JDBC PART-7

Batch Updates:

This feature allows to execute more than one sql query at a time.

It is also called as batch processing.

java.sql. Statement

  public abstract void addBatch(String) throws SQLException;

=>It is used to add SQL query

  public abstract int[] executeBatch() throws SQLException;

=>It is used to execute all SQL queries

Example:

```
import java.sql.*;

class BatchDemo

{

        public static void main(String args[])

        {

                try{

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

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

Statement stmt=con.createStatement();
stmt.addBatch("insert into student values(12, 'lll', 34)");
stmt.addBatch("update student set marks=45 where rollno=5");
stmt.addBatch("delete from student where rollno=4");
stmt.executeBatch();
        }catch(Exception e)
        {
                System.err.println(e);
        }
        }
}
```

Advanced Data Types:

1) BLOB 2) CLOB

1) BLOB:

BLOB stands for Binary Large OBject

It is used to store/retrieve large amount of binary data as a single entity in/from database.

It supports text, image, graphics, animation, audio, video, .. etc.,

BLOB is mapped into java.sql.Blob interface in Java language.

2) CLOB:

CLOB stands for Character Large OBject

It is used to store/retrieve large amount of character data as a single entity in/from database.

It supports text only.

CLOB is mapped into java.sql.Clob interface in Java language.

BLOB Example:

```
import java.io.*;

import java.sql.*;

class BlobDemo

{

public static void main(String args[])

{

try{

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

Connection

con=DriverManager.getConnection("jdbc:oracle:thin:@localhost:1521:x
e","syste

m","manager");

PreparedStatement pstmt=con.prepareStatement("insert into images
values(?,

?)");

pstmt.setString(1, args[0]);

FileInputStream fis=new FileInputStream(args[1]);

pstmt.setBinaryStream(2, fis, fis.available());

pstmt.executeUpdate();
System.out.println("One Image Inserted Successfully");
}catch(Exception e)
{
System.err.println(e);
}
}
}
```

Rowsets:

A rowset is an object that encapsulates set of rows from database.

Rowset is generated based on sql query.

Whenever rowset is generated then rowset pointer or cursor points to before first record.

There are 5 rowsets:

1) JdbcRowSet

2) CachedRowSet

3) WebRowSet

4) FilteredRowSet

5) JoinRowSet

The above rowsets are interfaces in javax.sql.rowset package.

The differences between ResultSet & Rowsets:

ResultSet	Rowsets
1) It is a non serializable object.	1) All rowsets are serializable objects.

2) It is a connected object.	2) JdbcRowSet is a connected object and remaining rowsets are disconnected objects.
3) It is not a javabean.	3) All rowsets are javabeans.
4) In order to get ResultSet we need Connection interface, DriverManager class & Statement interface.	4) In order to get RowSet we need RowSet implementation class only.

Serialization:

A serialization is a process of converting object into a series of bits.

In Java, object must be serializable to do the following operations:

1) Writing an object to a file.

2) Reading an object from file.

3) Writing an object to a network.

4) Reading an object from network.

Class must implements java.io.Serializable interface to make serializable object.

java.io.Serializable interface is a marker interface, tag interface or empty interface because it does not contain members.

Connected Object:

Connected object means the object always being connected to a database.

Java Bean:

A Java bean is a reusable software component.

A Java class is said to be java bean if it follows the following rules:

1) Class must be public

2) Class must implements java.io.Serializable interface

3) Class must be under package.

4) Class must contain one public default constructor.

5) All instance variables must be private(Instance variables are called properties).

6) Each & every property must contain setter & getter methods.

7) All setter & getter methods must be public.

Example:

```
package demo;
import java.io.*;
public class MessageBean implements Serializable
{
private String message;
public void setMessage(String message)
{
this.message=message;
}
public String getMessage()
{
        return message;
}
}
```

The above all rowsets are interfaces in javax.sql.rowset package.

The above all rowset interfaces are implemented by database vendors.

Oracle corporation implemented the above rowset interfaces in

the following classes:

1) OracleJDBCRowSet

2) OracleCachedRowSet

3) OracleWebRowSet

4) OracleFilteredRowSet

5) OracleJoinRowSet

The above classes are in oracle.jdbc.rowset package.

oracle.jdbc.rowset package in ojdbc6_g.jar file or ojdbc8_g.jar file in oracle database.

JdbcRowSet is a connected rowset where as CachedRowSet is a disconnected rowset.

WebRowSet is same as CachedRowSet and additionally it allows to write WebRowSet data to xml file.

FilteredRowSet is same as CachedRowSet and additionally it allows filtering operations on RowSet data.

JoinRowSet is same as CachedRowSet and additionally it allows to join two or more rowsets data.

JdbcRowSet Example:

```
import java.io.*;

import javax.sql.rowset.*;

import oracle.jdbc.rowset.*;

class JRSDemo

{

public static void main(String args[])

{

try{

JdbcRowSet jrs=new OracleJDBCRowSet();

jrs.setUrl("jdbc:oracle:thin:@localhost:1521:xe");

jrs.setUsername("system");

jrs.setPassword("manager");

jrs.setCommand("select * from student");

jrs.execute();

while(jrs.next())

{

System.out.print(jrs.getInt("rollno")+"\t");

System.out.print(jrs.getString("name")+"\t");

System.out.println(jrs.getInt("marks"));
}
}catch(Exception e)
{
System.err.println(e);
}
}
}
```

CachedRowSet Example:

```
import java.io.*;
import javax.sql.rowset.*;
import oracle.jdbc.rowset.*;
class CRSDemo
{
public static void main(String args[])
{
try{
CachedRowSet crs=new OracleCachedRowSet();
crs.setUrl("jdbc:oracle:thin:@localhost:1521:xe");
crs.setUsername("system");
crs.setPassword("manager");
crs.setCommand("select * from student");
crs.execute();
while(crs.next())
{
System.out.print(crs.getInt("rollno")+"\t");
System.out.print(crs.getString("name")+"\t");
System.out.println(crs.getInt("marks"));
}
}catch(Exception e)
{
System.err.println(e);
}
}
}
```

WebRowSet Example:

```
import java.io.*;
import javax.sql.rowset.*;
import oracle.jdbc.rowset.*;
class WRSDemo
{
public static void main(String args[])
{
try{
WebRowSet wrs=new OracleWebRowSet();
wrs.setUrl("jdbc:oracle:thin:@localhost:1521:xe");
wrs.setUsername("system");
wrs.setPassword("manager");
wrs.setCommand("select * from student");
wrs.execute();
FileOutputStream fos=new FileOutputStream("student.xml");
wrs.writeXml(fos);
}catch(Exception e)
{ System.err.println(e);
}
}
}
```

By

Mr. Venkatesh Mansani

Naresh i Technologies