PL/PDF Reporter Mail Merge

User Guide

V5.0

Contents

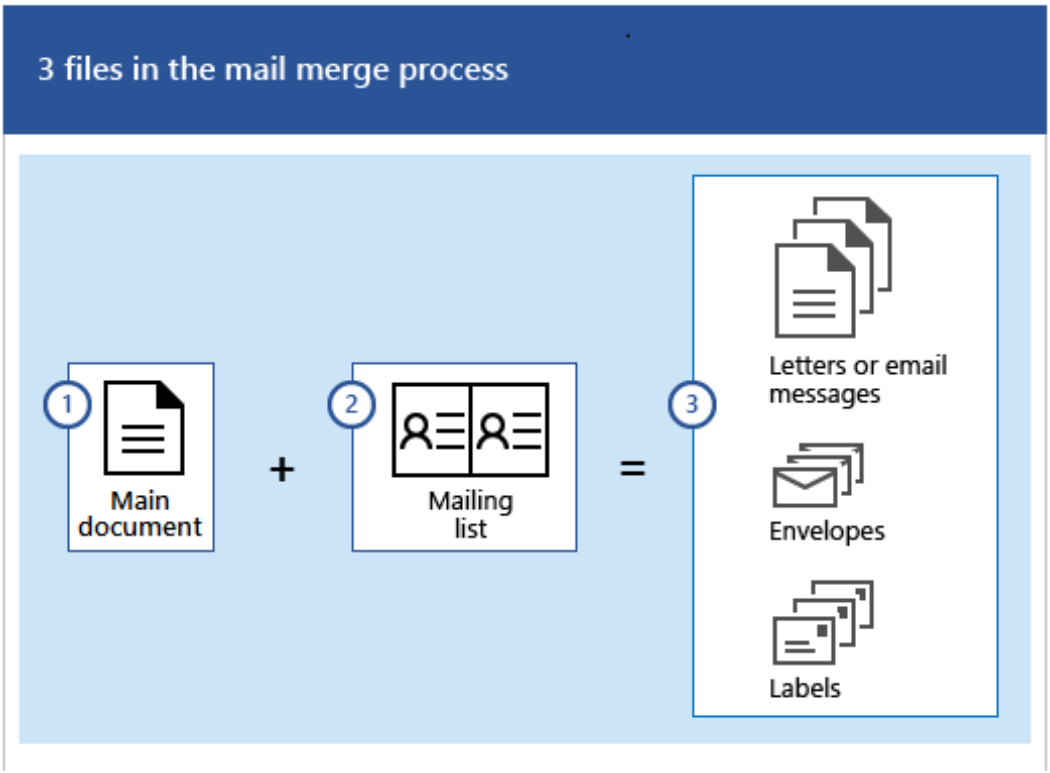
1. Concept

Main goals are the implementing the MS Word Mail Merge template (Merge Fields) and Oracle SQL Select (Data Source) merging function in PL/PDF. The result is set of the PDF or DOCX documents and supporting the creation of Data Source from SQL Select statement as XLSX document.

MS Office Mail Merge is a well know feature to create and to print form letters by using data from a Microsoft Excel worksheet. Here is a How to Article: https://www.webucator.com/how-to/how-use-mail-merge-microsoft-word.cfm

When you use the Word Mail Merge feature, Word merges a main document with a recipient list to generate a set of output documents:

- The main document contains the basic text that is the same in all of the output documents. It may contain a letterhead, text, and instructions in merge fields for inserting text (such as recipient names and addresses) that vary from one output document to another.
- The recipient list is a database that contains the data that is to be merged into the output documents. For example, the recipient list is a Microsoft Access database file or an Excel worksheet.
- This database is typically a list of names, addresses, phone numbers, and other categories of personal information.
- The output documents are the result of the mail merge. The text in an output document can be the same in all output documents, but you can apply formatting to specific documents.



PL/PDF supports the standard MERGEFIELD field only, see https://support.office.com/en-us/article/field-codes-mergefield-field-7a6d24a1-68a6-4b05-8359-1dc087daf4e6

The basic syntax is "MERGEFIELD FieldName". The PL/PDF generates related parameter in the parameter record as p_params. FieldName

2. Main steps in the PL/PDF

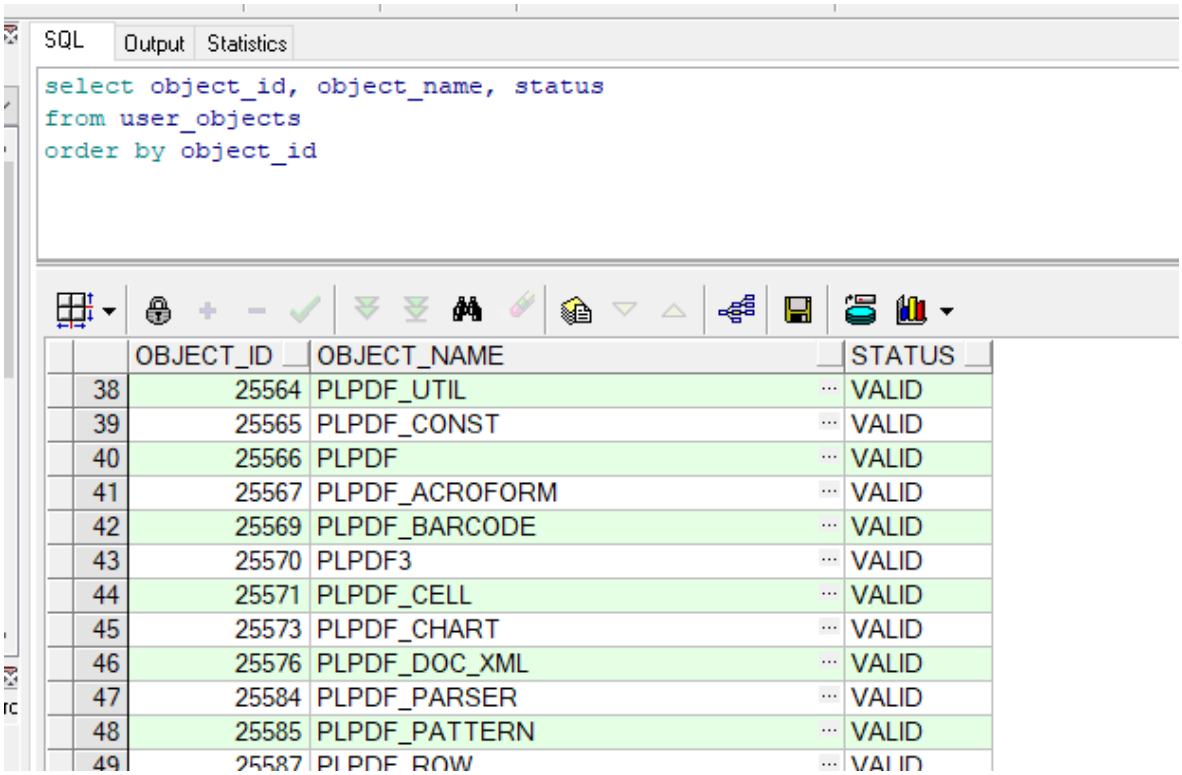
1. (SQL*Plus/Developer) Create SQL Select for Data Source creation
2. (PL/PDF) Export Data Source as XLSX
3. (MS Word) Start new template or open existing DOCX for template
4. (MS Word) Select Recipients XLSX file
5. (MS Word) Editing template, adding static text and fields (MERGEFIELD)
6. (PL/PDF) Upload DOCX file and Generate PL/SQL Code from DOCX template and the basic info
7. (SQL*Plus/Apex) Execute

We would like to help the implementation with a real example, this mail merge is based on the USER_OBJECTS table OBJECT_ID, OBJECT_NAME, STATUS fields only. This is very simple example for understanding basic steps only.

3. Create SQL Select for Data Source

The MS Word Mail Merge uses Recipient List for providing recipients and merge fields, this process implements Oracle SQL Select fields as **merge fields only**. Please use alias for expression field because field names become merge field names.
Our example Select:

```
select object_id, object_name, status
from user_objects
order by object_id
```



	OBJECT_ID	OBJECT_NAME	STATUS
38	25564	PLPDF_UTIL	VALID
39	25565	PLPDF_CONST	VALID
40	25566	PLPDF	VALID
41	25567	PLPDF_ACROFORM	VALID
42	25569	PLPDF_BARCODE	VALID
43	25570	PLPDF3	VALID
44	25571	PLPDF_CELL	VALID
45	25573	PLPDF_CHART	VALID
46	25576	PLPDF_DOC_XML	VALID
47	25584	PLPDF_PARSER	VALID
48	25585	PLPDF_PATTERN	VALID
49	25587	PLPDF_ROW	VALID

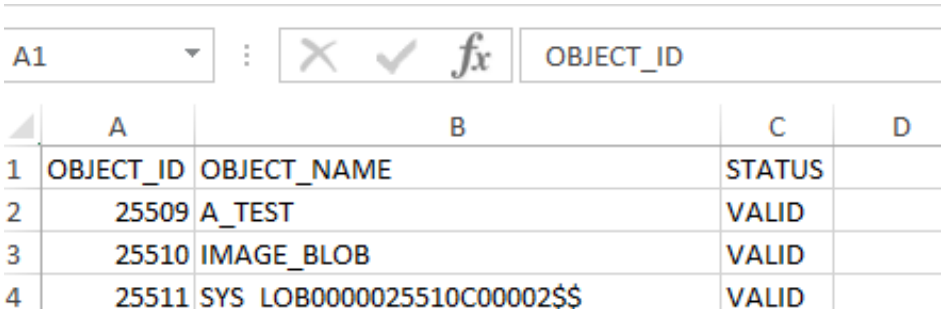
4. Export Data Source as XLSX

Before you proceed with the Mail Merge Wizard in MS Word, make sure that your Excel worksheet is well structured for this purpose. Note the following requirements for the data table:

- The first row should contain field names for each column -- for example, Title, Salutation, First Name, Middle Name, Last Name, Address1, and Address2.
- Each field name should be unique.
- Each row should provide information about a particular item. For example, in a mailing list, each row might include information about a particular recipient.
- The table should contain no blank rows.

The PL/PDF creates the proper XLSX file with own standard PLOFFX_XLSX package, you can find the related script in the PL/PDF Instalkit.

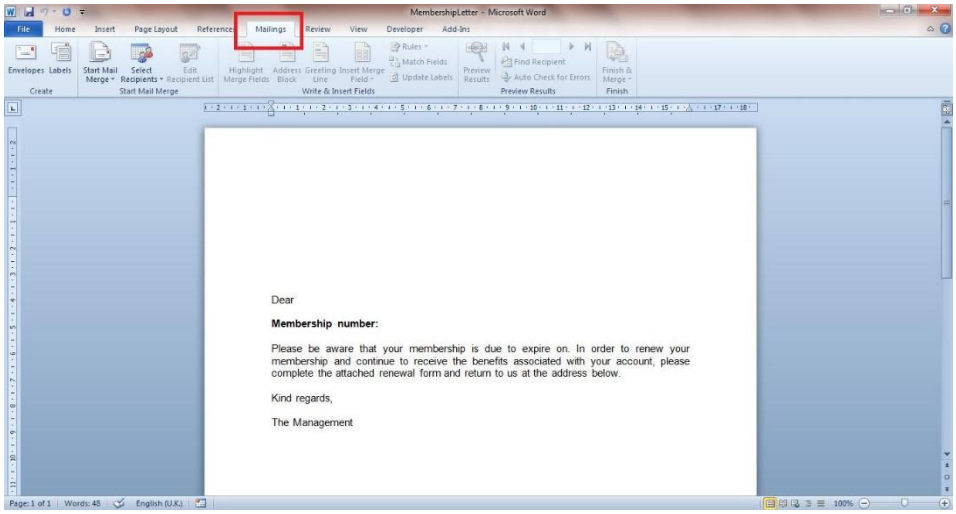
After you create your Excel data file, save it to the file system.



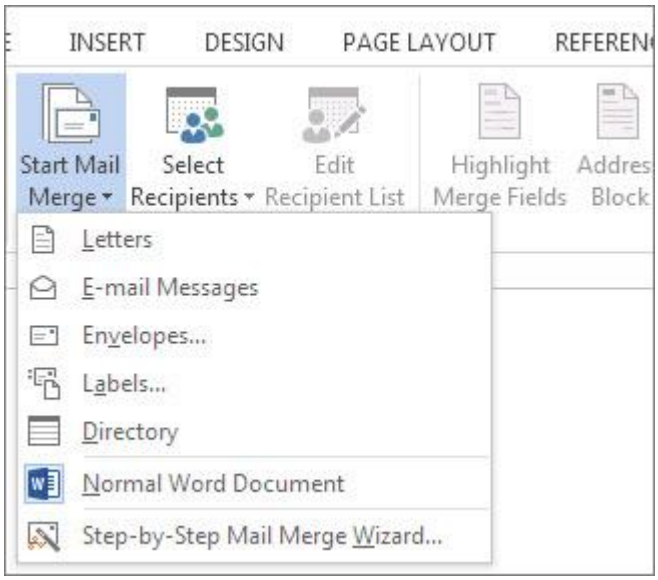
	A	B	C	D
1	OBJECT_ID	OBJECT_NAME	STATUS	
2	25509	A_TEST	VALID	
3	25510	IMAGE_BLOB	VALID	
4	25511	SYS_LOB0000025510C00002$$	VALID	

5. Creating DOCX template

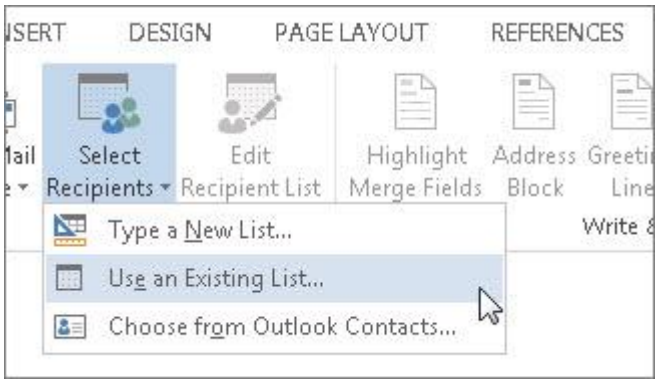
The main document contains the text and graphics that are the same for each version of the merged document. For example, the return address and the salutation in a form letter are the same for each version. This is a standard word processing step using MS Word or you can use an existing DOCX document too.



Set up the Mail Merge fields in Microsoft Office Word 2003 and in earlier versions of Word, point to Letters and Mailings on the Tools menu, and then click **Mail Merge Wizard**. You can setup template manually as a standard MS Office Mail Merge.



Choose Select Recipients > Use an Existing List

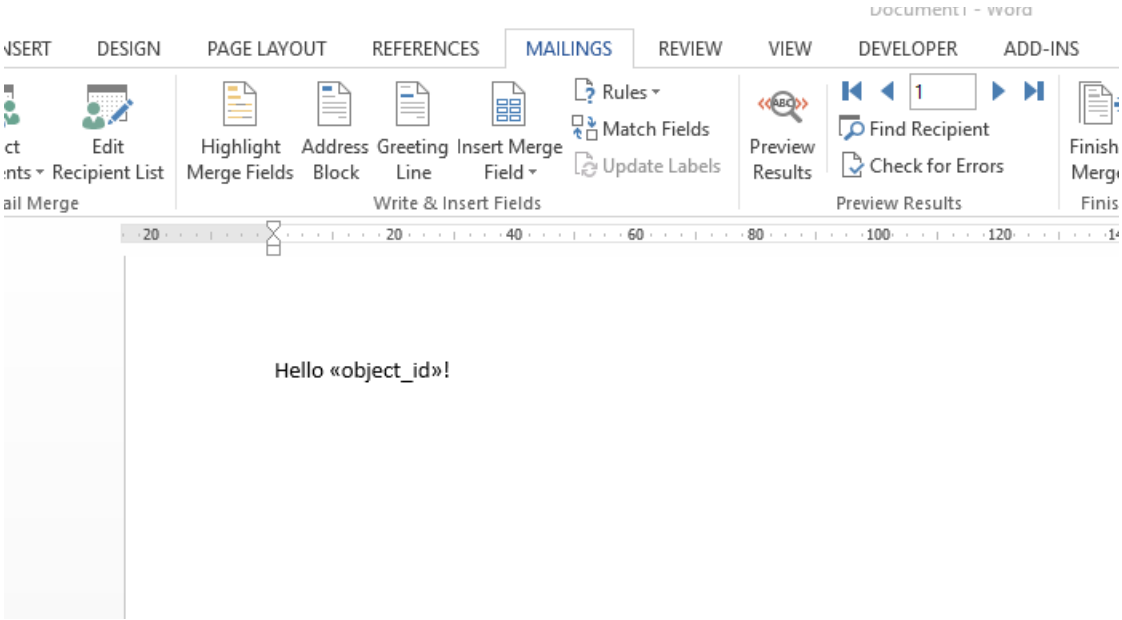


Browse to your Excel spreadsheet, and then choose Open.

You can insert one or more mail merge fields that pull the information from your spreadsheet into your document. To insert data from your spreadsheet in an email message or a letter:

- On the Mailings tab, in the Write & Insert Fields group, choose Insert Merge Field.
- In the Insert Merge Field dialog box, under Fields, choose a field name (column name in your spreadsheet), and then choose Insert.
- Repeat step 2 as needed, and choose Close when done.

- Choose File > Save.



After you insert the merge fields you want, preview the results to confirm that the content is okay. and then you're ready to complete the merge process.

On the Mailings tab, choose Preview Results.

You can finish the preview with clicking again on the Preview Results button.
Last step is saving document. When you save the mail merge document, it stays connected to your data source. You can reuse the mail merge document for your next bulk mailing.

6. Generate PL/SQL Code from 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, see PL/PDF Reporter User

The processing procedure is part of the plpdf_mailmerge package:
Create and compile full PL/SQL package (REQUIRED GRANT!: grant create procedure to plpdf;):

```
procedure createPDFpackage(
  p_packagename varchar2,
```

```
  p_file in out blob,
  p_encoding varchar2 default null,
  p_image_table varchar2 default null
  );
```

Create package header only:

```
function genPDFHeader(
  p_packagename varchar2,
  p_file in out blob
  ) return clob;
```

Create package body only:

```
function genPDFBody(
  p_packagename varchar2,
  p_file in out blob,
  p_encoding varchar2 default null,
  p_image_table varchar2 default null
  ) return clob;
```

And similar procedures for DOCX output:

```
procedure createDOCXpackage(
  p_packagename varchar2,
  p_file in out blob,
  p_encoding varchar2 default null,
  p_image_table varchar2 default null
  );

function genDOCXHeader(
  p_packagename varchar2,
  p_file in out blob
  ) return clob;

function genDOCXBody(
  p_packagename varchar2,
  p_file in out blob,
  p_encoding varchar2 default null,
  p_image_table varchar2 default null
  ) return clob;
```

Meaning of the parameters:

a. p_packagename: name of the generated PL/SQL package
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 image storage

The plpdf_mailmerge.createPDFpackage and plpdf_mailmerge.createDOCXpackage create a PL/PDF Generator conform PL/SQL package and compile into the database. If you want use different user as package owner please generate package header and body as CLOB value and compile it manually.

Example:

```
begin
  :result := plpdf_reporter_pdf.genMMHeader(p_packagename =>
:p_packagename,
                                    p_file => :p_file);

  execute immediate :result;

  -- Call the function
  :result := plpdf_reporter_pdf.genMMBody(p_packagename =>
:p_packagename,
                                    p_file => :p_file,
                                    p_encoding => :p_encoding,
                                    p_image_table => :p_image_table);

  execute immediate :result;
end;
```

The package contains the **t_params** structure as mail merge fields and **MailMergePDF** or **MailMergeDOCX** procedure calling the main mail merge function and the result is PDF or DOCX file as procedure name.

Our example package header:

```
create or replace package rep1 is
  type t_params is record(
    object_id   varchar2(4000 char),
    object_name varchar2(4000 char),
    status      varchar2(4000 char));

  procedure MailMergePDF(p_params t_params);

  v_pdf blob;
end;
```

and the body contains required PL/PDF calls.

7. Execute MailMerge

Runtime options:
- On Demand (One): direct call the PL/PDF Mail Merge Report from a custom Apex application or PL/SQL code
- Batch (Multiple): Oracle job like bulk PDF or DOCX creation

The direct call one result example:

```
declare
  l_params rep1.t_params;
begin
  delete from store_blob;
  commit;
    --set params
    l_params.object_id := 1;
    l_params.object_name := 'name';
    l_params.status := 'status';
    --run
```

```
    rep1.mailmergePDF(
     p_params => l_params
     );
    --save
    insert into store_blob (blob_file,created_date,filename) values (
rep1.v_pdf,sysdate, '1.pdf');
  end loop;
  commit;
end;
```

and the batch call

```
declare
  cursor c_obj is select object_id, object_name, status from
user_objects;
  l_params rep1.t_params;
begin
  delete from store_blob;
  commit;
  for f_obj in c_obj loop
    --set params
    l_params.object_id := f_obj.object_id;
    l_params.object_name := f_obj.object_name;
    l_params.status := f_obj.status;
    --run
    rep1.mailmergePDF(
     p_params => l_params
     );
    --save
    insert into store_blob (blob_file,created_date,filename) values (
rep1.v_pdf,sysdate, f_obj.object_id ||'.pdf');
  end loop;
  commit;
end;
```

Or you can use these packages in the Oracle Apex applications too.