Batch Processing in JDBC

Instead of executing a single query, we can execute a batch (group) of queries. It makes the performance fast.

The java.sql.Statement and java.sql.PreparedStatement interfaces provide methods for batch processing.

Advantage of Batch Processing

Fast Performance

Methods of Statement interface

The required methods for batch processing are given below:

Method	Description
void addBatch(String query)	It adds query into batch.
int[] executeBatch()	It executes the batch of queries.

Example of batch processing in jdbc

Let's see the simple example of batch processing in jdbc. It follows following steps:

- Load the driver class
- Create Connection
- Create Statement
- Add query in the batch
- Execute Batch
- Close Connection

```
import java.sql.*;
class FetchRecords{
public static void main(String args[])throws Exception{
Class.forName("oracle.jdbc.driver.OracleDriver");
Connection con=DriverManager.getConnection("jdbc:oracle:thin:@localhost:1521:xe","system","oracle");
con.setAutoCommit(false);

Statement stmt=con.createStatement();
stmt.addBatch("insert into user420 values(190,'abhi',40000)");
stmt.addBatch("insert into user420 values(191,'umesh',50000)");

stmt.executeBatch();//executing the batch
```

```
con.commit();
con.close();
}}
```

If you see the table user420, two records has been added.

Example of batch processing using PreparedStatement

```
import java.sql.*;
import java.io.*;
class BP{
public static void main(String args[]){
try{

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

PreparedStatement ps=con.prepareStatement("insert into user420 values(?,?,?)");

BufferedReader br=new BufferedReader(new InputStreamReader(System.in));
while(true){

System.out.println("enter id");
String s1=br.readLine();
int id=Integer.parseInt(s1);

System.out.println("enter name");
String name=br.readLine();

System.out.println("enter salary");
String s3=br.readLine();
int salary=Integer.parseInt(s3);

ps.setInt(1,id);
ps.setString(2,name);
ps.setInt(3,salary);

ps.addBatch();
System.out.println("Want to add more records y/n");
String ans=br.readLine();
if(ans.equals("n")){
```

```
break;
}

}
ps.executeBatch();

System.out.println("record successfully saved");

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

}}
```

It will add the queries into the batch until user press n. Finally it executes the batch. Thus all the added queries will be fired.