



Index

NOTE: If a letter follows a page number:

T = Tutorial Section

tb = Table

A

aborting

See also errors; exceptions
transactions, with rollback method
(Connection interface) 76T

absolute method (ResultSet interface)

argument, positive or negative 120T

absolute positioning

definition, glossary 1161
effect of deleting rows on 718
example of 710

Abstract Window Toolkit

See java.awt package

acceptsURL method (Driver interface) 606

access

See also retrieving
conflicts,
 avoiding with locks 75T
 transaction isolation level as
 mechanism for managing 393

data,

 dynamic 1084
 in a relational table 59T
dirty read avoidance 75T
restrictions, monitored by bytecode
 verifier 36
rows, with next method 65T
specifiers, protected 30
transaction isolation level value 75T
two-tier model compared with three-tier
 model 18
warnings, handling 84T

access error

 definition, glossary 1161
 defintion in context 7

active connection instance

 required to create Statement objects 57T

addBatch method (Statement interface)

 962, 973

addConnectionEventListener method (PooledConnection interface) 644

adding

 rows, INSERT use for 43

addRowSetListener method (RowSet interface) 825

ADO API

 JDBC comparison with 18

after method (Timestamp class) 1027

afterLast method (ResultSet interface)

 method explanation 694
 use with ResultSet.previous method
 119T

allProceduresAreCallable method (DatabaseMetaData interface) 466

allTablesAreSelectable method (DatabaseMetaData interface) 467

ALTER TABLE

 as DDL command, purpose 44

ANSI SQL92 Entry Level

 as JDBC Compliant requirement 21

API (Application Programming Interface)

 definition, glossary 1161
 high-level, planned for JDBC 15
 JDBC as Java, for executing SQL
 statements 13

applets

 See also URL (Uniform Resource
 Locator)s

 appletviewer use 102T
 creating, from an application 99T
 definition, glossary 1161
 displaying the content of 101T
 finalize methods, suggested for driver
 implementations 1114

 java.applet package handling of 35

 security in 941

**applications**

- JDBC, creating 79T
- standalone, applets differences 101T

ARCHIVE parameter

- use in applets to specify JAR file 106T

arguments

- method, JDBC types as 1033
- search patterns in, in DatabaseMetaData methods 451

ARRAY (SQL99 type)

- definition 283
- materializing on the client 284

ARRAY field (Types class)

- class definition 1036

ARRAY field (Types class)

- description 1076
- mapping to Java object type 1089tb
- mapping to Java type 1087tb

array in Java programming language

- storing in database 292

Array interface

- getArray method 293– 295
- getBaseType method 296
- getBaseTypeName method 296
- getResultSet method 297– 299

Array object

- as standard mapping of ARRAY 283
- base type of 284
- creating 284
- duration of 284
- index of first element 289
- LOCATOR(ARRAY), implemented using 283
- mapping by setObject method to JDBC types 1091tb
- mapping from Java object types to JDBC types 1090tb
- retrieving 283
- retrieving values from ResultSet objects 290
- standard mapping to JDBC types 1088tb
- storing 292
- using to materialize ARRAY data on the client 143T

attribute

- definition, glossary 1161

attribute fields

- attributeNotNulls field
(DatabaseMetaData interface) 551

attributeNullable field

- (DatabaseMetaData interface) 552

attributeNullableUnknown field

- (DatabaseMetaData interface) 552

attributes of SQL structured type

- See SQL structured type

auto-commit mode

- as default connection mode 392
- definition, term in context 73T
- disabled by default, conditions for 393
- disabled for XAConnection object 1040
- disabling 393
- enabling after batch update executed 136T
- impact on statement completion 954

AutoGenKeys.java

- example code using an automatically generated key 190T

automatic memory management

- See garbage collection

automatically generated keys

- definition in context 189T
- description 954
- determining whether supported 189T
- retrieving 190T, 956
- signalling the driver to make all keys available 956
- signalling the driver to make particular columns retrievable 957
- signalling the driver to make retrievable 189T, 954

B

backward compatibility of Java 2 platform 5, 1122, 1132**bandwidth**

- low, Java advantages for 37

base type

- definition, glossary 1161
- of an Array object 284

batch update

- definition, glossary 1161

batch updates

- BatchUpdateException, when thrown 137T, 963
- code example with SQL99 types 158T
- commands, list of 133T, 962



disabling auto-commit mode 134T
executed by
 CallableStatement objects 325
 PreparedStatement objects 654
 Statement objects 961
order of execution of commands 962
parameterized, using setter methods
 (PreparedStatement interface)
 136T
restrictions on, for CallableStatement
 objects 325
use of commit method (Connection
 interface) to commit updates 136T
use of executeBatch method (Statement
 interface) compared to
 executeUpdate method (Statement
 interface) 135T
using addBatch method (Statement
 interface) 133T, 135T
using clearBatch method (Statement
 interface) 134T
using executeBatch method (Statement
 interface) 134T, 135T

BatchUpdate.java
example code for a batch update 138T

BatchUpdateException class
BatchUpdateException constructors
 303–306
getUpdateCounts method 307

BatchUpdateException object
information in 137T, 301
retrieving information in 137T, 302
thrown by Statement.executeBatch
 method 301
when thrown 963

before method (Timestamp class) 1028

beforeFirst method (ResultSet interface)
 693

best row fields
bestRowNotPseudo field
 (DatabaseMetaData interface) 552
bestRowPseudo field
 (DatabaseMetaData interface) 552
bestRowSession field
 (DatabaseMetaData interface) 553
bestRowTemporary field
 (DatabaseMetaData interface) 553

bestRowTransaction field
 (DatabaseMetaData interface) 553

bestRowUnknown field
 (DatabaseMetaData interface) 553

big-endian data
definition, term in context 721

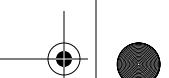
BIGINT field (Types class)
See also types
characteristics and comparison with
 standard SQL and Java types 1071
class definition 1036
description 1071
mapping to,
 database-specific SQL types 1094tb
 Java object type 1089tb
 Java type 1087tb

binary
data,
 retrieving very large amounts of data
 using getBinaryStream
 method (ResultSet interface)
 721
 sending very large amounts of data
 using setBinaryStream
 method (PreparedStatement
 interface) 653

BINARY field (Types class)
See also types
characteristics and comparison with
 standard SQL and Java types 1069
class definition 1036
description 1069
getInt method not usable for retrieving
 63T
mapping to,
 Java object type 1089tb
 Java type 1087tb

BIT field (Types class)
See also types
characteristics and comparison with
 standard SQL and Java types 1070
class definition 1036
description 1070
mapping to,
 database-specific SQL types 1094tb

BLOB (SQL99 type Binary Large Object)
definition, glossary 1162



- materializing on client as an array of bytes 311
- materializing on client as an input stream 310
- BLOB field (Types class)**
 - class definition 1036
 - description 1075
 - mapping to Java object type 1089tb
 - mapping to Java type 1087tb
- Blob interface**
 - getBinaryStream method 314
 - getBytes method 314
 - length method 315
 - position method 315, 316
 - setBinaryStream method 317
 - setBytes method 317, 318
 - truncate method 319
- Blob object**
 - as mapping of SQL BLOB 309
 - creating 310
 - duration of 309
 - finding patterns in 312
 - mapping by setObject method to JDBC types 1091tb
 - mapping from Java object types to JDBC types 1090tb
 - materializing on client 310
 - modifying with method setBytes 312
 - retrieving 310
 - standard mapping to JDBC types 1088tb
 - storing 311
- books**
 - contents 8
 - references, Java 80T
- Boolean class**
 - See also types
 - conversion by setObject to JDBC types 1091tb
 - mapping to, JDBC types 1090tb
- BOOLEAN field (Types class)**
 - class definition 1037
 - description 1078
 - mapping to Java object type 1089tb
 - mapping to Java type 1087tb
- boolean Java type**
 - DatabaseMetaData methods that return 212
 - mapping to JDBC type 1088tb
- byte array**
 - mapping by setObject method to JDBC types 1091tb
 - mapping to JDBC types 1090tb
 - standard mapping to JDBC types 1088tb
- byte Java type**
 - mapping to JDBC type 1088tb
- bytecode(s)**
 - compilation into, as source of Java portability 28
 - definition, glossary 1162
 - verification of, as Java security feature 36

C

- C language**
 - disadvantages of 17
- C++ language**
 - as foundation for Java 28
- CachedRowSet class**
 - as example implementation of RowSet interface 256T
 - creating instances 812
 - default type 815
 - methods defined in 812
 - modifying rows in 815
 - populating instances with data 813– 814
 - readers and writers, customizing in 815, 816
 - uses for 810
- calculated columns**
 - access and use 452
- Calendar object**
 - using time zone in 593
 - using to construct a date 589
 - using to construct a time 1011– 1013
 - using to construct a timestamp 1021– 1023
- call keyword 960**
- ?=call keyword 960**
- Call Level Interface**
 - See also Open Group X/Open SQL CLI, basis for JDBC and ODBC 18
- CallableStatement interface definition**
 - See also escape syntax, statements



as one of three classes for sending SQL statements to a database 391
getArray method 336
getBigDecimal method 337
getBigDecimal method (deprecated) 336
getBlob method 338
getBoolean method 338
getByte method 339
getBytes method 339
getClob method 340
getDate method 341
getDouble method 342
getFloat method 343
getInt method 344
getLong method 344
getObject method 345, 346
getRef method 347
getShort method 348
getString method 349
getTime method 349, 350
getTimestamp method 351
getURL method 352
registerOutParameter method 353, 354, 355
setAsciiStream method 356
setBigDecimal method 357
setBinaryStream method 357
setBoolean method 358
setByte method 358
setBytes method 359
setCharacterStream method 359
setDate method 360
setDouble method 361
setFloat method 361
setInt method 362
setLogin method 362
setNull method 363
setObject method 364, 365, 366
setShort method 367
setString method 367
setTime method 368
setTimestamp method 369
setURL method 370
warnings reported on 85T
wasNull method 370

CallableStatement objects
comparison with **Statement** and **PreparedStatement** objects 953

creating 322
finding out whether stored procedures are supported 322
getting information about parameters 331
holdable result sets, produced by 323
restrictions on, for executing batch updates 325
retrieving OUT parameter values 326
retrieving SQL OUT parameters as Java types 1065
scrollable result sets, produced by 323
updatable result sets, produced by 323

calling
stored procedures 78T

cancel method (Statement** interface)** 973

canceling an update
using **cancelRowUpdates method** (**ResultSet** interface) 124T

cancelRowUpdates method (ResultSet** interface)** 710

case sensitivity
DatabaseMetaData methods that test for 452
identifier name patterns 451
in SQL statements 56T

catalog
definition, glossary 1162

catching exceptions
See also exceptions
all possible, methods for 82T
catch block,
exception handling with 81T
printing exceptions with 81T
SQLException exception handling 76T

chaining
exceptions, through **SQLException** objects 908
warnings,
through **DataTruncation** objects 582
through **SQLWarning** objects 946

CHAR field (Types** class)**
See also **types**
characteristics and comparison with standard SQL and Java types 1067

class definition 1036
description 1067
getString method use for retrieving 63T



mapping to,
database-specific SQL types 1094tb
Java object type 1089tb
Java type 1087tb

choices field (*DriverPropertyInfo* class)
625

class(es)
See also interfaces; methods
Class.forName method, loading drivers
with 604, 612
definition,
glossary 1162
putting code into 80T
term in context 29
importing 80T
interfaces compared with 34
library,
as Java platform component 28
Java platform 34
loaders,
DriverManager tracking of driver
relationships with 612
security features built into 36
methods, See static

ClassNotFoundException class
handling 81T

clearBatch method (*Statement* interface)
973

clearing
clearParameters method,
PreparedStatement interface 658
RowSet interface 826
clearWarnings method,
Connection interface 401
ResultSet interface 733
Statement interface 974
parameter values 649

client/server configuration
definition, term in context 19

CLOB (SQL99 Character Large Object)
definition, glossary 1162
materializing
as a *String* object 374
as a Unicode stream 374
as an Ascii stream 374

CLOB field (*Types* class)
class definition 1037
description 1075

mapping to Java object type 1089tb
mapping to Java type 1087tb

Clob interface
getAsciiStream method 378
getCharacterStream method 378
getSubString method 379
length method 379
position method 380
setAsciiStream method 381
setCharacterStream method 382
setString method 382, 383
truncate method 383

Clob object
accessing 373
as mapping of a SQL CLOB value 373
creating 374
duration 373
mapping by *setObject* method to JDBC
types 1091tb
mapping from Java object types to JDBC
types 1090tb
materializing on the client 374
standard mapping to JDBC types 1088tb
storing 375
updating 375
adding an entire *String* object 376
adding Ascii bytes 376
adding part of a *String* object 376
adding Unicode characters 376

CLOSE_ALL_RESULTS field (*Statement* interface) **995**

CLOSE_CURRENT_RESULT field (*Statement* interface) **995**

CLOSE_CURSORS_AT_COMMIT field (*ResultSet* interface) **779**

closing
close method,
Connection interface 401
Connection interface, recommended
for external resources 395
PooledConnection interface 644
ResultSet interface 733
ResultSet interface, closing result
sets with 723
Statement interface 974
explicit, recommended for external
resources 395
result sets 723



Statement objects 974

code examples
See sample code

COFFEEBREAK database 51T
identifying 62T
number updates, determining 70T

columns
accessing information about 199T
calculated, access and use 452
`columnNoNulls` field
(`DatabaseMetaData` interface) 554
`columnNoNulls` field
(`ResultSetMetaData` interface) 795
`columnNullable` field
(`DatabaseMetaData` interface) 554
`columnNullable` field
(`ResultSetMetaData` interface) 796
`columnNullableUnknown` field
(`DatabaseMetaData` interface) 554
maximum name length 212T
names, handling result set duplicates 697
number,
accessing with `findColumn` method
(`ResultSet` interface) 697
values, retrieving from a result set 696

pseudo,
`DatabaseMetaData` constants that indicate 452
definition, term in context 452

comma
column entry separation with 55T

command string
property on a `RowSet` object 803

commands
SQL, categories of 43

commit
`commit` method (`Connection` interface) 394, 402
committing transactions with 73T
managing data integrity with 45
transactions terminated by 392
definition,
glossary 1162
term in context 45, 73T

of transactions,
disabling auto-commit mode 73T

comparing
numbers, in `SELECT` statements 41
strings, in `SELECT` statements 40

compilation
of SQL statements, `PreparedStatement` interface use 647

completion
statement 954

concatenating
strings 56T

CONCUR_READ_ONLY field (`ResultSet` interface) 701

CONCUR_UPDATABLE field (`ResultSet` interface) 702

concurrency
definition, glossary 1162
determining with
`supportsResultSetConcurrency` method (`DatabaseMetaData` interface) 225T
handling, in data transactions 44
levels of
optimistic concurrency 812, 817
pessimistic concurrency 811
setting for a `RowSet` object 803

conditions
`WHERE` clause use 41

conflicts
access, transaction isolation level as mechanism for managing 393
avoiding with locks 75T
between a `RowSet` object and its data source 811

conformance
database type,
issues 21
JDBC handling of 21

connect method (`Driver` interface) 606

connected
definition, glossary 1162
for `RowSet` object, differences from disconnected 799

connection
definition, glossary 1162

Connection interface definition 399
`clearWarnings` method 401
`close` method 401



commit method 402, 1039
createStatement method 402, 403, 404
 creating Statement objects 57T,
 952
 Statement objects created by 390
fields 428
getAutoCommit method 405
getCatalog method 405
getHoldability method 406
getMetaData method 407
 obtaining information about a
 database with 385
getTransactionIsolation method 407
 determining transaction isolation
 level with 76T
getTypeMap method 408
getWarnings method 408
isClosed method 409
isReadOnly method 409
nativeSQL method 410
prepareCall method 322, 410, 411, 413
 CallableStatement objects created
 by 391
PreparedStatement use with 67T
prepareStatement method 414, 415,
 416, 417, 418, 420
 PreparedStatement objects created
 by 390
releaseSavepoint method 421
rollback method 421, 422, 1039
 calling, conditions which mandate
 76T
 undoing table changes 75T
setAutoCommit method 422, 1040
setCatalog method 423
setHoldability method 424
setReadOnly method 424
setSavepoint method 425
setTransactionIsolation method 426
 setting transaction isolation level with
 76T
setTypeMap method 427
transaction isolation levels values
 provided by 75T
TRANSACTION_NONE field 428
TRANSACTION_READ_COMMITTED field 428
TRANSACTION_READ_UNCOMMITTED field
 428

TRANSACTION_REPEATABLE_READ field
 429
TRANSACTION_SERIALIZABLE field 429
warnings reported on 85T

Connection object
closing 639
creating savepoints with 394
creating statements with 390
creating with
 a ConnectionPoolDataSource
 object 439
 a DataSource object 386
 an XADataSource object 1056
 the DriverManager class 385
default mode 392
use in transactions 392
use of commit method 392
use of rollback method 392
using to send SQL statements 390

connection pooling
definition
 glossary 1162
 in context 168T, 637, 640
deployment for 173T
effect on application code 176T, 637
implementation of getConnection
 method (DataSource interface), for
 440
listener, module as 435
manager 642
obtaining and using 639
sequence of events 640
summary description 1151

connectionClosed method
(ConnectionEventListener interface)
435, 437

connectionErrorOccurred method
(ConnectionEventListener interface)
435, 437

ConnectionEvent class
constructor 432, 433
getSQLException method 434

ConnectionEvent constructor
(ConnectionEvent class) 432, 433

ConnectionEventListener interface
connectionClosed method 437
connectionErrorOccurred method 437



ConnectionEventListener object
registering with PooledConnection
object 436

ConnectionPoolDataSource interface
getLoginTimeout method 444
getLogWriter method 445
getPooledConnection method 445, 446
setLoginTimeout method 446
setLogWriter method 447

ConnectionPoolDataSource object
as factory for PooledConnection objects
439
similarity to DataSource object 439

ConnectionPoolingBean.java
example code using a pooled connection
176T

connections
between driver and database,
direct establishment with Driver
methods 603
connect method (Driver interface)
DriverManager use for driver testing
613
establishing a connection with 385
establishing with a DataSource object
570
establishing with DriverManager 613
making, to a DBMS 52T

consistency
data,
balancing against performance need
394
maintaining 44
truncation on write issues 582

constants
See also fields; literals; variables
Connection fields as 428
DatabaseMetaData
as possible values for columns in
ResultSet return value 449
that indicate pseudo columns 452
interface fields required to be 34
JDBC types, using 1033
ResultSet fields as 778
ResultSetMetaData fields as 783
Types fields as 1035

constructor
See also methods

definition, term in context 30

contents
organization of 8
summary of, by chapter 8

conversion functions
support for, as implementation
requirement 1107

conversions
by setObject method 651

correct code
Java features that support 33

CreateNewTable.java
as Sample Code 18 233T
translating non-standard names for SQL
data types 87T, 91T

CreateNewType.java
code for creating a new SQL99 UDT
246T
explanation of sample code 19 244T

CreateRef.java
as example code for creating an SQL REF
value 151T

**createStatement method (Connection
interface)** 402, 403, 404

CreateStores.java
as code example for inserting SQL99 types
in a batch update 154T
example code for using SQL99 types
154T

CreateUDTs.java
example code for creating user-defined
types 147T

creating
applets, from an application 99T
CallableStatement objects 322
prepareCall method use for 391
class definitions 80T
CREATE TABLE statement
as DDL command, purpose 43
executeUpdate method (Statement
interface) use 953
JDBC type use in 1066

CreateCoffees.java,
as Sample Code 1 87T
source code for 88T

CreateNewTable.java
as Sample Code 18 236T
explication of 233T



CreateNewType.java
as Sample Code 19 for creating a
 UDT 246T
explication of 244T
createStatement method (*Connection*
 interface) 402
 creating Statement objects 390, 952
CreateSuppliers.java,
 as Sample Code 3 91T
 as source code for 92T
Date objects 590
JDBC applications 79T
JDBC statements 57T
joins, sample code for 94T
PreparedStatement objects 67T
 prepareStatement method use for
 390
relational tables 54T
 executeUpdate method use for 57T
ResultSetMetaData objects 783
Statement objects 952
 createStatement method use for
 390
stored procedures 77T
tables 71T
 CREATE TABLE use for 43
 sample code for 87T, 91T
Time objects 1011
Timestamp objects 1021
transactions 73T
curly braces {{}}
as Statement object escape syntax 958T
as stored procedure escape syntax 78T
Java syntax use 29
current row
characteristics of 693
moving to, from insert row 127T
cursor
definition,
 glossary 1163
 term in context 44
duration of 693
movement
 effect of, if done before calling
 insertRow method (*ResultSet*
 interface) 127T
 forward and backward 118T
 in *ResultSet* objects 693
in *RowSet* objects 804
method implementation for *RowSet*
 objects 804
to a specific row 127T
trigger
 for *RowSet* event notification
 804
 for *RowSetListener* methods
 866
name, retrieving, with *getCursorName*
 method (*ResultSet* interface) 713
next method (*ResultSet* interface),
 movement of 61T
position
 result set tracking of 713
result set, definition, term in context 692
cursorMoved method (*RowSetListener*
 interface) 867, 868
curved arrow
See also notation
identifying output from JDBC code with
 30, 62T
custom mapping
creating a *SQLData* object 162T
creation
 putting entry in a type map 899
 writing a class that implements
 SQLData interface 897–899
DISTINCT type, of 601
examples in tutorial 162T–167T
getObject method, use in 164T
java.util.Map object, used for 395
of ARRAY elements being materialized on
 the client 398
of class defined in Java programming
 language 652
PreparedStatement.setObject, use in
 652
reason for with SQL structured type 998
retrieving a DISTINCT type 601
SQL structured type, of, compared to
 standard mapping 165T
SQLData implementation needed for 895
storing a UDT with 902
type map entry for 164T
type map, using new one instead of one
 associated with connection 166T
what happens behind the scenes 900–902

**customizing**

sample code files 85T

D**d keyword 960****DAO API**

JDBC comparison with 18

data

accessing,

dynamically 1084

in a relational table 59T

most recent with `refreshRow` method
(`ResultSet` interface) 132T

binary,

retrieving very large amounts of data
using `getBinaryStream`
method (`ResultSet` interface)
721

sending very large amounts of data
using `setBinaryStream`
method (`PreparedStatement`
interface) 653

concurrency, definition, term in context
45

consistency,

definition, term in context 45

truncation on write issues 582

displaying, SELECT use for 43

entering, into a relational table 58T

input, truncation exception, as
implementation requirement for
drivers 1109

integrity, transaction use to preserve 75T

large quantities, input streams use for
sending 653

members, See fields

modified in `RowSet` object, writing back to
data source 885

reading, handling problems with 85T

truncation, handling 582

types,

See also types

as SQL conformance issue 20

non-standard names, translating, with
`CreateNewTable.java` 87T

writing, handling problems with 85T

data definition

See also DDL (Data Definition Language)

`dataDefinitionCausesTransaction-`
`Commit` method (`DatabaseMetaData`
interface) 467

`dataDefinitionIgnoredInTransac-`
`tions` method (`DatabaseMetaData`
interface) 467

data source

definition

glossary 1163

in context 565

database(s)

accessing information about 449–563

connecting to 385–429

definition, glossary 1163

identifying, in JDBC URL 388

obtaining information about, with
`Connection.getMetaData` method
385

opening connections to 385

setting up 51T

setting up values for input to,
`PreparedStatement` methods for
647

system, definition, glossary 1163

DatabaseMetaData interface 458

`allProceduresAreCallable` method
466

`allTablesAreSelectable` method 467

`attributeNoNulls` field 551

`attributeNullable` field 552

`attributeNullableUnknown` field 552

`bestRowNotPseudo` field 552

`bestRowPseudo` field 552

`bestRowSession` field 553

`bestRowTemporary` field 553

`bestRowTransaction` field 553

`bestRowUnknown` field 553

`columnNoNulls` field 554

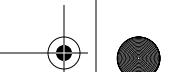
`columnNullable` field 554

`columnNullableUnknown` field 554

`dataDefinitionCausesTransaction-`
`Commit` method 467

`dataDefinitionIgnoredInTransac-`
`tions` method 467

`deletesAreDetected` method 468



doesMaxRowSizeIncludeBlobs method
 468
getAttributes method 469
getBestRowIdentifier method 455,
 471
getCatalogs method 472
getCatalogSeparator method 473
getCatalogTerm method 473
getColumnPrivileges method 473
getColumns method 454, 475
getConnection method 477
getCrossReference method 218T, 477
getDatabaseMajorVersion method 480
getDatabaseMinorVersion method 480
getDatabaseProductName method 481
getDatabaseProductVersion method
 481
getDefaultValueIsolation
 method 481
getDriverMajorVersion method 482
getDriverMinorVersion method 482
getDriverName method 482
getDriverVersion method 483
getExportedKeys method 218T, 483
getExtraNameCharacters method 485
getIdentifierQuoteString method
 486
getImportedKeys method 218T, 222T,
 486
getIndexInfo method 488
getJDBCMajorVersion method 490
getJDBCMajorVersion method 491
getMaxBinaryLiteralLength method
 491
getMaxCatalogNameLength method 491
getMaxCharLiteralLength method 492
getMaxColumnNameLength method
 212T, 492
getMaxColumnsInGroupBy method 492
getMaxColumnsInIndex method 493
getMaxColumnsInOrderBy method 493
getMaxColumnsInSelect method 493
getMaxColumnsInTable method 494
getMaxConnections method 494
getMaxCursorNameLength method 494
getMaxIndexLength method 495
getMaxProcedureNameLength method
 495
getMaxRowSize method 495
getMaxSchemaNameLength method 496
getMaxStatementLength method 496
getMaxStatements method 496
getMaxTableNameLength method 497
getMaxTablesInSelect method 497
getMaxUserNameLength method 497
getNumericFunctions method 498
getPrimaryKeys method 218T, 498
getProcedureColumns method 455, 499
getProcedures method 501
getProcedureTerm method 502
getResultSetHoldability method 503
getSchemas method 503
getSchemaTerm method 504
getSearchStringEscape method 209T,
 504
getSQLKeywords method 210T, 504
getSQLStateType method 505
getStringFunctions method 211T, 505
getSuperTables method 505
getSuperTypes method 506
getSystemFunctions method 508
getTablePrivileges method 508
getTables method 509
getTableTypes method 213T, 511
getTimeDateFunctions method 512
getTypeInfo method 214T, 454, 512
getUDTs method 454, 455, 514
getURL method 516
getUserName method 516
getVersionColumns method 516
importedKeyCascade field 554
importedKeyInitiallyDeferred field
 555
importedKeyInitiallyImmediate field
 555
importedKeyNoAction field 555
importedKeyNotDeferrable field 555
importedKeyRestrict field 556
importedKeySetDefault field 556
importedKeySetNull field 556
insertsAreDetected method 517
isCatalogAtStart method 518
isReadOnly method 519
locatorsUpdateCopy method 519
nullPlusNonNullIsNull method 519
nullsAreSortedAtEnd method 520



nullsAreSortedAtStart method 520
 nullsAreSortedHigh method 520
 nullsAreSortedLow method 521
 othersDeletesAreVisible method 521
 othersInsertsAreVisible method 522
 othersUpdatesAreVisible method 522
 ownDeletesAreVisible method 523
 ownInsertsAreVisible method 524
 ownUpdatesAreVisible method 524
 procedureColumnIn field 556
 procedureColumnInOut field 557
 procedureColumnOut field 557
 procedureColumnResult field 557
 procedureColumnReturn field 557
 procedureColumnUnknown field 558
 procedureNoNulls field 558
 procedureNoResult field 558
 procedureNullable field 558
 procedureNullableUnknown field 559
 procedureResultUnknown field 559
 procedureReturnsResult field 559
 sqlStateSQL99 field 559
 sqlStateXOpen field 559
 storesLowerCaseIdentifiers method
 525
 storesLowerCaseQuotedIdentifiers
 method 525
 storesMixedCaseIdentifiers method
 525
 storesMixedCaseQuotedIdentifiers
 method 526
 storesUpperCaseIdentifiers method
 526
 storesUpperCaseQuotedIdentifiers
 method 526
 supportsAlterTableWithAddColumn
 method 527
 supportsAlterTableWithDropColumn
 method 527
 supportsANSI92EntryLevelSQL method
 527
 supportsANSI92FullSQL method 528
 supportsANSI92IntermediateSQL
 method 528
 supportsBatchUpdates method 528
 supportsCatalogsInDataManipu-
 lation method 529

 supportsCatalogsInIndexDefini-
 tions method 529
 supportsCatalogsInPrivilegeDefi-
 nitions method 529
 supportsCatalogsInProcedureCalls
 method 530
 supportsCatalogsInTableDefini-
 tions method 530
 supportsColumnAliasing method 530
 supportsConvert method 531
 supportsCoreSQLGrammar method 532
 supportsCorrelatedSubqueries
 method 532
 supportsDataDefinitionAndData-
 ManipulationTransactions
 method 532
 supportsDataManipulationTransac-
 tionsOnly method 533
 supportsDifferentTableCorrela-
 tionNames method 533
 supportsExpressionsInOrderBy
 method 533
 supportsExtendedSQLGrammar method
 534
 supportsFullOuterJoins method 534
 supportsGetGeneratedKeys method
 534
 supportsGroupBy method 534
 supportsGroupByBeyondSelect method
 535
 supportsGroupByUnrelated method
 535
 supportsIntegrityEnhancementFa-
 cility method 535
 supportsLikeEscapeClause method
 536
 supportsLimitedOuterJoins method
 536
 supportsMinimumSQLGrammar method
 536
 supportsMixedCaseIdentifiers
 method 537
 supportsMixedCaseQuotedIdentifi-
 fers method 537
 supportsMultipleOpenResults method
 537
 supportsMultipleResultSets method
 538



supportsMultipleTransactions
 method 538
supportsNamedParameters method 538
supportsNonNullableColumns method
 539
supportsOpenCursorsAcrossCommit
 method 539
supportsOpenCursorsAcrossRollback
 method 539
supportsOpenStatementsAcrossCom-
 mit method 540
supportsOpenStatementsAcrossRoll-
 back method 540
supportsOrderByUnrelated method
 541
supportsOuterJoins method 541
supportsPositionedDelete method
 541
supportsPositionedUpdate method
 542
supportsResultSetConcurrency
 method 542
supportsResultSetHoldability
 method 543
supportsResultSetType method 543
supportsSavepoints method 544
supportsSchemasInDataManipulation
 method 544
supportsSchemasInIndexDefinitions
 method 544
supportsSchemasInPrivilegeDefini-
 tions method 545
supportsSchemasInProcedureCalls
 method 545
supportsSchemasInTableDefinitions
 method 545
supportsSelectForUpdate method 546
supportsStatementPooling method
 546
supportsStoredProcedures method
 546
supportsSubqueriesInComparisons
 method 547
supportsSubqueriesInExists method
 547
supportsSubqueriesInIns method 547
supportsSubqueriesInQuantifieds
 method 548

supportsTableCorrelationNames
 method 548
supportsTransactionIsolationLevel
 method 548
supportsTransactions method 549
supportsUnion method 549
supportsUnionAll method 550
tableIndexClustered field 560
tableIndexHashed field 560
tableIndexOther field 560
tableIndexStatistic field 560
typeNoNulls field 561
typeNullable field 561
typeNullableUnknown field 561
typePredBasic field 561
typePredChar field 562
typePredNone field 562
typeSearchable field 562
updatesAreDetected method 550
usesLocalFilePerTable method 551
usesLocalFiles method 551
using to find out about support for stored
 procedures 322
versionColumnNotPseudo field 562
versionColumnPseudo field 563
versionColumnUnknown field 563
**databaseName property (DataSource
 interface) 567**
DATALINK data type
 getURL method for retrieving 753
 mapping to `java.net.URL` object 1127
 methods for retrieving 698
 methods for retrieving, setting, reading,
 and writing 1127
DATALINK field (Types class)
 as a return value for some
 DatabaseMetaData methods 1079
 class definition 1037
 definition in context 1078
 mapping to Java object type 1089tb
 mapping to Java type 1087tb
 methods for retrieving 1079
 use as a `java.net.URL` object 1079
 using `getString` method to retrieve 1079
DataSource interface
 basic implementation 565, 572
 `getConnection` method 577
 `getLoginTimeout` method 578



`getLogWriter` method 578
implementation for connection pooling 572
implementation for distributed transactions 572, 573
`setLoginTimeout` method 579
`setLogWriter` method 579
URL for basic implementation 572

DataSource object
advantages of using 171T, 566, 575
and `DriverManager`, similarities and differences 565
as representation of a data source 565
creating and deploying 569
creating connections with, as preferred alternative to `DriverManager` for 565
getting a connection with 170T
implementations
basic 168T
that support both connection pooling and distributed transactions 1058
that support connection pooling 168T
that support distributed transactions 168T
properties of 566
tracing and logging 574
using 167T–182T
by `RowSet` object 260T
to make a connection 570

DataSource properties
set by system administrator 568
setting 568
vendor-specific 568

dataSourceName property (DataSource interface) 567

DataTruncation class definition
See also truncation
as `SQLWarning` subclass 85T
constructor 583
`getDataSize` method 585
`getIndex` method 585
`getParameter` method 586
`getRead` method 586
`getTransferSize` method 586
methods 584

DataType.java
as Sample Code 17 233T
source code for 234T

Date class
`Date` constructor 595
`setTime` method 596
`toString` method 596
`valueOf` method 596

Date constructor 595

DATE field (Types class)
class definition 1036
description 1073

Date object
creating with `Calendar` object 590
deprecated methods 592
mapping by `setObject` method to JDBC types 1091tb
mapping for SQL DATE value 589
retrieving 592
zero epoch 589

date(s)
See also escape syntax; time
Date class definition 594
constructors 594
conversion by `setObject` to JDBC types 1091tb
JDBC types mapped to 1087tb
mapping to, JDBC type 1088tb
methods 595
object type, mapped to JDBC type 1090tb

DATE field (Types class) 1036
characteristics and comparison with standard SQL and Java types 1073
mapping to,
database-specific SQL types 1094tb
Java object type 1089tb
Java type 1087tb

DATE type (SQL),
`getInt` method (`ResultSet` interface) not usable for retrieving 63T
functions, support for, as implementation requirement 1106tb
normalizing 589
representing SQL DATE value 590



Timestamp class 1021

DBMS (Database Management System)

See also database(s)

definition,
glossary 1163
term in context 38
establishing a connection with 51T
installing 50T
types, obtaining information about 214T

DCE (Distributed Computing Environment)

JDBC URL use 388

DDL (Data Definition Language)

commands 43
definition, glossary 1163
statements, executing 57T

deadlock

definition, glossary 1163

DECIMAL field (Types class)

See also types
characteristics and comparison with
standard SQL and Java types 1072
class definition 1036
description 1072
mapping to,
database-specific SQL types 1094tb
Java object type 1089tb
Java type 1087tb

default

for auto-commit mode in a distributed
transaction 180T

default connection

for stored procedure using SQLJ and
JDBC API 107T

deleteRow method (ResultSet interface) 711

**deletesAreDetected method
(DatabaseMetaData interface) 468**

deleting

DELETE, as DML command, purpose 43
DELETE statements, executeUpdate
method (Statement interface) use
953
positioned,
definition
glossary 1169
term in context 713
locks as implementation requirement
for drivers 1107
relational tables, DROP TABLE use for 44

row in a result set 711
row programmatically 130T

dependencies

transaction handling of 73T

deployment

DataSource object 568– 569
for distributed transactions 178T
of basic DataSource object 169T
of ConnectionPoolDataSource object,
example 173T
of XADataSource object 1043, 1057
of XADataSource object, example 178T

deprecated methods

complete listing 1146
Date class 592
Time class 1013

Dereference.java

example code showing one way to
dereference a Ref object 684

derefencing

a Ref object 683, 684, 686
attribute of a structured type 1002

**deregisterDriver methods (DriverManager
class) 615**

derived class

definition, term in context 31

**description field (DriverPropertyInfo
class) 625**

**description property (DataSource
interface) 567**

developers

driver,
DatabaseMetaData interface use by
449
Driver interface use 603– 609
tool, DatabaseMetaData interface use by
449

dirty read

definition,
glossary 1163
term in context 75T, 393

disabling

auto-commit mode 73T

disconnected

definition, glossary 1163
RowSet object, description 799

displaying

applet content 101T

data, SELECT use for 43

DISTINCT (SQL99 type)
creating and using 141
definition, glossary 1164

DISTINCT field (Types class)
class definition 1036
description 1076
mapping to Java object type 1089tb
mapping to Java type 1087tb

distinct type
custom mapping 601
defining 599
retrieving 599
storing 600

distributed application
as EJB application 267T, 268T

distributed transaction
application code requirements 573
compared to local transaction 1055
definition
 glossary 1164
deployment for 178T
effect on application code 180T, 1039
example code using 181T
infrastructure, contents 1040
requirements for 1040–1042
step-by-step sequence for 1047–1050
two-phase commit protocol 1045
use of by an EJB component 269T
using connections for 179T

DistributedTransactionBean.java
example code using a distributed
transaction 181T

DML (Data Manipulation Language)
commands 43

DNS (Domain Name System)
JDBC URL use 388

**doesMaxRowSizeIncludeBlobs method
(DatabaseMetaData interface) 468**

Double class
conversion by setObject to JDBC types
 1091tb
mapping to, JDBC types 1090tb

DOUBLE field (Types class)
characteristics and comparison with
 standard SQL and Java types 1071
class definition 1036
description 1071

mapping to,
 database-specific SQL types 1094tb
 Java object type 1089tb
 Java type 1087tb

double Java type
mapping to JDBC type 1088tb

downloading
requirements for using JDBC 2.0 API
 features 114T
sample code, procedures for 85T

driver(s)
definition, glossary 1164
developer information 1097–1115
developers,
 DatabaseMetaData interface use by
 449
 Driver interface use 603–609

Driver interface definition 605
connect method 606
getMajorVersion method 607
getMinorVersion method 607
getPropertyInfo method 608
jbdCompliant method 609
methods 606
registered, maintained by
 DriverManager class 385

DriverManager class definition 614
DBMS connection management by
 53T

deregisterDriver method 615

getConnection method
 method explanation 615
 opening a connection to a
 database with 385, 613
use by DriverManager class
 53T

getDriver method 617

getDrivers method 617

getLoginTimeout method 618

getLogStream method 618

getLogWriter method 619

methods 615

println method 619

registerDriver method
 drivers required to call 604
 method explanation 619
registering 611
responsibilities of 612



setLoginTimeout method 620
setLogStream method 620
setLogWriter method 621
DriverPropertyInfo class definition 624
 choices field 625
 constructor 624
 description field 625
 name field 625
 required field 626
 value field 626
implementation alternative 604
implementation requirements
 security 1112
 static section for creating and
 registering a driver 604, 1102
implementation suggestions
 avoid implementation-dependent
 states 1115
 prefetch rows 1114
 provide "finalize" methods for
 applets 1114
installing 50T
JDBC,
 categories of 25tb, 604
 obtaining 25
JDBC-ODBC bridge driver, as JDBC
 product component 22
loading 52T, 604, 611
manager, JDBC, as JDBC product
 component 22
multiple, access order 613
name, `forName` method (`Class` class) use
 to load 604, 612
registering 604
requirements for, all 1097
reserving subprotocol names for 390
DROP TABLE statements
as DDL command, purpose 44
`executeUpdate` method (`Statement`
 interface) use 953
dynamic
 data access 1084
 database access, OTHER field (`Types` class)
 use with 1034
 execution, obtaining information with
 `ResultSetMetaData` interface
 methods 783

E

EJB component (an enterprise Bean)
in example 573
See also Enterprise JavaBeans (EJB)
 component 573

email addresses
for errors in book 5
for suggestions for future JDBC directions 1159

encapsulation
advantages of 31
definition, term in context 30
of a set of operations, stored procedure use for 77T

endián
Java standard 721

entering
data, into a relational table 58T

enterprise Beans
See Enterprise JavaBeans (EJB)
 component

Enterprise JavaBeans (EJB) application
constituents
 client class 272T
 enterprise Bean 274T–279T
 home interface 272T
 remote interface 271T
 description 267T

Enterprise JavaBeans (EJB) component
definition in context 1050
overview 270T–271T
role in a distributed transaction 1040
specification, URL for 266T
`XACtion` object, importance of use in 1039

entity integrity
definition, term in context 38

equals method (`Timestamp` class) 1028

equals sign (=) 41

error
definition, glossary 1164

error(s)
 applet handling of 101T
 data truncation 581–587
 error code retrieving,
 with `getErrorCode` method
 (`SQLException` class) 909



with `getErrorCode` method
(`SQLWarning` class) 946

exceptions, handling, with `SQLException`
class 907–913

messages, printing 81T

vendor code number, as `SQLException`
component 82T

warnings, handling, with `SQLWarning`
class 945–950

escape syntax

- ?=call keyword 960
- as JDBC method of handling inconsistencies 21
- call keyword 960
- `CallableStatement` use 321
- d keyword
 - description 960
 - example of use 410
- definition, glossary 1164
- escape keyword 958
- example, for calling a stored procedure 108T
- fn keyword 959
- obtaining escape character information 210
- oj keyword 961
- `Statement` use 958
- stored procedures, curly braces as 78T
- t keyword 960
- ts keyword 960
- wildcard characters 959

establishing

- a connection with a DBMS 51T

event

- `ConnectionEvent` object 431
- `RowSetEvent` object 855

event notification

- cursor movement as trigger for 805
- done by driver 431
- for
 - `PooledConnection` objects
 - when a pooled connection is closed 431
 - when an error occurs 432
 - `RowSet` objects 261T, 799, 801

example code

- getting set up to run 50, 114T

See sample code
where to find 86

exceptions

- See also warnings
- catching, all possible, methods for 82T
- chained on
 - `BatchUpdateException` objects 302
 - `SQLException` objects 907
- handling,
 - try and catch use for 81T
 - with `rollback` method (`Connection` interface) 76T
 - with `SQLException` class 907–913
- identifying, with `SQLState` code 82T
- retrieving, completeness requirements 81T
- runtime, definition, glossary 1171

execute method

- modification of for `PreparedStatement` and `CallableStatement` interfaces 647
- `PreparedStatement` interface 659
- `RowSet` interface 805–807, 826
 - metadata, resetting, implementation of 871
- `RowSet` object's use of 262T
- `Statement` interface 975
 - executing stored procedures with 79T
 - handling multiple or optional result sets with 723, 965
 - method explanation 975, 976, 977
 - procedures for using 965
 - purpose of 952

EXECUTE_FAILED field (`Statement` interface)
995

executeBatch method (`Statement` interface)
978

executeQuery method

- executing stored procedures with 70T, 79T
- `PreparedStatement` interface 659
- `PreparedStatement` modification of 647
- return values compared with
 - `executeUpdate` method 70T
- `Statement` interface 57T, 326, 975
 - purpose of 952



- executeUpdate** 982
- executeUpdate method**
 - executing PreparedStatement objects 68T
 - PreparedStatement interface 660
 - PreparedStatement modification of 647
 - Statement interface 975
 - executing SELECT statements with 58T
 - executing stored procedures with 79T
 - method explanation 981, 982
 - purpose of 952
 - return values 70T
- executing**
 - multiple or optional result sets with execute method (Statement interface) 723, 965
 - PreparedStatement objects, with executeUpdate 68T
 - return values from executeQuery compared with executeUpdate 70T
 - statements 57T, 952
 - stored procedures 79T
- extensibility**
 - as Java feature 35
- F**
 - FETCH_FORWARD field (ResultSet interface)** 780
 - FETCH_REVERSE field (ResultSet interface)** 780
 - FETCH_UNKNOWN field (ResultSet interface)** 780
 - fetching data from the database**
 - hint to driver about fetch direction 743, 762
 - hint to driver about fetch size 744, 763
 - fields**
 - See also constants; variables
 - database, limits on, data truncation issues 581
 - definition, term in context 29
 - file**
 - URL specification 387
 - reading data from 882
- files**
 - sample code, customizing 85T
- finalization**
 - definition, glossary 1164
 - methods, suggested for driver implementation 1114
- finally block**
 - use to ensure recycling of a pooled connection 180T, 638
- findColumn method (ResultSet interface)**
 - accessing result set column number with 697
 - method explanation 734
- first method (ResultSet interface)** 695
- flexibility**
 - as characteristic of JDBC-Net pure Java driver 24
- FLOAT field (Types class)** 1036, 1072
- floating point**
 - Float class,
 - conversion by setObject to JDBC types 1091tb
 - mapping to, JDBC types 1090tb
 - FLOAT field (Types class) 1036
 - characteristics and comparison with standard SQL and Java types 1072
 - mapping to
 - database-specific SQL types 1094tb
 - Java object type 1089tb
 - Java type 1087tb
 - float Java type, mapping to JDBC type 1088tb
 - using f to indicate float values in Java code 123T
- fn keyword** 959
- for loop**
 - setting values for PreparedStatement input parameters with 69T
- foreign key**
 - defining in CREATE TABLE statement 221T
 - definition,
 - glossary 1164
 - term in context 42, 54T
- ForeignKeysCoffee.java**
 - as Sample Code 16 222T
 - source code for 222T



forName method (Class class)
exception handling in 81T
loading drivers with 604, 612

framework
SQL, JDBC as 22, 604

ftp (file transfer protocol)
URL specification 387

G

garbage collection
as Java feature 28
definition, term in context 33
external resources not managed by 395

generic
SQL type identifiers,
defined in Types class 1066
synonyms for 20
types, application for creating a table using
233T

getter methods
CallableStatement interface
INOUT parameter use 329
retrieving OUT parameter values
with 326

getArray method (Array interface) 293–
295
difference from method
 ResultSet.getArray 288
return value for 286

getArray method (CallableStatement
interface) 336

getArray method (ResultSet interface)
735
JDBC types retrieved by 699tb,
1092tb

getAsciiStream method (Clob interface)
378

getAsciiStream method (ResultSet
interface)
example 721
JDBC types retrieved by 699tb,
1092tb
retrieving very large amounts of data
using 721

getAttributes method
(DatabaseMetaData interface) 469

getAttributes method (Struct
interface) 998, 1008

getAutoCommit method (Connection
interface) 405

getBaseType method (Array interface)
296

getBaseTypeName method (Array
interface) 296

getBaseTypeName method (Ref interface)
688

getBestRowIdentifier method
(DatabaseMetaData interface) 471

getBigDecimal method
(CallableStatement interface)
336, 337

getBigDecimal method
(CallableStatement interface)
(deprecated) 336

getBigDecimal method (ResultSet
interface) 736, 737
JDBC types retrieved by 699tb,
1092tb

getBinaryStream method (Blob
interface) 314

getBinaryStream method (ResultSet
interface) 737
JDBC types retrieved by 699tb,
1092tb
retrieving very large amounts of data
using 721

getBlob method (CallableStatement
interface) 338

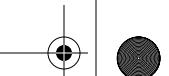
getBlob method (ResultSet interface)
738
JDBC types retrieved by 699tb,
1092tb

getBoolean method
(CallableStatement interface) 338

getBoolean method (ResultSet
interface) 738
JDBC types retrieved by 699tb,
1092tb

getBytes method (CallableStatement
interface) 339

getBytes method (ResultSet interface)
739
JDBC types retrieved by 699tb,
1092tb



getBytes method (*Blob interface*) 314
getBytes method (*CallableStatement interface*) 339
getBytes method (*ResultSet interface*) 739
JDBC types retrieved by 699tb, 1092tb
getCatalog method (*Connection interface*) 405
getCatalogName method (*ResultSetMetaData interface*) 786
getCatalogs method (*DatabaseMetaData interface*) 472
getCatalogSeparator method (*DatabaseMetaData interface*) 473
getCatalogTerm method (*DatabaseMetaData interface*) 473
getCharacterStream method (*Clob interface*) 378
getCharacterStream method (*ResultSet interface*) 739
JDBC types retrieved by 699tb, 1092tb
retrieving very large amounts of data using 721
getBlob method (*CallableStatement interface*) 340
getBlob method (*ResultSet interface*) 740
JDBC types retrieved by 699tb, 1092tb
getColumnClassName method (*ResultSetMetaData interface*) 207T, 787
getColumnCount method (*ResultSetMetaData interface*) 787
getColumnDisplaySize method (*ResultSetMetaData interface*) 787
getColumnLabel method (*ResultSetMetaData interface*) 788
getColumnName method (*ResultSetMetaData interface*) 788
getColumnPrivileges method (*DatabaseMetaData interface*) 473
getColumns method (*DatabaseMetaData interface*) 475
getColumnType method (*ResultSetMetaData interface*) 789
getColumnTypeName method (*ResultSetMetaData interface*) 199T, 203T, 789
getCommand method (*RowSet interface*) 827
getConcurrency method (*ResultSet interface*) 740
getConnection method (*DatabaseMetaData interface*) 477
getConnection method (*DataSource interface*) 577
getConnection method (*DriverManager class*) 615
opening a connection to a database with 52T, 385, 613
getConnection method (*PooledConnection interface*) 645
getConnection method (*RowSetInternal interface*) 860
getConnection method (*Statement interface*) 983
getCrossReference method (*DatabaseMetaData interface*) 219T, 477
getCursorName method (*ResultSet interface*) 741
retrieving a cursor name with 713
getDatabaseMajorVersion method (*DatabaseMetaData interface*) 480
getDatabaseMinorVersion method (*DatabaseMetaData interface*) 480
getDatabaseProductName method (*DatabaseMetaData interface*) 481
getDatabaseProductVersion method (*DatabaseMetaData interface*) 481
getDataSize method (*DataTruncation class*) 585
retrieving DataTruncation information 582
getDataSourceName method (*RowSet interface*) 827
getDate method (*CallableStatement interface*) 341
getDate method (*ResultSet interface*) 742
JDBC types retrieved by 699tb, 1092tb



`getDefautlTransactionIsolation`
method (`DatabaseMetaData`
interface) 481
`getDouble` method (`CallableStatement`
interface) 342
`getDouble` method (`ResultSet` interface)
743
 JDBC types retrieved by 699tb,
 1092tb
`getDriver` method (`DriverManager`
class) 617
`getDriverMajorVersion` method
 (`DatabaseMetaData` interface) 482
`getDriverMinorVersion` method
 (`DatabaseMetaData` interface) 482
`getDriverName` method
 (`DatabaseMetaData` interface) 482
`getDrivers` method (`DriverManager`
class) 617
`getDriverVersion` method
 (`DatabaseMetaData` interface) 483
`getErrorCode` method (`SQLException`
class) 912
 accessing `SQLException` vendor
 error code with 82T
 retrieving `SQLException` error code
 with 909
`getErrorCode` method (`SQLWarning`
class),
 retrieving `SQLWarning` error code
 with 946
`getEscapeProcessing` method (`RowSet`
interface) 828
`getExportedKeys` method
 (`DatabaseMetaData` interface)
 218T, 483
`getExtraNameCharacters` method
 (`DatabaseMetaData` interface) 485
`getFetchDirection` method (`ResultSet`
interface) 743
`getFetchDirection` method (`Statement`
interface) 983
`getFetchSize` method (`ResultSet`
interface) 704, 744
`getFloat` method (`CallableStatement`
interface) 343
`getFloat` method (`ResultSet` interface)
744
 JDBC types retrieved by 699tb,
 1092tb
 retrieving an SQL REAL type value
 61T
 retrieving an SQL VARCHAR type
 value 61T
 retrieving column values with 62T
 retrieving numeric values with 62T
`getHoldability` method (`Connection`
interface) 406
`getIdentiferQuoteString` method
 (`DatabaseMetaData` interface) 486
`getImportedKeys` method
 (`DatabaseMetaData` interface)
 218T, 222T
`getIndex` method (`DataTruncation`
class) 585
 retrieving `DataTruncation`
 information 582
`getIndexInfo` method
 (`DatabaseMetaData` interface) 488
`getInt` method (`CallableStatement`
interface) 344
`getInt` method (`ResultSet` interface)
745
 JDBC types retrieved by 699tb,
 1092tb
 retrieving numeric or character types
 63T
`getJDBCMajorVersion` method
 (`DatabaseMetaData` interface) 490
`getJDBCMinorVersion` method
 (`DatabaseMetaData` interface) 491
`getLoginTimeout` method
 (`ConnectionPoolDataSource`
interface) 444
`getLoginTimeout` method (`DataSource`
interface) 578
`getLoginTimeout` method
 (`DriverManager` class) 618
`getLoginTimeout` method
 (`XADatasource` interface) 1059
`getLogStream` method (`DriverManager`
class) 618
`getLogWriter` method
 (`ConnectionPoolDataSource`
interface) 445



getLogWriter method (*DataSource interface*) 578
getLogWriter method (*DriverManager class*) 619
getLogWriter method (*XADatasource interface*) 1060
getLong method (*CallableStatement interface*) 344
getLong method (*ResultSet interface*) 745
JDBC types retrieved by 699tb, 1092tb
getMajorVersion method (*Driver interface*) 607
getMaxCatalogNameLength method (*DatabaseMetaData interface*) 491
getMaxCharLiteralLength method (*DatabaseMetaData interface*) 492
getMaxColumnNameLength method (*DatabaseMetaData interface*) 212T, 492
getMaxColumnsInGroupBy method (*DatabaseMetaData interface*) 492
getMaxColumnsInIndex method (*DatabaseMetaData interface*) 493
getMaxColumnsInOrderBy method (*DatabaseMetaData interface*) 493
getMaxColumnsInSelect method (*DatabaseMetaData interface*) 493
getMaxColumnsInTable method (*DatabaseMetaData interface*) 494
getMaxConnections method (*DatabaseMetaData interface*) 494
getMaxCursorNameLength method (*DatabaseMetaData interface*) 494
getMaxFieldSize method (*RowSet interface*) 828
getMaxFieldSize method (*Statement interface*) 985
getMaxIndexLength method (*DatabaseMetaData interface*) 495
getMaxProcedureNameLength method (*DatabaseMetaData interface*) 495
getMaxRows method (*RowSet interface*) 829
getMaxRows method (*Statement interface*) 985
getMaxRowSize method (*DatabaseMetaData interface*) 495
getMaxSchemaNameLength method (*DatabaseMetaData interface*) 496
getMaxStatementLength method (*DatabaseMetaData interface*) 496
getMaxStatements method (*DatabaseMetaData interface*) 496
getMaxTableNameLength method (*DatabaseMetaData interface*) 497
getMaxTablesInSelect method (*DatabaseMetaData interface*) 497
getMaxUserNameLength method (*DatabaseMetaData interface*) 497
getMessage method (*SQLException class*)
 retrieving *SQLException* information with 909
getMessage method (*SQLWarning class*),
 retrieving *SQLwarning* information with 946
getMessage,
 accessing *SQLException* error message with 82T
 retrieving *DataTruncation* information 582
getMetaData method (*Connection interface*) 407
creating *DatabaseMetaData* objects with 450
obtaining information about a database with 385
getMetaData method (*PreparedStatement interface*) 661, 662
calling could be expensive 206T
use by tool 206T
getMetaData method (*ResultSet interface*) 745
getMinorVersion method (*Driver interface*) 607
getMoreResults method (*Statement interface*) 330, 986
 handling multiple or optional result sets with 723
getNanos method (*Timestamp class*) 1029



getNextException method
 (*SQLException* class) 913
 retrieving next *SQLException* object
 with 909

getNextWarning method (*SQLWarning* class) 950
 retrieving next *SQLWarning* object
 with 946

getNumericFunctions method
 (*DatabaseMetaData* interface) 498

getObject method (*CallableStatement* interface) 345, 346

getObject method (*Ref* interface) 688, 689

getObject method (*ResultSet* interface)
 retrieving any data type with 698

getObject method (*ResultSet* interface)
 164T, 746, 747
 JDBC types retrieved by 699tb, 1092tb

getOriginalRow method
 (*RowSetInternal* interface) 861

getParameter method (*DataTruncation* class) 586
 retrieving *DataTruncation* information 582

getParameterClassName method
 (*ParameterMetaData* interface) 631

getParameterCount method
 (*ParameterMetaData* interface) 631

getParameterMetaData method
 (*PreparedStatement* interface) 662

getParameterMode method
 (*ParameterMetaData* interface) 632

getParameterType method
 (*ParameterMetaData* interface) 632

getParameterTypeName method
 (*ParameterMetaData* interface) 633

getParams method (*RowSetInternal* interface) 862

getPassword method (*RowSet* interface) 829

getPooledConnection method
 (*ConnectionPoolDataSource* interface) 445, 446

getPrecision method
 (*ParameterMetaData* interface) 633

getPrecision method
 (*ResultSetMetaData* interface)
 206T, 790

getPrimaryKeys method
 (*DatabaseMetaData* interface)
 218T, 498

getProcedureColumns method
 (*DatabaseMetaData* interface) 499

getProcedures method
 (*DatabaseMetaData* interface) 501
 example 450
 obtaining description of available stored procedures 322

getProcedureTerm method
 (*DatabaseMetaData* interface) 502

getPropertyInfo method (*Driver* interface) 608

getQueryTimeout method (*RowSet* interface) 829

getQueryTimeout method (*Statement* interface) 987

getRead method (*DataTruncation* class) 586
 retrieving *DataTruncation* information 582

getRef method (*CallableStatement* interface) 347

getRef method (*ResultSet* interface)
 747
 JDBC types retrieved by 699tb, 1092tb

getResultSet method (*Array* interface) 297–299

getResultSet method (*Statement* interface) 330, 987
 handling multiple or optional result sets with 723

getResultSetConcurrency method
 (*Statement* interface) 988

getResultSetHoldability method
 (*DatabaseMetaData* interface) 503

getResultSetType method (*Statement* interface) 989

getRow method (*ResultSet* interface) 120T, 748

getSavepointId method (*Savepoint* interface) 892



getSavepointName method (*Savepoint interface*) 893
getScale method (*ParameterMetaData interface*) 634
getScale method (*ResultSetMetaData interface*) 206T, 791
getSchemaName method
 (*ResultSetMetaData interface*) 791
getSchemas method (*DatabaseMetaData interface*) 503
 example 450
getSchemaTerm method
 (*DatabaseMetaData interface*) 504
getSearchStringEscape method
 (*DatabaseMetaData interface*)
 209T, 504
getShort method (*CallableStatement interface*) 348
getShort method (*ResultSet interface*)
 748
 JDBC types retrieved by 699tb,
 1092tb
getSQLException method
 (*ConnectionEvent class*) 434
getSQLKeywords method
 (*DatabaseMetaData interface*)
 210T, 504
getSQLState method (*SQLException class*) 912
 accessing *SQLException* SQL state
 component with 82T
 retrieving *DataTruncation*
 information 582
 retrieving *SQLException* *SQLState*
 with 909
getSQLState method (*SQLWarning class*),
 retrieving *SQLwarning* *SQLState*
 with 946
getSQLStateType method
 (*DatabaseMetaData interface*) 505
getSQLTypeName method (*SQLData interface*) 898, 904
getSQLTypeName method (*Struct interface*) 1008
getStatement method (*ResultSet interface*) 749
getString method (*CallableStatement interface*) 349

getString method (*ResultSet interface*)
 749
 advantages of 63T
 JDBC types retrieved by 699tb,
 1092tb
 retrieving an SQL VARCHAR type
 value 61T
 retrieving column values with 62T,
 195T
getStringFunctions method
 (*DatabaseMetaData interface*) 505
getSubString method (*Clob interface*)
 379
getSuperTables method
 (*DatabaseMetaData interface*) 505
getSuperTypes method
 (*DatabaseMetaData interface*) 506
getSystemFunctions method
 (*DatabaseMetaData interface*) 508
getTableName method
 (*ResultSetMetaData interface*) 791
getTablePrivileges method
 (*DatabaseMetaData interface*) 508
getTables method (*DatabaseMetaData interface*) 509
getTableTypes method
 (*DatabaseMetaData interface*)
 213T, 511
getTime method (*CallableStatement interface*) 349, 350
getTime method (*ResultSet interface*)
 749, 750
 JDBC types retrieved by 699tb,
 1092tb
getTimeDateFunctions method
 (*DatabaseMetaData interface*) 512
getTimestamp method
 (*CallableStatement interface*)
 351, 352
getTimestamp method (*ResultSet interface*) 751
 JDBC types retrieved by 699tb,
 1092tb
getTransactionIsolation method
 (*Connection interface*) 407
 determining transaction isolation
 level with 75T



getTransactionIsolation method
 (ResultSet interface) 830

getTransferSize method
 (DataTruncation class) 586

 retrieving DataTruncation
 information 582

getType method (ResultSet interface)
 752

getTypeInfo method
 (DatabaseMetaData interface)
 214T, 234T, 512

getTypeMap method (RowSet interface)
 830

getUDTs method (DatabaseMetaData
 interface) 514

getUnicodeStream method (ResultSet
 interface) 752

 JDBC types retrieved by 699tb

 retrieving very large amounts of data
 using 721

getUpdateCount method (Statement
 interface) 989

 handling multiple or optional result
 sets with 723

getUpdateCounts method
 (BatchUpdateException class) 307

getURL method (CallableStatement
 interface) 352

getURL method (DatabaseMetaData
 interface) 516

getURL method (ResultSet interface)
 753

 JDBC types retrieved by 699tb,
 1092tb

getUrl method (RowSet interface) 831

getUserName method
 (DatabaseMetaData interface) 516

getUsername method (RowSet interface)
 831

getVersionColumns method
 (DatabaseMetaData interface) 516

getWarnings method (Connection
 interface) 408

getWarnings method,
 interfaces that provide 582

 ResultSet interface 754

 Statement interface 990

getXAConnection method
 (XADataSource interface) 1043,
 1060, 1061

getXAResource method (XAConnection
 interface) 1043, 1044, 1053

ResultSet use 692

retrieving information about databases
 with 450

retrieving warnings with 84T

use of by CallableStatement interface
 328

using 61T

global transaction

 definition, glossary 1164

 See distributed transaction

Graphics class (java.awt package)

 writing applets with 100T

GUI (graphical user interface)

 tools, accessing needed information for
 603

H

handling

 exceptions, try and catch use for 81T

HOLD_CURSORS_OVER_COMMIT field

 (ResultSet interface) 780

holdability of a ResultSet object

 a Connection object's holdability
 property 703

 constants specifying 702

 defined in context 702

 setting default holdability 703

 when a statement is created 703

host

 URL specification 387

HTML (Hypertext Markup Language)

 required for running applets 99T

HTTP (Hypertext Transfer Protocol)

 definition, glossary 1165

http (hypertext transfer protocol)

 URL specification 387

I

identifiers

 making database-independent 450



names,
 patterns, case sensitivity of 451
requirement variations 56T

identifying
 columns 62T
 database, in JDBC URL 389
 exceptions, with SQLState code 82T

implementation
 requirements, drivers 604

importedKeyCascade field
 (*DatabaseMetaData interface*) 554

importedKeyInitiallyDeferred field
 (*DatabaseMetaData interface*) 555

importedKeyInitiallyImmediate field
 (*DatabaseMetaData interface*) 555

importedKeyNoAction field
 (*DatabaseMetaData interface*) 555

importedKeyNotDeferrable field
 (*DatabaseMetaData interface*) 555

importedKeyRestrict field
 (*DatabaseMetaData interface*) 556

importedKeySetDefault field
 (*DatabaseMetaData interface*) 556

importedKeySetNull field
 (*DatabaseMetaData interface*) 556

importing
 classes 80T
 packages 80T

IN parameters
 in *CallableStatement* 321
 passing,
 to a *CallableStatement* object 324
 with *PreparedStatement* 648

PreparedStatement 647

very large, sending, Java input stream use
 for 653

inconsistencies
 SQL, JDBC handling of 21

index
 definition,
 glossary 1165
 term in context 62T
 of column, identifying columns by 62T
 of first array element,
 Array object compared to Java array
 289
 retrieving values with 64T

inheritance
 definition, term in context 31
 multiple, interfaces as Java replacement
 for 33

init method
 applet requirements 101T

initial naming context (JNDI) 569
 use for binding logical data source name to
 a *DataSource* object 170T

**initialPoolSize as property for connection
 pooling** 442

INOUT parameters
 CallableStatement use 321, 329

input
 data, truncation exception, as
 implementation requirement for
 drivers 1109
 parameters, *CallableStatement* use 79T
 streams, sending very large amounts of
 data using 653

insert row
 definition
 glossary 1165
 in context 712
 moving to current row from 127T
 using to build a new row 125T

INSERT statements
 as DML command, purpose 43

InsertCoffees.java
 as Sample Code 2 87T
 source code for 89T

inserting
 row in a result set 712
 row in database table with *insertRow*
 method (*ResultSet* interface) 126T

insertRow method (*ResultSet* interface) 712

InsertRows.java
 as Sample Code 20 128T

insertsAreDetected method
 (*DatabaseMetaData interface*) 226, 517

InsertStores.java
 example code for inserting SQL99 types in
 a batch update 158T

InsertSuppliers.java
 as Sample Code 4 91T
 source code for 93T



installing
DBMS 50T
drivers 50T
JDBC 50T

instance(s)
definition, term in context 29

int Java type
mapping to, JDBC type 1088tb

Integer class
conversion by `setObject` to JDBC types 1091tb
mapping to, JDBC types 1090tb

INTEGER field (Types class)
characteristics and comparison with standard SQL and Java types 1070
class definition 1036
description 1070
mapping to,
 database-specific SQL types 1094tb
 Java object type 1089tb
 Java type 1087tb

INTEGER type (SQL)
`getInt` method (`ResultSet` interface) use for retrieving 63T

integrity
data, transaction use to preserve 75T
rules, in relational tables 38

interface(s)
See also class(es); methods
advantages and implementation of 34
classes compared with 34
definition
 glossary 1165
 term in context 33
implementation of all methods, as driver implementation requirement 1098

Internet
See also URL (Uniform Resource Locator)s
Java security features importance for 36
locating resources on, URL use for 387
security requirements for driver implementations 1112

intranet
definition, glossary 1165

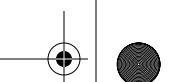
invoking
stored procedures,
 escape clause for 960

syntax for 321

is methods

- `isAfterLast` method (`ResultSet` interface) 121T, 755
- `isAutoIncrement` method (`ResultSetMetaData` interface) 792
- `isBeforeFirst` method (`ResultSet` interface) 755
- `isCaseSensitive` method (`ResultSetMetaData` interface) 792
- `isCatalogAtStart` method (`DatabaseMetaData` interface) 518
- `isClosed` method (`Connection` interface) 409
- `isCurrency` method (`ResultSetMetaData` interface) 793
- `isDefinitelyWritable` method (`ResultSetMetaData` interface) 793
- `isFirst` method (`ResultSet` interface) 755
- `isLast` method (`ResultSet` interface) 756
- `isNullable` method (`ParameterMetaData` interface) 634
- `isNullable` method (`ResultSetMetaData` interface) 793
- `ResultSetMetaData` fields as possible values for 783
- `isReadOnly` method (`Connection` interface) 409
- `isReadOnly` method (`DatabaseMetaData` interface) 519
- `isReadOnly` method (`ResultSetMetaData` interface) 794
- `isReadOnly` method (`RowSet` interface) 832
- `isSearchable` method (`ResultSetMetaData` interface) 794
- `isSigned` method (`ParameterMetaData` interface) 635
- `isSigned` method (`ResultSetMetaData` interface) 795
- `isWritable` method (`ResultSetMetaData` interface) 795

isolation level
See also transactions, isolation levels
definition, glossary 1165



J

J2EE

definition 1165
JDBC API relationship to 26

J2ME

definition 1165
JDBC API relationship to 27

J2SE

definition 1165
JDBC API relationship to 26

JAR file

advantages of using with applets 106T
for storing stored procedure using SQLJ
and JDBC API 108T
SQLJ procedure for installing 108T
use
for installing stored procedure in
DBMS 108T
with applets 106T

Java

advantages for a common relational
database interface 14
book references 80T
language overview 27
object types, conversion by `setObject` to
JDBC types 1091tb
platform components 28
programs, calling stored procedures from
78T
type, specification in setter method names
650
types,
mapping 1065–1093

Java class

mapping by `setObject` method to JDBC
types 1091tb
mapping to JDBC types 1090tb
standard mapping to JDBC types 1088tb

Java Data Objects (JDO) technology

for persisting Java objects 16
relationship to
EJB technology 16
JDBC API 16

Java objects

storing 1140

Java platforms and relationship to JDBC**API**

J2EE 26
J2ME 27
J2SE 26

Java Series books about JDBC API 4**Java Transaction API (JTA) 1047****java.applet package**

Applet class, writing applets with 100T
characteristics of 35

java.awt package

characteristics of 35
Graphics class, writing applets with
100T

java.awt.image package

characteristics of 35

java.io package

characteristics of 34

java.lang package

characteristics of 34

Exception class,
SQLException class inherits from
907

SQLWarning class inherits from 946
Runnable interface, multithreading with
100T

System class, `jdbc.drivers` property,
adding driver names to 612

java.math package

`BigDecimal` class 326
conversion by `setObject` to JDBC
types 1091tb
mapping to JDBC types 1088tb,
1090tb

java.sql package

See also classes (JDBC); interface (JDBC)
characteristics of 35
contents 4
importing 80T

java.util package

characteristics of 34
Vector class, applet use 100T

java.util.Date class

reasons why not usable for SQL DATE
representation 589

java.util.Map object

See type map

**JAVA_OBJECT field (Types class)**

- as UDT 1034
- class definition 1036
- description 1077
- mapping to Java object type 1089tb
- mapping to Java types 1087tb

JavaBeans component

- combining in an application 257T
- definition, glossary 1166
- RowSet object as 797

Java-relational DBMS

- definition in context 25

javax.sql package

- contents 4

javax.transaction.xa.XAResource interface definition 1047**JDBC (Java Database Connectivity)**

- advantages of using 14
- as Java API for executing SQL statements 13
- capabilities 14
- data type variant handling 20
- design history and strategies 1121– 1159
- installing 50T
- `jdbc.driver`, access order 613
- management layer, DriverManager class as 385
- products, URL for 22
- starting 50T
- types, See JDBC types

JDBC 1.0 API

- as provider of basic functionality 3
- compatibility of, with Java 2 SDK 5
- definition
 - glossary 1166
 - in context 113T

JDBC 2.0 API

- as provider of advanced, server-side functionality 3
- definition
 - glossary 1166
 - in context 4

JDBC 2.0 core API

- definition
 - glossary 1166
 - in context 4

JDBC 2.0 Standard Extension API

- definition, glossary 1166

JDBC 3.0 API

- as complete JDBC API 3
- what it includes 4

JDBC Compliant designation

- `jdbcCompliant` method (`Driver` interface) 609

JDBC Connector 26, 1117**JDBC Optional Package API**

- definition 4
- downloading 115T

jdbc subcontext (JNDI) 569**JDBC types**

- `ARRAY` field (Types class) 1036
 - characteristics and comparison with standard SQL and Java types 1076
- `BIGINT` field (Types class) 1036
 - characteristics and comparison with standard SQL and Java types 1071
- `BINARY` field (Types class) 1036
 - characteristics and comparison with standard SQL and Java types 1069
- `BIT` field (Types class) 1036
 - characteristics and comparison with standard SQL and Java types 1070
- `BLOB` field (Types class) 1036
 - characteristics and comparison with standard SQL and Java types 1075

BOOLEAN field (Types class) 1037

- characteristics and comparison with standard SQL and Java types 1078

CHAR field (Types class) 1036

- characteristics and comparison with standard SQL and Java types 1067

CLOB field (Types class) 1037

- characteristics and comparison with standard SQL and Java types 1075

conversion 697



- DATALINK field (*Types* class) 1037
 - characteristics and comparison with standard SQL and Java types 1078
- DATE field (*Types* class) 1036
 - characteristics and comparison with standard SQL and Java types 1073
- DECIMAL field (*Types* class) 1036
 - characteristics and comparison with standard SQL and Java types 1072
- DISTINCT field (*Types* class) 1036
 - characteristics and comparison with standard SQL and Java types 1076
- DOUBLE field (*Types* class) 1036
 - characteristics and comparison with standard SQL and Java types 1071
- FLOAT field (*Types* class) 1036
 - characteristics and comparison with standard SQL and Java types 1072
- implicitly specified in a setter method name 650
- INTEGER field (*Types* class) 1036
 - characteristics and comparison with standard SQL and Java types 1070
- JAVA_OBJECT field (*Types* class) 1036
 - characteristics and comparison with standard SQL and Java types 1077
- LONGVARBINARY field (*Types* class) 1036
 - characteristics and comparison with standard SQL and Java types 1069
- LONGVARCHAR field (*Types* class) 1036
 - characteristics and comparison with standard SQL and Java types 1067
- NULL field (*Types* class) 1036
 - conversion of 329
 - sending as an IN parameter 653
 - testing for 722
- NUMERIC field (*Types* class) 1036
 - characteristics and comparison with standard SQL and Java types 1072
- OTHER field (*Types* class) 1036
- REAL field (*Types* class) 1036
 - characteristics and comparison with standard SQL and Java types 1071
- REF field (*Types* class) 1037
 - characteristics and comparison with standard SQL and Java types 1077
- registering 326
 - as output parameter 329
- SMALLINT field (*Types* class) 1036
 - characteristics and comparison with standard SQL and Java types 1070
- SQL type relationship with 56T
 - as synonym for generic SQL type identifiers 20
- STRUCT field (*Types* class) 1036
 - characteristics and comparison with standard SQL and Java types 1076
- TIME field (*Types* class) 1036
 - characteristics and comparison with standard SQL and Java types 1073
- TIMESTAMP field (*Types* class) 1036
 - characteristics and comparison with standard SQL and Java types 1073
- TINYINT field (*Types* class) 1036
 - characteristics and comparison with standard SQL and Java types 1070
- VARBINARY field (*Types* class) 1036
 - characteristics and comparison with standard SQL and Java types 1069
- VARCHAR field (*Types* class) 1036
 - characteristics and comparison with standard SQL and Java types 1067



jdbc/pool
subcontext for
ConnectionPoolDataSource
objects 173T

jdbc/xa
subcontext for XADataSource objects
178T

jdbc:default:connection
default connection for stored procedure
using SQLJ and JDBC API 107T

jdbcCompliant method (Driver interface)
609

JDBC-Net pure Java driver
DBMS-independence of 24, 604

JDBC-ODBC Bridge
as implementation alternative 604
as JDBC product component 22
definition in context 17
driver 388
installing 50T
intended uses for 23
JDBC advantages 17
JDBC URL for 388
loading 52T

JDBCRowSet class
as example implementation of RowSet
interface 256T

JDK (Java Development Kit) 1.1
definition, glossary 1166

JNDI
advantages of 575
binding logical data source name with
DataSource object 170T
initial naming context 569
jdbc subcontext 569
summary description 1151
use by RowSet object 260T
use in creating an XAConnection object
1043

JNDI (Java Naming and Directory Interface)
definition, glossary 1166

JNDI naming service
registering DataSource object with 170T
using to access remote services over a
network 568

joins
conceptual overview 42
creating, sample code for 94T

definition
glossary 1167
term in context 42, 55T, 71T

Join.java,
as Sample Code 5 94T
source code for 95T

outer, escape syntax 961
using 71T

JSR (Java Specification Request) 169 27

K

**KEEP_CURRENT_RESULT field (Statement
interface) 996**

keys, primary and foreign
obtaining information about 218T

keywords
escape syntax 959
obtaining information about 210T

L

length method (Clob interface) 379

LIKE keyword 40

limits
database fields, data truncation issues 581

listener
definition in context 855
for a RowSet object 262T, 798, 799, 865
registering 800
for PooledConnection events, defined
435
RowSetListener object
registering and deregistering 866

literals
See also constants; fields
date, escape clause for 960
time, escape clause for 960
timestamp, escape clause for 960

loader
class, DriverManager tracking of driver
relationships with 612

loading
a driver 604, 612
drivers 52T

**locating**

rows, cursor use for 44

LOCATOR (SQL99 type)

performance advantage of 283

updates copy or value in the database 313,

377, 519

used to implement

 Array interface 283

 Blob interface 309, 313

 Clob interface 373, 377

locatorsUpdateCopy method

(*DatabaseMetaData interface*) 519

locks

database, minimizing length of holding
75T

definition

 glossary 1167

 term in context 45, 75T

for positioned updates and deletes, as
 implementation requirement for
 drivers 1107

released, by *commit* method 392

logging stream

permission for setting in applet code 941

setting

 using a *DataSource* object 574

 using the *DriverManager* class 621

turning off

 using a *DataSource* object 575

 using the *DriverManager* class 621

logical host

JDBC URL support of 388

logical name

bound to *DataSource* object 570

using 170T

XADatasource object, associated with
1043

Long class

conversion by *setObject* to JDBC types
1091tb

mapping to, JDBC types 1090tb

long Java type

mapping to, JDBC type 1088tb

LONGVARBINARY field (*Types class*)

characteristics and comparison with

 standard SQL and Java types 1069

class definition 1036

description 1069

mapping to,

 database-specific SQL types 1094tb

 Java object type 1089tb

 Java type 1087tb

LONGVARBINARY type (SQL)

getInt method (*ResultSet* interface) not
usable for retrieving 63T

LONGVARCHAR field (*Types class*)

characteristics and comparison with
 standard SQL and Java types 1067

class definition 1036

description 1067

mapping to,
 Java object type 1089tb
 Java type 1087tb

loops

setting values for *PreparedStatement*

 input parameters with 69T

M**Mac operating system**

See also operating systems

appletviewer use 102T

running *CreateNewTable.java* 87T

main method

application requirement for 81T

maintenance of code,

ease of, using JNDI and a *DataSource*
object 171T, 575

management

automatic memory, See garbage collection
JDBC, DriverManager class as layer for
611

of external resources, explicit closing
recommended for 395

mapping

by *ResultSet* getter methods from JDBC
 types to Java types 1091tb

by the method *setObject* from Java
 object types to JDBC types 1090tb,
1091tb

examples of 1080

Java object types to JDBC types 1090tb

Java types to JDBC types 650, 1088tb

setObject compared with setter
methods 651



JDBC types to database-specific SQL types 1094tb
JDBC types to Java object types 1089tb
JDBC types to Java types 1087tb
types, SQL and Java 1065–1095

materialize
definition, glossary 1167

materializing data on the client
definition in context 284, 285
using the method `Array.getArray` 285, 286
when the base type is a primitive type 285
using the method `Array.getResultSet` 285, 287
using the method `Blob.getBinaryStream` 310
using the method `Blob.getBytes` 311
using the method `Clob.getAsciiStream` 374
using the method
 `Clob.getCharacterStream` 374
 using the method `Clob.getString` 374

maxIdleTime as property for connection pooling 442

maxPoolSize as property for connection pooling 442

maxStatements as property for connection pooling 442

memory
management, as Java security feature 36

metadata
as JDBC method of handling inconsistencies 21
definition,
 glossary 1167
 term in context 46
establishing data types with 43
getting with
 `PreparedStatement.getMetaData` method 206T
 `PreparedStatement.getParameterMetaData` method 229T
 `ResultSet.getMetaData` method 206T
`RowSet` interface 807
setting for a `RowSet` object 807
typical users 194T

method(s)
class, interface methods compared with 34
definition,
 glossary 1167
 term in context 29
JDBC types as arguments 1033
signatures, as sole constituent of interface methods 34

Windows operating system
See also operating systems

middle tier
infrastructure for distributed transactions 1040
management of connection pooling and distributed transactions 1056

middleware
net server, JDBC-Net pure Java driver as 24

milliseconds
argument for `Date` constructor, determining 590
getting number of in a `Date` object 591
using L with number literal as argument to constructors for `Date`, `Time`, and `Timestamp` classes 591

minPoolSize as property for connection pooling 442

MIS (Management of Information Services)
definition, glossary 1167
three-tier database access model
advantages for 19

mode
See parameter mode

modifying
relational tables, `ALTER TABLE` use for 44
values, `UPDATE` use for 43

multimedia
Java advantage for 37
Java extensions being developed for 35

multithreading
as database access advantage 20
as Java feature 28
definition
 glossary 1167
 term in context 37
`Runnable` interface use 100T
support for, as implementation requirement for drivers 1108



N

names

- column,
 - handling result set duplicates 697
 - identifying columns by 62T
- driver, `forName` method (`Class` class) use to load 604, 612
- identifier, case sensitivity of patterns 451
- logical
 - See logical name
- name field (`DriverPropertyInfo` class) 625
- network naming system, JDBC URL translation by 388

nanosecond

- definition, term in context 1021

native API partly-Java driver

- conversion of JDBC calls into database-specific calls by 24

native code

- definition, glossary 1168

nativeSQL method (Connection interface)

- 410

net driver 604

network

- naming systems,
 - as JDBC URL subprotocol 388
 - JDBC URL use of 388

- protocol, definition, glossary 1168

networkProtocol property (DataSource interface) 567

new features

- JDBC 2.0 core API
 - batch updates 1134
 - character streams 1135
 - complete list of new API 1148– 1150
 - deprecated methods and constructors 1146
 - fields and methods added to existing interfaces 1142
 - full precision for `Java.math.BigDecimal` 1135
 - performance enhancements 1135
 - programmatic updates 1134
 - SQL99 data types 1136
 - summary of new data types 1123, 1133

summary of new functionality 1132
time zone support 1136

JDBC 3.0 API

- automatically generated keys 1125
 - data types `BOOLEAN` and `DATALINK` 1127
 - holdable cursors 1126
 - multiple open result sets 1125
 - new updating capabilities 1126
 - parameter metadata 1124
 - savepoints 1123
 - statement pooling 1124
- JDBC Standard Extension API
 - complete listing of 1152– 1155
 - summary of new features 1150

next method (ResultSet interface)

- accessing result set rows with 65T
- moving cursor with 61T
- use with scrollable and nonscrollable result sets 118T

NIS (Network Information Service)

- JDBC URL use 388

NO_GENERATED_KEYS field (Statement interface) 955, 996

non-repeatable read

- definition, glossary 1168

normalizing

- time components of a date 589

notation conventions

- curved arrow, identifying output of executing JDBC code 5, 30, 62T
- fixed font, code and system components denoted by 5

icons

- to indicate 2.0 feature 6
- to indicate 3.0 feature 6
- to indicate new features 5
- to indicate Optional Package feature 6

italic fixed font, variables denoted by 5

NULL field (Types class)

- class definition 1036

JDBC

- conversion of 330
- sending as an IN parameter with `setNull` method (`PreparedStatement` interface) 653



mapping to,
database-specific SQL types 1094tb
testing for 722

null value
definition
glossary 1168
term in context 38

nullPlusNonNullIsNull method
(*DatabaseMetaData* interface) 519

nullsAreSortedAtEnd method
(*DatabaseMetaData* interface) 520

nullsAreSortedAtStart method
(*DatabaseMetaData* interface) 520

nullsAreSortedHigh method
(*DatabaseMetaData* interface) 520

nullsAreSortedLow method
(*DatabaseMetaData* interface) 521

numbering
of bytes in a BLOB value 311
of elements in a Java array 311
of IN parameters
in *CallableStatement* objects 324
in *PreparedStatement* objects 648, 649
of OUT parameters
in *CallableStatement* objects 326, 327, 328
of placeholder parameters
in *CallableStatement* objects 328

numbers
advantages as argument to getter methods 63T
column, accessing with *findColumn* method (*ResultSet* interface) 697
comparing, in SELECT statements 41
numeric functions, support for, as implementation requirement 1104

NUMERIC field (Types class) 1036, 1072
mapping to,
Java type 1087tb

O

Object
as a return value
casting before assigning to a variable 288

of *Array.getArray* method 286

object(s)
definition, term in context 29
object-orientation, Java components and characteristics 29
references, as memory pointer replacement 33

obtaining
information about a database, with *Connection.getMetaData* method 385
JDBC drivers 25

ODBC (Open Database Connectivity)
definition, glossary 1168
drivers, Java Software bridge characteristics 604
drivers, Sun Microsystems bridge characteristics 24
JDBC relationship to 17
JDBC URL for 388
odbc subprotocol 389

OID
as possible name for column storing SQL REF values 1001
creating a table with the generated column OID 149T, 1001
definition of as column for storing REF values 1000
SQL code for selecting a value from the column OID 1002
template for creating a table with the generated column OID 1000
use in sample code 23 151T
use in sample code *Dereference.java* 684

oj keyword 961

OLE DB API
JDBC comparison with 18

omission of throws SQLException in method explanations 7

Open Group (Formerly X/Open)
definition, glossary 1168

opening
a connection to a database 385

operating systems
MacOS,
appletviewer use 102T
running sample code 86T



- Microsoft Windows,
 - running sample code 86T, 245T
- UNIX,
 - appletviewer use 102T
 - running sample code 86T, 245T
- Windows 95/NT,
 - appletviewer use 102T
- operators**
 - dereference (->), for accessing SQL REF values 1002
 - dot (.), for accessing attributes of SQL structured types 1002
- optimistic concurrency**
 - definition, glossary 1168
- optimization**
 - See also performance
 - PreparedStatement** advantages 67T
 - stored procedure use 77T
- ordinal position**
 - of parameters, using as argument for setter methods 648
- OTHER field (Types class)**
 - class definition of 1036
 - mapping to, database-specific SQL types 1034, 1094tb
- othersDeletesAreVisible method**
 - (**DatabaseMetaData interface**) 226, 521
- othersInsertsAreVisible method**
 - (**DatabaseMetaData interface**) 522
- othersUpdatesAreVisible method**
 - (**DatabaseMetaData interface**) 522
- OUT parameters** 322
 - as **CallableStatement** result parameter 321
 - in **CallableStatement** 321
 - numbering of 328
 - registering for stored procedures 326
 - retrieving Java types as 1065
 - retrieving results before 329
- outer joins**
 - See also joins 961
 - as inconsistently supported SQL functionality 21
 - escape syntax 961
- output**
 - parameter, **CallableStatement** use 79T
- OutputApplet.html**
 - as Sample Code 8 103T
 - source code for 105T
- OutputApplet.java**
 - as Sample Code 7 103T
 - source code for 103T
- ownDeletesAreVisible method**
 - (**DatabaseMetaData interface**) 523
- ownInsertsAreVisible method**
 - (**DatabaseMetaData interface**) 524
- ownUpdatesAreVisible method**
 - (**DatabaseMetaData interface**) 226, 524

P

- package(s)**
 - definition, term in context 34
 - importing 80T
 - Java examples of 34
- paint method**
 - applet requirements 101T
- parameter fields**
 - parameterModeIn** field
 - (**ParameterMetaData interface**) 635
 - parameterModeInOut** field
 - (**ParameterMetaData interface**) 635
 - parameterModeOut** field
 - (**ParameterMetaData interface**) 636
 - parameterModeUnknown** field
 - (**ParameterMetaData interface**) 636
 - parameterNoNulls** field
 - (**ParameterMetaData interface**) 636
 - parameterNullable** field
 - (**ParameterMetaData interface**) 636
 - parameterNullableUnknown** field
 - (**ParameterMetaData interface**) 636
- parameter mode**
 - discovering 230T
 - ParameterMetaData** fields describing 231T
- parameter placeholders**
 - CallableStatement** parameters, question mark (?) as 321
 - PreparedStatement** parameters, question mark (?) as 648
 - question mark (?) as 322
 - setting values for 649



ParameterMetaData interface
 getParameterClassName method 631
 getParameterCount method 631
 getParameterMode method 631, 632
 getParameterType method 632
 getParameterTypeName method 633
 getPrecision method 633
 getScale method 634
 isNullable method 634
 isSigned method 635
 parameterModeIn field 635
 parameterModeInOut field 635
 parameterModeOut field 636
 parameterModeUnknown field 636
 parameterNoNulls field 636
 parameterNullable field 636
 parameterNullableUnknown field 636

ParameterMetaData object
 creating 627
 definition in text 627
 getting information from 628
 need for using wisely 629

parameters
 CallableStatement use 79T
 for RowSet object's command string,
 setting 803
 getting information about with a
 ParameterMetaData object 331
 identifying by name in
 CallableStatement objects 323
 IN,
 determining the correct setter method
 for setting 651
 in CallableStatement 321
 passing to a CallableStatement
 object 324
 passing with PreparedStatement
 322, 648
 PreparedStatement use 647
 very large, Java input stream use for
 sending 653

 INOUT 322
 CallableStatement 321
 CallableStatement use 329
 example of using 1081

 named
 finding out if supported 324
 getting names of 323

not permitted with Statement objects 390
numbering 328
OUT 322
 as CallableStatement result
 parameter 321, 327
registering for stored procedures 326
retrieving results before 330
permitted with PreparedStatement and
 CallableStatement objects 391
PreparedStatement use with, advantages
 of 67T
PreparedStatement, supplying values
 for 67T

parent class
 definition, term in context 31

passing
 IN parameter, with PreparedStatement
 648

password property (DataSource interface)
 567

patterns
 finding in a Blob object 312
 finding in a Clob object with the method
 position 380
 LIKE keyword use 40
 search, as arguments in
 DatabaseMetaData methods 451

percent sign (%) wildcard 40, 959

performance
 balancing against data consistency needs
 394
 closing result sets explicitly 723
 database locking 75T
 explicit closing recommended for external
 resources 395
 factors affecting 37
 hints to driver
 ResultSet.setFetchDirection
 method 704
 ResultSet.setFetchSize method
 704
 Statement.setFetchDirection
 method 965
 Statement.setFetchSize method
 965

 improving
 hints to driver 704



- PreparedStatement** advantages 66T, 647
- PreparedStatement** objects, possible database-specific limitations 649
- stored procedure advantages 77T
- permanent changes**
 - commit** method (*Connection* interface) consequence 74T
- pessimistic concurrency**
 - definition, glossary 1169
- phantom read**
 - definition, glossary 1169
- placeholders**
 - PreparedStatement** parameters, question mark (?) as 67T, 647
 - setting the value of 648
- Plus (+)**
 - concatenating strings with 56T
- pointers**
 - creation of, prevented by bytecode verifier 36
 - memory, object references as safer replacement for 33
- pooled connection**
 - closing with a `finally` clause to ensure its reuse 180T
 - deployment of
 - ConnectionPoolDataSource** object to produce 172T
 - effect on application code 176T
 - getting and using 174T
 - requirements for 175T
- pooled statement**
 - checking whether supported 655
 - closing 440, 443
 - definition in context 440, 654
 - determined by properties of
 - ConnectionPoolDataSource** object 441
 - effect of closing a **PreparedStatement** object 440
 - properties for 441, 442
 - properties not available to clients 443
 - reuse of **PreparedStatement** objects 440
 - standard properties for 442
 - transparency 440
 - use of 655
- who can implement** 440
- PooledConnection interface**
 - addConnectionEventListener** method 644
 - close** method 644
 - getConnection** method 645
 - removeConnectionEventListener** method 645
- PooledConnection object**
 - closing 642
 - difference from *Connection* object 439, 641
 - life cycle of 642
 - obtaining and using 639
- populating**
 - relational tables 58T, 71
 - tables, sample code for 87T, 91T
- portability**
 - See also performance
 - advantage of using *DataSource* object to get connection 171T, 566
 - as JDBC advantage 18
 - avoiding implementation-dependent states, as driver implementation suggestion 1115
 - dates, increasing in retrieval of 593
 - EJB server with respect to JDBC drivers 178T
 - Java characteristics that provide 28
 - Java endian standard 721
 - making patterns database-independent 451
 - maximum field size recommendations 581
 - retrieving results before OUT parameters 329
 - SQL type mapping issues 1066
- portNumber property (*DataSource* interface)** 567
- position method (*Blob* interface)** 316
- position method (*Clob* interface)** 380
- positioned update/delete**
 - definition
 - glossary 1169
 - term in context 713
 - locks for, as implementation requirement for drivers 1107
 - requirements for 714

**precision**

definition,
glossary 1169
term in context 206

prefetching

of rows, as suggested driver
implementation 1114

prepareCall method (Connection interface)

410, 411, 413

prepared statement

definition, glossary 1169

PreparedStatement interface

setter methods, passing IN parameter
values with 324
`addBatch` method 658
as one of three classes for sending SQL
statements to a database 390
`clearParameters` method 658
`execute` method 659
`executeQuery` method 659
`executeUpdate` method 660
`getMetaData` method 661, 662
`getParameterMetaData` method 662
sending Java types as SQL IN parameters
1065
`setArray` method 662
`setAsciiStream` method 662
`setBigDecimal` method 357, 663
`setBinaryStream` method 357, 664
`setBlob` method 664
`setBoolean` method 665
`setByte` method 358, 665
`setBytes` method 359, 665
`setCharacterStream` method 359, 666
`setClob` method 667
`setDate` method 360, 667
`setDouble` method 361, 668
`setFloat` method 361, 668
`setInt` method 362, 669
`setLong` method 362, 669
`setNull` method 363, 669, 670
`setObject` method 364, 365, 366, 671,
672, 673
`setRef` method 673
`setShort` method 367, 674
`setString` method 367, 674
`setTime` method 368, 675

`setTimestamp` method 369, 676

`setUnicodeStream` method 677

`setURL` method 678

Statement and **CallableStatement**

interfaces compared with 953

warnings reported on 84T

when to use 66T

PreparedStatement object

as precompiled statement 647

creating 648

getting information about its parameters
650

IN parameters 647

pooling of 654

setting parameter placeholders with
SQL99 data types 649

setting parameter placeholders with values
648

that produce scrollable and/or updatable
ResultSet objects 648

when use is less efficient 649

prepareStatement method (Connection

interface) 414, 415, 416, 417, 418, 420
method explanation 414

PreparedStatement objects created by
390

previous method (ResultSet interface)

118T, 759

primary key

defining in CREATE TABLE statement 219T
definition,

glossary 1169

term in context 38

PrimaryKeysSuppliers.java
as Sample Code 15 219T

source code for 219T

selecting during relational table creation
54T

printing

error messages 81T

PrintColumns.java

as Sample Code 11 199T

source code for 200T

PrintColumnsTypes.java

as Sample Code 10 199T

source code for 199T

`println` method (**DriverManager** class)
619

**procedures**

- procedureColumnIn field
 - (DatabaseMetaData interface) 556
- procedureColumnInOut field
 - (DatabaseMetaData interface) 557
- procedureColumnOut field
 - (DatabaseMetaData interface) 557
- procedureColumnResult field
 - (DatabaseMetaData interface) 557
- procedureColumnReturn field
 - (DatabaseMetaData interface) 557
- procedureColumnUnknown field
 - (DatabaseMetaData interface) 558
- procedureNoNulls field
 - (DatabaseMetaData interface) 558
- procedureNoResult field
 - (DatabaseMetaData interface) 558
- procedureNullable field
 - (DatabaseMetaData interface) 558
- procedureNullableUnknown field
 - (DatabaseMetaData interface) 559
- procedureResultUnknown field
 - (DatabaseMetaData interface) 559
- procedureReturnsResult field
 - (DatabaseMetaData interface) 559
- stored, term definition 45

programmatic updates

- using updatable methods (ResultSet interface) 123T

programming practices

- See also performance; portability
- applet design 102T
- JDBC type name use 1033, 1034

properties

- DataSource object, of 566
- RowSet object, of 801
- setting
 - for ConnectionPoolDataSource object 172T
 - for DataSource object 170T
 - for RowSet object 258T
 - for XADataSource object 178T

propertyCycle as property for connection pooling 442**protected keyword**

- as field access specifier 30

protocol

- definition, glossary 1169
- Internet resource access 387

prototypes

- interface methods as 34

pseudo column

- definition, glossary 1169
- definition, in context 453

public keyword

- definition in context 30

pure Java

- definition, glossary 1169

Q**query(s)**

- definition,
 - glossary 1169
 - term in context 39
- encapsulating a set of, stored procedure use for 77T
- joins 196T
- results from stored in result sets 691
- SELECT use for 43

question mark (?) parameter placeholders

- CallableStatement use 321
- parameter numbers in reference to 328, 647
- PreparedStatement use 67T, 647
- setting value of, with PreparedStatement 648

quick reference card 10**quotation marks**

- double ("), to indicate String in Java code 58T
- quoted strings, case sensitivity of 56T
- single ('), to indicate character literal 58T

R**RDBMS (Relational Database Management System)**

- definition
 - glossary 1170
 - term in context 38



readData method (RowSetReader interface)
invocation of RowSet methods 883
method explanation 883
possible implementations 882
reading data from a regular file, possible
 implementation for 882
typical implementation of 806

reader
default 882
definition, glossary 1170
for RowSet object 798
reading non-SQL data, implementation of
 method RowSetReader.readData
 for 882
RowSetReader object 881

reader methods (SQLInput interface)

- readArray 918
- readAsciiStream 918
- readBigDecimal 919
- readBinaryStream 919
- readBlob 919
- readBoolean 920
- readByte 920
- readBytes 920
- readCharacterStream 921
- readClob 921
- readDate 922
- readDouble 922
- readFloat 922
- readInt 923
- readLong 923
- readObject 923
- readRef 924
- readShort 924
- readString 925
- readTime 925
- readTimestamp 926
- readURL 926

reader/writer framework 881, 885

reading
data from a regular file 882
data truncation on, handling 581
data, handling problems with 85T
dirty read avoidance 75T

readSQL method (SQLData interface) 899, 905

REAL field (Types class)
characteristics and comparison with
 standard SQL and Java types 1071

class definition of 1036
description 1071
mapping to,
 database-specific SQL types 1094
 Java object type 1089tb
 Java type 1087tb

record
definition, glossary 1170

REF (SQL type)
comparison with
 a primary key 679
 an SQL LOCATOR 679
creation of table containing 681
definition,
 glossary 1170
 in context 679
generated by DBMS 681
identifier for instance of SQL structured
 type 1000
persistence of 680
requirements for 680

REF field (Types class)
class definition of 1037
description 1077
mapping to Java object type 1089tb
mapping to Java type 1087tb

Ref interface

- getBaseTypeName method 688
- getObject method 688, 689
- setObject method 689

Ref object
creating 682
dereferencing 683, 684, 686
difference from SQL LOCATOR 679
duration of 680
mapping by setObject method to JDBC
 types 1091tb
mapping from Java object types to JDBC
 types 1090tb
SQL REF, mapping of 679
standard mapping to JDBC types 1088tb
storing 683

references
book, Java 80T

referential integrity
definition
 glossary 1170
 term in context 42



refreshRow method (ResultSet interface) 720
refreshRow method (ResultSet interface) 759
registering
 DataSource object with JNDI naming service 170T, 568
 drivers 604, 611, 619
 registerDriver Method
 (DriverManager class) 619
 drivers required to call 611
 registerDriver method
 (DriverManager class)
 drivers required to call 603
 registerOutParameter method
 (CallableStatement interface)
 353, 354, 355
 INOUT parameter use 329
 registering JDBC types with 326
 stored procedure OUT parameters 326
 subprotocols 390
relation
 definition
 glossary 1170
 term in context 38
relational databases
 conceptual overview 38
 definition, term in context 38
relational tables
 See tables
relative method (ResultSet interface)
 argument, positive or negative 120T
 erroneous use, example of 694
relative positioning
 definition, glossary 1170
releaseSavepoint method (Connection interface) 421
removeConnectionEventListener method (PooledConnection interface) 645
removeRowSetListener method (RowSet interface) 832
removing
 rows, DELETE use for 43
required field (DriverPropertyInfo class) 626

requirements

implementation
 for drivers 603
 for drivers, function support 1104tb
 permitted variants 1111

resource manager

definition in context 1058
role in two-phase commit protocol 1045
voting on whether to commit a transaction 1046

resource(s)

external, explicit closing recommended for 395

result parameter

in CallableStatement 321

result sets

concurrency
 methods to discover 708
 requesting when unsupported 709
concurrency of 701
contents, distinguishing from application output 62T

definition,
 term in context 44, 61T
getting information about 193T

in stored procedures 197T

multiple,

 execute method (Statement interface) use 953

getMoreResults method

 (Statement interface), handling multiple or optional result sets with 723

getResultSet method (Statement interface), handling multiple or optional result sets with 723

getUpdateCount method

 (Statement interface), handling multiple or optional result sets with 723

number of rows in, determining 694

obtaining information about 783

optional, handling 723

performance hints, providing 704



ResultSet interface definition 724
CallableStatement handling of large Out values compared with 328
close method, closing result sets with 723
creating 61T
findColumn method, accessing result set column number with 697
getCursorName method 713
getString method 195T
methods 730
next method, moving to next row of a result set with 691
objects, methods that return 213T
retrieving results before OUT parameters 330
retrieving values from 61T
values, JDBC types as 1033
warnings reported on 84T
wasNull method, testing for JDBC NULL value with 722, 778
ResultSet objects, as return values for DatabaseMetaData methods 450
retrieving very large values from 720
return value for DatabaseMetaData methods, as 450
type
 methods to discover 708
 requesting unsupported 708
 scroll-insensitive 717
 scroll-sensitive 717
types of 700
updatable
 queries that produce 714
visibility of changes
 detecting 719
 using getter methods to determine 718
ResultSet interface
 absolute method 731
 afterLast method 732
 beforeFirst method 732
 cancelRowUpdates method 733
 clearWarnings method 733
 close method 733
CLOSE_CURSORS_AT_COMMIT field 779
CONCUR_READ_ONLY field 779
CONCUR_UPDATABLE field 779
deleteRow method 734
FETCH_FORWARD field 780
FETCH_REVERSE field 780
FETCH_UNKNOWN field 780
findColumn method 734
first method 734
getArray method 735
getAsciiStream method 735
getBigDecimal method 736, 737
getBinaryStream method 737
getBlob method 738
getBoolean method 738
getByte method 739
getBytes method 739
getCharacterStream method 739
getClob method 740
getConcurrency method 740
getCursorName method 741
getDate method 742
getDouble method 743
getFetchDirection method 743
getFetchSize method 744
getFloat method 744
getInt method 745
getLong 745
getMetaData method 745
getObject method 746, 747
getRef method 747
getRow method 748
getShort method 748
getStatement method 749
getString method 749
getTime method 749, 750
getTimestamp method 751
getType method 752
getUnicodeStream method 752
getURL method 753
getWarnings method 754
HOLD_CURSORS_OVER_COMMIT field 780
insertRow method 754
isAfterLast method 755
isBeforeFirst method 755
isFirst method 755
isLast method 756



last method 756
moveToCurrentRow method 757
moveToInsertRow method 757
next method 758
previous method 759
refreshRow method 759
relative method 760
rowDeleted method 761
rowInserted 761
rowUpdated method 762
setFetchDirection method 762
setFetchSize method 763
TYPE_FORWARD_ONLY field 781
TYPE_SCROLL_INSENSITIVE field 781
TYPE_SCROLL_SENSITIVE field 781
updateArray method 763
updateAsciiStream method 764
updateBigDecimal method 765
updateBinaryStream method 765
updateBlob method 766
updateBoolean method 766
updateByte method 767
updateBytes method 768
updateCharacterStream method 768
updateClob method 769
updateDate method 769
updateDouble method 770
updateFloat method 771
updateInt method 771
updateLong method 772
updateNull method 772
updateObject method 773, 774
updateRef method 774
updateRow method 775
updateShort method 775
updateString method 776
updateTime method 776
updateTimestamp method 777
wasNull method 778

ResultSet objects

as materialized ARRAY elements,
example of retrieving values from
290
as return value for `Array.getResultSet`
method 287
creating with `prepareStatement` method
(`Connection` interface) 707

holdability 702
retrieving column values from 696
scrollability 700, 702
update capabilities (concurrency) 701

ResultSetMetaData interface

`columnNoNulls` field 795
`columnNullable` field 796
`columnNullableUnknown` field 796
fields 795
`getCatalogClassName` method 787
`getCatalogName` method 786
`getColumnCount` method 195T, 787
`getColumnDisplaySize` method 787
`getColumnLabel` method 788
`getColumnName` method 788
`getColumnType` method 789
`getColumnTypeName` method 199T,
203T, 789
`getPrecision` method 206T, 790
`getScale` method 206T, 791
`getSchemaName` method 791
`getTableName` method 203T, 791
`isAutoIncrement` method 206T, 792
`isCaseSensitive` method 203T, 792
`isCurrency` method 793
`isDefinitelyWritable` method 793
`isNullable` method 793
`isReadOnly` method 794
`isSearchable` method 794
`isSigned` method 206T, 795
`isWritable` method 203T, 795
methods 786

ResultSetMetaData objects

creating with
`PreparedStatement.getMetaData`
method 785
getting information from 784

retrieving

column values from a result set 696
`DataTruncation` information 582
methods for 582
driver information, with `Driver` methods
603
exceptions, completeness requirements
81T
information,
about a database 449– 563
about DBMS types 214T



result set values 691
result set, very large values 720
results before OUT parameters 326
SQLException information 909
SQLWarning information 946
values,
 from result sets 61T
 with getter methods 62T
warnings 83T

return values
 See also values 450
 DatabaseMetaData methods, ResultSet object as 450
 for executeUpdate method 70T

RETURN_GENERATED_KEYS field (Statement interface) 955, 996

roleName property (DataSource interface) 567

rollback
 definition,
 glossary 1170
 term in context 45
 rollback method (Connection interface) 392
 calling, conditions which mandate 76T
 transactions terminated by 392
 undoing changes 75T
 rollback statement, managing data integrity with 45

rollback method (Connection interface) 421, 422

rowChanged method (RowSetListener interface) 869

rowInserted method (ResultSet interface) 226

rows
 accessing, with next method 65T
 adding, INSERT use for 43
 cursor use to locate 44
 deleting programmatically 130T
 inserting
 new value or default value required
 for each column 127T
 inserting programmatically 125T
 lock, definition, term in context 45
 moving cursor to, with next method 61T

ordering of, in a result set as opposed to in a database table 125T
prefetching of, as suggested driver implementation 1114
removing, DELETE use for 43
result set,
 changing 713
 very large values, using streams for 720

rowset
 See also RowSet object
 definition
 glossary 1170
 in context 255T
 summary description 1151
uses 255T

RowSet interface
 addRowSetListener method 825
 clearParameters method 826
 execute method 826
 getCommand method 827
 getDataSourceName method 827
 getEscapeProcessing method 828
 getMaxFieldSize method 828
 getMaxRows method 829
 getPassword method 829
 getQueryTimeout method 829
 getTransactionIsolation method 830
 getTypeMap method 830
 getUrl method 831
 getUsername method 831
 implementation provided by driver vendor 1099
 isReadOnly method 832
 not part of driver implementation 1099
 planned implementations of 256T
 removeRowSetListener method 832
 setArray method 832
 setAsciiStream method 833
 setBigDecimal method 834
 setBinaryStream method 834
 setBlob method 835
 setBoolean method 835
 setByte method 835
 setBytes method 836
 setCharacterStream method 836
 setClob method 837



setCommand method 837
setConcurrency method 838
setDataSourceName method 838
setDate method 838, 839
setDouble method 840
setEscapeProcessing method 840
setFloat method 840
setInt method 841
setLong method 841
setMaxFieldSize method 841
setMaxRows method 842
setNull method 842, 843
setObject method 844, 845, 846
setPassword method 847
setQueryTimeout method 847
setReadOnly method 847
setRef method 848
setShort method 848
setString method 848
setTime method 849
setTimestamp method 850
setTransactionIsolation method 851
setType method 851
setTypeMap method 852
setUrl method 852
setUsername method 853

RowSet object
connected and disconnected 255T
DataSource object, use of for getting a connection 260T
definition in context 255T, 797
differences from ResultSet objects 797
disconnected
 facilities needed by 256T
EJB component, use in 277T
event notification for cursor movement 805
events on 799
executing its command string 805
JNDI used by 260T
listener associated with 798
listeners, registering 800
metadata
 retrieving 807
 setting 807
original values 806
properties 798, 801

scrollable and updatable, making 262T
setting properties on 258T, 802
traversing 804
type and concurrency, setting 803
TYPE_SCROLL_SENSITIVE, effect on connected and disconnected 803
updating 263T

rowSetChanged method (RowSetListener interface) 806, 869

RowSetEvent interface
constructor 857

RowSetEvent object
creating 856
definition in context 855

RowSetInternal interface
getConnection method 860
getOriginal method 860
getOriginalRow method 861
getParams method 862
setMetaData method 862

RowSetInternal object
definition in context 859
purpose 859

RowSetListener interface
cursorMoved method 868
methods described 865
rowChanged method 869
rowSetChanged method 869

RowSetListener object
definition in context 865
registering and deregistering 866
requirements for becoming 865

RowSetMetaData interface
implementation required if RowSet implementation provided by driver vendor 1099
setAutoIncrement method 873
setCaseSensitive method 873
setCatalogName method 874
setColumnCount method 874
setColumnDisplaySize method 875
setColumnName method 875
setColumnType method 876
setColumnName method 876
setCurrency method 876
setNullable method 877



- setPrecision** method 877
- setScale** method 878
- setSchemaName** method 878
- setSearchable** method 879
- setSigned** method 879
- setTableName** method 880
- RowSetMetaData object**
 - purpose 871
 - relation to **ResultSetMetaData** methods 871
- RowSetReader.readData** method's actions on 806
- RowSetReader interface**
 - readData** method 883
- RowSetReader object**
 - provided by 881
 - purpose 881
- RowSetWriter interface**
 - wroteData** method 887
- rowUpdated** method (**ResultSet** interface) 226
- RSMetaDataMethods.java**
 - as Sample Code 12 203
 - source code for 203T
- Runnable interface (java.lang package)**
 - multithreading with 100T
- running**
 - applets 102T
- runtime**
 - exception, definition, glossary 1171
- S**
- sample code**
 - 01, See *CreateCoffees.java*
 - 02, See *InsertCoffees.java*
 - 03, See *CreatSuppliers.java*
 - 04, See *InsertSuppliers.java*
 - 05, See *Join.java*
 - 06, See *TransactionPairs.java*
 - 07, See *OutputApplet.java*
 - 08, See *OutputApplet.html*
 - 09, See *SQLStatement.java*
 - 10, See *PrintColumnTypes.java*
 - 11, See *PrintColumns.java*
- scalar**
 - functions,
 - including in escape clauses 959
 - support for, as implementation requirement 1104tb
 - value, definition, glossary 1171
- scale**
 - definition,
 - glossary 1171
 - term in context 206
- 12, See *RSMetaDataMethods.java***
- 13, See *TableTypes.java***
- 14, See *TypeInfo.java***
- 15, See *PrimaryKeysSuppliers.java***
- 16, See *ForeignKeysCoffees.java***
- 17, See *DataType.java***
- 18, See *CreateNewTable.java***
- 19, See *CreateNewType.java***
- 20, See *InsertRows.java***
- 21, See *BatchUpdate.java***
- 22, See *CreateUDTs.java***
- 23, See *CreateRef.java***
- 24, See *CreateStores.java***
- 25, See *InsertStores.java***
- 27, See *DistributedTransactionBean.java***
- 28, See *SetSavepoint.java***
- 29, See *AutoGenKeys.java***
- 30, See *Dereference.java***
- customizing** 85T
- <http://java.sun.com/products/jdbc/book.html>** 86T

**schema**

definition, glossary 1171

scope

importing classes 80T

scripting languages

Java comparison with 37

scrollable result set

creating with

 PreparedStatement object 648

 Statement object 116T

cursor movement, methods for 693

definition, glossary 1171

driver implementation determines

 availability 117T

overhead for using 116T

produced only if driver supports 122T

reasons for making updatable 122T

specifying a type and concurrency,

 requirements 116T

support for, determining with

 supportsResultSetType method

 (DatabaseMetaData interface) 225T

search patterns

as arguments in DatabaseMetaData

 methods 451

security

as database access advantage 19

as Java feature 28, 36

DriverManager tracking of class loader

 and driver relationships 612

in applets 941

manager, access restrictions enforced by

 36, 941

requirements, for driver implementation
1112

selecting

SELECT statement,

 as DML command, purpose 43

conceptual overview 39

executeQuery method (Statement
interface) use 57T, 953

generating data values with 60T

InsertCoffees.java use 88T

tables 59T

sending

See also input

Java types as SQL IN parameters 1065

large amounts of data using streams 653

SQL statements 390

sensitive result set

support for, determining with

 supportsResultSetType method

 (DatabaseMetaData interface) 225T

server

definition, glossary 1171

serverName property (DataSource interface)

567

session

definition in context 385

SessionBean object

stateless 270T

SetSavepoint.java

example code using savepoints 183T

setter methods

JDBC type names, implicitly specified in
650

passing IN parameters with 324

setArray method (PreparedStatement
interface) 662

setArray method (RowSet interface) 832

setAsciiStream method

 (CallableStatement interface) 356

setAsciiStream method (Clob interface)
381

setAsciiStream method

 (PreparedStatement interface) 662

 sending very large amounts of data
 using 653

setAsciiStream method (RowSet
interface) 833

setAutoCommit method (Connection
interface) 422

setAutoIncrement method

 (RowSetMetaData interface) 873

setBigDecimal method

 (CallableStatement interface) 357

setBigDecimal method

 (PreparedStatement interface) 663

setBigDecimal method (RowSet
interface) 834

setBinaryStream method (Blob
interface) 317



setBinaryStream method
 (CallableStatement interface) 357

setBinaryStream method
 (PreparedStatement interface) 664

 sending very large amounts of data
 using 653

setBinaryStream method (RowSet
 interface) 834

setBlob method (PreparedStatement
 interface) 664

setBlob method (RowSet interface) 835

setBoolean method
 (CallableStatement interface) 358

setBoolean method
 (PreparedStatement interface) 665

setBoolean method (RowSet interface)
 835

setByte method (CallableStatement
 interface) 329, 358

setByte method (PreparedStatement
 interface) 665

setByte method (RowSet interface) 835

setBytes method (Blob interface) 317,
 318

setBytes method (CallableStatement
 interface) 359

setBytes method (PreparedStatement
 interface) 665

 capacity of 653

setBytes method (RowSet interface) 836

setCaseSensitive method
 (RowSetMetaData interface) 873

setCatalog method (Connection
 interface) 423

setCatalogName method
 (RowSetMetaData interface) 874

setCharacterStream method
 (CallableStatement interface) 359

setCharacterStream method (Clob
 interface) 382

setCharacterStream method
 (PreparedStatement interface) 666

setCharacterStream method (RowSet
 interface) 836

setClob method (PreparedStatement
 interface) 667

setClob method (RowSet interface) 837

setColumnCount method
 (RowSetMetaData interface) 874

setColumnDisplaySize method
 (RowSetMetaData interface) 875

setColumnLabel method
 (RowSetMetaData interface) 875

setColumnName method
 (RowSetMetaData interface) 875

setColumnType method
 (RowSetMetaData interface) 876

setColumnType method
 (RowSetMetaData interface) 876

setCommand method (RowSet interface)
 837

setConcurrency method (RowSet
 interface) 838

setCurrency method (RowSetMetaData
 interface) 876

setCursorName method (Statement
 interface) 991

setDataSourceName method (RowSet
 interface) 838

setDate method (CallableStatement
 interface) 360

setDate method (PreparedStatement
 interface) 667

setDate method (RowSet interface) 838,
 839

setDouble method (CallableStatement
 interface) 361

setDouble method (PreparedStatement
 interface) 668

setDouble method (RowSet interface)
 840

setEscapeProcessing method (RowSet
 interface) 840

setEscapeProcessing method
 (Statement interface) 992

 turning escape processing on and off
 with 961

setFetchDirection method (ResultSet
 interface) 705, 762

setFetchDirection method (Statement
 interface) 705, 992



setFetchSize method (ResultSet interface) 763
setFetchSize method (Statement interface) 704, 993
setFloat method (CallableStatement interface) 361
setFloat method (PreparedStatement interface) 668
setFloat method (RowSet interface) 840
setHoldability method (Connection interface) 424
setInt method (CallableStatement interface) 362
setInt method (PreparedStatement interface) 669
setInt method (RowSet interface) 841
setLoginTimeout method (ConnectionPoolDataSource interface) 446
setLoginTimeout method (DataSource interface) 579
setLoginTimeout method (DriverManager class) 620
setLoginTimeout method (XADatasource interface) 1062
setLogStream method (DriverManager class) 620
setLogWriter method (ConnectionPoolDataSource interface) 447
setLogWriter method (DataSource interface) 579
setLogWriter method (DriverManager class) 621
setLogWriter method (XADatasource interface) 1062
setLong method (CallableStatement interface) 362
setLong method (PreparedStatement interface) 669
setLong method (RowSet interface) 841
setMaxFieldSize method (RowSet interface) 841
setMaxFieldSize method (Statement interface) 994
setMaxRows method (RowSet interface) 842
setMaxRows method (Statement interface) 994
setMetaData method (RowSetInternal interface) 862
setNanos method (Timestamp class) 1029
setNextException method (SQLException class) 913
setNextException method (SQLWarning class) 950
setNull method (CallableStatement interface) 363
setNull method (PreparedStatement interface) 363, 669, 670
sending JDBC NULL value as an IN parameter with 653
setNull method (RowSet interface) 842, 843
setNullable method (RowSetMetaData interface) 877
setObject method (CallableStatement interface) 364, 365, 366
setObject method (PreparedStatement interface) 671, 672, 673, 1007 using 651
setObject method (Ref interface) 689
setObject method (RowSet interface) 844, 845, 846
setPassword method (RowSet interface) 847
setPrecision method (RowSetMetaData interface) 877
setQueryTimeout method (RowSet interface) 847
setQueryTimeout method (Statement interface) 994
setReadOnly method (Connection interface) 424
setRef method (PreparedStatement interface) 673
setRef method (RowSet interface) 848
setSavepoint method (Connection interface) 425
setScale method (RowSetMetaData interface) 878
setSchemaName method (RowSetMetaData interface) 878



setSearchable method
(RowSetMetaData interface) 879

setShort method (CallableStatement interface) 367

setShort method (PreparedStatement interface) 674

setShort method (RowSet interface) 848

setSigned method (RowSetMetaData interface) 879

setString method (CallableStatement interface) 367

setString method (Clob interface) 382, 383

setString method (PreparedStatement interface) 674

capacity of 653

setString method (RowSet interface) 848

setTableName method (RowSetMetaData interface) 880

setTime method (CallableStatement interface) 368

setTime method (Date class) 596

setTime method (PreparedStatement interface) 675

setTime method (RowSet interface) 849

setTimestamp method
(CallableStatement interface) 369

setTimestamp method
(PreparedStatement interface) 676

setTimestamp method (RowSet interface) 850

setting placeholder parameters with 648

setTransactionIsolation method
(Connection interface) 426

changing transaction isolation levels with 76T

setTransactionIsolation method
(RowSet interface) 851

setType method (RowSet interface) 851

setTypeMap method (Connection interface) 427

setTypeMap method (RowSet interface) 852

setUnicodeStream method
(PreparedStatement interface)

method explanation 677

sending very large amounts of data using 653

setURL method (CallableStatement interface) 370

setURL method (PreparedStatement interface) 678

setUrl method (RowSet interface) 852

setUsername method (RowSet interface) 853

used with CallableStatement INOUT parameters 329

values supplied by, for PreparedStatement parameters 67T

setting

transaction isolation level value 75T

up databases 51T

up tables 53T

short Java type

mapping to, JDBC type 1088tb

simplicity

as Java feature supporting correct code development 33

size

database field, data truncation issues 581

Java advantages 37

SMALLINT field (Types class)

See also types

characteristics and comparison with standard SQL and Java types 1070

class definition of 1036

description 1070

mapping to,

database-specific SQL types 1094tb

Java object type 1089tb

Java type 1087tb

Solaris platform (UNIX)

running CreateNewTable.java 87T

source

of a PooledConnection close event 642

of a PooledConnection exception event 643

of a RowSet event 855

spaces

significance in quoted strings 58T

SPACE(count) string function, scalar function supported by JDBC 1106

**special characters**

See curly braces ({ }); dot (.); equals sign (=); percent sign (%) wildcard; question mark (?); quotation marks; spaces; underbar(_) wildcard

SQL

command categories 43
conformance issues 20
definition
 glossary 1171
 in context 27, 39
overview of basic functionality 39– 46
type identifiers 20

SQL (Structured Query Language)

CallableStatement interface 321– 371
exceptions, SQLException class 907– 913
generic type identifiers 1033– 1037
generic types, JCBC types as 56T
precompiled statements,
 PreparedStatement interface use 647
PreparedStatement interface 647– 677
statements,
 format characteristics 55T
 IN parameters, type mapping
 example 1080

 INOUT parameters, type mapping
 example 1081
 sending to a database 390
 Statement interface 951– 995
 type mapping example 1080
type identifiers 326
types, mapping 1065– 1095
warnings, SQLWarning class 945– 950

SQL ARRAY

inserting into a table 157T

SQL REF

inserting into a table 157T

SQL state fields

sqlStateSQL99 field
 (DatabaseMetaData interface) 559
sqlStateXOpen field
 (DatabaseMetaData interface) 559

SQL structured type

See also UDT

adding instances to table of referenceable
instances 1001

as array element, custom mapping of 998
attributes
 custom mapping of 998
 defining 999
 ordering of 1005
creating 999
custom mapping of 997
definition and uses 997
inheritance 1003
inserting into a table 157T
storing referenceable instances of 1000

SQL92

as JDBC Compliant requirement, required
extensions 1102
definition, glossary 1171
entry level definition, glossary 1171
intermediate level
 definition, glossary 1172

SQL99

definition, glossary 1172

SQL99 data types

ARRAY 283, 1034
BLOB 309, 1034
BOOLEAN 1035, 1078
CLOB 373, 1034
DATALINK 1035, 1078
DISTINCT 141T, 146T, 1034
functionality, testing of code examples for
 115T
inserting into a table 156T
JAVA_OBJECT 1034
listed and described 1137
locators 1140
mapping for 140T, 1139
materializing data of BLOB, CLOB, and
 ARRAY values on the client 143T
methods for operating on 142T
performance advantages of BLOB, CLOB,
 and ARRAY 143T
REF 149T, 1034
STRUCT 1034
structured type 144T
summary of support for 1138
using 140T– 162T

**SQLData interface**

- implementation by a tool 895
- implementation possibilities 896
- methods
 - getSQLTypeName 898, 904
 - readSQL 899, 905
 - writeSQL 899, 905

SQLData object

- creating 162T
- custom mapping of UDTs, used for 895
- implementation of, example 163T

SQLException class 910

- aborting transactions upon receipt of 76T
- components 907
- components of 82T
- constructors 910, 911
- getErrorCode method 912
 - retrieving SQLException error code with 909
- getMessage method, retrieving
 - SQLException information with 909
- getNextException method 913
 - retrieving multiple exceptions with 582
 - retrieving next SQLException object with 909
- getSQLState method 912
 - retrieving SQLException SQLState with 909
- handling 81T
- indicating unsupported functionality with 1114
- meaning of 909
- methods 912
- object description 907
- reasons for throwing 907
- reported on data truncation while reading 581
- setNextException method 913
- throwing when a given form of data is not available 451

SQLInput interface

- as input stream for custom mapping of UDTs 915
- methods
 - readArray 918
 - readAsciiStream 918
 - readBigDecimal 919

readBinaryStream 919

- readBlob 919
- readBoolean 920
- readByte 920
- readBytes 920
- readCharacterStream 921
- readClob 921
- readDate 922
- readDouble 922
- readFloat 922
- readInt 923
- readLong 923
- readObject 923
- readRef 924
- readShort 924
- readString 925
- readTime 925
- readTimestamp 926
- readURL 926
- wasNull 926

reader methods, use of 916

SQLInput object

- created by driver 915
- how and when invoked 916
- ordering of data in 916

SQLJ

definition

- glossary 1172
- in context 107T

sqlj.install_jar

- SQLJ procedure, built-in 108T

SQLOutput interface

- as output stream for writing custom mapped UDTs 929
- implemented by driver writers 929
- methods
 - writeArray 932
 - writeAsciiStream 933
 - writeBigDecimal 933
 - writeBinaryStream 933
 - writeBlob 934
 - writeBoolean 934
 - writeByte 934
 - writeBytes 935
 - writeCharacterStream 935
 - writeClob 935
 - writeDate 936
 - writeDouble 936



writeFloat 936
writeInt 937
writeLong 937
writeObject 937
writeRef 938
writeShort 938
writeString 938
writeStruct 939
writeTime 939
writeTimestamp 939
writeURL 940

SQLOutput object
created by driver 930
order of attributes in output stream 931

SQLPermission class
class definition 942
SQLPermission constructor 942, 943

SQLPermission object
creating 942
purpose of 941
when needed 621, 941

SQLState
as SQLException component, meaning of 907
as SQLWarning component, meaning of 946
as X/OPEN code,
for errors, SQLException component 82T
retrieving,
with getSQLState method
(SQLException class) 909
with getSQLState method
(SQLWarning class) 946

SQLStatement.java
as Sample Code 9 196T
source code for 196T

SQLWarning class
as SQLException subclass 85T
constructors 947, 948, 949
getErrorCode method, retrieving
SQLWarning error code with 946
getMessage method, retrieving
SQLWarning information with 946
getNextWarning method 950
getSQLState method, retrieving
SQLWarning SQLState with 946
methods 949

reported on data truncation while reading 581
setNextWarning method 950

star (*)
importing all classes in a package 80T
selecting all columns with 59T

start method
applet requirements 101T

state
SQL, as SQLException component 82T

Statement interface definition
addBatch method 973
as one of three classes for sending SQL statements to a database 390
cancel method 973
clearBatch method 973
clearWarnings method 974
close method 974
creating objects 57T
execute method
execute method 975, 976, 977
handling multiple or optional result sets with 723, 965
procedures for using 965
purpose for 965
executeBatch method 978
executeQuery method 979
purpose of 953

executeUpdate method 979, 980
purpose of 953
getConnection method 983
getFetchDirection method 983
getFetchSize method 984
getMaxFieldSize method 985
getMaxRows method 985
getMoreResults method 985, 986
getQueryTimeout method 987
getResultSet method 987
getResultSetConcurrency method 988
getResultSetType method 989
getUpdateCount method 989
getWarnings method 990
methods 973

PreparedStatement
as subclass of 647
compared with 67T



PreparedStatement and
 CallableStatement interfaces
 compared with 953

setCursorName method 991

setEscapeProcessing method 992
 turning escape processing on and off
 with 961

setFetchDirection method 992

setFetchSize method 993

setMaxFieldSize method 994

setMaxRows method 994

setQueryTimeout method 994

updating tables with 64T

warnings reported on 84T

statement pooling
 See pooled statement

statements
 CallableStatement interface 321–371
 completion of 954
 creating 322, 391, 648, 952
 CallableStatement objects 322
 PreparedStatement objects 648
 Statement objects 952
 executing 57T
 format characteristics 56T
 JDBC, creating 57T
 PreparedStatement interface 647–677
 Statement interface 951–996

static
 methods
 DriverManager methods declared as
 614
 glossary definition 1172
 section, required in driver implementations
 1102
 static keyword, class methods identified
 by 81T

stored procedures
 as inconsistently supported SQL
 functionality 21
 as Java static methods 107T
 CallableStatement object execution of
 391
 calling 78T
 calling with a **CallableStatement** object
 321
 characteristics and use 77T

creating 77T
creating, using SQLJ and JDBC API
 107T–111T

definition
 glossary 1172
 term in context 45, 77T

escape clause for invoking 958

executing 79T

getting information about parameters 331

getting information about what is
 supported 322

identifying parameters by name 323

invoking, syntax for 321

JAR file, stored as when using SQLJ and
 JDBC API 108T

parameter numbering 328

passing in IN parameters 324

result sets 197T

retrieving OUT parameters 329

retrieving results before OUT parameters
 330

See also **DatabaseMetaData** stored
 procedures

setting INOUT parameters 329

SuppliersProcs as example code 107T

using to make batch updates 325

storesLowerCaseIdentifiers method
 (**DatabaseMetaData** interface) 525

storesLowerCaseQuotedIdentifiers
 method (**DatabaseMetaData** interface)
 525

storesMixedCaseIdentifiers method
 (**DatabaseMetaData** interface) 525

storesMixedCaseQuotedIdentifiers
 method (**DatabaseMetaData** interface)
 526

storesUpperCaseIdentifiers method
 (**DatabaseMetaData** interface) 526

storesUpperCaseQuotedIdentifiers
 method (**DatabaseMetaData** interface)
 526

streams
 input, sending very large amounts of data
 using 653
 retrieving very large amounts of data using
 720

**strings**

- comparing, in SELECT statements 40
- concatenating 56T
- functions, support for, as implementation requirement 1105tb
- Java length restrictions 56T
- quoted, case sensitivity of 56T
- String* class,
 - conversion by `setObject` to JDBC types 1091tb
 - mapping to JDBC types 1090tb
- String* Java type, mapping to, JDBC type 1088tb

STRUCT field (*Types* class)

- class definition of 1036
- description 1076
- mapping to Java object types 1089tb
- mapping to Java type 1087tb

Struct interface

- methods
 - `getAttributes` method 1008
 - `getSQLTypeName` method 1009
- standard mapping for SQL structured type 997

Struct object

- contents 1003
- creating 1003
- inheritance 1004
- mapping by `setObject` method to JDBC types 1091tb
- mapping from Java object types to JDBC types 1090tb
- ordering of attributes 1004, 1005
- retrieving
 - `CallableStatement.getObject` method 1004
 - `ResultSet.getObject` method 1003

- standard mapping to JDBC types 1088tb
- storing 1007

structured type (SQL)

- attributes of 144T
- creating 680
- creating instances of 150T
- definition, glossary 1172
- referenceable instances, table of 681

subclass

- definition, term in context 31

subcontext

- `jdbc/pool` used for `ConnectionPoolDataSource` objects 173T
- `jdbc/xa`, for `XADatasource` objects 178T
- use 170T

subcontext (JNDI)

- definition 569
- `jdbc` reserved for logical names of JDBC data sources 569

subprotocols

- JDBC driver, determining which to use 53T
- JDBC URL, syntax and use 388
- `odbc` 388
- registering names of 390

**SUCCESS_NO_INFO field (*Statement* interface)
996**

- SuppliersProcs.java**
 - example code for stored procedure using SQLJ and JDBC API 107T

supplying

- values for tables 71T

support testing

- See `DatabaseMetaData` methods

**supportsAlterTableWithAddColumn method
(*DatabaseMetaData* interface) 527****supportsAlterTableWithDropColumn
method (*DatabaseMetaData* interface)
527****supportsANSI92EntryLevelSQL method
(*DatabaseMetaData* interface) 527****supportsANSI92FullSQL method
(*DatabaseMetaData* interface) 528****supportsANSI92IntermediateSQL method
(*DatabaseMetaData* interface) 528****supportsBatchUpdates method
(*DatabaseMetaData* interface) 528****supportsCatalogsInDataManipulation
method (*DatabaseMetaData* interface)
529****supportsCatalogsInIndexDefinitions
method (*DatabaseMetaData* interface)
529****supportsCatalogsInPrivilegeDefinition
s method (*DatabaseMetaData* interface)
529**



supportsCatalogsInProcedureCalls method (*DatabaseMetaData interface*) 530
supportsCatalogsInTableDefinitions method (*DatabaseMetaData interface*) 530
supportsColumnAliasing method (*DatabaseMetaData interface*) 530
supportsConvert method (*DatabaseMetaData interface*) 531
supportsCoreSQLGrammar method (*DatabaseMetaData interface*) 532
supportsCorrelatedSubqueries method (*DatabaseMetaData interface*) 532
supportsDataDefinitionAndDataManipulationTransactions method (*DatabaseMetaData interface*) 532
supportsDataManipulationTransactionsOnly method (*DatabaseMetaData interface*) 533
supportsDifferentTableCorrelationNames method (*DatabaseMetaData interface*) 533
supportsExpressionsInOrderBy method (*DatabaseMetaData interface*) 533
supportsExtendedSQLGrammar method (*DatabaseMetaData interface*) 534
supportsFullOuterJoins method (*DatabaseMetaData interface*) 534, 543
supportsGetGeneratedKeys 534
supportsGetGeneratedKeys method (*DatabaseMetaData interface*) 534
supportsGroupBy method (*DatabaseMetaData interface*) 534
supportsGroupByBeyondSelect method (*DatabaseMetaData interface*) 535
supportsGroupByUnrelated method (*DatabaseMetaData interface*) 535
supportsIntegrityEnhancementFacility method (*DatabaseMetaData interface*) 535
supportsLikeEscapeClause method (*DatabaseMetaData interface*) 536
supportsLimitedOuterJoins method (*DatabaseMetaData interface*) 536
supportsMinimumSQLGrammar method (*DatabaseMetaData interface*) 536
supportsMixedCaseIdentifiers method (*DatabaseMetaData interface*) 537
supportsMixedCaseQuotedIdentifiers method (*DatabaseMetaData interface*) 537
supportsMultipleOpenResults 537
supportsMultipleOpenResults method (*DatabaseMetaData interface*) 537
supportsMultipleTransactions method (*DatabaseMetaData interface*) 538
supportsNamedParameters 538
supportsNamedParameters method (*DatabaseMetaData interface*) 538
supportsNonNullableColumns method (*DatabaseMetaData interface*) 539
supportsOpenCursorsAcrossCommit method (*DatabaseMetaData interface*) 539
supportsOpenCursorsAcrossRollback method (*DatabaseMetaData interface*) 539
supportsOpenStatementsAcrossCommit method (*DatabaseMetaData interface*) 540
supportsOpenStatementsAcrossRollback method (*DatabaseMetaData interface*) 540
supportsOrderByUnrelated method (*DatabaseMetaData interface*) 541
supportsOuterJoins method (*DatabaseMetaData interface*) 541
supportsPositionedDelete method (*DatabaseMetaData interface*) 541
supportsPositionedUpdate method (*DatabaseMetaData interface*) 542
supportsResultSetConcurrency method (*DatabaseMetaData interface*) 542
supportsResultSetHoldability 543
supportsResultSetHoldability method (*DatabaseMetaData interface*) 543
supportsSavepoints 544
supportsSavepoints method (*DatabaseMetaData interface*) 544
supportsSchemasInDataManipulation method (*DatabaseMetaData interface*) 544
supportsSchemasInIndexDefinitions method (*DatabaseMetaData interface*) 544



supportsSchemasInPrivilegeDefinitions
 method (*DatabaseMetaData interface*)
 545

supportsSchemasInProcedureCalls method
 (*DatabaseMetaData interface*) **545**

supportsSchemasInTableDefinitions
 method (*DatabaseMetaData interface*)
 545

supportsSelectForUpdate method
 (*DatabaseMetaData interface*) **546**

supportsStatementPooling **546**

supportsStatementPooling method
 (*DatabaseMetaData interface*) **546**

supportsStoredProcedures method
 (*DatabaseMetaData interface*) **546**

supportsSubqueriesInComparisons method
 (*DatabaseMetaData interface*) **547**

supportsSubqueriesInExists method
 (*DatabaseMetaData interface*) **547**

supportsSubqueriesInIns method
 (*DatabaseMetaData interface*) **547**

supportsSubqueriesInQuantifieds method
 (*DatabaseMetaData interface*) **548**

supportsTableCorrelationNames method
 (*DatabaseMetaData interface*) **548**

supportsTransactionIsolationLevel
 method (*DatabaseMetaData interface*)
 548

supportsTransactions method
 (*DatabaseMetaData interface*) **549**

supportsUnion method (*DatabaseMetaData interface*) **549**

supportsUnionAll method
 (*DatabaseMetaData interface*) **550**

synchronization
 definition
 glossary **1172**
 term in context **101T**
 of transactions **393**

syntax
 See also notation
 curly braces ({ }) **29**
 escape characters and,
 CallableStatement use **321**
 glossary definition **1164**
 JDBC method of handling
 inconsistencies **21**

Statement use **958**
stored procedures **78T**
invoking a stored procedure **321**
URLs, JDBC **388**

system
functions, support for, as implementation
 requirement **1107tb**

system administrator
deployer for XDataSource objects **1057**

T

t keyword **960**
 See also keywords

T, use of in index **6**

table
referenceable instances of an SQL
 structured type, creating **681, 1000**

tables
creating **53T**
 executeUpdate method use for **57T**
 sample code for **87T, 91T**
deleting, **DROP TABLE** use for **44**
inserting values in, sample code for **87T,**
 91T
integrity rules in **38**
joining **71T**
lock, definition, term in context **45**
making permanent changes, with **commit**
 method **74T**
manipulation statements, executing **58T**
populating **58T**
 sample code for **87T, 91T**
selecting, an entire **59T**
setting **53T**

tableIndexClustered field
 (*DatabaseMetaData interface*) **560**

tableIndexHashed field
 (*DatabaseMetaData interface*) **560**

tableIndexOther field
 (*DatabaseMetaData interface*) **560**

tableIndexStatistic field
 (*DatabaseMetaData interface*) **560**

TableTypes.java,
 as Sample Code **13 213**
 source code for **213**



update, return values 70T
updating 64T

tb, use of in index 6

termination
transaction, requiring explicitly with the
disabling of auto-commit mode 392

terminator
DBMS statement, not needed in JDBC
55T

testing
for case sensitivity, `DatabaseMetaData`
methods 452

threads
See also multithreading
definition, term in context 100T

three-tier
database access model, advantages 19
JDBC driver, glossary definition 1172

three-tier architecture
connection pooling, in 639
scenario for 169T

time
constructing with a `Calendar` object 1011
escape clause for 960
exception thrown if `Date` object used for
592
functions, support for, as implementation
requirement 1106tb
`Time` class definition 1011
constructor 1015
conversion by `setObject` to JDBC
types 1091tb
mapping as Java object type to, JDBC
types 1090tb
mapping to
database-specific SQL types
1095tb
Java object type 1089tb
JDBC type 1088tb
mapping to Java type 1087tb
methods 1017
`toString` method 1018
`valueOf` method 1018
zero epoch 1012

TIME field (Types class)
characteristics and comparison with
standard SQL and Java types 1073

description 1073

TIME field (Types class)
class definition of 1036

Time object
creating 1011
deprecated methods 1013
mapping by `setObject` method to JDBC
types 1091tb
retrieving 1013

time zone
calculating a date using 593
`Calendar` object, setting in 593
default 593
using to construct
a `Date` object 593
a `Time` object 1014
a `Timestamp` object 1024

Timestamp class
`after` method 1027
`before` method 1028
`equals` method 1028
`getNanos` method 1029
`setNanos` method 1029
`Timestamp` constructor 1026
`Timestamp` constructor (deprecated)
1025
`toString` method 1030
`valueOf` method 1030

TIMESTAMP field (Types class)
characteristics and comparison with
standard SQL and Java types 1073

class definition of 1036

description 1073

mapping to
database-specific SQL types 1095tb
Java object type 1089tb
Java type 1087tb

Timestamp object
converting to a `java.util.Date` object
1074

creating 1021

mapping by `setObject` method to JDBC
types 1091tb

mapping for `TIMESTAMP` value 1021

mapping to JDBC type 1088tb

object mapping to JDBC type 1090tb

retrieving 1023

using time zone to construct 1024

**timestamps**

escape clause for 960
TIMESTAMP statement, representing with
Timestamp class 960

TINYINT field (Types class)

See also types
characteristics and comparison with
standard SQL and Java types 1070
class definition of 1036
description 1070
mapping to
database-specific SQL types 1094tb
Java object type 1089tb
Java type 1087tb
type mapping example 329

tools

GUI, accessing needed information for
603

toString method

Date class 590, 596
Time class 1018
Timestamp class 1030
See also strings, types

tracing and logging

DataSource object 574
DriverManager class 611

transaction

definition, glossary 1172

transaction attributes

EJB component 1040
provided by EJB component vendor 1051
TX_BEAN_MANAGED 1051
TX_MANDATORY 1052
TX_NOT_SUPPORTED 1051
TX_REQUIRED 1051
TX_REQUIRES_NEW 1051
TX_SUPPORTS 1052
use by EJB components 269T

transaction branch

definition in context 1058

transaction isolation level

See transactions, isolation levels

transaction manager

committing or rolling back a distributed
transaction 573
role in EJB applications 1040

use of javax.transaction.xa.

XAResource object 1044
use of two-phase commit protocol 1045

TRANSACTION_NONE (Connection interface)

428

**TRANSACTION_READ_COMMITTED (Connection
interface)** 715**TRANSACTION_READ_UNCOMMITTED
(Connection interface)** 715**TRANSACTION_REPEATABLE_READ
(Connection interface)** 429**TRANSACTION_SERIALIZABLE (Connection
interface)** 716**transactions**

aborting, with rollback method
(Connection interface) 76T

committing,
disabling auto-commit mode 73T

creating 74T

definition,
term in context 44, 73, 392

distributed

See distributed transactions

initiation of, implicit 393

isolation levels

as conflict management mechanism
393

definition, term in context 75T

determining 75T

lock setting determined by 75T

setting 76T

TRANSACTION_NONE 428

TRANSACTION_READ_COMMITTED 428

TRANSACTION_READ_UNCOMMITTED
394, 428

TRANSACTION_REPEATABLE_READ
429

TRANSACTION_SERIALIZABLE 429

use by RowSet object 260T

sustaining, disabling auto-commit mode as
means of 392

TransactionPairs.java,

as Sample Code 6 96T

source code for 96T

visibility of changes to data 715

truncate method (Blob interface) 319**truncate method (Clob interface)** 383

**truncation**

contents of DataTruncation objects 582
of data, handling 85T, 581
of input, throwing exceptions, as
implementation requirement for
drivers 1109

try block

See also exceptions
exception handling with 81T

tuple

definition, glossary 1173

tutorials

advanced 113T– 192T
basic 49T– 111T
<http://java.sun.com/products/jdbc/book.html>, as sample code
location 86T
metadata 193T– 254T
rowset 255T– 280T
Web location of code for 51T

two-phase commit protocol

definition in context 1045
steps in 1045

two-tier

database access model 19
JDBC driver
glossary definition 1173
JDBC driver,
advantages of 604

type map

adding an entry 899
as an argument passed to a method 397
as argument to
 Array.getArray method 286, 398
 Array.getResultSet method 287
associated with a Connection object 396
creating 397
entry
 contents of 396
 two ways to handle 286
methods that take as an argument 900
setting a new one with the
 Connection.setTypeMap method
 397
using different ones to map the same UDT
 398

TYPE_FORWARD_ONLY field (ResultSet interface) 700**TYPE_SCROLL_INSENSITIVE field (ResultSet interface) 700****TYPE_SCROLL_SENSITIVE field (ResultSet interface) 701****TypeInfo.java**

as Sample Code 14 215T
source code for 215T

types

Java, specification in setter method names
650

JDBC types defined in 56T

JDBC,
 implicitly specified in a setter method
 name 650
NULL field (Types class) 322
 conversion of 330
 sending as an IN parameter 653
 testing for 722

mapping,

 SQL and Java 1065– 1095
 tables 1085– 1095

strong, as Java feature supporting correct
code development 33

table creation use of 43

typeNulls field (DatabaseMetaData interface) 561

typeNullable field (DatabaseMetaData interface) 561

typeNullableUnknown field

 (DatabaseMetaData interface) 561

typePredBasic field

 (DatabaseMetaData interface) 561

typePredChar field (DatabaseMetaData interface) 562

typePredNone field (DatabaseMetaData interface) 562

Types class definition 1035

 ARRAY field 1036

 BIGINT field 1036

 BINARY field 1036

 BIT field 1036

 BLOB field 1036

 BOOLEAN field 1037

 CHAR field 1036

 CLOB field 1037

 DATALINK field 1037

 DATE field 1036

 DECIMAL field 1036



DISTINCT field 1036
DOUBLE field 1036
FLOAT field 1036
INTEGER field 1036
JAVA_OBJECT field 1036
LONGVARBINARY field 1036
LONGVARCHAR field 1036
NULL field 1036
NUMERIC field 1036
OTHER field 1036
REAL field 1036
REF field 1037
SMALLINT field 1036
STRUCT field 1036
TIME field 1036
TIMESTAMP field 1036
TINYINT field 1036
VARBINARY field 1036
VARCHAR field 1036
typeSearchable field
(*DatabaseMetaData* interface) 562

U

ubiquitous

as Java characteristic 28

UDT

code for creating when DBMS uses
DBMS-specific data types 244T
custom mapping of
SQLData implementation required for
895
definition, glossary 1173
distinct type 599
methods for getting information about
get Attributes 469
get SuperTables 505
getSuperTypes 506
getUDTs 514

underbar (_) wildcard 41, 959

underlying type

definition, glossary 1173

undoing

table changes, with **rollback** method
75T

Unicode character set

data,
retrieving very large amounts of data
using **getUnicodeStream**
method (*ResultSet* interface)
721
sending very large amounts of data
using **setCharacterStream**
method (*PreparedStatement*
interface) 653
definition, glossary 1173

UNIX operating system

See also **operating systems**
appletviewer use 102T
running sample code 87T

updatable result sets

creating with **createStatement** method
(*Connection* interface) 122T
creating with *PreparedStatement* object
648
queries that produce 714

update capabilities

determining with
supportsBatchUpdates method
(*DatabaseMetaData* interface) 453

updater methods

effect on values in insert row as opposed to
values on current row of a result set
127T
updateArray method (*ResultSet*
interface) 763
updateAsciiStream method (*ResultSet*
interface) 764
updateBigDecimal method (*ResultSet*
interface) 765
updateBinaryStream method
(*ResultSet* interface) 765
updateBlob method (*ResultSet*
interface) 766
updateBoolean method (*ResultSet*
interface) 766
updateByte method (*ResultSet*
interface) 767
updateBytes method (*ResultSet*
interface) 768
updateCharacterStream method
(*ResultSet* interface) 768



updateClob method (*ResultSet interface*) 769
updateDate method (*ResultSet interface*) 769
updateDouble method (*ResultSet interface*) 770
updateFloat method (*ResultSet interface*) 771
updateInt method (*ResultSet interface*) 771
updateLong method (*ResultSet interface*) 772
updateNull method (*ResultSet interface*) 772
updateObject method (*ResultSet interface*) 773, 774
updateRef method (*ResultSet interface*) 774
updateRow method (*ResultSet interface*) 710, 775
updateShort method (*ResultSet interface*) 775
updateString method (*ResultSet interface*) 776
updateTime method (*ResultSet interface*) 776
updateTimestamp method (*ResultSet interface*) 777
using 208T, 709

updatesAreDetected method
(*DatabaseMetaData interface*) 226, 550

updating

- a Blob object 313
- a Clob object 375, 377
- a column value programmatically 123T
- canceling with *cancelRowUpdates* method (*ResultSet interface*) 124T
- locks as implementation requirement for drivers 110T
- permanent, *commit* method with 74T
- positioned,
 - definition, term in context 713
- tables 64T
 - PreparedStatement* advantages for 68T
 - return value as result of 70T

UPDATE statement,
DML command, purpose 43
executeUpdate method (*Statement interface*) use 953
values in the database 124T

URL (Uniform Resource Locator)s
See also applets; Internet
database connection established through 385
definition, glossary 1173
<http://java.sun.com/products/jdbc/> 387
as JDBC products URL 22
driver information available from 25
<http://java.sun.com/products/jdbc/book.html> 51T, 85T, 86T
<http://java.sun.com/products/JDK/CurrentRelease> 50T
JDBC, syntax and use 386
specifying when connecting to a DBMS 52T

URL object
mapping from Java object types to JDBC types 1090tb
standard mapping to JDBC types 1088tb

user interface
GUI, accessing needed information for 603
high-level, plans for 15

user property (*DataSource interface*) 567

usesLocalFilePerTable method
(*DatabaseMetaData interface*) 551

usesLocalFiles method (*DatabaseMetaData interface*) 551

V

valueOf method

- Date class 591, 596
- Time class 1013, 1018
- Timestamp class 1030

values

- column, retrieving from a result set 696
- date, representing 589



inserting into tables, sample code for 87T, 91T
modifying, UPDATE use for 43
of parameters, setting with setter methods 648
OUT parameters, retrieving 330
PreparedStatement parameters, setting with loops 69T
supplying with setter methods 67T
ResultSet, JDBC types as 1033
return,
 for executeUpdate method 70T
 isNullable method
 (ResultSetMetaData interface)
 783
setting for databases,
 PreparedStatement methods for 647
value field (DriverPropertyInfo class) 626

VARBINARY field (Types class)
characteristics and comparison with standard SQL and Java types 1069
class definition of 1036
description 1069
getInt method not usable for retrieving 63T
mapping to
 database-specific SQL types 1094tb
 Java object type 1089tb
 Java type 1087tb

VARCHAR field (Types class)
characteristics and comparison with standard SQL and Java types 1067
class definition of 1036
description 1067
getString method use for retrieving 63T
mapping to
 database-specific SQL types 1094tb
 Java object type 1089tb
 Java type 1087tb

variables
assigning values to 56T

variants
data type, handling 20
permitted 1111

Vector class (java.util package)
applet use of 100T

verification
of incoming code, as Java security feature 36

versionColumnNotPseudo field
(DatabaseMetaData interface) 562

versionColumnPseudo field
(DatabaseMetaData interface) 563

versionColumnUnknown field
(DatabaseMetaData interface) 563

Virtual Machine
as Java platform component 28
as source of Java portability 28
code verification by, as Java security feature 36

visibility
importing classes 80T

visibility of changes in result sets
after closing and reopening the result set 131T
differences between sensitive and insensitive result sets 117T
effect of transaction isolation level 131T
made by others while result set open 131T
methods for detecting 453

visibility of updates
support for, determining with DatabaseMetaData methods 226T

W

warnings

chaining
on SQLWarning objects 945
through DataTruncation objects 582
definition, glossary 1173
retrieving 65T, 83T
transitory nature of 84T

wasNull method

ResultSet interface
testing for JDBC NULL value with 722
testing for OUT parameter JDBC NULL value 330

**WebRowSet class**

as example implementation of RowSet interface 256T, 818

WHERE clause (SELECT statement)

conceptual overview 40
limiting selections with 65T

while loop

catching all possible exceptions with 82T
used with next method (ResultSet interface) 62T, 65T, 758

wildcards 40

in DatabaseMetaData method arguments 451

Windows operating system

appletviewer use 102T
running sample code 87T

writeData method (RowSetWriter interface)

method explanation 887
possible implementations 886
use of optimistic concurrency algorithm as possible implementation 817
use with disconnected RowSet object 885

writer

definition in context 885
definition, glossary 1173
for RowSet object 265T, 798
implemented by 885

writer methods (SQLOutput interface)

called by SQLOutput.writeSQL method 930
conversion from Java type to SQL type, done by 930
writeArray 932
writeAsciiStream 933
writeBigDecimal 933
writeBinaryStream 933
writeBlob 934
writeBoolean 934
writeByte 934
writeBytes 935
writeCharacterStream 935
writeClob 935
writeDate 936
writeDouble 936
writeFloat 936
writeInt 937
writeLong 937

writeObject 937

writeRef 938

writeShort 938

writeString 938

writeStruct 939

writeTime 939

writeTimestamp 939

writeURL 940

writeSQL method (SQLData interface) 905**writing**

data, handling problems with 85T

X

X/Open

See also Open Group

SQL CLI (Call Level Interface)
as basis for JDBC and ODBC 18

definition, glossary 1168

SQLState conventions, for errors,
SQLException component 82T

XACnection interface

getXAResource method 1053

XACnection object

as pooled connection 1043
auto-commit mode disabled by default 1040

connection for distributed transactions 1039
creating 1042

derived from PooledConnection 1039

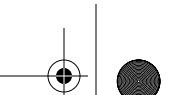
EJB component, importance of use in 1039

XADataSource interface

getLoginTimeout method 1059
getLogWriter method 1060
getXAConnection method 1060, 1061
setLoginTimeout method 1062
setLogWriter method 1062

XADataSource object

as factory for XACnection objects 1055
binding logical name with 1056
compared to DataSource and ConnectionPoolDataSource interfaces 1055



DataSource object that references 1057
deployment 1056
deployment example 178T
registering with a JNDI naming service
1056
resource manager it references 1058
retrieving 1057
retrieving with logical name for data
source 1043
setting properties 1056
when needed 1058

XAResource interface definition
See `javax.transaction.xa.`

XAResource interface definition
1047

XAResource object

correct one, using 1046
use by transaction manager 1043

Z

zero (0) return value

from `executeUpdate` method, meaning of
71T

zero epoch 589