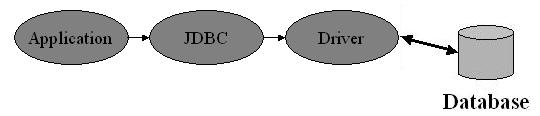
**JDBC (JAVA DATA BASE CONNECTIVITY)**

**JDBC** is API is used to connect java application to database.

A **JDBC driver** is a [software](http://en.wikipedia.org/wiki/Software) component that enables [Java](http://en.wikipedia.org/wiki/Java_(programming_language)) application to interact with a [database](http://en.wikipedia.org/wiki/Database)

**JDBC Architecture**



# JDBC Driver Types:

JDBC drivers are divided into four types or levels. The different types of JDBC drivers are:

Type 1: JDBC-ODBC Bridge driver (Bridge)

Type 2: Native-API/partly Java driver (Native)

Type 3: All Java/Net-protocol driver (Middleware)

Type 4: All Java/Native-protocol driver (Pure) THIN Driver.

**JDBC Type 1 Driver (JDBC-ODBC Bridge Driver):**

The Type 1 driver, also known as the JDBC-ODBC Bridge driver, is one of the earliest JDBC driver types. It provides a bridge between JDBC and ODBC (Open Database Connectivity) APIs. ODBC is a standard database access method that allows applications to access various databases using a common interface.

**JDBC-ODBC Bridge Driver (Type -1 Driver)**

Java Application

ODBC Library

Database

Here's how the Type 1 driver works:

1) Java application uses JDBC API calls to interact with the Type 1 driver.

2) The JDBC API calls are translated by the Type 1 driver into corresponding ODBC API calls. These ODBC calls are then sent to the ODBC driver manager.

3) The ODBC driver manager receives the ODBC calls and routes them to the appropriate ODBC driver.

4) The ODBC driver converts the ODBC calls into native database calls specific to the target database.

5) The native database calls are executed on the database server, and the results are sent back through the same path.

**Advantages:**

1. Easy to connect
2. Supports various databases

**Disadvantages:**

1. It is poor in performance
2. It is platform dependent
3. It requires ODBC libraries on client side

**Type 2 Driver - Native-API Driver / Partial Java Driver**

The Type 2 driver requires database-specific client side native libraries provided by the database vendor. These native libraries are written in the native language (C/C++) of the database system. The type 2 driver provides a set of Java classes and methods that mirror the JDBC API.

Java Application

Client side database native library

Database

When the Java application makes a JDBC API call, the corresponding method in the Type 2 driver's Java classes is invoked.

1. This Java method then interacts with the native libraries.
2. The native code translates the JDBC method calls and data into the appropriate commands understandable to the database.
3. The translated commands are sent to the database server, which processes the requests and returns the results.
4. The Type 2 driver's native code receives the response from the database server and translates it back into a format that the Java application can understand.

**Advantage:**

1. Faster than type-1 driver

**Disadvantages:**

1) Platform dependent

2) Database dependent

3) It can’t work without client side libraries

**Type 3 Drivers - Network-Protocol Driver**

Middleware Server

MySQL

Database Specific Calls

Oracle

DB2

Java Application

Java application uses JDBC API.

1. JDBC Driver Manager directs JDBC calls to Type 3 driver.
2. Type 3 driver communicates with middleware.
3. Middleware translates JDBC calls into database-specific calls
4. Middleware sends translated requests to database server.
5. Database server processes requests and sends results to middleware.
6. Middleware converts results into Type 3 driver-understandable format.
7. Type 3 driver sends results to Java application via JDBC Driver Manager.

**Advantages:**

1. Platform independent
2. Database independent

**Disadvantages:**

1. Server-side complexity,
2. Potential performance impact.

**Type 4 Driver - Native-Protocol Driver / Thin Driver / Pure Java Driver**

Database

Java Application

JDBC Type 4 driver provides a direct and efficient connection between a Java application and a database server using the database's native network protocol.

1. The type 4 driver is implemented purely in Java by database vendor. It can establish a direct connection with the database server using the database-specific protocol (e.g., TCP/IP).
2. The Java application's SQL queries are converted by the Type 4 driver into the format required by the database server's protocol.
3. The Type 4 driver sends the converted SQL queries over the network to the database server.
4. The database server processes the SQL queries, performs necessary database operations, and sends back the results over the network.
5. The Type 4 driver receives the database results and converts them into a format that the Java application can understand.
6. The Java application retrieves the results from the Type 4 driver and continues its processing.

**Advantages:**

1. Fastest of all JDBC drivers
2. Platform independent
3. Direct connection with DB makes it Secure
4. Easy to deploy
5. Easy to maintain

**Disadvantages:**

1. It is database dependent