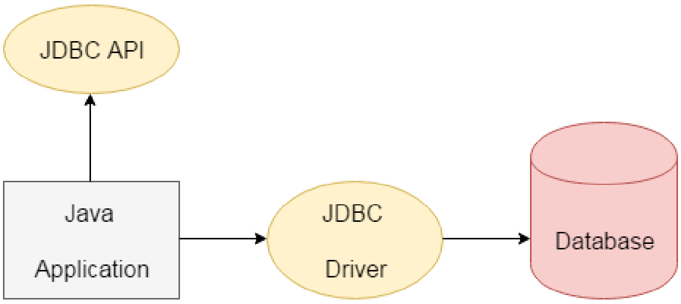
Practical - 9 Java connectivity with database

Java 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.



DBC API to handle database using Java program and can perform the following activities:

1. Connect to the database
2. Execute queries and update statements to the database
3. Retrieve the result received from the database.

Java Database Connectivity

1. Register the Driver class
2. Create connection
3. Create statement
4. Execute queries
5. Close 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 :

```
public static void forName(String className)throws ClassNotFoundException
```

E.g 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.

public static Connection getConnection(String url)throws SQLException

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

E.g establish connection with 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 : public Statement createStatement()throws SQLException

E. g   Statement stmt=con.createStatement();

4) Execute the query

public ResultSet executeQuery(String sql)throws SQLException

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

```
    while(rs.next()){
            System.out.println(rs.getInt(1)+" "+rs.getString(2));
      }
```

5) Close the connection object

It is used to close the connection

Syntax:
public void close()throws SQLException
e.g con.close()

Java Database Connectivity with MySQL

1. **Driver class:** The driver class for the mysql database is **com.mysql.jdbc.Driver**.
2. **Connection URL:** The connection URL for the mysql database is **jdbc:mysql://localhost:3306/Testdb** localhost is the server name on which mysql is running, we may also use IP address, 3306 is the port number and sonoo is the database name. Testdb is the database.
1. **Username:** The default username for the mysql database is **root**.

3. **Password:** password given by the user at the time of installing the mysql database.

To connect java application with the mysql atabase, **mysqlconnector.jar** file is required to be loaded.

download the jar file mysql-connector.jar

1. Paste the mysqlconnector.jar in your desire location (prefarabally in working folder where the program is available)
2. Set the class path to the jar file

E.g simple database connectivity with the mysql and display record

Step 1 : Add the jar file in library folder in netbeans project

Step 2 : write a code

```
import java.sql.*;
class MysqlCon{
public static void main(String args[]){
try{
// bind the class
      Class.forName("com.mysql.jdbc.Driver");
// get connection
        Connection con=DriverManager.getConnection(
"jdbc:mysql://localhost:3306/Testdb","root","mcalab");

//here Testdb  is database name, root is username and password  is mcalab
Statement stmt=con.createStatement();
// execute the query and return result into the Resultset
ResultSet rs=stmt.executeQuery("select * from emp");
while(rs.next())
        System.out.println(rs.getInt(1)+"  "+rs.getString(2)+"  "+ rs.getString(3));

con.close();
}catch(Exception e){ System.out.println(e);}
  }
}
```

E.g 2 database connectivity with prepared statement

```
import java.sql.*;
/**
 *
 * @author STUDENT
 */
public class SQLConnection {
     Connection con;
     Statement stmt;

    SQLConnection()
    {
       try{
      Class.forName("com.mysql.cj.jdbc.Driver");
      con=DriverManager.getConnection( "jdbc:mysql://localhost:3306/Testdb",
"root","mcalab");
    } catch(Exception e){System.out.println(e);}

    }

    public void insert(int i , String n , int sal){
      try{
       stmt = con.createStatement();
      stmt.executeUpdate("insert into emp values(" + i + ",'"+n +"',"+ sal + ");" );
      System.out.println("inserted Successfully");
      } catch(SQLException e)
               {System.out.println(e);}
    }
```

```
  public void search(int sal){
    try{
   stmt = con.createStatement();
  ResultSet rs=stmt.executeQuery("select *from emp where salary > " + sal + ";");

 while(rs.next())  {
       System.out.println(rs.getInt(1)+"  "+rs.getString(2)+"  "+rs.getString(3));
     }
  } catch(SQLException e){System.out.println(e);}

  }

 public void update(){
  try{
  String query = "update emp set name = ? where id = ?";
   PreparedStatement preparedStmt = con.prepareStatement(query);
   preparedStmt.setString   (1, "YYYY");
   preparedStmt.setInt(2, 3);
  int no= preparedStmt.executeUpdate();
   System.out.println(no + " rows updated");
  }catch(Exception e){}
  }


   public static void main(String[] args) {
     // TODO code application logic here
      SQLConnection ssc = new SQLConnection();
      ssc.insert(10, "XXXX", 34000);

      ssc.search(30000);
      ssc.update();

 }

}
```

For connecting with the oracle

- **Driver class:** The driver class for the oracle database is oracle.jdbc.driver.OracleDriver.

- **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.
- **Username:** The default username for the oracle database is system.
- **Password:** It is the password given by the user at the time of installing the oracle database.

Exercise

1	Create database Itemdb in mysql Create table Item(itemid,name,price,category) Write a java program for following connect java program with mysql database 1. Insert values into the item table (take the required values form the user input) 2. Display the all the details of the Item table 3. Update the category of item of given itemid 4. Delete the record of given item id. Display proper message for all the operations completion.