

## Symbols

{ } syntax 247

## A

Abstract Factory, in persistence layer design 330

access, property 68–70

ACID, applicability to conversations 337

ActiveWriter 293

Adapter pattern 314

AddAssembly() 40

AddCategory() 61

AddChildCategory() 60  
method 119

AddClass() 39

AddXmlFile() 39

ADO.NET 7, 9, 21, 34

connection management 111

connection pool 274

database access

configuration 41–44

DataSet. *See* DataSet

Entity Framework 11

ADO.NET connection 41

ADO.NET Entity

Framework 18–20

ADO.NET IDbConnection. *See*

IdbConnection

aggregate function 234

aggregation 82, 234

alias 216

naming convention 216

using with join 226–228

all quantifier 242

Ambler, Scott 92

analysis 52–53

and, logical operator 221

<any> 204–205

any quantifier 242

API (application programming interfaces)

APIs

Auditable 278

Criteria 208

FetchMode 224

Interceptor 280

MatchMode 220

UserType 302–303

CompositeUserType 176–178

UserType 173–176

application

architecture 260, 320

layered, design of 320, 336

legacy, porting to

NHibernate 215

application transaction 146

avoiding reassociations with new sessions 340–344

choosing implementation

approach 344

implementing the hard

way 337–338

use case 336

using detached objects

338–340

using long sessions 340–344

architecture 33–38

coupling 10

layered 7–9

three layers 9

argument, binding

arbitrary 213–214

ArgumentNullException 276

arithmetic expressions, support for 219

asc, ordering query results 222

ASP.NET 13, 342

applications, session

management 332–335

session, storing NHibernate

session in 340

sharing sessions, and 327

assembler, DTO 346

Assert class 264

association 12, 58, 82, 86–91

bidirectional 58, 85, 88–90

cardinality 60

defined 189

fetching 224–225

foreign key 190–192

inverse 89

joining 225

managed 59

many-to-many 60, 193–200

many-to-one 65, 87

polymorphic 201–202

mapping 189–200

for lazy initialization 250

multiplicity 86–87

one-to-many 87–88, 198–199

one-to-one 189–193

parent/child 90–91

polymorphic

mapping 200–205

table-per-concrete-

class 204–205

polymorphism 92

association (*continued*)  
 primary key 192–193  
 simplest 87–88  
 single point 127–129  
 ternary 197  
 unidirectional 85–86, 199  
 many-to-one 87–88

association class 193

association table 130, 193

association-level cascade style 116

associations, joining 222–231

asynchronous  
 programming 274

atomic 111

atomicity, guaranteeing 343–344

attribute 349  
 Column 290  
 foreign-key 290  
 index mapping 290  
 Length 290  
 not-null 290  
 sql-type 290  
 Unique 290  
 unique-key 290  
 XML mapping 289

attribute-oriented  
 programming 65–66

audit log 277

audit logging 277–284  
 automated 278  
 manually 278

Auditable API 278

AuditLog 278

AuditLog.LogEvent() 282

AuditLogRecord class 279

automated persistence 55–56

automatic dirty checking 33, 103, 113

availability 273

avg() function 234

## B

backtick 72

<bag> 196

bag collection 182, 194, 196  
 with set semantics 198–199

batch fetching 127, 250

batch update 113

BeginTransaction() 136–137

behavior entity 54

bidirectional 88–90

BindingList 314

BindingSource 314

BLOB 171

bottom-up development 293

bug-solving process 274

bug-tracking website 353

business entity 7  
*See also* domain model

business key 108–110

business layer 8, 266–268  
 business logic in 310  
 implementing 266  
 implementing manual version checks 338  
 testing 268

business logic 7  
 example in domain model 336  
 implementing 309–312  
 tiers of, separating from web tiers 345

business logic layer 8

business model 52

business object 5  
*See also* domain model

business rule 321  
 encapsulating in domain model 324  
 vs. test 312  
*See also* business logic

business transaction 146  
*See also* conversation

by value equality 108

## C

cache 104, 152  
 cluster scope 153  
 distributed 162  
 expiration policy 254  
 first level 156  
 managing 157  
 miss 153  
 policy 157  
 process scope 153  
 region 161  
 second level 157  
 controlling 164  
 timestamp 254  
 transaction scope 153

cache architecture 155–159

cache provider  
 choosing 159  
 Hashtable 159  
 local, setting up 161  
 MemCache 163

NCache 163

Prevalence 159

SysCache 159

caching 18, 142, 152–164  
 example 159  
 good and bad candidates for 155  
 maintaining consistency across clusters 322  
 methods used for lookups 300  
 object identity, and 154  
 queries 253–255  
 reference data 155  
 strategies 153–155  
 transaction isolation, and 154

callback API 36–37

CallContext API 327

candidate key 79

Cartesian product 223, 229, 350

cascade attribute 90

cascade style 116–117

cascading delete 90

cascading save 33, 90

CASE 52

CaveatEmptor 52–54  
 domain model 53–54

changes detection 103, 113

check constraint 289

class  
 association 193  
 coarse-grained 167  
 component, writing 187  
 DTO 345  
 entity 167  
 fine-grained 167  
 helper 321  
 immutable 71  
 parallel hierarchy 346  
 utility 321  
 value type 167  
 wrapping 314

class name qualification 74–75

ClassToTableName() 73

CLOB 171

cluster scope cache 153

cluster-safe design 322

coarse-grained transaction 145

code generation 10  
 bottom-up development, and 293  
 for domain model 12

code smell 346

CodeDom provider 70

CodeGenerator class 292

- CodeSmith 293
  - collection
    - bag 182, 194, 196
    - with set semantics 198–199
  - columns, avoiding
    - not-null 188
  - component 186
    - using for many-to-many association 196–198
  - fetching 128, 225
  - filtering 240–242
  - indexed 199
  - indexed map 194
  - list 183, 194, 196
  - map 184
  - nonindexed 198
  - ordered 184, 186
  - persistent 184
  - polymorphic 203
  - set 182
  - sorted 184
  - wrapper 129, 131
  - collections
    - comparison by identity 62
    - component 189
    - value-type, mapping 189
  - column 67
    - avoiding not-null 188
    - inconvenient type 302–303
  - Column attribute 290
  - <column> element 289
  - ColumnName() 73
  - COM+ assembly, distributed
    - NHibernate 348
  - command, custom 248
  - commit 135
  - Commit() 111, 137
  - CompareTo() 185
  - comparison operators and
    - restriction 218
  - component 82–86
    - collections of 186, 190
    - using for many-to-many association 196–198
  - composite identifier class
    - 299–301
  - composite key 80, 296
    - mapping table with 298–302
    - referencing entity with 301
  - <composite-id> mapping 297
  - CompositeUserType API
    - 176–178
  - composition 82
    - and minimizing redundancy 329
  - conceptual view 52
  - concurrency
    - database capabilities 7
    - optimistic checking, issue with
      - batching 113
      - reducing 345
  - concurrency strategy 158
    - nonstrict-read-write 158
    - read-only 158
    - read-write 158
  - concurrent requests, problems
    - with 343
  - concurrent transactions 339
  - Configuration 34–35
  - configuration
    - app.config file 44–46
    - connection string 46
    - database access 42–43, 71
    - NHibernate 31–32, 38–44
      - advanced 44–48
    - reflection 70–71
    - schema 46
    - steps 43
    - techniques 40
  - configuration document 31
  - Configuration.SetProperty() 41
  - Configure() 41
  - connection pool 41
  - connection string 46
    - securing 273
  - connection-release mode 139
  - consistency 135
  - control logic, separating from
    - data access code 324
  - control, data bound 314
  - controller, as part of business
    - layer 266
  - conversation 16–18
    - approach, choosing 344
    - ASP.NET
      - implementation 342
      - example 146
      - guaranteeing atomicity of 343–344
      - implementing 335–344
      - loading objects on each request 337–338
      - using detached objects 338–340
      - using the session-per-conversation pattern 340–344
      - vs. transaction 337
      - working with 146–152
  - core interfaces 35–36
  - correlated subquery 242
  - count() function 234
  - create command, custom 248
  - create, read, update, delete (CRUD) 7
    - hand-coded 14–15
    - operations,
      - implementing 14–15
    - with DataSets 14
    - with NHibernate 14
    - with the data access object pattern 325
  - CreateAlias() 227
  - CreateCriteria() 209, 227
  - createQuery() 208
  - createSQLQuery() 208
  - creating object. *See* persistence
  - Criteria API 19
    - comparison operators 218
    - FetchMode 224
    - implicit joins 228–229
    - logical operators 221
    - MatchMode 220
    - nesting 227
    - polymorphic queries 217
    - purpose of 208
    - QBC, and 208
    - QBE, and 239
    - restriction 217
    - results, ordering 221
    - SQL function calls, and 220
    - string matching 220
    - theta-style joins 230
    - where clause, and 219
  - criteria query
    - comparison operators 219
    - wildcard search 220
  - cross join 350
  - cross-cutting concern 55
  - CRUD. *See* create, read, update, delete (CRUD)
  - current\_session\_context\_class 327
  - CurrentSessionContext 328
  - custom mapping type 173
  - custom type 62
  - custom type API 37
- 
- D**
- data
    - deprecated, ignoring 218
    - externalization 345
  - data access approaches 6
  - Data Access Object pattern 324–326

- data binding 312–316
    - manual 313
    - using data-bound controls 314
    - using NHibernate 315
    - with ObjectViews 315
  - Data Definition Language (DDL) 349
  - data integrity, database capabilities 7
  - data language 349
  - Data Manipulation Language (DML) 350
  - data storage. *See* persistence
  - Data Transfer Object (DTO) 345
    - assembly 346
    - data transfer, and 346
    - need for, questioning 345–346
    - problems with 346
  - Data Transfer Object pattern 345
  - database
    - generation, executing arbitrary SQL during 291
    - identity 76–79
      - persistent object, and 102
    - live, updating 295
    - lock table 149
    - schema maintenance, automatic 294–296
    - setup 27
    - subsystem 8
    - systems that work with NHibernate 352
    - trigger 205, 278
    - triggers for 303–305
    - update
      - first commit wins 147
      - last commit wins 147
      - merge conflicting updates 147
  - database schema generation 67
  - database transaction 135–146
  - database-independent application 23
  - data-bound control 314
  - data-integrity test 264
  - DataReaders 337
  - DataSet 6–7
    - as entity 12
    - association 16
    - filling with entity data 316–317
    - granularity 16
    - obtaining 237
    - presentation layer 13
    - typed 346
    - using LINQ 20
  - DataSets 337
  - DataTable, avoiding 7
  - DDL schema
    - creating with hbm2ddl 290–291
    - tools for 288
  - debugging 274–276
  - delete command, custom 248
  - Delete() 113
  - deleting object. *See* persistence
  - Dependency Injection pattern 284
  - desc, query ordering 222
  - design goal 15, 272–274
    - availability 273
    - manageability 273
    - performance 273
    - reliability 273
    - scalability 273
    - securability 273
  - design pattern 7
  - detached object 78, 339
    - in conversations 336
    - when to use 344
  - development
    - database 295
    - list 354
    - tools 287
  - development process 287–296
    - bottom-up 293
    - domain-centric 261
    - meet-in-the-middle 294
    - middle-out 292–293
    - top-down 288–292
  - Dialect 37
    - specifying 42
  - dirty checking 62
    - automatic 33
  - dirty read 140
  - Disconnect() 139
  - discriminator 94
  - discriminator column 205
  - distinct 233
  - distributed cache 274
  - distributed transaction 345
    - enabling 346–348
  - domain expert 52
  - domain model 6–7, 53, 263–266
    - "smart" 323–324
    - adding logic 61–63
    - association 16, 58–61
    - attached to persistence 103, 106
    - behavior 54
    - business logic in 310
    - component 82–86
    - creating 26–27
    - detaching from persistence 104
    - discriminator 94
    - distinguishing transient and detached instances 106, 120–121
    - fine-grained 81–86
    - granularity 15
    - hand-coding 12
    - identifier 18, 102
    - identity 16, 64, 76–81
    - identity implementation 107
    - identity scope 104–105
    - immutable 71
    - implementation 55–63
    - implementing 166, 173, 263
    - instances, working with 338
    - joined-subclass 94
    - mapping 18
    - mapping to given schemas 294
    - mutable 71
    - ORM without 54
    - persistence lifecycle 101–110
    - proxy 122
    - reattaching to persistence 106
    - referencing 16
    - state 54
    - subclass 94
    - testing 264
    - See also* type
  - domain-centric 261
  - Domain-Driven Design 325
  - domain-driven development (DDD) 261–262
  - Dont Repeat Yourself (DRY) 326
  - drag and drop 6
  - DTO. *See* Data Transfer Object (DTO)
  - durability 135
  - dynamic instantiation 232
- 
- E**
- eager fetching 126, 224–225
  - elements() function 241, 243
  - embedded resource 30

EmptyInterceptor 281  
 EnableLike() 239  
 Enterprise Library 284  
 Enterprise Services application,  
   using NHibernate in  
   345–348  
 entity 167  
   applying Observer pattern  
   to 307–309  
   binding persistent 213  
   data binding 312–316  
   hand-coding 12  
   hidden 168  
   implementing 11–13  
   lifecycles of 167  
   persistence-abstracted 307  
   referencing with composite  
   key 301  
   retrieving multiple types in a  
   query 245  
   root 209, 228  
   *See also* domain model  
 entity framework 20  
 entity query 244  
 entity.ToString() 279  
 EntityNameDAO 268  
 Enumerable() method 210, 252  
 Enumerable() query 252  
 enumerated type 180–181  
 enumeration 181  
 Environment.BuildBytecode-  
   Provider() 71  
 equality  
   by value 108  
   consistency 107  
   using business key 108–110  
   using database identifier 107  
   using versioning 108  
   vs. identity 76–77  
   *See also* identity  
 Equals() 76, 106–110  
 Equals(object o) 77  
 equivalence 76  
 error, understanding and  
   solving 274–276  
 Evict() 156, 164  
 exception  
   .NET, built-in 276  
   from external libraries 276  
   throwing 276  
   understanding 274  
 exception.ToString() 274  
 Execute() 291  
 Expression class 217  
 expression, SQL 218

Expression.And() 221  
 Expression.Conjunction() 221  
 Expression.Disjunction() 221  
 Expression.Or() 221  
 extension API 37–38

## F

failure  
   accessing databases 321  
   request checks 321  
 fetch 224  
 fetch attribute 251  
 fetching strategy 125–127  
   batch 127  
   collections and 129–130  
   eager 126, 225  
   fetch depth 130–131  
   global 130–131  
   immediate 126  
   lazy 126, 225  
   runtime association 225  
   selecting in mapping 127–132  
 field 26  
 fine-grained transaction 145  
 first commit wins 147  
 fixture 264  
 fluent interface 39  
 flush mode 139  
 Flush() 139  
   executing trigger 304  
 flushing 139  
 FlushMode 139  
 FlushMode.Never 343  
 foreign key 12, 81, 230  
 foreign key association 190–192  
 foreign-key attribute 290  
 formula 68  
 from clause 217, 242  
   fetch join 224  
   implicit, in collection  
   filters 240

## G

garbage collection 102  
 Gateway pattern 325  
 generation database 67  
 generics, and minimizing  
   redundancy 329  
 Get() 112, 122  
 GetClassMetadata() 76  
 GetCollectionMetadata() 76  
 GetHashCode() 106–110, 300

getNamedQuery() 214  
 global assembly cache (GAC) 43  
 granularity of a session 150–151  
 group by clause 234  
 grouping, database  
   capabilities 7

## H

HashedSet 252  
 Hashtable cache provider 159  
 having clause 236  
   rules governing 236  
 hbm2ddl (SchemaExport) 67  
   DDL schema generation  
   with 290–291  
   middle-out development  
   with 292  
   parameter descriptions 291  
   preparing mapping  
   metadata 288–289  
   top-down development  
   with 288  
   XML mapping attributes  
   for 289  
 hbm2net (CodeGenerator) 292  
   middle-out development  
   with 292  
 hbm2net.config file 293  
 HbmSerializer.Serialize() 40  
 Hello World 25–33  
 helper class 321  
 helper/utility classes 8  
 Hibernate mapping types,  
   system 167–181  
 Hibernate Query Language  
   (HQL) 19, 123–124, 208  
   aggregation, using 234  
   aliases 216  
   and joins 226–228  
   basic queries 215–222  
   collection filters 240–242  
   comparison operators 218  
   distinct results, getting 233  
   dynamic instantiation 232  
   expressing queries with 208  
   fetch join and 224–225  
   grouping 234  
   implicit joins 228–229  
   keywords, writing 216  
   logical operators 221  
   polymorphic queries 217  
   projection 232–234  
   restricting groups with  
   having 236

Hibernate Query Language (HQL) (*continued*)  
   restriction 217  
   results, ordering 221  
   SQL function calls, and 220, 233  
   string matching 220  
   subqueries 242–243  
   theta-style joins 229  
   where clause, and 219  
 hibernate.cfg.xml 41, 163  
 hibernate.connection.release\_mode 140  
 HibernateException 276  
 high coupling, avoiding 284  
 HQL. *See* Hibernate Query Language (HQL)  
 HTTP request context for ASP.NET session management 327  
 HttpContext API 327

---

**I**  
 IAuditable 279, 293  
 ICache 37  
 ICacheProvider 37  
 IClassPersister 37  
 ICompositeUserType 34, 37  
 IConnectionProvider 37  
 ICriteria 34, 110, 124, 208  
   introducing 36  
   method chaining 210  
   pagination 209  
   results, ordering 222  
 ICriteria API 121  
 ICriterion 124, 217  
 ICurrentSessionContext 327  
 <idbag> 196  
 IDbConnection 136  
   manual 41  
 identifier 18, 77  
   native generator 78  
 identity 16, 64, 76–81  
   .NET 107  
   implementation 107  
   process-scoped 104  
   reference equality 105  
   scope 104–105  
   transaction-scoped 104  
   vs. equality 76–77  
   *See also* equality  
 identity map 17  
   cache usage 18  
 Identity Map pattern 17

id-type attribute 204  
 IEditableObject 314  
 IEnhancedUserType 178  
 Iesi.Collections 59  
   Iesi.Collections.library 252  
   Iesi.Collections.ListSet 186  
   Iesi.Collections.SortedSet 185  
 ignoreCase() 239  
 IHttpModule interface 334  
 IdentifierGenerator 34, 37, 79  
 IInterceptor 34, 37, 278, 310  
 IInterceptor API 343  
 ILifecycle 34, 36  
 IList 196  
 immediate fetching 126  
 immutable 71  
 impedance mismatch. *See* paradigm mismatch  
 implicit join 228–229  
 in quantifier 242  
 INamingStrategy 290  
   introducing 72  
 inconvenient column type 302–303  
 index mapping attribute 290  
 indexed collection 199  
   and inverse= 196  
 indexed map collection 194  
 indices() function 243  
 InfrastructureException 138  
 inheritance 7, 12, 16, 91  
   mapping 91  
   mapping strategy 98  
   minimizing redundancy, and 329  
   table per class hierarchy 92–95  
   table per concrete class 92–93  
   table per subclass 92, 95–98  
 inner join 222  
 INotifyPropertyChanged 308, 314  
 InsensitiveLike() operator 239  
 insert  
   control 71  
   dynamic 71  
 insertion 350  
 instance, detached 106  
 instantiation, dynamic 232  
 integrating services 277–284  
 integrity, guaranteeing referential 115  
 interceptor  
   enabling 282  
   writing 280–282  
 Interceptor API 280, 282

interfaces, core 35–36  
 INullableUserType 178  
 inverse attribute 89  
 Inversion of Control  
   pattern 284  
 IParameterizedType 37, 178  
 IPropertyAccessor 37  
 IProxyFactory 37  
 IQuery 34, 56, 110, 208  
   binding arbitrary arguments with 213  
   introducing 36  
   method chaining 210  
   pagination 209  
 IS NULL 219  
 is null operator 214, 219  
 ISession 34, 56, 110, 150  
   as first-level cache 155  
   Close 103  
   Delete 103  
   Evict 104  
   introducing 35  
   Save 102  
   transparent write-behind 138  
   *See also* persistence  
 ISession API  
   obtaining new instances of 321  
   query shortcuts 211  
 ISession.Connection  
   property 273, 316  
 ISession.CreateSQLQuery() 244  
 ISession.Get() 218  
 ISession.Load() 218  
 ISessionFactory 110, 157  
   creating 38–41  
   instance, configuring 44  
   introducing 35  
   naming 46  
 ISessionFactory.GetCurrentSession() 326–329  
 isolation issues 140  
 isolation level 140–141  
   choosing 141–143  
   read committed 141  
   read uncommitted 141  
   repeatable read 141  
   serializable 141  
   setting 143  
 ISQLQuery API 244–246  
 ISQLQuery instance, creating 244  
 ISQLQuery.SetResultSetMapping() 247  
 iterate() 210

ITransaction 34, 37, 110,  
137–138, 146  
introducing 36  
ITransactionFactory 37  
IUserCollectionType 37, 178  
IUserType 34, 37  
IValidatable 34  
IValidatable interface 36

## J

---

JDBC connections 340  
join 222–231, 350  
  ANSI-style 222  
  fetch 224–225  
  from clause and 223  
  implicit 223, 228–229  
  inner 222  
  means of expressing 223  
  options 223  
  outer 223  
  table 223  
  theta-style 223, 229  
  using alias with 226–228  
  where clause, and 223  
join condition 223  
joined-subclass 94

## K

---

key  
  candidate 79  
  composite 80, 303  
    mapping table with  
    298–302  
    referencing entity with 301  
  foreign 190–192, 230  
  generation 79  
  natural 79  
  natural, mapping table  
    with 297–298  
  natural/primary 297–298  
  primary 79, 192–193  
    choosing 79–81  
    surrogate 79  
keyword, case sensitivity 216

## L

---

last commit wins 147  
latency 345  
layer  
  application 260–270, 320  
  building and testing 263

business logic. *See* business  
  logic layer  
interaction 101  
persistence, designing  
  320–335  
persistence. *See* persistence  
  layer  
presentation. *See* presentation  
  layer  
  separating business and  
  presentation 320  
layered architecture 7–9  
lazy association, initializing 131  
lazy fetching, avoiding 202  
lazy loading 17, 121, 126  
  always enable 275  
  application 123  
LazyInitializationException 131  
leakage of concerns 55  
legacy application, porting to  
  NHibernate 215  
legacy column, mapping with  
  custom type 302–303  
legacy data 296–305  
legacy database schema  
  296–305  
  changes needed 296  
  composite key mapping  
  298–303  
  integrating database triggers  
  with 303–305  
  natural key mapping 297–298  
  problems with 296  
  required changes 297  
Length attribute 290  
library migrations 295  
lifecycle state  
  detached 167  
  persistent 167  
  transient 167  
like operator and wildcard  
  searches 220  
link table 130, 193  
LINQ 20  
  over DataSet 20  
  to Entities 20  
  to NHibernate 20  
  to SQL 11  
<list> 196  
list collection 183, 194, 196  
List() 210, 252  
Load() 122  
<load-collection> 246  
loading objects on each request  
  in conversations 336

load-objects-on-every-  
  request 339  
lock mode 144  
lock table 149  
Lock() 111  
lock() 339  
locking 140, 143  
  optimistic 147  
    alternative  
    implementations 151  
    offline 147–149  
  optimistic and pessimistic,  
  compared 149  
  pessimistic 143–146  
LockMode 112, 144  
log record, mapping 279  
log4net 284  
  configuration 47  
  *See also* logging  
LogEvent() 278  
logging 40, 47–48  
  integrating 284  
logging library 284  
logic test 264–265  
logic, ternary 218  
logical expression,  
  constructing 218  
logical operator 221  
LogType 278  
long session 151, 340  
  in conversations 336  
  when to use 344  
long transaction 146  
lost update 140  
lower() function 220

## M

---

maintainability 7, 9, 22  
MakePersistent() 330  
MakeTransient() 330  
manageability 273  
managed relationship 59  
managed versioning, for  
  optimistic locking 147–149  
many-to-many association  
  60, 130  
  bidirectional 195–196  
  component collections used  
  for 196–198  
  mapping 193–200  
  mappings 168  
  tables 168  
  unidirectional 193–194

- many-to-one 87
    - mapping 30
  - many-to-one association 65, 229
    - implicit join 228
    - polymorphic 201–202
  - <map> 196
  - map collection 184
  - mapping
    - assembly 40
    - association 16, 30, 86–91, 189–200
    - unidirectional 86
  - attributes 65
  - basic 66–76
  - bidirectional many-to-many 195–196
  - choice 66
  - class 40
  - component collection 187
  - composite key 298–303
  - correctness, testing 269
  - creating 29–30
  - data type 65
  - domain model 18
  - entity class 168
  - error 40
  - inheritance 91
  - inheritance strategy 98
  - IntelliSense 46
  - legacy columns 302–303
  - log record 279
  - many-to-many
    - association 168, 193–200
  - metadata 63–66
  - metamodels 173
  - natural key 297–298
  - one-to-many association 198–199
  - one-to-one association 189, 193
  - property 66–68
  - runtime 75–76
  - schema 46
  - strategy 98
  - synthetic identifier 297
  - table per class hierarchy 92–95
  - table per concrete class 92–93
  - table per subclass 92, 95–98
  - ternary association 197
  - unidirectional many-to-many 193–194
  - unidirectional one-to-many 199–200
  - XML 63–65
- mapping attributes, XML
    - injection 84
  - mapping document 30
  - mapping file
    - Hibernate 215
    - working with 39
  - mapping metadata, preparation of 288–289
  - mapping type, Hibernate
    - basic 169, 172–181
    - built-in 169–171
    - custom, creating 173
    - date and time 170
    - enumerated 181
    - Java primitive 169
    - JDK 171
    - large objects 171
    - object 171
    - system 167–181
    - using 172–181
  - marker attribute 278
  - marshal-by-reference object 272
  - MatchMode API 220
  - max() function 234
  - maxelement() function 243
  - maximum fetch depth 130
  - maxindex() function 243
  - medium-trust policy, issues with 271
  - meet-in-the-middle
    - development 294
  - MemCache distributed cache provider 163
  - merge conflicting updates 147
  - metadata 63–66
    - manipulating at runtime 75–76
  - metamodel 173
  - meta-type attribute 204
  - method chaining 39, 210
  - method, parameter-binding 211–214
  - middle-out development 292–293
  - migrations library 295
  - Migrator 295
  - min() function 234
  - minelement() function 243
  - minindex() function 243
  - model, presentation 314
  - modeling, object vs. relational 21
  - model-view-controller (MVC) 266
  - Mono 352
  - multilayered architecture 260, 320
  - multiplicity 86–87
  - multiversion concurrency control 140
  - mutable 71
  - MyGeneration 293
  - MySQL 10
    - See also* relational database management system
- ## N
- 
- n+1 select problem 249–252, 275
  - named parameter 212
  - named query 214–215
    - used in persistence layer 331
  - named SQL query 246–248
    - calling stored procedures 247
  - namespace, default 74–75
  - naming convention 72–73
  - native SQL 243–249
    - queries 121
  - natural key 77
    - mapping 297
    - mapping table with 297–298
  - navigation, unidirectional 193–194
  - NCache distributed cache provider 163
  - .NET
    - built-in exceptions 276
    - data binding 313
    - database access 6
    - features, solving issues related to 270–272
    - generics 329
    - reflection 347
    - remoting 271
    - security policy, issues with 271
  - NHibernate 4
    - advanced configuration 44–48
    - alternatives 9
    - API 34
    - bug-tracking website 353
    - compatible database systems 352
    - configuration 38–44
    - development list 354
    - downloads 352
    - forum 353
    - identity scope 105

NHibernate (*continued*)  
 installing 25  
 learning curve 277  
 making sure it's the right tool 276  
 online resources 353  
 reasons to use 15–20  
 SourceForge website 352  
 starting 43–44  
 uses of 320

NHibernate Query Analyzer 216, 255

NHibernate.Burrow project 344

NHibernate.Cache.ICache-ConcurrencyStrategy 158

NHibernate.Cache.ICache-Provider 159

NHibernate.ConnectionReleaseMode 139

NHibernate.Context namespace 327

NHibernate.SQL 275

NHibernate.Tasks.  
 Hbm2Net-Task 292

NHibernate.Tool.hbm2ddl.SchemaExport 288

NHibernate.Tool.hbm2net.Console 292

NHibernateHelper class 321

NHibernateUtil.Initialize() 131

NHibernateUtil.IsInitialized() 131

noise 338

nonindexed collection 198

non-intrusive 12

nonstrict-read-write concurrency strategy 158

nosetter.\* strategy 69

not-null attribute 290

not-null column, avoiding 188

null operator 219

null value, testing for 219

nullable type 179–180

## O

### object

creating. *See* persistence detached 78  
 equality 76–77  
*See also* identity  
 graph 10  
 identity 76–81  
 making persistent 110–111

retrieving 121–133  
 by identifier 122–123  
*See also* domain model

object identity and caching 154

object referencing 16

object retrieval, optimizing p249–255

object role modeling 21

object.ReferenceEquals() 76–77

object/relational impedance mismatch 12

object/relational mapping (ORM) 4, 14  
 defined 21–23  
 problems in 125  
 reasons to use 21–23  
 triggers combined with 303  
 without domain model 54

object/relational persistence 76

ObjectDataSource 314

object-oriented programming (OOP) 7, 12

object-oriented viewpoint 12

ObjectViews 13  
 data binding with 315

Observer pattern 278  
 applying to an entity 307–309

on clause, specifying join condition 223

one-to-many 87–88

one-to-many association 198

one-to-one association 189–193, 232  
 implicit join 228

OnFlushDirty() 281

OnSave() 281–282

OOP. *See* object-oriented programming (OOP)

OpenSession() 28

operations, grouping 60

operator  
 comparison, and restriction 218  
 logical 221

optimistic locking 142, 147  
 alternative implementations 151  
 vs. pessimistic 149

optimistic offline locking 147–149

optimistic-lock attribute 151

Oracle. *See* relational database management system

order by clause 221

order-by 185

ordered collection 184

ordered pair 226

ORM. *See* object/relational mapping (ORM)

outer join 223

outer-join attribute 251

outer-join loading 126

## P

pagination 209  
 in SQL 210

paradigm mismatch 5, 12, 15–16

parallel class hierarchies smell 346

parameters 212  
 binding 211–215  
 importance of 211  
 positional 212

parent/child relationship 90–91

pattern 261–262  
 Data Access Object 324–326  
 Data Transfer Object 345  
 Gateway 325  
 Repository 325

Pattern Language of Programs (PLoP) conference 262

performance 22, 273  
 advice on eager loading 334  
 improving 275  
 with report queries 236

performance tuning 249

persistence 5–9  
 API 110–114  
 approaches in .NET 9–15  
 automated 55–56  
 automatic dirty checking 103, 113  
 by reachability 115–116  
 cascading 116–117  
 choices 4  
 context 327  
 conversation 17  
 deleting 103, 113–114  
 detaching domain model 104  
 dirty checking 62  
 hand-coding 10, 14  
 ignorance of 101  
 lifecycle 36, 101–110  
 management 110–114  
 mechanism 5  
 optimization 132–133  
 querying  
 using HQL 123–124

persistence (*continued*)  
   retrieving 112  
   techniques 121–133  
   saving 102, 110–111  
   state 101  
   detached 103–105  
   persistent 102–103  
   transient 102  
   transitive 114–121  
   transparency issue 109  
   transparent 17, 55–56  
   tuning 132–133  
   updating 111–112  
     transparently 113  
   using NHibernate 10  
 persistence ignorance 305–309  
 persistence layer 8, 15, 268–269  
   choosing 9  
   designing 320–335  
   generic 326–335  
   implementing 268, 321–326  
   testing 269  
 persistence logic, testing 269  
 persistence manager  
   56, 110–114  
 persistence-abstracted  
   entity 307  
 persistence-related code  
   260, 320  
   abstracting 305–307  
 persistent instance, caches in  
   sessions 340  
 pessimistic lock 339  
 pessimistic locking 143–146  
   not available 325  
   vs. optimistic 149  
 phantom read 141  
 placeholder 245  
 Plain Old CLR Object  
   (POCO) 56, 344  
   DTO as 345  
   making serializable 345  
   persistent class  
     generation 293  
 POCO. *See* Plain Old CLR  
   Object (POCO)  
 polymorphic association 201  
   and table-per-concrete-  
   class 204–205  
 polymorphic collection 203  
 polymorphic query 217  
 polymorphism 7, 12  
   association 92  
   mapping 200–205  
   query 92

portability 56  
 positional parameter 212  
 PostFlush() 281  
 presentation layer 8, 269–270  
   DataSet-based 13  
   implementing 269  
   web pages as 271  
 presentation model 314  
 Prevalence cache provider 159  
 primary key 12, 16, 77, 230  
   choosing 79–81  
   natural 79  
 primary key association 192–193  
 primitive types 81  
 problem domain 53  
 process scope cache 153  
 process-scoped identity 103–104  
 productivity 22  
 profiler 255  
 projection 124, 232–234, 350  
 property  
   access 68–70  
   derived 68  
   mapping 66–68  
   reading and setting state 63  
 property accessor 68  
 PropertyToColumnName() 73  
 proxy 122, 127, 131  
   problems with 202  
 proxy-safe typecast 202

## Q

QBC. *See* Query by Criteria  
   (QBC) API  
 QBE. *See* Query by Example  
   (QBE)  
 <qualifyAssembly> 43  
 quantification 242, 351  
 query  
   advanced techniques 238–243  
   building with string  
   manipulations 238  
   by example 238–240  
   caching 253–255  
   comparisons between  
   keys 230  
   complex 15, 207  
   complex criteria 221  
   criteria, comparison  
   operators 219  
   dynamic 238–240  
   entity 244  
   Enumerable() 252  
   executing 208–215  
   externalizing strings to  
   mapping metadata 214  
   identifier value, comparisons  
   with 231  
   iterating results 210, 252  
   listing results 210  
   named 214–215  
   native SQL 243–249  
   NHibernate API 36  
   object reference comparisons  
   with 230  
   optimizing 215  
   parameter binding 211–214  
   polymorphic 217  
   polymorphism 16, 92  
   porting to mapping files 215  
   QBE and dynamic 239  
   report 231–237  
   restriction 217  
   results, ordering 221  
   retrieving multiple entity  
   types 245  
   root entities of criteria 209  
   scalar 244  
   simplest 215  
   SQL, named 246–248  
   calling stored  
   procedures 247  
   storing 214  
   substitutions 215  
   using aliases 216  
   using enumerated types  
   in 180  
   using LINQ 20  
   ways of expressing in  
   Hibernate 208  
 Query APIs  
   binding arbitrary arguments  
   with 213  
   creating a new instance  
   of 208–211  
   Enumerable() method  
   and 252  
   List() method and 252  
   method-chaining 210  
   purpose of 208  
   testing 216  
 Query by Criteria (QBC)  
   API 19, 124, 207  
 Query by Example (QBE)  
   124, 208, 239  
 <query> element 214  
 query engine,  
   implementing 18–20

**R**

RDBMS. *See* relational database, management system

read  
 committed 141  
 dirty 140  
 phantom 141  
 uncommitted 141  
 unrepeatable 141

readability 56

read-only concurrency  
 strategy 158

read-write concurrency  
 strategy 158

reassociation, selective, of  
 detached instances. 106

Reconnect() 139

redundancy, minimizing 329

reference data 155

reflection 65  
 optimization 70–71

Refresh() 304

relation, SQL 349

relational database 5, 7  
 management system 5  
 independence 22

relational model 349

relationship 91  
 "has a" 91  
 "is a" 91  
 between entities 12  
 managed 59  
*See also* association

relationship table 130

reliability 273

repeatable read 141

report query 124, 231–237  
 aggregation 234  
 grouping 234  
 improving performances  
 with 236  
 projection 232–234  
 restricting groups with  
 having 236  
 select clause 232

Repository pattern 325

requirePermission attribute 271

restriction 217, 350  
 ignoring deprecated data 218  
 with comparison  
 operators 218

result, distinct 233

<resultset> 246

retrieve command, custom 248

<return> 246

<return-join> 246

<return-scalar> 246

reusability 7

rich object model 54

roll back 135

Rollback() 136

root entity (criteria  
 queries) 209, 228

Ruby on Rails  
 migrations 295

Ruby on Rails migrations 295

---

**S**

Save() 110

SaveOrUpdate() 120

saving. *See* persistence

scaffolding code 58

scalability 273

scalar query 244

schema  
 creating with hbm2ddl  
 290–291  
 database 288  
 maintenance, automatic  
 294–296  
 mapping domain models to  
 given 294

schema definition 63

SchemaExport 292

SchemaUpdate 295

search  
 case-insensitive 220  
 string-based 220  
 wildcard 220

second-level cache 35

securability 273

security, database capabilities 7

select clause  
 calling aggregate functions  
 in 234  
 calling SQL functions  
 from 233  
 changing results with 230  
 elements of results 233  
 grouping, and 234  
 projection, and 232–234  
 rules governing 236  
 subqueries 242  
 using aliases in 226–227

select new 232, 236

select-before-update 304

selective reassociation of  
 detached instances 106

self-documenting code, achieved  
 through refactoring 326

separation of concerns  
 55, 261, 263

serializability 57

serializable 141

services 277  
 integrating 277–284  
 loose coupling 277

session 28  
 flushing 138–139  
 granularity 150–151  
 temporary 282–283  
*See also* persistence manager

Session API  
 kinds of state contained  
 in 340  
 long 340–344

session cache 156  
 managing 157

session factory 275

session management for  
 ASP.NET applications  
 332–335

session per request, when to  
 use 344

Session.Clear() 157

session.Connection  
 property 248

session.FlushMode 139

session.GetIdentifier(entity)  
 282

Session.Load() 202

session-context implementa-  
 tions built in to  
 NHibernate 327

SessionFactory 35

SessionFactory API  
 initialization 321  
 statelessness of 322  
 storage of 322

session-per-conversation 151

session-per-request 150, 333

session-per-request-with-  
 detached-objects 151

<set> 196  
 using for parent/child  
 relationships 199

set 59

set collection 181

SetDefaultAssembly() 75

SetDefaultNamespace() 75

SetEntity() 213

SetEnum() 213

SetFetchMode() 252

- SetInt32() 213
  - SetMaxResults() 209
  - SetParameter() 214
  - SetProperties() 214
  - SetString() 212
  - SetTimestamp() 213
  - SharpDevelop 352
  - shotgun-change smell 346
  - show\_sql 44
  - silver bullet, ORM 23
  - single point association 127–129
  - Singleton pattern 261
  - size() function 243
  - smell
    - code 346
    - parallel class hierarchies 346
    - shotgun-change 346
  - software development 3
  - some quantifier 242
  - sorted collection 184
  - sorting database capabilities 7
  - Sparx Systems Enterprise Architect 52
  - SQL (Structured Query Language) 6, 9
    - aggregate functions 234
    - arbitrary, executing during database generation 291
    - backtick 72
    - basic concepts 349
    - books 6
    - command 14
    - dynamic generation 103
    - expressing queries with 208
    - function 68
    - inner joins 222
    - keywords, writing 216
    - logging 44
    - named 246–248
      - calling stored procedures 247
    - outer joins 223
    - pagination 210
    - passthrough 243
    - query 18
    - query hints 207
    - quoted identifier 72
    - quoted identifiers 70
    - relation 349
    - schema 73–74
    - subselects 242–243
  - SQL databases. *See* relational database, management system
  - SQL function, calling 233
  - SQL injection attack 211
  - SQL Server 10, 42
    - See also* relational database, management system
  - SQL statement execution, timing of 111
  - SqlCommand 6
  - sql-type attribute 290
  - stale data 337
  - StaleObjectStateException 145, 149
  - state, entity 54
  - state-oriented NHibernate vs ADO.NET 335
  - stored procedure 7, 247
  - string
    - concatenation 220
    - matching 220
  - string manipulation, building queries with 238
  - structured data 7
  - subclass 94
  - subquery 242–243
    - correlated 242
    - uncorrelated 242
  - subselect 68, 242–243, 351
  - sum() function 234
  - surrogate key 16, 79, 109
  - synthetic identifier 79
    - mapping 297
  - SysCache cache provider 159
  - system transaction 135–146
  - System.Collections.IDictionary 41
  - System.Collections.SortedList 185
  - System.Collections.Specialized.ListDictionary 186
  - System.Data.IsolationLevel 143
  - System.Diagnostics API 284
  - System.EnterpriseServices assembly 346
  - System.String 185
- T**
- 
- table
    - audit logs 277
    - mapping with composite keys 298–302
    - mapping with natural keys 297–298
    - relationship 130
  - table adapter 14
  - table per class hierarchy 92–95
  - table per concrete class 92–93
    - and polymorphic associations 204–205
  - table per subclass 92, 95–98
  - TableName() 73
  - tables, mapping with composite keys 303
  - ternary association 197
  - ternary logic 218, 264
  - test
    - data integrity 264
    - logic 264–265
  - testability 56
  - test-driven development (TDD) 261–262
  - theta-style join 223, 229
  - tier, externalizing data between 345
  - timestamp cache 254
  - toolset, NHibernate hbm2ddl. *See* hbm2ddl (SchemaExport)
  - top-down development 288–292
  - ToString(object entity) method 317
  - Transaction 111
  - transaction 15
    - boundaries
      - declaring with ITransaction methods 137
      - marking 136
    - coarse-grained 145
    - committed 135
    - conversation 17
    - demarcation 136
    - entity 102
    - fine-grained 145
    - isolation 140
      - caching, and 154
      - levels 141
    - rolled back 135
    - system states 136
    - transparent write-behind 103, 113
  - transaction scope cache 153
  - transaction.WasCommitted 137
  - transactional write-behind 33
  - transaction-scoped identity 104
  - transience 113–114
  - transient object 6
  - transitive persistence 114–121
  - transparent persistence 55–56, 344

transparent transactional  
   write-behind 103, 113  
 transparent write behind 138  
 trigger, database 278, 303–305  
 tuple 349  
 type  
   custom 62  
     mapping legacy columns  
       with 302–303  
   mapping 66  
   NHibernate 37  
   primitive 81  
   value 81–82  
   *See also* also domain model  
 type discriminator 93, 204  
 type system 167–181  
 typesafe enumeration 179–181

---

## U

UML class diagram 52  
 UMLet 52  
 uncorrelated subquery 242  
 unidirectional many-to-one  
   87–88  
 Unique attribute 290  
 unique constraint 290  
 unique-key attribute 290  
 uniqueResult() 218  
 unit of work 16–18

Unit of Work pattern 17  
 unit test 55  
 unit testing 262  
 unrepeatable read 141  
 unsaved-value 297  
   attribute 120  
 update  
   command, custom 248  
   control 71  
   dynamic 71  
   lost 140  
 Update() 111  
 update() 339  
 upper() function 220  
 user interface. *See* presentation  
   layer  
 user transaction 146  
 user-defined column type 15  
 UserType API 173–176, 302–303  
 utility class 321

---

## V

validation for proxy 127  
 value type 81–82  
   collections of, mapping  
     181–189  
   storing 181–186  
 VelocityRenderer 293  
 version check, manual 338

versioning  
   unsaved-value 120  
   usage with equality 108  
 Visio 21, 52  
 Visual Studio 6, 14  
   creating an NHibernate  
   project 25–26

---

## W

web application 13, 271  
   security policy, issues with 271  
 web page as presentation  
   layer 271  
 web tier, separation from  
   business-logic tiers 345  
 where clause  
   achieving restriction 217  
   arithmetic expression  
     support by 219  
   calling SQL functions in 220  
   evaluating expressions  
     with 218  
   restricting rows with 236  
   specifying join condition 223  
   subqueries in 242  
 Windows application 13  
   implementing  
     conversations 342  
 wrapping class 314  
 write-behind 47