
Index

Symbols

- * (asterisk), in SQL, 171
- ? (question mark), parameter markers, 268
- \ (backward slash), in system variables, 64, 91
- '' (single quotation marks), character string delimiters, 143
- 2.6 kernel
 - installation problems on, 83
 - troubleshooting installation, 526

A

- ACID (atomicity, consistency, isolation, and durability) compliance, 14
- addBatch() method, 245, 273
- ADD COLUMN clause, 155
- ADD CONSTRAINT clause, 155
- ADD PRIMARY KEY statement, 155
- ADD UNIQUE statement, 155

- administration, 101
 - backups, 101
 - offline backup, 104-105
 - online backup with built-in copy, 102-103
 - online backup with read-only access, 103-104
 - restoring from, 105
 - compressing tables, 119
 - connection options, 43-45
 - consistency checks, 112-113
 - data movement operations, 105-106
 - exporting data, 109-111
 - importing data, 106-109
 - performance issues, 113
 - collecting statistics, 113-115
 - timing information in statistics, 115-118
- afterLast() method, 254
- after triggers, 225

- algorithm values for encryption
 - Algorithm option, 125-126
- aliases, 189
- allocating dynamic SQL statements (JDBC application development), 244-246
- ALL quantified predicate, 202
- ALTER statement, 139
- ALTER TABLE statement, 155-156
 - check constraints, 162-163
- Anderson, Jean, 4
- answer sets, 244
 - accessing in random order, 533
 - dynamically determining (JDBC application development), 264-267
 - exporting data, 110-111
 - modifying data with (JDBC application development), 274-279
 - multiple result sets, 288
 - retrieving
 - JDBC application development, 252-258
 - Perl application development, 450-451
 - PHP application development, 416-417
 - Python application development, 494-495
 - retrieving column values (JDBC application development), 258-264
 - stored procedures, 281, 284-287
- ANY quantified predicate, 202
- Apache Derby
 - advantages of, 3-5
 - application development, 23-24
 - attributes, network server connections, 241
 - business perspective, 10-11, 15-17
 - community hierarchy, 7-9
 - components of, 21
 - Apache Derby database engine, 21
 - Apache Derby network server, 21-22
 - ij tool, 22-23
 - database administration. *See* database administration
 - database engine, 21
 - databases. *See* databases
 - deployment model, 25
 - embedded framework, 25-31
 - server framework, 32-36
 - differences from IBM Cloudscape, 3-7
 - features of, 17-19
 - history of, 1-3
 - IBM support for, 2
 - installing on Linux, 75-76
 - manual installation, 90-93
 - migrating from previous versions, 76
 - post-installation file structure, 98-99
 - preparations for, 76-81
 - troubleshooting installation, 96-98
 - verifying installation, 93-96
 - installing on Windows, 49-50
 - manual installation, 63-66
 - migrating from previous versions, 50
 - post-installation file structure, 72-73
 - preparations for, 50-55
 - troubleshooting installation, 70-72
 - verifying installation, 67-69
 - licensing, 5
 - performance tuning, 526-527
 - security. *See* security
 - SQL support in, 19-21
 - technical perspective, 11
- Apache Derby Charter, 19
- Apache Derby Network Server, 21-22
 - applets, 299-301
 - cataloging, 311
 - communication with, 307
 - CLI, 307
 - DRDA protocol, 307
 - ODBC, 308
 - Perl/PHP/Python extensions, 308

- connections (JDBC application development), 240-244
- IBM DB2 JDBC Universal driver, 237-238
- installing DB2 Runtime Client, 308-311
- server framework deployment model, 32-36
- troubleshooting, 533
- Apache incubator projects, 5
- Apache License, 5
- applets (JDBC application development), 299-301
- application development, 23-24t
 - JDBC application development, 233-234
 - allocating dynamic SQL statements, 244-246
 - applets, 299-301
 - batch methods, 274
 - batch processing, 273-274
 - database connections, 238-239
 - dynamically determining result sets, 264-267
 - embedded JDBC drivers, 236-237
 - executing dynamic SQL statements, 246-248
 - IBM DB2 JDBC Universal driver, 237-238
 - import statements, 235-236
 - inserting and updating large objects, 279-281
 - locking issues, 302-305
 - modifying data with result sets, 274-279
 - network server connections, 240-244
 - prepared statements, 268-272
 - program structure, 234-235
 - retrieving column values, 258-264
 - retrieving result sets, 252-258
 - SQLExceptions versus SQLWarnings, 251-252
 - stored procedures, 281-288
 - transactions, 272
 - trapping SQLExceptions, 248-250
 - troubleshooting, 532-535
 - UDFs (user-defined functions), 288-299
- Perl application development
 - connection pooling, 446-447
 - database connections, 445-446
 - disconnecting from database, 446
 - GUI interfaces, 467-472
 - large object (LOB) data types, 456-458
 - module usage, 444
 - running scripts from command line, 445
 - SQL statements, 447-454
 - transaction management, 454-455
 - Web interfaces, 459-465
- PHP application development, 411
 - connection pooling, 413
 - database connections, 412-413
 - disconnecting from database, 413
 - large object (LOB) data types, 424-425
 - running scripts from command line, 411-412
 - SQL statements, 414-421
 - transaction management, 421-424
 - Web interfaces, 426-436
- Python application development
 - connection pooling, 491-492
 - database connections, 490
 - disconnecting from database, 491
 - GUI interfaces, 512-518
 - large object (LOB) data types, 501-503
 - module usage, 489-490
 - running scripts from command line, 490
 - SQL statements, 492-499
 - transaction management, 499-501
 - Web interfaces, 503-512
- arrays (PHP), 396-397. *See also* sequences (Python)
- asterisk (*), in SQL, 171

authentication, 126-128, 313
 built-in authentication, 130-132, 528
 LDAP directory service authentication,
 128-129
 precedence of authentication levels, 528
 remote authentication, 528
 user-defined authentication, 129
 authorization, 132-133
 authorization identifiers, 126-127
 autocommit() method, 500-501
 autocommit settings, 534
 JDBC application development, 272
 Perl application development, 455
 PHP application development, 423-424
 automatic global arrays (PHP), 427
 autonumbering. *See* identity columns
 AVG function, 194

B

backups, 101
 offline backup, 104-105
 online backup with built-in copy, 102-103
 online backup with read-only access,
 103-104
 restoring from, 105
 backward slash (\), in system variables, 64, 91
 basic predicates, 173
 batch methods (JDBC application
 development), 274
 batch processing (JDBC application
 development), 273-274
 bCalendar() method (YMLD example
 application), 344
 beforeFirst() method, 253
 begin_work() method, 454
 beginTransaction() method, 422
 BETWEEN predicate, 175

BIGINT data type, 141
 binary large objects. *See* BLOBs
 binary string data types, 144
 bind_columns() method, 450
 bind_param() method, 449
 bind_param_inout() method, 451, 458
 bindColumn() method, 424
 binding parameters to prepared statements
 (Perl application development), 449
 bindParam() method, 419, 425
 blob_read() method, 456
 BLOBs (binary large objects), 145. *See*
 also LOBs
 inserting/updating, 534
 Boolean data type
 PHP, 395-396
 Python, 476-477
 booting encrypted databases, 527
 bootPassword parameter, 43, 123
 break statement
 PHP, 401, 404
 Python, 483
 built-in authentication, 130-132, 528
 built-in functions, 191
 business perspective
 Apache Derby, 15-17
 open source, 10-11

C

cached connections (Perl), 446
 calculated columns. *See* derived columns
 callers
 Perl scripts, 459
 PHP scripts, 426
 Python scripts, 504
 calling stored procedures, 284
 callproc() method, 496-497

- Call Level Interface. *See* CLI
- CALL statement, 284
- Cartesian product, 179-181
- cascading triggers, 225
- CASE expressions, 204-205, 532
- case-sensitivity
 - of database names, 312
 - of user IDs/passwords, 130
- casting functions, 192
- CAST function, 190
- cataloging
 - Apache Derby Network Server, 311
 - databases, 311-312
- CGI (Common Gateway Interface), 426
 - GET/POST variable handling
 - Perl application development, 459-461
 - PHP application development, 426-430
 - Python application development, 504-507
- cgi module (Python application development), 504
- CGI.pm module (Perl application development), 439, 459
- CHAR data type, 140, 143
- character data types, 46. *See also* string data types
- character large objects, 145
- check constraints, 157, 161-163
- classes, creating
 - PHP, 404-406
 - Python, 484-485
- CLASSPATH system variable
 - setting, 62-66, 90-93
 - troubleshooting, 70-71, 96-97
- clearBatch() method, 245, 273
- clearWarnings() method, 252
- CLI (Call Level Interface), 307
 - Perl/PHP/Python extensions, 308
- ClickButton method (YMLD example application), 378-381
- CLOBs (character large objects), 145, 530.
 - See also* LOBs
 - inserting/updating, 534
- close method, 245, 491
- closing
 - dynamic SQL statements, 245
 - statements, 533
- Cloudscape
 - differences from Apache Derby, 3-7
 - history of, 1-3
 - installing on Linux, 75-76
 - with installation program, 81-90
 - manual installation, 90-93
 - migrating from previous versions, 76
 - post-installation file structure, 98-99
 - preparations for, 76-81
 - troubleshooting installation, 96-98
 - verifying installation, 93-96
 - installing on Windows, 49-50
 - with installation program, 55-63
 - manual installation, 63-66
 - migrating from previous versions, 50
 - post-installation file structure, 72-73
 - preparations for, 50-55
 - troubleshooting installation, 70-72
 - verifying installation, 67-69
 - licensing, 6
 - SQL support in, 19-21
 - support contract, 6
- CLOUDSCAPE_INSTALL environment variable, setting, 62-65, 90-91
- CLR (Common Language Runtime), 52
- column functions, 191-194
 - restriction with, 195-196
- column values, retrieving (JDBC application development), 258-264

- columns, 46
- command line, running scripts from
 - Perl, 445
 - PHP, 411-412
 - Python, 490
- commit() method, 422, 454, 500
- COMMIT statement, 231-232
 - JDBC application development, 272
- committers, 8-9
- Common Gateway Interface. *See* CGI
- Common Language Runtime (CLR), 52
- comparison operators
 - PHP, 399
 - in predicates, 173
 - Python, 481
- Comprehensive Perl Archive Network (CPAN), 439
- compressing tables, 119
- conditions
 - PHP, 399
 - Python, 481
- configuring
 - Perl
 - on Linux, 442-443
 - on Windows, 443-444
 - PHP
 - on Linux, 409-410
 - on Windows, 410-411
 - Python
 - on Linux, 488
 - on Windows, 489
- Confirm_Click() method (YMLD example application), 384
- connect() method, 445, 490
- connection contexts, stored procedures, 281
- connection handles, 308
- Connection objects (Python)
 - database connections, 490
 - disconnecting from database, 491
 - transaction management, 499-501
- connection options
 - environment, 45
 - maintenance, 43-45
 - security, 43
- connection pooling
 - Perl application development, 446-447
 - PHP application development, 413
 - Python application development, 491-492
- connection URL, creating databases, 39
- connections
 - Apache Derby Network Server. *See* Apache Derby Network Server
 - establishing (JDBC application development), 238-239
 - Perl application development, 445-446
 - connection pooling, 446-447
 - disconnections, 446
 - PHP application development, 412-413
 - connection pooling, 413
 - disconnections, 413
 - from procedures, 535
 - Python application development, 490
 - connection pooling, 491-492
 - disconnections, 491
 - testing, 312
 - troubleshooting, 526
 - uncataloged database connections, 313
 - YMLD example application, 327-329, 360-364
- consistency checks, 112-113
- constraints, 156, 168
 - ADD CONSTRAINT clause, 155
 - check constraints, 161-163

- DROP CONSTRAINT clause, 156
 - referential integrity constraints, 157-161
 - unique constraints, 157
- UPDATE statement, 217
- continue statement
 - PHP, 404
 - Python, 483
- contributors, 8
- control structures
 - PHP, 399
 - break statement, 404
 - continue statement, 404
 - do/while loops, 399-400
 - foreach loops, 402-403
 - for loops, 401-402
 - if/elseif/else structures, 400
 - switch structures, 401
 - while loops, 399-400
 - Python, 481
 - break statement, 483
 - continue statement, 483
 - for loops, 482-483
 - if/elif/else structure, 482
 - pass statement, 483
 - while loops, 481-482
- converting data types, 260
- correlated subqueries, 201
- correlation names, 186-187
- COUNT function, 194
 - DISTINCT keyword, 198
- CPAN (Comprehensive Perl Archive Network), 439
- CREATE FUNCTION statement, 223, 292-293
- CREATE INDEX statement, 169
- CREATE PROCEDURE statement, 230-231, 282-284

- CREATE SCHEMA statement, 148, 222
- CREATE statement, 139
- CREATE TABLE statement, 147-148
- CREATE TRIGGER statement, 227
- CREATE VIEW statement, 165-166
- createFrom parameter, 44
- createStatement() method, 244-245
- CURRENT_DATE function, 150
- CURRENT_SCHEMA function, 150
- CURRENT_TIMESTAMP function, 150
- CURRENT_TIME function, 150
- CURRENT_USER function, 150
- cursor() method, 492
- Cursor objects (Python), SQL statements
 - calling stored procedures, 495-499
 - issuing, 492
 - with different parameters, 493-494
 - with placeholders, 492
 - retrieving result sets, 494-495
- cursor stability isolation level, 302
- cursors, 276

D

- Data Control Language. *See* DCL
- Data Definition Language. *See* DDL
- Data Manipulation Language. *See* DML
- data movement operations, 105-106
 - exporting data, 109-111
 - importing data, 106-109
- data truncation, preventing in ij tool, 114
- data types, 46, 140
 - binary string data types, 144
 - converting, 260
 - date and time data types, 145-147
 - importance of, 530

- large objects (LOBs), 144-145, 530
 - Perl application development, 456-458
 - PHP application development, 424-425
 - Python application development, 501-503
- manipulating, 534
- modifiers, 148
 - default values, 149-151
 - identity columns, 151-153
 - null values, 149
- numeric data types, 140-142
- PHP data types
 - arrays, 396-397
 - Boolean, 395-396
 - numeric, 396
 - strings, 397-399
- promotion, 190
- Python data types
 - Boolean, 476-477
 - dictionaries, 480-481
 - numeric, 477-478
 - sequences, 478-479
 - strings, 479-480
- setNull() method, 278-279
- SQL data types
 - versus Java data types, 258, 366
 - versus ODBC data types, 385
- stored procedures, 281
- string data types, 140-144
- database administration, 101
 - backups, 101
 - offline backup, 104-105
 - online backup with built-in copy, 102-103
 - online backup with read-only access, 103-104
 - restoring from, 105
 - compressing tables, 119
 - connection options, 43-45
 - consistency checks, 112-113
 - data movement operations, 105-106
 - exporting data, 109-111
 - importing data, 106-109
 - performance issues, 113
 - collecting statistics, 113-115
 - timing information in statistics, 115-118
- database connections
 - Apache Derby Network Server. *See* Apache Derby Network Server
 - establishing (JDBC application development), 238-239
 - Perl application development, 445-446
 - connection pooling, 446-447
 - disconnections, 446
 - PHP application development, 412-413
 - connection pooling, 413
 - disconnections, 413
 - from procedures, 535
 - Python application development, 490
 - connection pooling, 491-492
 - disconnections, 491
 - testing, 312
 - troubleshooting, 526
 - uncataloged database connections, 313
 - YMLD example application, 327-329, 360-364
- database handles (Perl)
 - connection pooling, 446-447
 - database connections, 445-446
 - disconnecting from database, 446
- SQL statements
 - calling stored procedures, 451-454
 - issuing, 447

- issuing with prepared statements, 447-449
 - retrieving result sets, 450-451
 - transaction management, 454-455
- database installation, verifying, 67, 93-94
- database name attribute, network server connections, 241
- database objects, 528. *See also* data types; indexes; procedures; schemas; tables; triggers; views
- CREATE statement, 139
 - definition of, 37
 - DROP statement, 140
 - list of, 37, 45, 135
 - privileges, 163
 - schemas, 221-223
 - size limitations, 38
 - system catalog tables, 163-164
- database statistics. *See* statistics
- database-level authentication, precedence, 528
- database-level authorization, 132-133
- database-level users
- built-in authentication, 131
 - versus system-level users, precedence of, 132
- database-wide properties, 127
- databases
- cataloging, 311-312
 - connection options
 - environment, 45
 - maintenance, 43-45
 - security, 43
 - creating, 39-41, 529
 - deleting, 41, 529
 - directory structure, 37-38
 - encrypted databases, booting, 527
 - encryption, 527
 - generated files, 41
 - naming, 39-40, 312, 529
 - security. *See* security
 - system tables, list of, 41-42
 - testing connections, 312
 - troubleshooting, 528-529
 - uncataloged connections, 313
- dataEncryption parameter, 43, 123
- date and time data types, 46, 145-147
- DATE data type, 145-146
- date functions, 193
- DB2 JDBC drivers. *See* IBM DB2 JDBC Universal driver
- DB2 Runtime Client, installing, 308-311
- DB2 Universal Database (DB2 UDB), 2
- DBD:DB2 (Perl module), 440
- DBI (Perl module), 439
- DCL (Data Control Language), 135
- DDL (Data Definition Language), 135-138
- ALTER statement, 139
 - ALTER TABLE statement, 155-156
 - constraints, 156
 - check constraints, 161-163
 - referential integrity constraints, 157-161
 - unique constraints, 157
 - CREATE statement, 139
 - CREATE TABLE statement, 147-148
- data types, 140
- binary string data types, 144
 - date and time data types, 145-147
 - large object data types, 144-145
 - numeric data types, 140-142
 - string data types, 140-144
- data type modifiers, 148
- default values, 149-151
 - identity columns, 151-153
 - null values, 149
- DECLARE statement, 138-140, 153-154

- DROP statement, 140
- DROP TABLE statement, 156
- indexes, 167-169
- system catalog tables, 163-164
- views, 165-167
- DECIMAL data type, 46, 142
- DECLARE statement, 138-140, 153-154, 530
- declaring variables
 - PHP, 395
 - Python, 476
- default values, data type modifiers, 149-151
- deferPrepares attribute, network server connections, 242
- defining functions
 - PHP, 404
 - Python, 483-484
- DELETE rules (referential constraints), 160-161
- DELETE statement, 219-221
- deleting databases, 41, 529
- delimited identifiers, 138
- delimiters in character strings, 143
- delimiting PHP code, 394
- dependent tables, 158
- deployment model, 25
 - embedded framework, 25-31
 - server framework, 32-36
- derived columns, 189-191
- dictionaries (Python), 480-481
- directory for IBM Cloudscape on Linux
 - installation, 86
- directory structure of databases, 37-38
- dirty read. *See* uncommitted read
 - isolation level
- disaster recovery. *See* backups
- disconnect() method, 446
- disconnecting from database
 - Perl application development, 446
 - PHP application development, 413
 - Python application development, 491
- DISTINCT keyword, 196-198
- Distributed Relational Database Architecture.
 - See* DRDA protocol
- DML (Data Manipulation Language), 135, 170
 - DELETE statement, 219-221
 - functions, 191
 - column functions, 193-194
 - scalar functions, 191-193
 - INSERT statement, 213-215
 - multiple rows, 215-216
 - for specific columns, 215
 - subselects, 216-217
 - SELECT statement
 - Cartesian product, 179-181
 - CASE expressions, 204-205
 - correlation names, 186-187
 - derived columns, 189-191
 - DISTINCT keyword, 196-198
 - EXISTS predicate, 212-213
 - GROUP BY clause, 194-196
 - joins, 181-186
 - nested table expressions, 205-206
 - ORDER BY clause, 187-189
 - permutation, 172
 - projection, 171-172
 - quantified predicates, 202-204
 - restriction, 173-178
 - retrieving entire tables, 170-171
 - scalar fullselects, 206-209
 - subqueries, 198-202
 - UNION operator, 209-212
 - UPDATE statement, 217-219

- do() method, 447
 - do/while loops (PHP), 399-400
 - documentation
 - Perl, 440
 - PHP, 392
 - Python, 474
 - DOUBLE data type, 142
 - DRDA (Distributed Relational Database Architecture) protocol, 307, 533
 - DROP CONSTRAINT clause, 156
 - DROP INDEX statement, 169
 - DROP statement, 140
 - DROP TABLE statement, 156, 219
 - DROP VIEW statement, 167
 - duplicate rows, eliminating, 196-198
 - dynamically determining result sets (JDBC application development), 264-267
 - DYNAMIC RESULT SETS clause, 231, 285
 - dynamic SQL statements (JDBC application development)
 - allocating, 244-246
 - executing, 246-248
- E**
- Eclipse, 315, 325
 - IBM DB2 plug-in for Eclipse, 23
 - editing environments
 - for Perl, 441
 - for PHP, 393
 - for Python, 475
 - embedded framework (deployment model), 25-31
 - embedded .JAR files, 291-292
 - embedded JDBC drivers, 532
 - JDBC application development, 236-237
 - encrypted databases, booting, 527
 - encryption, 123-126, 527
 - encryptionAlgorithm parameter, 43, 123-125
 - algorithm values, 125-126
 - feedback modes, 126
 - encryptionProvider parameter, 43, 123
 - environment connection options, 45
 - environment handles, 308
 - environment variables
 - setting, 529
 - on Linux, 525
 - on Windows, 526
 - versus system variables, 59
 - equality predicate, 173
 - with ALL predicate, 204
 - ErrorCode values, 249-250
 - error handling, stored procedures, 287
 - error messages (YMLD example application), 363-364, 367-368
 - errors. *See also* SQLExceptions
 - database-level authorization, 133
 - getSQLException() method, 334
 - escape processing, 246
 - exceptions. *See* SQLExceptions
 - exec() method, 414
 - execute() method, 414, 448-449, 457, 492
 - executeBatch() method, 245, 273
 - executemany() method, 493
 - ExecuteNonQuery() method (YMLD example application), 385
 - executeQuery() method, 247, 252
 - executeUpdate() method, 247, 252
 - executing dynamic SQL statements (JDBC application development), 246-248
 - execution plan and statistics, 113
 - EXISTS predicate, 212-213
 - exitYMLD() method, 355
 - exporting data, 109-111
 - expressions, parentheses in, 532
 - extensions (PHP), 391-392. *See also* modules
 - EXTERNAL NAME clause, 223, 231, 292

F

feedback modes for encryptionAlgorithm
 option, 126

fetch() method, 416-417, 420, 450, 453

fetchall() method, 495-498

fetchmany() method, 495-498

fetchone() method, 495-498

file system, physical security, 121-123

first() method, 253

flat file systems, 13

floating point data types, 46

FLOAT data type, 142

foreach loops (PHP), 402-403

foreign keys, 158, 168

FOR BIT DATA specification, 144

FOR FETCH ONLY clause, 275

for loops
 PHP, 401-402
 Python, 482-483

FOR READ ONLY clause, 275

FOR UPDATE clause, 274-278

FROM clause, 170-171

fullselects. *See* scalar fullselects

functions, 191
 column functions, 193-194
 defining
 PHP, 404
 Python, 483-484

MONTH, 337, 372

scalar functions, 191-193, 223

UDFs (user-defined functions), 223-224
 JDBC application development,
 288-299

G

generated values. *See* identity columns

Get() methods (ODBC), 366

GET variables, handling
 Perl application development, 459-461
 PHP application development, 426-430
 Python application development, 504-507

getAsciiStream() method, 263

getAttribute() method, 423

getAutoCommit() method, 272

getBinaryStream() method, 262

getBlob() method, 262-264

getBytes() method, 262

getCatalogName() method, 266

getClob() method, 262-264

getColumnClassName() method, 265

getColumnCount() method, 265

getColumnDisplaySize() method, 265

getColumnLabel() method, 265

getColumnName() method, 265

getConnection() method, 238

getErrorCode() method, 249-250

getFetchDirection() method, 245

getFetchSize() method, 245

getMaxFieldSize() method, 245

getMaxRows() method, 246

getMessage() method, 248

getMoreResults() method, 288

getNextException() method, 250

getNextWarning() method, 251

getPrecision() method, 265

getProdDates() method (YMLD example
 application), 343-344, 376-377

getResultSet() method, 287-288

getScale() method, 265

getSchemaName() method, 266
getSQLException() method, 334
getSQLState() method, 249
getSubString() method, 263
getTableName() method, 265
getUnicodeStream() method, 263
getX() methods, 259-260
 converting data types, 260
global arrays (PHP), 427
graphical installation program, troubleshoot-
ing installation, 525
graphical interfaces. *See* GUI interfaces
GROUP BY clause, 187, 194-196
GUI interfaces
 Perl application development, 467-472
 Python application development, 512-518

H

Haderle, Don, 2
handlers
 Perl scripts, 459
 PHP scripts, 426
 Python scripts, 504
HAVING clause, 196, 200
history
 of IBM Cloudscape/Apache Derby, 1-3
 of PHP, 391
 of Python, 473
HTTP headers, setting
 PHP application development, 426
 Python application development, 503-504
Hypertext Transfer Protocol headers. *See*
 HTTP headers

I

IBM, commitment to open source, 9-10
IBM Cloudscape
 differences from Apache Derby, 3-7
 history of, 1-3
 installing on Linux, 75-76
 with installation program, 81-90
 manual installation, 90-93
 migrating from previous versions, 76
 post-installation file structure, 98-99
 preparations for, 76-81
 troubleshooting installation, 96-98
 verifying installation, 93-96
 installing on Windows, 49-50
 with installation program, 55-63
 manual installation, 63-66
 migrating from previous versions, 50
 post-installation file structure, 72-73
 preparations for, 50-55
 troubleshooting installation, 70-72
 verifying installation, 67-69
 licensing, 6
 SQL support in, 19-21
 support contract, 6
IBM Cloudscape system home directory,
 64, 91
IBM DB2 JDBC Universal driver,
 237-238, 532
 connections (JDBC application
 development), 240-244
 options, 533
IBM DB2 plug-in for Eclipse, 23
IBM developerWorks Web site, 6
identifiers, naming standards, 138
IDENTITY_VAL_LOCAL function, 153

- identity columns, 531
 - data type modifiers, 151-153
- if/elif/else structure (Python), 482
- if/elseif/else structures (PHP), 400
- ij tool, 22-23
 - creating databases, 40-41
 - preventing data truncation, 114
- importing
 - data, 106-109
 - modules (Python), 486-487
- import statements
 - JDBC application development, 235-236
 - Python, 489-490
- include files (PHP), 406-407
- include statement (PHP), 407
- incubator projects. *See* Apache incubator projects
- indexes, 47, 167-169, 531
 - and null values, 531
- initializing
 - variables for mod_perl, 452
 - YMLD example application, 326-327, 360
- inline views. *See* nested table expressions
- inner joins, 183
- INOUT parameters, calling stored
 - procedures, 231
 - Perl application development, 451-452
 - PHP application development, 419-420
 - Python application development, 498
- input-filtering functions (PHP), 431
- input streams, inserting/updating large objects, 279-281
- INSERT rule (referential constraints), 158-160
- INSERT statement, 213-215
 - default values, 150-151
 - multiple rows, 215-216
 - for specific columns, 215
 - subselects, 216-217
- installing
 - Apache Derby/IBM Cloudscape on Linux, 75-76
 - with installation program, 81-90
 - manual installation, 90-93
 - migrating from previous versions, 76
 - post-installation file structure, 98-99
 - preparations for, 76-81
 - troubleshooting installation, 96-98
 - verifying installation, 93-96
 - Apache Derby/IBM Cloudscape on Windows, 49-50
 - with installation program, 55-63
 - manual installation, 63-66
 - migrating from previous versions, 50
 - post-installation file structure, 72-73
 - preparations for, 50-55
 - troubleshooting installation, 70-72
 - verifying installation, 67-69
 - DB2 Runtime Client, 308-311
 - Perl, 441-442
 - on Linux, 442
 - on Windows, 443
 - PHP, 407
 - on Linux, 408-409
 - on Windows, 410
 - Python, 487
 - on Linux, 487
 - on Windows, 488-489
 - troubleshooting installation, 525-526
 - YMLD example application, 326, 359-360
- INTEGER data type, 46, 141
- interfaces
 - GUI interfaces
 - Perl application development, 467-472
 - Python application development, 512-518

- Web interfaces
 - cgi module (Python application development), 504
 - CGI.pm module (Perl application development), 459
 - handling GET/POST variables, 426-430, 459-461, 504-507
 - sessions, 435-436, 466, 511-512
 - setting HTTP headers, 426, 503-504
 - taint-checking user input, 430-432, 462, 507-508
 - XHTML conflicts with Perl reserved names, 465
 - YMLD example, 432-435, 462-465, 508-511
 - IN parameters, calling stored procedures, 176, 231
 - PHP application development, 417-418
 - Python application development, 496-497
 - IS NOT NULL predicate, 176-177
 - IS NULL predicate, 176-177
 - is_bool() method, 431
 - is_float() method, 431
 - is_int() method, 431
 - is_numeric() method, 431
 - is_string() method, 431
 - isAfterLast() method, 254
 - isAutoIncrement() method, 266
 - isBeforeFirst() method, 254
 - isCaseSensitive() method, 266
 - isCurrency() method, 266
 - isDefinitelyWritable() method, 266
 - isFirst() method, 254
 - isLast() method, 254
 - isNullable() method, 266
 - isolation levels
 - changing, 303-304
 - choosing, 303
 - cursor stability, 302
 - read stability, 302
 - repeatable read, 303
 - statement-level isolation levels, 304-305
 - uncommitted read, 302
 - isReadOnly() method, 266
 - isSearchable() method, 266
 - isSigned() method, 266
 - isWritable() method, 266
 - iterators. *See* control structures
- J**
- jar directory, 41
 - .JAR files, embedded, 291-292
 - Java. *See also* JDBC application development
 - benefits of, 233-234
 - data types, versus SQL data types, 258, 366
 - null values, 534
 - versus SQL null values, 260-262
 - Java Common Client (JCC), 6
 - Java Database Connectivity. *See* JDBC application development
 - Java Developer's Kit. *See* JDK
 - Java Runtime Environment. *See* JRE
 - Java Virtual Machine. *See* JVM
 - JCC (Java Common Client), 6
 - JCE providers, list of, 124-125
 - JDBC application development, 233-234
 - allocating dynamic SQL statements, 244-246
 - applets, 299-301
 - batch methods, 274
 - batch processing, 273-274
 - database connections, 238-239
 - dynamically determining result sets, 264-267
 - embedded JDBC drivers, 236-237

- executing dynamic SQL statements, 246-248
 - IBM DB2 JDBC Universal driver, 237-238
 - import statements, 235-236
 - inserting and updating large objects, 279-281
 - locking issues, 302-305
 - modifying data with result sets, 274-279
 - network server connections, 240-244
 - prepared statements, 268-272
 - program structure, 234-235
 - retrieving
 - column values, 258-264
 - result sets, 252-258
 - SQLExceptions versus SQLWarnings, 251-252
 - stored procedures, 281-288
 - transactions, 272
 - trapping SQLExceptions, 248-250
 - troubleshooting, 532-535
 - UDFs (user-defined functions), 288-299
 - YMLD example application
 - database connections, 327-329
 - exit routine, 355
 - functions of, 325
 - initializing, 326-327
 - installing, 326
 - Java components, 324-325
 - purchasing tickets, 336-351
 - retrieving current performances, 329-334
 - retrieving seat map, 335-336
 - retrieving seat prices, 334-335
 - viewing transaction log, 351-355
 - JDBC drivers, embedded, 532
 - JDBC scripting tool (ij tool), 22-23
 - creating databases, 40-41
 - preventing data truncation, 114
 - JDK (Java Developer's Kit), 52, 78
 - jEdit Java editor, 324
 - join predicates, 183
 - joins, 181-186
 - JRE (Java Runtime Environment), 51, 77
 - checking version of, 52, 78
 - installing for IBM Cloudscape/Apache Derby, 51, 77
 - setting PATH system variable for, 52-55, 78-81
 - JVM (Java Virtual Machine), 51, 77
- K-L**
- LANGUAGE clause, 224, 231, 292
 - large objects (LOBs), 144-145, 530
 - getBlob() and getClob() methods, 262-264
 - inserting/updating, 534
 - JDBC application development, 279-281
 - Perl application development, 456-458
 - PHP application development, 424-425
 - Python application development, 501-503
 - last() method, 253
 - LDAP directory service authentication, 128-129
 - left outer joins, 184-186
 - length() method, 263
 - licensing
 - Apache Derby, 5
 - IBM Cloudscape, 6
 - LIKE predicate, 174-175
 - Linux
 - configuring
 - Perl, 442-443
 - PHP, 409-410
 - Python, 488
 - environment variables, setting, 525

- installing
 - DB2 Runtime Client, 309-310
 - Perl, 442
 - PHP, 408-409
 - Python, 487
- installing Apache Derby/IBM Cloudscape, 75-76
 - with installation program, 81-90
 - manual installation, 90-93
 - migrating from previous versions, 76
 - post-installation file structure, 98-99
 - preparations for, 76-81
 - troubleshooting installation, 96-98
 - verifying installation, 93-96
- running
 - Perl scripts, 445
 - PHP scripts, 412
 - Python scripts, 490
- lists (Python), 479
- LOBs (large objects), 144-145, 530
 - getBlob() and getClob() methods, 262-264
 - inserting/updating, 534
 - JDBC application development, 279-281
 - Perl application development, 456-458
 - PHP application development, 424-425
 - Python application development, 501-503
- local data stores, advantages of, 12-13
- LOCK TABLE statement, 304
- locking in JDBC application development, 302-305
- LOCKSIZE option, 156
- log directory, 41
- log files, location of, 44-45
- logDevice parameter, 43
- logical security, 121
- logWriter attribute, network server connections, 242
- LONG VARCHAR data type, 143
- loops. *See* control structures

M

- maintenance. *See* database administration
- manageability of relational databases, 15
- manual installation
 - of IBM Cloudscape on Linux, 90-93
 - of IBM Cloudscape on Windows, 63-66
- math functions, 192
- MAX function, 194
- migrating from previous versions of Cloudscape, 50, 76
- MIN function, 194
- mod_perl, initializing variables for, 452
- modules. *See also* extensions
 - Perl, 439-440
 - CGI.pm module, 459
 - Tk module, 467-472
 - usage, 444
 - Python, 473-474
 - cgi module, 504
 - importing, 486-487
 - Tkinter module, 512-518
 - usage, 489-490
- MONTH function, 337, 372
- multiple predicates, 173
- multiple result sets, 288
- multiple rows, inserting, 215-216
- multiple SQLExceptions, 250
 - processing, 533

N

- naming
 - databases, 39-40, 312, 529
 - derived columns, 190
 - identifiers, 138
 - variables
 - PHP, 394
 - Python, 475-476

nested table expressions, 205-206
 nested views, 166
 network server installation, verifying,
 68-69, 93-96. *See also* Apache Derby
 Network Server
 next() method, 253-254
 nextRowset() method, 420
 nextset() method, 499
 nodes, 311
 non-unique indexes, 168
 NOT clause, 177-178
 NOT EXISTS predicate, 213
 null values
 data type modifiers, 149
 and indexes, 168, 531
 in Java, 534
 for restriction, 176-177
 searching for, 531
 setting columns to, 278-279
 SQL versus Java, 260-262
 troubleshooting, 530-531
 numeric data types, 140-142
 PHP, 396
 Python, 477-478
 Nuvola icon set, 324

O

objects. *See* database objects
 OCC (occasionally connected client), 12
 ODBC (Open Database Connectivity), 308.
 See also Windows application development
 data types, versus SQL data types, 385
 Get() methods, 366
 ODBC .NET Data Provider, 357
 offline backups, 104-105
 online backups
 with built-in copy, 102-103
 with read-only access, 103-104

ON NULL INPUT clause, 224
 Open Database Connectivity. *See* ODBC
 open source
 Apache Derby as, 3
 business perspective, 10-11
 IBM's commitment to, 9-10
 technical perspective, 11
 ORDER BY clause, 187-189
 ordinary identifiers, 138
 outer joins, 183-184
 output streams, inserting/updating large
 objects, 279-281
 OUT parameters, calling stored
 procedures, 231
 Perl application development, 451-452
 PHP application development, 419-420
 Python application development, 498

P

parameter markers, 268, 384-385
 PARAMETER STYLE clause, 224, 231, 292
 parameters
 binding to prepared statements (Perl
 application development), 449
 issuing SQL statements
 PHP application development, 414-416
 Python application development,
 493-494
 Parameters() method (YMLD example
 application), 384-385
 parentheses in expressions, 532
 parent keys, 158
 parent tables, 158
 pass statement (Python), 483
 password parameter, 43
 network server connections, 241
 path names, spaces in, 60
 PATH system variable, setting, 52-55, 78-81

- PDO (PHP Data Objects), 392
 - connection pooling, 413
 - database connections, 412-413
 - disconnecting from database, 413
 - large object (LOB) data types, 424-425
 - SQL statements, calling stored procedures, 417-421
 - transaction management, 421-424
- PDO_ODBC driver, 392
- PDOStatement objects, SQL statements
 - issuing, 413-414
 - with different parameters, 414-416
 - with placeholders, 414
 - retrieving result sets, 416-417
- PEAR (PHP Extension and Archive Repository), 391
- PECL (PHP Extension Community Library), 391
- performance, 113
 - collecting statistics, 113-115
 - Perl, 441
 - PHP, 393
 - Python, 474-475
 - timing information in statistics, 115-118
- performance tuning, 526-527
- PERFORMANCE table (YMLD example database), 318-320
- Perl, 439
 - and CLI, 308
 - configuring
 - on Linux, 442-443
 - on Windows, 443-444
 - documentation, 440
 - editing environments, 441
 - installing, 441-442
 - on Linux, 442
 - on Windows, 443
 - modules, 439-440
 - usage, 444
 - performance, 441
- Perl application development
 - connection pooling, 446-447
 - database connections, 445-446
 - disconnecting from database, 446
 - GUI interfaces, 467-472
 - large object (LOB) data types, 456-458
 - module usage, 444
 - running scripts from command line, 445
 - SQL statements
 - calling stored procedures, 451-454
 - issuing, 447
 - issuing with prepared statements, 447-449
 - retrieving result sets, 450-451
 - transaction management, 454-455
 - Web interfaces
 - CGI.pm module, 459
 - conflicts with XHTML and Perl reserved, 465
 - handling GET/POST variables, 459-461
 - sessions, 466
 - taint-checking user input, 462
 - YMLD example, 462-465
- permissions
 - for backups, 103
 - database-level authorization, 133
- permutation, 172
- pFinishPurchase() method (YMLD example application), 349-351
- PHP
 - classes, creating, 404-406
 - and CLI, 308
 - configuring
 - on Linux, 409-410
 - on Windows, 410-411
 - control structures, 399
 - break statement, 404
 - continue statement, 404
 - do/while loops, 399-400
 - foreach loops, 402-403

- for loops, 401-402
- if/elseif/else structures, 400
- switch structures, 401
- while loops, 399-400
- data types
 - arrays, 396-397
 - Boolean, 395-396
 - numeric, 396
 - strings, 397-399
- delimiting code, 394
- documentation, 392
- editing environments, 393
- extensions, 391-392
- functions, defining, 404
- history of, 391
- include files, 406-407
- installing, 407
 - on Linux, 408-409
 - on Windows, 410
- performance, 393
- variables
 - declaring, 395
 - naming, 394
- PHP application development, 411
 - connection pooling, 413
 - database connections, 412-413
 - disconnecting from database, 413
 - large object (LOB) data types, 424-425
 - running scripts from command line, 411-412
 - SQL statements
 - calling stored procedures, 417-421
 - issuing, 413-414
 - issuing with different parameters, 414-416
 - issuing with placeholders, 414
 - retrieving result sets, 416-417
 - transaction management, 421-424
- Web interfaces
 - handling GET/POST variables, 426-430
 - sessions, 435-436
 - setting HTTP headers, 426
 - taint-checking user input, 430-432
 - YMLD example, 432-435
- PHP Data Objects. *See* PDO
- PHP Extension and Archive Repository (PEAR), 391
- PHP Extension Community Library (PECL), 391
- physical security, 121-123, 527
- placeholders, issuing SQL statements
 - PHP application development, 414
 - Python application development, 492
- pooling connections
 - Perl application development, 446-447
 - PHP application development, 413
 - Python application development, 491-492
- portNumber attribute, network server
 - connections, 241
- position() method, 263
- positioned deletes, 219
- positioned updates, 217
- post-installation file structure
 - IBM Cloudscape on Linux installation, 98-99
 - IBM Cloudscape on Windows installation, 72-73
- POST variables, handling
 - Perl application development, 459-461
 - PHP application development, 426-430
 - Python application development, 504-507
- pProdSelection() method (YMLD example application), 336-339
- precedence of authentication levels, 528

- precision (numbers), 141
- predicates, 173
 - BETWEEN predicate, 175
 - EXISTS predicate, 212-213
 - IN predicate, 176
 - IS NOT NULL predicate, 176-177
 - IS NULL predicate, 176-177
 - join predicates, 183
 - LIKE predicate, 174-175
 - multiple predicates, 173
 - NOT clause, 177-178
 - quantified predicates, 202-204
- prepare() method, 414, 448
- prepareCall() method, 284
- prepared statements
 - JDBC application development, 268-272
 - Perl application development, 447-449
- prerequisites
 - for IBM Cloudscape for Linux installer, 82-83
 - for IBM Cloudscape for Windows installer, 57
- previous() method, 254
- PRICEPLAN table (YMLD example database), 322-323
- primary indexes, 168
- primary keys, 158, 168. *See also* unique constraints
 - ADD UNIQUE statement, 155
 - and indexes, 168
- privileges for database objects, 163
- procedures, 48, 230-231
 - calling, 284
 - Perl application development, 451-454
 - PHP application development, 417-421
 - Python application development, 495-499
 - connections from, 535
 - creating, 281-282
 - defining, 282-284
 - error handling, 287
 - for exporting data, 109-111
 - for importing data, 106-109
 - JDBC application development, 281-288
 - result sets, 284-287
- PRODUCTIONS table (YMLD example database), 317-318
- projection, 171-172
- promotion of data types, 190
- properties
 - database-wide, 127
 - system-wide, 127
- properties variable, establishing database connections, 239
- pShowProductions() method (YMLD example application), 329-334
- pShowTickets() method (YMLD example application), 346-349
- pyDB2 module
 - large object (LOB) data types, 501-503
 - stored procedures, 496
- PyPi (Python Package Index), 473
- Python
 - classes, creating, 484-485
 - and CLI, 308
 - configuring
 - on Linux, 488
 - on Windows, 489
 - control structures, 481
 - break statement, 483
 - continue statement, 483
 - for loops, 482-483
 - if/elif/else structure, 482
 - pass statement, 483
 - while loops, 481-482

- data types
 - Boolean, 476-477
 - dictionaries, 480-481
 - numeric, 477-478
 - sequences, 478-479
 - strings, 479-480
 - documentation, 474
 - editing environments, 475
 - functions, defining, 483-484
 - history of, 473
 - installing, 487
 - on Linux, 487
 - on Windows, 488-489
 - modules, 473-474
 - importing, 486-487
 - usage, 489-490
 - performance, 474-475
 - variables
 - declaring, 476
 - naming, 475-476
 - Python application development
 - connection pooling, 491-492
 - database connections, 490
 - disconnecting from database, 491
 - GUI interfaces, 512-518
 - large object (LOB) data types, 501-503
 - module usage, 489-490
 - running scripts from command line, 490
 - SQL statements
 - calling stored procedures, 495-499
 - issuing, 492
 - issuing with different parameters, 493-494
 - issuing with placeholders, 492
 - retrieving result sets, 494-495
 - transaction management, 499-501
 - Web interfaces
 - cgi module, 504
 - handling GET/POST variables, 504-507
 - sessions, 511-512
 - setting HTTP headers, 503-504
 - taint-checking user input, 507-508
 - YMLD example, 508-511
 - Python Package Index (PyPi), 473
- Q**
- q{ } syntax, 447
 - quantified predicates, 202-204
 - queries. *See* SELECT statement
 - query() method, 416
 - question mark (?), parameter markers, 268
- R**
- range operator for restriction, 175
 - read stability isolation level, 302
 - readOnly attribute, network server
 - connections, 242
 - READS SQL DATA clause, 231
 - REAL data type, 142
 - recovery. *See* backups
 - referential integrity constraints, 156-161, 168
 - and indexes, 168
 - registerOutParameter() method, 284
 - relational databases, advantages of, 13-15
 - relative() method, 254
 - remote authentication, 528
 - renaming. *See* naming
 - repeatable read isolation level, 303
 - require statement (PHP), 407
 - reserved Perl names, conflicts with
 - XHTML, 465
 - resizing VARCHAR columns, 156

- restoreFrom parameter, 44
 - restoring database from backups, 105
 - restriction, 173
 - BETWEEN predicate, 175
 - with GROUP BY clause and column functions, 195-196
 - IN predicate, 176
 - LIKE predicate, 174-175
 - multiple predicates, 173
 - NOT clause, 177-178
 - null values, 176-177
 - range operator, 175
 - resultSetCurrency parameter
 - (createStatement() method), 244-245
 - ResultSetMetaData interface, 265
 - resultSetType parameter (createStatement() method), 244-245
 - result sets, 244
 - accessing in random order, 533
 - dynamically determining (JDBC application development), 264-267
 - exporting data, 110-111
 - modifying data with (JDBC application development), 274-279
 - multiple result sets, 288
 - retrieving
 - JDBC application development, 252-258
 - Perl application development, 450-451
 - PHP application development, 416-417
 - Python application development, 494-495
 - retrieving column values (JDBC application development), 258-264
 - stored procedures, 281, 284-287
 - retrieveMessagesFromServerOnGetMessage attribute, network server connections, 241-243
 - retrieving
 - column values (JDBC application development), 258-264
 - current performances (YMLD example application), 329-334, 364-368
 - result sets
 - JDBC application development, 252-258
 - Perl application development, 450-451
 - PHP application development, 416-417
 - Python application development, 494-495
 - seat map (YMLD example application), 335-336, 369-370
 - seat prices (YMLD example application), 334-335, 368-369
 - RETURNS clause, 223, 292, 293
 - right outer joins, 186
 - rollback() method, 422, 454, 500
 - ROLLBACK statement, 231-232
 - JDBC application development, 272
 - rollForwardRecoveryFrom parameter, 43
 - row functions. *See* scalar functions
 - row-level locks, 156
 - rowCount() method, 415
 - .rowcount property, 494
 - rows, 46
 - runtime statistics. *See* statistics
- ## S
- Saunders, Kathy, 4
 - scalability of relational databases, 15
 - scalar fullselects, 206-209
 - UPDATE statement, 218-219
 - scalar functions, 191-193, 223
 - schema names, when creating tables, 148

- schemas, 46, 221-223, 532
 - CREATE SCHEMA statement, 148
 - SET SCHEMA command, 363
- searched deletes, 219
- searched updates, 217
- SEATMAP table (YMLD example database), 323
- SEATS table (YMLD example database), 320-322
- security
 - authentication, 313
 - connection options, 43
 - encryption, 123-126
 - JCE providers, list of, 124-125
 - logical security, 121
 - permissions for backups, 103
 - physical security, 121-123
 - troubleshooting, 527-528
 - user authentication, 126-128
 - built-in authentication, 130-132
 - LDAP directory service authentication, 128-129
 - user-defined authentication, 129
 - user authorization, 132-133
- securityMechanism attribute, network server connections, 242
- seg0 directory, 41
- SELECT statement. *See also* result sets
 - Cartesian product, 179-181
 - CASE expressions, 204-205
 - correlation names, 186-187
 - derived columns, 189-191
 - DISTINCT keyword, 196-198
 - EXISTS predicate, 212-213
 - GROUP BY clause, 194-196
 - joins, 181-186
 - nested table expressions, 205-206
 - ORDER BY clause, 187-189
 - permutation, 172
 - projection, 171-172
 - quantified predicates, 202-204
 - restriction, 173-178
 - retrieving entire tables, 170-171
 - scalar fullselects, 206-209
 - subqueries, 198-202
 - UNION operator, 209-212
- Selinger, Patricia, 2
- sequences (Python), 478-479. *See also* arrays (PHP)
- sequential value columns. *See* identity columns
- server framework (deployment model), 32-36
- server name attribute, network server connections, 240-241
- service.properties file, 41
 - special characters in, 529
- session_start() method, 436
- sessions
 - Perl application development, 466
 - PHP application development, 435-436
 - Python application development, 511-512
- SET CURRENT ISOLATION statement, 303
- SET keyword, resizing VARCHAR columns, 156
- set relationships, 14
- SET SCHEMA command, 363
- setAsciiStream() method, 279
- setAttribute() method, 423
- setAutoCommit() method, 272
- setBinaryStream() method, 280
- setCalendar() method (YMLD example application), 340-343, 374-376
- setCursorName() method, 276
- setEscapeProcessing() method, 246
- setFetchDirection() method, 245
- setFetchSize() method, 245
- setMaxFieldSize() method, 246

- setMaxRows() method, 246
- setNull() method, 278
- setProduction method (YMLD example application), 372
- setTransactionIsolation() method, 303
- setUnicodeStream() method, 279
- setX() methods, 269
- shutdown parameter, 44
- single-user implementation. *See* embedded framework (deployment model)
- single quotation marks ('), character string delimiters, 143
- slices of sequences (Python), 478
- SMALLINT data type, 141
- SOME quantified predicate, 202
- sorting. *See* ORDER BY clause
- spaces in path names, 60
- SQL (Structured Query Language), 14, 135.
 - See also* dynamic SQL statements;
 - SQL statements
 - Apache Derby support for, 19-21
 - categories of, 135
 - data types
 - versus Java data types, 258, 366
 - versus ODBC data types, 385
 - DDL (Data Definition Language), 137-138
 - ALTER statement, 139
 - ALTER TABLE statement, 155-156
 - constraints, 156-163
 - CREATE statement, 139
 - CREATE TABLE statement, 147-148
 - data types, 140-147
 - data type modifiers, 148-153
 - DECLARE statement, 138-140, 153-154
 - DROP statement, 140
 - DROP TABLE statement, 156
 - indexes, 167-169
 - system catalog tables, 163-164
 - views, 165-167
 - DML (Data Manipulation Language), 170
 - DELETE statement, 219-221
 - functions, 191-194
 - INSERT statement, 213-217
 - SELECT statement, 170-191, 194-213.
 - See also* result sets
 - UPDATE statement, 217-219
 - MONTH function, 337, 372
 - null values, versus Java null values, 260-262
 - prepared statements (JDBC application development), 268-272
 - procedures, 230-231
 - schemas, 221-223
 - transactions, 231-232
 - triggers, 224-225
 - activation, 225
 - body of, 225-226
 - example, 226-230
 - troubleshooting, 529-532
 - try blocks, 533
 - UDFs (user-defined functions), 223-224
 - JDBC application development, 288-299
- SQL clause, 224, 231, 292
- SQL statements. *See also* SQL
 - calling stored procedures
 - Perl application development, 451-454
 - PHP application development, 417-421
 - Python application development, 495-499
 - issuing
 - Perl application development, 447
 - PHP application development, 413-414
 - Python application development, 492

- issuing with different parameters
 - PHP application development, 414-416
 - Python application development, 493-494
- issuing with placeholders
 - PHP application development, 414
 - Python application development, 492
- issuing with prepared statements (Perl application development), 447-449
- retrieving result sets
 - Perl application development, 450-451
 - PHP application development, 416-417
 - Python application development, 494-495
- SQL92 identifiers, 127
- SQLExceptions
 - multiple SQLExceptions, processing, 533
 - trapping (JDBC application development), 248-250
 - versus SQLWarnings, 534
 - JDBC application development, 251-252
- SQLSTATE values, 249
- SQLWarnings versus SQLExceptions, 534
 - JDBC application development, 251-252
- statement handles, 308
- statement-level isolation levels, 304-305
- statements, closing, 533. *See also* dynamic SQL statements; SQL statements
- statistics
 - collecting, 113-115
 - and execution plan, 113
 - timing information in, 115-118
- stored procedures, 48, 230-231
 - calling, 284
 - Perl application development, 451-454
 - PHP application development, 417-421
 - Python application development, 495-499
 - connections from, 535
 - creating, 281-282
 - defining, 282-284
 - error handling, 287
 - for exporting data, 109-111
 - for importing data, 106-109
 - JDBC application development, 281-288
 - result sets, 284-287
- string data types, 140-144
- string functions, 192
- strings
 - PHP, 397-399
 - Python, 479-480
 - searching for, 174-175
- Structured Query Language. *See* SQL
- subqueries, 176, 198-202
 - EXISTS predicate, 212-213
 - versus subselects, 202
- subselects
 - DELETE statement, 220-221
 - inserting, 216-217
 - versus subqueries, 202
- subset of columns, importing data, 108-109
- Subversion (SVN) repository, 8
- SUM function, 194
- support contract for IBM Cloudscape, 6
- SVN (Subversion) repository, 8
- switch structures (PHP), 401
- SYSALIASES system table, 41, 163
- SYSCHECKS system table, 41, 163
- SYSCOLUMNS system table, 41, 163
- SYSCONGLOMERATES system table, 42, 163
- SYSCONSTRAINTS system table, 42, 163
- SYSDEPENDS system table, 42, 164
- SYSFILES system table, 42, 164
- SYSFOREIGNKEYS system table, 42, 164
- SYSKEYS system table, 42, 164

SYSSCHEMAS system table, 42, 164
SYSSTATEMENTS system table, 42, 164
SYSSTATISTICS system table, 42, 164
SYSTABLES system table, 42, 164
system-level authentication, precedence, 528
system-level users
 built-in authentication, 130-131
 versus database-level users, precedence
 of, 132
system-wide properties, 127
system properties, setting, 237
system tables, list of, 41-42, 163-164
system variables
 backward slash (\) in, 64, 91
 defined, 64
 versus environment variables, 59
SYSTRIGGERS system table, 42, 164
SYSVIEWS system table, 42, 164

T

table check constraints. *See* check constraints
table scans, 113
table-level locks, 156
tables, 46
 ALTER statement, 139
 ALTER TABLE statement, 155-156
 compressing, 119
 consistency checks, 112-113
 constraints, 156
 check constraints, 161-163
 referential integrity constraints, 157-161
 unique constraints, 157
 CREATE TABLE statement, 147-148
 DECLARE statement, 138-140,
 153-154, 530
 DELETE statement, 219-221
 DROP TABLE statement, 156, 219
 exporting data, 109-110
 importing data, 107-109

indexes, 167-169
INSERT statement, 213-215
 default values, 150-151
 multiple rows, 215-216
 for specific columns, 215
 subselects, 216-217
SELECT statement
 Cartesian product, 179-181
 CASE expressions, 204-205
 correlation names, 186-187
 derived columns, 189-191
 DISTINCT keyword, 196-198
 EXISTS predicate, 212-213
 GROUP BY clause, 194-196
 joins, 181-186
 nested table expressions, 205-206
 ORDER BY clause, 187-189
 permutation, 172
 projection, 171-172
 quantified predicates, 202-204
 restriction, 173-178
 retrieving entire tables, 170-171
 scalar fullselects, 206-209
 subqueries, 198-202
 UNION operator, 209-212
system catalog tables, 41-42, 163-164
triggers, 224-225
 activation, 225
 body of, 225-226
 example, 226-230
UPDATE statement, 217-219
views, 165-167
in YMLD example database, 136-137, 316
 PERFORMANCES table, 318-320
 PRICEPLAN table, 322-323
 PRODUCTIONS table, 317-318
 SEATMAP table, 323
 SEATS table, 320-322
 TRANSACTIONS table, 323-324

- taint-checking user input
 - Perl application development, 462
 - PHP application development, 430-432
 - Python application development, 507-508
 - TCP/IP nodes, 311
 - technical perspective on open source, 11
 - temporary tables, DECLARE statement, 138-140, 153-154, 530
 - territory, connection options, 45
 - testing database connections, 312
 - ThinG GUI design editor, 324
 - Thinlet GUI toolkit, 324
 - interfaces, 330
 - time and date data types, 46, 145-147
 - TIME data type, 145-146
 - TIMESTAMP data type, 145-147
 - timing information, adding to statistics, 115-118
 - Tk (Perl module), 440
 - Perl application development, 467-472
 - Tkinter module (Python application development), 512-518
 - tmp directory, 41
 - traceFile attribute, network server connections, 242
 - traceLevel attribute, network server connections, 242
 - transactions, 231-232
 - JDBC application development, 272
 - Perl application development, 454-455
 - PHP application development, 421-424
 - Python application development, 499-501
 - safety of, 30
 - viewing transaction log (YMLD example application)
 - JDBC application, 351-355
 - Windows application, 386-390
 - TRANSACTIONS table (YMLD example database), 323-324
 - trapping SQLExceptions (JDBC application development), 248-250
 - triggers, 47-48, 224-225, 532
 - activation, 225
 - body of, 225-226
 - example, 226-230
 - troubleshooting
 - databases, 528-529
 - IBM Cloudscape on Linux installation, 96-98
 - IBM Cloudscape on Windows installation, 70-72
 - installation, 525-526
 - JDBC application development, 532-535
 - performance tuning, 526-527
 - security issues, 527-528
 - SQL, 529-532
 - truncated data, preventing in ij tool, 114
 - try/catch blocks, 533
 - YMLD example application, 367-368
 - tuples (Python), 478
- ## U
- UDFs (user-defined functions), 191, 223-224
 - JDBC application development, 288-299
 - troubleshooting, 535
 - uncataloged database connections, 313
 - uncommitted read isolation level, 302
 - uncorrelated subqueries, 201
 - Unicode character encoding, 142-143, 262
 - UNION ALL operator, 210
 - UNION operator, 209-212
 - unique constraints, 156-157, 168
 - unique identifiers. *See* identity columns
 - unique indexes, 167-168

unique keys, 157, 168
 and indexes, 168
unit of work. *See* transactions
UPDATE rules (referential constraints), 161
UPDATE statement, 217-219
updateNull() method, 278
use statement (Perl), 444
user attribute, network server connections, 241
user authentication. *See* authentication
user authorization. *See* authorization
user input, taint-checking
 Perl application development, 462
 PHP application development, 430-432
 Python application development, 507-508
user names. *See* authorization identifiers
user parameter, 43
user-defined authentication, 129
user-defined functions. *See* UDFs

V

validation. *See* taint-checking user input
VALUES clause, 214
VARCHAR data type, 140, 143
 resizing VARCHAR columns, 156
variables
 initializing for mod_perl, 452
 PHP
 declaring, 395
 naming, 394
 Python
 declaring, 476
 naming, 475-476
vector functions. *See* column functions
verifying
 IBM Cloudscape on Linux installation,
 93-96
 IBM Cloudscape on Windows installation,
 67-69

version of JRE, checking, 52, 78
views, 47, 165-167
 versus nested table expressions, 206
virtual tables. *See* views

W

warnings. *See* SQLWarnings
wasnull() method, 260
Web interfaces
 cgi module (Python application
 development), 504
 CGI.pm module (Perl application
 development), 459
 handling GET/POST variables
 Perl application development, 459-461
 PHP application development, 426-430
 Python application development,
 504-507
 sessions
 Perl application development, 466
 PHP application development, 435-436
 Python application development,
 511-512
 setting HTTP headers
 PHP application development, 426
 Python application development,
 503-504
 taint-checking user input
 Perl application development, 462
 PHP application development, 430-432
 Python application development,
 507-508
XHTML conflicts with Perl reserved
 names, 465
YMLD example
 Perl application development, 462-465
 PHP application development, 432-435
 Python application development,
 508-511

- WHERE clause, 173
 - while loops
 - PHP, 399-400
 - Python, 481-482
 - Windows
 - configuring
 - Perl, 443-444
 - PHP, 410-411
 - Python, 489
 - environment variables, setting, 526
 - installing
 - DB2 Runtime Client, 310-311
 - Perl, 443
 - PHP, 410
 - Python, 488-489
 - installing Apache Derby/IBM Cloudscape, 49-50
 - with installation program, 55-63
 - manual installation, 63-66
 - migrating from previous versions, 50
 - post-installation file structure, 72-73
 - preparations for, 50-55
 - troubleshooting installation, 70-72
 - verifying installation, 67-69
 - running
 - Perl scripts, 445
 - PHP scripts, 411
 - Python scripts, 490
 - Windows application development, YMLD
 - example application
 - components, 357
 - database connections, 360-364
 - exit routine, 390
 - functions of, 358
 - initializing, 360
 - installing, 359-360
 - purchasing tickets, 370-386
 - retrieving
 - current performances, 364-368
 - seat map, 369-370
 - seat prices, 368-369
 - viewing transaction log, 386-390
- X-Z**
- XHTML, conflicts with Perl reserved names, 465
 - XML files, 13
 - YMLD example database
 - as JDBC application
 - database connections, 327-329
 - exit routine, 355
 - functions of, 325
 - initializing, 326-327
 - installing, 326
 - Java components, 324-325
 - purchasing tickets, 336-351
 - retrieving current performances, 329-334
 - retrieving seat map, 335-336
 - retrieving seat prices, 334-335
 - viewing transaction log, 351-355
 - tables in, 136-137, 316
 - PERFORMANCES table, 318-320
 - PRICEPLAN table, 322-323
 - PRODUCTIONS table, 317-318
 - SEATMAP table, 323
 - SEATS table, 320-322
 - TRANSACTIONS table, 323-324
 - Web interfaces
 - Perl application development, 462-465
 - PHP application development, 432-435
 - Python application development, 508-511

- as Windows application
 - components, 357
 - database connections, 360-364
 - exit routine, 390
 - functions of, 358
 - initializing, 360
 - installing, 359-360
 - purchasing tickets, 370-386
 - retrieving current performances,
364-368
 - retrieving seat map, 369-370
 - retrieving seat prices, 368-369
 - viewing transaction log, 386-390
- Your Momma Loves Drama. *See* YMLD
 - example database