PL/OFFX XLSX
Getting Started
V4.7.0

PLPDF
Oracle Reporting Made Simple

Contents

Oracle Reporting Made Simple

1. Introduction

PL/OFFX XLS is a PL/SQL package collection what you can use to create Microsoft Excel compatible XLSX files and process existing XLSX documents. This guide will help you from the beginning to generate your first PL/OFFX XLSX sample file.

2. First steps

First, you need to install an Oracle Database on your system. The easiest way is to download the Oracle Express Edition. After you finished the installation, you can download the PL/OFFX software from the following link: PL/OFFX download. To install the latest version of PL/OFFX please use the install guide, called ploffx_install_v400.pdf. You also need a PL/SQL Developer environment, or you can use the SQL Plus application and a text editor.

3. Creating your first XLSX sheet

After you finished the installation start the PL/SQL Developer or a simple text editor and let's begin:

```
procedure AddCellExample is
```

The l_xlsx variable is a requirement. This variable will contain the xlsx file after the generation is finished.

```
l_xlsx blob;
```

The xlsx documents have some main parts. The first is the Worksheet. In most cases one document has several numbers of it and every Worksheet has a lot of rows and cells. You need to follow this concept in the PL/OFFX XLSX, too. In the first sample, we only create one sheet and one row. You will need the following code to store these references.

```
sheet_id pls_integer;
row_id pls_integer;
begin
```

When you want to create the XLSX files with the PL/OFFX XLSX application, you need to start with the Init procedure which will initialize the whole application environment. It has got one optional parameter, the p_date1904 which's default value is true. This parameter specifies the system date. If you set it false, it means the first day of the system was 01/01/1900. If the default values are right for you, you will only need this line:

```
ploffx_xlsx.init;
```

The PLOFFX/XLSX is now ready to create the XLSX document. Now, you can create the first main part of the document. With the AddWorksheet procedure you can easily create the Worksheet. You only need one thing to do this. The p_name parameter is a requirement and it needs to be unique. So you can't add two worksheets to your document with the same name.

```
sheet_id := ploffx_xlsx.addWorksheet('NewWorksheet2');
```

After the sheet was created, you can add rows to thw existing Worksheet. This is a simple task what you can do with the addRow procedure which has one parameter. The p_ws parameter means the Worksheet reference.

```
row_id := ploffx_xlsx.addRow(sheet_id);
```

Now, the document structure is ready for the first cell. There are three ways to add cells to an existing row. The way to add cells depends from the data type you want to put in them. In this example we use the addCell_String procedure. This is the minimal form of this procedure, but it has got another parameter called p_format_id. It specifies a reference to a previously defined format. The p_ws parameter identifies the Worksheet, and the p_row parameter sets the target row. The p_cell parameter can receive two kinds of value. You can add the target cell with a number, where the value means the column number. The second way is to write content in the column letter. The p_data contains the text value of the cell.

```
ploffx_xlsx.addCell_String(
p_ws => sheet_id,
p_row => row_id,
p_cell => 'A',
p_data => 'Init'
);
```
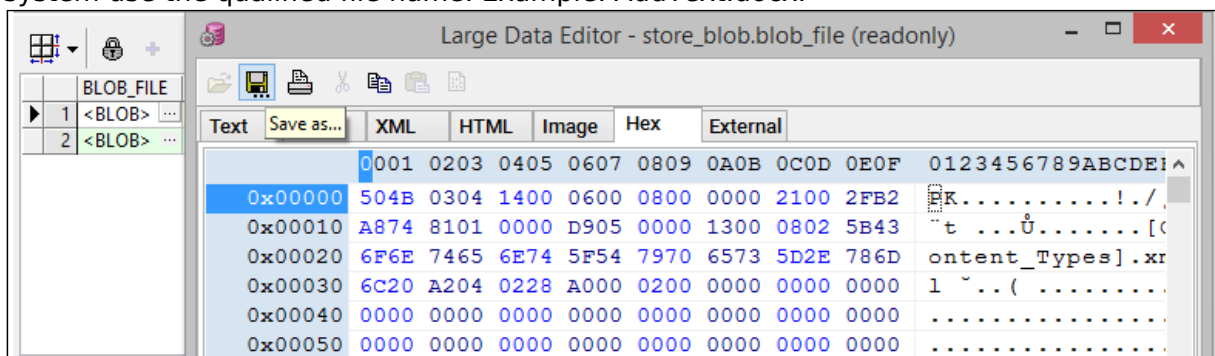
Now, the document structure is finished and we can generate to a simple XLSX file, that we will store in the l_xlsx variable. The getDoc procedure has one parameter, which specifies the the target BLOB type variable.

```
ploffx_xlsx.getDoc(p_blob_file => l_xlsx);
```

At this point, the file generation has successfully ended. The next step is optional. Store the result file, to a database table which is provided by, the PL/OFFX installkit, called STORE_BLOB.

```
insert into store_blob(blob_file,created_date,filename)
values (l_xls,sysdate,'AddCell.xlsx');
commit;
end;
```

If you use PL/SQL Developer, you can download the file. When you save it to a local file system use the qualified file name. Example: AddText.docx.

The generated example file in Microsoft Excel: