M.Sc. (Five Year Integrated) in Computer Science (Artificial Intelligence & Data Science)

First Semester

Assignment

21-805-0104Computational Thinking

Submitted in partial fulfillment
of the requirements for the award of degree in
Master of Science (Five Year Integrated)
in Computer Science (Artificial Intelligence & Data Science) of
Cochin University of Science and Technology (CUSAT)
Kochi

Submitted by

LANA ANVAR
(80522012)

DEPARTMENT OF COMPUTER SCIENCE
COCHIN UNIVERSITY OF SCIENCE AND TECHNOLOGY (CUSAT)
KOCHI-682022

AUGUST 2023

JAVA DATABASE CONNECTIVITY

Lana Anvar

August 2023

Contents

Introduction To Java Database Connectivity

JDBC (Java Database Connectivity) is the Java API that manages connecting to a database, issuing queries and commands, and handling result sets obtained from the database. Released as part of JDK 1.1 in 1997, JDBC was one of the earliest libraries developed for the Java language.

JDBC was initially conceived as a client-side API, enabling a Java client to interact with a data source. That changed with JDBC 2.0, which included an optional package supporting server-side JDBC connections.

JDBC is used to interact with a database from within a Java program. JDBC acts as a bridge from the code to the database. It is shown in the figure

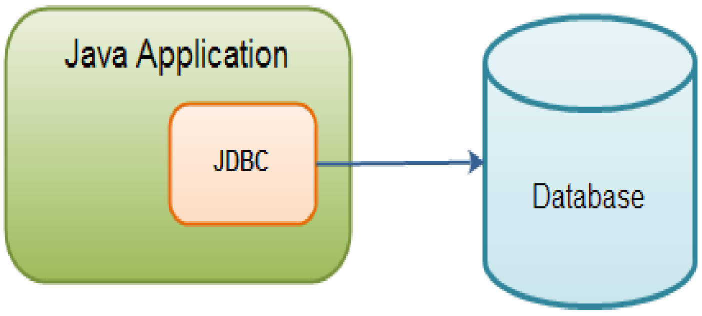

Figure 1: JDBC connects Java programs to databases.

JDBC Architecture

The JDBC interface consists of two layers:

- The JDBC API supports communication between the Java application and the JDBC manager.
- The JDBC driver supports communication between the JDBC manager and the database driver.

The JDBC API and JDBC driver have been refined extensively over the years, resulting in a feature-rich, performant, and reliable library.

JDBC Drivers

JDBC drivers implement the defined interfaces in the JDBC API, for interacting with your database server.

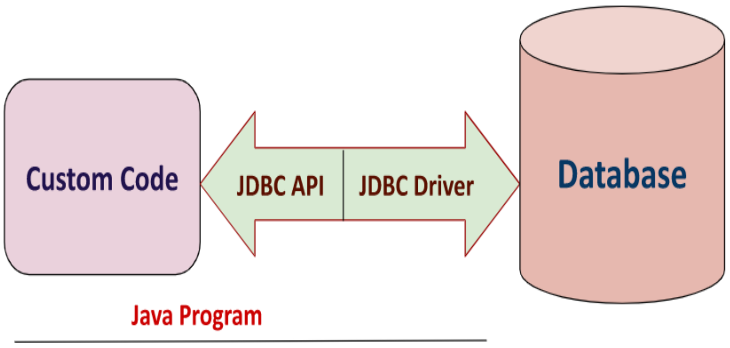

Figure 2: JDBC Architecture.

For example, using JDBC drivers enables to open database connections and to interact with it by sending SQL or database commands then receiving results with Java. The Java.sql package that ships with JDK, contains various classes with their behaviours defined and their actual implementaions are done in third-party drivers. Third party vendors implements the java.sql.Driver interface in their database driver.

There are four JDBC driver types:

Type 1 : JDBC-ODBC Bridge Driver

A JDBC bridge is used to access ODBC drivers installed on each client machine. Using ODBC, requires configuring on your system a Data Source Name (DSN) that represents the target database.

When Java first came out, this was a useful driver because most databases only supported ODBC access but now this type of driver is recommended only for experimental use or when no other alternative is available.
Example : The JDBC-ODBC Bridge that comes with JDK 1.2

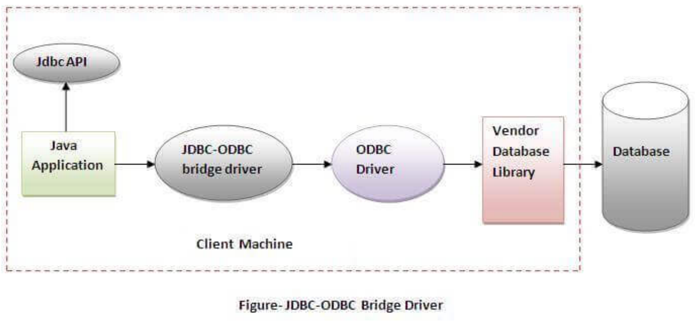

Figure 3: JDBC ODBC Bridge Driver

Type 2 : JDBC-Native API

JDBC API calls are converted into native C/C++ API calls, which are unique to the database. These drivers are typically provided by the database vendors and used in the same manner as the JDBC-ODBC Bridge. The vendor-specific driver must be installed on each client machine.Type 2 driver eliminates ODBC's overhead. Example : The Oracle Call Interface (OCI) driver

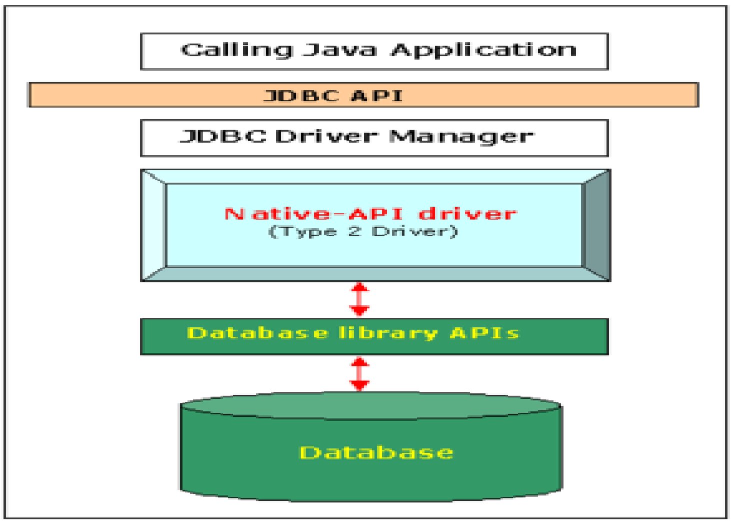

Figure 4: JDBC Native API

Type 3 : JDBC-Net pure Java

A three-tier approach is used to access databases. The JDBC clients use standard network sockets to communicate with a middleware application server. The socket information is then translated by the middleware application server into the call format required by the DBMS, and forwarded to the database server.

This kind of driver is extremely flexible, since it requires no code installed on the client and a single driver can actually provide access to multiple databases.

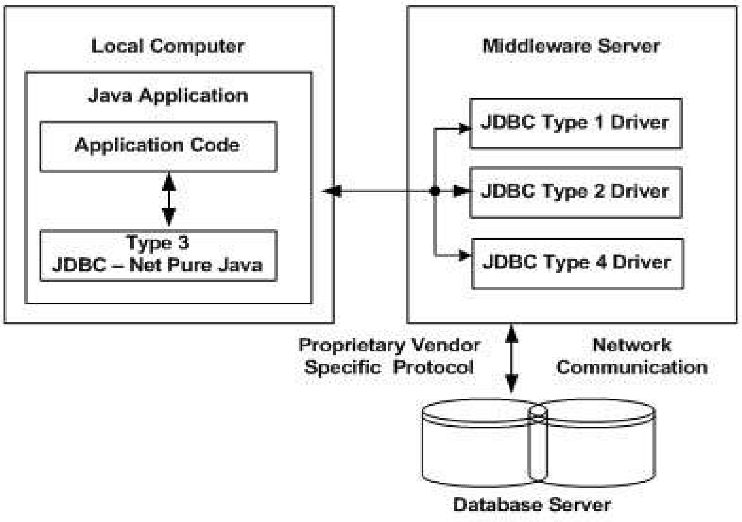

Figure 5: JDBC-Net Pure Java

Type 4 : Pure Java Driver

a pure Java-based driver communicates directly with the vendor's database through socket connection. This is the highest performance driver available for the database and is usually provided by the vendor itself. This kind of driver is extremely flexible, does not need to install special software on the client or server. Further, these drivers can be downloaded dynamically.

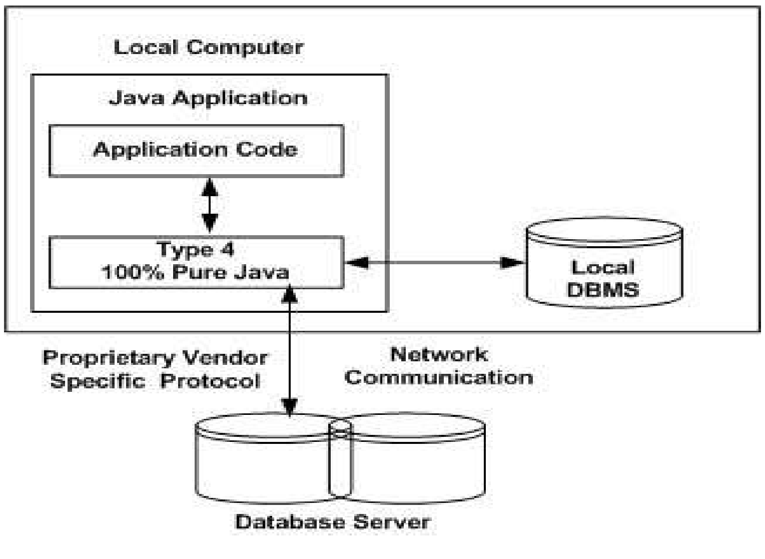

Figure 6: Pure Java Driver