**Handle ResultSet Data Manipulation Efficiently**

|  |  |  |  |
| --- | --- | --- | --- |
| **Created by –** | **Role** | | **Date of Creation** |
| Arpit Kumar Tiwari | Senior Software Developer | | 14th June 2016 |
| **Last Revised by** | **Role** | | **Date of Revision** |
|  |  | |  |
| **Reviewed by** | **Name** | | **Role** |
|  | Suresh C | | Project Lead |
| **Approval by** | **Name** | | **Role** |
|  | Suresh C | | Project Lead |
| **Circulation List** | All employees of CAPGEMINI | **Version Number** | 1.0 |

Contents

1. [Preface: ----------------------------------------------------------------------------------------------------------------------------------------------------------------------- 3](#_Toc454737808)
2. [ResultSetHandler: ---------------------------------------------------------------------------------------------------------------------------------------------------------- 3](#_Toc454737809)
3. [How to implement ResultSetHandler : -------------------------------------------------------------------------------------------------------------------------------- 3](#_Toc454737810)
4. [References: ------------------------------------------------------------------------------------------------------------------------------------------------------------------ 4](#_Toc454737811)
5. [Revision history-------------------------------------------------------------------------------------------------------------------------------------------------------------- 4](#_Toc454737812)

# Preface:

The most important and crucial thing in an application is “performance”, How an application is performing, how much time is it taking is all what matters once the application is Up. There are many ways like management tools, code optimization Database optimization, to optimize the performance of an application. But In this article we’ll focus only on 1 particular way –

Consider one scenario where you have thousands of rows in your ResultSet object and you want to perform some operations based on the values of ResultSet. For example:

Let’s say, a SQL query returns 100 columns each having 1000 records which later stored in ResultSet object. Now you want to change the order of columns from java based on some list. Then in general scenario what you’ll do is you start comparing the values of ResultSet by using nesting of loop. Like below

|  |
| --- |
| ArrayList columnName;//assume this ArrayList is initialized with list of column names  String value = null;  ResultSetMetaData rsMeta = rs.getMetaData();  while(rs.next()){  int colCount = rsMeta.getColumnCount();  for(int index=0;index < columnName.size();index++){  for(int counter = 0 ; counter< colCount ; counter++){  if(columnName.get(index).equalsIgnoreCase(  rs.getString(rsMeta.getColumnName(counter+1)))){  value = rs.getString(rsMeta.getColumnName(counter+1));  }  }  }  } |

By looking the code itself, you can easily guess that this will take lots of time hence results in performance inefficiency because we are comparing each column with the each value of list.

In order to get rid you such mess ResultSetHandler will come in handy.

# ResultSetHandler:

ResultSetHandler is an interface resides in commons-dbutils.jar file in the org.apache.commons.dbutils package. Implementations of this interface will convert ResultSets into other objects.

ResultSetHandler contains one method –

public Object handle(ResultSet rs) throws SQLException

**Parameter** – ResultSet which you want to convert.

**Return type** – Object type in which you want to convert e.g. ArrayList, String, and HashMap etc.

# How to implement ResultSetHandler :

In this example we are converting a result set object into an HashMap where the key will be column name and value will be its value.

1. Create a class having the following code.

|  |
| --- |
| public class ConvertResultSet implements ResultSetHandler {  @Override  public HashMap<String, String> handle(ResultSet rs) throws SQLException {  HashMap<String, String> row = new HashMap<String, String>();  ResultSetMetaData meta = rs.getMetaData();  int cols = meta.getColumnCount();  for (int i = 0; i < cols; i++) {  row.put(meta.getColumnName(i+1), rs.getObject(i+1).toString());  }  return row;  } |

1. Use the above class to convert ResultSet into Object(in our case HashMap)

|  |
| --- |
| ConvertResultSet convertResultSet = **new** ConvertResultSet();  HashMap<String, String> invoiceDetail = convertResultSet.handle(rs); |

1. Now instead of a ResultSet you have a HashMap with column name as key so without doing comparison you can easily pass the column name as key and get the value.

|  |
| --- |
| String value = invoiceDetail.get(columnName); |

# References:

<https://commons.apache.org/proper/commons-dbutils/apidocs/org/apache/commons/dbutils/ResultSetHandler.html>

<https://commons.apache.org/proper/commons-dbutils/examples.html>

<https://gist.github.com/jesperdj/888501>

# Revision history

|  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- |
| **#** | **Date** | **Version #** | **Section Changed** | **Details of changes made** | **Changes made by** | **Approved By** |
| 1. | 14-June-2016 | 1.0 | All | Original Version | Arpit Tiwari | - |
| 2. | 27-June-2016 | 1.1 | References  And Table of Content | Added references | Arpit Tiwari | - |