**Spring JDBC Batch:**

**ApplicationContext.xml:**

<bean id=*"ds"* class=*"org.springframework.jdbc.datasource.DriverManagerDataSource"*>

<property name=*"driverClassName"* value=*"oracle.jdbc.driver.OracleDriver"* />

<property name=*"url"* value=*"jdbc:oracle:thin:@localhost:1521:XE"* />

<property name=*"username"* value=*"haldia"* />

<property name=*"password"* value=*"nic"* />

</bean>

<bean id=*"jdbcTemplate"* class=*"org.springframework.jdbc.core.JdbcTemplate"*>

<property name=*"dataSource"* ref=*"ds"*></property>

</bean>

<bean id=*"edao"* class=*"org.EmployeeDAO"*>

<property name=*"jdbcTemplate"* ref=*"jdbcTemplate"*></property>

</bean>

Employee.java:

**public** **class** Employee {

**private** **int** id;

**private** String name;

**private** String address;

**public** Employee(){}

**public** Employee(**int** id,String nm,String addr)

{

**this**.id=id;

**this**.name=nm;

**this**.address=addr;

}

**public** Employee(**int** id)

{

**this**.id=id;

}

**public** **int** getId() {

**return** id;

}

**public** **void** setId(**int** id) {

**this**.id = id;

}

**public** String getName() {

**return** name;

}

**public** **void** setName(String name) {

**this**.name = name;

}

**public** String getAddress() {

**return** address;

}

**public** **void** setAddress(String address) {

**this**.address = address;

}

**public** String toString() {

**return** "Employee [id=" + id + ", name=" + name + ", address=" + address

+ "]";

}

}

BatchDAO.java

**public** **class** BatchDAO {

**private** JdbcTemplate jdbcTemplate;

**public** **void** setJdbcTemplate(JdbcTemplate jdbcTemplate) {

**this**.jdbcTemplate = jdbcTemplate;

}

**public** **int**[] insertBatch(**final** List<Employee> empList){

**int**[] count =jdbcTemplate.batchUpdate("insert into employee values (?,?,?)",**new** BatchPreparedStatementSetter() {

**public** **void** setValues(PreparedStatement ps, **int** i) **throws** SQLException {

Employee emp=empList.get(i);

ps.setInt(1,emp.getId());

ps.setString(2,emp.getName());

ps.setString(3,emp.getAddress());

}

**public** **int** getBatchSize() {

// **TODO** Auto-generated method stub

**return** empList.size();

}

});

**return** count;

}

}

BatchMain.java

**public** **class** BatchMain {

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

// **TODO** Auto-generated method stub

ApplicationContext ctx=**new** ClassPathXmlApplicationContext("org/ApplicationContext.xml");

BatchDAO dao=(BatchDAO)ctx.getBean("bdao");

Employee e=**new** Employee(4,"Sujoy","Tarekeshwar");

Employee e1=**new** Employee(5,"Debopriyo","Selimpur");

Employee e2=**new** Employee(6,"Papia","Asansol");

ArrayList list=**new** ArrayList();

list.add(e);

list.add(e1);

list.add(e2);

**int** count[]= dao.insertBatch(list);

System.*out*.println("Data Inserted :"+count.length);

}

}

Spring JDBC Query Examples:

EmployeeDAO.java

**public** **class** EmployeeDAO {

**private** JdbcTemplate jdbcTemplate;

**public** **void** setJdbcTemplate(JdbcTemplate jdbcTemplate) {

**this**.jdbcTemplate = jdbcTemplate;

}

**public** **int** saveEmployee(Employee e){

String query="insert into employee values('"+e.getId()+"','"+e.getName()+"','"+e.getAddress()+"')";

**return** jdbcTemplate.update(query);

}

**public** **int** updateEmployee(Employee e){

String query="update employee set name='"+e.getName()+"',address='"+e.getAddress()+"' where id='"+e.getId()+"' ";

**return** jdbcTemplate.update(query);

}

**public** **int** deleteEmployee(Employee e){

String query="delete from employee where id='"+e.getId()+"' ";

**return** jdbcTemplate.update(query);

}

**public** List<Employee> getEmployee(String id){

String query="select \* from employee where id=?";

**return** jdbcTemplate.query(query, **new** Object[]{id}, **new** EmployeeRowMapper());

}

**public** List<Employee> getAllEmployee(){

String query="select \* from employee order by id ";

**return** jdbcTemplate.query(query, **new** EmployeeRowMapper());

}

**public** **int** employeeCount(){

String sql ="select count(\*) from employee";

**return** jdbcTemplate.queryForObject(sql, Integer.**class**);

}

**public** String getEmployeeNameById(String id)

{

String name=**null**;

**try**

{

String sql="select name from employee where id=?";

name = jdbcTemplate.queryForObject(sql, **new** Object[]{id}, String.**class**);

}**catch**(EmptyResultDataAccessException e)

{

name="No name found ";

}

**return** name;

}

**public** List<String> getAllName() {

String sql = "select name from employee";

List<String> list = jdbcTemplate.queryForList(sql, String.**class**);

**return** list;

}

**public** List<String> getAllNamesBasedOnCharacter(String searchName) {

String sql = "select name from employee where name like ? ";

List<String> list = jdbcTemplate.queryForList(sql, String.**class**,"%" + searchName + "%");

**return** list;

}

**public** SqlRowSet queryForRowSet\_SqlRowSet(String tempId) {

String sql = "Select Name,address From Employee where id = ? " ; //

Object[] args = **new** Object[] {"id", tempId };

**int**[] argTypes = **new** **int**[] {Types.*VARCHAR*,Types.*VARCHAR*};

SqlRowSet rowSet = jdbcTemplate.queryForRowSet(sql, args, argTypes);

**return** rowSet;

}

**public** List<Employee> getAllEmployeeByResultSetExtractor(){

String query="select \* from employee order by id ";

**return** jdbcTemplate.query(query, **new** EmployeeResultSetExtractorMapper());

}

}

JDBCAllRowMapper.java

----------------------

**public** **class** JDBCAllRowMapper {

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

ApplicationContext ctx=**new** ClassPathXmlApplicationContext("org/ApplicationContext.xml");

EmployeeDAO dao=(EmployeeDAO)ctx.getBean("edao");

ArrayList empList =(ArrayList)dao.getAllEmployee();

Iterator itr=empList.iterator();

**while**(itr.hasNext())

{

System.*out*.println((Employee)itr.next());

}

}

}

EmployeeRowMapper.java:

**public** **class** EmployeeRowMapper **implements** RowMapper<Employee> {

**public** Employee mapRow(ResultSet rs, **int** rowNum) **throws** SQLException {

Employee employee = **new** Employee();

employee.setId(Integer.*parseInt*(rs.getString(1)));

employee.setName(rs.getString(2));

employee.setAddress(rs.getString(3));

**return** employee;

}

}

**JDBCSelectRowMapper.java**

------------------------------

**public** **class** JDBCSelectRowMapper {

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

ApplicationContext ctx=**new** ClassPathXmlApplicationContext("org/ApplicationContext.xml");

EmployeeDAO dao=(EmployeeDAO)ctx.getBean("edao");

Scanner sc=**new** Scanner(System.*in*);

System.*out*.println("Enter id which you want to retrieve:");

String id=sc.nextLine();

ArrayList empList =(ArrayList)dao.getEmployee(id);

Iterator itr=empList.iterator();

**while**(itr.hasNext())

{

System.*out*.println((Employee)itr.next());

}

System.*out*.println("No of Record Present:"+dao.employeeCount());

System.*out*.println("Name is:"+dao.getEmployeeNameById("6"));

System.*out*.println("Retrieving based on Name::");

ArrayList nameList =(ArrayList)dao.getAllName();

Iterator itr1=nameList.iterator();

**while**(itr1.hasNext())

{

System.*out*.println("Name is:"+(String)itr1.next());

}

System.*out*.println("Retrieving based on Input::");

System.*out*.println("Enter Character which you want to retrieve:");

String name=sc.nextLine();

ArrayList nList =(ArrayList)dao.getAllNamesBasedOnCharacter(name);

Iterator itr2=nList.iterator();

**while**(itr2.hasNext())

{

System.*out*.println("Name is:"+(String)itr2.next());

}

System.*out*.println("Enter id which you want to retrieve:");

String id1=sc.nextLine();

SqlRowSet row= dao.queryForRowSet\_SqlRowSet(id1);

**if**(row.next())

{

String name1=row.getString("name");

String address=row.getString("address");

System.*out*.println("Name ::::"+name1+" Address ::::"+address);

}

}

}

**EmployeeResultSetExtractorMapper.java**

**public** **class** EmployeeResultSetExtractorMapper **implements** ResultSetExtractor<List<Employee>>

{

**public** List<Employee> extractData(ResultSet rs) **throws** SQLException,

DataAccessException {

List<Employee> list=**new** ArrayList<Employee>();

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

Employee e=**new** Employee();

e.setId(rs.getInt(1));

e.setName(rs.getString(2));

e.setAddress(rs.getString(3));

list.add(e);

}

**return** list;

}

}

Spring PreparedStatement Demo:

**public** **class** EmployeeDAO {

**private** JdbcTemplate jdbcTemplate;

**public** **void** setJdbcTemplate(JdbcTemplate jdbcTemplate) {

**this**.jdbcTemplate = jdbcTemplate;

}

**public** Boolean save(**final** Employee e){

String query="insert into employee values (?,?,?)";

**return** jdbcTemplate.execute(query,**new** PreparedStatementCallback<Boolean>() {

**public** Boolean doInPreparedStatement(java.sql.PreparedStatement ps)

**throws** SQLException, DataAccessException {

ps.setInt(1,e.getId());

ps.setString(2,e.getName());

ps.setString(3,e.getAddress());

**return** ps.execute();

}

});

}

}

Stored Procedure:

SaveEmp.sql

create or replace procedure insertEmpData(eid in employee.id%type,

ename in employee.name%type,

eaddress in employee.address%type)

as

begin

insert into employee values(eid,ename,eaddress);

commit;

end insertEmpData;

/

EmployeeDAO.java

**public** **class** EmployeeDAO {

**private** JdbcTemplate jdbcTemplate;

**public** **void** setJdbcTemplate(JdbcTemplate jdbcTemplate) {

**this**.jdbcTemplate = jdbcTemplate;

}

**public** **void** insertDataUsingSP(Employee e)

{

String procName = "{call insertEmpData('"+e.getId()+"','"+e.getName()+"','"+e.getAddress()+"')}";

jdbcTemplate.update(procName);

}

}

NamedParameter JDBC Template:

<beans

xmlns=*"http://www.springframework.org/schema/beans"*

xmlns:xsi=*"http://www.w3.org/2001/XMLSchema-instance"*

xmlns:p=*"http://www.springframework.org/schema/p"*

xsi:schemaLocation=*"http://www.springframework.org/schema/beans http://www.springframework.org/schema/beans/spring-beans-3.0.xsd"*>

<bean id=*"ds"* class=*"org.springframework.jdbc.datasource.DriverManagerDataSource"*>

<property name=*"driverClassName"* value=*"oracle.jdbc.driver.OracleDriver"* />

<property name=*"url"* value=*"jdbc:oracle:thin:@localhost:1521:XE"* />

<property name=*"username"* value=*"haldia"* />

<property name=*"password"* value=*"nic"* />

</bean>

<bean id=*"namedParameterjdbcTemplate"* class=*"org.springframework.jdbc.core.namedparam.NamedParameterJdbcTemplate"*>

<constructor-arg>

<ref bean=*"ds"*></ref>

</constructor-arg>

</bean>

<bean id=*"namedParamdao"* class=*"org.NamedParamDAO"*>

<property name=*"namedParameterJdbcTemplate"* ref=*"namedParameterjdbcTemplate"*></property>

</bean>

</beans>

**public** **class** NamedParamDAO {

**private** NamedParameterJdbcTemplate namedParameterJdbcTemplate;

**public** NamedParamDAO() {

}

**public** NamedParameterJdbcTemplate getNamedParameterJdbcTemplate() {

**return** namedParameterJdbcTemplate;

}

**public** **void** setNamedParameterJdbcTemplate(NamedParameterJdbcTemplate namedParameterJdbcTemplate) {

**this**.namedParameterJdbcTemplate = namedParameterJdbcTemplate;

}

**public** **int** isnsertMethod(Employee e){

String query="insert into employee values(:id, :name, :address)";

Map paramMap=**new** HashMap();

paramMap.put("id",e.getId());

paramMap.put("name",e.getName());

paramMap.put("address",e.getAddress());

**return** namedParameterJdbcTemplate.update(query,paramMap);

}

}

package org;

import org.springframework.context.ApplicationContext;

import org.springframework.context.support.ClassPathXmlApplicationContext;

import org.springframework.jdbc.core.namedparam.NamedParameterJdbcTemplate;

import org.springframework.jdbc.datasource.DriverManagerDataSource;

public class NamedJdbcTest {

public NamedJdbcTest() {

}

public static void main(String[] args) {

ApplicationContext ctx=new ClassPathXmlApplicationContext("org/ApplicationContext.xml");

NamedParamDAO namedParamDAO=(NamedParamDAO)ctx.getBean("namedParamdao");

int status=namedParamDAO.isnsertMethod(new Employee("5","Suman","Bidhanpally"));

System.out.println(status);

}

}

InnerBean Demo:

<beans xmlns=*"http://www.springframework.org/schema/beans"*

xmlns:xsi=*"http://www.w3.org/2001/XMLSchema-instance"*

xsi:schemaLocation=*"http://www.springframework.org/schema/beans*

*http://www.springframework.org/schema/beans/spring-beans-3.0.xsd"*>

<bean id=*"outer"* class=*"com.Outer"*>

<property name=*"innner"*>

<bean class=*"com.Inner"*/>

</property>

</bean>

</beans>

Outer.java

**package** com;

**public** **class** Outer {

Inner innner;

**public** Inner getInnner() {

**return** innner;

}

**public** **void** setInnner(Inner innner) {

**this**.innner = innner;

}

**public** **void** check()

{

innner.display();

}

}

Inner.java

**package** com;

**public** **class** Inner {

**public** **void** display()

{

System.*out*.println("Calling the display method of Inner class");

}

**public** **void** m1()

{

System.*out*.println("Calling m1 method");

}

}

Properties:

**package** com;

**import** java.util.\*;

**public** **class** Emp {

**public** Properties addressProps;

**public** Properties getAddressProps() {

**return** addressProps;

}

**public** **void** setAddressProps(Properties addressProps) {

**this**.addressProps = addressProps;

}

}

package com;

import java.util.Enumeration;

import java.util.Iterator;

import java.util.List;

import java.util.Properties;

import java.util.Set;

import org.springframework.context.ApplicationContext;

import org.springframework.context.support.ClassPathXmlApplicationContext;

public class PropertiesInjectionMain {

public static void main(String[] args) {

ApplicationContext context = new ClassPathXmlApplicationContext("ApplicationContext.xml");

Emp ee=(Emp)context.getBean("eProps");

Properties props=ee.getAddressProps();

Enumeration<String> enums = (Enumeration<String>) props.propertyNames();

while (enums.hasMoreElements()) {

String key = enums.nextElement();

String value = props.getProperty(key);

System.out.println(key + " : " + value);

}

}

}

<beans xmlns=*"http://www.springframework.org/schema/beans"*

xmlns:xsi=*"http://www.w3.org/2001/XMLSchema-instance"*

xsi:schemaLocation=*"http://www.springframework.org/schema/beans*

*http://www.springframework.org/schema/beans/spring-beans-3.0.xsd"*>

<bean id=*"eProps"* class=*"com.Emp"*>

<property name=*"addressProps"*>

<props>

<prop key=*"1"*>Kolkata</prop>

<prop key=*"2"*>Delhi</prop>

<prop key=*"3"*>Mumbai</prop>

</props>

</property>

</bean>

</beans>

List Injection:

<beans xmlns=*"http://www.springframework.org/schema/beans"*

xmlns:xsi=*"http://www.w3.org/2001/XMLSchema-instance"*

xsi:schemaLocation=*"http://www.springframework.org/schema/beans*

*http://www.springframework.org/schema/beans/spring-beans-3.0.xsd"*>

<bean id=*"emp1"* class=*"com.Employee"*>

<property name=*"id"* value=*"1"* />

<property name=*"name"* value=*"Saswata Dhar"* />

<property name=*"address"* value=*"S.C Paul Road"* />

</bean>

<bean id=*"emp2"* class=*"com.Employee"*>

<property name=*"id"* value=*"2"* />

<property name=*"name"* value=*"Asim Das"* />

<property name=*"address"* value=*"Dumdum Road"* />

</bean>

<bean id=*"emp3"* class=*"com.Employee"*>

<property name=*"id"* value=*"3"* />

<property name=*"name"* value=*"Prasanta Dutta"* />

<property name=*"address"* value=*"Jadavpur Road"* />

</bean>

<bean id=*"eList"* class=*"com.Emp"*>

<property name=*"empList"*>

<list>

<ref bean=*"emp1"*/>

<ref bean=*"emp2"*/>

<ref bean=*"emp3"*/>

</list>

</property>

</bean>

</beans>

**package** com;

**import** java.util.List;

**public** **class** Emp {

**public** List<Employee> empList;

**public** List<Employee> getEmpList() {

**return** empList;

}

**public** **void** setEmpList(List<Employee> empList) {

**this**.empList = empList;

}

}

import java.util.Iterator;

import java.util.List;

import org.springframework.context.ApplicationContext;

import org.springframework.context.support.ClassPathXmlApplicationContext;

public class ListInjectionMain {

public static void main(String[] args) {

ApplicationContext context = new ClassPathXmlApplicationContext("ApplicationContext.xml");

Emp ee=(Emp)context.getBean("eList");

List empList=ee.getEmpList();

Iterator<Employee> itr=empList.iterator();

while(itr.hasNext())

{

Employee e=itr.next();

System.out.println("Id:"+e.getId()+ " Name:"+e.getName()+ " Address:"+e.getAddress());

}

}

}

**public** **class** Employee {

String id,name,address;

**public** String getId() {

**return** id;

}

**public** **void** setId(String id) {

**this**.id = id;

}

**public** String getName() {

**return** name;

}

**public** **void** setName(String name) {

**this**.name = name;

}

**public** String getAddress() {

**return** address;

}

**public** **void** setAddress(String address) {

**this**.address = address;

}

}