# Java Database Connectivity with 5 Steps

# [5 Steps to connect to the database in java](https://www.javatpoint.com/steps-to-connect-to-the-database-in-java)

# [Register the driver class](https://www.javatpoint.com/steps-to-connect-to-the-database-in-java" \l "step1)

# [Create the connection object](https://www.javatpoint.com/steps-to-connect-to-the-database-in-java" \l "step2)

# [Create the Statement object](https://www.javatpoint.com/steps-to-connect-to-the-database-in-java" \l "step3)

# [Execute the query](https://www.javatpoint.com/steps-to-connect-to-the-database-in-java" \l "step4)

# [Close the connection object](https://www.javatpoint.com/steps-to-connect-to-the-database-in-java" \l "step5)

|  |
| --- |
| There are 5 steps to connect any java application with the database using JDBC. These steps are as follows:Register the Driver classCreate connectionCreate statementExecute queriesClose connection |

# IMG_256

# 1) Register the driver class

|  |
| --- |
| The forName() method of Class class is used to register the driver class. This method is used to dynamically load the driver class. |

# Syntax of forName() method

# public static void forName(String className)throws ClassNotFoundException

# Note: Since JDBC 4.0, explicitly registering the driver is optional. We just need to put vender's Jar in the classpath, and then JDBC driver manager can detect and load the driver automatically.

# Example to register the OracleDriver class

# Here, Java program is loading oracle driver to esteblish database connection.

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

# 2) Create the connection object

|  |
| --- |
| The getConnection() method of DriverManager class is used to establish connection with the database. |

# Syntax of getConnection() method

# 1) public static Connection getConnection(String url)throws SQLException

# 2) public static Connection getConnection(String url,String name,String password)

# throws SQLException

# Example to establish connection with the Oracle database

# Connection con=DriverManager.getConnection(

# "jdbc:oracle:thin:@localhost:1521:xe","system","password");

# 3) Create the Statement object

|  |
| --- |
| The createStatement() method of Connection interface is used to create statement. The object of statement is responsible to execute queries with the database. |

# Syntax of createStatement() method

# public Statement createStatement()throws SQLException

# Example to create the statement object

# Statement stmt=con.createStatement();

# 4) Execute the query

|  |
| --- |
| The executeQuery() method of Statement interface is used to execute queries to the database. This method returns the object of ResultSet that can be used to get all the records of a table. |

# Syntax of executeQuery() method

# public ResultSet executeQuery(String sql)throws SQLException

# Example to execute query

# ResultSet rs=stmt.executeQuery("select \* from emp");

# 

# while(rs.next()){

# System.out.println(rs.getInt(1)+" "+rs.getString(2));

# }

# 5) Close the connection object

|  |
| --- |
| By closing connection object statement and ResultSet will be closed automatically. The close() method of Connection interface is used to close the connection. |

# Syntax of close() method

# public void close()throws SQLException

# Example to close connection

# con.close();

# Java Database Connectivity with Oracle

|  |
| --- |
| To connect java application with the oracle database, we need to follow 5 following steps. In this example, we are using Oracle 10g as the database. So we need to know following information for the oracle database:   1. **Driver class:**The driver class for the oracle database is **oracle.jdbc.driver.OracleDriver**. 2. **Connection URL:**The connection URL for the oracle10G database is **jdbc:oracle:thin:@localhost:1521:xe** where jdbc is the API, oracle is the database, thin is the driver, localhost is the server name on which oracle is running, we may also use IP address, 1521 is the port number and XE is the Oracle service name. You may get all these information from the tnsnames.ora file. 3. **Username:**The default username for the oracle database is **system**. 4. **Password:**It is the password given by the user at the time of installing the oracle database. |

|  |
| --- |
| Create a Table Before establishing connection, let's first create a table in oracle database. Following is the SQL query to create a table. |

1. create table emp(id number(10),name varchar2(40),age number(3));

### Example to Connect Java Application with Oracle database

In this example, we are connecting to an Oracle database and getting data from **emp** table. Here, **system** and **oracle** are the username and password of the Oracle database.

**import** java.sql.\*;

**class** OracleCon{

**public** **static** **void** main(String args[]){

**try**{

//step1 load the driver class

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

//step2 create  the connection object

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

//step3 create the statement object

Statement stmt=con.createStatement();

//step4 execute query

ResultSet rs=stmt.executeQuery("select \* from emp");

**while**(rs.next())

System.out.println(rs.getInt(1)+"  "+rs.getString(2)+"  "+rs.getString(3));

//step5 close the connection object

con.close();

}**catch**(Exception e){ System.out.println(e);}

}

}

The above example will fetch all the records of emp table.

To connect java application with the Oracle database ojdbc14.jar file is required to be loaded.

[download the jar file ojdbc14.jar](https://static.javatpoint.com/src/jdbc/ojdbc14.jar)

### Two ways to load the jar file:

1. paste the ojdbc14.jar file in jre/lib/ext folder
2. set classpath

### 1) paste the ojdbc14.jar file in JRE/lib/ext folder:

|  |
| --- |
| Firstly, search the ojdbc14.jar file then go to JRE/lib/ext folder and paste the jar file here. |

### 2) set classpath:

|  |
| --- |
| There are two ways to set the classpath:   * temporary * permanent |

### How to set the temporary classpath:

|  |
| --- |
| Firstly, search the ojdbc14.jar file then open command prompt and write: |

1. C:>set classpath=c:\folder\ojdbc14.jar;.;

### How to set the permanent classpath:

Go to environment variable then click on new tab. In variable name write **classpath** and in variable value paste the path to ojdbc14.jar by appending ojdbc14.jar;.; as C:\oraclexe\app\oracle\product\10.2.0\server\jdbc\lib\ojdbc14.jar;.;

To see the slides of seting parmanent path [click here](https://www.javatpoint.com/how-to-set-path-in-java" \t "https://www.javatpoint.com/_blank)