

# Contents

<b>Preface</b> . . . . .	<b>iii</b>
The database server and the connectivity layer . . . . .	iv
The application data access layer . . . . .	iv
The ORM framework . . . . .	iv
The native query builder framework . . . . .	v
 <b>I JDBC and Database Essentials</b> . . . . .	 <b>1</b>
<b>1. Performance and Scaling</b> . . . . .	<b>3</b>
1.1 Response time and throughput . . . . .	3
1.2 Database connections boundaries . . . . .	5
1.3 Scaling up and scaling out . . . . .	6
1.3.1 Master-Slave replication . . . . .	7
1.3.2 Multi-Master replication . . . . .	8
1.3.3 Sharding . . . . .	9
 <b>2. JDBC Connection Management</b> . . . . .	 <b>13</b>
2.1 DriverManager . . . . .	14
2.2 DataSource . . . . .	16
2.2.1 Why is pooling so much faster? . . . . .	19
2.3 Queuing theory capacity planning . . . . .	21
2.4 Practical database connection provisioning . . . . .	24
2.4.1 A real-life connection pool monitoring example . . . . .	25
2.4.1.1 Concurrent connection request count metric . . . . .	26
2.4.1.2 Concurrent connection count metric . . . . .	27
2.4.1.3 Maximum pool size metric . . . . .	28
2.4.1.4 Connection acquisition time metric . . . . .	28
2.4.1.5 Retry attempts metric . . . . .	29
2.4.1.6 Overall connection acquisition time metric . . . . .	29
2.4.1.7 Connection lease time metric . . . . .	30
 <b>3. Batch Updates</b> . . . . .	 <b>31</b>
3.1 Batching Statements . . . . .	31
3.2 Batching PreparedStatements . . . . .	34
3.2.1 Choosing the right batch size . . . . .	36
3.2.2 Bulk processing . . . . .	37

3.3	Retrieving auto-generated keys . . . . .	38
3.3.1	Sequences to the rescue . . . . .	41
<b>4.</b>	<b>Statement Caching . . . . .</b>	<b>45</b>
4.1	Statement lifecycle . . . . .	45
4.1.1	Parser . . . . .	46
4.1.2	Optimizer . . . . .	46
4.1.2.1	Execution plan visualization . . . . .	47
4.1.3	Executor . . . . .	50
4.2	Caching performance gain . . . . .	50
4.3	Server-side statement caching . . . . .	51
4.3.1	Bind-sensitive execution plans . . . . .	53
4.4	Client-side statement caching . . . . .	57
<b>5.</b>	<b>ResultSet Fetching . . . . .</b>	<b>61</b>
5.1	ResultSet scrollability . . . . .	62
5.2	ResultSet changeability . . . . .	64
5.3	ResultSet holdability . . . . .	65
5.4	Fetching size . . . . .	65
5.5	ResultSet size . . . . .	68
5.5.1	Too many rows . . . . .	68
5.5.1.1	SQL limit clause . . . . .	69
5.5.1.2	JDBC max rows . . . . .	70
5.5.1.3	Less is more . . . . .	72
5.5.2	Too many columns . . . . .	73
<b>6.</b>	<b>Transactions . . . . .</b>	<b>75</b>
6.1	Atomicity . . . . .	76
6.2	Consistency . . . . .	78
6.3	Isolation . . . . .	80
6.3.1	Concurrency control . . . . .	80
6.3.1.1	Two-phase locking . . . . .	80
6.3.1.2	Multi-Version Concurrency Control . . . . .	84
6.3.2	Phenomena . . . . .	87
6.3.2.1	Dirty write . . . . .	88
6.3.2.2	Dirty read . . . . .	89
6.3.2.3	Non-repeatable read . . . . .	90
6.3.2.4	Phantom read . . . . .	91
6.3.2.5	Read skew . . . . .	92
6.3.2.6	Write skew . . . . .	93
6.3.2.7	Lost update . . . . .	94
6.3.3	Isolation levels . . . . .	95
6.3.3.1	Read Uncommitted . . . . .	96
6.3.3.2	Read Committed . . . . .	97
6.3.3.3	Repeatable Read . . . . .	99
6.3.3.4	Serializable . . . . .	100

6.4	Durability . . . . .	102
6.5	Read-only transactions . . . . .	104
6.5.1	Read-only transaction routing . . . . .	106
6.6	Transaction boundaries . . . . .	107
6.6.1	Distributed transactions . . . . .	111
6.6.1.1	Two-phase commit . . . . .	111
6.6.2	Declarative transactions . . . . .	112
6.7	Application-level transactions . . . . .	115
6.7.1	Pessimistic and optimistic locking . . . . .	117
6.7.1.1	Pessimistic locking . . . . .	117
6.7.1.2	Optimistic locking . . . . .	117

## **II JPA and Hibernate . . . . . 121**

<b>7.</b>	<b>Why JPA and Hibernate matter . . . . .</b>	<b>123</b>
7.1	The impedance mismatch . . . . .	124
7.2	JPA vs. Hibernate . . . . .	125
7.3	Schema ownership . . . . .	127
7.4	Entity state transitions . . . . .	129
7.5	Write-based optimizations . . . . .	131
7.6	Read-based optimizations . . . . .	134
7.7	Wrap-up . . . . .	137
<b>8.</b>	<b>Connection Management and Monitoring . . . . .</b>	<b>139</b>
8.1	JPA connection management . . . . .	139
8.2	Hibernate connection providers . . . . .	140
8.2.1	DriverManagerConnectionProvider . . . . .	141
8.2.2	C3P0ConnectionProvider . . . . .	141
8.2.3	HikariCPConnectionProvider . . . . .	142
8.2.4	DatasourceConnectionProvider . . . . .	143
8.2.5	Connection release modes . . . . .	143
8.3	Monitoring connections . . . . .	145
8.3.1	Hibernate statistics . . . . .	147
8.3.1.1	Customizing statistics . . . . .	149
8.4	Statement logging . . . . .	152
8.4.1	Statement formatting . . . . .	153
8.4.2	Statement-level comments . . . . .	154
8.4.3	Logging parameters . . . . .	155
8.4.3.1	DataSource-proxy . . . . .	155
8.4.3.2	P6Spy . . . . .	156
<b>9.</b>	<b>Mapping Types and Identifiers . . . . .</b>	<b>159</b>
9.1	Types . . . . .	161
9.1.1	Primitive types . . . . .	161
9.1.2	String types . . . . .	161

9.1.3	Date and Time types . . . . .	162
9.1.4	Numeric types . . . . .	163
9.1.5	Binary types . . . . .	163
9.1.6	UUID types . . . . .	163
9.1.7	Other types . . . . .	164
9.1.8	Custom types . . . . .	164
9.2	Identifiers . . . . .	170
9.2.1	UUID identifiers . . . . .	171
9.2.1.1	The assigned generator . . . . .	173
9.2.2	The legacy UUID generator . . . . .	174
9.2.2.1	The newer UUID generator . . . . .	174
9.2.3	Numerical identifiers . . . . .	175
9.2.3.1	Identity generator . . . . .	175
9.2.3.2	Sequence generator . . . . .	177
9.2.3.3	Table generator . . . . .	178
9.2.3.4	Optimizers . . . . .	180
9.2.3.4.1	The hi/lo algorithm . . . . .	181
9.2.3.4.2	The default sequence identifier generator . . . . .	183
9.2.3.4.3	The default table identifier generator . . . . .	184
9.2.3.4.4	The pooled optimizer . . . . .	185
9.2.3.4.5	The pooled-lo optimizer . . . . .	187
9.2.3.5	Optimizer gain . . . . .	188
9.2.3.5.1	Sequence generator performance gain . . . . .	188
9.2.3.5.2	Table generator performance gain . . . . .	189
9.2.3.6	Identifier generator performance . . . . .	189
10.	<b>Relationships . . . . .</b>	<b>193</b>
10.1	Relationship types . . . . .	194
10.2	@ManyToOne . . . . .	196
10.3	@OneToMany . . . . .	197
10.3.1	Bidirectional @OneToMany . . . . .	198
10.3.2	Unidirectional @OneToMany . . . . .	203
10.3.3	Ordered unidirectional @OneToMany . . . . .	205
10.3.3.1	@ElementCollection . . . . .	207
10.3.4	@OneToMany with @JoinColumn . . . . .	209
10.3.5	Unidirectional @OneToMany Set . . . . .	211
10.4	@OneToOne . . . . .	214
10.4.1	Unidirectional @OneToOne . . . . .	214
10.4.2	Bidirectional @OneToOne . . . . .	217
10.5	@ManyToMany . . . . .	219
10.5.1	Unidirectional @ManyToMany . . . . .	219
10.5.2	Bidirectional @ManyToMany . . . . .	221
10.5.3	The @OneToMany alternative . . . . .	223
11.	<b>Inheritance . . . . .</b>	<b>229</b>
11.1	Single table . . . . .	233

11.1.1	Data integrity constraints . . . . .	237
11.2	Join table . . . . .	239
11.3	Table-per-class . . . . .	243
11.4	Mapped superclass . . . . .	247
<b>12.</b>	<b>Flushing . . . . .</b>	<b>253</b>
12.1	Flush modes . . . . .	254
12.2	Events and the action queue . . . . .	256
12.2.1	Flush operation order . . . . .	257
12.3	Dirty Checking . . . . .	259
12.3.1	The default dirty checking mechanism . . . . .	259
12.3.1.1	Controlling the Persistence Context size . . . . .	260
12.3.2	Bytecode enhancement . . . . .	262
<b>13.</b>	<b>Batching . . . . .</b>	<b>267</b>
13.1	Batching insert statements . . . . .	269
13.2	Batching update statements . . . . .	270
13.3	Batching delete statements . . . . .	273
<b>14.</b>	<b>Fetching . . . . .</b>	<b>279</b>
14.1	DTO projection . . . . .	280
14.1.1	DTO projection pagination . . . . .	281
14.1.2	Native query DTO projection . . . . .	283
14.2	Query fetch size . . . . .	292
14.3	Fetching entities . . . . .	293
14.3.1	Direct fetching . . . . .	293
14.3.1.1	Fetching a Proxy reference . . . . .	294
14.3.1.2	Natural identifier fetching . . . . .	295
14.3.2	Query fetching . . . . .	297
14.3.3	Fetching associations . . . . .	299
14.3.3.1	FetchType.EAGER . . . . .	300
14.3.3.2	FetchType.LAZY . . . . .	304
14.3.3.2.1	The N+1 query problem . . . . .	306
14.3.3.2.2	How to catch N+1 query problems during testing . . . . .	308
14.3.3.2.3	LazyInitializationException . . . . .	311
14.3.3.2.4	The Open Session in View Anti-Pattern . . . . .	312
14.3.3.2.5	Temporary Session Lazy Loading Anti-Pattern . . . . .	315
14.3.3.3	Associations and pagination . . . . .	316
14.3.4	Attribute lazy fetching . . . . .	318
14.3.5	Fetching subentities . . . . .	321
14.4	Entity reference deduplication . . . . .	323
14.5	Query plan cache . . . . .	326
<b>15.</b>	<b>Caching . . . . .</b>	<b>329</b>
15.1	Caching flavors . . . . .	329
15.2	Cache synchronization strategies . . . . .	331

15.2.1	Cache-aside . . . . .	331
15.2.2	Read-through . . . . .	332
15.2.3	Write-invalidate . . . . .	332
15.2.4	Write-through . . . . .	333
15.2.5	Write-behind . . . . .	333
15.3	Database caching . . . . .	334
15.4	Application-level caching . . . . .	337
15.4.1	Entity aggregates . . . . .	337
15.4.2	Distributed key-value stores . . . . .	338
15.4.3	Cache synchronization patterns . . . . .	339
15.4.4	Synchronous updates . . . . .	339
15.4.5	Asynchronous updates . . . . .	339
15.4.5.1	Change data capture . . . . .	340
15.5	Second-level caching . . . . .	342
15.5.1	Enabling the second-level cache . . . . .	343
15.5.2	Entity cache loading flow . . . . .	344
15.5.3	Entity cache entry . . . . .	344
15.5.3.1	Entity reference cache store . . . . .	348
15.5.4	Collection cache entry . . . . .	350
15.5.5	Query cache entry . . . . .	351
15.5.6	Cache concurrency strategies . . . . .	353
15.5.6.1	READ_ONLY . . . . .	354
15.5.6.1.1	Inserting READ_ONLY cache entries . . . . .	354
15.5.6.1.2	Updating READ_ONLY cache entries . . . . .	357
15.5.6.1.3	Deleting READ_ONLY cache entries . . . . .	358
15.5.6.2	NONSTRICT_READ_WRITE . . . . .	360
15.5.6.2.1	Inserting NONSTRICT_READ_WRITE cache entries . . . . .	360
15.5.6.2.2	Updating NONSTRICT_READ_WRITE cache entries . . . . .	362
15.5.6.2.3	Risk of inconsistencies . . . . .	362
15.5.6.2.4	Deleting NONSTRICT_READ_WRITE cache entries . . . . .	364
15.5.6.3	READ_WRITE . . . . .	365
15.5.6.3.1	Inserting READ_WRITE cache entries . . . . .	365
15.5.6.3.2	Updating READ_WRITE cache entries . . . . .	367
15.5.6.3.3	Deleting READ_WRITE cache entries . . . . .	370
15.5.6.3.4	Soft locking concurrency control . . . . .	371
15.5.6.4	TRANSACTIONAL . . . . .	373
15.5.6.4.1	XA_Strict mode . . . . .	373
15.5.6.4.2	XA mode . . . . .	374
15.5.6.4.3	Inserting TRANSACTIONAL cache entries . . . . .	375
15.5.6.4.4	Updating TRANSACTIONAL cache entries . . . . .	376
15.5.6.4.5	Deleting TRANSACTIONAL cache entries . . . . .	378
15.5.7	Query cache strategy . . . . .	379
15.5.7.1	Table space query cache invalidation . . . . .	381
15.5.7.2	Native SQL statement query cache invalidation . . . . .	383

<b>16. Concurrency Control</b>	<b>387</b>
16.1 Hibernate optimistic locking	387
16.1.1 The implicit optimistic locking mechanism	387
16.1.1.1 Resolving optimistic locking conflicts	390
16.1.1.2 Splitting entities	392
16.1.1.3 Versionless optimistic locking	394
16.1.1.3.1 OptimisticLockType.DIRTY update caveat	397
16.2 The explicit locking mechanism	400
16.2.1 PESSIMISTIC_READ and PESSIMISTIC_WRITE	401
16.2.1.1 Lock scope	404
16.2.1.2 Lock timeout	409
16.2.2 LockModeType.OPTIMISTIC	417
16.2.2.1 Inconsistency risk	419
16.2.3 LockModeType.OPTIMISTIC_FORCE_INCREMENT	420
16.2.4 LockModeType.PESSIMISTIC_FORCE_INCREMENT	424
 <b>III JOOQ</b>	 <b>431</b>
<b>17. Why jOOQ matters</b>	<b>433</b>
17.1 How jOOQ works	433
17.2 DML statements	433
17.3 Java-based schema	435
17.4 Upsert	437
17.4.1 Oracle	438
17.4.2 SQL Server	439
17.4.3 PostgreSQL	439
17.4.4 MySQL	440
17.5 Batch updates	440
17.6 Inlining bind parameters	441
17.7 Complex queries	442
17.8 Stored procedures and functions	445
17.9 Streaming	447
17.10 Keyset pagination	451
 <b>Index</b>	 <b>455</b>