ResearchGate

See discussions, stats, and author profiles for this publication at: https://www.researchgate.net/publication/200034291

Database Systems: The Complete Book

Chapter · January 2002

CITATIONS
1,003

READS
79,471

6 authors, including:

Hector Hector
Fundación Universitaria Tecnológico Comfenalco
23 PUBLICATIONS **2,081** CITATIONS

SEE PROFILE

All content following this page was uploaded by Hector Hector on 16 September 2015.

The user has requested enhancement of the downloaded file.

Database Systems: The Complete Book

Hector Garcia-Molina
Jeffrey D. Ullman
Jennifer Widom

Department of Computer Science
Stanford University

An Alan R. Apt Book

Prentice Hall
Upper Saddle River, New Jersey 07458

About the Authors

JEFFREY D. ULLMAN is the Stanford W. Ascherman Professor of Computer Science at Stanford University. He is the author or co-author of 16 books, including *Elements of ML Programming* (Prentice Hall 1998). His research interests include data mining, information integration, and electronic education. He is a member of the National Academy of Engineering, and recipient of a Guggenheim Fellowship, the Karl V. Karlstrom Outstanding Educator Award, the SIGMOD Contributions Award, and the Knuth Prize.

JENNIFER WIDOM is Associate Professor of Computer Science and Electrical Engineering at Stanford University. Her research interests include query processing on data streams, data caching and replication, semistructured data and XML, and data warehousing. She is a former Guggenheim Fellow and has served on numerous program committees, advisory boards, and editorial boards.

HECTOR GARCIA-MOLINA is the L. Bosack and S. Lerner Professor of Computer Science and Electrical Engineering, and Chair of the Department of Computer Science at Stanford University. His research interests include digital libraries, information integration, and database application on the Internet. He was a recipient of the SIGMOD Innovations Award and is a member of PITAC (President's Information-Technology Advisory Council).

Table of Contents

1 The Worlds of Database Systems 1
- 1.1 The Evolution of Database Systems . . . 2
  - 1.1.1 Early Database Management Systems . . . 2
  - 1.1.2 Relational Database Systems . . . 4
  - 1.1.3 Smaller and Smaller Systems . . . 5
  - 1.1.4 Bigger and Bigger Systems . . . 6
  - 1.1.5 Client-Server and Multi-Tier Architectures . . . 7
  - 1.1.6 Multimedia Data . . . 8
  - 1.1.7 Information Integration . . . 8
- 1.2 Overview of a Database Management System . . . 9
  - 1.2.1 Data-Definition Language Commands . . . 10
  - 1.2.2 Overview of Query Processing . . . 10
  - 1.2.3 Storage and Buffer Management . . . 12
  - 1.2.4 Transaction Processing . . . 13
  - 1.2.5 The Query Processor . . . 14
- 1.3 Outline of Database-System Studies . . . 15
  - 1.3.1 Database Design . . . 16
  - 1.3.2 Database Programming . . . 17
  - 1.3.3 Database System Implementation . . . 17
  - 1.3.4 Information Integration Overview . . . 19
- 1.4 Summary of Chapter 1 . . . 19
- 1.5 References for Chapter 1 . . . 20

2 The Entity-Relationship Data Model 23
- 2.1 Elements of the E/R Model . . . 24
  - 2.1.1 Entity Sets . . . 24
  - 2.1.2 Attributes . . . 25
  - 2.1.3 Relationships . . . 25
  - 2.1.4 Entity-Relationship Diagrams . . . 25
  - 2.1.5 Instances of an E/R Diagram . . . 27
  - 2.1.6 Multiplicity of Binary E/R Relationships . . . 27
  - 2.1.7 Multiway Relationships . . . 28
  - 2.1.8 Roles in Relationships . . . 29

2.1.9 Attributes on Relationships . . . 31
2.1.10 Converting Multiway Relationships to Binary . . . 32
2.1.11 Subclasses in the E/R Model . . . 33
2.1.12 Exercises for Section 2.1 . . . 36
2.2 Design Principles . . . 39
2.2.1 Faithfulness . . . 39
2.2.2 Avoiding Redundancy . . . 39
2.2.3 Simplicity Counts . . . 40
2.2.4 Choosing the Right Relationships . . . 40
2.2.5 Picking the Right Kind of Element . . . 42
2.2.6 Exercises for Section 2.2 . . . 44
2.3 The Modeling of Constraints . . . 47
2.3.1 Classification of Constraints . . . 47
2.3.2 Keys in the E/R Model . . . 48
2.3.3 Representing Keys in the E/R Model . . . 50
2.3.4 Single-Value Constraints . . . 51
2.3.5 Referential Integrity . . . 51
2.3.6 Referential Integrity in E/R Diagrams . . . 52
2.3.7 Other Kinds of Constraints . . . 53
2.3.8 Exercises for Section 2.3 . . . 53
2.4 Weak Entity Sets . . . 54
2.4.1 Causes of Weak Entity Sets . . . 54
2.4.2 Requirements for Weak Entity Sets . . . 56
2.4.3 Weak Entity Set Notation . . . 57
2.4.4 Exercises for Section 2.4 . . . 58
2.5 Summary of Chapter 2 . . . 59
2.6 References for Chapter 2 . . . 60

3 The Relational Data Model **61**
3.1 Basics of the Relational Model . . . 61
3.1.1 Attributes . . . 62
3.1.2 Schemas . . . 62
3.1.3 Tuples . . . 62
3.1.4 Domains . . . 63
3.1.5 Equivalent Representations of a Relation . . . 63
3.1.6 Relation Instances . . . 64
3.1.7 Exercises for Section 3.1 . . . 64
3.2 From E/R Diagrams to Relational Designs . . . 65
3.2.1 From Entity Sets to Relations . . . 66
3.2.2 From E/R Relationships to Relations . . . 67
3.2.3 Combining Relations . . . 70
3.2.4 Handling Weak Entity Sets . . . 71
3.2.5 Exercises for Section 3.2 . . . 75
3.3 Converting Subclass Structures to Relations . . . 76
3.3.1 E/R-Style Conversion . . . 77

3.3.2 An Object-Oriented Approach . . . 78
3.3.3 Using Null Values to Combine Relations . . . 79
3.3.4 Comparison of Approaches . . . 79
3.3.5 Exercises for Section 3.3 . . . 80
3.4 Functional Dependencies . . . 82
3.4.1 Definition of Functional Dependency . . . 83
3.4.2 Keys of Relations . . . 84
3.4.3 Superkeys . . . 86
3.4.4 Discovering Keys for Relations . . . 87
3.4.5 Exercises for Section 3.4 . . . 88
3.5 Rules About Functional Dependencies . . . 90
3.5.1 The Splitting/Combining Rule . . . 90
3.5.2 Trivial Functional Dependencies . . . 92
3.5.3 Computing the Closure of Attributes . . . 92
3.5.4 Why the Closure Algorithm Works . . . 95
3.5.5 The Transitive Rule . . . 96
3.5.6 Closing Sets of Functional Dependencies . . . 98
3.5.7 Projecting Functional Dependencies . . . 98
3.5.8 Exercises for Section 3.5 . . . 100
3.6 Design of Relational Database Schemas . . . 102
3.6.1 Anomalies . . . 103
3.6.2 Decomposing Relations . . . 103
3.6.3 Boyce-Codd Normal Form . . . 105
3.6.4 Decomposition into BCNF . . . 107
3.6.5 Recovering Information from a Decomposition . . . 112
3.6.6 Third Normal Form . . . 114
3.6.7 Exercises for Section 3.6 . . . 117
3.7 Multivalued Dependencies . . . 118
3.7.1 Attribute Independence and Its Consequent Redundancy 118
3.7.2 Definition of Multivalued Dependencies . . . 119
3.7.3 Reasoning About Multivalued Dependencies . . . 120
3.7.4 Fourth Normal Form . . . 122
3.7.5 Decomposition into Fourth Normal Form . . . 123
3.7.6 Relationships Among Normal Forms . . . 124
3.7.7 Exercises for Section 3.7 . . . 126
3.8 Summary of Chapter 3 . . . 127
3.9 References for Chapter 3 . . . 129

4 Other Data Models **131**
4.1 Review of Object-Oriented Concepts . . . 132
4.1.1 The Type System . . . 132
4.1.2 Classes and Objects . . . 133
4.1.3 Object Identity . . . 133
4.1.4 Methods . . . 133
4.1.5 Class Hierarchies . . . 134

4.2 Introduction to ODL . . . 135
4.2.1 Object-Oriented Design . . . 135
4.2.2 Class Declarations . . . 136
4.2.3 Attributes in ODL . . . 136
4.2.4 Relationships in ODL . . . 138
4.2.5 Inverse Relationships . . . 139
4.2.6 Multiplicity of Relationships . . . 140
4.2.7 Methods in ODL . . . 141
4.2.8 Types in ODL . . . 144
4.2.9 Exercises for Section 4.2 . . . 146
4.3 Additional ODL Concepts . . . 147
4.3.1 Multiway Relationships in ODL . . . 148
4.3.2 Subclasses in ODL . . . 149
4.3.3 Multiple Inheritance in ODL . . . 150
4.3.4 Extents . . . 151
4.3.5 Declaring Keys in ODL . . . 152
4.3.6 Exercises for Section 4.3 . . . 155
4.4 From ODL Designs to Relational Designs . . . 155
4.4.1 From ODL Attributes to Relational Attributes . . . 156
4.4.2 Nonatomic Attributes in Classes . . . 157
4.4.3 Representing Set-Valued Attributes . . . 158
4.4.4 Representing Other Type Constructors . . . 160
4.4.5 Representing ODL Relationships . . . 162
4.4.6 What If There Is No Key? . . . 164
4.4.7 Exercises for Section 4.4 . . . 164
4.5 The Object-Relational Model . . . 166
4.5.1 From Relations to Object-Relations . . . 166
4.5.2 Nested Relations . . . 167
4.5.3 References . . . 169
4.5.4 Object-Oriented Versus Object-Relational . . . 170
4.5.5 From ODL Designs to Object-Relational Designs . . . 172
4.5.6 Exercises for Section 4.5 . . . 172
4.6 Semistructured Data . . . 173
4.6.1 Motivation for the Semistructured-Data Model . . . 173
4.6.2 Semistructured Data Representation . . . 174
4.6.3 Information Integration Via Semistructured Data . . . 175
4.6.4 Exercises for Section 4.6 . . . 177
4.7 XML and Its Data Model . . . 178
4.7.1 Semantic Tags . . . 178
4.7.2 Well-Formed XML . . . 179
4.7.3 Document Type Definitions . . . 180
4.7.4 Using a DTD . . . 182
4.7.5 Attribute Lists . . . 183
4.7.6 Exercises for Section 4.7 . . . 185
4.8 Summary of Chapter 4 . . . 186

4.9 References for Chapter 4 . . . 187

5 Relational Algebra 189
5.1 An Example Database Schema . . . 190
5.2 An Algebra of Relational Operations . . . 191
5.2.1 Basics of Relational Algebra . . . 192
5.2.2 Set Operations on Relations . . . 193
5.2.3 Projection . . . 195
5.2.4 Selection . . . 196
5.2.5 Cartesian Product . . . 197
5.2.6 Natural Joins . . . 198
5.2.7 Theta-Joins . . . 199
5.2.8 Combining Operations to Form Queries . . . 201
5.2.9 Renaming . . . 203
5.2.10 Dependent and Independent Operations . . . 205
5.2.11 A Linear Notation for Algebraic Expressions . . . 206
5.2.12 Exercises for Section 5.2 . . . 207
5.3 Relational Operations on Bags . . . 214
5.3.1 Why Bags? . . . 214
5.3.2 Union, Intersection, and Difference of Bags . . . 215
5.3.3 Projection of Bags . . . 216
5.3.4 Selection on Bags . . . 217
5.3.5 Product of Bags . . . 218
5.3.6 Joins of Bags . . . 219
5.3.7 Exercises for Section 5.3 . . . 220
5.4 Extended Operators of Relational Algebra . . . 221
5.4.1 Duplicate Elimination . . . 222
5.4.2 Aggregation Operators . . . 222
5.4.3 Grouping . . . 223
5.4.4 The Grouping Operator . . . 224
5.4.5 Extending the Projection Operator . . . 226
5.4.6 The Sorting Operator . . . 227
5.4.7 Outerjoins . . . 228
5.4.8 Exercises for Section 5.4 . . . 230
5.5 Constraints on Relations . . . 231
5.5.1 Relational Algebra as a Constraint Language . . . 231
5.5.2 Referential Integrity Constraints . . . 232
5.5.3 Additional Constraint Examples . . . 233
5.5.4 Exercises for Section 5.5 . . . 235
5.6 Summary of Chapter 5 . . . 236
5.7 References for Chapter 5 . . . 237

6 The Database Language SQL 239
6.1 Simple Queries in SQL . . . 240
6.1.1 Projection in SQL . . . 242
6.1.2 Selection in SQL . . . 243
6.1.3 Comparison of Strings . . . 245
6.1.4 Dates and Times . . . 247
6.1.5 Null Values and Comparisons Involving NULL . . . 248
6.1.6 The Truth-Value UNKNOWN . . . 249
6.1.7 Ordering the Output . . . 251
6.1.8 Exercises for Section 6.1 . . . 252
6.2 Queries Involving More Than One Relation . . . 254
6.2.1 Products and Joins in SQL . . . 254
6.2.2 Disambiguating Attributes . . . 255
6.2.3 Tuple Variables . . . 256
6.2.4 Interpreting Multirelation Queries . . . 258
6.2.5 Union, Intersection, and Difference of Queries . . . 260
6.2.6 Exercises for Section 6.2 . . . 262
6.3 Subqueries . . . 264
6.3.1 Subqueries that Produce Scalar Values . . . 264
6.3.2 Conditions Involving Relations . . . 266
6.3.3 Conditions Involving Tuples . . . 266
6.3.4 Correlated Subqueries . . . 268
6.3.5 Subqueries in FROM Clauses . . . 270
6.3.6 SQL Join Expressions . . . 270
6.3.7 Natural Joins . . . 272
6.3.8 Outerjoins . . . 272
6.3.9 Exercises for Section 6.3 . . . 274
6.4 Full-Relation Operations . . . 277
6.4.1 Eliminating Duplicates . . . 277
6.4.2 Duplicates in Unions, Intersections, and Differences . . . 278
6.4.3 Grouping and Aggregation in SQL . . . 279
6.4.4 Aggregation Operators . . . 279
6.4.5 Grouping . . . 280
6.4.6 HAVING Clauses . . . 282
6.4.7 Exercises for Section 6.4 . . . 284
6.5 Database Modifications . . . 286
6.5.1 Insertion . . . 286
6.5.2 Deletion . . . 288
6.5.3 Updates . . . 289
6.5.4 Exercises for Section 6.5 . . . 290
6.6 Defining a Relation Schema in SQL . . . 292
6.6.1 Data Types . . . 292
6.6.2 Simple Table Declarations . . . 293
6.6.3 Modifying Relation Schemas . . . 294
6.6.4 Default Values . . . 295

6.6.5 Indexes . . . 295
6.6.6 Introduction to Selection of Indexes . . . 297
6.6.7 Exercises for Section 6.6 . . . 300
6.7 View Definitions . . . 301
6.7.1 Declaring Views . . . 302
6.7.2 Querying Views . . . 302
6.7.3 Renaming Attributes . . . 304
6.7.4 Modifying Views . . . 305
6.7.5 Interpreting Queries Involving Views . . . 308
6.7.6 Exercises for Section 6.7 . . . 310
6.8 Summary of Chapter 6 . . . 312
6.9 References for Chapter 6 . . . 313

7 Constraints and Triggers 315
7.1 Keys and Foreign Keys . . . 316
7.1.1 Declaring Primary Keys . . . 316
7.1.2 Keys Declared With UNIQUE . . . 317
7.1.3 Enforcing Key Constraints . . . 318
7.1.4 Declaring Foreign-Key Constraints . . . 319
7.1.5 Maintaining Referential Integrity . . . 321
7.1.6 Deferring the Checking of Constraints . . . 323
7.1.7 Exercises for Section 7.1 . . . 326
7.2 Constraints on Attributes and Tuples . . . 327
7.2.1 Not-Null Constraints . . . 328
7.2.2 Attribute-Based CHECK Constraints . . . 328
7.2.3 Tuple-Based CHECK Constraints . . . 330
7.2.4 Exercises for Section 7.2 . . . 331
7.3 Modification of Constraints . . . 333
7.3.1 Giving Names to Constraints . . . 334
7.3.2 Altering Constraints on Tables . . . 334
7.3.3 Exercises for Section 7.3 . . . 335
7.4 Schema-Level Constraints and Triggers . . . 336
7.4.1 Assertions . . . 337
7.4.2 Event-Condition-Action Rules . . . 340
7.4.3 Triggers in SQL . . . 340
7.4.4 Instead-Of Triggers . . . 344
7.4.5 Exercises for Section 7.4 . . . 345
7.5 Summary of Chapter 7 . . . 347
7.6 References for Chapter 7 . . . 348

8 System Aspects of SQL 349
8.1 SQL in a Programming Environment . . . 349
8.1.1 The Impedance Mismatch Problem . . . 350
8.1.2 The SQL/Host Language Interface . . . 352
8.1.3 The DECLARE Section . . . 352

8.1.4 Using Shared Variables . . . 353
8.1.5 Single-Row Select Statements . . . 354
8.1.6 Cursors . . . 355
8.1.7 Modifications by Cursor . . . 358
8.1.8 Protecting Against Concurrent Updates . . . 360
8.1.9 Scrolling Cursors . . . 361
8.1.10 Dynamic SQL . . . 361
8.1.11 Exercises for Section 8.1 . . . 363
8.2 Procedures Stored in the Schema . . . 365
8.2.1 Creating PSM Functions and Procedures . . . 365
8.2.2 Some Simple Statement Forms in PSM . . . 366
8.2.3 Branching Statements . . . 368
8.2.4 Queries in PSM . . . 369
8.2.5 Loops in PSM . . . 370
8.2.6 For-Loops . . . 372
8.2.7 Exceptions in PSM . . . 374
8.2.8 Using PSM Functions and Procedures . . . 376
8.2.9 Exercises for Section 8.2 . . . 377
8.3 The SQL Environment . . . 379
8.3.1 Environments . . . 379
8.3.2 Schemas . . . 380
8.3.3 Catalogs . . . 381
8.3.4 Clients and Servers in the SQL Environment . . . 382
8.3.5 Connections . . . 382
8.3.6 Sessions . . . 384
8.3.7 Modules . . . 384
8.4 Using a Call-Level Interface . . . 385
8.4.1 Introduction to SQL/CLI . . . 385
8.4.2 Processing Statements . . . 388
8.4.3 Fetching Data From a Query Result . . . 389
8.4.4 Passing Parameters to Queries . . . 392
8.4.5 Exercises for Section 8.4 . . . 393
8.5 Java Database Connectivity . . . 393
8.5.1 Introduction to JDBC . . . 393
8.5.2 Creating Statements in JDBC . . . 394
8.5.3 Cursor Operations in JDBC . . . 396
8.5.4 Parameter Passing . . . 396
8.5.5 Exercises for Section 8.5 . . . 397
8.6 Transactions in SQL . . . 397
8.6.1 Serializability . . . 397
8.6.2 Atomicity . . . 399
8.6.3 Transactions . . . 401
8.6.4 Read-Only Transactions . . . 403
8.6.5 Dirty Reads . . . 405
8.6.6 Other Isolation Levels . . . 407

8.6.7 Exercises for Section 8.6 . . . 409
8.7 Security and User Authorization in SQL . . . 410
8.7.1 Privileges . . . 410
8.7.2 Creating Privileges . . . 412
8.7.3 The Privilege-Checking Process . . . 413
8.7.4 Granting Privileges . . . 414
8.7.5 Grant Diagrams . . . 416
8.7.6 Revoking Privileges . . . 417
8.7.7 Exercises for Section 8.7 . . . 421
8.8 Summary of Chapter 8 . . . 422
8.9 References for Chapter 8 . . . 424

9 Object-Orientation in Query Languages 425
9.1 Introduction to OQL . . . 425
9.1.1 An Object-Oriented Movie Example . . . 426
9.1.2 Path Expressions . . . 426
9.1.3 Select-From-Where Expressions in OQL . . . 428
9.1.4 Modifying the Type of the Result . . . 429
9.1.5 Complex Output Types . . . 431
9.1.6 Subqueries . . . 431
9.1.7 Exercises for Section 9.1 . . . 433
9.2 Additional Forms of OQL Expressions . . . 436
9.2.1 Quantifier Expressions . . . 437
9.2.2 Aggregation Expressions . . . 437
9.2.3 Group-By Expressions . . . 438
9.2.4 HAVING Clauses . . . 441
9.2.5 Union, Intersection, and Difference . . . 442
9.2.6 Exercises for Section 9.2 . . . 442
9.3 Object Assignment and Creation in OQL . . . 443
9.3.1 Assigning Values to Host-Language Variables . . . 444
9.3.2 Extracting Elements of Collections . . . 444
9.3.3 Obtaining Each Member of a Collection . . . 445
9.3.4 Constants in OQL . . . 446
9.3.5 Creating New Objects . . . 447
9.3.6 Exercises for Section 9.3 . . . 448
9.4 User-Defined Types in SQL . . . 449
9.4.1 Defining Types in SQL . . . 449
9.4.2 Methods in User-Defined Types . . . 451
9.4.3 Declaring Relations with a UDT . . . 452
9.4.4 References . . . 452
9.4.5 Exercises for Section 9.4 . . . 454
9.5 Operations on Object-Relational Data . . . 455
9.5.1 Following References . . . 455
9.5.2 Accessing Attributes of Tuples with a UDT . . . 456
9.5.3 Generator and Mutator Functions . . . 457

9.5.4 Ordering Relationships on UDT's . . . 458
9.5.5 Exercises for Section 9.5 . . . 460
9.6 Summary of Chapter 9 . . . 461
9.7 References for Chapter 9 . . . 462

10 Logical Query Languages 463
10.1 A Logic for Relations . . . 463
10.1.1 Predicates and Atoms . . . 463
10.1.2 Arithmetic Atoms . . . 464
10.1.3 Datalog Rules and Queries . . . 465
10.1.4 Meaning of Datalog Rules . . . 466
10.1.5 Extensional and Intensional Predicates . . . 469
10.1.6 Datalog Rules Applied to Bags . . . 469
10.1.7 Exercises for Section 10.1 . . . 471
10.2 From Relational Algebra to Datalog . . . 471
10.2.1 Intersection . . . 471
10.2.2 Union . . . 472
10.2.3 Difference . . . 472
10.2.4 Projection . . . 473
10.2.5 Selection . . . 473
10.2.6 Product . . . 476
10.2.7 Joins . . . 476
10.2.8 Simulating Multiple Operations with Datalog . . . 477
10.2.9 Exercises for Section 10.2 . . . 479
10.3 Recursive Programming in Datalog . . . 480
10.3.1 Recursive Rules . . . 481
10.3.2 Evaluating Recursive Datalog Rules . . . 481
10.3.3 Negation in Recursive Rules . . . 486
10.3.4 Exercises for Section 10.3 . . . 490
10.4 Recursion in SQL . . . 492
10.4.1 Defining IDB Relations in SQL . . . 492
10.4.2 Stratified Negation . . . 494
10.4.3 Problematic Expressions in Recursive SQL . . . 496
10.4.4 Exercises for Section 10.4 . . . 499
10.5 Summary of Chapter 10 . . . 500
10.6 References for Chapter 10 . . . 501

11 Data Storage 503
11.1 The "Megatron 2002" Database System . . . 503
11.1.1 Megatron 2002 Implementation Details . . . 504
11.1.2 How Megatron 2002 Executes Queries . . . 505
11.1.3 What's Wrong With Megatron 2002? . . . 506
11.2 The Memory Hierarchy . . . 507
11.2.1 Cache . . . 507
11.2.2 Main Memory . . . 508

11.2.3 Virtual Memory . . . 509
11.2.4 Secondary Storage . . . 510
11.2.5 Tertiary Storage . . . 512
11.2.6 Volatile and Nonvolatile Storage . . . 513
11.2.7 Exercises for Section 11.2 . . . 514
11.3 Disks . . . 515
11.3.1 Mechanics of Disks . . . 515
11.3.2 The Disk Controller . . . 516
11.3.3 Disk Storage Characteristics . . . 517
11.3.4 Disk Access Characteristics . . . 519
11.3.5 Writing Blocks . . . 523
11.3.6 Modifying Blocks . . . 523
11.3.7 Exercises for Section 11.3 . . . 524
11.4 Using Secondary Storage Effectively . . . 525
11.4.1 The I/O Model of Computation . . . 525
11.4.2 Sorting Data in Secondary Storage . . . 526
11.4.3 Merge-Sort . . . 527
11.4.4 Two-Phase, Multiway Merge-Sort . . . 528
11.4.5 Multiway Merging of Larger Relations . . . 532
11.4.6 Exercises for Section 11.4 . . . 532
11.5 Accelerating Access to Secondary Storage . . . 533
11.5.1 Organizing Data by Cylinders . . . 534
11.5.2 Using Multiple Disks . . . 536
11.5.3 Mirroring Disks . . . 537
11.5.4 Disk Scheduling and the Elevator Algorithm . . . 538
11.5.5 Prefetching and Large-Scale Buffering . . . 541
11.5.6 Summary of Strategies and Tradeoffs . . . 543
11.5.7 Exercises for Section 11.5 . . . 544
11.6 Disk Failures . . . 546
11.6.1 Intermittent Failures . . . 547
11.6.2 Checksums . . . 547
11.6.3 Stable Storage . . . 548
11.6.4 Error-Handling Capabilities of Stable Storage . . . 549
11.6.5 Exercises for Section 11.6 . . . 550
11.7 Recovery from Disk Crashes . . . 550
11.7.1 The Failure Model for Disks . . . 551
11.7.2 Mirroring as a Redundancy Technique . . . 552
11.7.3 Parity Blocks . . . 552
11.7.4 An Improvement: RAID 5 . . . 556
11.7.5 Coping With Multiple Disk Crashes . . . 557
11.7.6 Exercises for Section 11.7 . . . 561
11.8 Summary of Chapter 11 . . . 563
11.9 References for Chapter 11 . . . 565

12 Representing Data Elements 567
12.1 Data Elements and Fields . . . 567
12.1.1 Representing Relational Database Elements . . . 568
12.1.2 Representing Objects . . . 569
12.1.3 Representing Data Elements . . . 569
12.2 Records . . . 572
12.2.1 Building Fixed-Length Records . . . 573
12.2.2 Record Headers . . . 575
12.2.3 Packing Fixed-Length Records into Blocks . . . 576
12.2.4 Exercises for Section 12.2 . . . 577
12.3 Representing Block and Record Addresses . . . 578
12.3.1 Client-Server Systems . . . 579
12.3.2 Logical and Structured Addresses . . . 580
12.3.3 Pointer Swizzling . . . 581
12.3.4 Returning Blocks to Disk . . . 586
12.3.5 Pinned Records and Blocks . . . 586
12.3.6 Exercises for Section 12.3 . . . 587
12.4 Variable-Length Data and Records . . . 589
12.4.1 Records With Variable-Length Fields . . . 590
12.4.2 Records With Repeating Fields . . . 591
12.4.3 Variable-Format Records . . . 593
12.4.4 Records That Do Not Fit in a Block . . . 594
12.4.5 BLOBS . . . 595
12.4.6 Exercises for Section 12.4 . . . 596
12.5 Record Modifications . . . 598
12.5.1 Insertion . . . 598
12.5.2 Deletion . . . 599
12.5.3 Update . . . 601
12.5.4 Exercises for Section 12.5 . . . 601
12.6 Summary of Chapter 12 . . . 602
12.7 References for Chapter 12 . . . 603

13 Index Structures 605
13.1 Indexes on Sequential Files . . . 606
13.1.1 Sequential Files . . . 606
13.1.2 Dense Indexes . . . 607
13.1.3 Sparse Indexes . . . 609
13.1.4 Multiple Levels of Index . . . 610
13.1.5 Indexes With Duplicate Search Keys . . . 612
13.1.6 Managing Indexes During Data Modifications . . . 615
13.1.7 Exercises for Section 13.1 . . . 620
13.2 Secondary Indexes . . . 622
13.2.1 Design of Secondary Indexes . . . 623
13.2.2 Applications of Secondary Indexes . . . 624
13.2.3 Indirection in Secondary Indexes . . . 625

13.2.4 Document Retrieval and Inverted Indexes . . . 626
13.2.5 Exercises for Section 13.2 . . . 630
13.3 B-Trees . . . 632
13.3.1 The Structure of B-trees . . . 633
13.3.2 Applications of B-trees . . . 636
13.3.3 Lookup in B-Trees . . . 638
13.3.4 Range Queries . . . 638
13.3.5 Insertion Into B-Trees . . . 639
13.3.6 Deletion From B-Trees . . . 642
13.3.7 Efficiency of B-Trees . . . 645
13.3.8 Exercises for Section 13.3 . . . 646
13.4 Hash Tables . . . 649
13.4.1 Secondary-Storage Hash Tables . . . 649
13.4.2 Insertion Into a Hash Table . . . 650
13.4.3 Hash-Table Deletion . . . 651
13.4.4 Efficiency of Hash Table Indexes . . . 652
13.4.5 Extensible Hash Tables . . . 652
13.4.6 Insertion Into Extensible Hash Tables . . . 653
13.4.7 Linear Hash Tables . . . 656
13.4.8 Insertion Into Linear Hash Tables . . . 657
13.4.9 Exercises for Section 13.4 . . . 660
13.5 Summary of Chapter 13 . . . 662
13.6 References for Chapter 13 . . . 663

14 Multidimensional and Bitmap Indexes 665
14.1 Applications Needing Multiple Dimensions . . . 666
14.1.1 Geographic Information Systems . . . 666
14.1.2 Data Cubes . . . 668
14.1.3 Multidimensional Queries in SQL . . . 668
14.1.4 Executing Range Queries Using Conventional Indexes . . . 670
14.1.5 Executing Nearest-Neighbor Queries Using Conventional Indexes . . . 671
14.1.6 Other Limitations of Conventional Indexes . . . 673
14.1.7 Overview of Multidimensional Index Structures . . . 673
14.1.8 Exercises for Section 14.1 . . . 674
14.2 Hash-Like Structures for Multidimensional Data . . . 675
14.2.1 Grid Files . . . 676
14.2.2 Lookup in a Grid File . . . 676
14.2.3 Insertion Into Grid Files . . . 677
14.2.4 Performance of Grid Files . . . 679
14.2.5 Partitioned Hash Functions . . . 682
14.2.6 Comparison of Grid Files and Partitioned Hashing . . . 683
14.2.7 Exercises for Section 14.2 . . . 684
14.3 Tree-Like Structures for Multidimensional Data . . . 687
14.3.1 Multiple-Key Indexes . . . 687

14.3.2 Performance of Multiple-Key Indexes . . . 688
14.3.3 *kd*-Trees . . . 690
14.3.4 Operations on *kd*-Trees . . . 691
14.3.5 Adapting *kd*-Trees to Secondary Storage . . . 693
14.3.6 Quad Trees . . . 695
14.3.7 R-Trees . . . 696
14.3.8 Operations on R-trees . . . 697
14.3.9 Exercises for Section 14.3 . . . 699
14.4 Bitmap Indexes . . . 702
14.4.1 Motivation for Bitmap Indexes . . . 702
14.4.2 Compressed Bitmaps . . . 704
14.4.3 Operating on Run-Length-Encoded Bit-Vectors . . . 706
14.4.4 Managing Bitmap Indexes . . . 707
14.4.5 Exercises for Section 14.4 . . . 709
14.5 Summary of Chapter 14 . . . 710
14.6 References for Chapter 14 . . . 711

15 Query Execution **713**
15.1 Introduction to Physical-Query-Plan Operators . . . 715
15.1.1 Scanning Tables . . . 716
15.1.2 Sorting While Scanning Tables . . . 716
15.1.3 The Model of Computation for Physical Operators . . . 717
15.1.4 Parameters for Measuring Costs . . . 717
15.1.5 I/O Cost for Scan Operators . . . 719
15.1.6 Iterators for Implementation of Physical Operators . . . 720
15.2 One-Pass Algorithms for Database Operations . . . 722
15.2.1 One-Pass Algorithms for Tuple-at-a-Time Operations . . . 724
15.2.2 One-Pass Algorithms for Unary, Full-Relation Operations 725
15.2.3 One-Pass Algorithms for Binary Operations . . . 728
15.2.4 Exercises for Section 15.2 . . . 732
15.3 Nested-Loop Joins . . . 733
15.3.1 Tuple-Based Nested-Loop Join . . . 733
15.3.2 An Iterator for Tuple-Based Nested-Loop Join . . . 733
15.3.3 A Block-Based Nested-Loop Join Algorithm . . . 734
15.3.4 Analysis of Nested-Loop Join . . . 736
15.3.5 Summary of Algorithms so Far . . . 736
15.3.6 Exercises for Section 15.3 . . . 736
15.4 Two-Pass Algorithms Based on Sorting . . . 737
15.4.1 Duplicate Elimination Using Sorting . . . 738
15.4.2 Grouping and Aggregation Using Sorting . . . 740
15.4.3 A Sort-Based Union Algorithm . . . 741
15.4.4 Sort-Based Intersection and Difference . . . 742
15.4.5 A Simple Sort-Based Join Algorithm . . . 743
15.4.6 Analysis of Simple Sort-Join . . . 745
15.4.7 A More Efficient Sort-Based Join . . . 746

15.4.8 Summary of Sort-Based Algorithms . . . 747
15.4.9 Exercises for Section 15.4 . . . 748
15.5 Two-Pass Algorithms Based on Hashing . . . 749
15.5.1 Partitioning Relations by Hashing . . . 750
15.5.2 A Hash-Based Algorithm for Duplicate Elimination . . . 750
15.5.3 Hash-Based Grouping and Aggregation . . . 751
15.5.4 Hash-Based Union, Intersection, and Difference . . . 751
15.5.5 The Hash-Join Algorithm . . . 752
15.5.6 Saving Some Disk I/O's . . . 753
15.5.7 Summary of Hash-Based Algorithms . . . 755
15.5.8 Exercises for Section 15.5 . . . 756
15.6 Index-Based Algorithms . . . 757
15.6.1 Clustering and Nonclustering Indexes . . . 757
15.6.2 Index-Based Selection . . . 758
15.6.3 Joining by Using an Index . . . 760
15.6.4 Joins Using a Sorted Index . . . 761
15.6.5 Exercises for Section 15.6 . . . 763
15.7 Buffer Management . . . 765
15.7.1 Buffer Management Architecture . . . 765
15.7.2 Buffer Management Strategies . . . 766
15.7.3 The Relationship Between Physical Operator Selection and Buffer Management . . . 768
15.7.4 Exercises for Section 15.7 . . . 770
15.8 Algorithms Using More Than Two Passes . . . 771
15.8.1 Multipass Sort-Based Algorithms . . . 771
15.8.2 Performance of Multipass, Sort-Based Algorithms . . . 772
15.8.3 Multipass Hash-Based Algorithms . . . 773
15.8.4 Performance of Multipass Hash-Based Algorithms . . . 773
15.8.5 Exercises for Section 15.8 . . . 774
15.9 Parallel Algorithms for Relational Operations . . . 775
15.9.1 Models of Parallelism . . . 775
15.9.2 Tuple-at-a-Time Operations in Parallel . . . 777
15.9.3 Parallel Algorithms for Full-Relation Operations . . . 779
15.9.4 Performance of Parallel Algorithms . . . 780
15.9.5 Exercises for Section 15.9 . . . 782
15.10 Summary of Chapter 15 . . . 783
15.11 References for Chapter 15 . . . 784

16 The Query Compiler **787**
16.1 Parsing . . . 788
16.1.1 Syntax Analysis and Parse Trees . . . 788
16.1.2 A Grammar for a Simple Subset of SQL . . . 789
16.1.3 The Preprocessor . . . 793
16.1.4 Exercises for Section 16.1 . . . 794

16.2 Algebraic Laws for Improving Query Plans . . . 795
16.2.1 Commutative and Associative Laws . . . 795
16.2.2 Laws Involving Selection . . . 797
16.2.3 Pushing Selections . . . 800
16.2.4 Laws Involving Projection . . . 802
16.2.5 Laws About Joins and Products . . . 805
16.2.6 Laws Involving Duplicate Elimination . . . 805
16.2.7 Laws Involving Grouping and Aggregation . . . 806
16.2.8 Exercises for Section 16.2 . . . 809
16.3 From Parse Trees to Logical Query Plans . . . 810
16.3.1 Conversion to Relational Algebra . . . 811
16.3.2 Removing Subqueries From Conditions . . . 812
16.3.3 Improving the Logical Query Plan . . . 817
16.3.4 Grouping Associative/Commutative Operators . . . 819
16.3.5 Exercises for Section 16.3 . . . 820
16.4 Estimating the Cost of Operations . . . 821
16.4.1 Estimating Sizes of Intermediate Relations . . . 822
16.4.2 Estimating the Size of a Projection . . . 823
16.4.3 Estimating the Size of a Selection . . . 823
16.4.4 Estimating the Size of a Join . . . 826
16.4.5 Natural Joins With Multiple Join Attributes . . . 829
16.4.6 Joins of Many Relations . . . 830
16.4.7 Estimating Sizes for Other Operations . . . 832
16.4.8 Exercises for Section 16.4 . . . 834
16.5 Introduction to Cost-Based Plan Selection . . . 835
16.5.1 Obtaining Estimates for Size Parameters . . . 836
16.5.2 Computation of Statistics . . . 839
16.5.3 Heuristics for Reducing the Cost of Logical Query Plans . 840
16.5.4 Approaches to Enumerating Physical Plans . . . 842
16.5.5 Exercises for Section 16.5 . . . 845
16.6 Choosing an Order for Joins . . . 847
16.6.1 Significance of Left and Right Join Arguments . . . 847
16.6.2 Join Trees . . . 848
16.6.3 Left-Deep Join Trees . . . 848
16.6.4 Dynamic Programming to Select a Join Order and Grouping 852
16.6.5 Dynamic Programming With More Detailed Cost Functions 856
16.6.6 A Greedy Algorithm for Selecting a Join Order . . . 857
16.6.7 Exercises for Section 16.6 . . . 858
16.7 Completing the Physical-Query-Plan . . . 859
16.7.1 Choosing a Selection Method . . . 860
16.7.2 Choosing a Join Method . . . 862
16.7.3 Pipelining Versus Materialization . . . 863
16.7.4 Pipelining Unary Operations . . . 864
16.7.5 Pipelining Binary Operations . . . 864
16.7.6 Notation for Physical Query Plans . . . 867

16.7.7 Ordering of Physical Operations . . . 870
16.7.8 Exercises for Section 16.7 . . . 871
16.8 Summary of Chapter 16 . . . 872
16.9 References for Chapter 16 . . . 874

17 Coping With System Failures **875**
17.1 Issues and Models for Resilicnt Operation . . . 875
17.1.1 Failure Modes . . . 876
17.1.2 More About Transactions . . . 877
17.1.3 Correct Execution of Transactions . . . 879
17.1.4 The Primitive Operations of Transactions . . . 880
17.1.5 Exercises for Section 17.1 . . . 883
17.2 Undo Logging . . . 884
17.2.1 Log Records . . . 884
17.2.2 The Undo-Logging Rules . . . 885
17.2.3 Recovery Using Undo Logging . . . 889
17.2.4 Checkpointing . . . 890
17.2.5 Nonquiescent Checkpointing . . . 892
17.2.6 Exercises for Section 17.2 . . . 895
17.3 Redo Logging . . . 897
17.3.1 The Redo-Logging Rule . . . 897
17.3.2 Recovery With Redo Logging . . . 898
17.3.3 Checkpointing a Redo Log . . . 900
17.3.4 Recovery With a Checkpointed Redo Log . . . 901
17.3.5 Exercises for Section 17.3 . . . 902
17.4 Undo/Redo Logging . . . 903
17.4.1 The Undo/Redo Rules . . . 903
17.4.2 Recovery With Undo/Redo Logging . . . 904
17.4.3 Checkpointing an Undo/Redo Log . . . 905
17.4.4 Exercises for Section 17.4 . . . 908
17.5 Protecting Against Media Failures . . . 909
17.5.1 The Archive . . . 909
17.5.2 Nonquiescent Archiving . . . 910
17.5.3 Recovery Using an Archive and Log . . . 913
17.5.4 Exercises for Section 17.5 . . . 914
17.6 Summary of Chapter 17 . . . 914
17.7 References for Chapter 17 . . . 915

18 Concurrency Control **917**
18.1 Serial and Serializable Schedules . . . 918
18.1.1 Schedules . . . 918
18.1.2 Serial Schedules . . . 919
18.1.3 Serializable Schedules . . . 920

18.1.4 The Effect of Transaction Semantics 921
18.1.5 A Notation for Transactions and Schedules 923
18.1.6 Exercises for Section 18.1 924
18.2 Conflict-Serializability 925
18.2.1 Conflicts 925
18.2.2 Precedence Graphs and a Test for Conflict-Serializability 926
18.2.3 Why the Precedence-Graph Test Works 929
18.2.4 Exercises for Section 18.2 930
18.3 Enforcing Serializability by Locks 932
18.3.1 Locks 933
18.3.2 The Locking Scheduler 934
18.3.3 Two-Phase Locking 936
18.3.4 Why Two-Phase Locking Works 937
18.3.5 Exercises for Section 18.3 938
18.4 Locking Systems With Several Lock Modes 940
18.4.1 Shared and Exclusive Locks 941
18.4.2 Compatibility Matrices 943
18.4.3 Upgrading Locks 943
18.4.4 Update Locks 945
18.4.5 Increment Locks 946
18.4.6 Exercises for Section 18.4 949
18.5 An Architecture for a Locking Scheduler 951
18.5.1 A Scheduler That Inserts Lock Actions 951
18.5.2 The Lock Table 954
18.5.3 Exercises for Section 18.5 957
18.6 Managing Hierarchies of Database Elements 957
18.6.1 Locks With Multiple Granularity 957
18.6.2 Warning Locks 958
18.6.3 Phantoms and Handling Insertions Correctly 961
18.6.4 Exercises for Section 18.6 963
18.7 The Tree Protocol 963
18.7.1 Motivation for Tree-Based Locking 963
18.7.2 Rules for Access to Tree-Structured Data 964
18.7.3 Why the Tree Protocol Works 965
18.7.4 Exercises for Section 18.7 968
18.8 Concurrency Control by Timestamps 969
18.8.1 Timestamps 970
18.8.2 Physically Unrealizable Behaviors 971
18.8.3 Problems With Dirty Data 972
18.8.4 The Rules for Timestamp-Based Scheduling 973
18.8.5 Multiversion Timestamps 975
18.8.6 Timestamps and Locking 978
18.8.7 Exercises for Section 18.8 978

18.9 Concurrency Control by Validation 979
18.9.1 Architecture of a Validation-Based Scheduler 979
18.9.2 The Validation Rules 980
18.9.3 Comparison of Three Concurrency-Control Mechanisms . 983
18.9.4 Exercises for Section 18.9 984
18.10 Summary of Chapter 18 985
18.11 References for Chapter 18 987

19 More About Transaction Management **989**
19.1 Serializability and Recoverability 989
19.1.1 The Dirty-Data Problem 990
19.1.2 Cascading Rollback 992
19.1.3 Recoverable Schedules 992
19.1.4 Schedules That Avoid Cascading Rollback 993
19.1.5 Managing Rollbacks Using Locking 994
19.1.6 Group Commit 996
19.1.7 Logical Logging 997
19.1.8 Recovery From Logical Logs 1000
19.1.9 Exercises for Section 19.1 1001
19.2 View Serializability 1003
19.2.1 View Equivalence 1003
19.2.2 Polygraphs and the Test for View-Serializability 1004
19.2.3 Testing for View-Serializability 1007
19.2.4 Exercises for Section 19.2 1008
19.3 Resolving Deadlocks 1009
19.3.1 Deadlock Detection by Timeout 1009
19.3.2 The Waits-For Graph 1010
19.3.3 Deadlock Prevention by Ordering Elements 1012
19.3.4 Detecting Deadlocks by Timestamps 1014
19.3.5 Comparison of Deadlock-Management Methods 1016
19.3.6 Exercises for Section 19.3 1017
19.4 Distributed Databases 1018
19.4.1 Distribution of Data 1019
19.4.2 Distributed Transactions 1020
19.4.3 Data Replication 1021
19.4.4 Distributed Query Optimization 1022
19.4.5 Exercises for Section 19.4 1022
19.5 Distributed Commit 1023
19.5.1 Supporting Distributed Atomicity 1023
19.5.2 Two-Phase Commit 1024
19.5.3 Recovery of Distributed Transactions 1026
19.5.4 Exercises for Section 19.5 1028

19.6 Distributed Locking 1029
19.6.1 Centralized Lock Systems 1030
19.6.2 A Cost Model for Distributed Locking Algorithms 1030
19.6.3 Locking Replicated Elements 1031
19.6.4 Primary-Copy Locking 1032
19.6.5 Global Locks From Local Locks 1033
19.6.6 Exercises for Section 19.6 1034
19.7 Long-Duration Transactions 1035
19.7.1 Problems of Long Transactions 1035
19.7.2 Sagas 1037
19.7.3 Compensating Transactions 1038
19.7.4 Why Compensating Transactions Work 1040
19.7.5 Exercises for Section 19.7 1041
19.8 Summary of Chapter 19 1041
19.9 References for Chapter 19 1044

20 Information Integration **1047**
20.1 Modes of Information Integration 1047
20.1.1 Problems of Information Integration 1048
20.1.2 Federated Database Systems 1049
20.1.3 Data Warehouses 1051
20.1.4 Mediators 1053
20.1.5 Exercises for Section 20.1 1056
20.2 Wrappers in Mediator-Based Systems 1057
20.2.1 Templates for Query Patterns 1058
20.2.2 Wrapper Generators 1059
20.2.3 Filters 1060
20.2.4 Other Operations at the Wrapper 1062
20.2.5 Exercises for Section 20.2 1063
20.3 Capability-Based Optimization in Mediators 1064
20.3.1 The Problem of Limited Source Capabilities 1065
20.3.2 A Notation for Describing Source Capabilities 1066
20.3.3 Capability-Based Query-Plan Selection 1067
20.3.4 Adding Cost-Based Optimization 1069
20.3.5 Exercises for Section 20.3 1069
20.4 On-Line Analytic Processing 1070
20.4.1 OLAP Applications 1071
20.4.2 A Multidimensional View of OLAP Data 1072
20.4.3 Star Schemas 1073
20.4.4 Slicing and Dicing 1076
20.4.5 Exercises for Section 20.4 1078
20.5 Data Cubes 1079
20.5.1 The Cube Operator 1079
20.5.2 Cube Implementation by Materialized Views 1082
20.5.3 The Lattice of Views 1085

20.5.4 Exercises for Section 20.5 1087
20.6 Data Mining 1089
20.6.1 Data-Mining Applications 1089
20.6.2 Finding Frequent Sets of Items 1092
20.6.3 The A-Priori Algorithm 1093
20.6.4 Exercises for Section 20.6 1096
20.7 Summary of Chapter 20 1097
20.8 References for Chapter 20 1098

Index **1101**

Chapter 1

The Worlds of Database Systems

Databases today are essential to every business. They are used to maintain internal records, to present data to customers and clients on the World-Wide-Web, and to support many other commercial processes. Databases are likewise found at the core of many scientific investigations. They represent the data gathered by astronomers, by investigators of the human genome, and by biochemists exploring the medicinal properties of proteins, along with many other scientists.

The power of databases comes from a body of knowledge and technology that has developed over several decades and is embodied in specialized software called a *database management system*, or *DBMS*, or more colloquially a "database system." A DBMS is a powerful tool for creating and managing large amounts of data efficiently and allowing it to persist over long periods of time, safely. These systems are among the most complex types of software available. The capabilities that a DBMS provides the user are:

1. *Persistent storage.* Like a file system, a DBMS supports the storage of very large amounts of data that exists independently of any processes that are using the data. However, the DBMS goes far beyond the file system in providing flexibility, such as data structures that support efficient access to very large amounts of data.

2. *Programming interface.* A DBMS allows the user or an application program to access and modify data through a powerful query language. Again, the advantage of a DBMS over a file system is the flexibility to manipulate stored data in much more complex ways than the reading and writing of files.

3. *Transaction management.* A DBMS supports concurrent access to data, i.e., simultaneous access by many distinct processes (called "transac-

tions") at once. To avoid some of the undesirable consequences of simultaneous access, the DBMS supports *isolation*, the appearance that transactions execute one-at-a-time, and *atomicity*, the requirement that transactions execute either completely or not at all. A DBMS also supports *durability*, the ability to recover from failures or errors of many types.

1.1 The Evolution of Database Systems

What is a database? In essence a database is nothing more than a collection of information that exists over a long period of time, often many years. In common parlance, the term *database* refers to a collection of data that is managed by a DBMS. The DBMS is expected to:

1. Allow users to create new databases and specify their *schema* (logical structure of the data), using a specialized language called a *data-definition language.*
2. Give users the ability to *query* the data (a "query" is database lingo for a question about the data) and modify the data, using an appropriate language, often called a *query language* or *data-manipulation language.*
3. Support the storage of very large amounts of data — many gigabytes or more — over a long period of time, keeping it secure from accident or unauthorized use and allowing efficient access to the data for queries and database modifications.
4. Control access to data from many users at once, without allowing the actions of one user to affect other users and without allowing simultaneous accesses to corrupt the data accidentally.

1.1.1 Early Database Management Systems

The first commercial database management systems appeared in the late 1960's. These systems evolved from file systems, which provide some of item (3) above; file systems store data over a long period of time, and they allow the storage of large amounts of data. However, file systems do not generally guarantee that data cannot be lost if it is not backed up, and they don't support efficient access to data items whose location in a particular file is not known.

Further, file systems do not directly support item (2), a query language for the data in files. Their support for (1) — a schema for the data — is limited to the creation of directory structures for files. Finally, file systems do not satisfy (4). When they allow concurrent access to files by several users or processes, a file system generally will not prevent situations such as two users modifying the same file at about the same time, so the changes made by one user fail to appear in the file.

The first important applications of DBMS's were ones where data was composed of many small items, and many queries or modifications were made. Here are some of these applications.

Airline Reservations Systems

In this type of system, the items of data include:

1. Reservations by a single customer on a single flight, including such information as assigned seat or meal preference.
2. Information about flights — the airports they fly from and to, their departure and arrival times, or the aircraft flown, for example.
3. Information about ticket prices, requirements, and availability.

Typical queries ask for flights leaving around a certain time from one given city to another, what seats are available, and at what prices. Typical data modifications include the booking of a flight for a customer, assigning a seat, or indicating a meal preference. Many agents will be accessing parts of the data at any given time. The DBMS must allow such concurrent accesses, prevent problems such as two agents assigning the same seat simultaneously, and protect against loss of records if the system suddenly fails.

Banking Systems

Data items include names and addresses of customers, accounts, loans, and their balances, and the connection between customers and their accounts and loans, e.g., who has signature authority over which accounts. Queries for account balances are common, but far more common are modifications representing a single payment from, or deposit to, an account.

As with the airline reservation system, we expect that many tellers and customers (through ATM machines or the Web) will be querying and modifying the bank's data at once. It is vital that simultaneous accesses to an account not cause the effect of a transaction to be lost. Failures cannot be tolerated. For example, once the money has been ejected from an ATM machine, the bank must record the debit, even if the power immediately fails. On the other hand, it is not permissible for the bank to record the debit and then not deliver the money if the power fails. The proper way to handle this operation is far from obvious and can be regarded as one of the significant achievements in DBMS architecture.

Corporate Records

Many early applications concerned corporate records, such as a record of each sale, information about accounts payable and receivable, or information about employees — their names, addresses, salary, benefit options, tax status, and

Offset 572–573
Offset table 580–581, 598
OID
See Object identifier
OLAP 1047, 1070–1089
See also MOLAP, ROLAP
OLD ROW/TABLE 341–344
Olken, F. 785
OLTP 1070
ON 271
On-demand swizzling 585
O'Neil, E. 424
O'Neil, P. 424, 712
One-one relationship 28–29, 140–141
One-pass algorithm 722–733, 850, 862
On-line analytic processing
See OLAP
On-line transaction processing
See OLTP
Open 720
Operand 192
Operator 192
Optical disk 512–513
Optimistic concurrency control
See Timestamp, Validation
Optimization
See Query optimization
OQL 425–449, 570
ORDER BY 251–252, 284
Ordering relationship, for UDT 458–460
Outerjoin 222, 228–230, 272–274
Output action 881, 918
Output attribute 802
Overflow block 599, 616–617, 619, 649, 656
Overloaded method 142
Ozsu, M. T. 1045

P

Pad character 570
Page 509
See also Disk block
Palermo, F. P. 874
Papadimitriou, C. H. 987, 1044
Papakonstantinou, Y. 188, 1099
Parallel computing 6–7, 775–782, 983
Parameter 392, 396–397
Parity bit 548, 552–553
Parse tree 788–789, 810
Parser 713–715, 788–795
Partial-match query 667, 681, 684, 688–689, 692
Partition attribute 438
Partitioned hash function 666, 682–684
Pascal 350
Path expression 426, 428
Paton, N. W. 348
Pattern 791
Patterson, D. A. 566
PCDATA 180
Pelagatti, G. 1044
Pelzer, T. 314
Percentiles
See Equal-height histogram
Persistence 1, 301
Persistent stored modules
See PSM
Peterson, W. W. 664
Phantom 961–962
Physical address 579, 582
Physical query plan 714–715, 787, 821, 842–845, 859–872
Piatetsky-Shapiro, G. 1099
Pinned block 586–587, 768, 995
Pipelining 859, 863–867
See also Iterator
Pippenger, N. 663
Pirahesh, H. 348, 502, 916, 1044, 1099
Plan selection 1022
See also Algorithm selection, Capability-based plan selection, Cost-based enumeration, Cost-based plan selection, Heuristic plan selection, Physical query plan, Top-down plan selection

Platter;515 517
PL/I 350
Pointer swizzling
See Swizzling
Polygraph 1004–1008
Precedence graph 926–930
Precommitted transaction 1025
Predicate 463–464
Prefetching
See Double-buffering
PREPARE 362, 392
Prepared statement 394–395
Preprocessor 793–794
Preservation, of FD's 115–116, 125
Preservation of value sets 827
Price, T. G. 874
Primary index 622
See also Dense index, Sparse index
Primary key 48, 316–317, 319, 576, 606
Primary-copy locking 1032–1033
PRIOR 361
Privilege 410–421
Probe relation 847, 850
Procedure 365, 376–377
Product 192–193, 197–198, 218, 254–255, 476, 730, 737, 796, 798–799, 803, 805, 832
Projection 112–113, 192–193, 195, 205, 216–217, 242, 245, 473, 724–725, 737, 802–805, 823, 832, 864
See also Extended projection, Pushing projections
Projection, of FD's 98–100
Prolog 501
Pseudotransitivity 101
PSM 349, 365–378
PUBLIC 410
Pushing projections 802–804, 818
Pushing selections 797, 800–801, 818
Putzolo, F. 566, 988

Q

Quad tree 666, 695–696
Quantifier
See ALL, ANY, EXISTS
Quass, D. 187, 237, 712, 785, 1099
Query 297, 466, 504–505
See also Decision-support query, Lookup, Nearest-neighbor query, Partial-match query, Range query, Where-am-I query
Query compiler 10, 14–15, 713–715, 787
See also Query optimization
Query execution 713, 870–871
Query language 2, 10
See also Datalog, OQL, Relational algebra, SQL
Query optimization 15, 714–715
See also Plan selection
Query plan 10, 14
See also Logical query plan, Physical query plan, Plan selection
Query processing 17–18, 506
See also Execution engine, Query compiler
Query processor
See Query compiler, Query execution
Query rewriting 714–715, 788, 810–821
See also Algebraic law
Quicksort 527
Quotient 213

R

RAID 551–563, 876–877
Rajaraman, A. 1099
RAM disk 514
Ramakrishnan, R. 502
Random-access memory 508
Range query 638–639, 652, 667, 673, 681, 689, 692–693

Raw-data cube 1072
See also Data cube, Fact table
Read action 881, 918
READ COMMITTED 407–408
Read lock
See Shared lock
Read set 979
Read time 970
READ UNCOMMITTED 407–408
Read-locks-one-write-locks-all 1034
Read-only transaction 403–404, 958
Real number 293, 569
Record 567, 572–577, 598–601
See also Sliding records, Spanned record, Tagged field, Variable-format record, Variable-length record
Record address
See Database address
Record fragment 595
Record header 575–576
Record structure
See Struct
Recoverable schedule 992–994
Recovery 12, 875, 889–890, 898–902, 904–905, 913, 990, 1000–1001, 1026–1028
Recovery manager 879
Recursion 463, 480–500
Redo logging 887, 897–903
Redundancy 39–40, 103, 118–119, 125
Redundant arrays of independent disks
See RAID
Redundant disk 552
Reference 133, 167, 169–171, 452, 455–456
Reference column 452–454
REFERENCES 320, 410
REFERENCING 341
Referential integrity 47, 51–53, 232
See also Foreign key
Reflexivity 99
Relation 61, 303, 463, 791, 793–794
See also Build relation, Dimension table, Fact table, Probe relation, Table, View
Relation schema 62, 66, 73, 194, 292–301
Relational algebra 189–237, 259–260, 463, 471–480, 795–808, 811
Relational atom 464
Relational database schema 24, 62, 190–191, 379–381, 383
Relational model 4–5, 61–130, 155–164, 173
See also Nested relation, Object-relational model
Relational OLAP
See ROLAP
Relationship 25, 31–32, 40–44, 67–70, 138–141, 162–163
See also Binary relationship, Isa relationship, Many-many relationship, Many-one relationship, Multiway relationship, One-one relationship, Supporting relationship
Relationship set 27
RELATIVE 361
Renaming 193, 203–205, 304–305
REPEAT 373
REPEATABLE READ 407–408
Repeating field 590–593
Replicated data 1021, 1031–1032
Resilience 875
RETURN 367
Reuter, A. 916, 988
Revoking privileges 417–421
Right outerjoin 228, 273
Right-deep join tree 848
Right-recursion 484
Rivest, R. L. 712
Robinson, J. T. 712, 988
ROLAP 1073
Role 29–31
Rollback 402, 404–405
See also Abort, Cascading rollback
Roll-up 1079

Root 174, 633
Root tag 179
Rosenkrantz, D. J. 1045
Rotation, of disk 517
Rotational latency 520, 540
See also Latency
Rothnie, J. B. Jr. 712, 987
Roussopoulos, N. 712
Row-level trigger 342
R-tree 666, 696–699
Rule 465–468
Run-length encoding 704–707

S

Safe rule 467, 482
Saga 1037–1040
Sagiv, Y. 1099
Salem, K. 566, 1044
Salton, G. 664
Schedule 918, 923–924
See also Serial schedule, Serializable schedule
Scheduler 917, 932, 934–936, 951–957, 969, 973–975, 979–980
Schema 49, 85, 167, 173, 504, 572, 575
See also Database schema, Global schema, Relation schema, Relational database schema, Star schema
Schneider, R. 711
Schwarz, P. 916, 1044
Scope, of names 269
Scrolling cursor 361
Search key 605–606, 612, 614, 623, 665
See also Hash key
Second normal form 116
Secondary index 622–625
See also Inverted index
Secondary storage 6, 510–513
See also Disk, Optical disk
Second-chance algorithm
See Clock algorithm
Sector 516, 518
Seeger, B. 711
Seek time 519–520, 535, 540
SELECT 240–243, 284, 410, 428, 431–432, 789–790
See also Single-row select
Selection 192–193, 196, 205, 217–218, 221, 241, 243, 245–246, 473–475, 724–725, 737, 758–760, 777–779, 797–801, 805, 818, 823–826, 844, 860–862, 864, 868
See also Filter, Pushing selections, Two-argument selection
Selectivity, of a join 858
Self-describing data 175
Selinger, P. G. 874
See also Griffiths, P. P.
Selinger-style optimization 845, 857
Sellis, T. K. 712
Semantic analysis
See Preprocessor
Semijoin 213
Semistructured data 16, 131, 173–178
Sequential file 606–607
Serial schedule 919–920
Serializability 397–400, 407, 918, 921–923, 927, 989–990
See also Conflict-serializability, View-serializability
Serializable schedule 920–921, 994
Server 7, 382
See also Client-server system
Session 384, 413
SET 289, 325, 367–368, 381, 383–384, 404, 729, 797–798, 803
Set type 144–145, 158–160, 166–167, 217, 446
Sethi, R. 789
Set-null policy 322
Sevcik, K. 712
Shapiro, L. D. 785
Shared disk 776, 778
Shared lock 940–942, 956

Shared memory 775–776, 778
Shared variable 352–354
Shared-nothing machine 776–777
Shaw, D. E. 785
Sheth, A. 1099
Signature 141–142
Silberschatz, A. 988
Silo 512
Simon, A. R. 314
Simple projection 802
Simplicity 40
Single-row select 354, 370
Single-value constraint 47, 51
See also Functional dependency, Many-one relationship
Size estimation 822–834, 836–839
Size, of a relation 717, 822, 840, 842
Skeen, D. 1045
Slicing 1076–1078
Sliding records 616
Smalltalk 132
Smith, J. M. 874
Smyth, P. 1099
Snodgrass, R. T. 712
Sort join 743–747, 844, 862–863
Sort key 526, 606, 636
Sorted file
See Sequential file
Sorted sublist 529, 738, 770
Sorting 222, 227–228, 526–532, 737–749, 755–756, 771–773, 845
See also ORDER BY, Ordering relationship, for UDT
Sort-scan 716–717, 719, 721–722, 868
Source 1047
Spanned record 594–595
Sparse index 609–612, 622, 636
Splitting law 797–798
Splitting nodes 640–642, 645, 698–699
Splitting rule 90–91
SQL 4–5, 131, 189, 239–424, 449–461, 492–500, 789–793
SQL agent 385
SQLSTATE 352–353, 356, 374
Srikant, R. 1099
Stable storage 548–550
Star schema 1073–1075
Start action 884
START TRANSACTION 402
Start-checkpoint action 893
Start-dump action 911
Starvation 1016–1017
State, of a database 879, 1039
Statement record 386–388
Statement-level trigger 342
Statistics 13, 836, 839–840
See also Histogram
Stearns, R. E. 1045
Stemming 629
Stern, R. C. 210
Stonebraker, M. 21, 785, 1045
Stop word 629
Storage manager 12, 17–18
See also Buffer
Stratified negation 486–490, 494–496
Strict locking 994
String 245–247, 292
See also Bit string
Stripe 676
Striping 596
Strong, H. R. 663
Struct 132–133, 137–138, 144–145, 157, 166–167, 431, 446, 568
Structured address 580–581
Sturgis, H. 566, 1044
Subclass 33–36, 76–80, 149–151
Subgoal 465
Subquery 264–276, 431–432, 812–819
See also Correlated subquery
Subrahmanian, V. S. 712
Suciu, D. 187–188, 1099
Sum 223, 279, 437
Superkey 86, 105
Support 1093
Supporting relationship 56, 72, 74–75
Swami, A. 1099
Swizzling 581–586

Syntactic category 788–789
Syntax analysis
See Parser
System failure 876–877
System R 21, 314, 874

T

Table 293, 301, 303
See also Relation
Table-scan 716, 719, 721, 861–862, 867–868
Tag 178
Tagged field 593
Tanaka, H. 785
Tape 512
Template 1058–1059
Tertiary memory 512–513
Tertiary storage 6
Thalheim, B. 60
THEN 368
Theta-join 199–201, 205, 220, 477, 731, 796–799, 802, 805, 819–820, 826–827
Theta-outerjoin 229
Third normal form
See 3NF
Thomas, R. H. 1045
Thomasian, A. 988
Thrashing 766
3NF 114–116, 124–125
Three-valued logic 249–251
Thuraisingham, B. 988
TIME 247–248, 293, 571–572
Timeout 1009–1010
TIMESTAMP 248, 575, 577, 969–979, 984, 1014–1017
Tombstone 581, 600
Top-down plan selection 843
TPMMS
See Two-phase, multiway merge-sort
Track 515–517, 579
Traiger, I. L. 987–988
Training set 1091
Transaction 1–2, 12, 17–19, 397–409, 877–883, 923–924, 1020–1021
See also Incomplete transaction, Long-duration transaction
Transaction component 1020
Transaction manager 878, 917
Transaction processing
See Concurrency, Deadlock, Locking, Logging, Scheduling
Transfer time 520, 535
Transitive rule 96–97, 121
Translation table 582–583
Tree
See B-tree, Bushy tree, Decision tree, Expression tree, Join tree, *kd*-tree, Left-deep join tree, Parse tree, Quad tree, Right-deep join tree, R-tree
Tree protocol 963–969
Trigger 315, 336, 340–345, 410–411, 876, 879
Trivial FD 92, 105
Trivial MVD 120–122, 127
Tuple 62–63, 170
See also Dangling tuple
Tuple variable 256–257
Tuple-based check 327, 330–331, 339
Turing-complete language 189
Two-argument selection 812–817
Two-pass algorithm 737–757
Two-phase commit 1024–1028
Two-phase locking 936–938
Two-phase, multiway merge-sort 0, 528–532, 536–537
Type 794, 1049
Type constructor 132
Type system 132–133, 144–146, 171

U

UDT 449–452
Ullman, J. D. 21, 130, 474, 502, 530, 726, 789, 852, 1099–1100
UNDER 410–411
UNDO 375

Undo logging 884–896
Undo/redo logging 887, 903–909
Union 192–194, 215–217, 260–262, 278, 442, 472, 722–723, 728–729, 741, 747, 751–752, 755, 779, 796–798, 803, 833
Union rule 127
UNIQUE 316–319
UNKNOWN 249–251
Unknown value 248
Unstratified negation
See Stratified negation
Unswizzling 586
Updatable view 305–307
Update 289–290, 410, 601, 615–616, 709, 1052
See also Modification
Update anomaly 103
Update lock 945–946
Update record 885–886, 897, 903
Upgrading locks 943–945, 957
See also Update lock
USAGE 410
User-defined type
See UDT
Uthurusamy, R. 1099

V

Valduriez, P. 1045
Valid XML 178–179
Validation 969, 979–985
Value count 719, 822, 840
VALUES 286
Van Gelder, A. 502
VARCHAR 292
Variable-format record 590, 593–594
Variable-length record 570–571, 589–594, 998–999
Vassalos, V. 1099
Vertical decomposition 1020
Vianu, V. 21
View 301–312, 345, 1053
See also Materialized view
View-serializability 1003–1009
Virtual memory 509–510, 578
Vitter, J. S. 566
Volatile storage 513–514

W

Wade, B. W. 424
Wait-die 1014–1017
Waiting bit 955
Waits-for graph 1010–1012
Walker, A. 502
Warehouse 1048, 1051–1053, 1071
Warning protocol 958–961
Weak entity set 54–59, 71–75, 154
Weiner, J. L. 187
Well-formed XML 178–180
Westwood, J. N. 210
WHEN 340, 342
WHERE 240–241, 243–244, 264, 284, 288, 428–429, 789
Where-am-I query 667, 697
WHILE 373
White, S. 424
Widom, J. 187–188, 348, 1099
Wiederhold, G. 604, 1100
WITH 492–493
Wong, E. 21, 874
Wood, D. 785
Workflow 1036
World-Wide-Web consortium 187
Wound-wait 1014–1017
Wrapper 1048, 1057–1064
Wrapper generator 1059–1060
Write action 881, 918
Write failure 546, 550
See also System failure
Write lock
See Exclusive lock
Write set 979
Write time 970
Write-ahead logging rule 897
See also Redo logging
Write-through cache 508

X

XML 16, 131–132, 173, 178–186, 629

Y

Yerneni, R. 1100
Youssefi, K. 874

Z

Zaniolo, C. 130, 712
Zicari, R. 712
Zig-zag join 762–763
Zip disk 513
Zipfian distribution 632, 825