, 409  
parameters for sprocs, 284–285  
SMO, 745  
variables in scripts, 62, 249

#### **DEFAULT constraints**

adding to existing table, 121  
cascading actions for setting, 116  
defaults compared to, 128  
as domain constraints, 103  
ignored by DELETE statement, 120  
overview, 119–121  
specifying in CREATE TABLE statement, 84, 120–121  
specifying in INSERT statement, 60  
UPDATE statement with, 120  
values for, 119

#### **defaults. See also model database**

binding and unbinding, 128  
creating, 128  
DEFAULT constraints compared to, 128  
as domain constraints, 103  
dropping, 128  
as legacy items, 10, 126  
other data integrity methods compared to, 129–131  
overview, 128  
types of, 10  
viewing dependencies for, 129

#### **deferred name resolution, 129**

#### **DEGREES function, 779**

#### **DELAY parameter (WAITFOR), 277**

#### **DELETE statement**

cascading deletes, 112–116  
DEFAULT constraints ignored by, 120  
overview, 64–66  
syntax, 64  
two FROM clauses for, 64  
user rights for, 649  
with views, 236–237  
WHERE clause with, 65

#### **DELETE triggers**

checking the delta of an update, 369–371  
CREATE TRIGGER parameter for, 366  
creating from assemblies, 412–417  
firing order for, 378  
INSTEAD OF triggers, 384–385  
for setting condition flags, 373–375

#### **DELETED tables for triggers, 366, 370**

#### **deleting. See also DELETE statement; DROP statement**

application roles, 663–664  
cascading deletes, 112–116  
custom error messages, 301  
database using SMO, 750  
existing index when creating new one of same name, 211–212  
model database, avoiding, 4  
tables from database, 184  
tables from diagram, 184  
user-defined database roles, 661  
users from user-defined database roles, 660

#### **de-normalization. See also normalization**

situations benefiting from, 179–180  
strategic, 674  
triggers for feeding data into reporting tables, 372–373

#### **DENY statement, 652–653**

#### **dependencies**

connectivity considerations, 818  
with deferred name resolution, 129  
normal forms for handling, 157–158  
retained by ALTER PROC statement, 283  
viewing with sp\_depends, 129

#### **dependency chains, 114**

#### **deployment**

of Notification Services, 840  
of report models, 512  
of Report Server Project reports, 522–523  
sharing logical model with customers before, 166

#### **derived tables, 144–146**

#### **DESC keyword**

CREATE INDEX statement, 209–210  
ORDER BY clause, 51

#### **design. See also advanced query design**

de-normalization in, 179–180  
diagramming, 159–165, 181–187  
for file-based information, 168–170  
as foundation for all else, 281  
logical models for, 165–167  
logical versus physical, 165  
normalization issues, 155–158  
partitioning for scalability, 180–181  
performance tuning as part of, 670–671  
relationships overview, 158–159  
for reusability, 177–179  
subcategories in, 171–177  
triggers and architecture changes, 377

#### **deterministic functions, 242, 316–317**

#### **diagramming with SQL Server tools**

adding CHECK constraints, 187  
adding existing tables to diagram, 182, 183  
adding foreign keys, 185–186  
adding new table to diagram and database, 183–184

- adding primary keys, 184, 185
- AdventureWorks example, 7, 8
- deleting CHECK constraints, 187
- deleting foreign keys, 187
- deleting primary keys, 185
- dropping tables, 184
- editing CHECK constraints, 187
- editing foreign keys, 186–187
- limitations of, 7
- opening the tools, 181
- saving changes to database, 184
- table windows, 183
- diagrams. See also ERDs (entity-relationship diagrams)**
- IDEFIX paradigm for, 159
- IE paradigm for, 159
- overview, 7
- SQL Server tools for, 7, 181–187
- third-party tools for, 159–160
- dictionary order for indexes, 194–195**
- DIFFERENCE function, 800**
- differential backup, 724. See also backing up**
- Differential Changed Map (DCM), 193**
- DirectoryList() function, 404**
- DirectorySearch() function, 405**
- dirty pages, 336**
- dirty reads, 343, 347**
- DISABLE option**
- ALTER FULLTEXT INDEX statement, 618
- ALTER INDEX statement, 217, 735
- disabling**
- actions on replication subscribers, 588
- constraints, for existing data, 122–124
- constraints, temporarily, 124–126
- full-text indexing for database, 610, 611
- indexes, 217, 735
- logins with ALTER LOGIN, 643
- triggers, 377–378
- diskadmin role, 657**
- DISTINCT clause**
- for derived tables, 146
- overview, 57–59
- for UNIONS, 70
- Distributed Management Objects (DMO), 740**
- Distribution Agent, 572**
- distributor (replication)**
- defined, 568
- topology models, 582–583
- DLLs, compiling for assemblies, 394**
- DML (Data Manipulation Language). See also T-SQL (Transact-SQL)**
- basic statements, 35
- triggers, 362
- DMO (Distributed Management Objects), 740**
- Do Case statement (Xbase). See CASE statement**
- documents**
- attributes of, 171–172
- logical model for subcategory implementation, 173–175
- physical model for subcategory implementation, 175–176
- subcategories of, 172
- domain constraints, 103. See also specific kinds**
- domain data. See columns**
- domain listing, 45**
- domain tables**
- defined, 104
- non-identifying relationships with, 160
- referential integrity for, 104
- for subcategories, 174
- double quotes (“) for names, 16**
- downtime risks, 685**
- DRI (declarative referential integrity), 362, 367–369**
- driver support, 686**
- DROP parameter (ALTER FULLTEXT INDEX), 618**
- DROP statement**
- for custom data types, 419
- for database objects, 95
- for defaults, 128
- error for schema-bound tables, 112
- for full-text catalogs, 614
- for full-text indexes, 619
- for indexes, 218, 222
- for logins, 643
- for rules, 128
- for sprocs, 283
- syntax, 95
- for tables, 95
- for triggers, 390
- for views, 238
- for XML schema collections, 463
- DROP\_EXISTING option (CREATE INDEX), 211–214**
- dropping. See DELETE statement; deleting; DROP statement**
- DTS (Data Transformation Services), 31, 545, 546. See also SSIS (SQL Server Integration Services)**
- dual core processors, 683**
- durability of transactions, 359**
- dynamic cursors**
- avoiding when possible, 442
- CursorTest example, 443–445
- FAST\_FORWARD cursor conversion to, 446
- keyset-driven cursors versus, 442, 445
- non-unique index for, avoiding, 445
- rebuilt with each FETCH, 442
- sensitivity, 435, 442, 445
- dynamic SQL. See EXEC command**

## E

### **ELSE statement, 267–268**

#### **enabling**

- CLR, 394
- full-text indexing for database, 610–611
- logins with `ALTER LOGIN`, 643
- NetLibs, 21–22
- triggers, 377–378

#### **encryption**

- for password storage, 638, 646
- for triggers, 363
- for user profile storage, 638
- for views, 239–241

#### **entity boxes in ERDs, 160–161**

#### **entity constraints, 103–104. See also specific kinds**

#### **entity data. See rows**

#### **equals sign (=) in comparison operators, 47**

#### **erasing. See deleting**

#### **ERDs (entity-relationship diagrams). See also diagramming with SQL Server tools; diagrams**

- defined, 7
- entity boxes, 160–161
- examples, 164–165
- for exclusive subcategories, 173
- IDEFIX paradigm for, 159
- IE paradigm for, 159
- need for, 159
- for non-exclusive subcategories, 172–173
- relationship lines, 161–162
- SQL Server tool for, 159
- for subcategories logical model, 174–175
- for subcategories physical model, 176
- terminators, 162–163
- third-party tools for, 159–160

#### **@@ERROR function**

- error 547 trapped by, 293–294
- error 2714 not trapped by, 295
- @Error variable versus, 293
- `ERROR_NUMBER()` function compared to, 292
- `INSERT` example using, 292–293
- overview, 251, 764
- saving the value from, 292–293
- `TRY/CATCH` blocks versus, 295
- using in sprocs, 293–295

#### **error handling**

- @@ERROR for, 292–295
- manually raising errors, 296–299
- for sprocs, 290–301
- `TRY/CATCH` blocks for, 277–280, 291, 295

#### **error handling for sprocs**

- common types of errors, 290
- custom error messages, 299–301
- @@ERROR for, 292–295
- inline errors, 291

- manually raising errors, 296–299
- `TRY/CATCH` blocks for, 291, 295

#### **error levels**

- `ERROR_NUMBER()` function for, 279
- `ERROR_SEVERITY()` function for, 279
- returned by `RAISERROR` command, 296–297
- table summarizing, 278, 297

#### **error messages. See also error handling**

- for batches, 255
- custom, for sprocs, 299–301
- custom, triggers for, 371
- for `DROP` statement with schema-bound tables, 112
- error 547, 291, 293–294
- error 1205 (deadlock), 355
- error 2714, 295
- functions for error condition retrieval, 279
- for `GO` command in pass-through queries, 255
- passed through by data access object models, 291
- system, viewing all, 279–280

#### **@Error variable, 293**

#### **ERROR\_LINE() function, 279**

#### **ERROR\_MESSAGE() function, 279**

#### **ERROR\_NUMBER() function**

- @@ERROR compared to, 292
- overview, 279
- testing for object existence with, 287

#### **ERROR\_PROCEDURE() function, 279**

#### **ERROR\_SEVERITY() function, 279**

#### **ERROR\_STATE() function, 279**

#### **escalation of locking, 346**

#### **ETL (Extract, Transform, and Load), 546. See also SSIS (SQL Server Integration Services)**

#### **Evaluate statement (COBOL). See CASE statement**

#### **ExampleAggregate assembly**

- `Accumulate` method, 407, 409–410
- changing class name to `Product`, 408, 409
- creating aggregate function from, 411
- creating Visual Studio project for, 408
- declaring member variables, 408, 409
- `fContainsNull` variable, 409
- `Init` method, 407, 409
- initializing member variables, 409
- `Merge` method, 407, 410
- support methods for calls, 407
- template for, 408
- `Terminate` method, 407, 410
- testing the function, 411–412
- uploading to SQL Server, 410

#### **ExampleProc assembly**

- adding a sproc to the assembly project, 395
- command object for, 396
- creating `spCLRExample` sproc from, 398–399
- creating Visual Studio project for, 395
- database connection for, 396
- `ExampleSP` method for, 396

- executing the command with `.Pipe` object, 397
- populating the output variable, 397
- setting references, 395
- `StoredProcedures` class for, 396
- uploading to SQL Server, 397–398
- Visual Studio directive for, 396
- ExampleSP method, 396**
- ExampleTrigger assembly**
  - CASE statement for event types, 413–414
  - creating class for, 413
  - creating `trgExampleTrigger` trigger from, 415
  - creating Visual Studio project for, 412
  - including all event types, 413
  - obtaining context for, 412–413
  - template for, 412–413
  - uploading to SQL Server, 414
- ExampleTVF assembly**
  - creating `fExampleTVF` function from, 406
  - creating Visual Studio project for, 404
  - `EXTERNAL_ACCESS` permission set for, 406
  - `FillRow()` function, 405
  - function to enumerate directory list, 405
  - `GetFiles` method call, 405
  - populating the array, 405
  - setting references, 404
  - top-level function call, 404
  - uploading to SQL Server, 406
- ExampleUDF assembly**
  - code for validating e-mail fields in tables, 401–402
  - creating `fCLRExample` UDF from, 402
  - template for, 400–401
  - uploading to SQL Server, 402
- exclamation mark (!)**
  - in comparison operators, 47
  - separating groups in universal table column names, 475
- exclusive locks**
  - compatibility with other lock modes, 349
  - intent exclusive, 348
  - overview, 347
- EXEC command**
  - AdventureWorks example, 260–262
  - capturing return values for sprocs, 289
  - concatenation performed before calling, 262, 265
  - connection context of, 262
  - Management Console for running, 260
  - not allowed in UDFs, 262, 265
  - restrictions for, 262
  - scope of, 262–264
  - security context for, 262, 264
  - in sprocs, 264
  - sprocs in, 265, 303, 304
  - syntax, 260
  - temp table to communicate between scopes, 264
  - transaction context of, 262
  - WITH `RECOMPILE` option for sprocs, 303, 304
- EXEC keyword for sproc calls, 288, 399**
- Execute Package Utility, 560–562**
- Execute Process Task (SSIS), 554**
- Execute SQL Task (SSIS), 555, 556**
- Execute Tasks (SSIS), 550**
- EXECUTE user rights, 649**
- executing packages (SSIS)**
  - Execute Package Utility for, 560–562
  - Management Studio for, 562–563
  - within a program, 560
  - with SQL Server Agent, 560
- .exist method (XML), 463, 468–469**
- EXISTS operator**
  - IF . . . ELSE examples using, 267, 268, 269–270
  - indexes compared to, 200
  - join-based syntax versus, 147
  - nested subquery example, 146–147
  - overview, 48
  - performance benefits of, 199
  - testing object existence before creating, 147–148
  - TRUE/FALSE return from, 146
- EXP function, 779**
- expiration of passwords, 635–636**
- EXPLICIT option (FOR XML clause)**
  - `cdata` directive, 487
  - column naming by, 475–479
  - described, 470, 474
  - directives, associating information with attributes, 477–478
  - directives, attribute names and, 476
  - directives, overview, 478–479
  - `element` directive, 479–480
  - exclamation mark separating groups in column names, 475
  - format for column names, 476
  - granularity of control provided by, 474
  - `hide` directive, 480–482
  - `id` directive, 482–483
  - `idref` directive, 483–484
  - `idrefs` directive, 483, 484–486
  - minimum query using, 476–477
  - `Parent` metadata column, 475
  - `PATH` option compared to, 486
  - sufficient metadata required by, 474–475
  - `Tag` metadata column, 475, 476
  - universal table created by, 474–475
  - `xml` directive, 480
  - `xmltext` directive, 486
- exporting, BCP for, 534–535**
- expressions**
  - in CASE statements, 270–271
  - for computed columns, 87

### expressions (continued)

- in COUNT () function, 55–56
  - in IF . . . ELSE statements, 266
  - in SET clause for UPDATE statement, 64
- extended sprocs (XPs), 304–305**
- extensibility, subcategories for, 176–177. See also scalability**

**Extensible Markup Language. See XML**

**extents, 190–191, 346**

**external calls, 150–151**

**EXTERNAL\_ACCESS permission set**

- conditions required for, 406
- datasource connections possible with, 400
- described, 398
- for ExampleProc assembly, 400
- for ExampleTVF assembly, 406

**Extract, Transform, and Load (ETL), 546. See also SSIS (SQL Server Integration Services)**

## F

**factorials, computing, 306**

**FAST\_FORWARD cursors, 445–446**

**fCLRExample assembly-based scalar UDF. See also**

**ExampleUDF assembly**

- applying as CHECK constraint, 403
- CREATE FUNCTION statement for, 400, 402
- creating from ExampleUDF assembly, 402
- testing, 402–403

**FETCH statement**

- ABSOLUTE argument, 453
- for altering data within a cursor, 454–455
- dynamic cursor example, 443–444
- dynamic cursors rebuilt each time issued, 442
- FIRST argument, 433, 453
- for forward-only cursors, 432
- global scope example, 429
- keyset-driven cursor example, 439–440
- LAST argument, 433, 453
- local scope example, 431
- NEXT argument, 432, 453
- overview, 452
- PRIOR argument, 432, 453
- RELATIVE argument, 453
- for scrollable cursors, 432–433
- SCROLL\_LOCKS example, 449
- simple cursor example, 424, 425
- spCursorScope example, 429, 430, 431
- spCursorScroll example, 433
- static cursor example, 436–437
- TYPE\_WARNING example, 451

**@@FETCH\_STATUS function**

- overview, 251, 764
- possible values, 424
- simple cursor example, 424

**fExampleTVF table-valued function, 406–407**

**fifth normal form, 158**

**file storage for BLOBs, 168–170**

**File System Tasks (SSIS), 550**

**FILEGROUP\_ID function, 786**

**FILEGROUP\_NAME function, 786**

**FILEGROUPPROPERTY function, 786**

**filegroups**

- ON keyword for, 88
- overview, 7
- primary, 7, 78
- secondary, 7
- storing full-text catalog internal structures with, 612
- TEXTIMAGE\_ON clause for, 88
- for VLDBs, 78

**FILEGROWTH parameter (CREATE DATABASE), 79**

**FILE\_ID function, 785–786**

**FILE\_NAME function, 786**

**FILENAME parameter (CREATE DATABASE), 78**

**FILEPROPERTY function, 787**

**FILLFACTOR option**

- CREATE INDEX statement, 211
- with DBCC DBREINDEX command, 227, 228

**FillRow () function, 405**

**filtering. See also WHERE clause**

- client- versus server-side processing, 673
- data for replication, 570, 596
- SQL Server Profiler Column Filters for, 695–696
- table as source for, 49

**firing order for triggers**

- controlling for logic reasons, 379–380
- controlling for performance reasons, 380
- overview, 378–379
- setting with sp\_settriggerorder, 378

**first normal form (1NF), 157**

**547 error**

- CHECK constraints for monitoring, 344
- from foreign key violations, 291
- for inline errors in sprocs, 291, 293–294
- trapped by @@ERROR, 293–294

**fixed database roles, 656, 658–659**

**flags and switches**

- BCP, 527–530
- condition, triggers for setting, 373–375
- OPENXML function mapping, 494
- RAISERROR command, 298
- SQLCMD, 259

**float data type, 11**

**FLOOR function, 275, 779**

**fnContactList () UDF, 311**

**fnContactSearch () UDF, 311**

**fnGetReports () UDF, 313–314**

**FOR ATTACH parameter (CREATE DATABASE), 80**

**FOR clause**

- CREATE TRIGGER statement, 364
- DECLARE statement for cursors, 452

**For Each Container task (SSIS), 549****For Loop Container task (SSIS), 548****FOR XML clause. See also specific options**

- AUTO option, 470, 473–474
- BINARY BASE64 option, 470
- ELEMENTS option, 470
- EXPLICIT option, 470, 474–487
- overview, 57
- PATH option, 470, 488–492
- RAW option, 470, 471–473
- ROOT option, 471
- syntax, 469
- TYPE option, 471
- XMLDATA option, 470

**foreign keys**

- adding to existing table, 110
- adding with diagramming tools, 185–186
- avoiding for reusability, 178
- BCP enforcement of, 531
- cascading actions with, 112–116
- deleting with diagramming tools, 187
- dropping before moving columns, 94
- editing with diagramming tools, 186–187
- in entity boxes, 161
- error 547 from violating, 291, 293–294
- in identifying relationships, 160
- indexes on, 218–219
- in logical model, 167
- maximum per table, 110
- in non-identifying relationships, 160
- NULLs in, 108, 117
- other data integrity methods compared to, 129–131
- overview, 108–116
- primer row for, 110, 111, 112
- referencing versus referenced tables, 108
- as referential integrity constraints, 104
- required versus optional values in, 117
- in self-referencing tables, 110–112
- specifying in CREATE TABLE statement, 87, 109
- trigger performance versus, 179

**format files**

- for data file with fewer columns than the table, 538–539
- for data file with more columns than the table, 539
- default, 536
- example using, 540
- for mismatched field order, 539–540
- non-XML, 536–537
- performance, maximizing, 541
- situations benefiting from, 536
- for timestamp or computed columns with BCP, 531
- XML, 537

**FORMATMESSAGE function, 806****FORMSOF ( ) function (FTS), 631****forward slash (/) with FOR XML PATH clause, 491–492****forward-only cursors**

- client-side implementation, 673
- as default, 432
- described, 432
- efficiency of, 434

**fourth normal form, 158****fragmentation of indexes**

- ALTER INDEX statement for managing, 227, 734–736
- DBCC DBREINDEX command for managing, 227–228, 733
- defined, 223
- identifying, 224–227
- problems from, 223–224

**FREETEXT statement, 626, 628****FREETEXTTABLE function, 628, 794****FROM clause**

- CREATE ASSEMBLY statement, 398
- CREATE LOGIN statement, 641–642
- DELETE statement, 64
- SELECT statement, 36

**FTP Tasks (SSIS), 550****FTS (Full-Text Search). See also full-text catalogs; full-text indexes**

- advantages of, 607–608
- architecture, 608–610
- against BLOBs, data type for, 170
- against BLOBs, file storage for, 170
- Booleans in searches, 629
- changing startup to automatic, 609
- Configuration Manager for service management, 19
- CONTAINS statement, 624–625
- CONTAINSTABLE statement, 626–627
- document types supported by, 607
- enabling for database, 610–611
- FORMSOF ( ) function, 631
- FREETEXT statement, 626
- FREETEXTTABLE statement, 628
- INFLECTIONAL keyword, 631
- ISABOUT ( ) function, 630–631
- NEAR keyword, 629–630
- noise word list, 631–632
- phrases in searches, 628
- proximity searches, 629–630
- query syntax, 624–631
- WEIGHT keyword, 630–631

**full backup, 724. See also backing up****FULL JOINS**

- with DELETE statement, 65–66
- inclusive nature of, 42
- overview, 43–44

**Full recovery model, 728**

## full-text catalogs

- altering, 613–614
- creating with new syntax, 611–613
- creating with old syntax, 619–620
- defined, 11
- dropping, 614
- removed when indexing is disabled, 611
- in storage hierarchy, 194

## full-text indexes

- altering, 617–619
- checking if enabled, 610
- creating with new syntax, 614–617
- creating with old syntax, 621–622
- creating without populating, 617
- disabling, 610, 611
- dropping, 619
- enabling, 610–611
- populating, 609, 618–619, 622–624
- removed when indexing is disabled, 611
- SQL Server indexes versus, 609–610

## Full-Text Search. *See* FTS

### FULLTEXTCATALOGPROPERTY function, 787

### FULLTEXTSERVICEPROPERTY function, 787–788

### fully qualified names, 73

### functions. *See also* aggregate functions; system functions; UDFs (user-defined functions); specific functions

- deterministic, 242, 316–317
- for error condition retrieval, 279
- referenced by indexed views, 242
- table-valued, creating from assemblies, 403–407

## G

### GAM (Global Allocation Map), 192

### GETANSINULL function, 806–807

### GETDATE function

- DayOnly() UDF using, 310
- described, 309–310
- as non-deterministic, 316
- overview, 776

### GetFiles method, 405

### GETUTCDATE function, 776

### Global Allocation Map (GAM), 192

### global variables (legacy system functions), 762–770.

*See also* specific functions

### Globally Unique Identifiers. *See* GUIDs

### GO statement

- not a T-SQL command, 254, 255
- not sent to server with batches, 255
- in pass-through queries, avoiding, 255
- separate line required for, 253, 254
- for separating scripts into batches, 253–254

### GRANT statement, 650–652

### graphical showplan, 689–690

### greater than sign (>) in comparison operators, 47

### GROUP BY clause

- aggregate functions with, 53–56
- HAVING clause with, 56–57
- overview, 52–56

### GROUPING function, 772

### guest account, 664

### GUIDs (Globally Unique Identifiers)

- defined, 86
- NEWID() function for, 86
- primary keys versus, 106
- ROWGUIDCOL parameter for, 85–86
- specifying for logins, 641

## H

### hardware considerations

- budgeting, 678
- CPU intensive tasks, 683–684
- dedicated SQL Server, 679
- diminishing returns, 685–686
- driver support, 686
- ideal system, 686
- I/O intensive tasks, 680–683
- I/O versus CPU intensive tasks, 679–680
- lost data, 685
- OLTP and OLAP on separate servers, 684
- on-site versus off-site, 684
- questions to ask when purchasing, 678–679
- RAID, 680–683
- risks of being down, 685

### HAS\_DATABASE function, 796

### HAVING clause, 56–57

### heap

- defined, 201
- non-clustered indexes on, 203–205
- row ID (RID) for, 201, 203

### hints. *See* optimizer hints

### HOLAP (Hybrid OLAP), 831

### HOLDLOCK/SERIALIZABLE optimizer hint, 349, 355

### horizontal filtering or partitioning, 570, 596

### HOST\_ID function, 807

### HOST\_NAME function, 807

### HTTP endpoints

- creating and managing, 499–500
- defined, 498
- methods, 499
- not supported in SQL Server Express, 498
- not supported in Windows XP, 498
- realm of, 497
- security, 498
- SOAP for, 498

### Hunter, David (*Beginning XML*), 458

### Hybrid OLAP (HOLAP), 831

- I**
- ideal system, 686**
- IDEFIX diagrams, 159**
- IDENT\_CURRENT function, 807**
- identifying relationships**
  - defined, 160, 162
  - entity box representation of, 161
  - relationship lines for, 162
- IDENT\_INCR function, 807**
- identity columns**
  - Access AutoNumber columns compared to, 85
  - choosing in Management Studio, 97
  - data types for, 84
  - defined, 84
  - @IDENTITY function for, 251, 252
  - primary keys versus, 85
  - ROWGUIDCOL compared to, 85–86
  - specifying in CREATE TABLE statement, 84–85
  - uses for, 85
  - values automatically generated for, 60, 236
- IDENTITY function, 808**
- @IDENTITY function**
  - moving value to local variable, 252
  - overview, 251, 765
  - using in scripts, 252
- IDENTITY parameter (CREATE TABLE), 84–85**
- IDENTITY property, disabling on replication subscribers, 588**
- IDENT\_SEED function, 807**
- @IDLE function, 765**
- IE (Information Engineering) diagrams, 159**
- IEnumerable interface, 404**
- IF...ELSE statement**
  - BEGIN...END blocks with, 268–270
  - for creating table if not already existing, 267, 268, 269–270
  - ELSE clause, 267–268
  - expressions in, 266
  - syntax, 266
  - testing NULLs in, 266
- IGNORE\_DUP\_KEY option (CREATE INDEX), 211**
- image data type, 13**
- image system functions, 812–813**
- immediate-update subscribers, 578, 580–581**
- impersonation contexts, 80**
- implicit transactions, 340–341**
- implied indexes created with constraints, 215**
- importing, BCP for, 531–534**
- IN operator, 48**
- IN PATH parameter (CREATE FULLTEXT CATALOG), 612**
- INCLUDE columns option (CREATE INDEX), 210**
- INDEX\_COL function, 788**
- indexed views**
  - creating the index for, 243
  - defined, 8
  - example, 242–243
  - index used by SQL Server, 244
  - performance impacts of, 8–9
  - restrictions for, 242
- indexes (SQL Server). See also full-text indexes**
  - adjusting page densities, 211
  - altering, 215–217
  - BCP with, 531, 534, 541
  - B-Tree structure for, 195–199
  - clustered, 6, 109, 193, 200–203
  - collation options, 194–195
  - column order in, 222
  - creating as implied object with constraints, 215
  - creating for indexed views, 243
  - creating with CREATE INDEX statement, 208–214
  - as critical to planning and maintenance, 189
  - Database Engine Tuning Advisor for, 222, 672
  - defined, 194
  - disabling, 217, 735
  - dropping, 218, 222
  - dropping and recreating, 211–212
  - for dynamic cursors, 445
  - EXISTS operator compared to, 200
  - on foreign keys, 218–219
  - fragmentation of, 223–228, 733–736
  - full-text indexes versus, 609–610
  - for keyset-driven cursors, 438, 440, 441
  - maintaining, 223–228
  - maintenance, 733–736
  - non-clustered, defined, 6
  - non-clustered on a clustered table, 205–208
  - non-clustered on a heap, 203–205
  - not usable with LIKE operator, 607
  - overview, 5–6, 228–229
  - pages for, 192
  - performance considerations, 189, 200, 219, 671–672
  - questions to ask about, 229
  - rebuilding, 211–212, 216–217, 734–735
  - recreating after moving columns, 94
  - reorganizing, 217, 735–736
  - selectivity within, 218–219
  - sort order for, 194–195, 209–210
  - storing separately from the data, 214
  - sysindexes table for, 203
  - TableCursor for rebuilding all indexes in database, 423–427
  - types of, 200
  - use by SQL Server, 200
  - XML, 214–215, 497
- INDEXKEY\_PROPERTY function, 788**
- INDEXPROPERTY function, 788–789**

**INFLECTIONAL keyword (FTS), 631**

**Information Engineering (IE) diagrams, 159**

**INFORMATION\_SCHEMA access path, 37**

**inline errors, 291**

**inline use of UDFs, 309**

**inline views (derived tables), 144–146**

**INNER JOINS**

alternative (legacy) syntax for, 67

as default type, 41–42

exclusive nature of, 40

overview, 39–42

self-referencing, 40, 41

syntax, 40

table aliases with, 40

**INSERT INTO...SELECT statement, 61–62**

**INSERT statement**

column list with, 59, 60

DEFAULT keyword, 60

dependency chains with, 114

INTO keyword, 59

syntax, 59–60

with views, 236–237

**INSERT triggers**

checking the delta of an update, 369–371

CREATE TRIGGER parameter for, 366

creating from assemblies, 412–417

for feeding data into de-normalized tables, 372–373

firing order for, 378

INSTEAD OF triggers, 381–383

for requirements sourced from other tables, 367–369

for setting condition flags, 373–375

**INSERT user rights, 649**

**INSERTED tables for triggers, 366, 370**

**installing Microsoft Sample set, 394**

**INSTEAD OF triggers**

for changing data with views containing joins, 236, 237

defined, 380

DELETE triggers, 384–385

described, 364

FOR | AFTER triggers versus, 364–365, 380

INSERT triggers, 381–383

tables for examples, 380–381

types of, 380

UPDATE triggers, 384

**int data type**

for identity columns, 84

overview, 11

**Integration Services. See SSIS (SQL Server Integration Services)**

**integrity. See data integrity**

**Intelligence Studio. See Business Intelligence Studio**

**intent locks, 348, 349**

**INTO keyword (INSERT), 59**

**inversion keys, 105**

**I/O intensive tasks**

CPU intensive tasks versus, 679–680

RAID for, 680–683

**@@IO\_BUSY function, 765**

**ISABOUT() function (FTS), 630–631**

**ISDATE function, 808**

**ISDETERMINISTIC property, 316**

**IS\_MEMBER function, 796–797**

**ISNULL function**

with correlated subqueries, 143–144

overview, 808

**ISNUMERIC function, 808**

**isolation levels**

performance tuning, 675

for preventing dirty reads, 343, 353

for preventing non-repeatable reads, 344, 354–355

for preventing phantoms, 345

READ COMMITTED, 343, 353

READ UNCOMMITTED, 354

REPEATABLE READ, 344, 354–355

SERIALIZABLE, 344, 345, 355

syntax for switching, 353

types available, 353

using lowest possible, 357

**isolation of transactions, 359**

**isql.exe (older versions). See SQLCMD**

**IS\_SRVROLEMEMBER function, 797**

## J

**jobs. See scheduling jobs**

**JOINS. See also specific kinds**

across multiple servers, 76

alternative (legacy) syntax for, 66–68

ANSI syntax recommended for, 66

changing data with views containing, 236, 237

CROSS JOINS, 44–45, 68

with DELETE statement, 65–66

for derived tables, 144–146

exclusive, 40

FULL JOINS, 43–44

inclusive, 42

INNER JOINS, 39–42, 67

OUTER JOINS, 42–43, 67–68

overview, 38–39

subqueries versus, 134–135, 152–153

on tables returned by UDFs, 311, 312, 315

UNIONS versus, 69

WHERE clause with, 48–49

## K

**KEY INDEX parameter (CREATE FULLTEXT INDEX), 616**

**keys. See also foreign keys; primary keys**

- importance of, 106
- inversion, 105
- locking, 346
- other data integrity methods compared to, 129–131

**keyset-driven cursors**

- CursorTest example, 438–441
- dynamic cursors versus, 442, 445
- FAST\_FORWARD cursor conversion to, 446
- keys for, 438
- overview, 438
- sensitivity, 441
- unique index required for, 438, 440, 441

**keywords in names, avoiding, 16****L****@LANGID function, 765****@LANGUAGE function, 765****LANGUAGE parameter (CREATE FULLTEXT INDEX), 615****latency**

- defined, 567
- in merge replication, 574
- replication concerns, 567

**.ldf extension, 78, 190. See also transaction log****leaf-level nodes of B-Trees**

- for clustered indexes, 201
- defined, 196, 200
- page splits at, 199

**LEFT function, 800****LEFT OUTER JOINS, 43****legacy items**

- BCP as, 526
- CREATE INDEX syntax, 209
- defaults as, 10, 126
- JOIN syntax, 66–68
- login creation, 643–646
- password maintenance, 646
- rules as, 10, 126
- sa role as, 657
- sp\_grantdbaccess spoc, 648
- system functions, 762–770

**LEN function, 800****less than sign (<) in comparison operators, 47****lifespan of a cursor**

- closing, 424
- deallocating, 424
- declaring, 423
- full example, 424–425
- main parts, 422–423
- opening, 423–424
- utilizing and navigating, 424

**LIKE operator, 48, 607****linked servers, 76****Locals window (Debugger), 321–322****locking**

- ALLOW settings for indexes, 213
- bulk update locks, 349
- compatibility of lock modes, 349
- cursors, OPTIMISTIC option for, 450
- cursors, READ\_ONLY option for, 447
- cursors, SCROLL\_LOCKS option for, 447–450
- determining locks using Management Studio, 352–353
- escalation of, 346
- exclusive locks, 347
- granularity of, 346
- intent locks, 348
- lock, defined, 190, 342
- lock manager, 342
- lockable resources, 346
- modes, 347–349
- ONLINE option of CREATE INDEX for preventing, 213
- optimizer hints, 349–352
- performance impacts of, 346
- problems prevented by, 342–345
- schema locks, 348
- shared locks, 347
- subcategory issues for, 177
- update locks, 347–348

**@@LOCK\_TIMEOUT function, 765–766****LOG function, 779****LOG ON parameter (CREATE DATABASE), 79****Log Reader Agent, 578, 598****logging. See also transaction log**

- by BCP, 534
- custom error messages with sp\_addmessage, 300
- errors with RAISERROR, 299
- filename extension for log files, 78
- LOG ON parameter for, 79

**logical errors in sprocs, 290****logical models**

- constraints in, 167
- data types in, 167
- parts of, 166–167
- physical models versus, 165
- purpose of, 165–166
- for reusability, 178
- rules in, 165–166, 167
- sharing with customers, 166
- structure in, 167
- for subcategories, 173–175

**logical reads statistics, 691****logins**

- adding users to a database, 647–648
- altering, 642–643
- creating logins WMI, 645
- creating with CREATE LOGIN, 640–642

### logins (continued)

- creating with Management Studio, 643–644
- creating with SMO, 644
- creating with `sp_addlogin`, 640, 644, 645–646
- creating with `sp_grantlogin`, 646
- creating with SQL-DMO, 645
- double password scenarios, avoiding, 638
- dropping, 643
- guest account, 664
- for Management Studio, 26
- number of tries allowed, 637
- one person, one login, one password principle, 634–635
- `securityadmin` role for managing, 657
- SQL Server Authentication, 26, 27, 639–646
- unlocking with `ALTER LOGIN`, 643
- Windows authentication, 26–27, 639, 646–647

### LOG10 function, 779–780

#### lookup (domain) tables

- defined, 104
- non-identifying relationships with, 160
- referential integrity for, 104
- for subcategories, 174

### lost data, costs of, 685

### lost updates, 345

### LOWER function, 800

### lowercase. *See case and case sensitivity*

### LTRIM function, 800

## M

### maintenance. *See also administration tasks*

- for index fragmentation, 223–228, 733–736
- indexes as critical to, 189
- for passwords, 642–643, 646
- performance tuning, 674–675
- rebuilding indexes, 211–212, 216–217, 734–735
- updating statistics, 212

### Maintenance Tasks (SSIS), 551

#### Management Studio

- authentication type for, 26–27
- backing up using, 722–725
- Business Intelligence Studio versus, 504
- changing maximum column length, 461
- columns truncated by, 472
- Configure Distribution Wizard, 589–592, 593
- connection dialog, 25–27
- creating databases using, 96–97
- creating jobs and tasks using, 704–712
- creating logins using, 643–644
- creating operators using, 701–702
- creating tables using, 97–99
- Database Engine Tuning Advisor, 222
- deleting jobs using, 721
- editing databases using, 97

- editing jobs using, 721
- editing tables using, 99
- encrypted views not displayed by, 241
- getting started, 25–27
- identity column selection using, 97
- lock determination using, 352–353
- New Publication Wizard, 593–598, 599
- New Subscription Wizard, 599–603
- Object Explorer, 31–32, 722
- overview, 24–32
- query governor, 692–693
- Query window overview, 27–31
- replication publication configuration using, 592–598
- replication server configuration using, 588–592
- replication subscriber setup using, 598–603
- server type for login, 26
- SQL Server for login, 26
- for SSIS package execution, 562–563
- uses for, 25

### many-to-many relationships, 159

### MARS (Multiple Active Result Sets), 817

### master database, 3. *See also specific tables*

#### materialized (indexed) views

- creating the index for, 243
- defined, 8
- example, 242–243
- index used by SQL Server, 244
- performance impacts of, 8–9
- restrictions for, 242

### mathematical functions, 777–781

### MAX function, 54, 772

### @@MAX\_CONNECTIONS function, 766

### @@MAX\_PRECISION function, 766

### MAXSIZE parameter (CREATE DATABASE), 79

### .mdf extension, 78, 190

### MDX (multidimensional query language), 838

### memory (RAM), 683–684

### Merge Agent, 575

#### merge replication

- autonomy, 574
- defined, 574
- latency, 574
- Merge Agent for, 575
- mixing with other types, 581
- overview, 574–575
- planning requirements, 576
- process of, 575–576
- uses for, 576

### Message Queue Task (SSIS), 550

#### metadata

- access paths to, 37
- constraints as, 10
- defaults as, 10
- defined, 5
- system functions, 781–794

**Microsoft Access, 85****Microsoft Sample set, installing, 394****MIN function, 54, 772****mobile devices, replication with, 588****model database**

- altering, considerations for, 3–4
- as default for CREATE DATABASE statement, 77, 78
- as default for CREATE TABLE statement, 82
- deleting, avoiding, 4
- minimum database size determined by, 4, 78
- as template for new databases, 3–4

**.modify method (XML), 463, 465–466****modulus operator**

- described, 271
- searched CASE example, 273
- simple CASE examples, 271–272

**MOLAP (Multidimensional OLAP), 831****money data type, 11****MONTH function, 776****moving columns**

- using Management Studio, 99
- using T-SQL, problems with, 94

**@mp:id metaproperty, 495****@mp:localname metaproperty, 495****@mp:namespaceui metaproperty, 495****@mp:parentid metaproperty, 495****@mp:parentlocalname metaproperty, 495****@mp:parentnamespaceui metaproperty, 495****@mp:parentprefix metaproperty, 495****@mp:prefix metaproperty, 495****@mp:prev metaproperty, 495****@mp:xmltext metaproperty, 495****MS Search service, 170, 608****msdb database, 4****Multidimensional OLAP (MOLAP), 831****multidimensional query language (MDX), 838****Multiple Active Result Sets (MARS), 817****multiprocessors, 683****N****NAME parameter (CREATE DATABASE), 78****Named Pipes, 22. See also NetLibs (network libraries)****namespaces**

- for sp\_xml\_preparedocument sproc, 493
- SQL Namespaces (SQL NS), 740–741
- WITH NAMESPACES () declaration, 464
- for XML .exist method, 468
- for XML .modify method, 466
- for XML .nodes method, 467
- for XML .query method, 464
- for XML .value method, 465
- XML\_SCHEMA\_NAMESPACE () function, 460–461

**naming. See also aliases**

- of columns by FOR XML EXPLICIT, 475–479
- consistency in, 84
- constraints, 4–5
- database-qualified names, 76
- deferred name resolution, 129
- fully qualified names, 73
- in indexed views, 242
- keywords in names, avoiding, 16
- linked servers, 76
- object names, 73–76
- objects having names, 15
- optional parts of names, 73
- parameters for sprocs, 284
- rules for, 15–16, 83
- schemas, 74–76
- spaces embedded in names, avoiding, 16, 83
- tables, standards for, 83–84
- underscore use in, avoiding, 83

**nchar data type, 12****NCHAR function, 800–801****.ndf extension, 78****NEAR keyword (FTS), 629–630****nested sprocs, 301****nested subqueries**

- ALL operator with, 139
- ANY operator with, 139
- correlated subqueries versus, 140
- defined, 135
- EXISTS example, 146–147
- nested SELECT to find orphaned records, 138
- returning multiple values, 137–138
- returning single value, 136–137
- SOME operator with, 139
- syntax, 135

**nested triggers, 376****.NET. See also .NET assemblies**

- extended sprocs (XPs) with, 304–305
- file storage implementation using, 170
- online help interface for BOL, 18
- programming possibilities increased by, 281
- RMO, 605–606
- T-SQL compliance with, 35

**.NET assemblies. See also specific assemblies**

- compiling, 394–397
- creating aggregate functions from, 407–412
- creating custom data types from, 417–419
- creating scalar UDFs from, 400–403
- creating sprocs from, 395–400
- creating table-valued functions from, 403–407
- creating triggers from, 412–417
- defined, 394
- enabling CLR for, 394
- EXTERNAL\_ACCESS permission set for, 398, 400, 406

### **.NET assemblies (continued)**

installing Microsoft Sample set for examples, 394  
overview, 419–420

SAFE permission set for, 398, 417  
UNSAFE permission set for, 394, 398  
uploading to SQL Server, 397–398  
windowing not supported by, 394

### **@@NETLEVEL function, 766**

### **NetLibs (network libraries). See also specific NetLibs**

communication process, 20  
enabling, 21–22  
included with SQL Server, 20  
performance impacts of, 22  
security issues for, 22  
support required on both client and server, 20

### **New Publication Wizard, 593–598, 599**

### **New Subscription Wizard, 599–603**

### **NEWID function, 86, 808**

### **.nodes method (XML), 463, 466–468**

### **nodes of B-Trees**

non-leaf level, 196–197  
root, 195–196

### **noise word list for FTS, 631–632**

### **NOLOCK/READUNCOMMITTED optimizer hint, 350, 354**

### **non-clustered indexes**

on a clustered table, 205–208  
defined, 6  
on a heap, 203–205  
performance impacts of, 204, 205, 208  
selectivity within, 218–219

### **NONCLUSTERED option (CREATE TABLE), 219**

### **non-identifying relationships**

defined, 160, 162  
entity box representation of, 161  
relationship lines for, 162

### **non-leaf level nodes of B-Trees, 196–197, 200**

### **non-repeatable reads, 343–344**

### **non-typed XML, 459**

### **normalization. See also de-normalization**

defined, 155  
misunderstandings about, 156  
normal form, defined, 155  
normal forms, overview, 157–158  
normalized database, defined, 38, 155  
prerequisites for, 157  
situations benefiting from de-normalization, 179–180

### **NorthwindSecure.sql script, 634**

### **NOT Boolean operator**

with FTS (AND NOT), 629  
overview, 48

### **NOT EXISTS operator, 199**

### **NOT FOR REPLICATION parameter**

CREATE TABLE statement, 85  
CREATE TRIGGER statement, 367  
disabling actions on subscribers, 588

### **NOT NULL parameter**

in CREATE TABLE statement, 86  
for foreign key columns, 117  
in nested SELECT to find orphaned records, 138  
with XML columns, 459

### **Notification Services**

architecture, 839–840  
deployment, 840  
described, 825, 838  
event types, 838–839  
learning curve for, 840–841  
uses for, 839

### **ntext data type, 13**

### **n-tier approach, 166, 169**

### **NULLIF function, 808**

### **NULLs**

allowing with UNIQUE constraints, 117  
ANSI\_NULLS option, 242, 266  
cascading actions for setting, 116  
changing during transfer process, 33  
comparison tests against, 138  
CREATE TABLE statement parameters for, 86  
in foreign keys, 108, 117  
ignored by aggregate functions, 56  
not allowed for primary keys, 106  
overview, 14–15  
specifying explicitly, 86  
terminator indications for, 163  
testing in IF . . . ELSE statements, 266  
testing in OUTER JOIN, 66  
testing with ISNULL () function, 143–144  
with XML columns, 459

### **numeric data type, 11**

### **nvarchar data type, 12**

### **nvarchar (max) data type, 12, 13, 168**

## **O**

### **Object Explorer**

creating a backup from, 722  
overview, 31–32

### **OBJECT\_ID function, 789**

### **OBJECT\_NAME function, 789**

### **OBJECTPROPERTY function, 789–792**

### **OBJECTPROPERTYEX function, 792**

### **objects. See also specific objects**

administration-related, 2  
for connectivity, 815–816  
database (overview), 2–5  
diagrams (overview), 7  
filegroups (overview), 7  
full-text catalogs (overview), 11  
identifiers for, 15–16  
important, list of, 2  
indexes (overview), 5–6

- naming, 73–76
- roles (overview), 10
- rules (overview), 10
- schemas (overview), 6
- sprocs as, 9
- tables (overview), 5–6
- testing existence before creating, 147–148
- transaction log (overview), 5
- triggers (overview), 6
- UDFs (overview), 9
- user-defined data types (overview), 10
- users (overview), 10
- using in same order, 356–357
- viewing dependencies for, 129
- views (overview), 7–9
- off-site versus on-site server, 684**
- OLAP (Online Analytical Processing)**
  - archiving data, 736
  - concurrency in, 342
  - data warehouse tools, 831
  - FILLFACTOR option for indexes with, 211
  - indexes and performance issues for, 672
  - as lesser focus of this book, 155
  - OLTP versus, with Analysis Services, 826–829
  - OLTP versus, with DBCC SHOWCONTIG, 226–227
  - running on separate server from OLTP, 684
  - SSIS for transforming data from OLTP, 546
- OLEDB providers for MS Search service, 170**
- OLTP (Online Transaction Processing)**
  - concurrency in, 342
  - data warehouse use of, 831
  - de-normalizing databases, 674
  - as emphasis of this book, 155
  - FILLFACTOR option for indexes with, 211
  - indexes and performance issues for, 672
  - locks in, 354
  - OLAP versus, with Analysis Services, 826–829
  - OLAP versus, with DBCC SHOWCONTIG, 226–227
  - running on separate server from OLAP, 684
  - SSIS for transforming data for OLAP, 546
  - third normal form for, 155
  - triggers for feeding data into reporting tables, 372–373
- ON clause**
  - CREATE DATABASE statement, 78
  - CREATE FULLTEXT INDEX statement, 616
  - CREATE INDEX statement, 209, 214
  - CREATE TABLE statement, 88
  - CREATE TRIGGER statement, 363
  - of foreign key, for cascading, 112–113
- ON FILEGROUP parameter (CREATE FULLTEXT CATALOG), 612**
- OnError event (SSIS), 551**
- one-to-many relationships, 158, 162**
- one-to-one relationships, 158**
- OnExecStatusChanged event (SSIS), 551**
- Online Analytical Processing. See OLAP**
- Online Transaction Processing. See OLTP**
- OnPostExecute event (SSIS), 552**
- OnProgress event (SSIS), 552**
- on-site versus off-site server, 684**
- OPENDATASOURCE function, 794–795**
- open-ended transactions, not allowing, 358**
- opening or running**
  - commands with no data set in C#, 820–821
  - commands with no data set in VB.NET, 822–823
  - cursors, 423–424
  - Execute Package Utility, 560
  - packages (SSIS), 560–563
  - scripts with SQLCMD, 259–260
  - sprocs in EXEC procedures, 265
  - SQL Server Profiler, 693
- OPENQUERY function, 795**
- OPENROWSET (BULK) function, 542–543**
- OPENROWSET function, 795**
- OPENXML function**
  - cleaning up memory with sp\_xml\_removedocument, 496
  - described, 493
  - mapping flags, 494
  - metaproperties, 495
  - overview, 795–796
  - script example, 496–497
  - setting up document with sp\_xml\_preparedocument, 493
  - syntax, 494
  - WITH clause for schema declaration, 494–495
  - XPath to a node for calling, 494
- operators for jobs**
  - creating using Management Studio, 701–702
  - creating using T-SQL, 702–704
- operators (T-SQL). See also specific kinds**
  - Boolean, 48, 268, 270–271, 272–275, 629
  - standard comparison, 47, 138, 139, 266
  - for WHERE clause, 47–48
- OPTIMISTIC option for cursors, 450**
- optimizer hints**
  - defined, 349
  - examples using, 351–352
  - OPTION clause for, 57
  - syntax for, 351
  - table summarizing, 349–351
- OPTION clause, 57**
- @OPTIONS function, 766–767**
- OR Boolean operator**
  - with FTS, 629
  - overview, 48
- ORDER BY clause**
  - columns available for, 50
  - DESC keyword in, 51
  - overview, 49–52
  - placed after WHERE clause, 50

**orphaned records, finding with nested SELECT, 138**

**osql.exe (older versions). See SQLCMD**

## **OUTER JOINS**

- alternative (legacy) syntax for, 67–68
- with DELETE statement, 66
- inclusive nature of, 42
- LEFT, 43
- left and right tables, defined, 42
- overview, 42–43
- RIGHT, 43
- testing for NULLs, 66

**OUTPUT keyword for sprocs, 287**

**Output window (Debugger), 322**

**ownership. See also naming**

- access for created objects, 77
- ANSI-compliant, 74
- chains of, 80
- changes with SQL Server 2005, 74–75
- of database (dbo), 75–76
- of objects created by db\_owner role, 76
- sa role as dbo, 76
- schema as owner, 74

## **P**

**packages (SSIS)**

- building, 552–559
- connections for, 556–557, 559
- creating a project for, 546–548
- defined, 546
- executing with Execute Package Utility, 560–562
- executing with Management Studio, 562–563
- executing with SQL Server Agent, 560
- executing within a program, 560
- Package Explorer tab, 552
- vbScript file for generating data, 552–553

**@@PACKET\_ERRORS function, 768**

**@@PACK\_RECEIVED function, 767**

**@@PACK\_SENT function, 767**

**PAD\_INDEX option (CREATE INDEX), 210**

**Page Free Space (PFS), 192**

**page splits**

- with B-Trees, 197–199
- clustered index exceptions to, 193, 220–221
- defined, 193
- illustrated, 197–198
- at leaf level, 199
- performance impacts of, 198–199
- at root node level, 198

**pages**

- BCM, 193
- BLOB, 192
- data, 192
- DCM, 193
- in an extent, 190, 191

GAM, 192

index, 192

locking, 346

PFS, 192

rows contained by, 191

SGAM, 192

splits, 193, 197–199, 220–221

in storage hierarchy, 191–193

types of, 191

**PAGLOCK optimizer hint, 350**

**parallelism, MAXDOP index option for, 213**

**PARSENAME function, 809**

**partition-aware applications, 181**

**partitioned tables, 180–181**

**partitioned views, 180, 181, 244**

**passwords**

- changing, authentication types and, 26
  - changing with ALTER LOGIN, 643
  - changing with sp\_password, 646
  - CREATE LOGIN . . . WITH options, 641
  - double, avoiding, 638
  - encrypting, 638, 646
  - expiration of, 635–636
  - length of, 637
  - makeup of, 637
  - number of tries to log in, 637
  - one person, one login, one password principle, 634–635
  - sharing, prohibiting, 635
  - sniffers to enforce, limitations of, 636
  - with SQLCMD, 259
  - storage of, 637–638
- PATH option (FOR XML clause)**
- at symbol (@) with column names, 489–490, 492
  - described, 470
  - EXPLICIT option compared to, 488
  - forward slash (/) with, 491–492
  - named columns using, 489
  - unnamed columns using, 488–489
  - XPath standard with, 488

**PATINDEX function, 801**

**percent sign (%)**

- as modulus operator, 271
- as wildcard, 48

**perfmom (Performance Monitor), 33, 696–697**

**performance. See also hardware considerations; performance tuning; troubleshooting performance**

- asterisk (\*) in selection criteria reducing, 38
- BCP, maximizing, 541
- BLOBs' impact on, 168, 170
- clustered index benefits for, 203
- connectivity considerations, 816–819
- of cursors, 434
- EXISTS and NOT EXISTS benefits for, 199
- firing order for triggers impacting, 380

- of foreign keys versus triggers, 179
- index benefits for, 189, 200
- index fragmentation's impact on, 223–224
- index impacts when modifying data, 219
- indexed views' impact on, 8–9
- join-based syntax versus EXISTS, 147
- locking impacts on, 346
- making reasonable tests for, 153
- meanings of, 670
- NetLibs' impact on, 22
- non-clustered indexes' impact on, 204, 205, 208
- page splits' impact on, 198–199
- partitioning considerations for, 180, 181
- perceived, 670, 671
- portability versus, 36
- reusability's impact on, 179
- Shared Memory advantages for, 22
- sproc benefits for, 284, 302–303
- sproc optimization incorrect for, 303–304
- subcategories for improving, 176
- subcategory bottlenecks, 177
- subqueries versus joins and, 135, 152–153
- trigger considerations, 388–390
- user-defined data types' impact on, 10
- of views, 179, 233

**Performance Monitor (perfmon), 33, 696–697**

**performance tuning. See also hardware considerations;**

**performance; troubleshooting performance**

- client- versus server-side processing, 673–674
- diminishing returns for, 685–686
- index choices, 671–672
- maintenance, 674–675
- OLTP and OLAP on separate servers, 684
- as ongoing task, 671
- on-site versus off-site server, 684
- overview, 669–670
- in requirements-gathering stage, 670–671
- small gains in repetitive processes, 677–678
- sproc organization, 675–677
- strategic de-normalization, 674
- temporary tables, 677

**permissions**

- assigned with roles, 75
- for HTTP endpoints, 498
- for .NET assemblies, 398
- object, granting to users, 648–654
- retained by ALTER PROC statement, 283
- retained by ALTER VIEW statement, 237
- for sprocs, 283
- statement-level, 654–655

**PERMISSIONS function, 809**

**PFS (Page Free Space), 192**

**phantoms, 344–345**

**physical models**

- defined, 165
- logical models versus, 165
- for reusability, 178
- for subcategories, 175–176
- subcategories in, 171

**physical reads statistics, 690–691**

**PI function, 780**

**planning for replication**

- autonomy concerns, 566–567
- connections, 568
- data concerns, 567, 587–588
- importance of, 587
- latency concerns, 567
- merge replication, 576
- for mobile devices, 588
- NOT FOR REPLICATION clause, 588
- schema consistency concerns, 567–568
- snapshot replication, 574
- transactional replication, 580

**Pointer task (SSIS), 548**

**pooled connections, 816, 817**

**populating full-text indexes**

- after altering, 618–619
- creating without, 617
- examples, 623–624
- full population, 618, 622
- incremental population, 618, 622
- overview, 622–624
- process of, 609
- sp\_fulltext\_table sproc options for, 622, 623
- update (change tracking) population, 619, 622–623

**port settings for TCP/IP, 664–665**

**portability, 36**

**POWER function, 780**

**precedence, establishing with batches, 256–258**

**primary filegroup, 7, 78**

**primary keys**

- adding to existing table, 107–108
- adding with diagramming tools, 185
- BCP enforcement of, 531
- as clustered index default, 219, 220
- defined, 106
- deleting with diagramming tools, 185
- in entity boxes, 161
- as entity constraints, 104
- GUIDs versus, 106
- in identifying relationships, 160
- identity columns versus, 85
- indexing every table with, 672
- in logical model, 167
- need for, 106
- in non-identifying relationships, 160
- NULLs not allowed for, 106
- other data integrity methods compared to, 129–131

### primary keys (continued)

overview, 106–108  
specifying in `CREATE TABLE` statement, 87, 107  
`UPDATE` statement for, avoiding, 64

**PRIMARY keyword (CREATE DATABASE), 78**

**PRIVILEGES keyword (GRANT), 650**

**processadmin role, 657**

**processors. See CPU intensive tasks**

**@@PROCID function, 768**

**Profiler. See SQL Server Profiler**

### protocols

in Configuration Manager, 21  
SOAP, 498  
TCP/IP, 21–22, 24, 664–665

**publications. See subscriptions (replication)**

**publisher (replication)**

configuring a publication, 592–597  
defined, 568  
topology models, 582–583, 584–586

## Q

**Query Analyzer (older versions). See Query window**

**query design. See advanced query design**

### query execution plans

forcing sort order for, 50  
graphical showplan for, 689–690  
Include Actual Execution Plan option, 30–31  
Show Estimated Execution Plan option, 30–31  
`SHOWPLAN` options for, 687–689

**query governor, 692–693**

**query hints. See optimizer hints**

**.query method (XML), 463–464**

**Query Optimizer, 30**

### Query window

color coding in, 27  
DB combo box, 31  
described, 27  
execute button, 28  
getting started, 27–29  
Include Actual Execution Plan option, 30–31  
Results in Grid option, 29  
Results in Text option, 29  
Results to File option, 30  
selecting default database for queries, 31  
Show Estimated Execution Plan option, 30–31

**QUOTED\_IDENTIFIER option, 242**

**QUOTENAME function, 801**

## R

**RADIANS function, 780**

### RAID

backups more important for data safety, 683  
in ideal system, 686

in larger installations, 682  
levels (table), 681–682  
meanings of acronym, 680–681  
minimum level for database installations, 682  
transaction log integrity with, 682

### RAISERROR command

arguments with, 297–299  
flags, 298  
message ID/message string for, 296  
placeholder type indicators, 298  
setting parameter width, precision, and long/short status, 298–299  
severity parameter, 296–297  
state parameter, 297  
syntax, 296  
`WITH LOG` option, 299  
`WITH NOWAIT` option, 299  
`WITH SETERROR` option, 299

**RAM (memory), 683–684**

**RAND function, 780**

**Random Array of Independent Disks. See RAID**

**Random Array of Individual Disks. See RAID**

**ranged queries, clustered indexes for, 220–221**

**RAW option (FOR XML clause)**

attribute names, 471  
columns truncated by Management Studio, 472  
described, 470, 471  
hierarchical nature of data not displayed by, 473  
simple example, 471–472  
XML DOM less deep with, 473

**RDBMSs (Relational Database Management Systems), 1**

**RDL (Report Definition Language), 517, 521–522**

**READ COMMITTED isolation level**

dirty reads prevented by, 343  
overview, 353

**READ UNCOMMITTED isolation level, 354**

**read-ahead reads statistics, 691**

**READCOMMITTED optimizer hint, 350**

**READCOMMITTEDLOCK optimizer hint, 350**

**READ\_ONLY option for cursors, 447**

**READPAST optimizer hint, 350**

**READUNCOMMITTED/NOLOCK optimizer hint, 350, 354**

**REBUILD option**

`ALTER FULLTEXT CATALOG` statement, 613  
`ALTER INDEX` statement, 216–217, 425–427, 734–735

**rebuilding indexes**

all in database, `TableCursor` for, 423–427  
`ALTER INDEX` statement for, 216–217, 425–427, 734–735

by dropping and recreating, 211–212

**recovery. See also backing up**

defined, 729  
to different location, 730

- models for, 728–729
- to original location, 729
- status, 730
- T-SQL for, 730–733
- recovery models, 728–729**
- recursion**
  - defined, 305
  - looping versus, 307
  - in sprocs, 305–307
  - 32-level limit for, 306–307
  - in triggers, 376
- Redundant Array of Inexpensive Disks. See RAID**
- REFERENCES user rights, 649**
- referential integrity. See also data integrity; foreign keys**
  - constraints, defined, 104
  - for domain tables, 104
  - DRI (declarative referential integrity), 362, 367–369
  - triggers for enforcing, 362
- referential integrity actions, 113–114**
- Relational Database Management Systems (RDBMSs), 1**
- Relational OLAP (ROLAP), 831**
- relationship lines in ERDs, 161–162**
- relationships. See also ERDs (entity-relationship diagrams)**
  - identifying, 160, 161, 162
  - many-to-many, 159
  - non-identifying, 160, 161, 162
  - one-to-many, 158, 162
  - one-to-one, 158
  - subcategories, 171–177
  - types of, 158–159
- removing. See deleting**
- @@REMSERVER function, 768**
- REORGANIZE option**
  - ALTER FULLTEXT CATALOG statement, 613–614
  - ALTER INDEX statement, 217, 735–736
- REPEATABLE READ isolation level**
  - non-repeatable reads prevented by, 344
  - overview, 354–355
- REPEATABLEREAD optimizer hint, 350, 355**
- REPLACE function, 801**
- REPLICATE function, 801**
- replication. See also specific kinds**
  - autonomy, 566–567, 570, 571, 572
  - configuring a publication, 592–598
  - configuring the server for, 588–592
  - connections for, 568, 570, 581
  - for creating database copies for direct access, 302
  - data concerns, 567, 587–588
  - data convergence, 567
  - defined, 85, 565
  - filtering data, 570, 596
  - latency, 567
  - load and data distribution issues solved by, 565
  - merge, 574–576
  - mixing types of, 581
  - for mobile devices, 588
  - models, overview, 570–571
  - not firing triggers for tasks related to, 367
  - NOT FOR REPLICATION parameter, 85, 367
  - planning for, 566–568, 574, 576, 580, 587–588
  - pull subscriptions, 569–570
  - push subscriptions, 569–570
  - RMO for, 605–606
  - roles, 568–569
  - schema consistency, 567–568
  - setting up in Management Studio, 588–603
  - snapshot, 571–574
  - subscribers, defined, 569
  - subscribers, immediate-update, 578, 580–581
  - subscribers, setting up, 598–603
  - subscribers, types of, 570
  - topology models, 581–586
  - transactional, 577–580
  - transactional consistency, 567
  - using the replicated database, 603–605
- Replication Management Objects (RMO), 605–606**
- Report Definition Language (RDL), 517, 521–522**
- Report Model Wizard, 509–511**
- report models**
  - advantages of, 517
  - building, 509–512
  - changing table properties, 511–512
  - Connection Manager for, 506–507
  - Data Source Views for, 507–508
  - Data Source Wizard for, 505, 507
  - deploying, 512
  - making available, 516–517
  - opening a project for, 504
  - programmability of, 517
  - RDL for reports, 517
  - report creation using, 512–517
  - Report Model Wizard for, 509–511
  - rights for report design and execution, 517
  - saving reports, 516
  - setting up Visual Studio for, 505
- Report Server Projects**
  - deploying the report, 522–523
  - formatting dates, 520–521
  - opening a project for, 517
  - RDL for reports, 521–522
  - Report Wizard for, 518–519
- Report Wizard, 518–519**
- Reporting Services**
  - building report models, 504–512
  - Configuration Manager for service management, 19
  - Data Source Views, 507–508
  - deploying report models, 512
  - making the model available, 516–517

## Reporting Services (continued)

- overview, 33, 504
- programmability of report models, 517
- report creation, 512–517
- Report Server Projects, 517–523
- requirements for useful reports, 503
- rights for report design and execution, 517
- resizing databases, logical filename for, 78**
- RESTORE command**
  - example using, 732–733
  - options, 731
  - syntax, 730–731
- restoring. See recovery**
- RETURN statement**
  - ALTER PROC example, 290
  - CREATE PROC example, 288–289
  - sproc unconditionally exited by, 288
  - syntax, 288
- return values for sprocs**
  - altering, 290
  - capturing with EXEC command, 289
  - as integers, 288
  - RETURN statement for, 288–290
  - uses for, 288
- reusability**
  - database candidates for, 177–178
  - designing databases for, 177–179
  - logical versus physical modeling for, 178
  - performance impacts of, 179
- REVERSE function, 801**
- REVOKE statement, 653–654**
- RID (row ID)**
  - for a heap, 201, 203
  - locking, 346
- RIGHT function, 802**
- RIGHT OUTER JOINS, 43**
- rights**
  - for EXEC commands in sprocs, 264
  - granting for sprocs, 264
  - user, 647–655
- RMO (Replication Management Objects), 605–606**
- ROLAP (Relational OLAP), 831**
- roles. See also application roles; specific roles**
  - with CREATE authority as default, 75
  - defined, 10, 656
  - fixed database, 656, 658–659
  - overview, 656
  - ownership of objects created by, 76
  - permissions granted to, 75
  - server, 656, 657–658
  - user-defined database, 659–661
- ROLLBACK TRAN statement**
  - described, 331
  - example using, 332–333, 334–335
  - with nested triggers, 376
  - overview, 331
  - syntax, 331
- @@TRANCOUNT decremented to 0 by, 252
- within triggers, avoiding, 389–390
- root node of B-Trees**
  - page splits at, 198
  - pointing at non-leaf level nodes, 196
  - pointing directly at data, 195–196
- ROUND function, 780**
- row ID (RID)**
  - for a heap, 201, 203
  - locking, 346
- @@ROWCOUNT function**
  - EXEC scope example, 264
  - moving value to local variable, 253
  - overview, 251, 768
  - using in scripts, 252–253
- ROWCOUNT\_BIG function, 809**
- ROWGUIDCOL, identity columns compared to, 85–86**
- ROWLOCK optimizer hint, 350**
- rows**
  - contained by pages, 191
  - counting in a query, 54–56
  - default value for, 84
  - as entity data, 5
  - extending the size limit, 193–194
  - locking, 346
  - maximum columns per, 193
  - maximum size of, 193–194
  - primer, for foreign keys, 110, 111, 112
  - @@ROWCOUNT returning number affected by last statement, 251
  - in storage hierarchy, 193–194
- rowset functions, 794–796**
- rowversion data type, 12**
- RTRIM function, 802**
- rules**
  - binding and unbinding, 127
  - CHECK constraints versus, 126
  - constraints versus, 10
  - creating, 126–127
  - as domain constraints, 103
  - dropping, 128
  - ignored by BCP, 531
  - as legacy items, 10, 126
  - in logical model, 165–166, 167
  - other data integrity methods compared to, 129–131
  - overview, 10, 126–128
  - viewing dependencies for, 129
  - viewing with sp\_helptext, 126–127
- running. See opening or running**
- runtime errors. See also error handling**
  - in batches, 253, 255
  - higher-level, with SQL Server, 291, 296–297
  - in sprocs, types of, 290

**S****sa role. See also sysadmin role**

- as alias for dbo, 76
- CREATE authority as default for, 75
- creating sysadmin role member using, 639
- as legacy, 657
- limiting use of, 639, 665
- SQL Server Authentication for, 639

**SAFE permission set, 398, 417****SAVE TRAN statement**

- described, 332
- example using, 332–333, 334
- overview, 331–332
- syntax, 332

**scalability**

- partitioning for, 180–181, 244
- as performance issue, 670
- subcategory benefits for, 176–177

**scalar UDFs**

- assembly-based, creating, 400–403
- datetime field example, 309–310
- inline use of, 309
- T-SQL-based, creating, 308–310

**scan count statistics, 691****scheduling jobs**

- for backing up, 725
- branching rules for, 700
- creating jobs and tasks using Management Studio, 704–712
- creating jobs and tasks using T-SQL, 712–721
- creating operators using Management Studio, 701–702
- creating operators using T-SQL, 702–704
- deleting jobs and tasks using Management Studio, 721
- deleting jobs using T-SQL, 722
- editing jobs using Management Studio, 721
- editing jobs using T-SQL, 722
- jobs, defined, 700
- notification of success or failure, 700, 702, 704, 706
- overview, 700
- tasks, defined, 700
- tasks stored in msdb database, 4

**schema binding, 241, 242–243, 317****schema collections. See XML schema collections****schema modification locks (Sch-M), 348, 349****schema stability locks (Sch-S), 348, 349****SCHEMA\_ID function, 792****SCHEMA\_NAME function, 792****schemas. See also XML schema collections**

- avoiding, 6
- changes during replication, 567–568
- changing default, caveats for, 76
- consistency in use of, 74
- default (dbo), 75–76
- as foundation for all else, 281

- naming, 74–76
- for object references, 6
- overview, 6
- as owners, 74

**scope**

- of cursors, 428–432
- of database roles, 658
- of @@ERROR versus @Error, 293
- of EXEC command, 262–264

**SCOPE\_IDENTITY function, 809****Script Tasks (SSIS), 549****scripts. See also batches**

- CASE statements in, 270–275
  - common parameter-less system functions for, 251–253
  - control-of-flow statements, 265–280
  - for creating table if not already existing, 267, 268, 269–270
  - DECLARE statement in, 249
  - defined, 61
  - establishing precedence with batches, 256–258
  - EXEC command for dynamic, 260–265
  - @@IDENTITY function in, 252
  - IF . . . ELSE statements in, 266–270
  - for INSERT INTO . . . SELECT statement, 61–62
  - for OPENXML function, 496–497
  - for recursive spoc, 305–307
  - @@ROWCOUNT function in, 252–253
  - running with SQLCMD, 259–260
  - separating into batches, 253–254
  - setting variable values using SELECT, 250–251
  - setting variable values using SET, 250, 251
  - simple example, 248
  - SMO, 752–753
  - statements as parts of, 247
  - stored as text files, 248
  - tools for writing, 248
  - TRY/CATCH blocks in, 277–280
  - unified goal characteristic of, 247
  - USE statement in, 248–249
  - variables in, 62
  - WAITFOR statements in, 277
  - WHILE statements in, 275–277
- scrollability of cursors, 432–434**
- SCROLL LOCKS option for cursors, 447–450**
- searched CASE statement**
- examples, 272–275
  - syntax, 270–271, 805
- second normal form (2NF), 157**
- secondary filegroups, 7**
- secondary files, 7**
- security. See also passwords; roles**
- application roles, 661–664
  - asymmetric keys, 642, 666–667
  - avoiding global user accounts, 634–635
  - carte blanche access, limiting, 635

### security (continued)

- certificates, 642, 666–667
- database for examples, 634
- disabling triggers and issues for, 378
- enabling NetLibs, cautions for, 22
- encrypting triggers for, 363
- encrypting views for, 239–241
- EXEC command security context, 262, 264
- exposing server to the Internet, avoiding, 22
- with file storage for BLOBs, 169
- groups (pre-SQL Server 7.0), 655–656
- guest account issues, 664
- for HTTP endpoints, 498
- limiting CREATE authority access for, 75
- loading NorthwindSecure.sql script for examples, 634
- number of tries to log in, 637
- one person, one login, one password principle, 634–635
- sa role issues, 639, 665
- sprocs as tools for, 665
- sprocs for, 301–302
- system functions, 796–798
- TCP/IP port settings, 664–665
- total, not possible, 633
- UDFs as tools for, 665
- user rights, 647–655
- variety of approaches to, 633–634
- views as tools for, 665
- xp\_cmdshell issues, 665

**securityadmin role, 657**

**SEEK lookup, 200**

**Select Case statement (Visual Basic). See CASE statement**

**SELECT statement. See also specific clauses and keywords**

- asterisk (\*) in selection criteria, avoiding, 38
- basic example, 36
- correlated subqueries in select list, 142–144
- eliminating duplicate data, 57–59
- inline CASE statements with, 271–275
- nested subqueries returning multiple values, 137–138
- nested subqueries returning single values, 136–137
- nested subquery syntax, 135
- nested, to find orphaned records, 138
- optimizer hints in, 351–352
- overview, 36–37
- for setting variables scripts, 250–251
- specifying current database for, 37
- specifying tables for, 37
- syntax, 36, 52–53
- for tables returned by UDFs, 311–315
- user rights for, 649

**selectivity within indexes, 218–219**

**self-referencing INNER JOINS, 40, 41**

**self-referencing tables, 110–112**

### Send Mail task (SSIS), 550

#### sensitivity of cursors

- dynamic cursors, 435, 442, 445
- keyset-driven cursors, 441
- overview, 435
- static cursors, 435

### Sequence Container task (SSIS), 549

#### SERIALIZABLE isolation level

- non-repeatable reads prevented by, 344
- overview, 355
- phantoms prevented by, 345

#### SERIALIZABLE/HOLDLOCK optimizer hint, 349, 355

#### server roles, 656–657. See also specific roles

#### serveradmin role, 657

#### @@SERVERNAME function, 252, 768–769

#### SERVERPROPERTY function, 809–811

#### server-side versus client-side processing, 672–673

#### Service Broker, 825, 841–842

#### Service Master Key, 666

#### @@SERVICENAME function, 769

#### SESSIONPROPERTY function, 811–812

#### SESSION\_USER function, 811

#### SET clause

- ALTER DATABASE statement, 92
- with expressions, for UPDATE statement, 64

#### SET DEFAULT, cascading, 116

#### SET NULL, cascading, 116

#### SET QUOTED\_IDENTIFIER option, 16

#### SET statement

- for IMPLICIT\_TRANSACTIONS option, 341
- for isolation levels, 353
- separating the query from, 250
- for setting variables scripts, 250, 251

#### setupadmin role, 657

#### severity of errors. See error levels

#### SGAM (Shared Global Allocation Map), 192

#### shared locks

- compatibility with other lock modes, 349
- dirty reads prevented by, 347
- with intent exclusive, 348
- intent shared, 348

#### Shared Memory, 22, 24. See also NetLibs (network libraries)

#### ShortDept.fmt file, 540

#### ShortDept.txt file, 540

#### ShortDeptx.fmt file, 540

#### SHOWPLAN

- ALL option, 687–688, 689
- TEXT option, 687, 688–689

#### SIGN function, 780–781

#### simple CASE statement

- examples, 271–272
- syntax, 270, 805

#### Simple Object Access Protocol (SOAP), 498

#### Simple recovery model, 728, 729

#### SIN function, 781

- single quotes (') within strings, 263**
- SIZE parameter (CREATE DATABASE), 78**
- smalldate data type, 12**
- smallint data type, 11**
- smallmoney data type, 11**
- SMO (SQL Management Objects)**
  - backing up a database, 750–751
  - complete form code, 753–758
  - connection and server references, 745
  - creating a database, 745–746
  - creating a Windows Application project for, 744–745
  - creating logins with, 644
  - creating tables, 746–750
  - declarations, 745
  - defined, 739
  - dropping a database, 750
  - history of, 740–742
  - object model, 742–743
  - overview, 758–759
  - scripting, 752–753
- Snapshot Agent, 572, 598**
- snapshot folder, 590**
- snapshot replication**
  - defined, 571
  - Distribution Agent for, 572
  - ease of, 571
  - mixing with other types, 581
  - planning requirements, 574
  - process of, 572–573
  - Snapshot Agent for, 572, 598
  - transactional replication versus, 577
  - uses for, 573–574
- SOAP (Simple Object Access Protocol), 498**
- Solution Explorer (SSIS), 552**
- SOME operator, 48, 139**
- sort order. See also ORDER BY clause**
  - COLLATE parameter for setting, 80
  - CREATE INDEX options for, 209–210
  - for indexes, 194–195, 209–210
- SOUNDEX function, 802**
- SPACE function, 802**
- spaces embedded in names, avoiding, 16, 83**
- sp\_addapprole sproc, 662**
- sp\_add\_job sproc, 712, 713–714, 722**
- sp\_add\_jobschedule sproc**
  - described, 712, 719
  - @freq\_interval parameter, 719–720
  - @freq\_recurrence\_factor parameter, 721
  - @freq\_relative\_interval parameter, 721
  - @freq\_subday\_interval parameter, 720
  - @freq\_subday\_type parameter, 720
  - @freq\_type parameter, 719–720
  - related sprocs, 722
  - syntax, 719
  - using, 721
- sp\_add\_jobserver sproc, 714–715, 722**
- sp\_add\_jobstep sproc**
  - @cmexec\_success\_code parameter, 717
  - @command parameter, 716–717
  - @database\_name parameter, 717
  - described, 712
  - @flags parameter, 718
  - @job\_id parameter, 716
  - @job\_name parameter, 716
  - @on\_fail\_action parameter, 717
  - @on\_fail\_step\_id parameter, 717
  - @on\_success\_action parameter, 717
  - @on\_success\_step\_id parameter, 717
  - @os\_run\_priority parameter, 718
  - related sprocs, 722
  - @retry\_interval parameter, 717
  - @server parameter, 717
  - @step\_id parameter, 716
  - @step\_name parameter, 716
  - @subsystem parameter, 716
  - syntax, 715
- sp\_addlogin sproc**
  - deprecated by CREATE LOGIN, 640
  - described, 644, 645
  - parameters, 645–646
  - syntax, 645
- sp\_addmessage sproc**
  - for custom error messages, 299–301
  - @lang parameter, 300
  - message added to master database sysmessages table, 301
  - migrating messages to new server, 301
  - @replace parameter, 300
  - syntax, 299
  - using, 300
  - @with\_log parameter, 300
- sp\_add\_notification sproc, 704**
- sp\_add\_operator sproc, 702–704**
- sp\_addrole sproc, 659**
- sp\_addrolemember sproc, 660**
- sp\_addserver sproc, 252**
- sp\_addumpdevice sproc, 725**
- sp\_attach\_db sproc, 80**
- sp\_bindefault sproc, 128**
- sp\_bindrule sproc, 127**
- spCLRExample assembly-based sproc. See also ExampleProc assembly**
  - creating from ExampleProc assembly, 398–399
  - test call to, 399–400
- sp\_configure sproc**
  - enabling CLR, 394
  - setting query governor, 692
  - setting split point with, 150
  - turning on nested triggers, 376

## **spCursorScope sproc**

- global scope example, 428–431
- local scope example, 431–432

## **spCursorScroll sproc, 433–434**

## **sp\_dboption sproc**

- checking database recovery, 732–733
- setting cursor default to local, 429

## **sp\_delete\_job sproc, 722**

## **sp\_delete\_jobschedule sproc, 722**

## **sp\_delete\_jobserver sproc, 722**

## **sp\_delete\_jobstep sproc, 722**

## **sp\_delete\_operator sproc, 704**

## **sp\_depends sproc, 129**

## **sp\_detach\_db sproc, 80**

## **sp\_dropapprole sproc, 663–664**

## **sp\_dropmessage sproc, 301**

## **sp\_droprole sproc, 661**

## **sp\_droprolemember sproc, 660–661**

## **spEmployee sproc**

- creating, 282–283
- declaring parameters, 284–285

## **spFactorial sproc, 305–306**

## **sp\_fulltext\_catalog sproc**

- @action argument, 620
- example using, 620
- @ftcat argument, 619
- @path argument, 620
- syntax, 619

## **sp\_fulltext\_database sproc, 610–611**

## **sp\_fulltext\_table sproc**

- @action argument, 621–622, 623
- @ftcat argument, 621
- @keyname argument, 621
- options for populating indexes, 622, 623
- syntax, 621
- @tablename argument, 621

## **sp\_grantdbaccess sproc, 648**

## **sp\_grantlogin sproc, 646**

## **sp\_help sproc**

- for verifying table creation, 89
- for viewing existing table settings, 93

## **sp\_helpconstraint sproc**

- clustered index indication from, 109
- for constraint information, 109
- with self-referencing tables, 110
- verifying UNIQUE constraints, 118

## **sp\_helppdb sproc, 81–82, 89–90**

## **sp\_help\_operator sproc, 704**

## **sp\_helptext sproc**

- for displaying code for views, 238–239
- encrypted views not displayed by, 240
- for viewing rules, 126–127

## **@@SPID function, 769**

## **spInsertValidatedStoreContact sproc, 293–294**

## **split point, 150**

## **sp\_password sproc, 646**

**sprocs (stored procedures).** See also *Debugger; specific procedures*

advantages of, 9

basic example, 282–283

changing with ALTER statement, 283

for creating callable processes, 301

for creating cursors, 428–432

creating from assemblies, 395–400

creating output parameters for, 285–288

custom error messages for, 299–301

declaring parameters, 284–285

DML triggers versus, 362

dropping, 283

error handling, 290–301

extended (XPs), 304–305

granting rights for, 264

incorrect optimization for, 303–304

keeping transactions short, 675

nesting (calling sprocs within), 301

OUTPUT keyword in declaration, 287

overview, 9

parameterization, 284–288

performance benefits of, 284, 302–303

performance tuning, 675–677

RAISERROR command with, 296–299

recursive, 305–307

RETURN statement with, 288–290

return values, 288–290

running in EXEC procedures, 265

for security, 301–302

security context of EXEC commands in, 264

as security tools, 665

stored in sysdatabases table, 3

for subcategory implementation, 174

syntax for creating, 282

syntax for parameter declarations, 284

UDFs versus, 9

WITH RECOMPILE option, 303, 304

## **sp\_setapprole sproc, 662–663**

## **sp\_settriggerorder sproc, 378**

## **spTestReturns sproc, 289–290**

## **spTriangular sproc**

creating, 306–307

debugging, 322–327

opening in Debugger, 318–320

## **sp\_unbindefault sproc, 128**

## **sp\_unbindrule sproc, 127**

## **sp\_update\_job sproc, 722**

## **sp\_update\_jobschedule sproc, 722**

## **sp\_update\_jobstep sproc, 722**

## **sp\_update\_operator sproc, 704**

## **sp\_xml\_preparedocument sproc, 493**

`sp_xml_removedocument` sproc, 496

**SQL Management Objects. See SMO**

**SQL NS (SQL Namespaces), 740–741**

**SQL Server Agent**

- Configuration Manager for service management, 19
- configuring for replication, 589–590, 598
- for SSIS package execution, 560
- tasks stored in `msdb` database, 4

**SQL Server Authentication**

- advantages of, 27, 639
- altering logins, 642–643
- creating logins with `CREATE LOGIN`, 640–642
- creating logins with Management Studio, 643–644
- creating logins with SMO, 644
- creating logins with `sp_addlogin`, 640, 644, 645–646
- creating logins with `sp_grantlogin`, 646
- creating logins with SQL-DMO, 645
- creating logins with WMI, 645
- described, 27, 639
- disadvantages of, 639
- dropping logins, 643
- requirements for using, 26
- for `sa` account, 639

**SQL Server Books Online (BOL), 18–19**

**SQL Server Browser, 19**

**SQL Server Business Intelligence Studio. See Business Intelligence Studio**

**SQL Server Configuration Manager. See Configuration Manager**

**SQL Server Integration Services. See SSIS**

**SQL Server Management Studio. See Management Studio**

**SQL Server Profiler**

- capturing traces, 693–695
- Column Filters, 695–696
- importance of, 696
- opening, 693
- overview, 33–34
- Performance Monitor versus, 33
- selecting information to collect, 694–695
- templates, 694
- for troubleshooting performance, 693–696

**SQLCMD**

- case-sensitivity of flags, 259
- described, 259
- including password with, 259
- older tools replaced by, 34, 247, 259
- overview, 34
- running a query with, 259
- running a script with, 259–260
- syntax for running, 259

**sql\_variant data type, 13**

**SQL\_VARIANT\_PROPERTY function, 792–793**

**SQRT function, 781**

**square brackets ([ ]) for names, 16**

**SQUARE function, 781**

**SSIS (SQL Server Integration Services)**

- building a simple package, 552–559
- Configuration Manager for service management, 19
- connections for packages, 556–557, 559
- Control Flow tab, 551
- creating a project, 546–548
- Data Flow tab, 551
- as DTS successor, 545
- Event Handlers tab, 551–552
- events (table), 551–552
- executing packages, 560–563
- Main window, 546–548, 551–552
- opening from Business Intelligence Studio, 546
- overview, 32–33
- Package Explorer tab, 552
- packages, defined, 546
- Precedence Constraint Editor, 555–556
- Properties window, 552
- reorganizing tasks, 548
- Solution Explorer, 552
- tasks, defined, 546, 548
- tasks (table summarizing), 548–551
- transformations, 32–33

**START FULL POPULATION parameter (ALTER FULLTEXT INDEX), 618**

**START INCREMENTAL POPULATION parameter (ALTER FULLTEXT INDEX), 618**

**START UPDATE POPULATION parameter (ALTER FULLTEXT INDEX), 619**

**starting. See opening or running**

**statement-level permissions, 654–655**

**static cursors**

- client-side implementation, 435, 673
- `CursorTest` example, 436–437
- defined, 435
- `FAST_FORWARD` cursor conversion to, 446
- sensitivity, 435
- using temporary tables instead of, 435, 677

**statistics**

- client, 692
- `STATISTICS IO`, 690–691
- `STATISTICS TIME`, 692
- for troubleshooting performance, 690–692
- `WHILE` statement for updating, 276–277

**STATS\_DATE function, 812**

**STDEV function, 772–773**

**STDEV function, 773**

**STOP parameter (ALTER FULLTEXT INDEX), 619**

**storage hierarchy**

- database as highest level of, 189
- extents, 190–191
- full-text catalogs, 194
- pages, 191–193

### storage hierarchy (continued)

- physical database file, 190
  - rows, 193–194
  - transaction log, 190
- storage types for Analysis Services, 830–831**
- stored procedures. See spprocs**
- StoredProcedures class, 396**
- STR function, 802**
- strings**
- concatenation performed before calling EXEC, 262
  - quotation marks within, 263
  - system functions, 798–803

### structure in logical model, 167

#### STUFF function, 802–803

#### subcategories

- bottlenecks from, 177
- defined, 171
- domain tables for, 174
- ERDs for, 172–173, 174–175, 176
- example for documents, 171–172
- exclusive versus non-exclusive, 172–173
- extensibility provided by, 176–177
- logical model for, 173–175
- performance improved by, 176
- physical model for, 175–176
- in physical models, 171
- sprocs for implementing, 174
- views for implementing, 174

#### subqueries. *See also specific kinds*

- correlated, 139–144
- defined, 134
- joins versus, 134–135, 152–153
- nested, 135–139, 146–147
- uses for, 134

#### subscribers (replication)

- anonymous, 570
- defined, 569
- global, 570
- immediate-update, 578, 580–581
- local, 570
- NOT FOR REPLICATION clause with, 588
- preparing for transactional replication, 577–578
- setting up using Management Studio, 598–603
- topology models, 583–586

#### subscriptions (replication)

- pull, 569, 570
- push, 569, 570

#### SUBSTRING function, 803

#### SubType relationships. *See subcategories*

#### SUM function

- with GROUP BY clause, 53–54
- overview, 773

#### Supertype relationships. *See subcategories*

#### SUSER\_ID function, 797

#### SUSER\_NAME function, 797

#### SUSER\_SID function, 798

#### SUSER\_SNAME function, 798

#### Switch statement (C, C++, C#, Delphi). *See CASE statement*

#### switches. *See flags and switches* “sys” functions, 37

#### sysadmin role. *See also sa role*

- as alias for dbo, 76
- CREATE authority as default for, 75
- overview, 657
- Windows Administrators group mapped into, 657

#### syscomments table, 239

#### sysdatabases table, 3

#### sys.indexes function, 203

#### sysindexes table, 203

#### sys.messages function, 279–280

#### system databases, 3–4. *See also specific databases*

#### system functions. *See also specific functions*

- aggregate, 771–773
- categories of T-SQL functions, 761
- cursor, 773–774
- date and time, 774–777
- EXEC command scope with, 264
- legacy (global variables), 762–770
- mathematical, 777–781
- metadata, 781–794
- moving values into variables, 249
- parameter-less (table), 251–252
- rowset, 794–796
- security, 796–798
- string, 798–803
- system, 803–812
- text and image, 812–813
- using instead of system tables, 239

#### system tables, 3, 239. *See also specific tables*

#### SYSTEM\_USER function, 812

## T

#### table data type, 13

#### table scans, 199

#### TableCursor example

- adding ALTER INDEX functionality, 425–427
- closing, 424
- code listing, 424–425
- deallocating, 424
- declaring, 423
- FETCH statement for, 424
- @FETCH\_STATUS variable for, 424
- opening, 423–424

#### tables. *See also ALTER TABLE statement; CREATE*

#### TABLE statement; *specific kinds*

- adding existing tables to diagram, 182, 183
- adding new, with diagramming tools, 183–184
- aliases for, 37, 40

- assembly-based UDF for validating e-mail fields in, 400–403
- FROM clause for specifying, 36
- client- versus server-side processing, 673
- creating using Management Studio, 97–99
- creating using SMO, 746–750
- derived, 144–146
- domain data versus entity data in, 5
- dropping with diagramming tools, 184
- dropping with DROP statement, 95
- editing using Management Studio, 99
- identifying at runtime using EXEC, 261–262
- locking, 346
- making views look like, 241
- moving columns using Management Studio, 99
- moving columns using T-SQL, problems with, 94
- naming, 83–84
- overview, 5–6
- partitioned, 180–181
- self-referencing, 110–112
- system, 3
- testing existence before creating, 147–148
- UDFs returning, 310–316
- using rule or default, determining, 129
- viewing dependencies for, 129
- table-valued functions**
  - creating from assemblies, 403–407
  - IEnumerable interface for, 404
  - uses for, 403
- TABLOCK optimizer hint, 350, 534, 541**
- TABLOCKX optimizer hint, 350**
- TAN function, 781**
- tasks (job). See also scheduling jobs**
  - creating using Management Studio, 704–712
  - creating using T-SQL, 712–721
  - defined, 700
  - deleting using Management Studio, 721
  - stored in msdb database, 4
- TCP/IP. See also NetLibs (network libraries)**
  - default port for IP NetLib, 21
  - exposing server to the Internet, avoiding, 22
  - overview, 22
  - port settings, 664–665
  - Shared Memory versus, for local servers, 24
- tempdb database**
  - creating objects directly in, avoiding, 4
  - forcing index to sort in, 212–213
  - overview, 4
- temporary tables**
  - for EXEC procedures, 264
  - for performance tuning, 677
  - using instead of static cursors, 435, 677
- terminators**
  - for BCP import, 531
  - in ERDs, 162–165
- text data type**
  - for BLOBs requiring FTS, 170
  - overview, 12
- text editors for script writing, 248**
- text system functions, 812–813**
- TEXTIMAGE\_ON clause (CREATE TABLE), 88**
- TEXTPTR function, 812**
- @@TEXTSIZE function, 769**
- TEXTVALID function, 813**
- THEN clause**
  - in searched CASE statements, 270–271, 272–275
  - in simple CASE statements, 270, 271–272
- third normal form (3NF), 155, 157**
- 32-level limit for recursion, 306–307**
- time data type, 12**
- TIME parameter (WAITFOR), 277**
- timestamp columns**
  - BCP with, 531
  - replication with, 587
- timestamp data type, 12**
- @@TIMETICKS function, 769**
- tinyint data type, 11**
- TO clause (GRANT), 650**
- topology models for replication**
  - central publisher/distributor, 582
  - central publisher/remote distributor, 582–583
  - central subscriber, 583–584
  - mixed models, 584–586
  - multiple subscribers/multiple publishers, 586
  - publisher/subscriber, 585
  - publishing subscriber, 584–585
  - self-publishing, 586
  - simple models, 581–583
- @@TOTAL\_ERRORS function, 769**
- @@TOTAL\_READ function, 769**
- @@TOTAL\_WRITE function, 769**
- @@TRANCOUNT function, 252, 770**
- transaction context of EXEC command, 262**
- transaction log**
  - active portion, defined, 337
  - backing up, 724, 728
  - BCP use of, 534
  - changes propagated at checkpoints, 5, 336
  - Checkpoint on Recovery option, 337–338
  - checkpoints, 337–339
  - circumstances issuing checkpoints, 337
  - dirty pages, 336
  - failure and recovery, 339–340
  - issuing checkpoints manually, 337
  - .ldf extension for, 78, 190
  - overview, 5, 336–337
  - RAID and integrity of, 682
  - restoring, 730–732
  - as serial file, 5
  - in storage hierarchy, 190

## **transactional consistency, 567**

### **transactional replication**

- defined, 577
- with immediate-updating subscribers, 578
- Log Reader Agent for, 578, 598
- mixing with other types, 581
- overview, 577
- planning requirements, 580
- preparing subscribers for, 577–578
- process of, 578–579
- snapshot replication versus, 577
- testing, 604–605
- unlogged bulk operations not replicated by, 577
- uses for, 579

### **transactions. See also isolation levels; locking;**

#### **transaction log**

- ACID test for, 359
- atomicity of, 329–330
- beginning, 331
- committing, 331
- creating savepoints for, 331–332
- example using `TRAN` commands, 332–335
- implicit, 340–341
- keeping as short as possible, 357, 675
- marking definite begin and end points for, 330
- open-ended, not allowing, 358
- rolling back, 331
- `TRAN` commands for, 330–332
- `@@TRANCOUNT` returning active number of, 252

### **Transact-SQL. See T-SQL**

### **Transfer Tasks (SSIS), 550**

### **transformations, 32–33**

#### **trgExampleTrigger trigger**

- creating from `ExampleTrigger` assembly, 415
- creating table for testing, 414
- testing for delete action, 416–417
- testing for insert action, 415
- testing for update action, 416

### **triangular**

- defined, 306
- `spTriangular` sproc for computing, 306–307

### **triggers. See also INSTEAD OF triggers**

- architecture changes' impact on, 377
- BCP with, 531, 534
- CHECK constraints versus, 362, 367–371
- for checking the delta of an update, 369–371
- `COLUMNS_UPDATED()` function with, 386–388
- concurrency issues not present for process firing, 389
- creating from assemblies, 412–417
- for custom error messages, 371
- for data integrity implementation, 129, 367–371
- DDL, 362
- debugging, 376–377, 390–392
- defined, 6, 362

- DELETED tables for, 366, 370
- disabling on replication subscribers, 588
- DML versus DDL, 362
- dropping, 390
- encrypting, 363
- events for, 363
- extremism, avoiding, 362
- for feeding data into de-normalized tables, 372–373
- firing order for, 378–380
- FOR | AFTER versus INSTEAD OF, 364–365, 380
- foreign key performance versus, 179
- INSERTED tables for, 366, 370
- keeping as short as possible, 389
- logged versus unlogged activities and, 363
- nested, 376
- not firing for replication-related tasks, 367
- optimizing indexes for, 389
- other data integrity methods compared to, 129–131
- performance considerations, 388–390
- as reactive rather than proactive, 388
- recursive, 376
- for requirements sourced from other tables, 367–369
- reusability considerations for, 178
- rolling back within, avoiding, 389–390
- for setting condition flags, 373–375
- for 6.5 compatibility mode, 366
- syntax for creating, 363
- turning off without removing, 377–378
- `UPDATE()` function with, 386
- for updating summary information, 372
- uses for, 362

### **troubleshooting performance. See also performance;**

#### **performance tuning**

- client statistics for, 692
- DBCC for, 692
- Performance Monitor for, 696–697
- query governor for, 692–693
- SHOWPLAN options for, 687–690
- SQL Server Profiler for, 693–696
- STATISTICS IO for, 690–691
- STATISTICS TIME for, 692
- tools for, 686–687

### **Truncate On Checkpoint option, 338**

#### **TRUSTWORTHY parameter**

- ALTER DATABASE statement, 406
- CREATE DATABASE statement, 80

#### **TRY/CATCH blocks**

- for avoiding script termination on errors, 293
- calling `uspLogError` sproc, 286–288
- CREATE script example, 278–279
- error condition retrieval functions, 279
- `@@ERROR` function versus, 295
- error levels, 278
- syntax, 278

trapping 2714 error, 295  
 using error condition retrieval, 286–288  
 viewing all system error messages, 279–280

**T-SQL (Transact-SQL). See also specific statements**

ANSI entry-level compliance of, 27, 36  
 backing up using, 726–728  
 CLR compliance, 35  
 creating backup device using, 725  
 creating jobs and tasks using, 712–721  
 creating operators using, 702–704  
 defined, 27  
 deleting jobs using, 722  
 editing jobs using, 722  
 portability issues, 36  
 restoring data using, 730–733

**1205 error. See deadlocks**

**2714 error, 295**

**TYPE COLUMN parameter (CREATE FULLTEXT INDEX), 615–616**

**TYPE\_WARNING option for cursors, 446, 450–452**

## U

**UDFs (user-defined functions). See also Debugger**

applying as CHECK constraints, 403  
 creating from assemblies, 400–403  
 datetime field example, 309–310  
 defined, 308  
 deterministic, 316–317  
 EXEC command not allowed in, 262, 265  
 file storage implementation using, 170  
 in indexed views, 242  
 inline use of, 309  
 overview, 9  
 returning a scalar value, 308–310  
 returning a table, 310–316  
 as security tools, 665  
 sprocs versus, 9  
 syntax for creating, 308  
 for validating e-mail fields in tables, 400–403

**unbinding**

defaults, 128  
 rules, 127

**underscore ( \_ )**

in table names, avoiding, 83  
 as wildcard, 48

**UNICODE function, 803**

**UNIONS**

compatibility required for columns, 70  
 default return option for, 70  
 DISTINCT clause with, 70  
 headings returned for, 70  
 JOINS versus, 69  
 overview, 69–72  
 rules for, 70

**UNIQUE constraints**

adding to existing table, 118  
 allowing NULLS, 117  
 BCP enforcement of, 531  
 as entity constraints, 104  
 IGNORE\_DUP\_KEY option with, 211  
 in logical model, 167  
 overview, 116–117  
 specifying in CREATE TABLE statement, 117–118

**uniqueidentifier columns, replication with, 587**

**uniqueidentifier data type, 12**

**UNSAFE permission set, 394, 398**

**UPDATE ( ) function, 386**

**update locks**

compatibility with other lock modes, 349  
 deadlocks prevented by, 347–348  
 overview, 347

**UPDATE statement**

cascading updates, 112  
 changing multiple columns, 64  
 DEFAULT constraints with, 120  
 overview, 62–64  
 for primary keys, avoiding, 64  
 SET clause with expressions for, 64  
 syntax, 62  
 update locks, 347–348  
 user rights for, 649  
 with views, 236–237

**UPDATE STATISTICS statement, 212**

**UPDATE triggers**

checking the delta of an update, 369–371  
 COLUMNS\_UPDATED ( ) function with, 386–388  
 CREATE TRIGGER parameter for, 366  
 creating from assemblies, 412–417  
 firing order for, 378  
 INSTEAD OF triggers, 384  
 for requirements sourced from other tables, 367–369  
 UPDATE ( ) function with, 386  
 for updating summary information, 372

**UPDLOCK optimizer hint, 351**

**UPPER function, 803**

**uppercase. See case and case sensitivity**

**USE statement**

in scripts, 248–249  
 specifying current database, 37

**USER function, 798**

**user rights**

categories of, 647  
 DELETE, 649  
 denying for targeted object, 652–653  
 EXECUTE, 649  
 granting access to a specific database, 647–648  
 granting object permissions, 648–654  
 granting statement-level permissions, 654–655  
 INSERT, 649

### **user rights (continued)**

- REFERENCES, 649
- revoking, 653–654
- SELECT, 649
- UPDATE, 649
- user-defined data types**
  - accessing, 418–419
  - caveats for using, 10
  - creating assembly for, 417
  - creating from assemblies, 417–419
  - dropping, 419
  - overview, 10
  - replication with, 587
  - rules restricting, 10
- user-defined database roles**
  - adding users to, 660
  - creating, 659
  - described, 656
  - dropping, 661
  - fixed database roles versus, 659
  - removing users from, 660–661
- user-defined functions. See UDFs**
- USER\_ID function, 798**
- USER\_NAME function, 812**
- users. See also roles**
  - adding to a database, 647–648
  - adding to user-defined database roles, 660
  - defined, 10
  - direct access for, 302
  - global accounts, avoiding for security reasons, 634–635
  - guest account, 664
  - one person, one login, one password principle, 634–635
  - removing from user-defined database roles, 660–661
  - rights, 647–655
  - storage of profiles, 637–638
  - termination by ALTER DATABASE statement, 92
- uspLogError spdoc**
  - code for, 285–286
  - output parameter in, 286
  - TRY/CATCH example calling, 286–288

## V

- validating e-mail fields in tables, 400–403**
- .value method (XML), 463, 464–465**
- VALUES keyword (INSERT), 59–60**
- VAR function, 773**
- varbinary data type, 13**
- varbinary(max) data type, 13, 168, 194**
- varchar data type, 12**
- varchar(max) data type**
  - for BLOBs, 168
  - for file names and paths, 169

- overview, 12
- replacing text data type, 12
- row size limit extended by, 194
- variables**
  - declaring in scripts, 62, 249
  - global (legacy system functions), 762–770
  - moving system function values into, 249
  - setting values in scripts using SELECT, 250–251
  - setting values in scripts using SET, 250, 251
- VARP function, 773**
- VB.NET connectivity example, 821–823**
- @VERSION function, 770**
- vertical filtering or partitioning, 570**
- Very Large Databases (VLDBs), filegroups for, 78**
- VIA (Virtual Interface Adapter), 20, 23. See also NetLibs (network libraries)**
- Vieira, Robert (Beginning SQL Server 2005 Programming), 17, 25, 44, 329, 457**
- VIEW\_METADATA option for views, 241**
- views**
  - added columns missing from, 94
  - changing data through, 236–237
  - complex examples, 233–236
  - Data Source Views, 507–508
  - defined, 231
  - displaying code for, 238–239
  - dropping, 238
  - editing with T-SQL, 237
  - encrypting, 239–241
  - indexed, 8–9, 242–244
  - inline, as derived tables, 144
  - INSTEAD OF DELETE triggers for, 384–385
  - INSTEAD OF INSERT triggers for, 381–383
  - INSTEAD OF UPDATE triggers for, 384
  - joined data with, 236, 237
  - making look like tables, 241
  - overview, 7–9, 244–245
  - partitioned, 180–181, 244
  - pass-through, 232
  - performance considerations, 179, 233
  - required fields with, 237
  - restricting insertions and updates in, 237
  - reusability considerations for, 178
  - schema binding for, 241, 242–243
  - as security tools, 665
  - simple example, 232
  - as stored queries, 231
  - for subcategory implementation, 174
  - supporting client-side cursors, 241
  - tips for using, 244–245
  - uses for, 7, 231, 244
  - VIEW\_METADATA option for, 241
  - WHERE clause with, 234
  - WITH CHECK OPTION for, 237

**Virtual Interface Adapter (VIA), 20, 23. See also NetLibs (network libraries)**

**Visio diagramming tool, 159**

**Visual Studio**

- adding a sproc to assembly project, 395
- creating a project for assemblies, 395, 404, 408, 412
- setting up for Business Intelligence Studio, 505

**VLDBs (Very Large Databases), filegroups for, 78**

## W

**WAITFOR statement, 276–277**

**Watch window (Debugger), 322**

**Web Service Task (SSIS), 550**

**WEIGHT keyword (FTS), 630–631**

**WHEN clause**

- in searched CASE statements, 270–271, 272–275
- in simple CASE statements, 270, 271–272

**WHERE clause**

- correlated subqueries in, 140–142
- in DELETE statement, 65
- EXISTS operator in, 199
- with JOINS, 48–49
- nested subquery syntax, 135
- operators (table), 47–48
- overview, 45–49
- placed before ORDER BY clause, 50
- for tables returned by UDFs, 311, 312, 313–314, 315
- using table as filtering source, 49
- with views, 234

**WHILE statement**

- BEGIN . . . END blocks with, 276
- BREAK statement with, 276
- CONTINUE statement with, 276
- described, 275
- example for updating statistics, 276–277
- syntax, 276
- for tables returned by UDFs, 314

**wildcards for LIKE operator, 48**

**windowing not supported by .NET assemblies, 394**

**Windows authentication**

- advantages and disadvantages of, 26–27
- described, 26, 639
- overview, 646–647

**Windows domain, domain listing versus, 45**

**Windows Management Instrumentation (WMI), 550, 645, 741**

**WITH ACCENT\_SENSITIVITY parameter (CREATE FULLTEXT CATALOG), 612**

**WITH APPEND parameter (CREATE TRIGGER), 366**

**WITH CHANGE\_TRACKING parameter (CREATE FULLTEXT INDEX), 616–617**

**WITH CHECK OPTION for views, 237**

**WITH clause**

- CREATE INDEX statement, 210
- CREATE LOGIN statement, 640–641
- OPENXML function schema declaration, 494–495
- RAISERROR command, 299

**WITH DB CHAINING parameter (CREATE DATABASE), 80**

**WITH ENCRYPTION option**

- ALTER TRIGGER statement, 363
- ALTER VIEW statement, 240
- CREATE TRIGGER statement, 363

**WITH GRANT OPTION (GRANT), 650**

**WITH NO POPULATION parameter (CREATE FULLTEXT INDEX), 616–617**

**WITH NOCHECK option for constraints, 122–124**

**WITH PERMISSION\_SET options (CREATE ASSEMBLY), 398**

**WITH RECOMPILE option for sprocs, 303, 304**

**WITH SCHEMABINDING option**

- ALTER FUNCTION statement, 317
- ALTER VIEW statement, 242–243
- CREATE VIEW statement, 241
- for DayOnly() UDF, 317
- required for indexed views, 242

**WITH XMLNAMESPACES() declaration**

- overview, 464
- for XML .exist method, 468
- for XML .modify method, 466
- for XML .nodes method, 467
- for XML .query method, 464
- for XML .value method, 465

**WMI Tasks (SSIS), 550**

**WMI (Windows Management Instrumentation), 550, 645, 741**

## X

**XLOCK optimizer hint, 351**

**XML data type**

- defining a column as, 458–459
- enforcing constraints, 458, 469
- .exist method, 463, 468–469
- methods available, 463
- .modify method, 463, 465–466
- .nodes method, 463, 466–468
- overview, 13, 458
- .query method, 463–464
- schema collections, 458, 460–463
- .value method, 463, 464–465

**XML (Extensible Markup Language). See also FOR XML clause; XML data type**

- format files, 537
- further information, 458
- HTTP endpoints for, 497–500

## XML (continued)

---

### **XML (continued)**

- importance of understanding, 457
- indexes, 214–215, 497
- naming every column when using, 471
- non-typed, 459
- OPENXML function, 493–497
- schema collections, 458, 460–463

### **XML indexes, 214–215, 497**

#### **XML schema collections**

- altering, 462
- creating, 461–462
- defined, 460
- described, 458
- dropping, 463
- enforcing constraints beyond, 469
- XML\_SCHEMA\_NAMESPACE() function for, 460–461

### **XML Tasks (SSIS), 551**

- XML\_SCHEMA\_NAMESPACE() function, 460–461

- xp\_cmdshell, limiting access to, 665

- XPs (extended sprocs), 304–305

## **Y**

- YEAR function, 776–777