JDBC

Prepared by Harish Patnaik

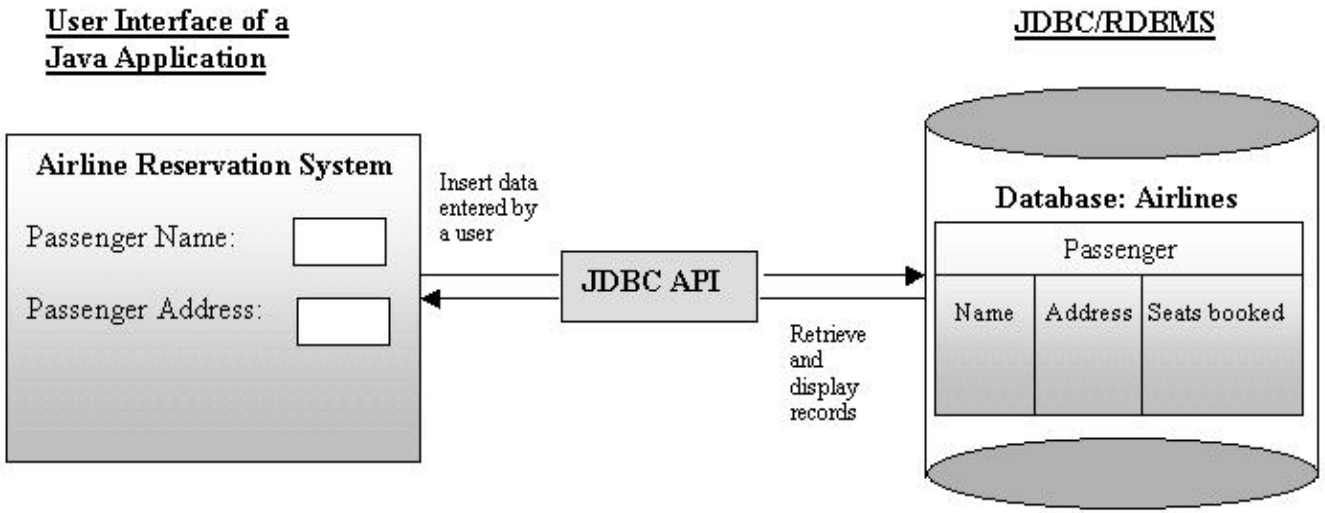
School of Computer Engineering, KIIT Deemed to be University

Content

1. Intro to JDBC
2. Types of driver
3. JDBC Architecture
4. JDBC classes & interfaces
5. Steps in developing JDBC application

Intro to JDBC

- It is an Application Programming Interface (API) developed by JavaSoft
- It includes interfaces & classes for creating connectivity between Java applications and databases
- It supports wide range of databases such as Oracle, MySQL, Sybase, DB2
- It is defined in the package java.sql



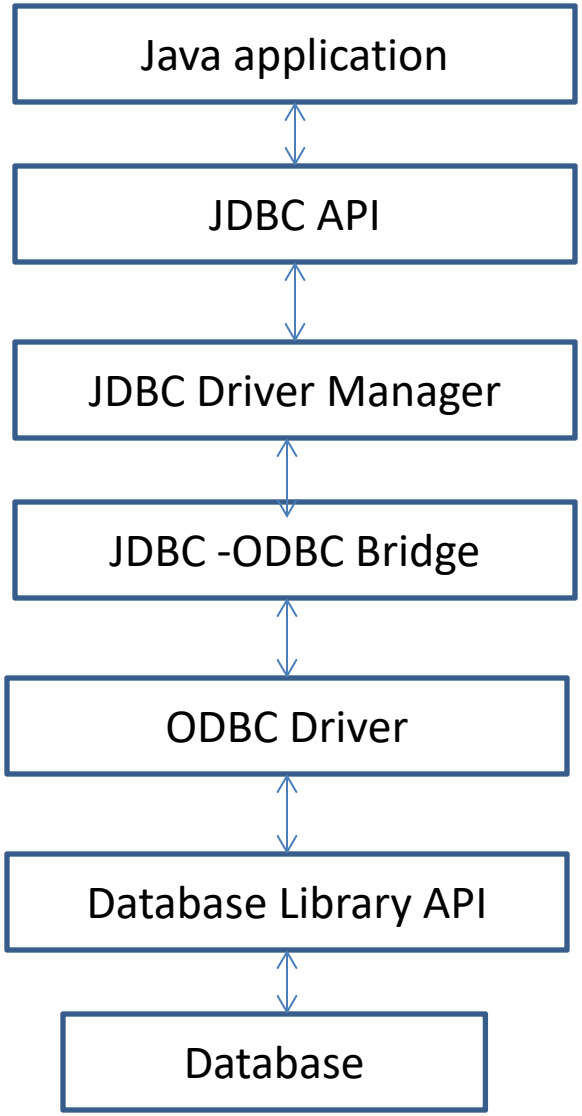
Intro to JDBC cont ..

- Almost all types of RDBMS support SQL
- JDBC helps the Java developers to send SQL queries to databases

- The Driver of a database connects and interacts with the database
- Each database has it’s own driver
- Driver manager locates and loads the required driver for database connectivity
- JVM uses JDBC drivers to translate databse access request to database specific API.

Type - 1 (JDBC -ODBC bridge) driver

It converts JDBC method calls into ODBC function calls. Then it employs ODBC driver to connect to the database.



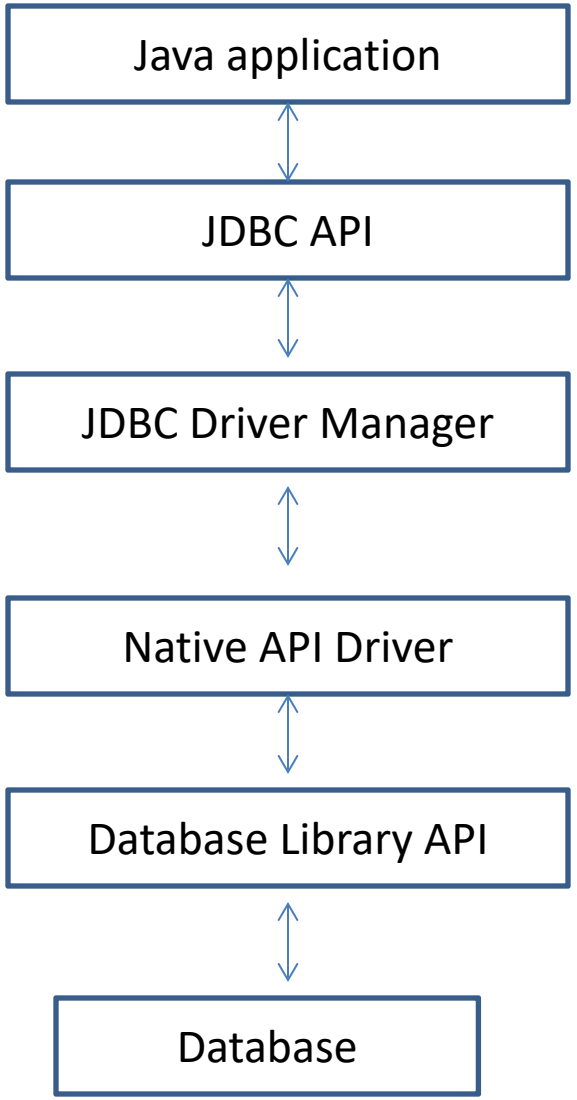
Type 1 driver

- Easy to install

- Disadvantages
  - Application should install ODBC driver
  - All JDBC method calls should be converted to ODBC method calls before they are processed . So the performance degrades.

Type - 2 (Native API) driver

It converts JDBC database calls into vendor specific Native API calls. The database will process the request and send the results back through the API.



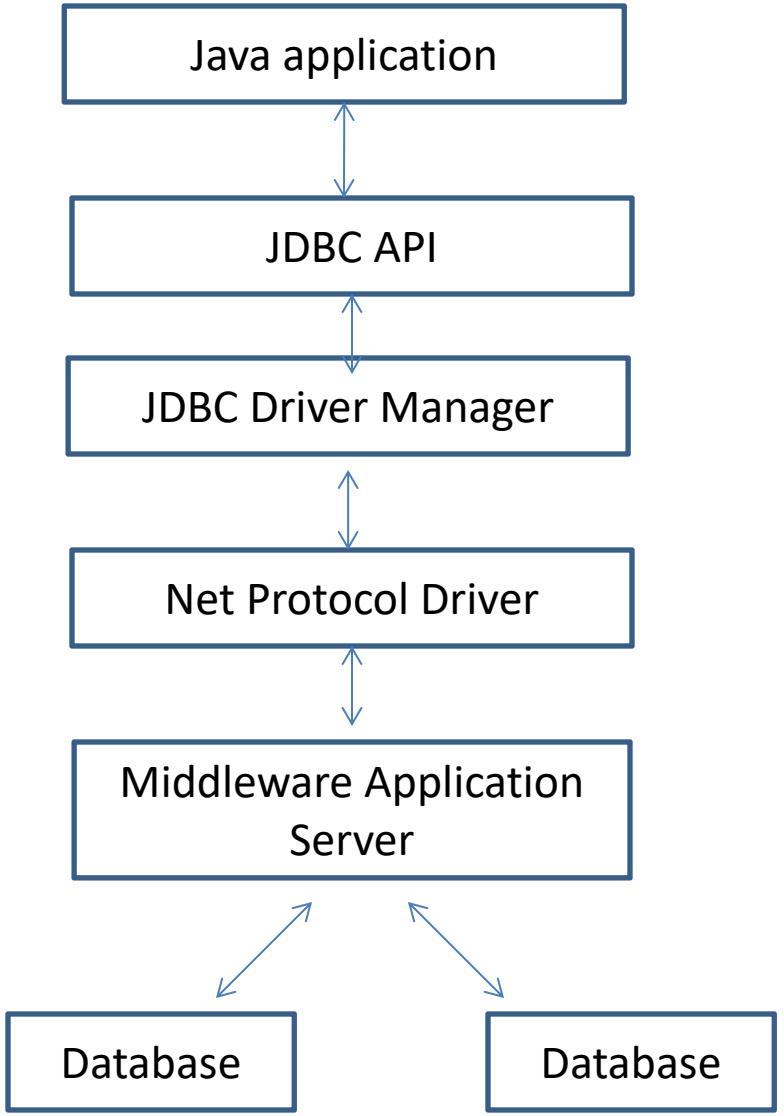
Type 2 driver

- Faster than type 1 because the JDBC method calls are not converted to ODBC method calls

- Disadvantages
  - Application should install Native API of each database
  - Driver is platform dependent

Type - 3 (Net protocol) driver

Java client sends a JDBC call using sockets to the intermediate data access server which completes the request to the database using another driver.



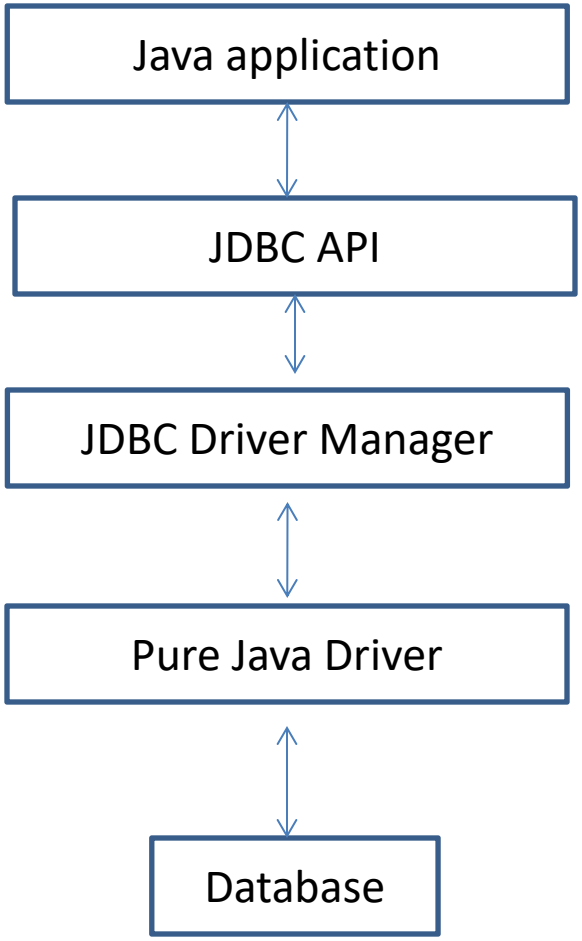
Type 3 driver

- It is platform independent
- Middleware server provides additional security, load balancing

- Disadvantages
  - Required database specific coding need to be done in the middle tier

Type - 4 (Pure Java) driver

It converts the JDBC API calls to direct network calls using vendor specific networking protocols.



Type 4 driver

- It is platform independent as it is written completely in Java
- Performance improved as no Middleware server required

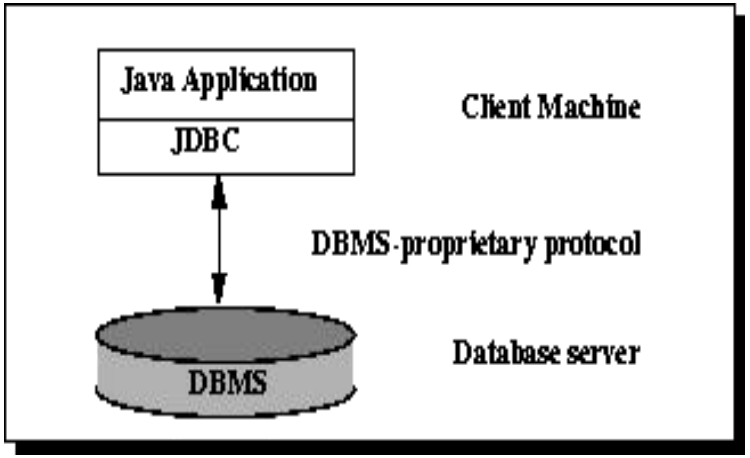
- Disadvantages
  - Separate driver is needed for each database at client side

JDBC Architecture

JDBC API supports both 2-tier and 3-tier processing models for database access.

- 2-tier Architecture -

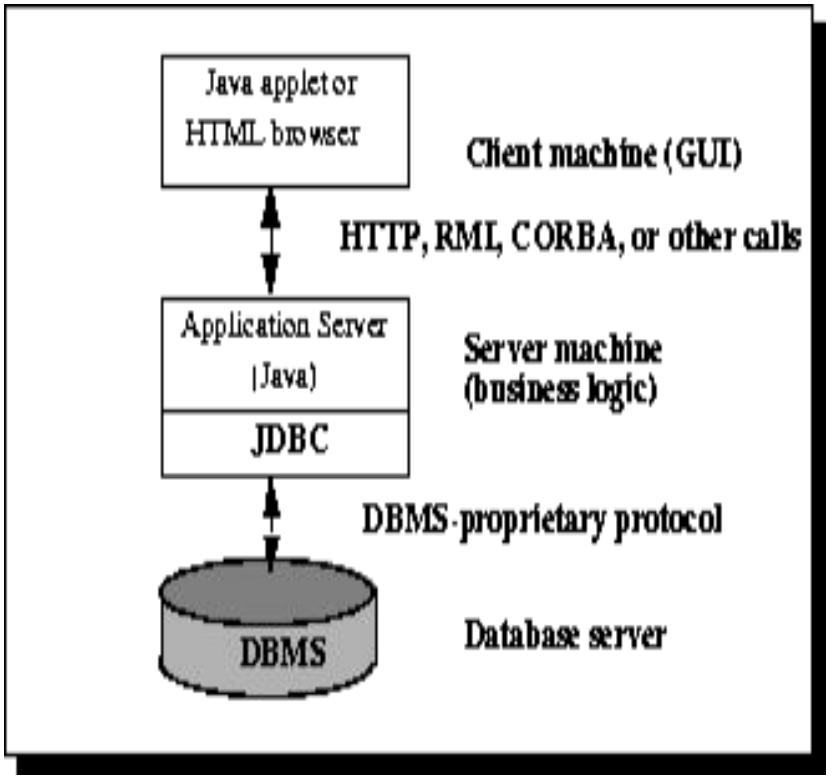
Java application communicates directly with the database.



- 3-tier Architecture

The application communicates with the middle-tier service, from which commands are sent to the database.

The results from database are sent back to the middle tier and from there to the application.



JDBC classes & interfaces

- many classes and interfaces are needed to develop database enabled Java applications.

class / interface	description
Driver Manager	Manages a list of database drivers
Driver	Provides interface to specific database
Connection	All communication happens through this interface
Statement	A static SQL query can be submitted to the database by this object
ResultSet	It holds data retrieved from a database after executing a SQL query
SQL Exception	It handles exceptions that occur during database connection

Steps in developing JDBC application

1. Importing the required package - java.sql
2. Rgistering JDBC driver - Registration is done by using the method Class.forName()
3. Formulate Database URL & establish connection -

   A connection with the database can be established using the statement DriverManager.getConnection() and this method needs a database URL.
4. Creating a statement - A Statement object is created by using createStatement() method
5. Execute a query - to execute a SQL statement the following methods are used -

   boolean execute() - DDL statements

   int executeUpdate() -insert,update,delete statement

   ResultSet executeQuery() - select statement

Example - InsertRecord.java

Database metadata

- DatabaseMetaData interface contains methods to gather metadata of the database.

method	description
getDriverName()	gets the name of the JDBC driver
getDriverVersion()	gets the version no. of the JDBC driver
getUserName()	gets the name of the user
getDatabaseProductName()	gets the name of the database product
getMaxRowSize()	gets the maximum number of bytes the database allows in a single row.
getMaxColumnsInTable()	gets the maximum number of columns the database allows in a table.

Example - metadata.java

Resultset metadata

- ResultsetMetaData interface contains methods to gather metadata of the table.

method	description
getColumnCount()	gets the number of columns in this ResultSet object
getColumnName()	gets the column name of the specified column index
getColumnTypeName()	gets the designated column's database-specific type name
getColumnLabel()	Gets the designated column's title for use in displays.

Example - rsmetadata.java

Thank you