JDBC(JAVA DATABASE CONNECTIVITY):

**QUERY PROCESSING SYSTEM:**

When we submit an SQL Query to the Database then Database Engine(DBE) will perform the following Steps:

**Step1:**

**QUERY TOKENIZATION:**

This Phase will take SQL Query as an Input, divide it into no.of tokens and Generate Stream of tokens as an output.

**Step2:**

**QUERY PROCESSING:**

This phase will take Stream of tokens as an Input,constructs Query Tree with the Tokens, if Query Tree build up is Success then no Syntax error is available in the provided SQL Query.If Query Tree is not Success(not built up) then there are some syntax errors in the provided SQL Query.

**Step3:**

**QUERY OPTIMIZATION:**

The main purpose of Query Optimization phase is to perform optimization on Query Tree in order to reduce execution time and to optimize memory utilization.

**Step4:**

**QUERY EXECUTION:**

This phase will take optimized Query Tree as an input and execute the Query by using interpreters.

**JDBC (JAVA DATABASE CONNECTIVITY):**

-->**The process of interacting with the database from Java Applications is called as JDBC.**

-->**JDBC is an API,which will provide very good predefined library to connect with database from JAVA Applications in order to perform the database operations.**

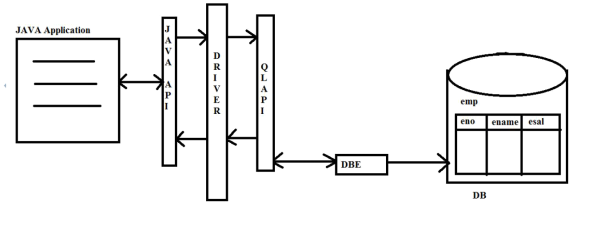
**--> In case of JDBC Applications we will define the database logic in Java application and we will send the Java represented database logic to Database Engine. But database engine is unable to execute the Java represented database logic, it should require the database logic in Query Language Representations.**

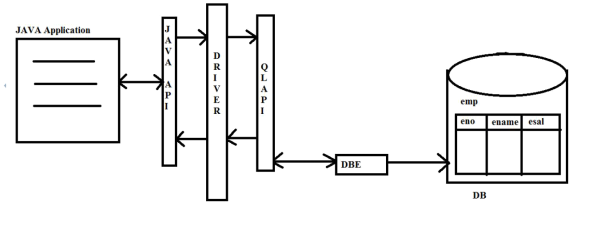
**In the above context, to execute JDBC applications we should require a conversion mechanism to convert the database logic from Java representations to Query language representations and from Query language representations to Java representations.**

-->**In the above situation the required conversion mechanisms are available in the form of a software(Interace) called as "Driver".**

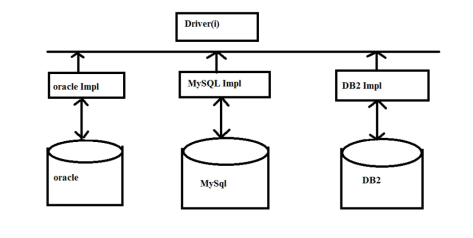
**DRIVER:**

**-->Driver is an interface existed between Java application and database to map Java API calls to Query language API calls and Query language API calls to Java API calls.**



**-->To provide driver as a product Sun MicroSystems has provided Driver as an interface and Sun MicroSystems lets the database vendors to provide implementation classes to the driver interface as part of their database software's.**

**-->If we want to use Drivers in JDBC applications then we have to get Driver interface implementation from the respective database software.**



**->There are 180+ numbers of drivers but all these drivers could be classified into the following Four types:**

* 1. **Type-1**
  2. **Type-2**
  3. **Type-3**
  4. **Type-4**

**TYPE-1 DRIVER:**

**-->Type-1 Driver is also called as JDBC-ODBC Driver and Bridge Driver.**

**-->JDBC-ODBC Driver is a driver provided by Sun Micro Systems as an Implementation class to Driver Interface.**

**-->Sun MicroSystems has provided JDBC-ODBC Driver with the inter dependency on the Microsoft’s product ODBC Driver.**

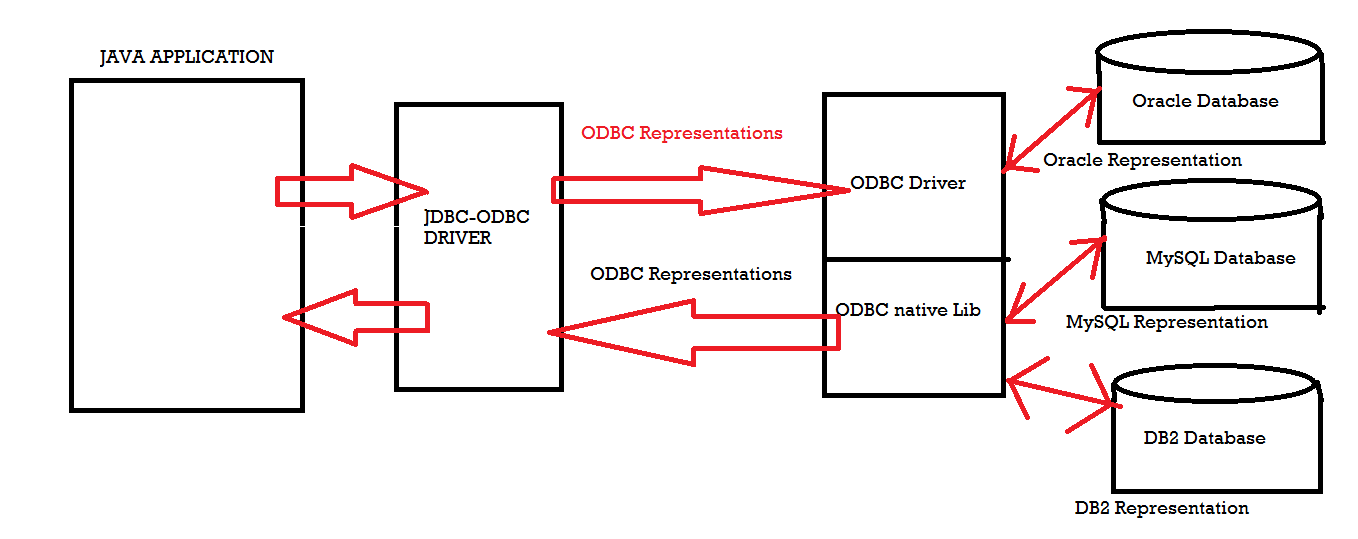
**-->ODBC Driver is a Open Specification, it will provide very good environment to interact with any type of database from JDBC-ODBC Driver.**

**-->If we want to use JDBC-ODBC Driver in our JDBC Applications first we have to configure the MicroSoft Product ODBC Driver native library.**

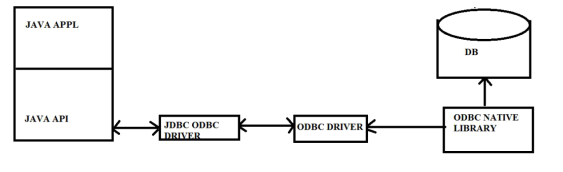
**-->To interact with the database from Java Application if we use JDBC-ODBC Driver then we require two types of conversions that makes JDBC-ODBC Driver as Slower Driver.**

**-->JDBC-ODBC Driver is highly recommended for standalone applications, it is not suitable for web applications, distributed applications.**

**-->The portability of the JDBC-ODBC Driver is very less. Due to its interdependency on microsoft product ODBC Driver.**

****

**OR**



1. **TYPE-2 DRIVER:**

**-->Type-2 Driver is also called part java,part native driver that is Type-2 Driver was implemented by using Java implementations and the database vendor provided native library.**

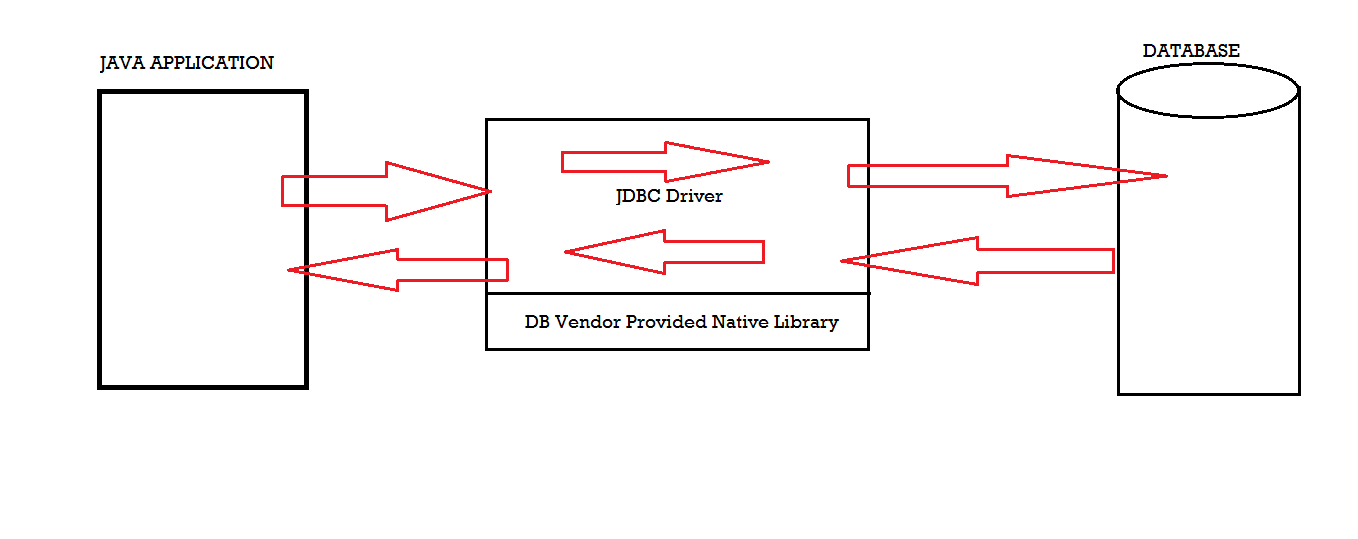
**-->When compared to Type-1 Driver, Type-2 Driver is faster Driver because it do not require two times conversions to interact with the Database from Java Applications.**

**-->When compared to Type-1 Driver Type-2 driver portability is more.**

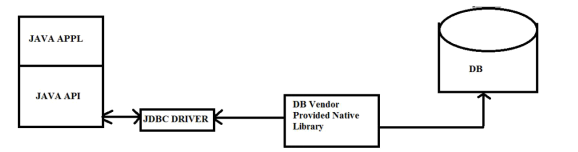
**-->Type-2 Driver is still recommended for standalone applications and not suggested for web applications and Enterprise applications.**

**-->If we want to use Type-2 Driver in our Jdbc applications then we have to install the database vendor provided native library.**

**-->Type-2 Driver's portability is not good when compared to Type-3 Driver and Type-4 Driver.**



**OR**



**3) TYPE-3 DRIVER:**

**-->Type-3 Driver is also called as MiddleWare DataBase Server Access Driver and NetWorkDriver.**

**-->Type-3 Driver is purely designed for Enterprise applications. It is not suggested for standalone applications.**

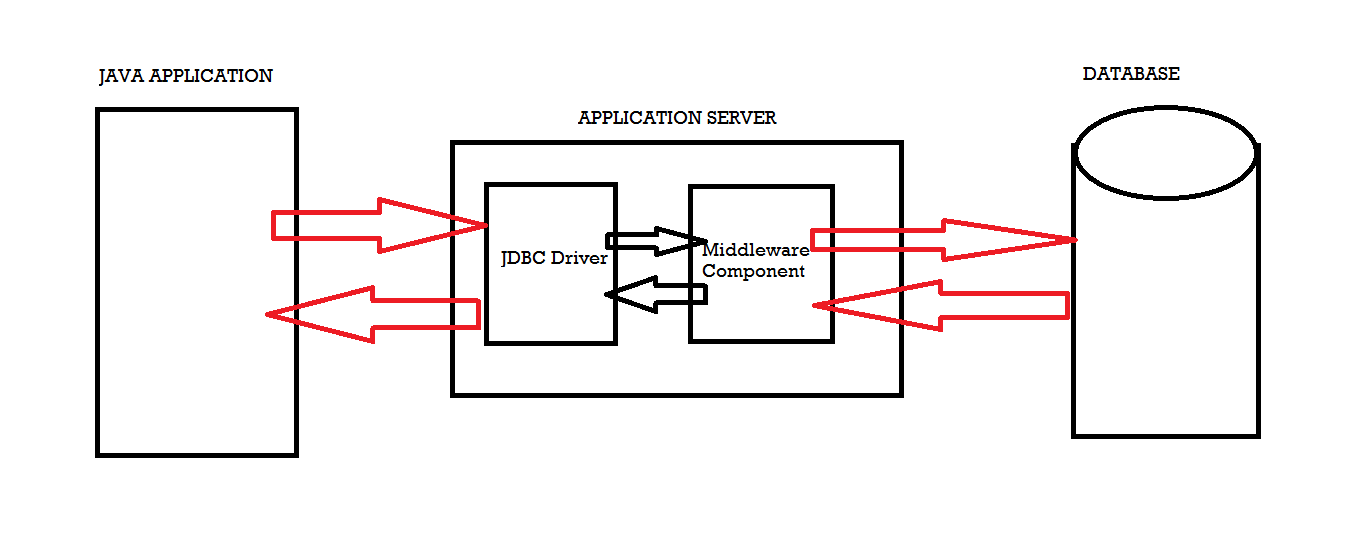
**-->Type-3 Driver portability is very good when compared to Type-1 and Type-2 Driver's.**

**-->Type-3 Driver will provide very good environment to interact with multiple number of databases.**

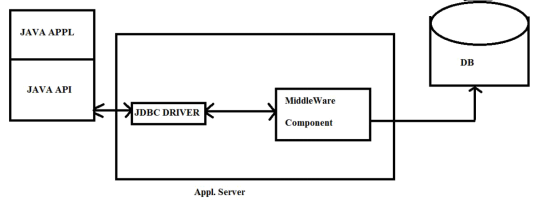
**-->Type-3 Driver will provide very good environment to switch from one database to another database without having modifications to client applications.**

**-->Type-3 Driver do not require any native library installations, it must require the Compatibility with application server.**

**-->Type-3 Driver is fastest Driver when compared to all the Drivers.**

****

**OR**



**4) TYPE-4 DRIVER:**

**-->Type-4 Driver is also called as pure Java Driver and Thin Driver because Type-4 Driver is implemented completely by using java conventions.**

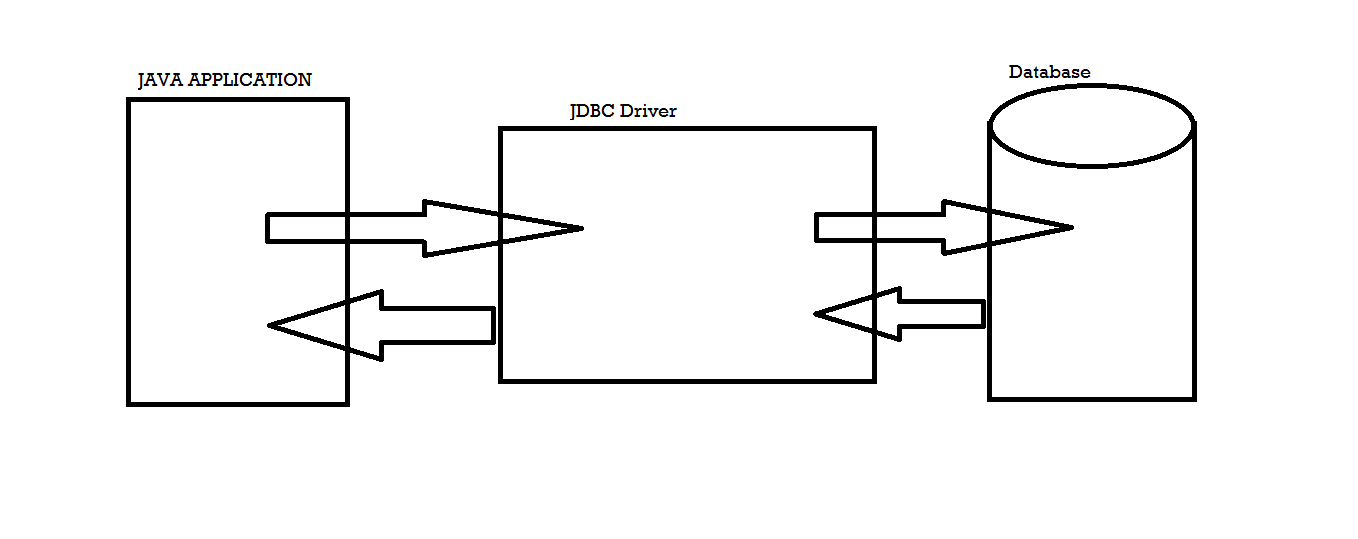
**-->Type-4 Driver is the most frequent used Driver when compared to all other remaining Drivers.**

**-->Type-4 Driver is recommended for any type of application including standalone applications, Network Applications, Distributed Aplications.**

**-->Type-4 Driver portability is very good when compared to all the remaining Drivers.**

**-->Type 4 driver do not require any native library dependences and it requires only one time conversion to interact with database from Java Applications.**

**-->Type 4 is the cheapest Driver amongst all Drivers.**

****

**OR**

