DATABASE

A database is an organized collection of structured information, or data, typically stored electronically in a computer system. A database is usually controlled by a database management system (DBMS). Together, the data and the DBMS, along with the applications that are associated with them, are referred to as a database system, often shortened to just database.

SQL

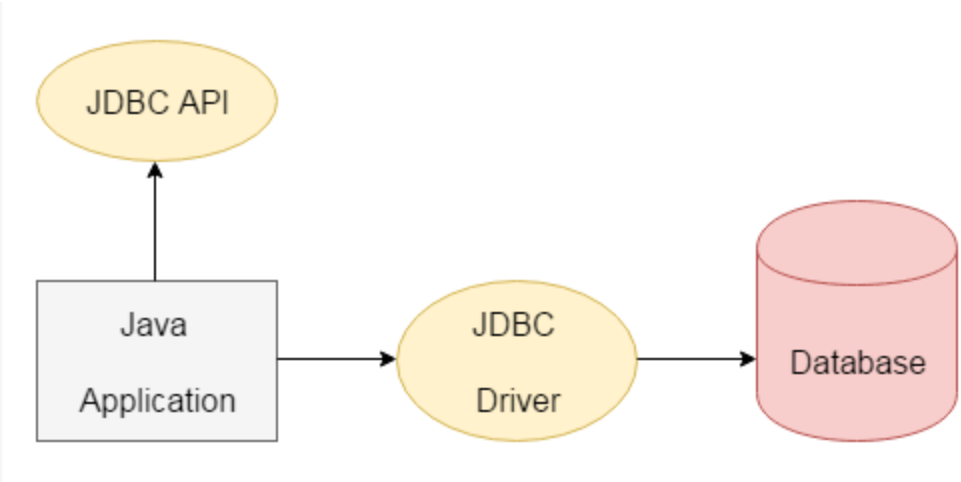
SQL tutorial provides basic and advanced concepts of SQL. Our SQL tutorial is designed for both beginners and professionals.SQL (Structured Query Language) is used to perform operations on the records stored in the database, such as updating records, inserting records, deleting records, creating and modifying database tables, views, etc.

SQL is not a database system, but it is a query language.Suppose you want to perform the queries of SQL language on the stored data in the database. You are required to install any database management system in your systems, for example, Oracle, MySQL, MongoDB, PostgreSQL, SQL Server, DB2, etc.

JDBC

JDBC stands for **Java Database Connectivity**. JDBC is a Java API to connect and execute the query with the database. It is a part of JavaSE (Java Standard Edition). JDBC API uses JDBC drivers to connect with the database.

We can use JDBC API to access tabular data stored in any relational database. By the help of JDBC API, we can save, update, delete and fetch data from the database. It is like Open Database Connectivity (ODBC) provided by Microsoft.



JDBC Driver

JDBC Driver is a software component that enables java application to interact with the database. There are 4 types of JDBC drivers:

1. JDBC-ODBC bridge driver
2. Native-API driver
3. Network Protocol driver
4. Thin driver

JDBC-ODBC bridge driver

The JDBC-ODBC bridge driver uses ODBC driver to connect to the database. The JDBC-ODBC bridge driver converts JDBC method calls into the ODBC function calls. This is now discouraged because of thin driver.

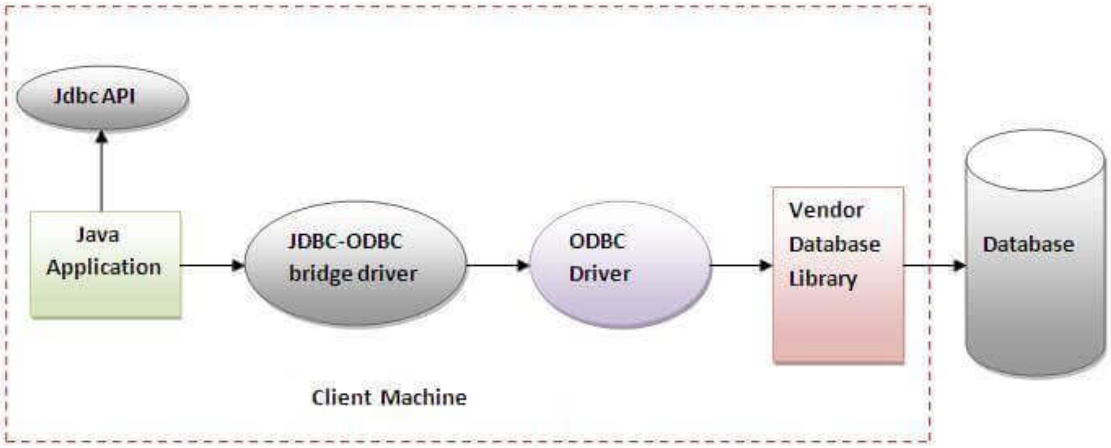


Figure- JDBC-ODBC Bridge Driver

Native API driver

The Native API driver uses the client-side libraries of the database. The driver converts JDBC method calls into native calls of the database API. It is not written entirely in java.

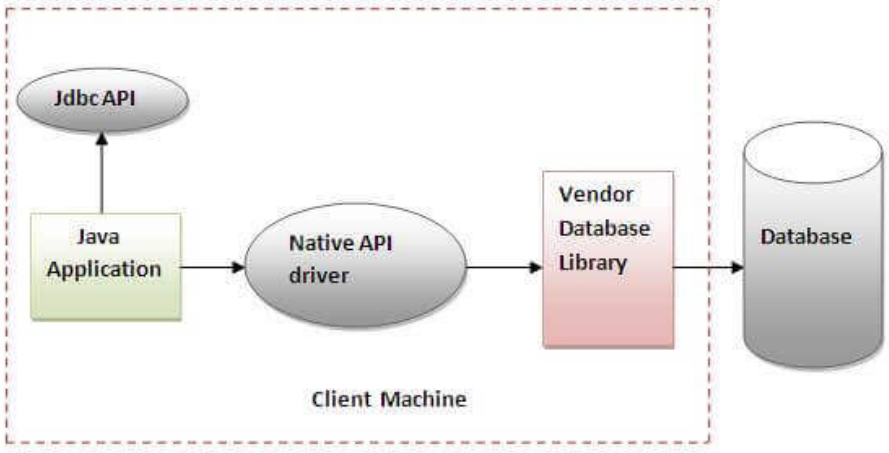


Figure- Native API Driver

Network Protocol driver

The Network Protocol driver uses middleware (application server) that converts JDBC calls directly or indirectly into the vendor-specific database protocol. It is fully written in java.

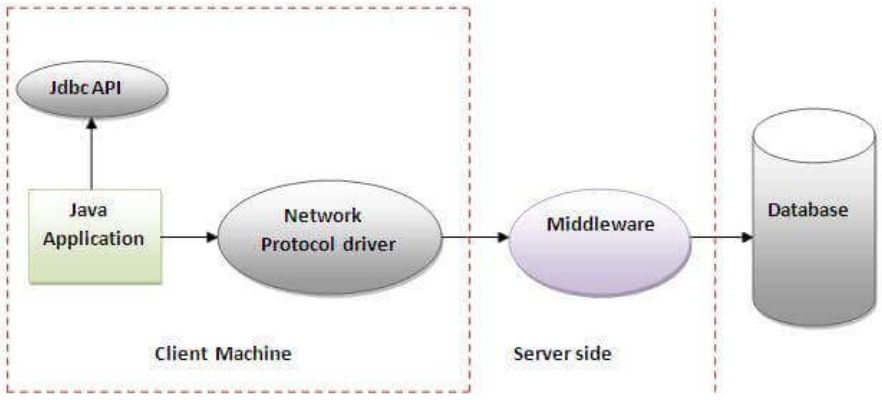


Figure- Network Protocol Driver

Thin driver

The thin driver converts JDBC calls directly into the vendor-specific database protocol. That is why it is known as thin driver. It is fully written in Java language.

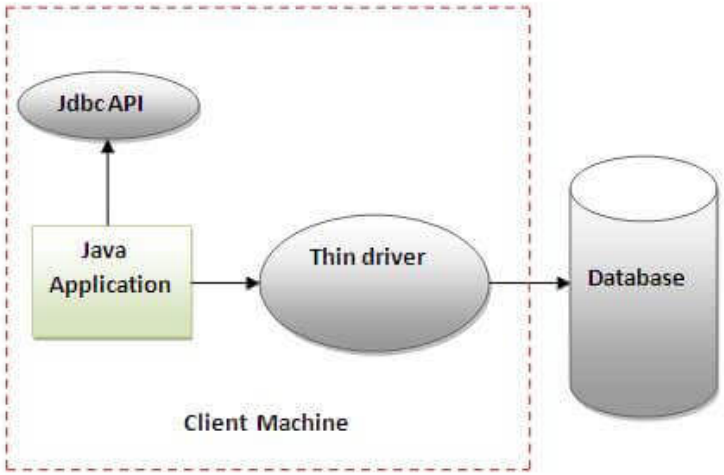


Figure- Thin Driver

Java Database Connectivity

There are 5 steps to connect any java application with the database using JDBC. These steps are as follows:

- Register the Driver class
- Create connection
- Create statement
- Execute queries
- Close connection

https://www.javatpoint.com/steps-to-connect-to-the-database-in-java

```
import java.beans.Statement;
import java.sql.Connection;
import java.sql.DriverManager;
import java.sql.ResultSet;
import java.sql.SQLException;
import java.util.Scanner;

public class JdbcExample {

	public static void main(String[] args) throws ClassNotFoundException, SQLException {

		Class.forName("oracle.jdbc.driver.OracleDriver");

		Connection con =
DriverManager.getConnection("jdbc:oracle:thin:@localhost:1521:xe", "system", "123");

		java.sql.Statement st = con.createStatement();

		ResultSet rs = st.executeQuery("select * from student");
		while (rs.next()) {
			System.out.println(rs.getInt(1) + " " + rs.getString(2));
		}

		/*
		 * Scanner scan = new Scanner(System.in);
		 *
		 * System.out.println("Type rollno");
		 * int rn = scan.nextInt();
		 * System.out.println("Type Name");
		 * scan.nextLine();
		 * String name = scan.nextLine();
		 * int rs = st.executeUpdate("insert into student values (" +rn + ",'" + name + "')");
		 * if (rs > 0)
		 * System.out.println("Inserted ");
		 * String query = "insert into student values (?, ?)";
		 * java.sql.PreparedStatement pt = con.prepareStatement(query);
		 * pt.setInt(1, rn);
		 * pt.setString(2, name);
		 * int rs = pt.executeUpdate();
		 * if (rs > 0)
		 * System.out.println("Inserted");
```

```
                 */
                con.close();

        }

}
```