



Index

Symbol

@@error function, 391

A

ABP. *See* adjacent broker protocol (ABP)

ACID (Atomicity, Consistency, Isolation, and Durability), 361

adjacent broker protocol (ABP)

certificate authentication, 453–455

connections for endpoints, 451

overview, 449–450

Windows authentication, 452–453

AFTER triggers

auditing example, 333–335

COLUMNS_UPDATED function,
329–332

compared to INSTEAD OF, 336

deleted tables, 316–318

identifying number of rows, 318–321

identifying type, 321–323

inserted tables, 316–318

nesting, 328–329

not firing for specific statements, 324–328

overview, 316

recursion, 328–329

UPDATE predicate, 329–332

arguable constructs, 111

arithmetic operations, 58–59

asymmetric key authentication in dialog

conversations, 444–445

asynchronous delivery in dialog

conversations, 414–415

asynchronous processing in Service Broker,

461

Atomicity, Consistency, Isolation, and Durability (ACID), 361

audience for this book, xxi

auditing example in AFTER triggers,
333–335

automatic handling of sequences, 342–344

B

BEGIN TRAN/COMMIT TRAN block, 362

beginning dialogs, 424–426

birthday problem, 8–10

BizTalk vs. Service Broker, 463

BROKER_INSTANCE parameter, 456

BULK rowset provider, 34–35

bulk-update (BU) locks, 365

C

C# .NET-based ComplexNumberCS UDA,
60–61

career phases of T-SQL programmers,
111–112

CASE expressions, 36

case-sensitive filters, 31

certificate authentication for endpoints,
453–455

character-related problems

case-sensitive filters, 31

CHECK constraints in pattern matching,
27–28

IP patterns, 29–30

LIKE predicate in pattern matching,
26–27

overview, 25

- character-related problems, *continued*
 - parsing character strings in pattern matching, 28
 - pattern matching overview, 26
- CHECK constraints
 - pattern matching, 27–28
 - scalar UDFs, 220
- CHECK OPTION, 204–205
- checkpoint process in transactions, 362
- CLR routines
 - benefits, 223
 - CLRUtilities database, 466
 - creating projects, 466–467
 - deploying, 467–468
 - deployment overview, 467
 - developing code, 467
 - developing solutions in Visual Studio, 466
 - overview, 465
 - risks, 223
 - testing using C# code, 468
 - testing using C# code without CREATE statements, 482
 - testing using Visual Basic code, 475
 - testing using Visual Basic code without CREATE statements, 486
- CLR scalar UDFs
 - CLR routines, 223
 - explicit vs. implicit conversions, 228–231
 - overview, 222–223
 - regular expressions, 224–228
- CLR split UDFs
 - C# definition, 243–244
 - compared to T-SQL performance, 247
 - creating Arrays table, 246–247
 - registering C# version, 245
 - registering Visual Basic version, 245
 - Visual Basic version, 244
- CLR SQL Signature UDFs
 - C#, 236
 - compared to T-SQL performance, 236–239
 - Visual Basic, 233–235
- CLR stored procedures
 - example 1, 305–309
 - example 2, 309–313
 - overview, 305
- CLR triggers
 - example, 351–353
 - example C# code, 353–356
 - example output, 359–360
 - example registering and attaching, 359
 - overview, 351
 - Visual Basic code, 356–359
- CLR user-defined types (UDTs)
 - arithmetic operations, 58–59
 - background, 41
 - C# .NET-based ComplexNumberCS UDA, 60–61
 - complex classes, 47
 - complex domains, 44–46
 - CREATE ASSEMBLY command, 56
 - CREATE TABLE command, 56
 - CREATE TYPE command, 56
 - creating, 51–55
 - deploying using T-SQL overview, 55
 - domains and classes, 44–45
 - domains and relations, 41–42
 - enabling CLR, 55
 - language for creating, 47–48
 - NULL number, 58
 - output of stored procedures, 273
 - overview, 40
 - programming overview, 48
 - relations and classes, 42–44

- requirements, 48–50
- SELECT statement, 57
- ToString method, 57
- Visual Basic .NET-based
 - ComplexNumberVB UDT, 61
- Visual Basic .NET-based
 - ComplexNumberVB_SUM UDA, 61–64
 - Write method, 58
- CLRUtilities database, 466
- code samples, downloading, xxii, 1
- COLUMNS_UPDATED function, 329–332
- common language runtime (CLR)
 - routines. *See* CLR routines
 - scalar UDFs. *See* CLR scalar UDFs
 - split UDFs. *See* CLR split UDFs
 - SQL Signature UDFs. *See* CLR SQL Signature UDFs
 - stored procedures. *See* CLR stored procedures
 - triggers. *See* CLR triggers
 - user-defined types. *See* CLR user-defined types (UDTs)
- compatibility of locks, 366
- concurrency in transactions, 361–362
- concurrency model, 370
- conflict detection in snapshot isolation level, 377–379
- contracts in dialog conversations, 417–418
- conversations
 - asymmetric key authentication, 444–445
 - beginning and ending dialogs, 424–426
 - configuring dialog security, 445–448
 - contracts, 417–418
 - DEFAULT message type, 418
 - definition, 412
 - dialog asynchronous delivery, 414–415
 - dialog order, 413–414
 - dialog overview, 411–412
 - dialog reliability, 413
 - dialog security overview, 443–444
 - dialog vs. monolog, 412
 - endpoints. *See* endpoints in Service Broker groups, 428–430
 - messages, 415–417
 - monolog vs. dialog, 412
 - poison messages, 442–443
 - queues, 418–422
 - receiving messages, 432–433
 - routing. *See* routing in Service Broker sample, 434–442
 - sending messages, 430–431
 - services, 423–424
- CREATE ASSEMBLY command, 56
- CREATE TABLE command, 56
- CREATE TYPE command, 56
- cursors
 - background knowledge, 112
 - compared to set-based querying solutions, 112–114
 - compared to set-based solution for dealing with individual rows, 115–116
 - compared to set-based solution for scanning data, 114–115
 - compared to set-based solution overhead, 114–115
 - complexity of problems, 129
 - custom aggregates, 116–118
 - matching problem algorithm (guaranteed solution), 134–135
 - matching problem algorithm (nonguaranteed solution), 137
 - matching problem code (guaranteed solution), 134
 - matching problem code (nonguaranteed solution), 135

cursors, *continued*

- matching problem Events and Rooms tables, 131–133
- matching problem overview, 131
- maximum concurrent sessions
 - benchmark code, 127–129
- maximum concurrent sessions overview, 122
- maximum concurrent sessions Sessions table, 122–123
- maximum concurrent sessions solution, 125–126
- order-based access overview, 116
- overview, 112
- reasons to avoid using, 112–113
- running aggregations, 118–122
- custom aggregates in cursor solutions, 116–118

D

- data-definition language (DDL) triggers
 - database-level, 346–349
 - overview, 344–345
 - server-level, 350–351
- database-level DDL triggers, 346–349
- database mirroring in Service Broker routes, 459
- datatypes
 - character-related problems. *See* character-related problems
 - CREATE XML SCHEMA COLLECTION T-SQL statement, 75
 - DATE vs. TIME, 7
 - DATEADD function, 6
 - DATEDIFF function, 6
 - DATENAME function, 6–7
 - DATEPART function, 6
 - DATETIME birthday problem, 8–10
 - DATETIME earliest date, 3
 - DATETIME functions, 6–7
 - DATETIME generating series of dates, 24–25
 - DATETIME grouping by week, 20–22
 - DATETIME grouping overlaps, 13–16
 - DATETIME identifying overlaps, 11–13
 - DATETIME identifying weekdays, 18–20
 - DATETIME ISO weeks, 22–23
 - DATETIME leap year issues, 8–10
 - DATETIME literals, 4–5
 - DATETIME manipulation issues, 3
 - DATETIME maximum number of overlapping sessions, 16–18
 - DATETIME overlaps overview, 10–11
 - DATETIME overview, 2
 - DATETIME rounding issues, 5–6
 - DATETIME set-based solution, 16–18
 - DATETIME storage format, 2–3
 - DATETIME vs. SMALLDATETIME, 2
 - DATETIME working days, 23–24
 - DECIMAL implicit conversions, 36
 - GETDATE function, 7
 - implicit conversions. *See* implicit conversions
 - LOB. *See* Large Object (LOB)
 - overview, 1–2
 - SMALLDATETIME vs. DATETIME, 2
 - TIME vs. DATE, 7
 - XML changing structures, 67
 - XML column from well-formed to schema-validated, 77
 - XML contacts table, 75
 - XML creation script for ContactOtherAttributes schema collection, 76
 - XML data insertion, 77–78
 - XML database model diagrams (ER diagrams), 69–72
 - XML format for DDL triggers, 67
 - XML index on filtered column, 73–75

- XML modifying parts of data based on structure, 68
- XML .nodes method, 80–81
- XML open schema environment, 75
- XML overview, 65
- XML parameter of stored procedure, 81–82
- XML passing array as parameter to stored procedure, 68
- XML .query method, 72–73, 78–79
- XML querying catalog views, 76
- XML relational model limits, 66–67
- XML serialized objects in database, 68
- XML showplans, 67
- XML sparse data, 67
- XML support in relational database, 65
- XML table column with Visio documents, 69–72
- XML test database, 69
- XML .value method, 79–80, 81
- XML Visio diagrams, 69
- XML XQuery modification statements, 82–83
- DATE vs. TIME datatypes, 7
- DATEADD function, 6
- DATEDIFF function, 6
- DATENAME function, 6–7
- DATEPART function, 6
- DATETIME datatype
 - birthday problem, 8–10
 - compared to SMALLDATETIME, 2
 - DATE vs. TIME, 7
 - DATEADD function, 6
 - DATEDIFF function, 6
 - DATENAME function, 6–7
 - DATEPART function, 6
 - earliest date, 3
 - functions, 6–7
 - generating series of dates, 24–25
 - GETDATE function, 7
 - grouping by week, 20–22
 - grouping overlaps, 13–16
 - identifying overlaps, 11–13
 - identifying weekdays, 18–20
 - ISO weeks, 22–23
 - leap year issues, 8–10
 - literals, 4–5
 - manipulation issues, 3
 - maximum number of overlapping sessions, 16–18
 - overlaps overview, 10–11
 - overview, 2
 - rounding issues, 5–6
 - set-based solution, 16–18
 - storage format, 2–3
 - TIME vs. DATE, 7
 - working days, 23–24
- datetime-related querying problems
 - birthday problem, 8–10
 - generating series of dates, 24–25
 - grouping by week, 20–22
 - grouping overlaps, 13–16
 - identifying overlaps, 11–13
 - identifying weekdays, 18–20
 - ISO weeks, 22–23
 - leap year issues, 8–10
 - maximum number of overlapping sessions, 16–18
 - overlaps overview, 10–11
 - set-based solution, 16–18
 - working days, 23–24
- DBCC statement output in stored procedures, 272
- DDL. *See* data-definition language (DDL) triggers
- deadlocks in transactions
 - caused by missing indexes, 385–388
 - overview, 383
 - simple example, 384–385
 - single table, 388–390
 - trapping errors, 404–409
- DECIMAL datatype, 36

- DEFAULT constraints for scalar UDFs, 219–220
- DEFAULT message type, 418
- deleted tables in AFTER triggers, 316–318
- deploying UDTs using T-SQL
 - arithmetic operations, 58–59
 - C# .NET-based ComplexNumberCS UDA, 60–61
 - CREATE ASSEMBLY command, 56
 - CREATE TABLE command, 56
 - CREATE TYPE command, 56
 - enabling CLR, 55
 - NULL number, 58
 - overview, 55
 - SELECT statement, 57
 - ToString method, 57
 - Visual Basic .NET-based
 - ComplexNumberVB UDT, 61
 - Visual Basic .NET-based
 - ComplexNumberVB_SUM UDA, 61–64
 - Write method, 58
- dialog conversations
 - asymmetric key authentication, 444–445
 - asynchronous delivery, 414–415
 - beginning and ending dialogs, 424–426
 - compared to monolog conversations, 412
 - configuring security, 445–448
 - contracts, 417–418
 - DEFAULT message type, 418
 - definition, 412
 - endpoints. *See* endpoints in Service Broker
 - groups, 428–430
 - messages, 415–417
 - order, 413–414
 - overview, 411–412
 - poison messages, 442–443
 - queues, 418–422
 - receiving messages, 432–433
 - reliability, 413
 - routing. *See* routing in Service Broker
 - sample, 434–442
 - security overview, 443–444
 - sending messages, 430–431
 - services, 423–424
- dirty reads, 370
- distribution in Service Broker
 - adjacent broker protocol (ABP), 449–450
 - BROKER_INSTANCE parameter, 456
 - database mirroring in routes, 459
 - endpoints. *See* endpoints in Service Broker
 - forwarding in routes, 459–460
 - incoming routes, 457
 - load balancing in routes, 458–459
 - overview, 448–449
 - route overview, 455–456
 - special addresses in routes, 457–458
 - wildcards in routes, 457–458
- dynamic filters, 160–166
- dynamic maintenance activities, 153–156
- dynamic pivot in stored procedures
 - input parameters, 294
 - solution 1, 294–301
 - solution 2, 301–305
- dynamic SQL
 - direct permissions, 140
 - dynamic filters, 160–166
 - environmental settings, 140, 153
 - EXEC AT syntax, 146–149
 - EXEC concatenating variables, 145–146
 - EXEC example, 141–142
 - EXEC no interface, 142–145
 - EXEC overview, 141
 - EXEC vs. sp_executesql, 140
 - local temporary table created in calling
 - batch, 141
 - local variable visible only to batch where
 - declared, 141

- maintenance activities, 153–156
- overview, 139
- pivoting, 166–170
- separate batch from calling batch, 140
- sp_executesql interface, 149–152
- sp_executesql overview, 149
- sp_executesql statement limit, 152
- sp_executesql vs. EXEC, 140
- SQL injection code constructed at client, 172–173
- SQL injection code constructed at server, 173–177
- SQL injection overview, 139, 172
- SQL injection protection, 177–179
- storing computations, 156–160
- unpivoting, 170–171

E

- earliest date for DATETIME datatype, 3
- ENCRYPTION option, 202
- ending dialogs, 424–426
- endpoints in Service Broker
 - certificate authentication, 453–455
 - columns in view, 426–428
 - configuring ABP connections, 451
 - conversations, 426–428
 - encryption settings, 451–452
 - overview, 426, 450–451
 - states, 427–428
 - sys.conversation_endpoints view, 426
 - Windows authentication, 452–453
- environmental settings for dynamic SQL, 140, 153
- @@error function, 391
- errors
 - stored procedures, 272
 - transactions, 399
- exception handling
 - deadlock trapping, 404–409
 - errors in transactions, 399

- new functions, 396–399
- overview, 391
- prior to SQL Server 2005, 391–395
- save points, 401–404
- TRY/CATCH construct, 395–396
- update conflicts, 404–409
- XACT_STATE function, 399–400
- exclusive locks, 365
- EXEC command
 - AT clause, 146–149
 - compared to sp_executesql, 140
 - concatenating variables, 145–146
 - example, 141–142
 - no interface, 142–145
 - overview, 141
- EXECUTE AS clause in stored procedures, 288–289
- EXECUTE command. *See* EXEC command
- execution plan reuse in stored procedures, 275–280
- explicit vs. implicit conversions in CLR
 - scalar UDFs, 228–231
- extended stored procedures, 267

F

- filter expressions, 37–40
- forwarding in Service Broker routes, 459–460

G

- generating series of dates, 24–25
- GETDATE function, 7
- global temporary tables
 - compared to local temporary tables, 94
 - example, 94–96
 - overview, 94
- grouping by week, 20–22
- grouping overlaps, 13–16
- groups in dialog conversations, 428–430

I

identifying

- number of affected rows in AFTER triggers, 318–321
- overlaps, 11–13
- type of AFTER triggers, 321–323
- weekdays, 18–20

implicit conversions

- CASE expressions, 36
- compared to explicit conversions in CLR scalar UDFs, 228–231
- DECIMAL datatype, 36
- filter expressions, 37–40
- scalar expressions, 36–37

incoming routes in Service Broker, 457

inconsistent analysis, 370

indexed views, 206–210

inline table-valued UDFs, 239–241

input parameters for stored procedures, 267–269

inserted tables in AFTER triggers, 316–318

INSTEAD OF triggers

- automatic handling of sequences, 342–344
- compared to AFTER, 336
- firing before checking constraints, 336
- overview, 335–336
- per-row, 336–339
- used with views, 339–342

intent locks, 365

interface

- sp_executesql, 149–152
- stored procedure input parameters, 267–269
- stored procedure output parameters, 269–273

IP patterns, 29–30

ISO weeks, 22–23

isolation levels in transactions

- concurrency model, 370
- conflict detection in snapshot, 377–379
- dirty reads, 370
- inconsistent analysis, 370
- lost updates, 370
- nonrepeatable reads, 370
- overview, 370
- phantoms, 370
- query level vs. session level, 371
- read committed, 372–373
- read committed snapshot, 379–380
- read uncommitted, 371–372
- repeatable read, 373–374
- row versioning, 375
- serializable, 374–375
- session level vs. query level, 371
- snapshot, 375–379
- summary, 380–381

K-L

key-range locks, 366

language for creating CLR UDTs, 47–48

Large Object (LOB)

- BULK rowset provider, 34–35
- MAX specifier, 32–34
- overview, 32

leap year issues, 8–10

LIKE predicate, 26–27

load balancing in Service Broker routes, 458–459

LOB. *See* Large Object (LOB)

local temporary tables

- compared to global temporary tables, 94
- compared to table expressions, 104
- compared to table variables, 104

- name resolution, 90–92
- overview, 86
- schema changes in dynamic batches, 92–94
- scope, 87–88
- statistics, 88–90
- summary exercise, 104–109
- tempdb, 86–87, 101–102
- transaction context, 88
- visibility, 87–88

locks in transactions

- blocking scenario, 366–370
- bulk-update (BU), 365
- compatibility, 366
- exclusive, 365
- intent, 365
- key-range, 366
- modes, 365
- overview, 364–365
- schema, 365

lost updates, 370

M

matching problems

- cursor algorithm (guaranteed solution), 134–135
- cursor algorithm (nonguaranteed solution), 137
- cursor code (guaranteed solution), 134
- cursor code (nonguaranteed solution), 135
- Events and Rooms tables, 131–133
- overview, 131

materializing data temporarily

- overview, 85
- table expressions. *See* table expressions
- table variables. *See* table variables
- temporary tables. *See* temporary tables

MAX specifier, 32–34

maximum concurrent sessions

- cursor-based solution, 125–126
- cursor benchmark code, 127–129
- overview, 122
- Sessions table, 122–123
- set-based auxiliary table, 123–124
- set-based execution plan, 124–125

maximum number of overlapping sessions, 16–18

Message Queuing (MSMQ), 462–463

messages in dialog conversations, 415–417

modular approach in views

- overview, 189
- Sales solution, 191–194
- Sales solution optimized, 194–198
- Sales table example, 189–191

monolog conversations, 412

MSMQ (Message Queuing), 462–463

multiple row sets for stored procedures, 271

multistatement table-valued UDFs, 248–252

N

name resolution for local temporary tables, 90–92

nesting in AFTER triggers, 328–329

nonrepeatable reads, 370

NULL number, 58

number of rows affected in stored procedures, 272

O

options for views

- CHECK OPTION, 204–205
- ENCRYPTION, 202
- overview, 202
- SCHEMABINDING, 203–204
- VIEW_METADATA, 205–206

order-based access

- cursor-based solution, 125–126
- cursor benchmark code, 127–129
- custom aggregates, 116–118
- maximum concurrent sessions overview, 122
- overview, 116
- running aggregations, 118–122
- Sessions table, 122–123
- set-based auxiliary table, 123–124
- set-based execution plan, 124–125
- ORDER BY clause in views, 183–187
- order of dialog conversations, 413–414
- organization of this book, xxi
- output parameters for stored procedures, 269–273
- overhead for cursor vs. set-based solution, 114–115
- overlaps
 - grouping, 13–16
 - identifying, 11–13
 - maximum number of overlapping sessions, 16–18
 - overview, 10–11
 - set-based solution, 16–18

P

- parameter sniffing problem, 284–288
- parameterizing sort order in stored procedures
 - solution 1, 289–290
 - solution 2, 290–291
 - solution 3, 291–292
 - solution 4, 292–294
- parsing character strings, 28
- pattern matching
 - CHECK constraints, 27–28
 - IP patterns, 29–30

- LIKE predicate, 26–27
- overview, 26
- parsing character strings, 28
- per-row user-defined functions (UDFs), 252–255
- performance
 - CLR SQL Signature vs. T-SQL, 236–239
 - CLR vs. T-SQL UDFs, 247
 - scalar UDFs, 217–219
 - stored procedures, 257
 - T-SQL vs. CLR SQL Signature, 236–239
 - T-SQL vs. CLR UDFs, 247
 - views, 182
- phantoms, 370
- pivoting in dynamic SQL, 166–170
- poison messages in dialog conversations, 442–443
- PRIMARY KEY constraints for scalar UDFs, 221–222
- problems
 - character-related. *See* character-related problems
 - cursors. *See* cursors
 - datetime-related querying. *See* datetime-related querying problems
 - matching. *See* matching problems
 - parameter sniffing, 284–288
 - set-based querying. *See* set-based querying solutions
- programming CLR UDTs
 - arithmetic operations, 58–59
 - C# .NET-based ComplexNumberCS UDA, 60–61
 - CREATE ASSEMBLY command, 56
 - CREATE TABLE command, 56
 - CREATE TYPE command, 56
 - creating UDTs, 51–55
 - deploying using T-SQL overview, 55
 - enabling CLR, 55

- NULL number, 58
- overview, 48
- requirements, 48–50
- SELECT statement, 57
- ToString method, 57
- Visual Basic .NET-based
 - ComplexNumberVB UDT, 61
- Visual Basic .NET-based
 - ComplexNumberVB_SUM UDA, 61–64
- Write method, 58

Q

- querying
 - datetime-related. *See* datetime-related
 - querying problems
 - set-based. *See* set-based querying solutions
- queues in dialog conversations, 418–422

R

- read committed isolation level, 372–373
- read committed snapshot isolation level, 379–380
- read uncommitted isolation level, 371–372
- receiving messages in dialog conversations, 432–433
- recompilations in stored procedures, 281–284
- recursion in AFTER triggers, 328–329
- refreshing views, 187–189
- regular expressions for CLR scalar UDFs, 224–228
- relational division exercise, 104–109
- reliable dialog conversations, 413
- repeatable read isolation level, 373–374
- resolution for stored procedures, 273–275
- return value in stored procedures, 272
- rollback with triggers, 315

- routing in Service Broker
 - adjacent broker protocol (ABP), 449–450
 - BROKER_INSTANCE parameter, 456
 - database mirroring in routes, 459
 - endpoints. *See* endpoints in Service Broker
 - forwarding in routes, 459–460
 - incoming routes, 457
 - load balancing in routes, 458–459
 - overview, 448–449
 - route overview, 455–456
 - special addresses in routes, 457–458
 - wildcards in routes, 457–458
- row set schema retrieved with
 - SqlDataReader, 273
- row versioning isolation levels, 375
- running aggregations, 118–122

S

- sample databases, installing, xxii
- sample dialog conversations, 434–442
- save points in transactions, 381–383, 401–404
- scalar expressions, 36–37
- scalar user-defined functions (UDFs)
 - CHECK constraints, 220
 - CLR overview, 222–223
 - CLR routines, 223
 - CLR SQL Signature in C#, 236
 - CLR SQL Signature in Visual Basic, 233–235
 - CLR SQL Signature vs. T-SQL performance, 236–239
 - DEFAULT constraints, 219–220
 - explicit vs. implicit conversions, 228–231
 - overview, 214–215
 - performance issues, 217–219
 - PRIMARY KEY constraints, 221–222
 - regular expressions, 224–228

- scalar user-defined functions (UDFs),
 - continued*
 - SQL Signature overview, 231
 - T-SQL, 215–217
 - T-SQL SQL Signature, 231–233
 - T-SQL vs. CLR SQL Signature performance, 236–239
 - UNIQUE constraints, 221–222
- scenarios in Service Broker
 - asynchronous processing, 461
 - overview, 460
 - Service Oriented Architecture (SOA), 460–461
- schema changes
 - temporary tables in dynamic batches, 92–94
 - testing, 2
- schema locks, 365
- SCHEMABINDING option, 203–204
- scope
 - local temporary tables, 87–88
 - table expressions, 103
 - table variables, 97
- security of dialog conversations
 - asymmetric key authentication, 444–445
 - configuring, 445–448
 - overview, 443–444
- SELECT statement, 57
- sending messages in dialog conversations, 430–431
- serializable isolation level, 374–375
- server-level DDL triggers, 350–351
- Service Broker
 - adjacent broker protocol (ABP), 449–450
 - asymmetric key authentication, 444–445
 - asynchronous processing, 461
 - beginning and ending dialogs, 424–426
 - BROKER_INSTANCE parameter, 456
 - certificate authentication for endpoints, 453–455
 - compared to BizTalk, 463
 - compared to MSMQ, 462–463
 - compared to WCF, 463
 - configuring ABP connections for endpoints, 451
 - configuring dialog security, 445–448
 - contracts, 417–418
 - database mirroring in routes, 459
 - DEFAULT message type, 418
 - dialog conversation asynchronous delivery, 414–415
 - dialog conversation definition, 412
 - dialog conversation groups, 428–430
 - dialog conversation order, 413–414
 - dialog conversation overview, 411–412
 - dialog conversation reliability, 413
 - dialog sample, 434–442
 - dialog security overview, 443–444
 - dialog vs. monolog conversations, 412
 - distribution overview, 448–449
 - endpoint columns in view, 426–428
 - endpoint encryption settings, 451–452
 - endpoint overview, 426, 450–451
 - endpoint states, 427–428
 - features, 462
 - forwarding in routes, 459–460
 - incoming routes, 457
 - limitations, 462
 - load balancing in routes, 458–459
 - messages, 415–417
 - monolog vs. dialog conversations, 412
 - overview, 411
 - poison messages, 442–443
 - queues, 418–422
 - receiving messages, 432–433
 - route overview, 455–456

- routing overview, 448–449
- scenario overview, 460
- sending messages, 430–431
- Service Oriented Architecture (SOA), 460–461
- services, 423–424
- special addresses in routes, 457–458
- sys.conversation_endpoints view, 426
- wildcards in routes, 457–458
- Windows authentication for endpoints, 452–453
- Service Oriented Architecture (SOA), 460–461
- services in dialog conversations, 423–424
- set-based querying solutions
 - compared to cursor solution for dealing with individual rows, 115–116
 - compared to cursor solution for scanning data, 114–115
 - compared to cursor solution overhead, 114–115
 - compared to cursor solutions, 112–114
 - complexity of problems, 129
 - DISTINCT applied to remove duplicates, 130–131
 - importance of using good sample data, 130
 - maximum concurrent sessions auxiliary table, 123–124
 - maximum concurrent sessions benchmark code, 127–129
 - maximum concurrent sessions execution plan, 124–125
 - maximum concurrent sessions overview, 122
 - maximum concurrent sessions Sessions table, 122–123
 - maximum number of overlapping sessions, 16–18
 - ORDER BY in OVER clause for aggregations, 129
- single row set for stored procedures, 271
- SMALLDATETIME vs. DATETIME datatype, 2
- snapshot isolation level, 375–379
- SOA (Service Oriented Architecture), 460–461
- special addresses in Service Broker routes, 457–458
- special stored procedures (sp_ prefix), 262–264
- sp_executesql command
 - compared to EXEC command, 140
 - interface, 149–152
 - overview, 149
 - statement limit, 152
- split array
 - CLR split UDFs, 243–247
 - CLR vs. T-SQL UDF performance, 247
 - overview, 242
 - T-SQL split UDFs, 242–243
 - T-SQL vs. CLR UDF performance, 247
- sp_prefix (special) stored procedures, 262–264
- SQL injection
 - code constructed dynamically at client, 172–173
 - code constructed dynamically at server, 173–177
 - overview, 139, 172
 - protecting against, 177–179
- SQL Signature function
 - CLR SQL Signature in C#, 236
 - CLR SQL Signature in Visual Basic, 233–235
 - CLR SQL Signature vs. T-SQL performance, 236–239
 - overview, 231
 - T-SQL SQL Signature UDFs, 231–233
 - T-SQL vs. CLR SQL Signature performance, 236–239

statement limit for `sp_executesql` command, 152

statistics

- local temporary tables, 88–90
- table variables, 98–101

storage format for DATETIME datatype, 2–3

stored procedures

- CLR example 1, 305–309
- CLR example 2, 309–313
- CLR overview, 305
- compared to triggers, 315
- compared to UDFs, 257
- DBCC statement output, 272
- dynamic pivot input parameters, 294
- dynamic pivot solution 1, 294–301
- dynamic pivot solution 2, 301–305
- encapsulation, 257
- errors, 272
- EXECUTE AS clause, 288–289
- execution plan reuse, 275–280
- extended, 267
- input parameters, 267–269
- interface, 267–273
- multiple row sets, 271
- number of rows affected, 272
- output parameters, 269–273
- overview, 257
- parameter sniffing problem, 284–288
- parameterizing sort order solution 1, 289–290
- parameterizing sort order solution 2, 290–291
- parameterizing sort order solution 3, 291–292
- parameterizing sort order solution 4, 292–294
- performance, 257
- recompilations, 281–284
- resolution, 273–275
- return value, 272

- row set schema retrieved with `SqlDataReader`, 273
- single row set, 271
- special (`sp_` prefix), 262–264
- system, 264–266
- T-SQL PRINT statement output, 272
- temporary, 266
- types, 258
- user-defined, 258–262
- user-defined types (UDTs), 273
- warnings, 272
- XML output, 273

storing computations in dynamic SQL, 156–160

support for this book, xxii–xxiii, 1

system requirements, xxi

system stored procedures, 264–266

T

T-SQL PRINT statement output in stored procedures, 272

T-SQL scalar UDFs, 215–217

T-SQL split UDFs, 242–243, 247

T-SQL SQL Signature UDFs, 231–233, 236–239

table expressions

- compared to local temporary tables, 104
- compared to table variables, 104
- example, 103–104
- overview, 103
- scope, 103
- summary exercise, 104–109
- visibility, 103
- when to use, 103

table-valued user-defined functions (UDFs)

- CLR split, 243–247
- CLR vs. T-SQL split performance, 247
- inline, 239–241
- multistatement, 248–252

- overview, 239
- split array overview, 242
- T-SQL split, 242–243
- T-SQL vs. CLR split performance, 247
- table variables
 - compared to local temporary tables, 104
 - compared to table expressions, 104
 - example, 96
 - limitations, 96–97
 - myths, 96
 - scope, 97
 - statistics, 98–101
 - tempdb, 97, 101–102
 - transaction context, 98
 - visibility, 97
- tempdb
 - local temporary tables, 86–87, 101–102
 - table variables, 97, 101–102
- temporary stored procedures, 266
- temporary tables
 - benefits, 86
 - compared to table expressions, 104
 - compared to table variables, 104
 - example, 94–96
 - global temporary tables overview, 94
 - global vs. local, 94
 - local temporary tables overview, 86
 - local vs. global, 94
 - name resolution, 90–92
 - overview, 85
 - schema changes in dynamic batches, 92–94
 - scope, 87–88
 - statistics, 88–90
 - summary exercise, 104–109
 - tempdb, 86–87, 101–102
 - transaction context, 88
 - types, 86
 - visibility, 87–88
- TIME vs. DATE datatypes, 7
- ToString method, 57
- transaction context
 - local temporary tables, 88
 - table variables, 98
- transactions
 - ACID, 361
 - BEGIN TRAN/COMMIT TRAN block, 362
 - checkpoint process, 362
 - concurrency, 361–362
 - creating tables, 363–364
 - deadlock caused by missing indexes, 385–388
 - deadlock overview, 383
 - deadlock simple example, 384–385
 - deadlock with single table, 388–390
 - dirty reads, 370
 - errors, 399
 - exclusive locks, 365
 - inconsistent analysis, 370
 - intent locks, 365
 - isolation concurrency model, 371
 - isolation level conflict detection, 377–379
 - isolation level read committed, 372–373
 - isolation level read committed snapshot, 379–380
 - isolation level read uncommitted, 371–372
 - isolation level repeatable read, 373–374
 - isolation level row versioning, 375
 - isolation level serializable, 374–375
 - isolation level session vs. query level, 371
 - isolation level snapshot, 375–379
 - isolation level summary, 380–381
 - isolation levels overview, 370
 - issuing INSERT statements, 364
 - key-range locks, 366
 - lock blocking scenario, 366–370
 - lock compatibility, 366
 - lock modes, 365

transactions, *continued*

- locks overview, 364–365
- lost updates, 370
- maintaining, 362
- nonrepeatable reads, 370
- overview, 361
- phantoms, 370
- save points, 381–383, 401–404
- schema locks, 365
- submitting changes, 362
- update conflicts, 404–409
- XACT_STATE function, 399–400

triggers

- AFTER auditing example, 333–335
- AFTER COLUMNS_UPDATED function, 329–332
- AFTER deleted tables, 316–318
- AFTER identifying number of rows, 318–321
- AFTER identifying type, 321–323
- AFTER inserted tables, 316–318
- AFTER nesting, 328–329
- AFTER not firing for specific statements, 324–328
- AFTER overview, 316
- AFTER recursion, 328–329
- AFTER UPDATE predicate, 329–332
- AFTER vs. INSTEAD OF, 336
- CLR example, 351–353
- CLR example C# code, 353–356
- CLR example output, 359–360
- CLR example registering and attaching, 359
- CLR example Visual Basic code, 356–359
- CLR overview, 351
- compared to stored procedures, 315
- DDL database-level, 346–349
- DDL overview, 344–345
- DDL server-level, 350–351
- INSTEAD OF automatic handling of sequences, 342–344

INSTEAD OF firing before checking

constraints, 336

INSTEAD OF overview, 335–336

INSTEAD OF per-row, 336–339

INSTEAD OF used with views, 339–342

INSTEAD OF vs. AFTER, 336

overview, 315–316

rollback, 315

TRY/CATCH construct in exception

handling, 395–396

U

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

UDTs. *See* CLR user-defined types (UDTs)

UNIQUE constraints for scalar UDFs, 221–222

unpivoting in dynamic SQL, 170–171

update conflicts in transactions, 404–409

UPDATE predicate, 329–332

updates for SQL Server, xxii

updating views, 198–202

user-defined functions (UDFs)

CLR scalar CLR routines, 223

CLR scalar explicit vs. implicit conversions, 228–231

CLR scalar overview, 222–223

CLR scalar regular expressions, 224–228

CLR split, 243–247

CLR SQL Signature in C#, 236

CLR SQL Signature in Visual Basic, 233–235

CLR SQL Signature vs. T-SQL performance, 236–239

CLR vs. T-SQL split performance, 247

compared to stored procedures, 257

facts about, 214

inline table-valued, 239–241

multistatement table-valued, 248–252

overview, 213

per-row, 252–255

- scalar CHECK constraints, 220
- scalar DEFAULT constraints, 219–220
- scalar overview, 214–215
- scalar performance issues, 217–219
- scalar PRIMARY KEY constraints, 221–222
- scalar SQL Signature overview, 231
- scalar UNIQUE constraints, 221–222
- split array overview, 242
- T-SQL scalar, 215–217
- T-SQL split, 242–243
- T-SQL SQL Signature, 231–233
- T-SQL vs. CLR split performance, 247
- T-SQL vs. CLR SQL Signature performance, 236–239
- table-valued overview, 239
- user-defined stored procedures, 258–262
- user-defined types. *See* CLR user-defined types (UDTs)

V

- VIEW_METADATA option, 205–206
- views
 - abstraction mechanism, 181
 - CHECK OPTION, 204–205
 - ENCRYPTION option, 202
 - indexed, 206–210
 - INSTEAD OF triggers, 339–342
 - modular approach overview, 189
 - modular Sales solution, 191–194
 - modular Sales solution optimized, 194–198
 - modular Sales table example, 189–191
 - options overview, 202
 - ORDER BY clause, 183–187
 - overview, 181–183
 - performance, 182
 - query requirements, 182–183
 - refreshing, 187–189
 - SCHEMABINDING option, 203–204

- sys.conversation_endpoints, 426–428
- updating, 198–202
- VIEW_METADATA option, 205–206
- XML querying catalog, 76
- visibility
 - local temporary tables, 87–88
 - table expressions, 103
 - table variables, 97
- Visual Basic .NET-based ComplexNumberVB UDT, 61
- Visual Basic .NET-based ComplexNumberVB_SUM UDA, 61–64

W

- warnings in stored procedures, 272
- WCF (Windows Communication Foundation), 463
- wildcards in Service Broker routes, 457–458
- Windows authentication for endpoints, 452–453
- Windows Communication Foundation (WCF), 463
- working days, 23–24
- Write method, 58

X

- XACT_STATE function in transactions, 399–400
- XML datatype
 - changing column from well-formed to schema-validated, 77
 - changing structures, 67
 - contacts table, 75
 - CREATE XML SCHEMA COLLECTION T-SQL statement, 75
 - creation script for ContactOtherAttributes schema collection, 76
 - data insertion, 77–78

XML datatype, *continued*

database model diagrams (ER diagrams),
69–72

DDL triggers, 67

index on filtered column, 73–75

modifying parts of data based on structure,
68

.nodes method, 80–81

open schema environment, 75

overview, 65

parameter of stored procedure, 81–82

passing array as parameter to stored
procedure, 68

.query method, 72–73, 78–79

querying the catalog views, 76

relational model limits, 66–67

serialized objects in database, 68

showplans, 67

sparse data, 67

support in relational database, 65

table column with Visio documents, 69–72

test database, 69

.value method, 79–80, 81

Visio diagrams, 68–69

XQuery modification statements, 82–83

XML output in stored procedures, 273