

# 5 Steps to connect to the database in java

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

- Register the driver class
- Creating connection
- Creating statement
- Executing queries
- Closing connection

## 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
```

## Example to register the `OracleDriver` class

```
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();
```

[< prev](#)[next >](#)

## Please Share



## Learn Latest Tutorials



JPA



JavaFX



Tika



Kotlin



OS



.Net

