Types of Drivers

While communicating with Database, we have to convert Java Calls into Database specific Calls and Database specific Calls into Java Calls. For this Driver Software is required. In the Market Thousands of Driver Softwares are available. But based on Functionality all Driver Software Drivers are divided into 4 Types.

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

2) **Type-2 Driver (Native API-Partly Java Driver OR Native Driver)**

3) **Type-3 Driver (All Java Net Protocol Driver OR Network Protocol Driver OR Middleware Driver)**

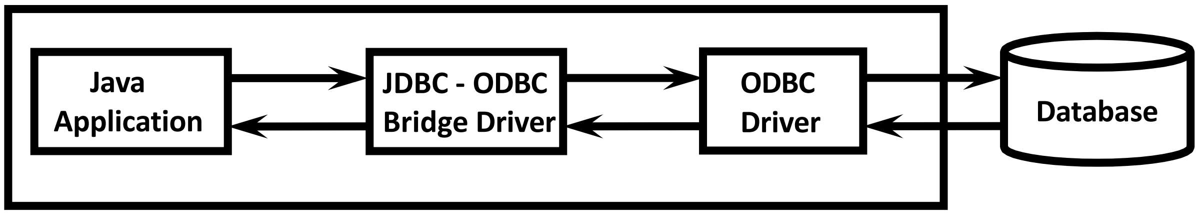
4) **Type-4 Driver (All Java Native Protocol Driver OR Pure Java Driver OR Thin Driver)**

Note:

Progress Data direct Software Company introduced Type-5 Driver, but it is not Industry Recognized Driver.

Type-1 Driver:

Also known as *JDBC-ODBC Bridge Driver* OR *Bridge Driver.*



Client Machine

This Driver provided by Sun Micro Systems as the Part of JDK. But this Support is available until 1.7 Version only.

Internally this Driver will take Support of ODBC Driver to communicate with Database.

Type-1 Driver converts JDBC Calls (Java Calls) into ODBC Calls and ODBC Driver converts ODBC Calls into Database specific Calls.

Hence Type-1 Driver acts as Bridge between JDBC and ODBC.

Advantages :

1. It is very easy to use and maintain.

2. We are not required to install any separate Software because it is available as the Part of JDK.

3. Type-1 Driver won't communicates directly with the Database. Hence it is Database Independent Driver. Because of this migrating from one Database to another Database will become Easy.

Limitations:

1. It is the slowest Driver among all JDBC Drivers (Snail Driver), because first it will convert JDBC Calls into ODBC Calls and ODBC Driver converts ODBC Calls into Database specific Calls.

2. This Driver internally depends on ODBC Driver, which will work only on Windows Machines. Hence Type-1 Driver is Platform Dependent Driver.

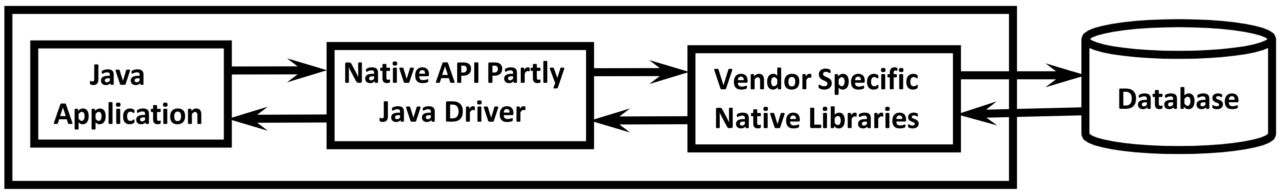
3. No Support from JDK 1.8 Version onwards.

Note:

Because of above Limitations it is never recommended to use Type-1 Driver.

Type-2 Driver:

It is also known as *Native API -Partly Java Driver* OR *Native Driver.*


Java Application
Native API Partly Java Driver
Vendor Specific Native Libraries
Database

Client Machine

Type-2 Driver is exactly same as Type-1 Driver except that ODBC Driver is replaced with Vendor specific Native Libraries.

Type-2 Driver internally uses Vendor specific Native Libraries to Communicate with Database.

Native Libraries means the Set of Functions written in Non-Java (Mostly C OR C++).

We have to install Vendor provided Native Libraries on the Client Machine.

Type-2 Driver converts JDBC Calls into Vendor specific Native Library Calls, which can be understandable directly by Database Engine.

Advantages:

1. When compared with Type-1 Driver Performance is High, because it required only one Level Conversion from JDBC to Native Library Calls.

2. No need of arranging ODBC Drivers.

3. When compared with Type-1 Driver, Portability is more because Type-1 Driver is applicable only for Windows Machines.

Limitations:

1. Internally this Driver using Database specific Native Libraries and hence it is Database

Dependent Driver. Because of this migrating from one Database to another Database will become Difficult.

2. This Driver is Platform Dependent Driver.

3. On the Client Machine compulsory we should install Database specific Native Libraries.

4. There is no Guarantee for every Database Vendor will provide This Driver.
(Oracle People provided Type-2 Driver but MySql People won't provide this Driver)

Eg: **OCI (Oracle Call Interface) Driver is Type-2 Driver provided by Oracle.**
OCI Driver internally uses OCI Libraries to communicate with Database.

OCI Libraries contain "C Language Functions"

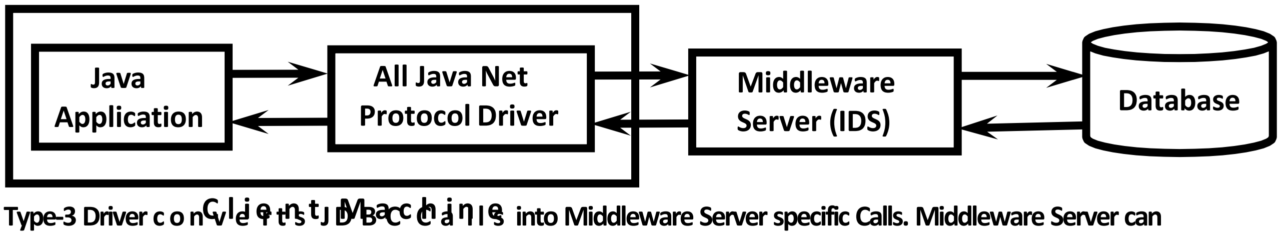
OCI Driver and corresponding OCI Libraries are available in the following Jar File. Hence we have to place this Jar File in the Class Path.

ojdbc14.jar ➔ Oracle 10g (Internally Uses Java 1.4V)
ojdbc6.jar ➔ Oracle 11g (Internally Uses Java 1.6V)
ojdbc7.jar ➔ Oracle 12c (Internally Uses Java 1.7V)

Note: **The only Driver which is both Platform Dependent and Database Dependent is Type-2 Driver. Hence it is not recommended to use Type-2 Driver.**

Type-3 Driver:

Also known as *All Java Net Protocol Driver* OR *Network Protocol Driver* OR *Middleware Driver*



Type-3 Driver converts JDBC Calls into Middleware Server specific Calls. Middleware Server can convert Middleware Server specific Calls into Database specific Calls.

Internally Middleware Server may use Type-1, 2 OR 4 Drivers to communicates with Database.

Advantages:

1. This Driver won't communicate with Database directly and hence it is Database Independent Driver.

2. This Driver is Platform Independent Driver.

3. No need of ODBC Driver OR Vendor specific Native Libraries

Limitations:

1. Because of having Middleware Server in the Middle, there may be a chance of Performance Problems.

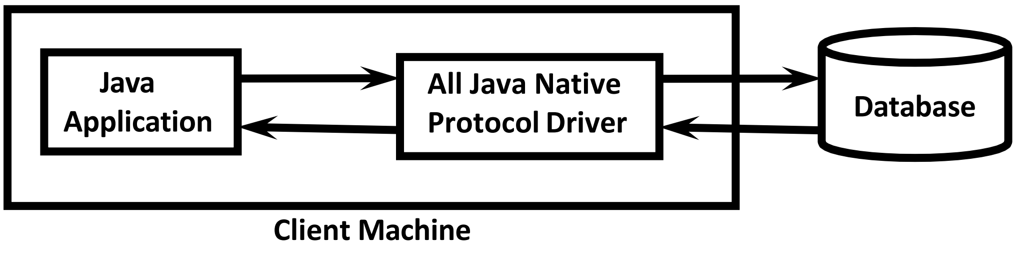
2. We need to purchase Middleware Server and hence the cost of this Driver is more when compared with remaining Drivers.

<u>Eg:</u> IDS Driver (Internet Database Access Server)

<u>Note:</u> The only Driver which is both Platform Independent and Database Independent is Type-3 Driver. Hence it is recommended to use.

<u>Type-4 Driver:</u>

Also known as *Pure Java Driver* OR *Thin Driver.*



This Driver is developed to talk with the Database directly without taking Support of *ODBC Driver* OR *Vendor Specific Native Libraries* OR *Middleware Server*.

This Driver uses Database specific Native Protocols to communicate with the Database.

This Driver converts JDBC Calls directly into Database specific Calls.

This Driver developed only in Java and hence it is also known as Pure Java Driver. Because of this, Type-4 Driver is Platform Independent Driver.
This Driver won't require any Native Libraries at Client side and hence it is light weighted. Because of this it is treated as Thin Driver.

<u>Advantages:</u>

1. It won't require any Native Libraries, *ODBC Driver* OR *Middleware Server*

2. It is Platform Independent Driver

3. It uses Database Vendor specific Native Protocol and hence Security is more.

<u>Limitation:</u>

The only Limitation of this Driver is, it is Database Dependent Driver because it is communicating with the Database directly.

<u>Eg:</u> Thin Driver for Oracle
Connector/J Driver for MySQL

<u>Note:</u> It is highly recommended to use Type-4 Driver.

Java Application ➔ Type-1 Driver ➔ ODBC Driver ➔ DB
Java Application ➔ Type-2 Driver ➔ Vendor Specific Native Libraries ➔ DB
Java Application ➔ Type-3 Driver ➔ Middleware Server ➔ DB
Java Application ➔ Type-4 Driver ➔ DB

Which Driver should be used?

1. If we are using only one Type of Database in our Application then it is recommended to use Type-4 Driver.

Eg: Stand Alone Applications, Small Scale Web Applications

2. If we are using multiple Databases in our Application then Type-3 Driver is recommended to use.

Eg: Large Scale Web Applications and Enterprise Applications

3. If Type-3 and Type-4 Drivers are not available then only we should go for Type-2 Driver.

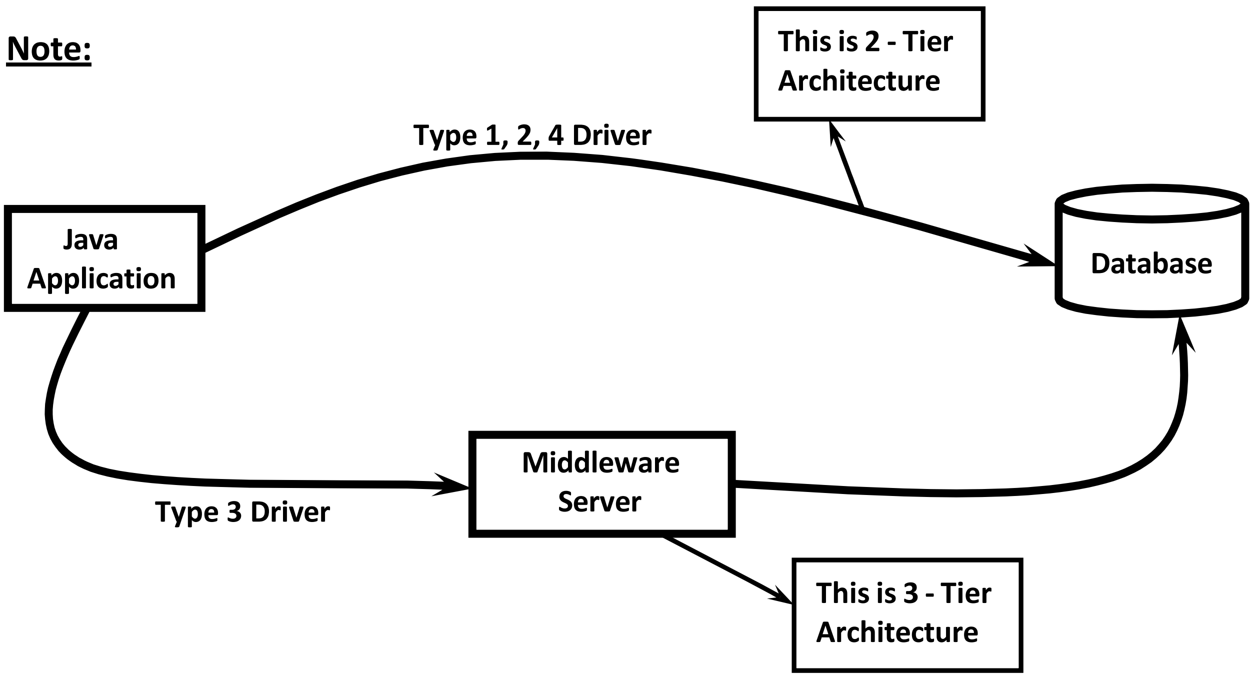
4. If no other Driver is available then only we should go for Type-1 Driver.

Differences between *Thin* and *Thick* Driver:

If Driver won't require any extra Component to communicate with Database, such type of Driver is called Thin Driver.
Eg: Type-4 Driver

If Driver require some extra Component (like *ODBC Driver* OR *Vendor specific Native Libraries* OR *Middleware Server*), such Type of Driver is called Thick Driver.
Eg: Type-1, Type-2 and Type-3 Drivers

Note:



Type-1, Type-2 and Type-4 Drivers follow 2-Tier Architecture.
Type-3 Driver follows 3-Tier Architecture.

Comparison Table of All JDBC Drivers

Property	Type - 1	Type - 2	Type - 3	Type - 4
1) Conversion	From JDBC Calls To ODBC Calls	From JDBC Calls To Native Library Calls	From JDBC Calls To Middleware Server Specific Calls	From JDBC Calls To Database Specific Calls
2) Implemented In	Only In Java	Java + Native Language	Only In Java	Only In Java
3) Architecture	2 - Tier	2 - Tier	3 - Tier	2 - Tier
4) Is It Platform Independent?	No	No	Yes	Yes
5) Is It Database Independent?	Yes	No	Yes	No
6) Is It Thin OR Thick?	Thick	Thick	Thick	Thin