

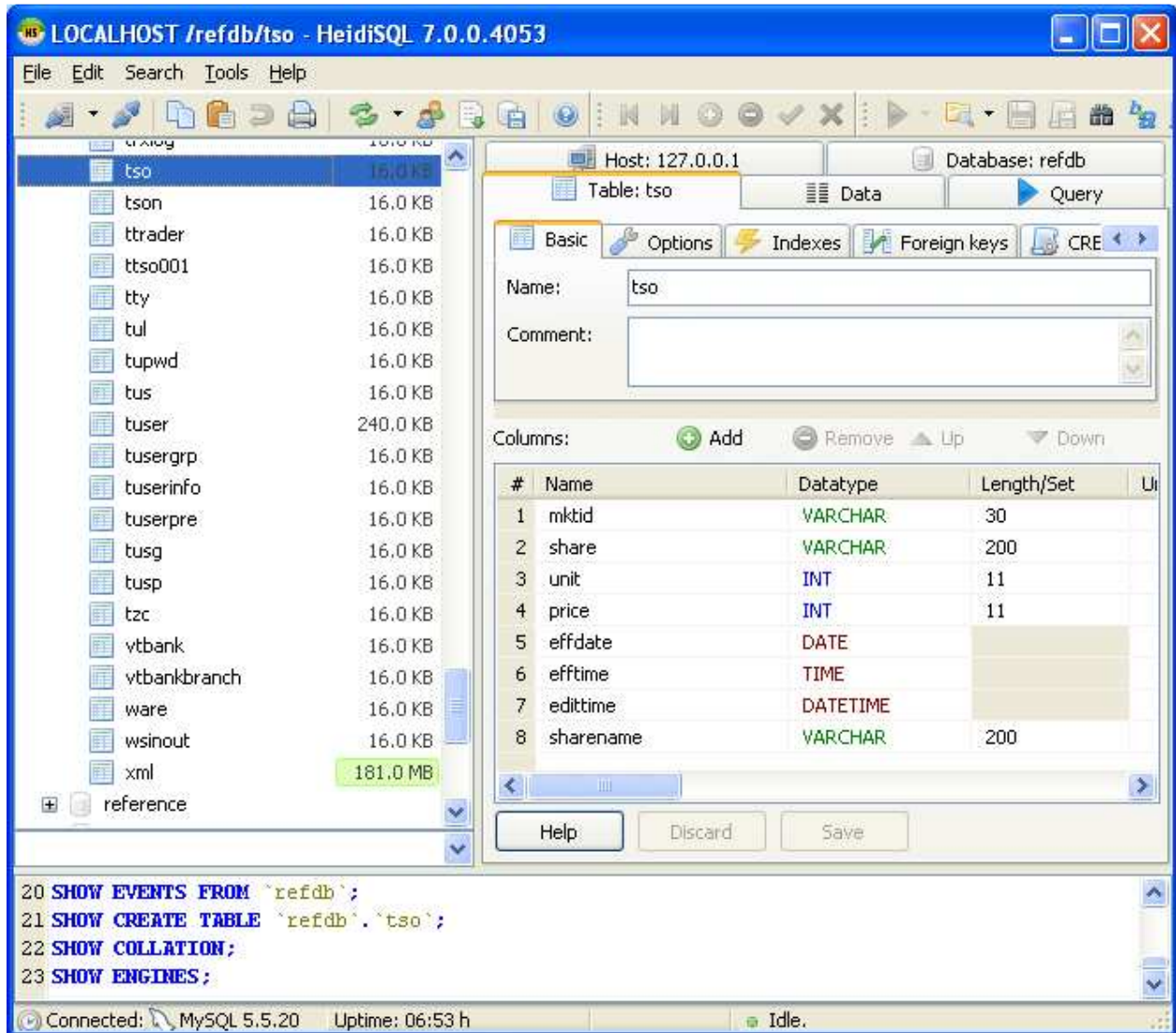
## XDoc Report Manual Guide

### Requirement

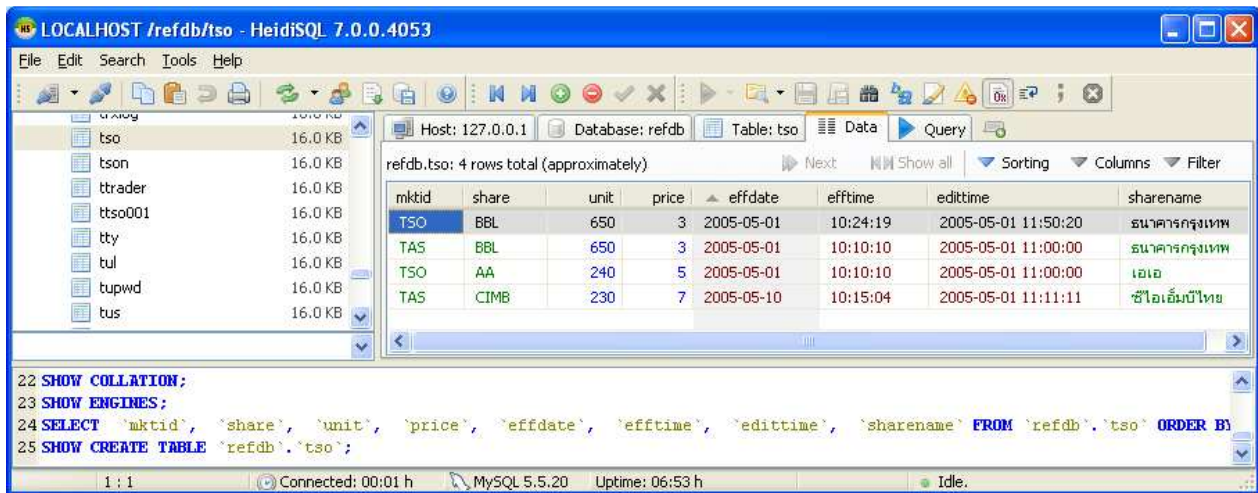
+ beancore.jar, jreport.jar, xreport.jar and xdoc library

### I. Create Multiple Word & PDF Document from data source

We have table tso schema like this

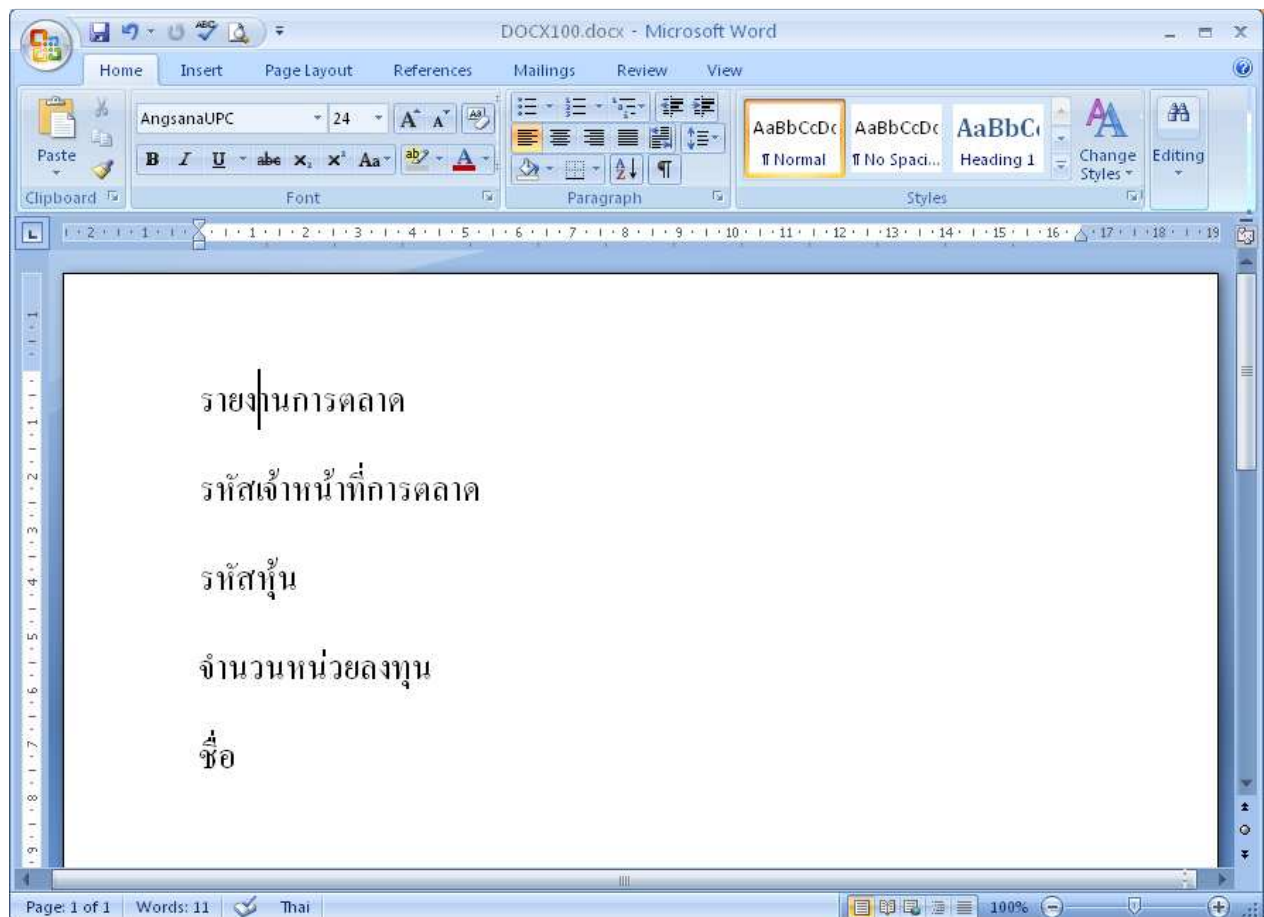


And data in this table



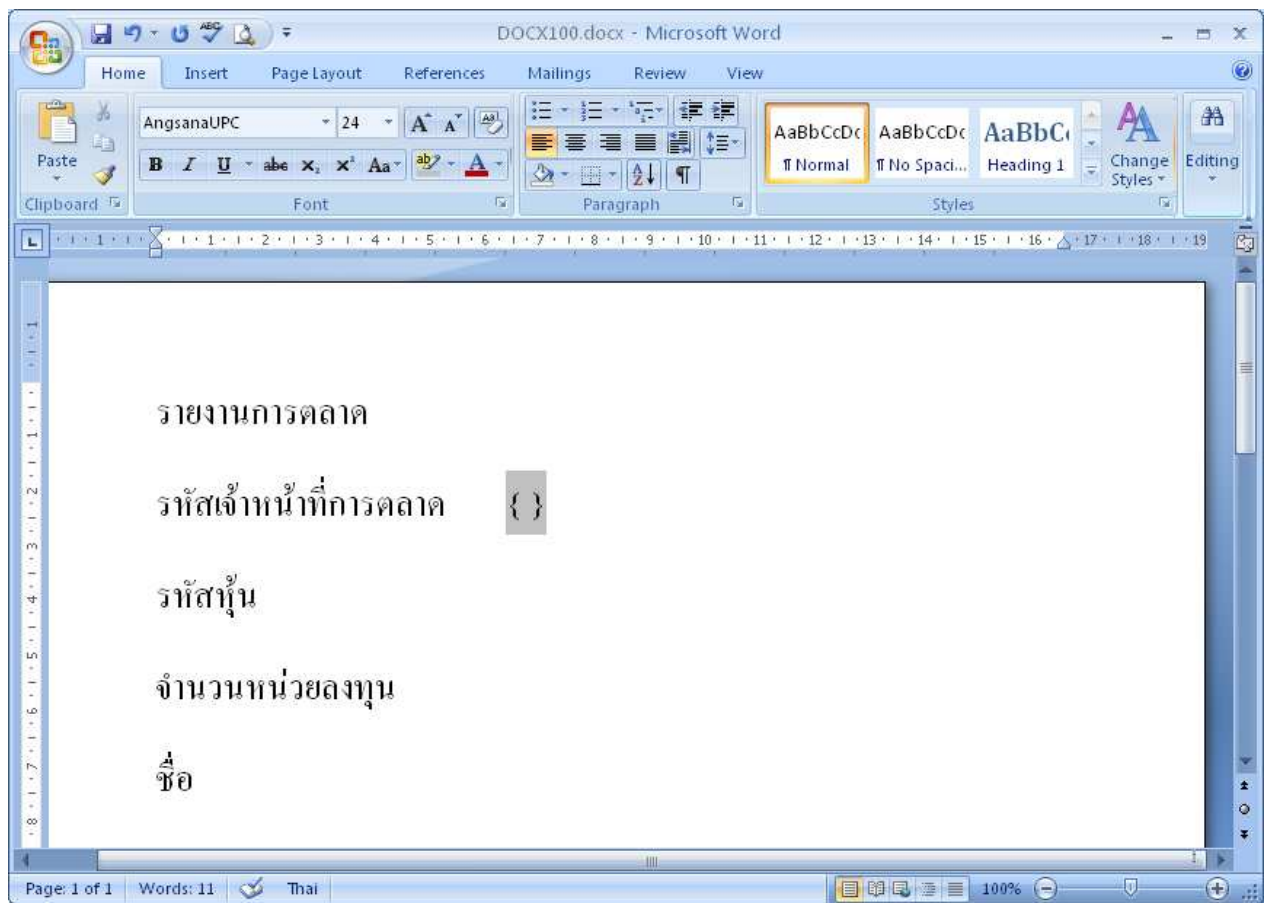
## Step by Step

1. Create docx template by Open MS Word and then create your report layout like below with AngsanaUPC 24 point save as DOCX100.docx



2. We need to defined merge field mktid, share, unit & sharename (same name as in db) into docx template file (DOCX100.docx)

a. Cursor after รหัสเจ้าหน้าที่การตลาด and press Ctrl+F9, {} will be display



b. Right mouse -> Edit Field

c. Scroll to select MergeField and specify Field name = \$mktid (need \$ sign preceeding field name) and Press OK

Field

Please choose a field

Categories:

(All)

Field names:

If

IncludePicture

IncludeText

Index

Info

Keywords

LastSavedBy

Link

ListNum

MacroButton

MergeField

MergeRec

MergeSeq

Next

NextIf

NoteRef

NumChars

NumPages

Description:

Insert a mail merge field

Field Codes

Field properties

Field name:

\$mktid

Format:

(none)

Uppercase

Lowercase

First capital

Title case

Field options

☐ Text to be inserted before:

☐ Text to be inserted after:

☐ Mapped field

☐ Vertical formatting

☒ Preserve formatting during updates

OK

Cancel

DOCX100.docx - Microsoft Word

Home

Insert

Page Layout

References

Mailings

Review

View

AngsanaUPC

24

A

A

AB

B

I

U

abe

x

x

Aa

ab

A

Font

Normal

No Spaci...

Heading 1

Change Styles

1

2

3

4

5

6

7

8

9

10

11

12

13

14

15

16

17

18

19

รายงานการตลาด

รหัสเจ้าหน้าที่การตลาด <<\$MKTID>>

รหัสรุ่น

จำนวนหน่วยลงทุน

ชื่อ

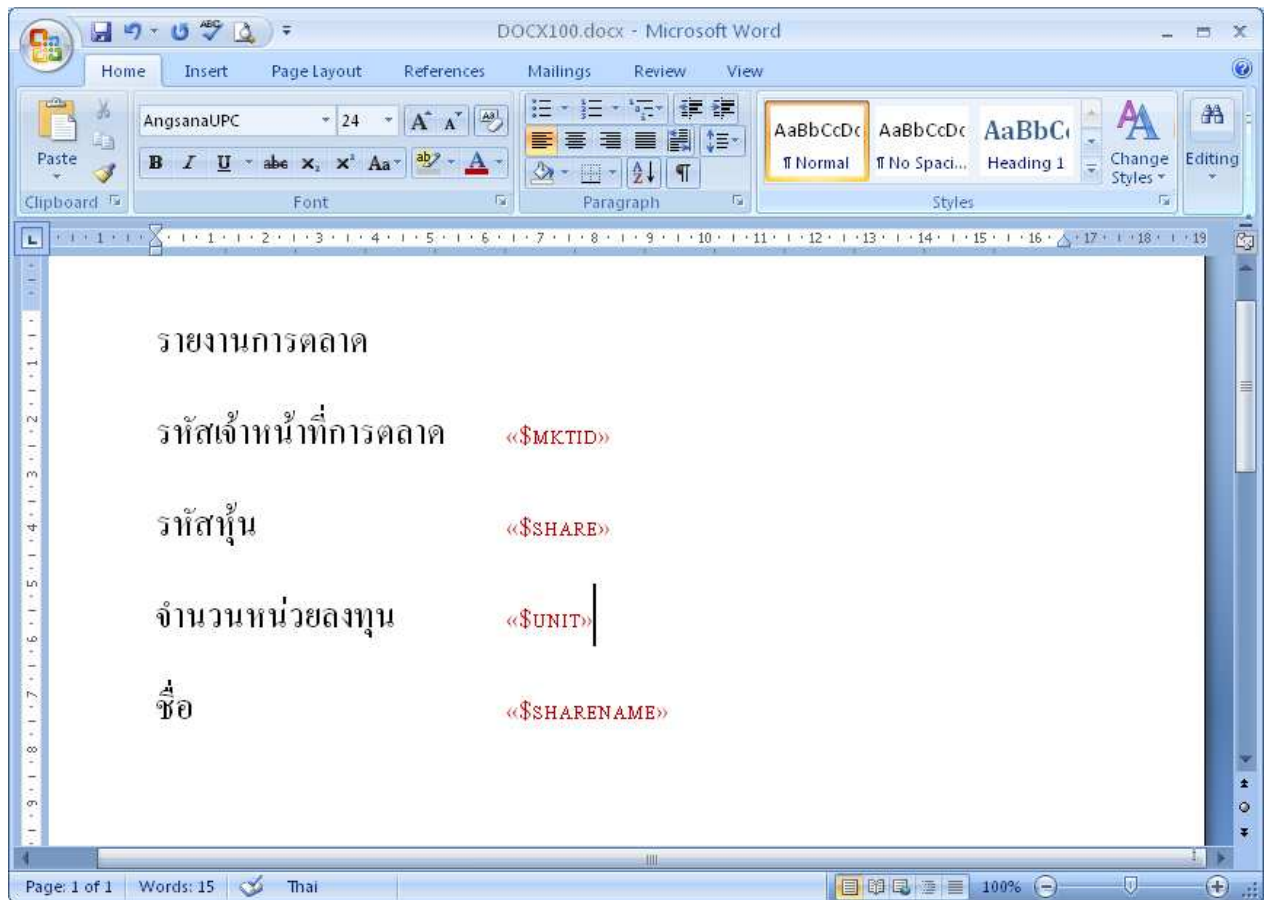
Page: 1 of 1

Words: 12

Thai

100%

- d. Repeat step a to c with share, unit & sharename field name (you can change font color or style to emphasis) and then save changed

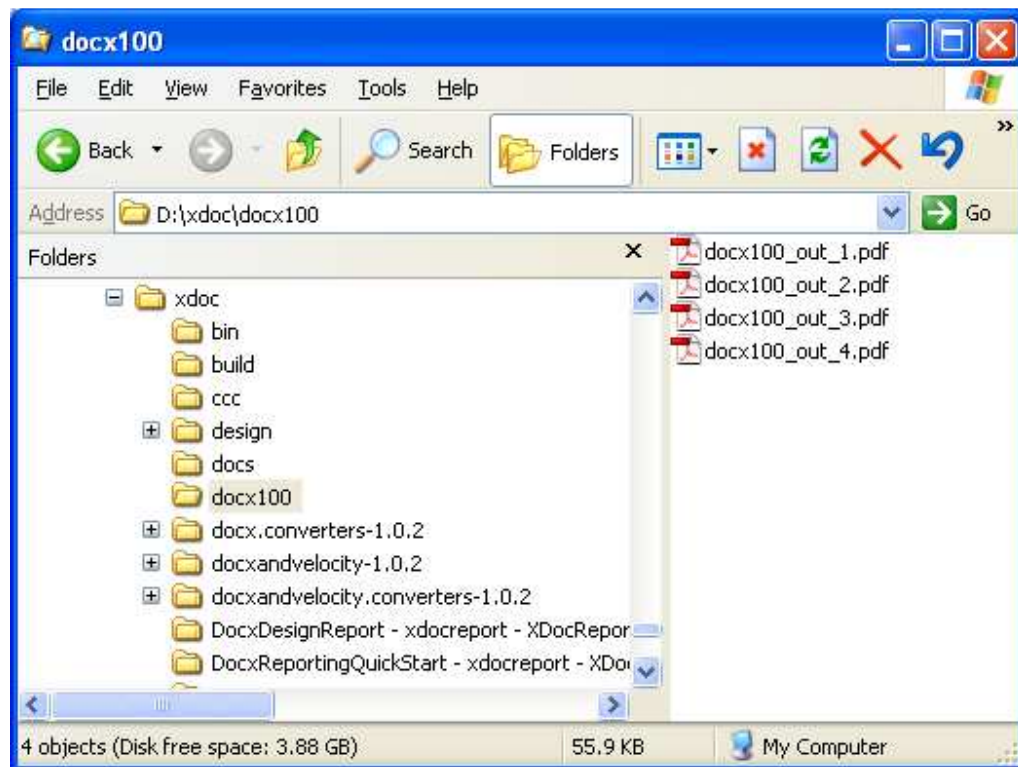


3. Now after finish docx layout (DOCX100.docx) try to render report by java engine MXReport tool (Multi XReport)  
Examine

```
java com/fs/dev/report/MXReport -ms REFDB -layout d:\xdoc\tso\DOCX100.docx -doc  
d:\xdoc\docx100\docx100_out.pdf -sql "select * from tso"
```

the result will be on folder d:\xdoc\docx100 like this

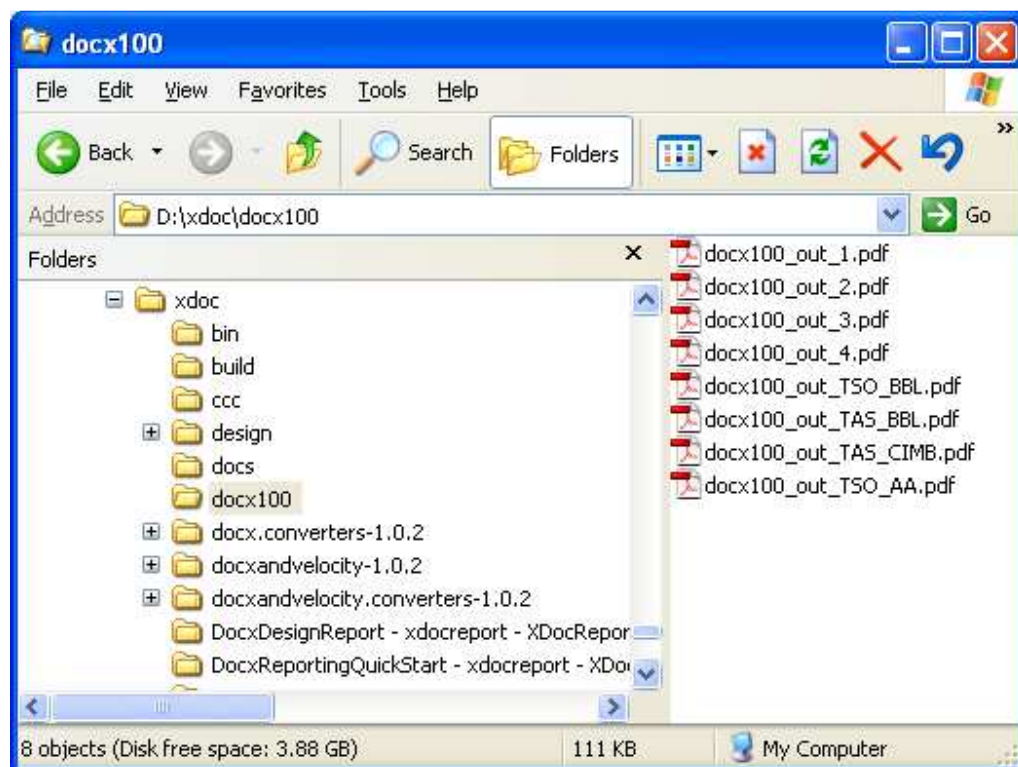




Since the engine will generate file depend on number of record in result set and using index counter as a part of file name

Examine

```
java com/fs/dev/report/MXReport -ms REFDB -layout d:\xdoc\tso\DOCX100.docx -doc
d:\xdoc\docx100\docx100_out.pdf -sql "select * from tso" -key "mktid,share" -delim "_"
```

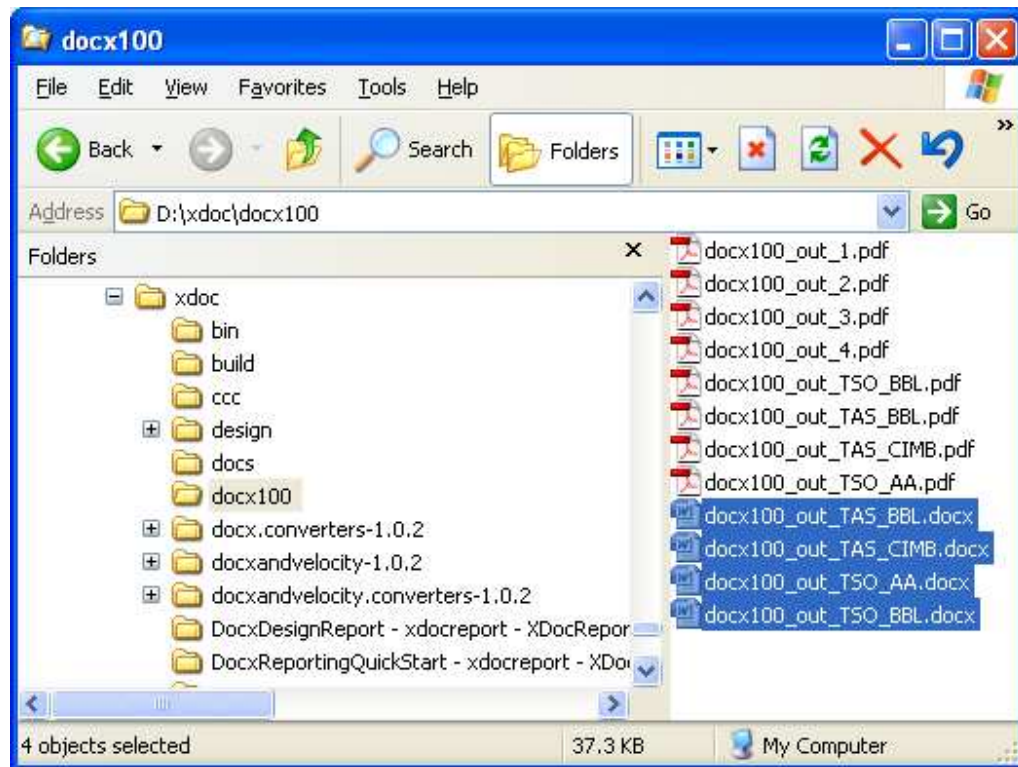


The result will use keys mktid & share with delimiter ( ) as a part of file name

Since the default extension renderer is PDF file, you can change to the docx report output by option -t DOCX

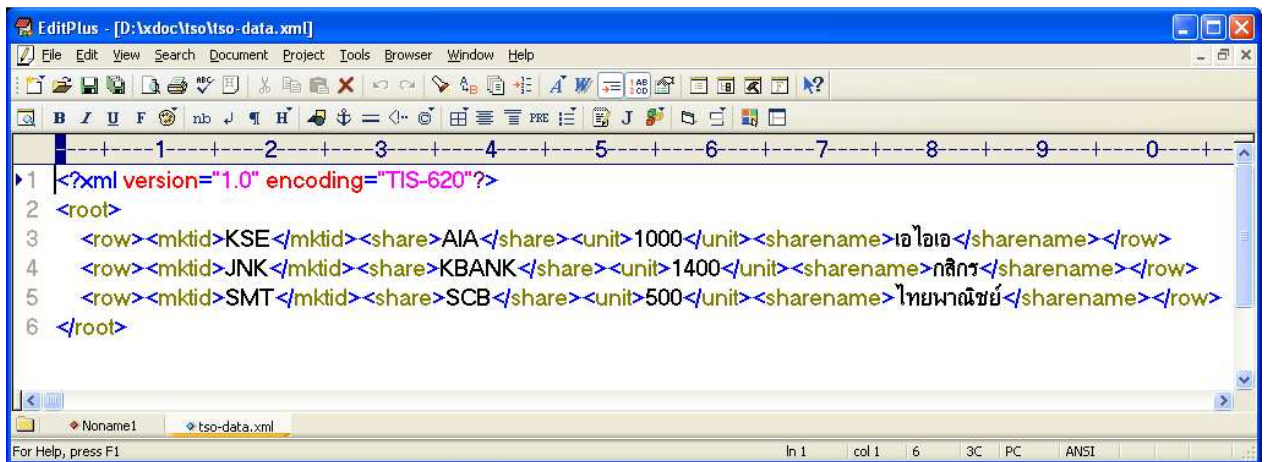
Examine

```
java com/fs/dev/report/MXReport -ms REFDB -layout d:\xdoc\tso\DOCX100.docx -doc  
d:\xdoc\docx100\docx100_out.docx -sql "select * from tso" -key "mktid,share" -delim " " -t DOCX
```



The result will be word document file as your need

4. Try out with xml data
  - a. If you have xml data file like this (tso-data.xml)

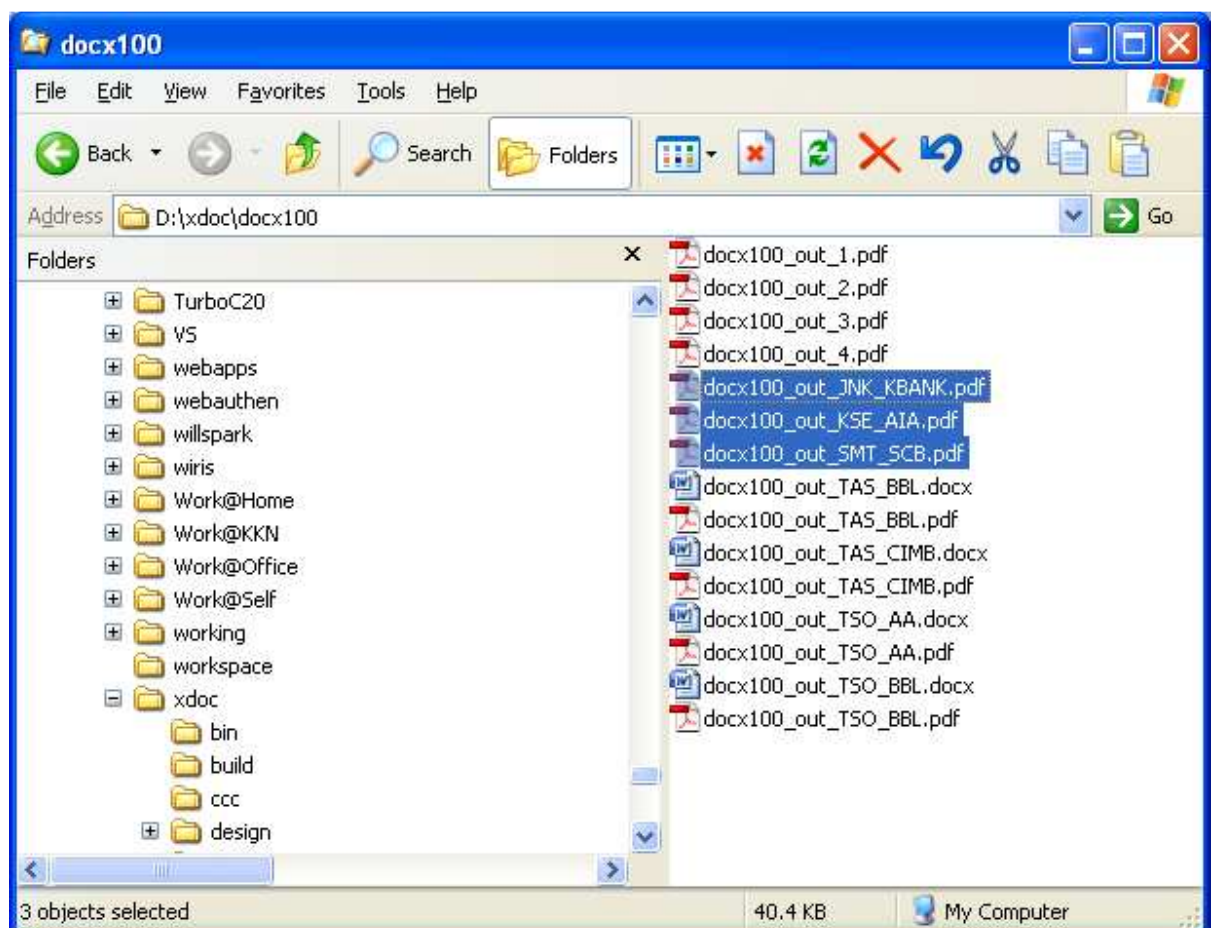


```

1 <?xml version="1.0" encoding="TIS-620"?>
2 <root>
3   <row><mktid>KSE</mktid><share>AIA</share><unit>1000</unit><sharename>เอไอเอ</sharename></row>
4   <row><mktid>JNK</mktid><share>KBANK</share><unit>1400</unit><sharename>กสิกร</sharename></row>
5   <row><mktid>SMT</mktid><share>SCB</share><unit>500</unit><sharename>ไทยพาณิชย์</sharename></row>
6 </root>

```

- b. Then examine with xml file
- java com/fs/dev/report/MXReport -ms REFDB -layout d:\xdoc\tso\DOCX100.docx -doc  
d:\xdoc\docx100\docx100\_out.pdf -key "mktid,share" -delim "\_" -f D:\xdoc\tso\tso-data.xml





Caution: PDF renderer is hard to say that font family may frustrated , so you need to config font file by font\_config.properties using default font name specify in your location existed

Ex.

```
DEFAULT=c:\\windows\\fonts\\ANGSA.ttf
DEFAULT_BOLD=c:\\windows\\fonts\\ANGSAB.ttf
DEFAULT_ITALIC=c:\\windows\\fonts\\ANGSAI.ttf
DEFAULT_UNDERLINE=c:\\windows\\fonts\\ANGSAU.ttf
DEFAULT_STRIKE=c:\\windows\\fonts\\ANGSAZ.ttf
```

In this case that why document layout file (DOCX100.docx) need AngsanaUPC font family

## II. Create XDoc from BeanData (JSPP Template)

Data source can be made from BeanData

Now we have TSOData.java class extends from BeanData as default in collect method it query from tso table like this

```
public int collect(java.sql.Connection connection,java.sql.Connection
centerConnection,java.sql.Connection globalConnection,java.util.Map transientVar) throws
Exception {
    removeAll();
    ExecuteStatement sql = createQueryForCollect(connection);
    if(getMktid()!=null && getMktid().trim().length()>0) {
        sql.append(" where mktid like '"+getMktid()+"%' ");
    }
    int result = 0;
    java.sql.ResultSet rs = sql.executeQuery(connection);
    while(rs.next()) {
        result++;
        TSOData aTSOData = new TSOData();
        aTSOData.fetchResult(rs);
        add(aTSOData);

        put(aTSOData); //masters element
    }
    close(rs);
    return result;
}
```

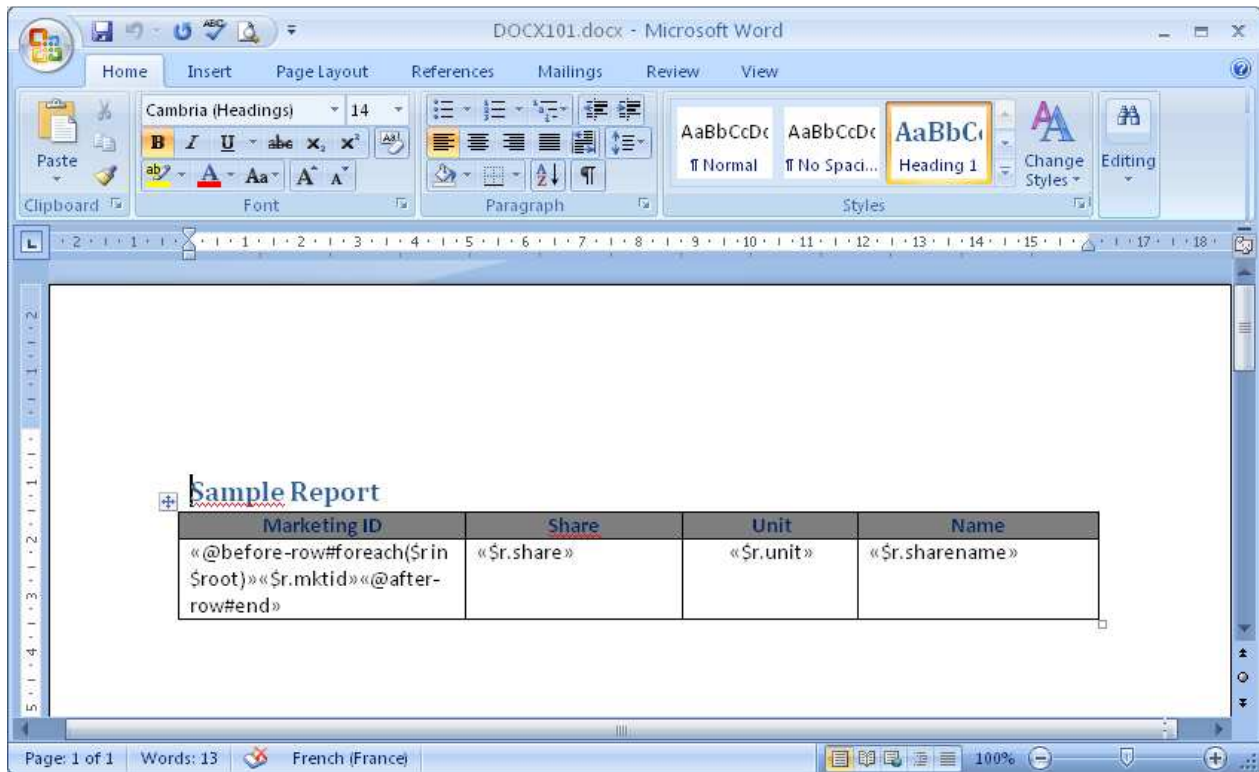
Then we can run report with –provider option specified by BeanData class name

Ex.

```
java com/fs/dev/report/MXReport -ms REFDB -layout d:\xdoc\tso\DOCX001.docx -doc
d:\xdoc\tso\docx001_out.pdf -provider com.fs.bean.TSOData
```

### III. Create Single Word & PDF from Data Source

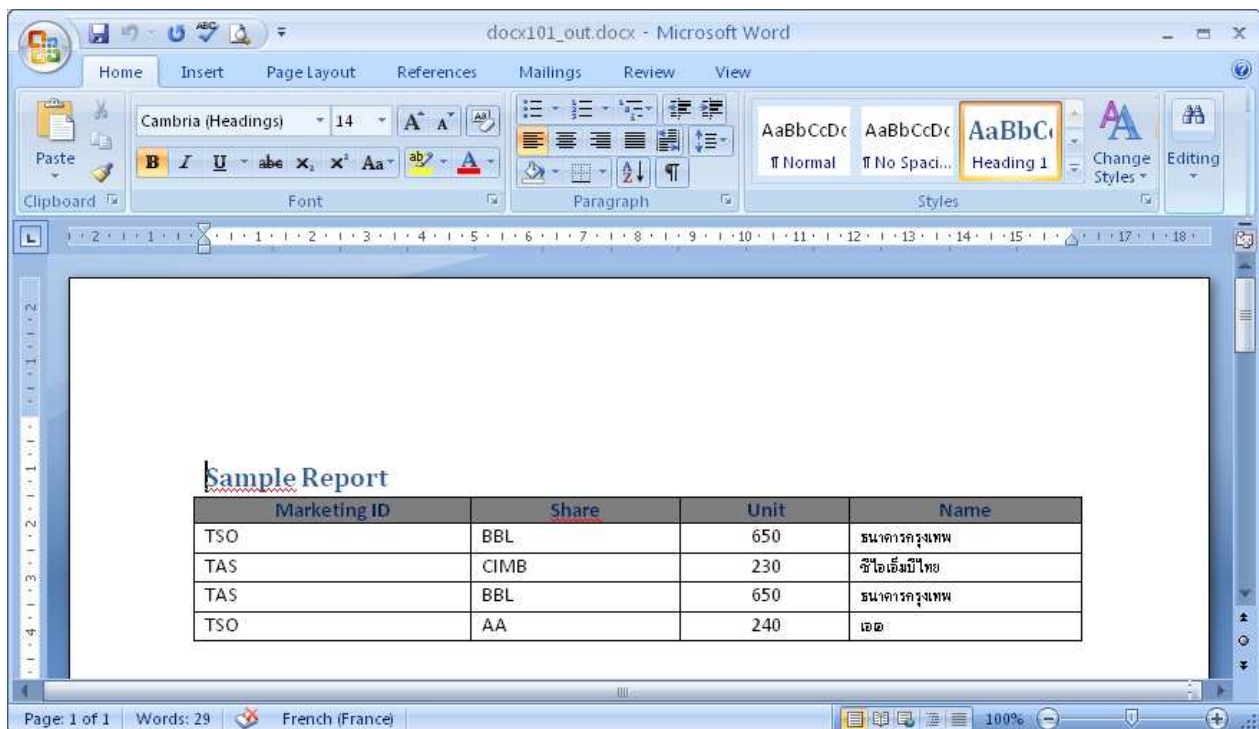
See layout file DOCX101.docx



This will be script foreach element into documentation (root tag as default repository stored name)

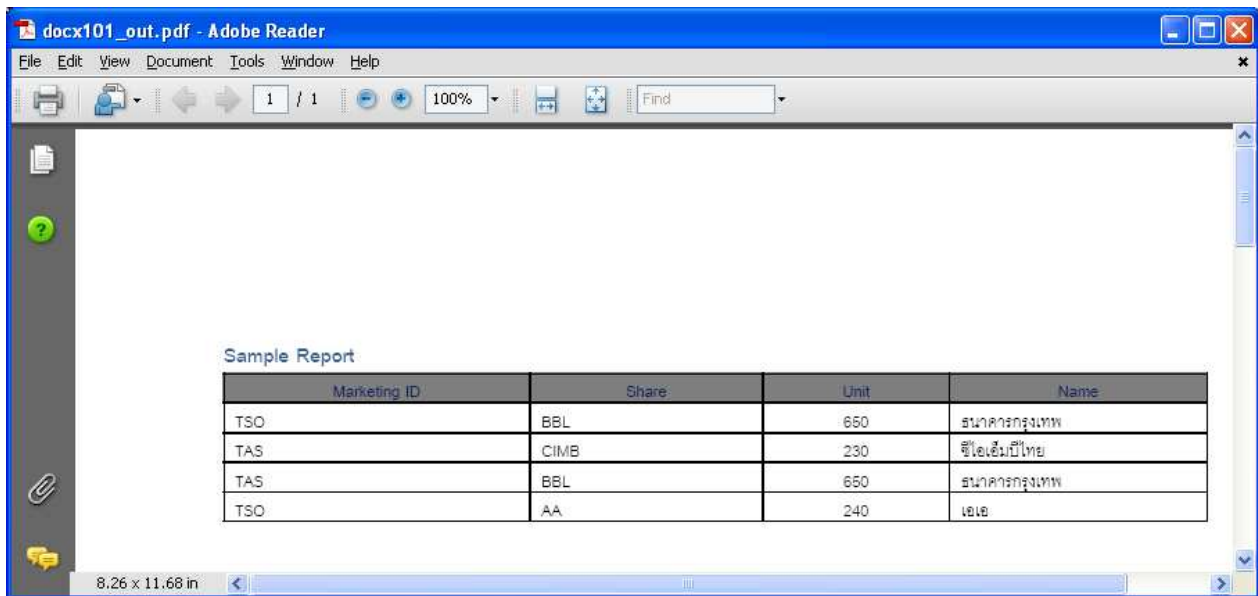
Examine

```
java com/fs/dev/report/JXReport -layout d:\xdoc\tso\DOCX101.docx -doc d:\xdoc\tso\docx101_out.docx -sql  
"select * from tso" -defcon true
```



Default renderer is docx then change to PDF

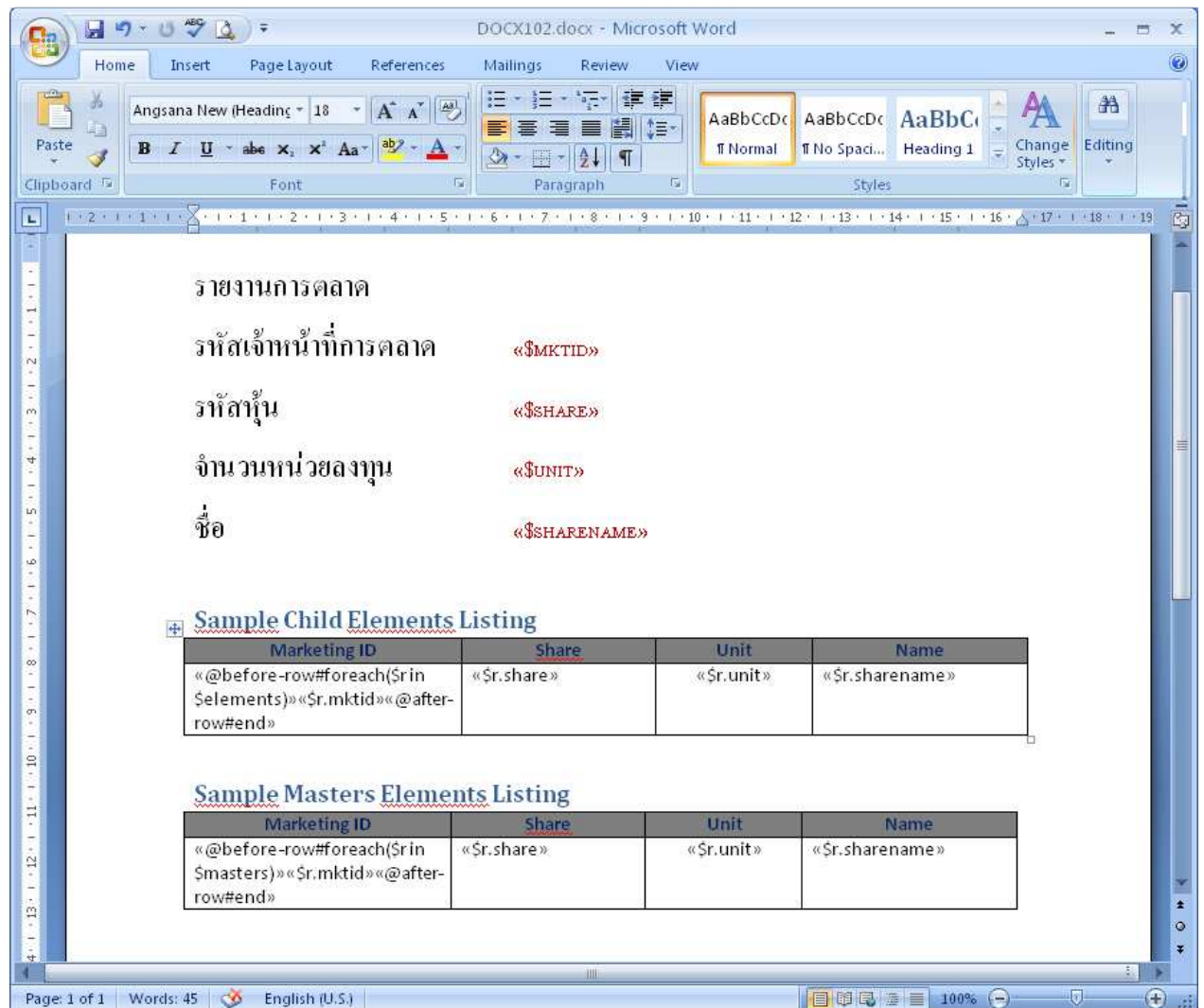
```
java com/fs/dev/report/JXReport -layout d:\xdoc\tso\DOCX101.docx -doc d:\xdoc\tso\docx101_out.pdf -sql  
"select * from tso" -defcon true -t PDF
```





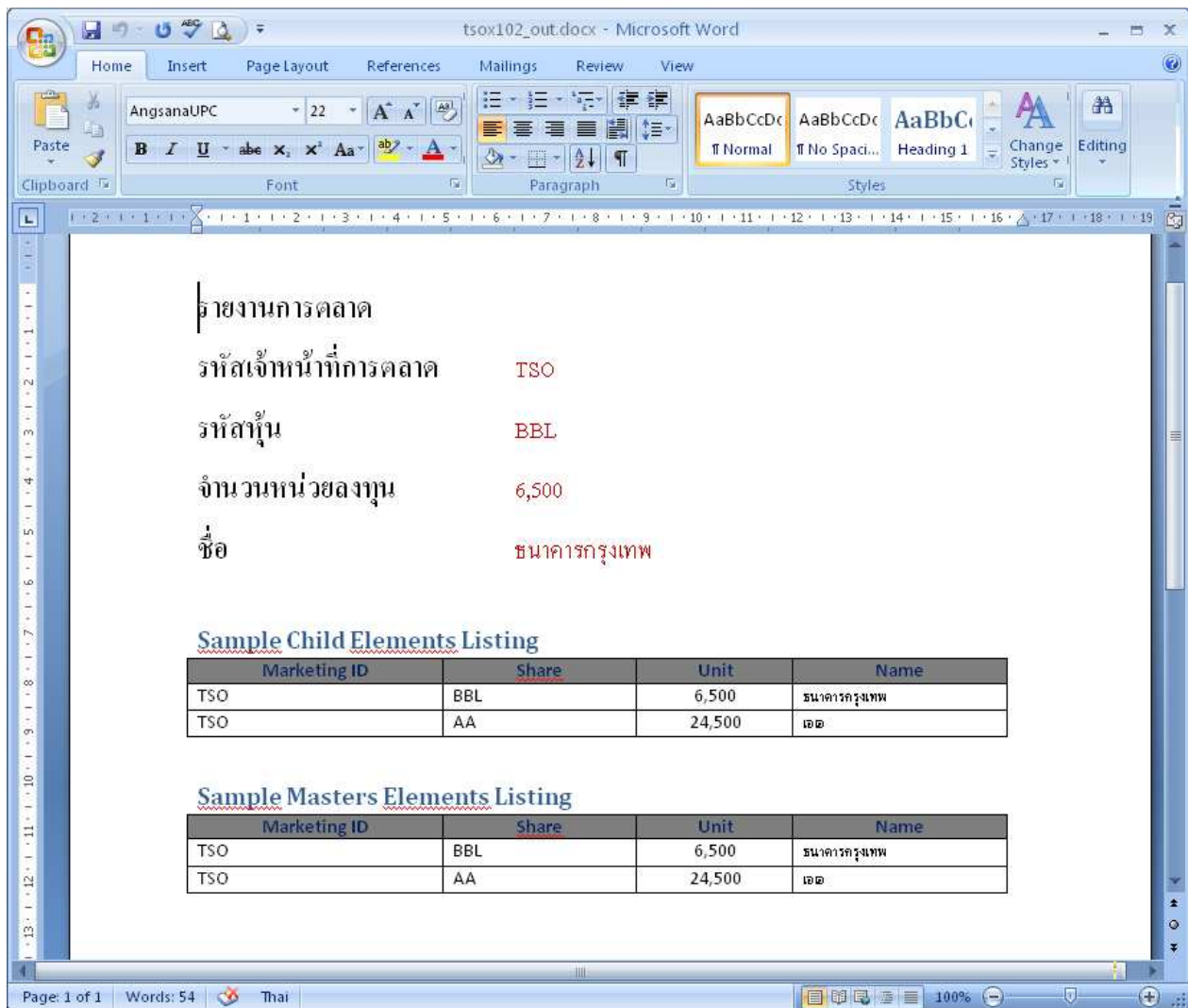
#### IV. Create Single Document with Multiple Element Listing

See layout file DOCX102.docx



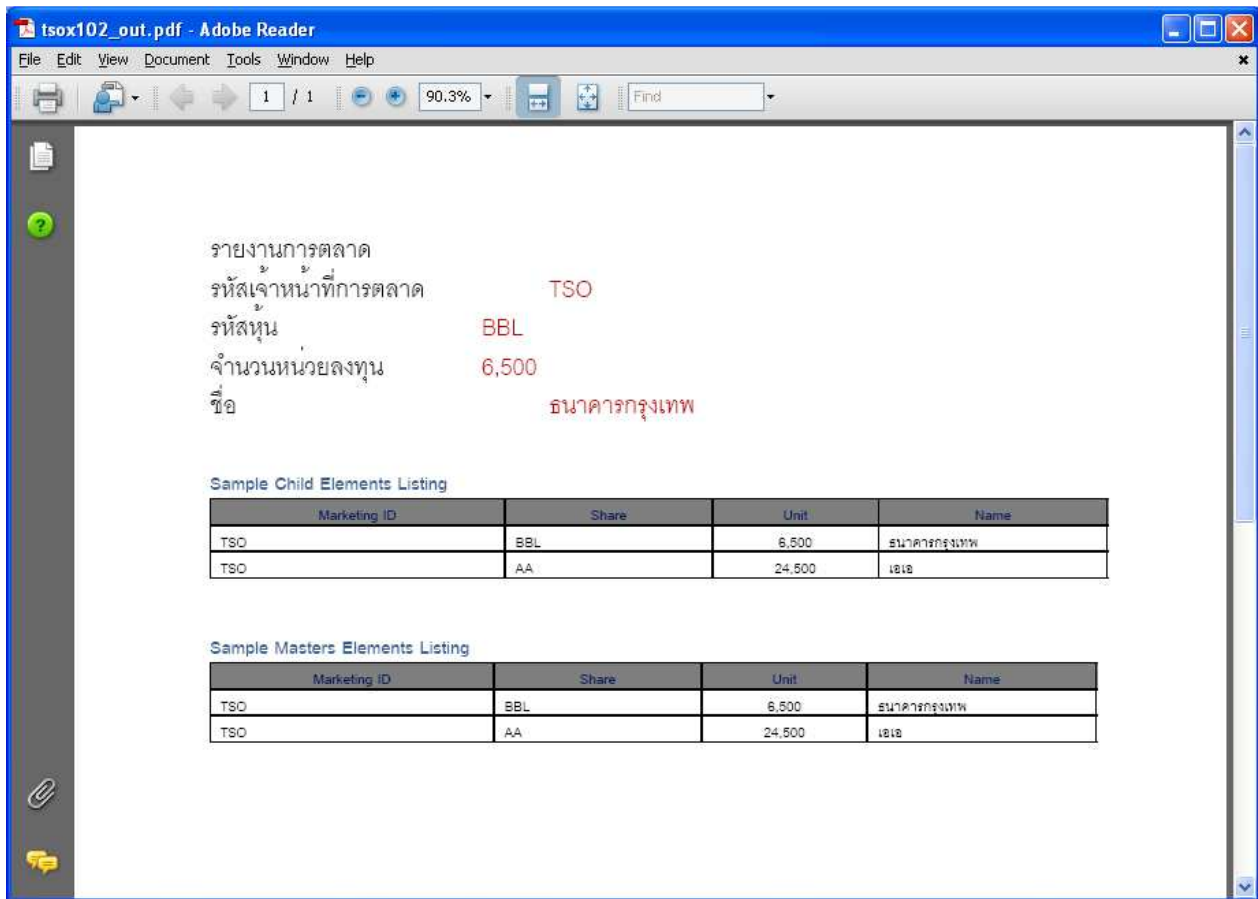
Examine

```
java com/fs/dev/report/JXReport -ms REFDB -layout d:\xdoc\tso\DOCX102.docx -doc  
d:\xdoc\tso\tsox102_out.docx -provider com.fs.bean.TSODData -action retrieve
```



Default renderer is docx then change to PDF

```
java com/fs/dev/report/JXReport -ms REFDB -layout d:\xdoc\tso\DOCX102.docx -doc
d:\xdoc\tso\tsox102_out.pdf -provider com.fs.bean.TSOData -action retrieve -t PDF
```



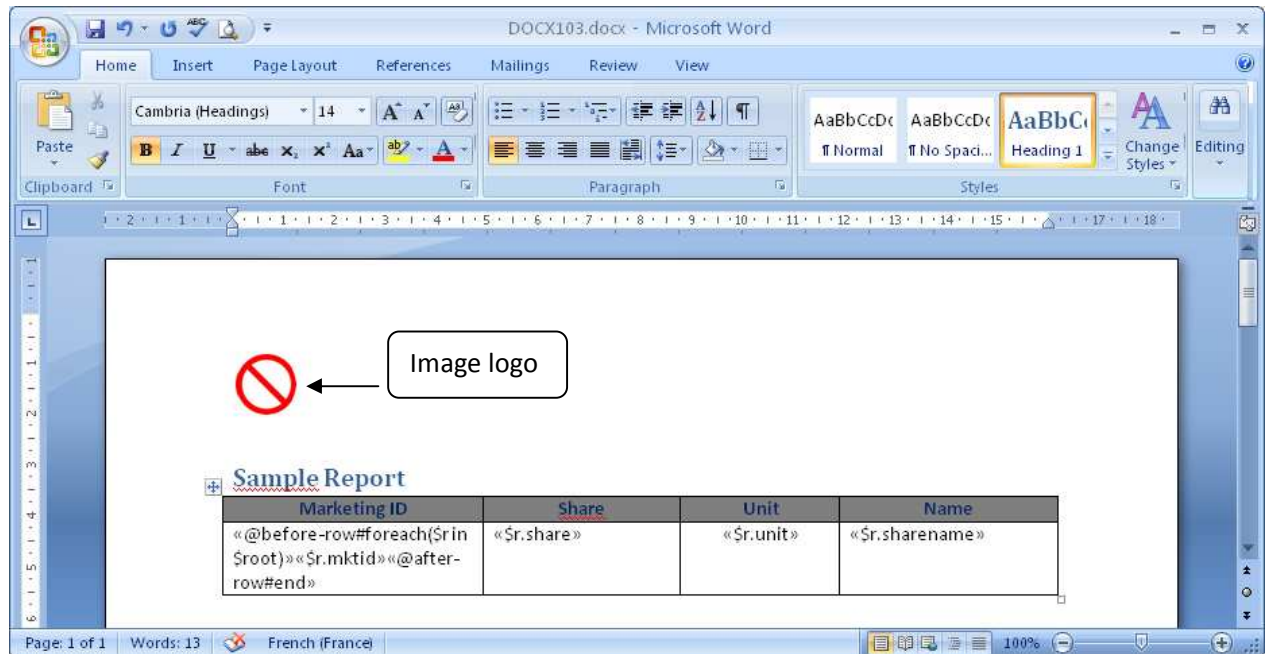
## How to Force Page Break

If you want to add page break into the script, try to add merge field with

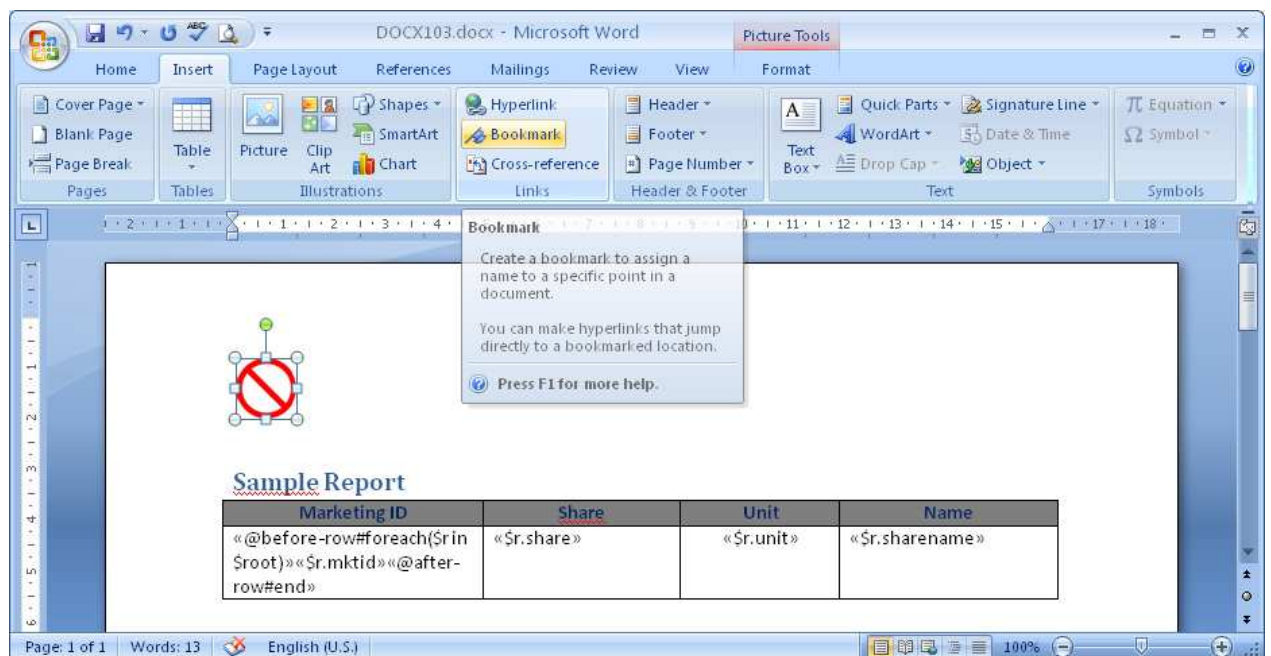
1. «\$page\_break\_before» or
2. «\$page\_break\_after»

## V. Create Simple Image Report

1. Create new document (DOCX103.docx) with layout below, by adding image logo

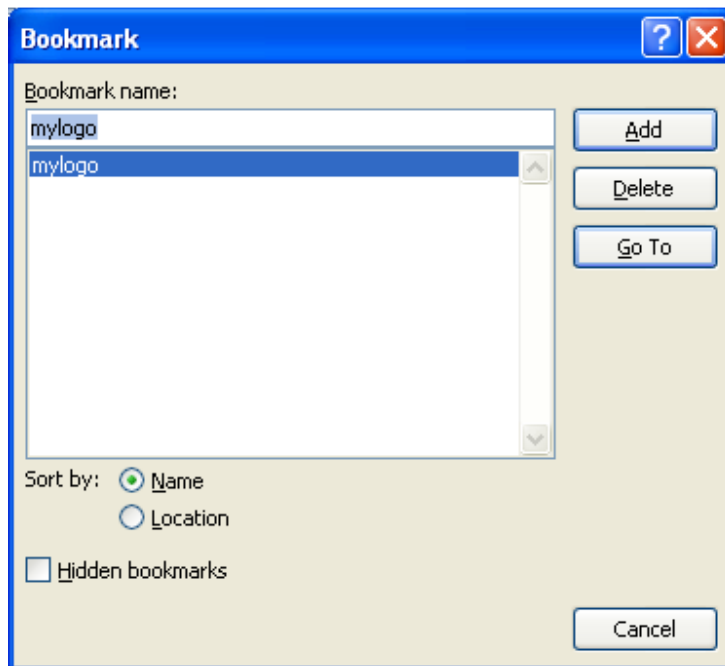


2. Add book mark under image logo  
2.1 select image logo, then select insert tab and click Bookmark



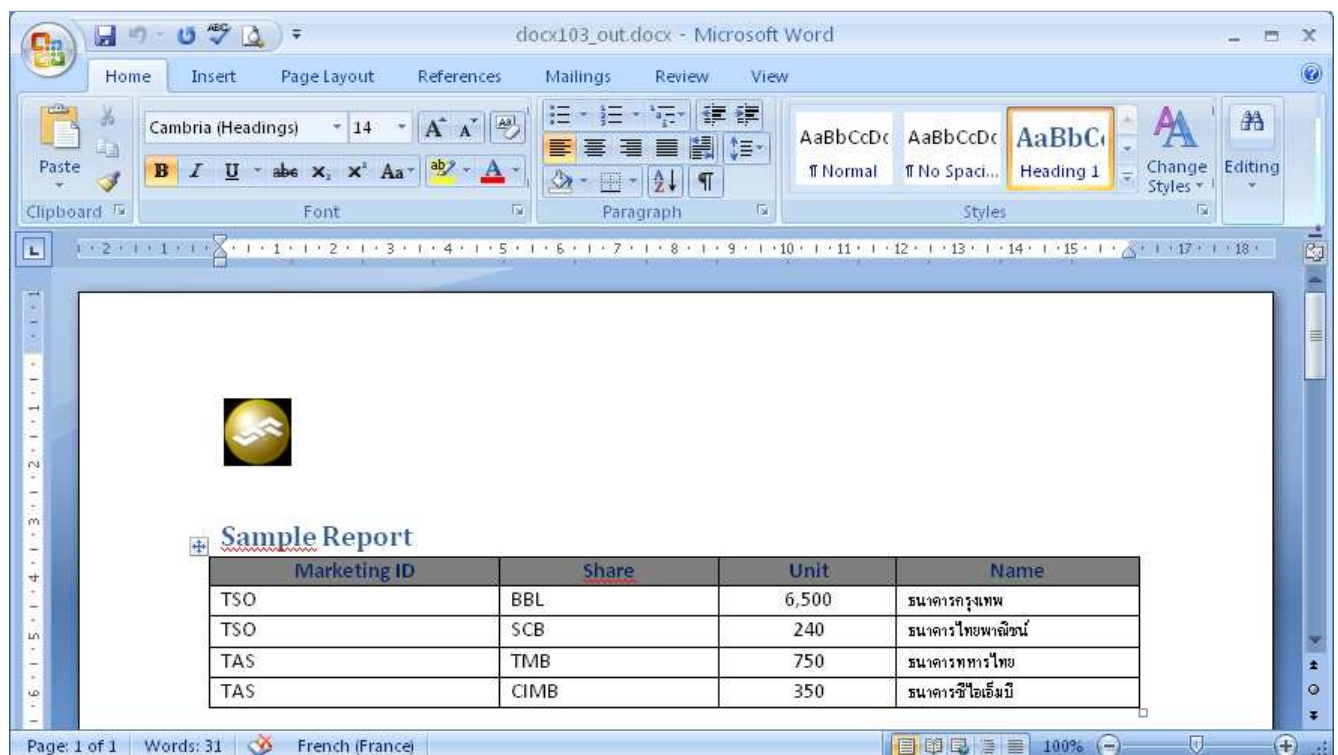


2.2 under book mark dialog try to add book mark with mylogo



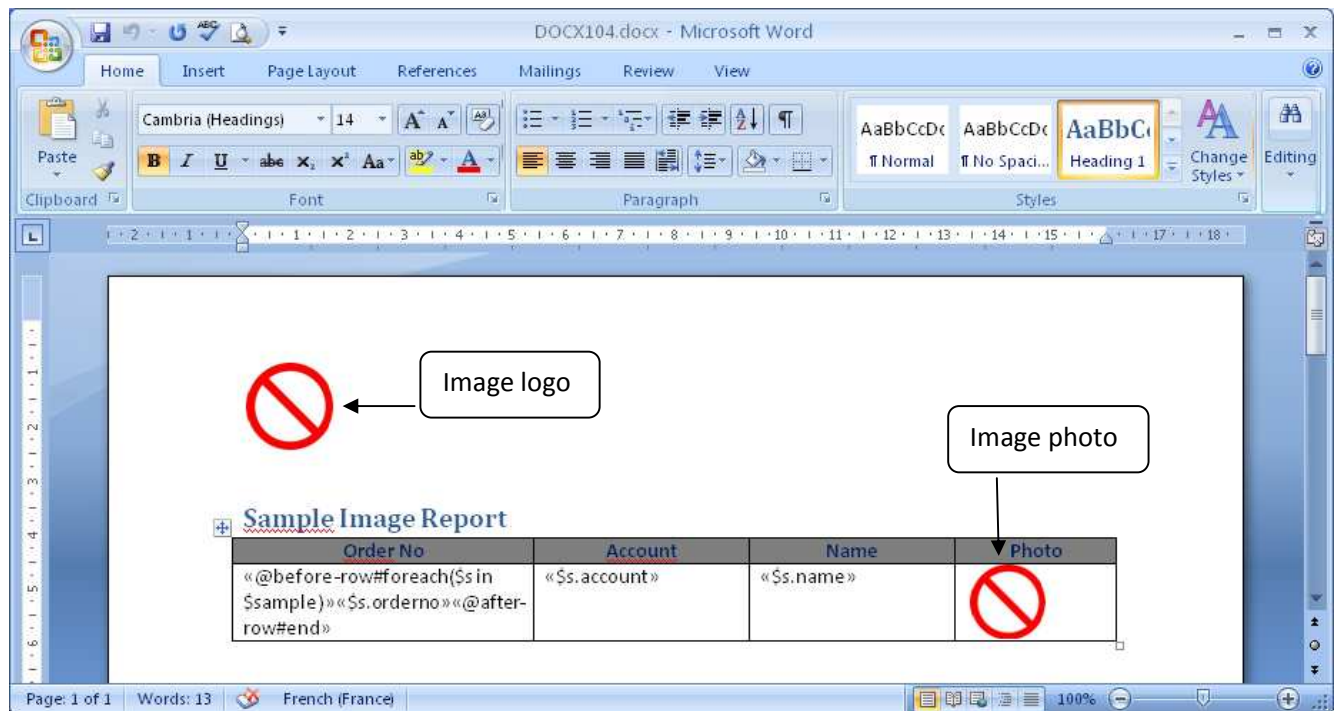
3. Save change and Examine

```
java com/fs/dev/report/JXReport -layout D:\xdoc\tso\DOCX103.docx -doc D:\xdoc\tso\docx103_out.docx  
-sql "select * from tso" -defcon true -image mylogo=D:\images\freewill.jpg
```

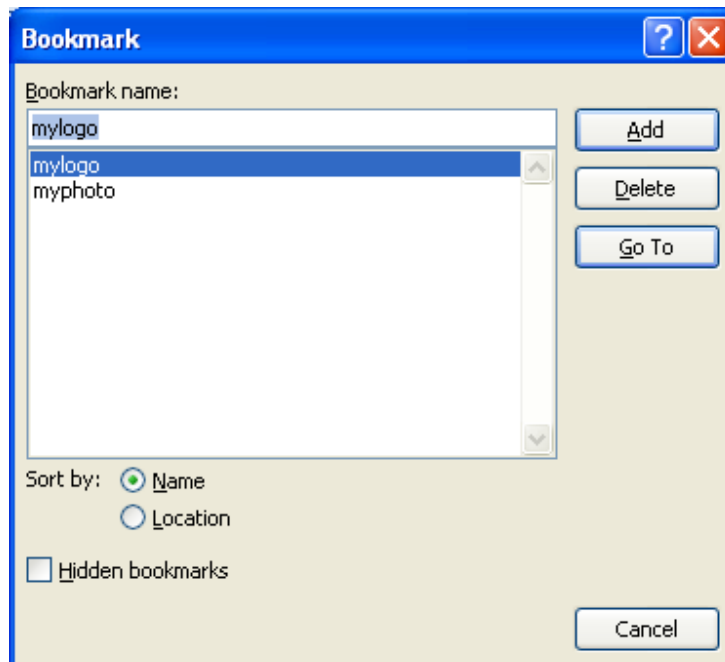


## VI. Create Image List Report

1. Create new document (DOCX104.docx) with layout below, by adding 2 image



2. Add book mark under image logo & image photo with mylogo & myphoto



3. Create java class DOCX104.java & inner class SampleBean

```
import com.fs.dev.report.*;

public class DOCX104 extends JXReport {

    private String imagepath = "D:\\TrainingXDoc\\images\\";

    public class SampleBean {

        private String orderno = null;

        private String account = null;

        private String name = null;

        private FileImage photo = null;

        public SampleBean(String orderno,String account,String name,String imagefile)
        {

            this.orderno = orderno;

            this.account = account;

            this.name = name;

            this.photo = new FileImage(new java.io.File(imagefile));

        }

        public String getAccount() { return account; }

        public String getName() { return name; }

        public String getOrderno() { return orderno; }

        private FileImage getPhoto() { return photo; }

    }

}
```

#### 4. Override method processDataSource

```
private java.util.List<SampleBean> composeDataSource(java.sql.Connection
connection,java.util.Map transientVar) throws Exception {

    java.util.Vector<SampleBean> list = new java.util.Vector<SampleBean>();

    list.add(new SampleBean("1","tso","tassun",imagepath+"tso.png"));

    list.add(new SampleBean("2","tdo","taddao",imagepath+"tdo.png"));

    list.add(new SampleBean("3","tpco","tadpicha",imagepath+"tpco.png"));

    list.add(new SampleBean("4","spo","Supara",imagepath+"spo.png"));

    return list;

}

@Override

public Object processDataSource(java.sql.Connection connection,java.util.Map transientVar)
throws Exception {

    java.util.List<SampleBean> list = composeDataSource(connection,transientVar);

    java.util.HashMap<String,java.util.List<SampleBean>> result = new
java.util.HashMap<String,java.util.List<SampleBean>>();

    result.put("sample", list);

    addImagesList("mylogo", imagepath+"freewill.jpg");

    addImagesFieldList("myphoto","s.Photo");

    return result;

}
```



## 5. Create main method

```
public static void main(String[] args) {  
    try {  
        if(args.length>0) {  
            DOCX104 report = new DOCX104();  
            report.setConnectionPool(false);  
            report.runApplication(args);  
        } else {  
            usage(DOCX104.class);  
        }  
    } catch(Throwable ex) {  
        ex.printStackTrace(System.err);  
        if(ex instanceof java.sql.SQLException)  
            System.err.println("ERROR CODE : "+((java.sql.SQLException)ex).getErrorCode());  
    }  
}
```

6. Try to compile & examine with java command line

```
java DOCX104 -layout d:\TrainingXDoc\DOCX104.docx -doc d:\TrainingXDoc\DOCX104_out.docx
```

