**Assignment No 7.2**

1. Write down the steps to connect with a database using JDBC.

|  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- |
| Answer: 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.   1. 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.   1. Connection con=DriverManager.getConnection( 2. "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. |   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. |  1. ResultSet rs=stmt.executeQuery("select \* from emp"); 3. **while**(rs.next()){ 4. System.out.println(rs.getInt(1)+" "+rs.getString(2)); 5. }  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. |  1. con.close(); |

2. Describe the use of following methods with example-

a. executeQuery()

**1) public ResultSet executeQuery(String sql):** is used to execute SELECT query. It returns the object of ResultSet.

Example: Let’s see the simple example of ResultSet interface to retrieve the data of 3rd row.

**import** java.sql.\*;

**class** FetchRecord{

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

Class.forName("com.mysql.jdbc.Driver");

Connection con=DriverManager.getConnection("jdbc:mysql://localhost:3306/acadgild","root",");

Statement stmt=con.createStatement(ResultSet.TYPE\_SCROLL\_SENSITIVE,ResultSet.CONCUR\_UPDATABLE);

ResultSet rs=stmt.executeQuery("select \* from emp765");

rs.absolute(3);

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

con.close();

}}

b. executeUpdate()

is used to execute specified query, it may be create, drop, insert, update, delete etc.

Example: Let’s see the simple example of Statement interface to insert, update and delete the record.

**import** java.sql.\*;

**class** FetchRecord{

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

Class.forName("com.mysql.jdbc.Driver");

Connection con=DriverManager.getConnection("jdbc:mysql://localhost:3306/acadgild","root",");

Statement stmt=con.createStatement();

**int** result=stmt.executeUpdate("delete from emp765 where id=33");

System.out.println(result+" records affected");

con.close();

}}

c. next()

**public boolean next():**is used to move the cursor to the one row next from the current position. Moves the cursor to the next row. This method returns false if there are no more rows in the result set.

//STEP 1. Import required packages

import java.sql.\*;

public class JDBCExample {

// JDBC driver name and database URL

static final String JDBC\_DRIVER = "com.mysql.jdbc.Driver";

static final String DB\_URL = "jdbc:mysql://localhost/EMP";

// Database credentials

static final String USER = "username";

static final String PASS = "password";

public static void main(String[] args) {

Connection conn = null;

Statement stmt = null;

try{

//STEP 2: Register JDBC driver

Class.forName("com.mysql.jdbc.Driver");

//STEP 3: Open a connection

System.out.println("Connecting to database...");

conn = DriverManager.getConnection(DB\_URL,USER,PASS);

//STEP 4: Execute a query

System.out.println("Creating statement...");

stmt = conn.createStatement();

String sql = "UPDATE Employees set age=30 WHERE id=103";

// Let us check if it returns a true Result Set or not.

Boolean ret = stmt.execute(sql);

System.out.println("Return value is : " + ret.toString() );

// Let us update age of the record with ID = 103;

int rows = stmt.executeUpdate(sql);

System.out.println("Rows impacted : " + rows );

// Let us select all the records and display them.

sql = "SELECT id, first, last, age FROM Employees";

ResultSet rs = stmt.executeQuery(sql);

//STEP 5: Extract data from result set

while(rs.next()){

//Retrieve by column name

int id = rs.getInt("id");

int age = rs.getInt("age");

String first = rs.getString("first");

String last = rs.getString("last");

//Display values

System.out.print("ID: " + id);

System.out.print(", Age: " + age);

System.out.print(", First: " + first);

System.out.println(", Last: " + last);

}

//STEP 6: Clean-up environment

rs.close();

stmt.close();

conn.close();

}catch(SQLException se){

//Handle errors for JDBC

se.printStackTrace();

}catch(Exception e){

//Handle errors for Class.forName

e.printStackTrace();

}finally{

//finally block used to close resources

try{

if(stmt!=null)

stmt.close();

}catch(SQLException se2){

}// nothing we can do

try{

if(conn!=null)

conn.close();

}catch(SQLException se){

se.printStackTrace();

}//end finally try

}//end try

System.out.println("Goodbye!");

}//end main

}//end JDBCExample