

# Index

## Symbols

- - hyphen, 238, 319
  - minus, 238
  - subtraction, 321–322
  - unary minus, 320–321
- ‘ (apostrophe)
  - attribute syntax and, 155
  - escape sequence, 313
  - named entity references, 163
  - overview, 312–313
  - sting notation, 18
  - syntax errors by misuse of, 238
- “ (quotation mark)
  - attribute syntax and, 155
  - escape sequence, 308
  - named entity references, 163
  - overview, 307
  - sting notation, 18
  - syntax errors and, 238
- & (ampersand)
  - ampersand entity reference, 163, 310
  - apostrophe entity reference, 310–311
  - decimal character reference, 309
  - greater-than sign entity reference, 311
  - hexadecimal character reference, 309–310
  - less-than sign entity reference, 311–312
  - quote entity reference, 312
- \$ (dollar sign), 30–31, 308–309, 457
- () (parentheses)
  - empty sequence, 17
  - overview, 313–315
  - placement of, 198–199
  - regular expressions, 459
  - using with sequences and expressions, 23
- : (XQueryComment), 16, 314–315, 325
- - asterisk, 68, 315–316
  - multiplication, 316–317
  - wildcard, 68, 316
- , (comma), 23, 320
- . (current context item), 322–323
- .. (parent navigation), 323
- / (slash)
  - overview, 323–324
  - path navigation and, 30
  - root of tree and, 65
- // (navigation operator), 68, 324–325
- />. *see* < (XML element constructor)
- :: (axis separator), 325–326
- :). *see* (: (XQuery comment))
- ::: (pragma/extension), 315, 326
- ? (question mark), 335–336
- ?>. *see* <? (XML processing instruction constructor)
- @ (at), 30, 336
- [ ] (square brackets), 337–338
- ^ (caret), 457
- { } (curly braces)
  - enclosing expressions in, 20, 154
  - escaping, 20, 154, 338–339
- | (vertical bar), 339, 455
- + (addition), 318–319
- + (plus), 175, 238, 317–318
- + (unary plus), 318
- < (less than), 163, 326–327
- < (XML element constructor), 327–328
- <!-- (XML comment constructor), 328
- ]]> (CDATA section constructor), 338
- <![CDATA] (CDATA section constructor), 328–329
- </ (XML element constructor), 329
- <? (XML processing instruction constructor), 331
- << (before), 103, 329
- <= (less than or equal to), 330–331
- = (equality symbol), 332–333
- = (equals), 228, 331–332
- != (inequality sign), 228, 305–307
- := (assignment), 326



- xs:IDREFS, 288
- xs:int, 289
- xs:integer, 289–290
- xs:language, 290–291
- xs:long, 291–292
- xs:Name, 292
- xs:NCName, 292
- xs:negativeInteger, 293
- xs:NMTOKEN, 293
- xs:NMTOKENS, 293
- xs:nonNegativeInteger, 294
- xs:nonPositiveInteger, 294
- xs:normalizedString, 295
- xs:NOTATION, 295
- xs:positiveInteger, 295–296
- xs:QName, 296–297
- xs:short, 297
- xs:string, 297–298
- xs:time, 298
- xs:token, 298–299
- xs:unsignedByte, 299
- xs:unsignedInt, 300
- xs:unsignedLong, 300–301
- xs:unsignedShort, 301–302
- Atomic types
  - atomic values and, 41
  - boolean, 42–43
  - calendar, 46–50
  - hierarchy of, 269
  - kinds of, 12
  - numeric, 43–46
  - overview, 42
  - qualified name, 50–51
  - string, 46
  - untyped data and, 42
- Atomic values
  - atomization, 57
  - item classification and, 41
  - as item kind, 263
- Atomization
  - of sequences, 401
  - type conversion and, 57
- Atoms, regular expressions, 455
- attribute (axis), 65–66, 342
- attribute (constructors), 342–343
- attribute (expression), 156
- attribute value normalization, 172
- attribute( ) test, 67–68
- Attributes
  - camel-case syntax of attribute names, 195–196
  - computed constructors, 156–157
  - construction methods, 155
  - content rules, 163, 166
  - direct constructors, 155–156
  - nodes, 53
  - selecting simultaneously with elements, 203
- avg( ) function, 114, 394
- Axes, of navigation
  - :: (axis separator), 325–326
  - abbreviated, 230
  - ancestor, 205–206, 340
  - ancestor-or-self, 205–206, 340
  - attribute, 65–66, 342
  - child, 65–66, 345
  - defined, 64
  - descendant, 352–353
  - descendant-or-self, 353–355
  - following-sibling, 207, 367, 367–368
  - list of basic, 65–66
  - other types, 68–69
  - parent, 378–379
  - preceding-sibling, 207, 380–381
  - predicates and, 230
  - self, 65–66, 381–382
- axis separator (::), 325–326
- B**
  - Base URI declaration, 117–119, 346
  - base64 binary data, 212–214
  - base64Binary (atomic type), 275–276
  - base-uri( ) function, 56, 85, 394–395
  - before (<<), 329. *see also* Order comparison operators

## 482 Index

- Benchmarks, 259
  - Binary data
    - converting to/from, 186
    - type idioms for, 212–215
  - Body section, XQuery anatomy, 14
  - Boolean
    - boolean XOR, 215–216
    - boolean(), 395–396
    - converting to/from boolean types, 183–184
    - creating boolean types, 42–43
    - false(), 410
    - not(), 433–434
    - overview, 17–18, 276–277
    - predicates, 70
    - true(), 451
  - boolean (atomic type), 17–18, 42–43, 276–277
  - boolean() function, 395–396
  - Boundary whitespace, 165
  - Braces. *see* {} (curly braces)
  - Built-in functions
    - arithmetic expressions, 111
    - library, 87–88
    - overview, 22–23
    - simplifying, 254
  - Built-in namespaces, 120
  - byte (atomic type), 277
- C**
- Calendar, 46–50. *see also* Date/time functions;
    - Gregorian calendar
    - converting to/from calendar types, 184–186
    - date values, 47–48
    - duration values, 49–50
    - QName values, 50–51
    - time values, 47–48
    - time zones and, 49
  - Camel case, 195–196
  - Canonical representation, 181, 273
  - Captured substrings, 459
  - Case
    - camel, 195–196
    - lower, 201–202, 428
    - operator, 343
    - Pascal, 196
    - sensitivity, 195–196, 203–204
    - title, 201–202
    - upper, 201–202, 452
  - case (expression), 343
  - cast (expression), 180–182, 203–204, 343–344
  - castable (expression), 180–181, 344
  - CDATA section
    - constructors, 157–158, 328–329, 338
    - entity escapes eliminated by, 164
    - in XML data model, 36–37
  - ceiling() function, 395–396
  - Character encoding, 172–174
    - kinds of, 173
    - overview, 172
    - UTF-8 and UTF-16, 172–173
    - working with, 173–174
  - Character entities. *see also* Unicode characters
    - attribute values and, 155
    - curly braces and, 154
    - hexadecimal, 163–164, 309–310
    - metacharacters in regular expressions, 457
    - regular expression character properties,
      - 461–462
    - supported by XQuery, 164
    - translating into other characters, 450–451
    - XML content rules and, 163–164
  - Character models, 169–170, 177
  - Character references
    - &# (decimal character reference), 309
    - &#x (hexadecimal character reference),
      - 309–310
    - kinds of, 163
    - resolving into strings, 166–167
    - string constants and, 18
  - child (axis), 65–66, 345
  - Child nodes, 53
  - Code points, 27, 170–171
  - codepoints-to-string() function, 26,
    - 396–397

- Collation
  - declaring in query prolog, 119
  - default, 346–347, 404
  - in .NET, 175
  - in Java, 174
  - parameter in strings, 26–27
  - static context and, 75
  - Unicode code point collation, 174
- collation (expression), 345
- collection() function, 397
- comma (,), 23, 320
- Comma-separated values (CSV), 201
- comment (constructor), 345–346
- comment() test, 67
- Comments
  - (: (XQuery comment), 314–315
  - comment (constructor), 322, 345–346
  - nodes, 67, 159
  - overview, 16
- compare() function, 397–398
- Comparison operators, 27–28
- Comparisons, expressions
  - general comparisons, 100–103
  - node comparisons, 103–104
  - overview, 98
  - sequence and tree comparisons, 104–105
  - value comparisons, 98–100
- Comparisons, sequences
  - existential comparison, 142–143
  - memberwise comparison, 143–145
  - universal comparison, 145–146
- Complex numbers, 222–223
- Complexity, in XQuery
  - case analysis, 253
  - double numbers vs. decimal numbers, 227–228
  - FLWOR expressions not moving current context, 234–235
  - <foo/><foo/>, 238
  - irregularities as source of, 253
  - meaning confusions, 227
  - names, 233–234
  - nested FLWOR, 235–237
  - node sequences vs. node siblings, 230–231
  - non associative arithmetic, 229–230
  - not(=) vs. !=, 228–229
  - path grammar, 239
  - predicates and abbreviated axes and, 230
  - predicates indicating sequences not strings, 228
  - punctuation, 238
  - redundancy as source of, 253
  - size of, 1-471. *see also* Date/time functions
  - sub-tree pruning, 231–232
  - syntax confusions, 237
  - type conversion and, 232–233, 253-254
  - XML numbers, 238–239
- Composition
  - of constructed nodes, 161–162
  - defined, 153
- Computed constructors
  - attributes, 156–157
  - comment nodes, 159
  - document nodes, 158–159
  - elements, 154–155
  - namespace declaration, 161
  - processing instruction nodes, 160
  - text nodes, 157
- concat() function, 87, 175, 398–399
- Concatenation
  - of sequences, 105–106
  - of strings, 26, 175, 398, 442
- Conditional statements
  - else, 359
  - if, 369–370
  - logic operators, 24, 116
  - then, 384
- Constants, 17–19
  - boolean, 17–18
  - numeric, 18–19
  - other, 19
  - string, 18
- Constructors. *see* XML, constructing
- contains() function, 399
- content (expression), 157

## 484 Index

- Content, XML
    - character escapes, 163–164
    - rules for, 163
    - sequences, 166–168
    - whitespace, 164–166
  - context (expression), 346
  - Context item, 72. *see also* . (current context item)
  - Context position, 72. *see also* position( )
    - function
  - Context size, 72. *see also* last( ) function
  - Context, navigation, 72–75
    - collations, 75
    - expression context types, 72–73
    - expression context values, 85
    - focus, 72–74
    - function declaration, 75
    - input sequence, 72
    - namespace declaration, 74–75
    - overview, 72
    - variable declaration, 74
  - Conversion rules. *see* Type conversion
  - count( ) function, 22, 75, 87, 399–400
  - Cross-product joins, 134–135. *see also* Joins
  - curly braces. *see* { } (curly braces)
  - Current context
    - FLWOR expressions not changing, 234–235
    - last( ), 427
    - node, 64–65
    - position( ), 435
    - symbol for, 322–323
  - current-date( ) function, 85, 400
  - current-dateTime( ) function, 85, 400
  - current-time( ) function, 85, 401
- D**
- Data manipulation. *see* DML (Data Manipulation Language)
  - Data Manipulation Language. *see* DML (Data Manipulation Language)
  - Data models
    - further reading, 61
    - serialization, 241–242
  - Data models, XML, 35–40
    - DOM and, 37–38
    - examples, 36–37
    - Infoset and, 38–39
    - overview, 35–36
    - PSVI and, 39
    - XPath 1.0 and, 38
    - XQuery data model and, 40
  - Data models, XQuery, 40. *see also* Type system
  - data( ) function, 401
  - Database evaluation, 246
  - Databases, XML and, 9–10
  - date (atomic type), 47–48, 184–185, 278
  - Date/time functions. *see also* Calendar
    - current-date( ), 85, 400
    - current-dateTime( ), 85, 400
    - current-time( ), 85, 401
    - get-day-from-date( ), 411
    - get-day-from-dateTime( ), 411–412
    - get-days-from-dayTimeDuration( ), 412
    - get-hours-from-dateTime( ), 412
    - get-hours-from-dayTimeDuration( ), 413
    - get-hours-from-time( ), 413
    - get-minutes-from-dateTime( ), 414–415
    - get-minutes-from-dayTimeDuration( ), 415
    - get-minutes-from-time( ), 415–416
    - get-month-from-date( ), 416
    - get-month-from-dateTime( ), 416
    - get-months-from-yearMonthDuration( ), 417
    - get-seconds-from-dateTime( ), 418
    - get-seconds-from-dayTimeDuration( ), 418–419
    - get-seconds-from-time( ), 419
    - get-timezone-from-date( ), 419–420
    - get-timezone-from-dateTime( ), 420
    - get-timezone-from-time( ), 420
    - get-year-from-date( ), 421
    - get-year-from-dateTime( ), 421
    - get-years-from-yearMonthDuration( ), 421–422
    - subtract-dateTimes-yielding-dayTimeDuration( ), 447–448
    - subtract-dateTimes-yielding-yearMonthDuration( ), 448
  - dateTime (atomic type), 47–48, 184–185, 278–280

- dayTimeDuration (atomic type), 280
  - Debugging, 450
  - decimal (atomic type), 18–19, 45, 280–281
  - Decimals
    - decimal character reference (&#), 309
    - double numbers compared with, 227–228
    - numeric entities, 18, 163–164
    - XQuery support for, 163
  - Declarations
    - base URI, 117–119, 346
    - default collation, 119, 346–347
    - default element namespace, 347
    - default function namespace, 93, 347–348
    - functions, 90, 348–349
    - modules, 121–122
    - namespace, 119–120, 349–350
    - schema imports, 122–123
    - validation, 122–123, 350–351
    - variables, 121, 351–352
    - version, 117
    - XML space policy, 117, 352
  - declare. *see* Declarations
  - Deep equality, 98
  - deep-equal( ) function, 102, 104, 143, 401–403
  - default (expression), 352
  - default-collation( ) function, 404
  - Derivation, 268–269
  - Derived types, XML Schema, 11
  - descendant axis, 68, 352–353
  - descendant-or-self axis, 68, 353–355
  - descending (expression), 355
  - Design by committee. *see* Complexity, in XQuery
  - Digits, XQuery vs. XML, 238–239
  - Direct constructors
    - attributes, 155–156
    - comments, 159
    - elements, 153–154
    - namespace attributes, 160
    - processing-instructions, 159
  - distance( ) example, 91
  - distinct-values( ) function, 106, 404–405
  - Division (div expression), 110, 112–113, 355–356
  - DML (Data Manipulation Language), 256–258
    - SiXDML, 257
    - XQuery DML, 256–257
    - XUpdate, 258
  - doc( ) function, 29, 71, 405–406
  - document (constructor), 356–357
  - Document nodes, 158–159
  - Document Object Model (DOM), 37–38
  - Document order, 42, 53, 266
  - document-node( ) test, 58, 67
  - Documents
    - data model hierarchy and, 53
    - trees and, 266
    - XML unifying with databases and programs, 9–10
  - document-uri( ) function, 406
  - DOM (Document Object Model), 37–38
  - double (atomic type), 18–19, 46, 281–282
  - double curly brackets {{ }}, 20
  - double quotes (“). *see* “ (quotation mark)
  - Double value, numeric constants, 18
  - Double-precision floating point numbers, 227–228, 281–282, 357
  - Down-casting operators, 186
  - DTDs, 37
  - Duration
    - Calendar types and, 49–50, 184–186, 283
    - dayTimeDuration, 280
    - subtypes, 252, 254
    - year and month, 303
  - duration (atomic type), 49–50, 283
    - Calendar types and, 184–186
  - Dynamic errors, 33
  - Dynamic types
    - vs. static types, 179
    - XQuery expressions, 11
    - XQuery processing model and, 15
- ## E
- e (expression), 357
  - E notation (expression), 357
  - Early evaluation, 246

## 486 Index

- EBNF grammar, 463
- EBV (Effective Boolean Value)
  - boolean() function and, 395–396
  - logic operators and, 115–116
  - type conversion and, 57
- Effective Boolean Value. *see* EBV (Effective Boolean Value)
- element (constructor), 154–155, 358–359
- element() test, 67
- Elements
  - computed constructors, 154–155
  - construction methods, 154
  - content rules, 167–168
  - conventions for writing simple and complex, 197
  - direct constructors, 154
  - identifiers, 288, 422
  - rules for content handling, 163
  - selecting simultaneously with attributes, 203
  - validation of constructed elements, 163
- else (expression)
  - instance of, 187
  - order of evaluation in, 34
  - overview, 359
- empty greatest (expression), 147–148, 359
- empty least (expression), 147–148, 359
- Empty sequences
  - constant values and, 17
  - type matching and, 58
- empty() function, 58, 406
- ends-with() function, 406–407
- ENTITIES (atomic type), 283
- ENTITY (atomic type), 284
- Entity references
  - ampersand (&#amp;#26;#38;#39;), 310
  - apostrophe (&#39;), 310–311
  - CDATA section eliminate need for
    - escapes, 164
  - ENTITIES, 283–284
  - ENTITY, 284
  - greater-than sign, 311
  - less-than sign (&#38;#39;), 311–312
  - quote (&#39;), 312
  - resolving into strings, 166–167
  - string constants and, 18
  - syntax, 163
- eq (value comparison operator), 359–360
- equality symbol (=), 332–333
- Equality/inequality
  - deep-equal(), 401–403
  - equality symbol (=), 332–333
  - equals (=), 228
  - string values and, 175
- equals (=), 228
- equi-joins, 135, 140. *see also* Joins
- Error cases, XML content, 167
- Error handling, 33–34
- Error preservation, as barrier to query optimization, 249
- error() function, 33, 407
- Escapes
  - apostrophe escape sequence, 313
  - character escapes, 163–164
  - curly brace escape sequence, 338–340
  - escape-uri(), 408
  - quote escape sequence, 308
  - regular expressions, 458–459
- escape-uri() function, 408
- Evaluation context, navigation
  - defined, 72
  - focus and, 72–74
  - input sequence and, 72
  - path steps and, 64
  - values, 85
- every (expression)
  - FLWOR expressions, 133
  - overview, 360
  - sequence comparisons and, 144
- exactly-one() function, 408–409
- except (expression), 107, 209–211, 361
- Existential comparison of sequences, 142–143
- Existential quantification, 134
- exists() function, 409
- Expanded names, 82–83
- expanded-QName() function, 409–410

- Exponential
  - functions, 114–115
  - notation, 357
- Expression reference
  - ancestor (axis), 340
  - ancestor-or-self (axis), 340
  - and, 341
  - as, 341
  - ascending, 341
  - at, 341
  - attribute (axis), 342
  - attribute (constructors), 342–343
  - case, 343
  - cast as, 343–344
  - castable as, 344
  - child (axis), 345
  - for clause, 362–363
  - collation, 345
  - comment (constructor), 345–346
  - context, 346
  - declare base-uri, 346
  - declare default collation, 346–347
  - declare default element namespace, 347
  - declare default function namespace, 347–348
  - declare function, 348–349
  - declare namespace, 349–350
  - declare validation, 350–351
  - declare variable, 351–352
  - declare xmlspace, 352
  - default, 352
  - descendant, 352–353
  - descendant-or-self, 353–355
  - descending, 355
  - div, 355–356
  - document (constructor), 356–357
  - e, 357
  - E notation, 357
  - element (constructor), 358–359
  - else, 359
  - empty greatest, 359
  - empty least, 359
  - eq, 359–360
  - every, 360
  - except, 361
  - extension, 361
  - external, 361
  - FLWOR, 362–366
  - following-sibling, 367–368
  - global, 368
  - gt, 368
  - idiv, 369
  - if, 369–370
  - import module, 370–371
  - import schema, 371–372
  - in, 372
  - instance of, 372
  - intersect, 372–373
  - is, 373
  - lax, 373–374
  - let, 374
  - let clause, 363–364
  - lt, 374–375
  - mod, 375–376
  - module, 376
  - namespace, 377
  - namespace (constructor), 376–377
  - ne, 377
  - nillable, 378
  - or, 378
  - order by clause, 365
  - parent, 378–379
  - pragma, 380
  - preceding, 380
  - preceding-sibling, 380–381
  - preserve, 381
  - processing-instruction (constructor), 379–380
  - return clause, 366–367
  - satisfies, 381
  - self, 381–382
  - skip, 382
  - some, 382
  - stable, 382
  - strict, 383
  - strip, 383

## 488 Index

- Expression reference (*continued*)
- text (constructor), 383–384
  - then, 384
  - to, 384
  - treat as, 384–385
  - type(), 385
  - typeswitch, 385–388
  - union, 388
  - validate, 389
  - version, 389
  - where clause, 364–365
  - xquery version, 390
- Expression reference, symbols, 305–390
- ‘ (apostrophe escape sequence), 313
  - ’ (apostrophe), 312–313
  - (hyphen), 319
  - (subtraction), 321–322
  - (unary minus), 320–321
  - &# (decimal character reference), 309
  - &#x (hexadecimal character reference), 309–310
  - != (inequality sign), 305–307
  - &amp; (ampersand entity reference), 310
  - &apos; (apostrophe entity reference), 310–311
  - &gt; (greater-than sign entity reference), 311
  - &quot; (quote entity reference), 312
  - &t; (less-than sign entity reference), 311–312
  - “ (quotation mark), 307
  - ” (quote escape sequence), 308
  - \$ (variable), 308–309
  - ( (left parenthesis), 313–314
  - (: (XQuery comment), 314–315
  - :: (pragma/extension), 315
  - ) (right parenthesis), 315
  - ° (asterisk), 315–316
  - ° (multiplication), 316–317
  - ° (wildcard), 316
  - , (comma), 320
  - . (current context item), 322–323
  - .. (parent navigation), 323
  - / (slash), 323–324
  - // (navigation operator), 324–325
  - /> (XML element constructor), 325
  - : (color), 325
  - :) (XQuery comment), 325
  - :: (axis separator), 325–326
  - ::: (pragma or extension), 326
  - := (assignment), 326
  - ? (question mark), 335–336
  - ?> (XML processing instruction constructor), 336
  - @ (at), 336
  - [ (left square bracket), 337
  - ] (right square bracket), 338
  - ]> (CDATA section constructor), 338
  - { (left curly brace), 338
  - {{ (left curly brace escape sequence), 338–339
  - | (vertical bar), 339
  - } (right curly brace), 339
  - }} (right curly brace escape sequence), 339–340
  - + (addition), 318–319
  - + (plus sign), 317–318
  - + (unary plus), 318
  - < (less than), 326–327
  - < (XML element constructor), 327–328
  - <!-- (XML comment constructor), 328
  - <![CDATA (CDATA section constructor), 328–329
  - </ (XML element constructor), 329
  - <? (XML processing instruction constructor), 331
  - << (before), 329
  - <= (less than or equal to), 330–331
  - = (equality symbol), 332–333
  - = (equals), 331–332
  - > (greater than), 333–334
  - >= (greater than or equal to), 334–335
  - >> (after), 334
  - > (XML comment constructor), 322
- Expressions, 97–124. *see also* Regular expressions (regexps/regexes)
- arithmetic, 110–115
  - base URI declaration, 117–119

- collation declaration, 119
  - comparisons, 98
  - context, 72–73
  - converting to double, 434
  - FLWOR, 31–33
  - further reading, 124
  - general comparisons, 100–103
  - logic, 115–116
  - module declarations, 121–122
  - namespace declaration, 119–120
  - node comparisons, 103–104
  - notation, 20
  - overview, 97
  - processing sequences, 108–110, 452
  - query prolog, 117
  - schema imports, 122–123
  - sequence comparisons, 104–105
  - sequence construction, 105–108
  - sequence notation, 23
  - sequences, 105
  - tree comparisons, 104–105
  - validation declaration, 122–123
  - value comparisons, 98–100
  - variable declaration, 121
  - version declaration, 117
  - XML space declaration, 117
  - extension (expression), 361
  - Extension mechanisms, 315
  - external (expression), 361
  - External functions, 93, 348
  - External variables, 121, 351
- F**
- Factorials, 221
  - false( ) function, 17–18, 410
  - First common ancestor, 206
  - Fixed-point values
    - as numeric type, 18, 43
    - operators for, 113
  - Fixed-width character encoding, 172
  - float (atomic type), 19, 46, 284–285
  - Floating-point values
    - double-precision, 227–228, 281–282, 357
    - as numeric type, 18, 43
    - operators for, 113
    - single-precision, 284–285
  - floor( ) function, 410–411
  - FLWOR expressions, 362–366
    - for clause, 362–363
    - conventions for writing simple and
      - complex, 197
    - current context, 234–235
    - existential comparison with, 142–143
    - grouping, 149–152
    - iteration, 125–126
    - joins, 133–134
    - let clause, 363–364
    - nesting, 235–237
    - order by clause, 365
    - overview, 31–33
    - parenthesis syntax in, 199
    - quantification, 133–134
    - return clause, 366–367
    - sorting, 146–148
    - SQL compared with, 126–127
    - tuples and, 129–133
    - variable declaration, 128–129
    - where clause, 364–365
    - XSLT compared with, 127–128
  - fn prefix, 89. *see also* Function reference (fn:)
  - Focus, 72–74
  - following-sibling axis, 68, 207, 367–368
  - for clause, 31–33, 362–363. *see also* FLWOR
    - expressions
  - Formal Semantics, 250
  - Fragments
    - of trees, 266
    - XQuery data model hierarchy, 53
  - Full text searches, 258–259
  - Function reference (fn:)
    - abs( ), 391–392
    - adjust-dateTime-to-timezone( ), 392–393
    - adjust-date-to-timezone( ), 392

## 490 Index

### Function reference (fn:) (*continued*)

- adjust-time-to-timezone( ), 393–394
- avg( ), 394
- base-uri( ), 394–395
- boolean( ), 395–396
- ceiling( ), 395–396
- codepoints-to-string( ), 396–397
- collection( ), 397
- compare( ), 397–398
- concat( ), 398–399
- contains( ), 399
- count( ), 399–400
- current-date( ), 400
- current-dateTime( ), 400
- current-time( ), 401
- data( ), 401
- deep-equal( ), 401–403
- default-collation( ), 404
- distinct-values( ), 404–405
- doc( ), 405–406
- document-uri( ), 406
- empty( ), 406
- ends-with( ), 406–407
- error( ), 407
- escape-uri( ), 408
- exactly-one( ), 408–409
- exists( ), 409
- expanded-QName( ), 409–410
- false( ), 409–410
- floor( ), 410–411
- get-day-from-date( ), 411
- get-day-from-dateTime( ), 411–412
- get-days-from-dayTimeDuration( ), 412
- get-hours-fromdateTime( ), 412
- get-hours-fromdayTimeDuration( ), 413
- get-hours-from-time( ), 413
- get-inscope-namespace-prefixes( ), 413–414
- get-local-name-from-QName( ), 414
- get-minutes-fromdateTime( ), 414–415
- get-minutes-from-dayTimeDuration( ), 415
- get-minutes-from-time( ), 415–416
- get-month-from-date( ), 416
- get-month-from-dateTime( ), 416
- get-months-from-yearMonthDuration( ), 417
- get-namespace-uri-for-prefix( ), 417
- get-namespace-uri-from-QName( ), 417–418
- get-seconds-from-dateTime( ), 418
- get-seconds-from-dayTimeDuration( ), 418–419
- get-seconds-from-time( ), 419
- get-timezone-from-date( ), 419–420
- get-timezone-from-dateTime( ), 420
- get-timezone-from-time( ), 420
- get-year-from-date( ), 421
- get-year-from-dateTime( ), 421
- get-years-from-yearMonthDuration, 421–422
- id( ), 422
- idref( ), 423
- implicit-timezone( ), 423
- index-of( ), 424
- insert-before( ) function, 425
- lang( ), 426–427
- last( ), 427
- local-name( ), 427–428
- lower-case( ), 428
- matches( ), 428–429
- max( ), 429
- min( ), 430
- name( ), 430–431
- namespace-uri( ), 431
- node-name( ), 431–432
- normalize-space( ) function, 432
- normalize-unicode( ) function, 433
- not( ), 433–434
- number( ), 434
- one-or-more( ), 434–435
- position( ), 435
- remove( ), 435–436
- replace( ), 436–437
- resolve-QName( ), 437
- resolve-uri( ), 437–438
- reverse( ), 438
- root( ) function, 438
- round( ), 438–439
- round-half-to-even( ), 439–440
- starts-with( ) function, 440–441
- string( ), 441–442

string-join(), 442  
 string-length(), 443  
 string-to-codepoints(), 443  
 subsequence(), 444  
 substring(), 444–445  
 substring-after(), 445–446  
 substring-before(), 446–447  
 subtract-dateTimes-yielding-dayTimeDuration(), 447–448  
 subtract-dateTimes-yielding-yearMonthDuration(), 448  
 sum(), 448–449  
 tokenize(), 449–450  
 trace(), 450  
 translate(), 450–451  
 true(), 451  
 unordered() function, 452  
 upper-case(), 452  
 zero-or-one(), 453  
 function-available() function, 75

#### Functions

built-in, 22–23, 87–88  
 conversion rules, 89–90. *see also* Type  
     conversion  
 declaration, 75, 348–349  
 default namespace, 93  
 external, 93  
 invoking, 88–89  
 keywords, 16  
 navigating, 71–72  
 overloading, 87, 256  
 recursion, 92–93  
 signature, 22  
 user-defined, 17, 90–92

#### Further reading

data models, 61  
 expressions, 124  
 idioms, 225  
 iteration, 152  
 navigation, 86  
 query optimization, 250  
 standards, 258–259  
 text processing, 177

type operators, 192  
 XML construction, 168  
 XQuery, 34  
 Fuzzy searches, 258

## G

gDay (atomic type), 285–286  
 ge (value comparison operator), 367  
 General comparison operators  
     != (inequality sign), 305–307  
     < (less than), 326–327  
     <= (less than or equal to), 330–331  
     > (greater than), 333–334  
     >= (greater than or equal to), 334–335  
     comparing expressions, 100–103  
     defined, 98  
     kinds of comparison operators, 27–28  
     list of, 101  
     sequences and, 101  
     string values and, 175  
     type conversion and, 102  
 get-day-from-date() function, 411  
 get-day-from-dateTime() function, 411–412  
 get-days-from-dayTimeDuration() function, 412  
 get-hours-fromdateTime() function, 412  
 get-hours-fromdayTimeDuration() function, 413  
 get-hours-from-time() function, 413  
 get-in-scope-prefixes() function, 413–414  
 get-local-name-from-QName() function, 414  
 get-minutes-fromdateTime() function, 414–415  
 get-minutes-from-dayTimeDuration()  
     function, 415  
 get-minutes-from-time() function, 415–416  
 get-month-from-date() function, 416  
 get-month-from-dateTime() function, 416  
 get-months-from-yearMonthDuration()  
     function, 417  
 get-namespaces-in-scope() function, 83  
 get-namespace-uri-for-prefix() function, 83–84, 417  
 get-namespace-uri-from-QName() function,  
     417–418  
 get-seconds-from-dateTime() function, 418

## 492 Index

get-seconds-from-dayTimeDuration() function, 418–419  
 get-seconds-from-time() function, 419  
 get-timezone-from-date() function, 419–420  
 get-timezone-from-dateTime() function, 420  
 get-timezone-from-time() function, 420  
 get-year-from-date() function, 421  
 get-year-from-dateTime() function, 421  
 get-years-from-yearMonthDuration, 421–422  
 global (expression), 368  
 Global validation, 163, 192  
 Global variable declaration, 121  
 gMonth (atomic type), 286  
 gMonthDay (atomic type), 286  
 Grammar, paths, 239  
 Grammar, XQuery, 463–471  
   listed, 464–469  
   operator precedence, 470–471  
   overview, 463  
   reserved keywords, 469–470  
 greater than (>), 163  
 greater than or equal to (>=), 334–335  
 Greater-than sign entity reference (&gt;), 311  
 Gregorian calendar. *see also* Calendar  
   calendar types, 184–185  
   day type, 285–286  
   month and day type, 285–286  
   month and year type, 287  
   month type, 285–286  
   year type, 286–287  
 Grouping, FLWOR expressions, 149–152  
 gt (value comparison operator), 368  
 gYear (atomic type), 286–287  
 gYearMonth (atomic type), 287

## H

Hexadecimal characters  
   formats, 163–164  
   hexadecimal character reference (&#x), 309–310  
   XQuery support for, 163  
 hexBinary (atomic type), 287–288

hexBinary data, 215  
 Hex-encoded binary value, 287–288  
 Hierarchy  
   atomic types, 269  
   node types, 53–54  
   numeric types, 45  
   type conversion and, 182–183  
 hyphen (-), 238, 319

## I

ID (atomic type), 288  
 id() function, 422  
 Idioms  
   arithmetic, 217–224  
   further reading, 225  
   logic idioms, 215–217  
   navigation, 203–209  
   sequence, 209–212  
   text, 200–203  
   type, 212–215  
 idiv (expression), 369  
 IDREF (atomic type), 288  
 idref() function, 423  
 IDREFS (atomic type), 288  
 if (expression)  
   instance of, 187  
   order of evaluation in, 34  
   overview, 369–370  
 Implementation options, XQuery, 3–4  
 Implicit existential quantification, 228  
 implicit-timezone() function, 85, 423  
 import module (expression), 17, 370–371  
 import schema (expression), 122, 371–372  
 Imports  
   module imports, 17, 370–371  
   schema imports, 122–123, 189, 371–372  
 in (expression), 372  
 index-of() function, 109, 424  
 Inequality. *see* Equality/inequality  
 inequality sign (!=), 228, 305–307  
 Information items, Infoset, 39

Infoset (XML Information Set), 38–39  
 Inheritance, 268  
 Inner joins, 135  
 Input sequence, 72  
 insert-before( ) function, 109, 425  
 instance of (expression), 187–188, 372  
 int (atomic type), 289  
 integer (atomic type), 18–19, 44–45, 289–290

### Integers

- converting to, 233
- dividing, 369
- integer (atomic type), 289–290
- long (atomic type), 291–292
- negativeInteger (atomic type), 293
- nonNegativeInteger (atomic type), 294
- nonPositiveInteger (atomic type), 294
- overview, 18
- positiveInteger (atomic type), 295–296
- range of values, 384
- rounding, 438–440
- short (atomic type), 297
- unsignedLong (atomic type), 300–301
- unsignedShort (atomic type), 301–302

intersect (expression), 107, 209–211, 372–373

Invoking functions, 88–89

is (expression), 103, 373

IsNaN example, 217

item( ) function, 11–12

### Items

- classified as atomic values or nodes, 41
- index-of( ), 424
- item( ), 11–12
- in XQuery data model, 40–41, 263

Iteration, 125–152

- existential comparison of sequences, 142–143
- FLWOR expression, 125–126
- further reading, 152
- grouping, 149–152
- joins, 134–141
- memberwise comparison of sequences, 143–145
- overview, 125
- quantification, 133–134

sorting, 146–148

SQL compared to FLWOR, 126–127

tuples, 129–133

universal comparison of sequences, 145–146

variables, 128–129

XSLT compared to FLWOR, 127–128

Iterative expressions, 125

## J

Java, 174

Joins, 134–141

- changing conditions of, 137

- cross-products, 134–135

- equi-joins, 135, 140

- inner joins, 135

- many-to-many, 138–139

- one-to-one, 136

- outer, 139–140

- self-joins, 141

- string-join( ) function, 175, 201, 442

## K

Keywords. *see also* Expression reference

- default, 352

- DML instructions, 256–257

- functions, 16

- lower-case syntax of, 195

- reserved, 469–470

## L

lang( ) function, 425–426

Language. *see also* Grammar, XQuery

- of current node, 426–427

- language (atomic type), 290–291

- programming languages, 5–6

- query languages, 5–9

- regular expressions, 460–461

- language (atomic type), 290–291

- last( ) function, 426

## 494 Index

lax (expression), 373–374  
 Lazy evaluation, 245  
 LCG (linear congruential generator), 220  
 le (value comparison operator), 374  
 Leaf elements, 205  
 left curly brace (`()`), 338  
 left parenthesis (`()`), 313–314  
 left square bracket (`[]`), 337  
 less than (`<`), 163, 326–327  
 less than or equal to (`<=`), 330–331  
 less-than sign entity reference (`&lt;`), 311–312  
 let clause, 363–364. *see also* FLWOR expressions  
 Lexical form, 273  
 Library modules
     built-in functions, 87–88, 254  
     overview, 93–94  
 Limited-precision numbers, 44  
 Linear algebra, 223–224  
 linear congruential generator (LCG), 220  
 Lists, reversing, 201  
 Local name, qualified names, 82  
 local-name(`()`) function, 54, 426–427  
 Logic idioms, 215–217
     boolean XOR, 215–216  
     three-valued, 216–217  
 Logic operators
     and, 341  
     conditional statements, 116  
     EBV and, 115–116  
     list of, 116  
     or, 378  
     overview, 24  
 long (atomic type), 291–292  
 Lower-case
     lower-case(`()`) function, 201, 204, 427  
     text idiom for, 201–202  
 lt (value comparison operator), 374–375

## M

Main module, 93  
 Many-to-many joins, 138–139  
 matches(`()`) function, 428–429

max(`()`) function, 114, 429  
 Maximum depth of subtree, 207–209  
 Median, 218  
 Memberwise comparison of sequences, 143–145  
 Metacharacters, 457  
 Michigan Benchmark, 259  
 min(`()`) function, 114, 429  
 minus (`-`), 238  
 mod (expression), 375–376  
 Modifiers, regular expressions, 455, 457  
 module (expression), 376  
 Modules
     changes to XQuery standard, 252  
     declaring in query prolog, 121–122  
     dividing queries into, 17  
     library modules, 87–88, 93–94, 254  
     main module, 93  
     module (expression), 376  
     XQuery standard and, 252  
 Modulus
     arithmetic operators, 110, 112–113  
     mod (expression), 375–376  
 MRG (multiple recursive generator), 220  
 multiple recursive generator (MRG), 220  
 Multiplication
     arithmetic operators, 110, 112–113  
     symbol (`*`), 316–317

## N

Name (atomic type), 292  
 Name tests, nodes, 66–67  
 name(`()`) function, 54, 430–431  
 Named entities, 163  
 Names
     accuracy in use of, 233–234  
     nodes, 54–55  
     types, 264–265  
     XML name value, 292  
 namespace (atomic type), 12, 42  
 namespace (constructor), 376–377  
 namespace (expression), 377

- Namespace declarations
  - declare default element namespace, 347
  - declare default function namespace, 347–348
  - declare namespace, 349–350
  - direct constructors, 160
  - in query prolog, 119–120
  - static context, 74–75
- Namespaces
  - functions, 54, 87, 89, 430
  - get-into-scope-namespace-prefixes( ), 413–414
  - get-namespace-uri-for-prefix( ), 417
  - get-namespace-uri-from-QName( ), 417–418
  - namespace (atomic type), 12, 42
  - navigation, 81–84
  - node construction methods, 160–161
  - nodes, 53–54, 430
  - XQuery standard and, 252
- namespace-uri( ) function, 54, 431
- NaN values, 147–148
- Navigation, 63–86
  - axes, 65–66, 68–69
  - complexities, 81
  - constructed nodes, 161–162
  - context, 72–75, 85
  - examples, 75–81
  - functions, 71–72
  - further reading, 86
  - namespaces, 81–84
  - node comparisons and, 104
  - node identity, 84–85
  - node tests, 66–68
  - overview, 63
  - path beginnings, 64–65
  - path types and, 28–30
  - paths, 63–64
  - predicates, 69–71
- Navigation idioms, 203–209
  - ancestor selection, 205–206
  - case-sensitivity and, 203–204
  - elements and attributes selected simultaneously, 203
  - leaf element selection, 205
  - maximum depth of subtree, 207–209
  - node selection by type, 204–205
  - sibling selection, 207
  - testing if nodes are in same tree, 205
- navigation operator (//), 324–325
- NCName (atomic type), 292–293
- ne (value comparison operator), 377
- negativeInteger (atomic type), 293
- Nested FLWOR, 235–237
- Nested predicates, 70–71
- .NET collation, 175
- NFC (normalization form C), 171
- nilable, 378
- nilled property, 56
- NMTOKEN (atomic type), 293
- NMTOKENS (atomic type), 293
- Node comparison operators
  - comparing expressions, 103–104
  - defined, 98
  - as kind of comparison operators, 28
- Node identity
  - as barrier to query optimization, 246–247
  - navigation and, 84–85
  - property, 53, 266
- Node kind
  - property, 52–53
  - tests, 67–68
- Node tests
  - defined, 64
  - name tests, 66–67
  - node kind tests, 67–68
  - types of, 66
  - wildcards and, 68
- Node types
  - hierarchy, 53–54
  - item classification and, 41
  - list of, 51
  - names, 54–55
  - overview, 41–42
  - properties, 52–53, 56
  - values, 55–56
- node( ) function
  - type matching and, 41–42, 58
  - XQuery type system and, 11–12

## 496 Index

node-name() function, 431–432

### Nodes

collection() function, 397

constructors, 19, 266

idref() function, 423

as item kind, 263

local-name() function, 427–428

name property, 54

name() function, 430–431

namespace-uri() function, 431

navigation idiom for selecting by type, 204–205

navigation idiom for testing if in same tree, 205

node sequences vs. node siblings, 230–231

node-name() function, 431–432

properties, 267, 268

sequences vs. siblings, 230–231

value comparisons and, 100

XQuery support for, 266

Non-breaking space character, 164

nonNegativeInteger (atomic type), 294

nonPositiveInteger (atomic type), 294

### Normalization

normalization form C (NFC), 171

normalizedString (atomic type), 295

normalize-space() function, 172, 431

normalize-unicode() function, 171, 432

Unicode normalization of strings, 432

W3C Character Model, 171–172

normalizedString (atomic type), 295

normalize-space() function, 172, 431

normalize-unicode() function, 171, 432

not() function, 24, 228–229, 432–433

### Notation

exponential, 357

NOTATION (atomic type), 295

sequences, 23

variables, 4

NOTATION (atomic type), 295

number() function, 434

Numbers. *see also* Integers

absolute values, 391

arbitrary-precision, 44

ceiling(), 395–396

floor(), 410–411

number() function, 434

numeric constants, 18–19

numeric entities, 163–164

numeric predicates, 69–70

random number generation (RNG), 220–221

round(), 438–440

XML, 238–239

### Numeric types, 43–46

background of, 43–44

converting to/from, 183–184

hierarchy of, 45

listed with ranges, 270

promoting, 59–60, 112

in XQuery, 44–46

## O

Occurrence indicators, sequences types, 41, 264

one-or-more() function, 434–435

One-to-one joins, 136

Operators, 23–28. *see also* Expression reference

arithmetic, 25

comparison, 27–28

logic, 24

overview, 23–24

precedence, 470–471

text, 25–27

type. *see* Type operators

or (logical operator), 378

order by clause, 365. *see also* FLWOR expressions

Order comparison operators (<<, >>), 98, 103–104

Outer joins, 139–140

Overloading functions, 87, 256

## P

Parameters, xQuery Serialization, 243

parent (axis), 65–66, 378–379

parent navigation (..), 323

Parent nodes, 53

Parentheses. *see* parentheses ( )

Paths. *see also* Navigation

- beginnings, 64–65
- grammar, 239
- history and development of, 63
- navigation operators, 323–324
- overview, 28–30
- path expressions for existential comparisons, 142
- sequence of steps in, 63–64

Performance benchmarks, XQuery, 259

Permutations, of sequence, 212

plus (+), 175, 238

position( ) function, 435

positiveInteger (atomic type), 91, 295–296

Post-Schema-Validation Infoset (PSVI), 39

pow( ) example, 92

pragma (expression), 380

pragma/extension (::, 315, 326

preceding (expression), 380

preceding-sibling (axis), 68, 207, 379–380

Predicates

- abbreviated axes and, 230
- boolean, 70
- bracket notation for, 337, 338
- defined, 64
- indicating sequences not strings, 228
- numeric, 69–70
- overview, 69
- selecting members from sequences, 108
- successive and nested, 70–71

prefix:\*, node test wildcards, 68

Prefixes, qualified names, 82

preserve (expression), 381

Preserve value, XML space declaration, 117

Primitive types, XML Schema, 11

Principal node kind, axes, 68

processing-instruction (constructor), 160, 379–380

processing-instruction( ) test, 67

Processing instructions

- construction methods, 160, 379–380
- nodes, 67

Processing model, static typing vs. dynamic typing, 15

Programming languages

- compared with query languages, 5–6
- descriptive nature of, 6

Programs, XML and, 9–10

Prolog section, of queries

- base URI declaration, 117–119
- global variable declaration, 121
- module imports and declaration, 121–122
- namespace declaration, 119–120
- overview, 16–17
- schema imports and validation, 122–123
- version declaration, 117
- XML space declaration, 117
- XQuery anatomy, 14
- XQuery standard and, 252

PSVI (Post-Schema-Validation Infoset), 39

Punctuation, 238

## Q

QName (atomic type), 50, 296–297

Qualified names

- Calendar types and, 50–51
- compared with expanded names, 81–84
- expanded-QName( ), 409–410
- get-local-name-from-QName( ), 414
- get-namespace-uri-from-QName( ), 417–418
- overview, 50–51
- QName (atomic type), 50, 296–297
- resolve-QName( ), 437

Quantification

- FLWOR expressions, 133–134
- regular expressions, 456
- satisfies, 381
- some, 382

Queries

- body section of, 14
- dividing into modules, 17
- prolog section of, 14, 16–17

## 498 Index

Query languages  
 compared with programming languages, 5–6  
 compared with XQuery, 6–9  
 declarative nature of, 6

Query optimization, 245–250  
 barriers to, 246–249  
 common approaches to, 245–246  
 early evaluation, 246  
 error preservation, 249  
 Formal Semantics, 250  
 further reading, 250  
 lazy evaluation, 245  
 node identity and, 246–247  
 overview, 245  
 sequence order and, 247–248  
 side effects, 249  
 streaming and database evaluation, 246

Query prolog. *see* Prolog section, of queries

question mark (?), 335–336

quotation mark. *see* “ (quotation mark)

quote entity reference (&quot;), 312

quote escape sequence (“”), 308

## R

Random number generation (RNG), 220–221

range operator, 107–108

Recursive functions, 92–93. *see also* Recursive functions

Regular expressions, 455–462. *see also* Expressions

Expressions

- advanced, 459–460
- basic, 456
- character properties, 461–462
- escaping, 457–459
- language (or grammar), 460–461
- modifiers, 457
- overview, 455–459
- quantifiers, 456
- replacing strings with, 436
- strings matching, 428–429

Relative paths, 65

Reluctant quantifiers, 456

remove() function, 435–436

replace() function, 435–436

resolve-QName() function, 436

resolve-uri() function, 437–438

Result tree fragments, 161

return clause, 366–367. *see also* FLWOR expressions

reverse() function, 437

Root nodes

- navigating to, 438
- of trees, 266
- XQuery data model hierarchy, 53

Root value, XQuery Serialization, 242–243

root() function, 65, 437

round() function, 114, 218, 438–439

round-half-to-even() function, 439–440

Rounding

- arithmetic expressions, 114–115
- arithmetic idiom for, 218–220
- round() function, 438–439
- round-half-to-even() function, 439–440

## S

satisfies (expression), 381

Scale, of decimals, 44

Schema imports

- import schema (expression), 371–372
- query prolog and, 122–123
- user-defined types and, 189

Schema types, 10

Schema validation, 190–192, 346

Scores, in full text searches, 258

Searches, full text, 258

self (axis), 65–66, 381–382

Self-joins, 141

Sequence idioms, 209–212

- permutations, 212
- selecting every other member, 211
- union, intersection and difference, 209–211

Sequence types, 263, 264

- Sequences
  - atomization of, 401
  - average values, 394
  - built-in functions, 106
  - comparisons, 104–105. *see also* Comparisons, sequences
  - computing length of, 109, 399–400
  - constructing, 105–108
  - containing more than one item, 453
  - content rules, 166–168
  - empty, 406
  - existence of, 409
  - general comparison operators and, 101
  - input nodes, 424
  - inserting subsequences into, 425
  - joining with FLWOR expressions, 134–141
  - maximum values in, 429
  - minimum values in, 430
  - one-or-more() function, 434–435
  - order as barrier to query optimization, 247–248
  - overview, 105
  - parts of, 41
  - passing to functions, 89
  - predicates indicating sequences not strings, 228
  - processing, 108–110
  - removing, 435–436
  - serialization, 242
  - subsequences of, 444
  - type conversion and, 58–59
  - XQuery data model and, 40–41
  - XQuery values, 11
- Serialization. *see* XQuery Serialization
- short (atomic type), 297
- Siblings
  - navigation idiom for selecting, 207
  - Nodes, 230–231
- Side effects, as barrier to query optimization, 249
- single quotes ('). *see* ' (apostrophe)
- Single types, 263
- Single-precision floating-point values, 284–285
- Singletons
  - exactly-one(), 408–409
  - sequences and, 11
  - XQuery data model, 40
- SiXDMML, 257
- skip (expression), 382
- slash (/). *see* / (slash)
- some (expression)
  - existential comparison and, 142
  - FLWOR expressions, 133
  - overview, 382
- Sort keys, 146
- Sorting
  - ascending, 341
  - descending, 355
  - empty greatest, 359
  - empty least, 359
  - FLWOR expressions, 146–148
  - stable, 382
- Space. *see* Whitespace
- special characters, XQuery, 163
- SQL (Structured Query Language)
  - compared with XQuery, 8–9
  - comparing FLWOR expressions to SQL statements, 126–127
  - relational data and, 3
- square brackets ([]), 337–338
- Square roots, 221
- stable (expression), 147, 382
- Standards, XQuery, 251–254
  - additional types, 252–253
  - built-in functions, 254
  - data manipulation, 256–258
  - features projected for version 1.1, 255–256
  - full text searches, 258–259
  - further reading, 258–259
  - modules and prolog, 252
  - namespace values, 252
  - simplifications, 253–254
  - standards roadmap, 254–255
- starts-with() function, 88, 440–441
- Static context
  - collations, 75
  - function declaration, 75

## 500 Index

- Static context (*continued*)
  - namespace declaration, 74–75
  - navigation, 72
  - variable declarations, 74
- Static errors, 33
- Static types, 11, 15, 179
- Streaming evaluation, 246
- strict (expression), 383
- string (atomic type), 42, 46, 297–298
- string( ) function, 441–442
- string-join( ) function, 175, 201, 442
- string-length( ) function, 26, 443
- Strings
  - canonical representation, 181–182
  - code points, 170, 443
  - collation parameter, 26–27
  - computing length, 443
  - concatenation, 26, 175, 398, 442
  - converting to, 440–441
  - ending with substring, 406–407
  - equality comparison, 397–398
  - functions, 175
  - overview, 18
  - predicates and, 228
  - regular expressions matched to, 428–429
  - regular expressions replacing, 436
  - string (atomic type), 42, 46, 297–298
  - string( ) function, 440–441
  - string-join( ) function, 175, 201, 442
  - string-length( ) function, 26, 443
  - string-to-codepoints( ) function, 26, 443
  - substrings at start of, 440–441
  - substrings of, 399, 444–445
  - substrings preceding/following, 445–447
  - text idiom for reversing, 200–201
  - text operators and, 175
  - tokenizing, 449–450
  - translating characters into other characters, 450–451
  - type conversion, 183
  - types, 46
  - Unicode normalization and, 433
    - whitespace normalization and, 295
- string-to-codepoints( ) function, 26, 443
- string-value nodes, 55–56
- strip (expression), 383
- Strip value, XML space declaration, 117
- Structured queries, 258
- Structured Query Language. *see* SQL (Structured Query Language)
- Stylistic considerations, query composition, 195–200
  - brace placement, 198–199
  - case, 195–196
  - conciseness, 199–200
  - modularizing XML into separate functions, 199
  - space, 196–197
- subsequence( ) function, 444
- Subsequences, 425, 444
- subtraction (-), 321–322
- substring( ) function, 26
  - function reference (fn:), 444–445
- substring-after( ) function, 445–446
- substring-before( ) function, 446–447
- Substrings
  - captured, 459
  - contained, in strings, 399
  - strings ending with substring, 406–407
  - strings starting with, 440–441
  - substring( ) function, 26, 444–445
  - substring-after( ) function, 445–446
  - substring-before( ) function, 446–447
- subtract-dateTimes-yielding-dayTimeDuration( ) function, 447–448
- subtract-dateTimes-yielding-yearMonthDuration( ) function, 448
- Subtraction, 110
- Sub-tree pruning, 231–232
- Subtype substitution, 59, 182
- Successive predicates, 70–71
- sum( ) function, 114, 448–449
- Syntax
  - arithmetic expressions, 110–111

- attributes, 155, 195–196
  - confusions, 237
  - entity references, 163
  - errors, 238
  - FLWOR expressions, 199
  - Keywords, 195
  - regular expressions, 455
- T**
- Text, 169–177
    - character encoding, 172–174
    - character models, 169–170
    - code points, 170–171
    - collations, 174–175
    - constructor, 383–384
    - functions, 175–177
    - further reading, 177
    - normalization, 171–172
    - operators, 25–27, 175
    - overview, 169
    - text (constructor), 383–384
    - text() test, 67, 157–158
  - Text idioms, 200–203
    - ASCII characters, 202–203
    - comma-separated values, 201
    - lower-, upper-, and title-case, 201–202
    - reversing strings, 200–201
  - then (expression)
    - instance of, 187
    - order of evaluation in, 34
    - overview, 384
  - Three-valued logic, 216–217
  - time (atomic type), 47–48, 184–185, 298
  - Time functions. *see* Date/time functions
  - Time zones
    - adjust-dateTime-to-timezone(), 392–393
    - adjust-date-to-timezone(), 392
    - adjust-time-to-timezone(), 393–394
    - Calendar types and, 49
    - implicit-timezone(), 85, 423
  - Title-case, 201–202
  - to (expression), 107–108, 384
  - token (atomic type), 298–299
  - tokenize() function, 449–450
  - TPC, benchmarking, 259
  - trace() function, 450
  - Transformations, XML, 153
  - translate() function, 450–451
  - treat as (expression)
    - converting up/down in type hierarchy, 182
    - down-casting with, 186
    - overview, 384–385
    - as type operator, 180, 186–187
  - Trees
    - comparisons, 104–105
    - nodes of, 266
    - root of, 65
  - true() function, 17–18, 451
  - Tuple space, FLWOR expressions, 129–133
  - Type conversion
    - across type hierarchy, 182–183
    - atomization, 57
    - binary types, 186
    - boolean types, 183–184
    - calendar types, 184–186, 273
    - duration types, 272
    - EBV, 57
    - functions, 89–90
    - general comparisons and, 102
    - hidden complexities of, 232–233
    - numeric types, 59–60, 183–184, 271–272
    - overview, 56–57, 181–182
    - sequence types, 58–59
    - simplifying, 253–254
    - string types, 183
    - subtype substitution, 59
    - up/down in type hierarchy, 182
    - value comparisons and, 99
  - Type operators, 179–192
    - cast and castable, 180–181
    - conversion rules, 181–184
    - further reading, 192
    - instance of, 187–188

## 502 Index

Type operators (*continued*)  
 overview, 179  
 treat as, 186–187  
 typeswitch, 187–188  
 user-defined types, 188–192

Type system, 11–14. *see also* Atomic types  
 additional types in new versions of XQuery,  
 252–253  
 atomization, 57  
 boolean types, 42–43  
 calendar types, 46–50  
 camel-case syntax of, 195–196  
 constructors, 19  
 EBV, 57  
 idioms, 212–215  
 kinds of types, 263  
 names of sequence types, 41  
 node hierarchy, 53–54  
 node kinds, 51, 266–268  
 node names, 54–55  
 node properties, 52–53, 56  
 node type values, 55–56  
 numeric type promotion, 59–60  
 numeric types, 43–46  
 overview, 263–265  
 primitive type conversion, 271–273  
 qualified name type, 50–51  
 sequence type matching, 58–59  
 string types, 46  
 structure of, 40–42  
 subtype substitution, 59  
 type checking, 33  
 type system (atomic type), 42, 268–271  
 type( ) function, 385  
 untyped data, 42  
 value comparisons and, 100  
 XML nodes and atomic values, 11–13  
 XML Schema types that are not needed, 13–14

type system (atomic type), 42, 268–271  
 type( ) function, 385  
 Typed XML, 10  
 typed-value, nodes, 55

typename (value), 180  
 Types, user-defined, 51  
 typeswitch (expression)  
 order of evaluation in, 34  
 overview, 385–388  
 as type operator, 187–188  
 variable declaration and, 74

## U

UCS (Universal Character Set), 169. *see also*  
 Unicode characters

UDFs. *see* User-defined functions

Unary operators  
 overview, 110  
 unary minus (-), 320–321  
 unary plus (+), 318

Unicode characters  
 code points, 27, 170–171  
 normalization, 171–172  
 notational conventions, 4  
 overview, 169–170  
 Unicode normalization of strings, 432

Unicode code points  
 codepoints-to-string( ), 396–397  
 collation, 174, 203  
 regular expression syntax and, 455

Unicode Collation Algorithm, 175, 177

Uniform Resource Identifier. *see* URI (Uniform  
 Resource Identifier)

union (expression), 106, 388  
 union operator (|), 209–211, 339, 455

Unique identifiers, 288  
 unique-id( ) function, 56

Universal Character Set (UCS), 169. *see also*  
 Unicode characters

Universal comparison of sequences, 145–146  
 unordered( ) function, 110, 452  
 unsignedByte (atomic type), 299  
 unsignedInt (atomic type), 300  
 unsignedLong (atomic type), 300–301  
 unsignedShort (atomic type), 301–302

Untyped XML data, 10, 42  
 untypedAtomic (atomic type), 42, 55, 302–303  
 Up-casting operators, 186  
 Upper-case  
   text idiom for, 201–202  
   upper-case() function, 201, 204, 452  
 URI (Uniform Resource Identifier)  
   anyURI (atomic type), 274–275  
   document-uri(), 406  
   escape-uri(), 408  
   get-namespace-uri-for-prefix(), 417  
   namespace-uri(), 431  
   resolve-uri(), 437–438  
 User-defined functions  
   declare function, 348–349  
   overview, 17, 90–92  
   recursion, 92–93  
 User-defined types, 51, 188–192, 256  
 UTF-16, 171, 172–173  
 UTF-8, 172

## V

validate (expression), 190–191, 389  
 Validation  
   constructed XML and, 163  
   declaration, 122–123, 350–351  
   lax mode, 373–374  
   schema validation, 190–192  
   skip mode, 382  
   strict mode, 383  
   type(), 385  
   validate (expression), 190–191, 389  
   whitespace and, 166  
   XML Schema, 10–11  
 Value comparison operators  
   comparing expressions, 98–100  
   defined, 98  
   eq, 359–360  
   ge, 367  
   gt, 368  
   le, 374

list of, 99  
 lt, 374–375  
 ne, 377  
 string values and, 175  
 type conversion and, 99  
 as type of comparison operator, 27

## Values

distinct-values(), 404–405  
 node types, 55–56  
 sum of, 448–449

## Variables

bindings, 74  
 declaring, 121, 351–352  
 FLWOR expressions, 128–129  
 lower-case syntax of variable names, 195  
 overview, 30–31  
 quantification introduces multiple, 133–134  
 static context, 74  
 variable (\$), 30–31, 308–309

Variable-width character encoding, 172

Vector operations, 223–224

## Version

declaring in query prolog, 117  
 version (expression), 389  
 xquery version, 390

version (expression), 389

Vertical bar (|). *see* union operator (|)

Views, data organized into, 153

## W

W3C (World Wide Web Consortium)

  Character Model, 169–170, 177

  development of XQuery, 3

  normalization, 171–172

  XQuery standards, 254–255

where clause, 364–365. *see also* FLWOR expressions

## Whitespace

  characters, 15–16

  content rules and, 164–166

  declaring, 117, 352

## 504 Index

- Whitespace (*continued*)  
   normalize-space( ), 432  
   strip mode, 383  
   stylistic considerations in use of, 196–197
- Wildcards  
   node tests, 68  
   regular expressions and, 456  
   symbol (\*), 316
- Word breakers, in full text searches, 258
- World Wide Web Consortium. *see* W3C (World Wide Web Consortium)
- X**
- xdt prefix, 42. *see also* Atomic type reference
- XMark, benchmarks, 259
- XML  
   as API, 5  
   attribute value normalization, 172  
   data access with doc( ) function, 29  
   data model. *see* Data models, XML  
   document loading, 405–406  
   modularizing into separate functions, 199  
   node types supported in XQuery, 19  
   numbers, 238–239  
   outputting XQuery as, 19–22  
   space declaration in query prolog, 117  
   space policy, 165  
   typed vs. untyped, 10  
   unifying documents, databases, and programs, 9–10
- XML 1.0 Recommendations*, 255
- XML Information Set (Infoset), 38–39
- XML Query Requirements*, 254
- XML Query Use Cases*, 254
- XML Schema  
   imports, 189  
   namespace, 42  
   types not needed by XQuery, 13–14  
   validation, 163, 190–192  
   XQuery type system and, 10–11
- XML, constructing, 153–168
- attribute nodes, 155–157, 342–343  
   character escapes, 163–164  
   comment nodes, 159, 322, 328, 345–346  
   composition of constructed nodes with navigation, 161–162  
   content sequence, 166–168  
   document nodes, 158–159, 356–357  
   element nodes, 154–155, 325, 327–329, 358–359  
   further reading, 168  
   namespace nodes, 160–161, 376–377  
   overview, 153  
   processing instruction nodes, 160, 331, 336, 379–380  
   text nodes, 157–158, 383–384  
   validation, 163  
   whitespace, 164–166
- XOO7, benchmarks, 259
- XPath  
   compared with XQuery, 6–7  
   complexity of, 227  
   data model, 38  
   string functions, 175
- XQuery, 3–32  
   comments, 15–16, 314–315, 325  
   constants, 17–19  
   data model. *see* Data models, XQuery  
   DML, 256–257  
   documents and databases and, 9–10  
   error handling, 33–34  
   FLWOR expression, 31–33  
   Formal Semantics, 250  
   functions, 22–23  
   further reading, 34  
   implementation options, 3–4  
   notational conventions, 4  
   operators, 23–28  
   overview, 3  
   paths, 28–30  
   performance benchmarks, 259  
   processing model, 15  
   prolog section, of queries, 16–17

- query languages compared with programming languages, 5–6
  - sample query, 14
  - type system, 11–14
  - variables, 30–31
  - version, 390
  - whitespace, 15–16
  - XML output and, 19–22
  - XML Schema and, 10–11
  - XQuery compared with other query languages, 6–9
  - XQuery 1.0 and XPath 2.0 Data Model*, 255
  - XQuery 1.0 and XPath 2.0 Formal Semantics*, 255
  - XQuery 1.0 and XPath 2.0 Functions and Operators*, 254
  - XQuery 1.0: An XML Query Language*, 254
  - XQuery Serialization, 241–244
    - overview, 241–242
    - parameters, 243
    - root value, 242–243
    - sequences of values, 242
    - XQueryX, 243–244
  - xquery version (expression), 390
  - XQueryX, 243–244
  - xs prefix, 42. *see also* Atomic type reference
  - XSLT
    - compared to FLWOR, 127–128
    - compared with XQuery, 7–8
    - result tree fragments, 161
    - XQueryX and, 243
  - XSLTMark, 259
  - XUpdate, 258
- Y**
- yearMonthDuration (atomic type), 303
- Z**
- zero-or-one( ) function, 453

















