PL/PDF Reporter

User Guide

V4.9.0

Contents

1. Concept

PL/PDF Generator is a well-known way of creating PDF files using only standard Oracle PL/SQL language. PL/PDF Reporter extends PL/PDF Generator with a MS Word based template mechanism and separates the creation of data from the process of formatting. PL/PDF Reporter is simply the easiest and most flexible way to create professional reports from your Oracle database.

2. Components

PL/PDF Reporter separates the data logic from the layout design and processes the components into a runnable PL/PDF Generator package. Components of this solution are:

- Template: PL/PDF Reporter templates can be designed using the Microsoft Word. Templates created using this tool contain embedded placeholders with properties that determine how the Oracle data should be merged into the template using PL/SQL syntax to precisely match the server's engine.
- Data Model: PL/PDF Reporter's Data Model is a collection of PL/SQL objects which describe the required data sources (cursors) and variables in a PL/SQL package header.
- PL/OFFX DOCX Parser: PL/OFFX Suit can load a DOCX based template into an internal structure and store all the required information in a standard PL/SQL structure.
- PL/PDF Reporter Processor: The Processor interprets the template structure and the data model and build a PL/PDF conform PL/SQL code (package), which contains all required parts to create the PDF output.
- PL/PDF Generator: Generator can run already built PL/SQL packages as a standard PL/PDF Generator program and produces PDF outputs as a BLOB variable.

3. Benefits

PL/PDF Reporter provides a single PL/SQL for authoring, managing, and delivering reports, and all types of highly formatted documents:

- Fast Development: End users can easily create report layouts using Microsoft Word. MS Word is a well-known desktop tool, there is no need of learning a new program.
- Separated Data Model and Layout: Simplified Report Maintenance, easy modification.
- Separated Placeholders: PL/PDF Reporter uses MS Word Content Control items and users can switch visibility of the control items off/on. When control items are switched off then the report's layout contains no data, you can only see the previously built template.
- Everything is PL/SQL: All parts of the PL/PDF Reporter are implemented in native Oracle PL/SQL language, so there is no need to learn a 3rd-party programming language.

4. MS Word Content Control usage

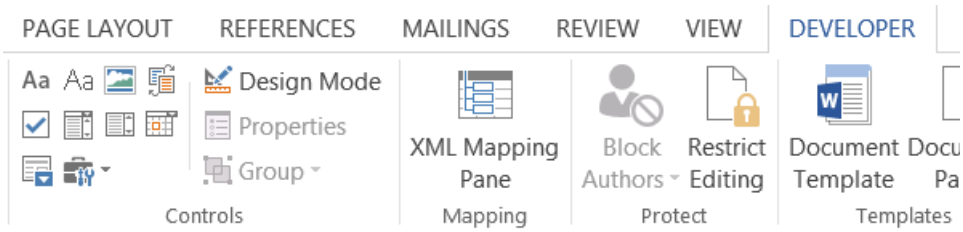
PL/PDF Reporter uses the MS Word Content Control feature for adding dynamic Oracle database related content to PDF reports. Content controls are individual controls that you can add and customize in the documents. These are available since Microsoft Word 2007 and the PL/PDF Reporter stores templates in DOCX files. You can find the content controls for your document in the **Developer** tab in MS Word.

View the Developer tab (MS Word 2007)

1. Click the Microsoft Office Button image, and then click Word Options.

2. Click Popular.
3. Select the Show Developer tab in the Ribbon check box, and then click OK.

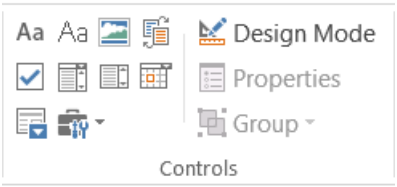
Developer ribbon in MS Word 2013



Switching the Design mode on/off: This is a big feature of the Content Control because the switched off state does not contain the Control items and the layout is 100% equal with original content. When you switch the Design Mode on you can easily select and modify existing Controls.

Content Controls in PL/PDF Reporter: PL/PDF Reporter uses 2 types of Content Controls:

- Rich Text: This is our commonly used Control, we use it as a placeholder of text, PL/SQL code or repeating rows in a table.
- Picture: This is a placeholder of the dynamic images, but if you want use a static image in your template then please use the standard image inserting feature in MS Word.



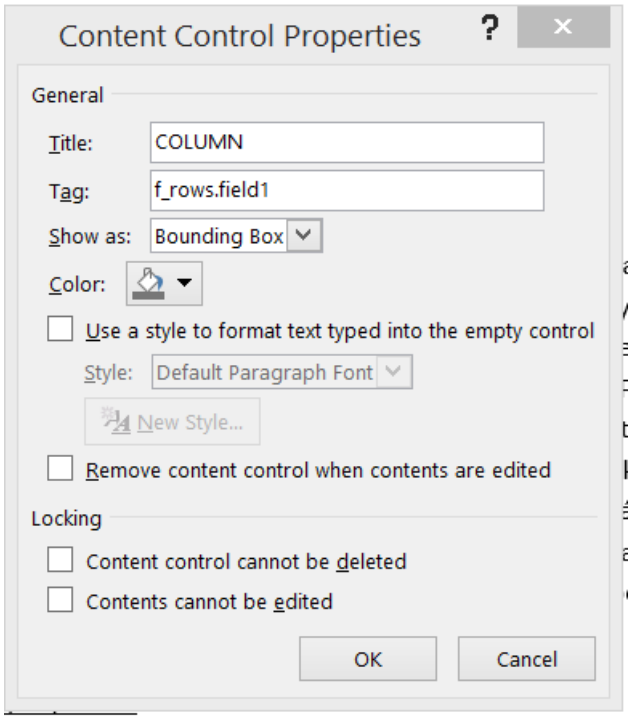
Usage of Content Controls: Content Controls is a placeholder of a dynamic text, PL/SQL code or custom Reporter features. Follow these steps:

1. Switch to Design Mode
2. Select the affected part of the document content, possible options are:
   a. Small part of the text (MS Word name is “inline-level”)
   b. Paragraph
   c. Table Cell
   d. Table Row (one row only)

   Selecting the desired parts of the document is not an easy work in MS Word because there is no way for checking the type of selected parts, please double check the selection.
3. Select the proper Content Control from the Developer ribbon (MS Word add to doc), we use:
   a. Rich Text
   b. Picture (only for image content)
4. Click on the Properties button on the Developer ribbon and set the Reporter related info
   a. Title: uppercase name of the PL/PDF Reporter controller, example GROUP, COLUMN, etc.

b. Tag: parameters of the PL/PDF Reporter Control, how to use the Control, language of the parameters' description is PL/SQL. There is no need of XML related knowledge. Example, if Control is the COLUMN (column of a cursor) the parameter value is the <cursor_name>.<column_name>

5. Click OK



Viewing/Modifying a PL/PDF Reporter Control is the same:

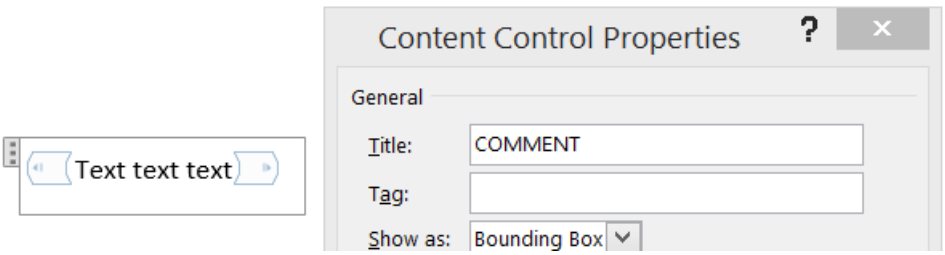
1. Switch to Design Mode
2. Select the Content Control
3. Click on Properties button on the Developer ribbon
4. View/Modify the Title or the Tag property on the Content Control Properties window
5. Click OK or Cancel

5. PL/PDF Reporter Controls

We will explain the title and tag settings of the Content Controls and the resulting PL/SQL codes. Types of the PL/PDF Reporter Controls:

COMMENT

Comment is for inserting PL/SQL comment into the generated PL/PDF code. The parameter is not used, the selected text (paragraph) becomes a multiline PL/SQL comment (/* */).



The result in MS Word:

COMMENT Text text text COMMENT

The result in the PL/PDF Generator Code:

/*Text text text*/

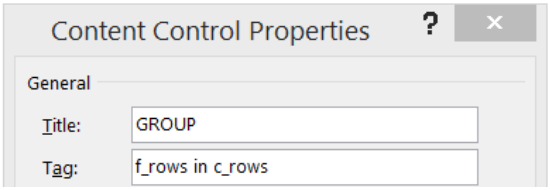
GROUP

Group surround of a table row is equivalent with the PL/SQL FOR-LOOP repeating structure. The parameter describes the cursor and the cursor variable, just the same as the PL/SQL code part between FOR and LOOP keywords. The repeating loop prints (is fired) once for each record of the cursor/group. Cursor is implemented as a PL/PDF Reporter Data Model item in the report package header.
Start with a table:

Header1	Header2	Header3
field1	field2	field2

Select the second row and add the GROUP Control:

GROUP er1	Header2	Header3
GROUP field1	field2	field2 GROUP



-
The result in MS Word

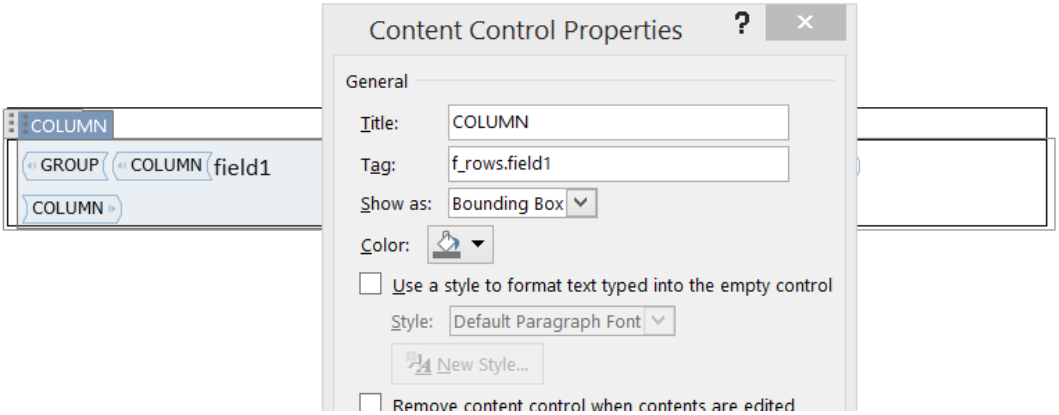
Header1	Header2	Header3
GROUP field1	field2	field2 GROUP

The generated code in PL/SQL:

```
FOR f_rows in c_rows LOOP
…
END LOOP;
```

COLUMN

Column is the reference to a field/column of the GROUP, this is the implementation of the data print out. Reporter supports the text Columns or PL/SQL converts the Column value to string. Parameter is the qualified name of the field of the cursor variable: <cursor_name>.<field_name>. Example based on previous GROUP:
Select the first column text in the second row and add Control:



The result in the MS Word:

Header1	Header2	Header3
GROUP COLUMN field1 COLUMN	field2	field2 GROUP

The result of the PL/SQL code is

:= f_rows.field1;

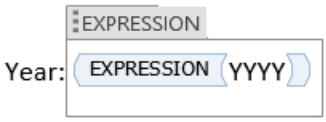
EXPRESSION

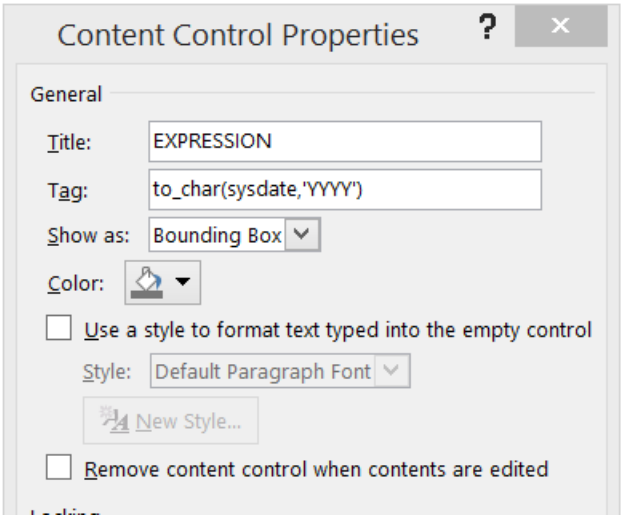
PL/PDF Reporter Expression Control is similar to Column Control because the result of the Control is a text or an implicit converted text. Expression Control is a PL/SQL Expression and the surrounded text is replaced with the result string. The user can use items from the related Data Model or accessed database items. Example: print out the current year (YYYY) on to the report:

1. Add placeholder text to the doc

Year: YYYY

2. Add Control





3. The result in DOCX

Year: EXPRESSION YYYY

4. The result in DOCX Design Mode Off

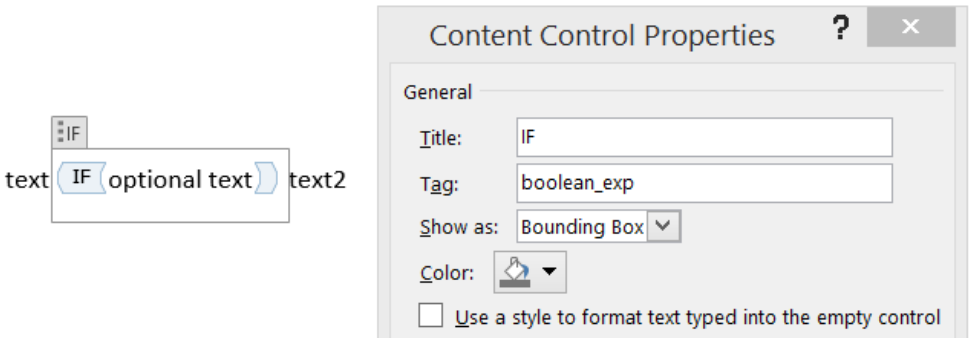
Year: YYYY

5. The result in PDF

Year: 2015

IF

PL/PDF Reporter IF Control is a tool for implementing optional content. The name of the Control is IF and the parameter is a Boolean expression: if the expression is true then the surrounded content is visible in the report. Example:

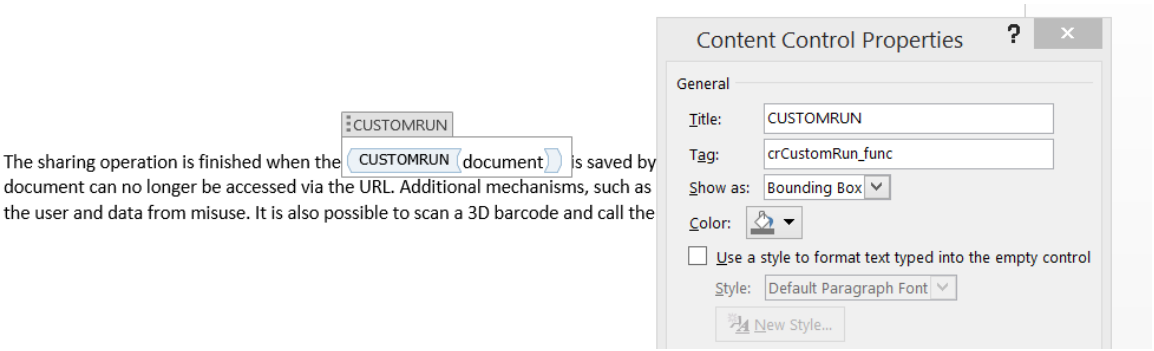


The PL/SQL result is

```
IF boolean_exp THEN
…
END IF;
```

CUSTOMRUN

Custom Control implements creating the text part (run) from PL/SQL directly, and PL/PDF Reporter will replace the associated content with this new content. The parameter is an executable PL/SQL function and the return value type is plpdf3.t_Run. Example:

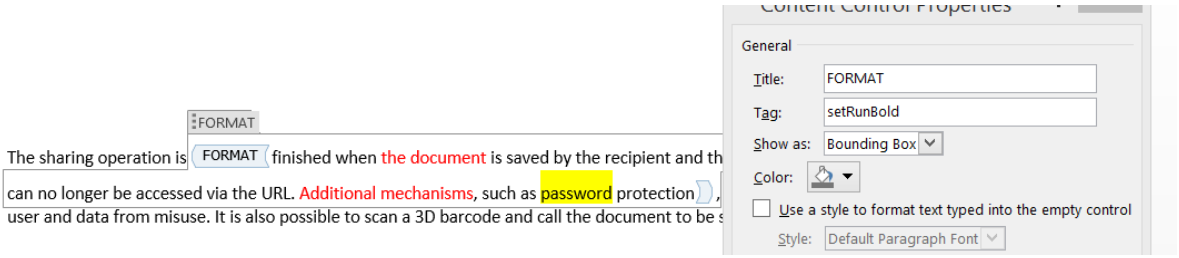


The PL/SQL result is

```
:= crCustomRun_func;
```

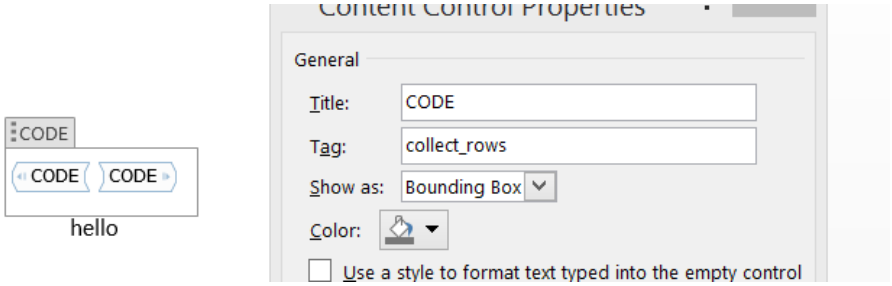
FORMAT

Format Control is a possible way of formatting a text (run) from PL/SQL code. The parameter of the Control is the name of the PL/SQL procedure, the required parameter list of this procedure is (p_run IN OUT plpdf3.t_Run) and the implemented code can modify all properties of this text. Example:



CODE

Code Control is a simple way of adding specific PL/SQL programming codes to Report runtime, its common usage is a preprocessor for creating content of the Report or part of the Report. The parameter of the Code control is the name of the called PL/SQL procedure, example:

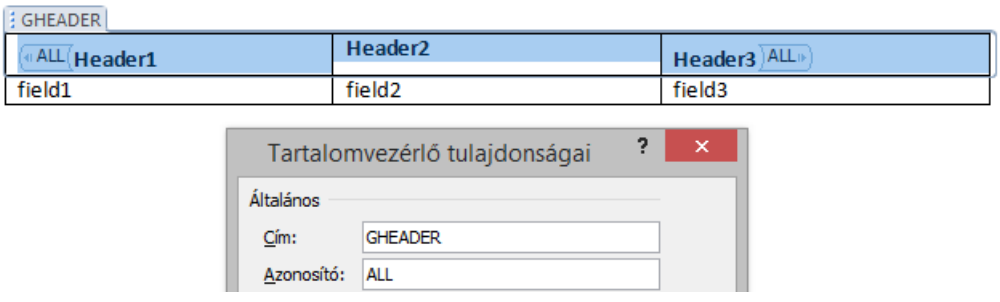


GHEADER

Gheader surrounds a table row, and it is a simple way of adding a header row to the table. The parameter describes the header type. It can be ALL or FIRST.
Start with a table:

Header1	Header2	Header3
field1	field2	field2

Select the first row and add the GHEADER Control:



The PL/SQL result is:

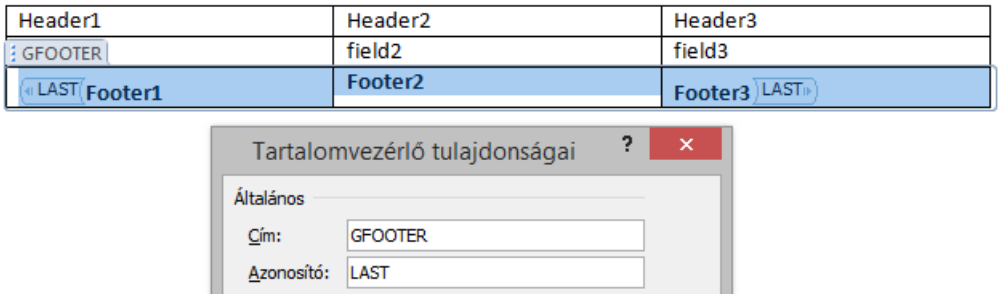
:= plpdf3.addHeaderRow(…);

GFOOTER

Gfooter surrounds a table row, and it is a simple way of adding a footer row to the table.
The parameter describes the footer type. It can be ALL or LAST.
Start with a table:

Header1	Header2	Header3
field1	field2	field3
Footer1	**Footer2**	**Footer3**

Select the last row and add the GFOOTER Control:



The PL/SQL result is:

:= plpdf3.addFooterRow(…);

MASTER-DETAIL MODE TABLE

Thas 'MASTER' and the 'DETAIL' are postfixes of some control. These controls are GHEADER, GFOOTER and GROUP. Using these postfixed controls are a simple way of adding a header/footer covered master-detail structured table.
Start with a table:

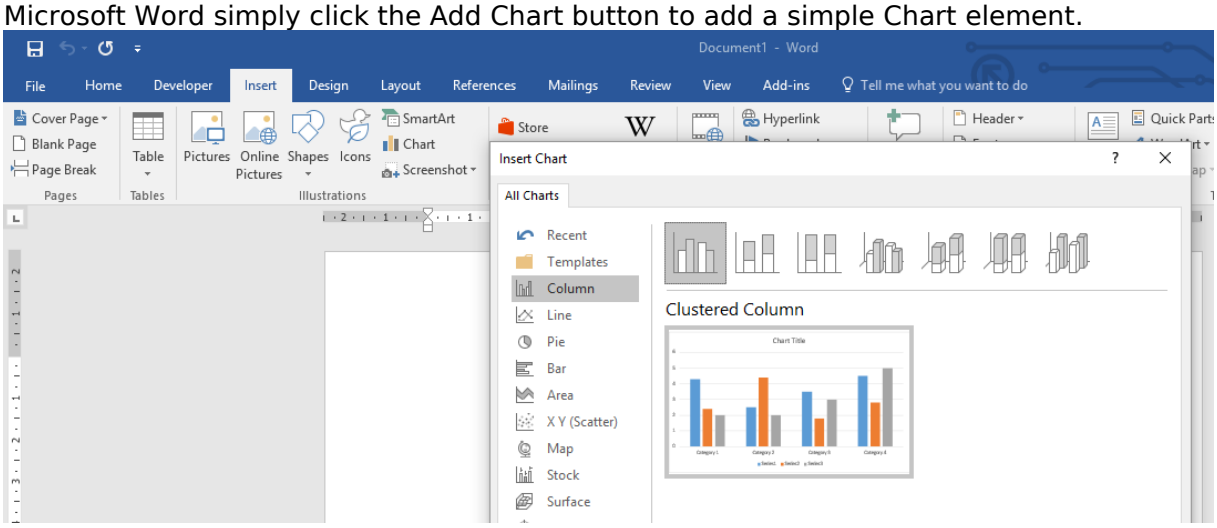
Header Title		
Header1	Header2	Header3
field1	field2	field3
Detail header title		
Header4	Header5	Header6
field4	field5	field6
field7		
Detail footer1	Detail footer2	Detail footer3
Master footer1	Master footer2	Master footer3
Master footer4		

Use the GHEADER, GFOOTER and GROUP controls in the following way:

GHEADER_MASTER Header Title GHEADER_MASTER		
GHEADER_MASTER Header1	Header2	Header3 GHEADER_MASTER
GROUP_MASTER COLUMN field 1 COLUMN	COLUMN field2 COLUMN	COLUMN field3 COLUMN GROUP_MASTER
GHEADER_DETAIL Detail header title GHEADER_DETAIL		
GHEADER_DETAIL Header4	Header5	Header6 GHEADER_DETAIL
GROUP_DETAIL COLUMN field 4 COLUMN	COLUMN field5 COLUMN	COLUMN field6 COLUMN GROUP_DETAIL
GROUP_DETAIL COLUMN field7 COLUMN GROUP_DETAIL		
GFOOTER_DETAIL Detail footer1	Detail footer2	Detail footer3 GFOOTER_DETAIL
GFOOTER_MASTER Master footer1	Master footer2	Master footer3 GFOOTER_MASTER
GFOOTER_MASTER Master footer4 GFOOTER_MASTER		

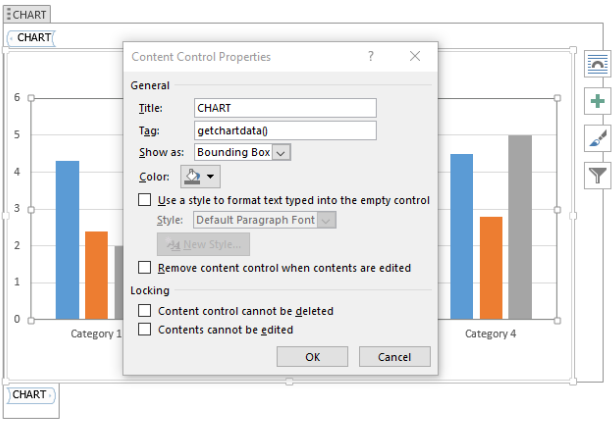
CHART

The chart element is not an independent content control like previous elements. In Microsoft Word simply click the Add Chart button to add a simple Chart element.



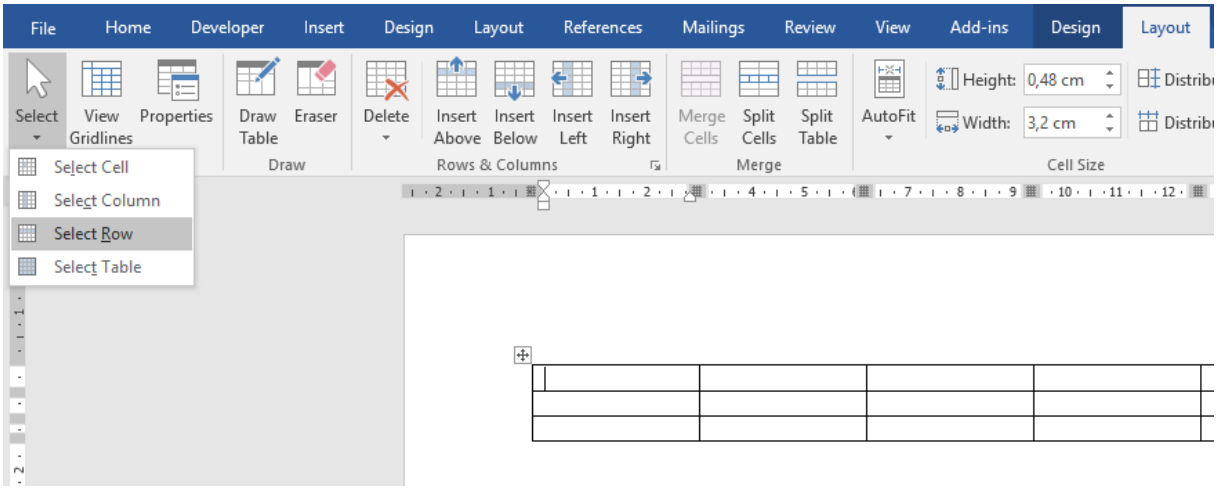
There are three types of Chart which are currently supported in PL/PDF Reporter.
- Bar chart
- Pie chart
- Line chart

When the chart added to the document structure there are only one thing left. Set up the chart properties. Customize the style and update the data source of the chart. After that, the chart is ready for the PL/PDF Reporter template processing. The PL/PDF Reporter CHART control parameter is the dynamic data generator procedure name.



6. How to select elements is MS Word

- Text or paragraph: You can select it easy with double click on the text.
- Table cell: Click inside the table cell. After that go to the LAYOUT option in the end of the menubar. The first option is select. Choose select cell.
- Table row: Click inside one cell of the table row. After that go to the LAYOUT option in the end of the menubar. The first option is select. Choose select row.



7. Creating the Data Model

PL/PDF Reporting Data Model is a simple solution for implementing the report related data sources and variables. The user creates the report data model manually as the report package header and PL/PDF Reporter Processor creates the report runtime as the package body. Typically the Data Model contains PL/SQL cursor definitions and GROUP control references in the template. Example cursor for our GROUP example:

```
cursor c_rows is
select field1, field2, field3
from custom_table;
```

The user can add package level (global) variables to the report header to store and interchange data between the caller and the report (as a parameter) or handling internal data while the report is running.
The restriction of the report package's header:

- the name of the package header is the same as the name of the generated report package's body (see processing).
- Report procedure: "`procedure genPDF;`"
- Report result variable: "`v_pdf blob;`"

Here is the minimal report package header:

```
create or replace package x_vertalign is
  -- Required procedure
  procedure genPDF;

  -- Required variable
  v_pdf blob;
end;
```

And here is another example with cursor and variable:

```
create or replace package x_report is

  cursor c_sample is
      select rownum, owner, object_name
      from all_objects
      where rownum <= 100;

  -- Expression example
  v_customer_name varchar2(30) := 'Scott Tiger';

  -- Required procedure
  procedure genPDF;

  -- Required variable
  v_pdf blob;
end;
```

8. Processing the PL/PDF DOCX Template

The processing of the DOCX template contains of parsing the DOCX structure and creating the PL/PDF conform PL/SQL package body for the Data Model header.
Requirements of the running:

a. Granted execute on PL/PDF packages (PL/PDF Generator, PL/OFFX Suit, PL/PDF Reporter)
b. Granted execute on report related procedures and access of the related tables.
c. Created DOCX based template
d. Created Data Model package header
e. Created table for static images
f. Font mapping

Static image table stores the extracted images from the DOCX template. Structure of table:

```
create table <reportx_images>
(
```

```
    id          varchar2(255),
    image_file  blob,
    filename    varchar2(255)
);
```

Font mapping is an important step in creating a similar output in PDF. The problem is that the fonts in the DOCX are not accessible resources in the database. The solution is to upload the DOCX template font into the database, use PDF Standard fonts or use a different font instead of the original fonts. These three ways are implemented via the font mapping feature. The font mapping is a table for pairing the original DOCX font and the new PDF font. If you use a custom TTF font in PL/PDF Generator then producing the TTF font info is a required step in the PL/PDF Generator, see PL/PDF Generator related documentation.

Supported DOCX features in the template:

Text (Run) Processing

- Text color
- Text size
- Text font name
- Underlined, bold and italic text
- Hyperlink
- Text padding
- Text border color, width, style
- Text background color
- Fit text
- Text width
- Text vertical alignment
- Text vertical position

Paragraph processing

- Paragraph indentation
- Paragraph alignment
- Paragraph spacing
- Page break
- Paragraph vertical alignment
- Paragraph border color, width, style
- Paragraph margin
- Paragraph padding
- Paragraph background color

Image processing

- Add inline image to the document
- Image height and width
- Vertical position
- Image border color, width, height

Table processing

- Table width
- Table grid

- Table alignment

Cell processing
- Adding paragraphs to a cell
- No wrap property
- Vertical Align
- Cell border color, width, style
- Cell padding
- Cell background

PL/PDF Report Processor supports several DOCX features for formatting your content but PL/PDF Reporter is not a DOCX converter or printer, some parts of the result PDF is not equal with the original DOCX.

Main unsupported features:
a. Table in table
b. List items

The processing procedure is part of the plpdf_reporter package:

```
function genCode(
  p_packagename varchar2,
  p_file in out blob,
  p_encoding varchar2 default null,
  p_image_table varchar2 default null,
  p_package_header boolean default false,
  p_ascii_mode boolean default false,
  p_force_vars boolean default false,
  p_append_slash boolean default true,
  p_header_height number default null,
  p_footer_height number default null
  ) return clob;
```

Meaning of the parameters:
a. p_packagename: name of the generated PL/SQL package, this is the same as the Data Model package header's name
b. p_file: this is the DOCX template as BLOB
c. p_encoding: PL/PDF Generator encoding setting, see the plpdf.SetEncoding procedure
d. p_image_table: table name for DOCX static images
e. p_package_header: create package header
f. p_ascii_mode: generate full ASCII source code
g. p_force_vars: ignores variable duplicates
h. p_append_slash: appends slash after package body
i. p_header_height:overwrites page header height
j. p_footer_height: overwrites page footer height

The plpdf_reporter.genCode creates a PL/PDF Generator conform PL/SQL package and is set as return value. You can compile it into the database with "execute immediate" or insert it into a PL/SQL Editor and compile it as a normal package body. If the generated PL/SQL package body has an error, compile the raises related error and you can check the source of the body as a standard PL/SQL package.

9. Move the report to PROD

The generated report is a PL/PDF Generator conform PL/SQL package, moving the report to production is easy:

1. Install the PL/PDF Generator in the production database
2. Move the image static table with the content
3. Install the report PL/SQL package

There is no need of running other components.

10. Questions and Answers

a. How to run the generated report?
Here is a simple example, x_report is the name of the report package:

```
  begin
 -- run the report
 x_report.genpdf;

 --store the result
 insert into store_blob
   (blob_file,created_date,filename)
 values
   (x_report.v_pdf,sysdate,'report1.pdf');
 commit;
   end;
```

b. How to implement report parameters?
The best way is using variables in the Data Model package header and the caller can set and get this variable runtime, example:

```
--data model
create or replace package x_report is
  -- Expression example
  p_id number;
  p_par2 varchar2;

  -- Required procedure
  procedure genPDF;

  -- Required variable
  v_pdf blob;
end;

--run
begin
  -- set parameters
  x_report.p_id := 10;
  x_report.p_par2 := 'tiger';
  -- run the report
  x_report.genpdf;

  --store the result
  insert into store_blob
    (blob_file,created_date,filename)
  values
```

```
  (x_report.v_pdf,sysdate,'report1.pdf');
commit;
    end;
```

c. Is Font mapping a required step?
   No, font mapping uses two levels for handling a missing font. The default is the "Arial", if font mapping is missing then the Processor uses Arial for all text.