ORA_EXCEL Professional

Reference guide

ORA_EXCEL is PL/SQL package for Oracle database
that produces Microsoft Excel XSLX documents

Contents

Package methods description and usage examples

Function ORA_EXCEL.new_document

Description:

pls_integer ORA_EXCEL.new_document

Create new Excel document

Mandatory parameters:

- No parameters

Optional parameters:

- No parameters

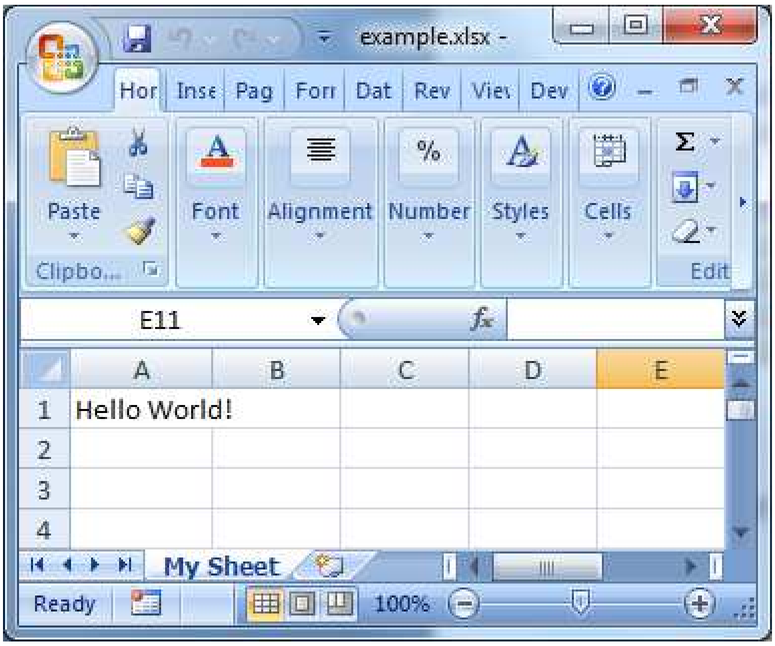
Returns:

Returns current document id

Example:

```
DECLARE
     doc_id PLS_INTEGER;
     sheet_id PLS_INTEGER;
     row_id PLS_INTEGER;
BEGIN
     doc_id := ORA_EXCEL.new_document;
     sheet_id := ORA_EXCEL.add_sheet('My Sheet', doc_id);
     row_id := ORA_EXCEL.add_row(doc_id, sheet_id);
     ORA_EXCEL.set_cell_value('A', 'Hello World!', doc_id, sheet_id, row_id);
     ORA_EXCEL.save_to_file('EXPORT_DIR', 'example.xlsx', doc_id);
END;
```

Output:



Hello World!

Procedure ORA_EXCEL.new_document

Description:

ORA_EXCEL.new_document

Create new Excel document and returns the document id.

Mandatory parameters:

- No parameters

Optional parameters:

- No parameters

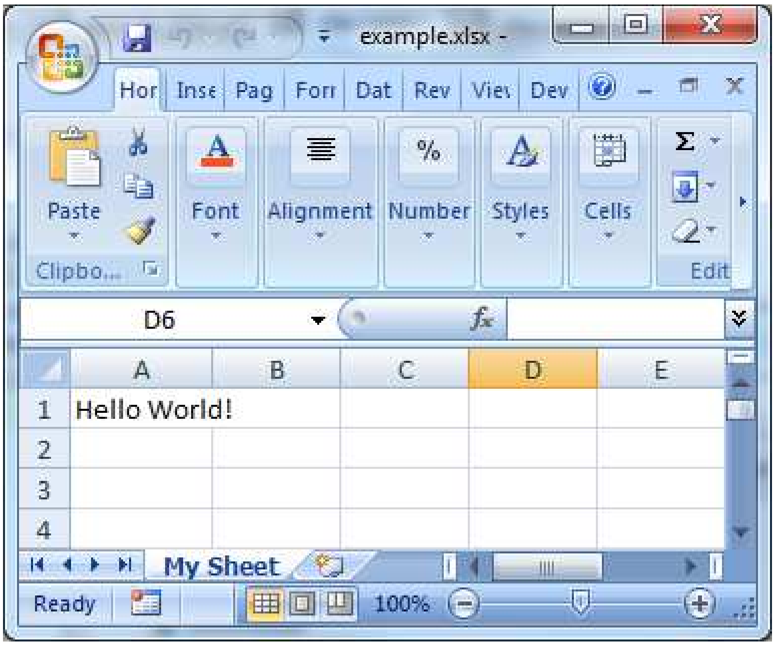
Returns:

Procedure, does not return any value.

Example:

```
BEGIN
      ORA_EXCEL.new_document;
      ORA_EXCEL.add_sheet('My Sheet');
      ORA_EXCEL.add_row;
      ORA_EXCEL.set_cell_value('A', 'Hello World!');
      ORA_EXCEL.save_to_file('EXPORT_DIR', 'example.xlsx');
END;
```

Output:



Hello World!

Function ORA_EXCEL.add_sheet

Description:

pls_integer ORA_EXCEL.add_sheet(sheet_name varchar2, [doc_id pls_integer])

Adds sheet to current document or document with document id specified with parameter doc_id

Mandatory parameters:

- sheet_name – name of added sheet, max. 31 characters long

Optional parameters:

- doc_id – document id, default value is current document id

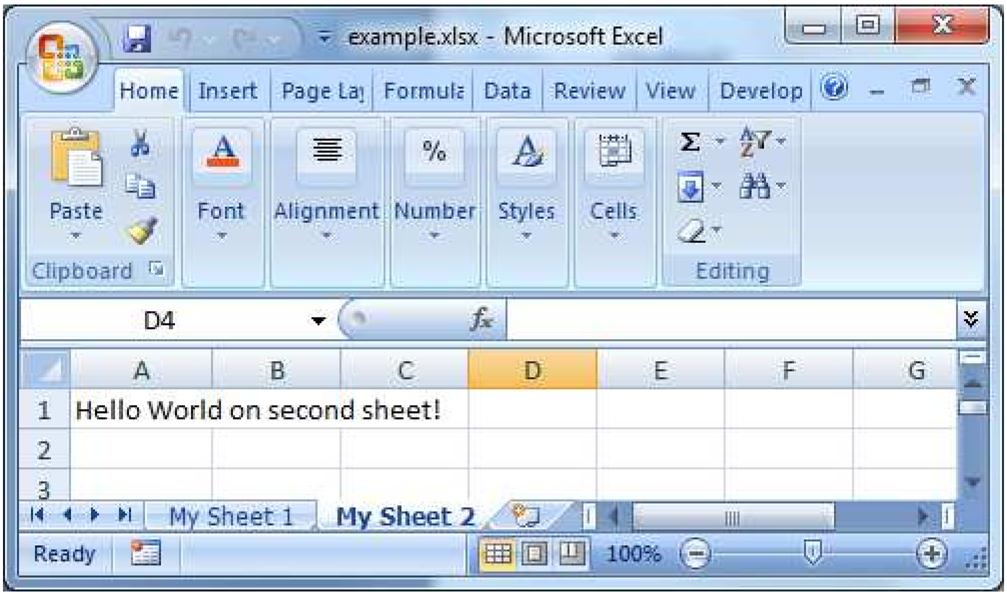
Returns:

Returns current sheet id

Example:

```
DECLARE
     doc_id PLS_INTEGER;
     sheet_id PLS_INTEGER;
     row_id PLS_INTEGER;
BEGIN
     doc_id := ORA_EXCEL.new_document;

     sheet_id := ORA_EXCEL.add_sheet('My Sheet 1', doc_id);
     row_id := ORA_EXCEL.add_row(doc_id, sheet_id);
     ORA_EXCEL.set_cell_value('A', 'Hello World!', doc_id, sheet_id, row_id);

     sheet_id := ORA_EXCEL.add_sheet('My Sheet 2', doc_id);
     row_id := ORA_EXCEL.add_row(doc_id, sheet_id);
     ORA_EXCEL.set_cell_value('A', 'Hello World on second sheet!', doc_id,
     sheet_id, row_id);

     ORA_EXCEL.save_to_file('EXPORT_DIR', 'example.xlsx');
END;
```

Output:



Hello World on second sheet!

Procedure ORA_EXCEL.add_sheet

Description:

ORA_EXCEL.add_sheet(sheet_name varchar2, [doc_id pls_integer])

Adds sheet to current document or document with id of the document specified with parameter doc_id

Mandatory parameters:

- sheet_name – name of added sheet, max. 31 characters long

Optional parameters:

- doc_id – document id, default value is current document id

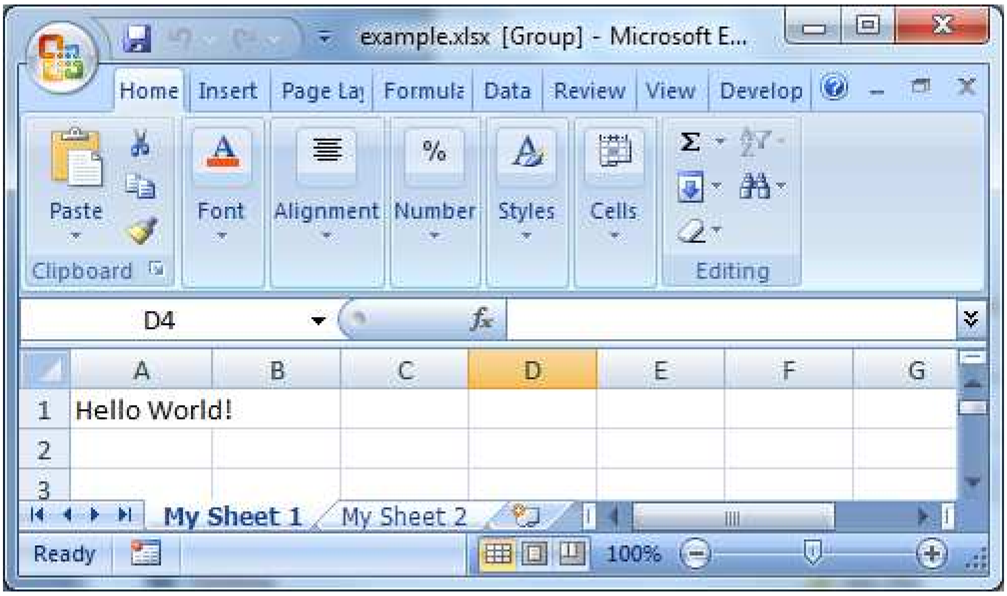
Returns:

Procedure, does not return any value.

Example:

```
BEGIN
     ORA_EXCEL.new_document;

     ORA_EXCEL.add_sheet('My Sheet 1');
     ORA_EXCEL.add_row;
     ORA_EXCEL.set_cell_value('A', 'Hello World!');

     ORA_EXCEL.add_sheet('My Sheet 2');
     ORA_EXCEL.add_row;
     ORA_EXCEL.set_cell_value('A', 'Hello World on second sheet!');

     ORA_EXCEL.save_to_file('EXPORT_DIR', 'example.xlsx');
END;
```

Output:



Hello World!

Function ORA_EXCEL.add_row

Description:

pls_integer ORA_EXCEL.add_row([doc_id pls_integer, sheet_id pls_integer])

Adds row to current sheet or sheet with id of the sheet specified with parameter sheet_id

Mandatory parameters:

- no parameters

Optional parameters:

- doc_id – id of the document of document where row will be added
- sheet_id – id of the sheet where row will be added

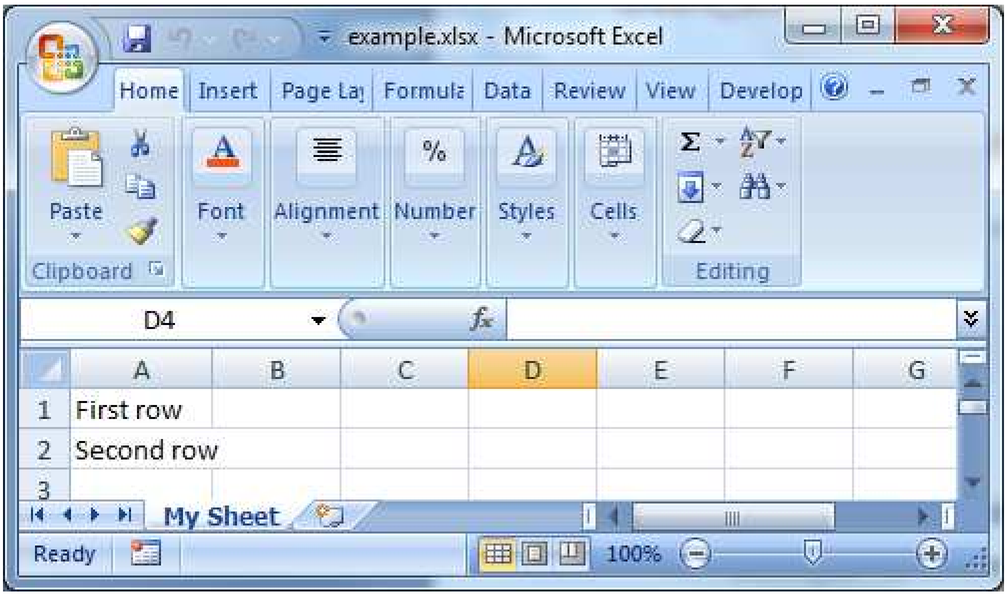
Returns:

Returns current row id

Example:

```
DECLARE
     doc_id PLS_INTEGER;
     sheet_id PLS_INTEGER;
     row_id PLS_INTEGER;
BEGIN
     doc_id := ORA_EXCEL.new_document;

     sheet_id := ORA_EXCEL.add_sheet('My Sheet', doc_id);
     row_id := ORA_EXCEL.add_row(doc_id, sheet_id);
     ORA_EXCEL.set_cell_value('A', 'First row', doc_id, sheet_id, row_id);

     row_id := ORA_EXCEL.add_row(doc_id, sheet_id);
     ORA_EXCEL.set_cell_value('A', 'Second row', doc_id, sheet_id, row_id);

     ORA_EXCEL.save_to_file('EXPORT_DIR', 'example.xlsx');
END;
```

Output:



	A	B	C	D	E	F	G
1	First row						
2	Second row						
3							

Procedure ORA_EXCEL.add_row

Description:

ORA_EXCEL.add_row([doc_id pls_integer, sheet_id pls_integer])

Adds row to current sheet or sheet with id of the sheet specified with parameter sheet_id

Mandatory parameters:

- no parameters

Optional parameters:

- doc_id – id of the document of document where row will be added
- sheet_id – id of the sheet where row will be added

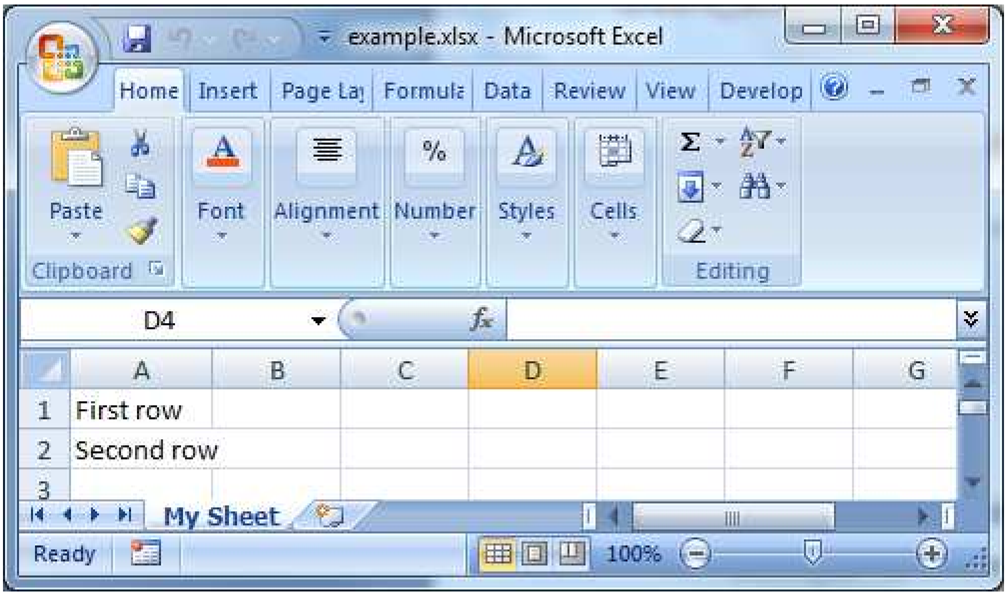
Returns:

Procedure, does not return any value

Example:

```
BEGIN
	ORA_EXCEL.new_document;

	ORA_EXCEL.add_sheet('My Sheet');

	ORA_EXCEL.add_row;
	ORA_EXCEL.set_cell_value('A', 'First row');

	ORA_EXCEL.add_row;
	ORA_EXCEL.set_cell_value('A', 'Second row');

	ORA_EXCEL.save_to_file('EXPORT_DIR', 'example.xlsx');
END;
```

Output:



	A
1	First row
2	Second row
3	

Procedure ORA_EXCEL.set_row_height

Description:

ORA_EXCEL.set_row_height(height number [, doc_id pls_integer, sheet_id pls_integer, row_id pls_integer])

Change row height

Mandatory parameters:

- height – row height

Optional parameters:

- doc_id – id of document
- sheet_id – id of sheet
- row_id – id of row on which height will be adjusted

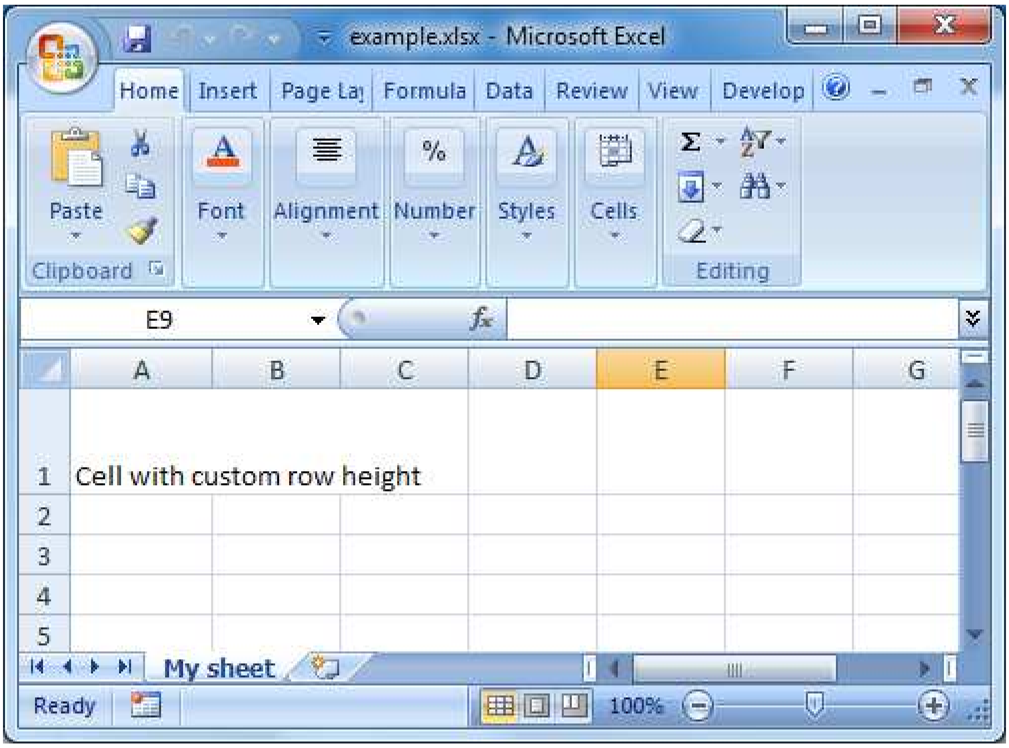
Returns:

Procedure, does not return any value

Example:

```
BEGIN
     ORA_EXCEL.new_document;
     ORA_EXCEL.add_sheet('My sheet');
     ORA_EXCEL.add_row;
     ORA_EXCEL.set_row_height(40);
     ORA_EXCEL.set_cell_value('A', 'Cell with custom row height');
     ORA_EXCEL.save_to_file('EXPORT_DIR', 'example.xlsx');
END;
```

Output:



Cell with custom row height

Procedure ORA_EXCEL.set_cell_value

Description:

ORA_EXCEL.set_cell_value(name varchar2, value varchar2, [doc_id pls_integer, sheet_id pls_integer, row_id pls_integer])

Set cell value with string value type

Mandatory parameters:

- name – cell name example 'A' (first cell in the sheet)
- value – cell value

Optional parameters:

- doc_id – id of document
- sheet_id – id of sheet
- row_id – id of row on which height will be adjusted

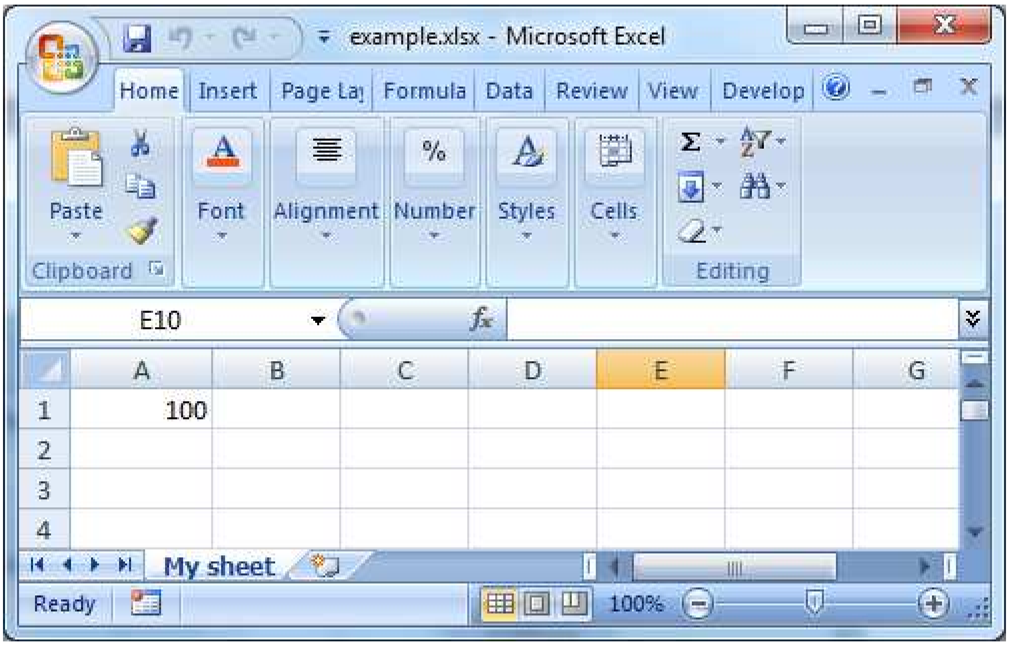
Returns:

Procedure, does not return any value

Example:

```
BEGIN
        ORA_EXCEL.new_document;
        ORA_EXCEL.add_sheet('My sheet');
        ORA_EXCEL.add_row;
        ORA_EXCEL.set_row_height(15);
        ORA_EXCEL.set_cell_value('A',  100);
        ORA_EXCEL.save_to_file('EXPORT_DIR',  'example.xlsx');
END;
```

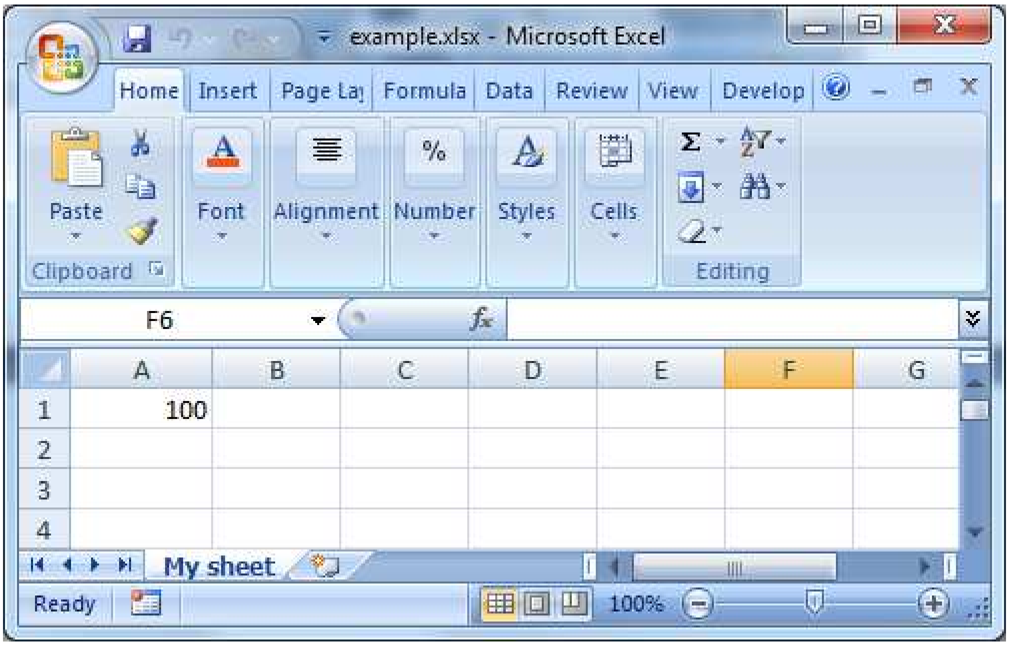
Output:



Example:

```
DECLARE
      doc_id PLS_INTEGER;
      sheet_id PLS_INTEGER;
      row_id PLS_INTEGER;
BEGIN
      doc_id := ORA_EXCEL.new_document;
      sheet_id := ORA_EXCEL.add_sheet('My sheet', doc_id);
      row_id := ORA_EXCEL.add_row(doc_id, sheet_id);
      ORA_EXCEL.set_row_height(15, doc_id, sheet_id, row_id);
      ORA_EXCEL.set_cell_value('A', 100, doc_id, sheet_id, row_id);
      ORA_EXCEL.save_to_file('EXPORT_DIR', 'example.xlsx');
END;
```

Output:



Procedure ORA_EXCEL.set_cell_value

Description:

ORA_EXCEL.set_cell_value(name varchar2, value date, [doc_id pls_integer, sheet_id pls_integer, row_id pls_integer])

Set cell value with date value type

Mandatory parameters:

- name – cell name example 'A' (first cell in the sheet)
- value – cell value

Optional parameters:

- doc_id – id of document
- sheet_id – id of sheet
- row_id – id of row on which height will be adjusted

Returns:

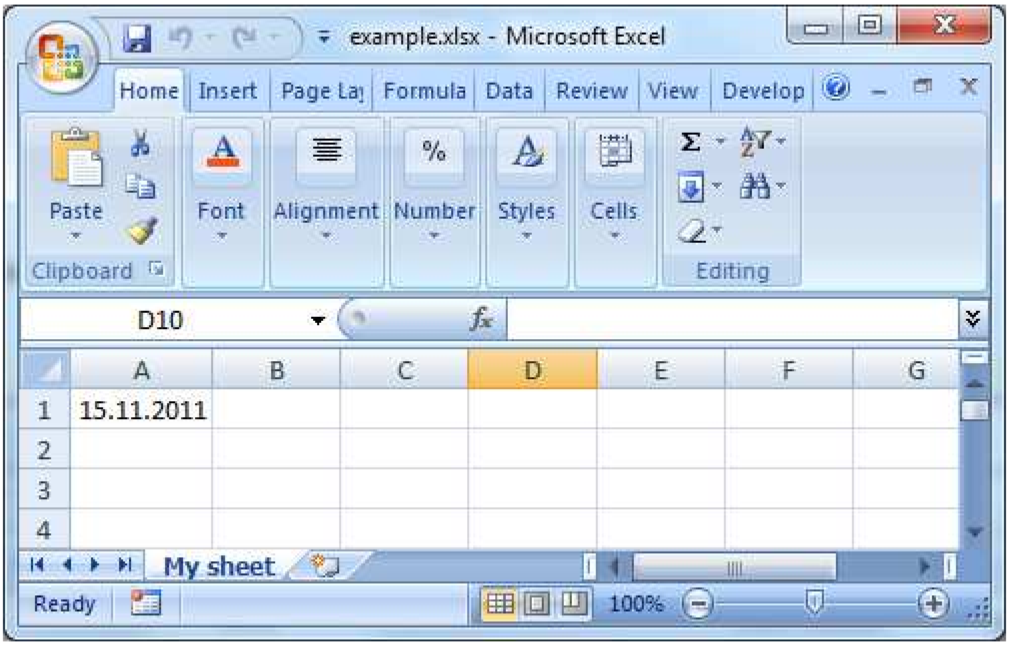
Procedure, does not return any value

Example:

```
DECLARE
      doc_id PLS_INTEGER;
      sheet_id PLS_INTEGER;
      row_id PLS_INTEGER;
BEGIN
      doc_id := ORA_EXCEL.new_document;
      sheet_id := ORA_EXCEL.add_sheet('My sheet', doc_id);
      row_id := ORA_EXCEL.add_row(doc_id, sheet_id);
      ORA_EXCEL.set_row_height(15, doc_id, sheet_id, row_id);
```

```
     ORA_EXCEL.set_cell_value('A', TO_DATE('2011-11-15', 'yyyy-mm-dd'), doc_id,
     sheet_id, row_id);
     ORA_EXCEL.save_to_file('EXPORT_DIR', 'example.xlsx');
END;
```

Output:

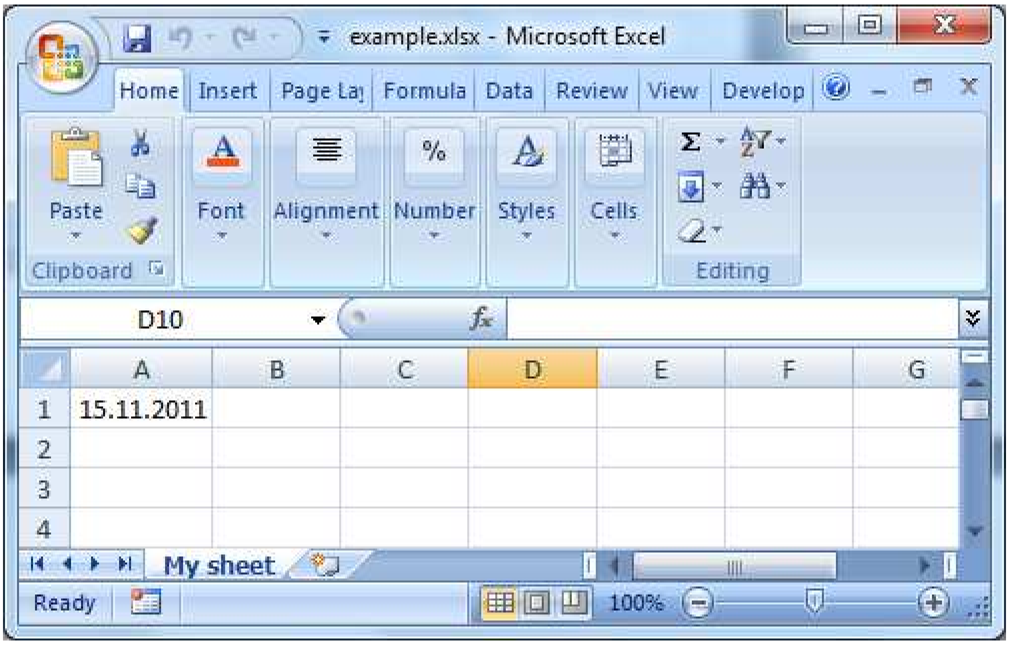


	A	B	C	D	E	F	G
1	15.11.2011						
2							
3							
4							

Example:

```
BEGIN
     ORA_EXCEL.new_document;
     ORA_EXCEL.add_sheet('My sheet');
     ORA_EXCEL.add_row;
     ORA_EXCEL.set_cell_value('A', TO_DATE('2011-11-15', 'yyyy-mm-dd'));
     ORA_EXCEL.save_to_file('EXPORT_DIR', 'example.xlsx');
END;
```

Output:



	A	B	C	D	E	F	G
1	15.11.2011						
2							
3							
4							

Procedure ORA_EXCEL.set_cell_value

Description:

ORA_EXCEL.set_cell_value(name varchar2, value number, [doc_id pls_integer, sheet_id pls_integer, row_id pls_integer])

Set cell value with number value type

Mandatory parameters:

- name – cell name example 'A' (first cell in the sheet)
- value – cell value

Optional parameters:

- doc_id – id of document
- sheet_id – id of sheet
- row_id – id of row on which height will be adjusted

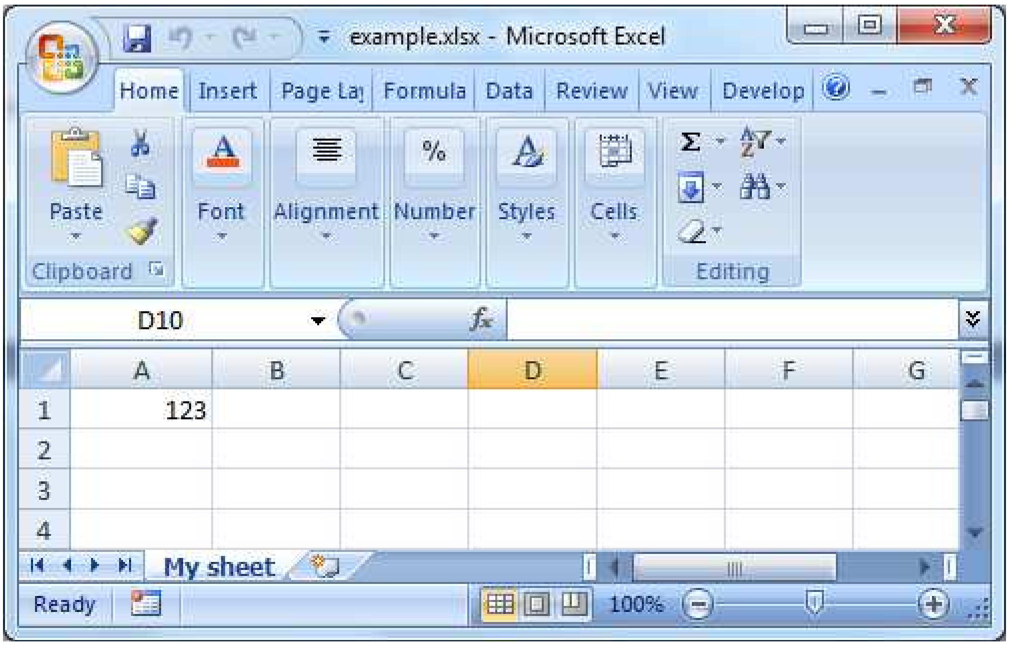
Returns:

Procedure, does not return any value

Example:

```
BEGIN
     ORA_EXCEL.new_document;
     ORA_EXCEL.add_sheet('My sheet');
     ORA_EXCEL.add_row;
     ORA_EXCEL.set_row_height(15);
     ORA_EXCEL.set_cell_value('A', 123);
     ORA_EXCEL.save_to_file('EXPORT_DIR', 'example.xlsx');
END;
```

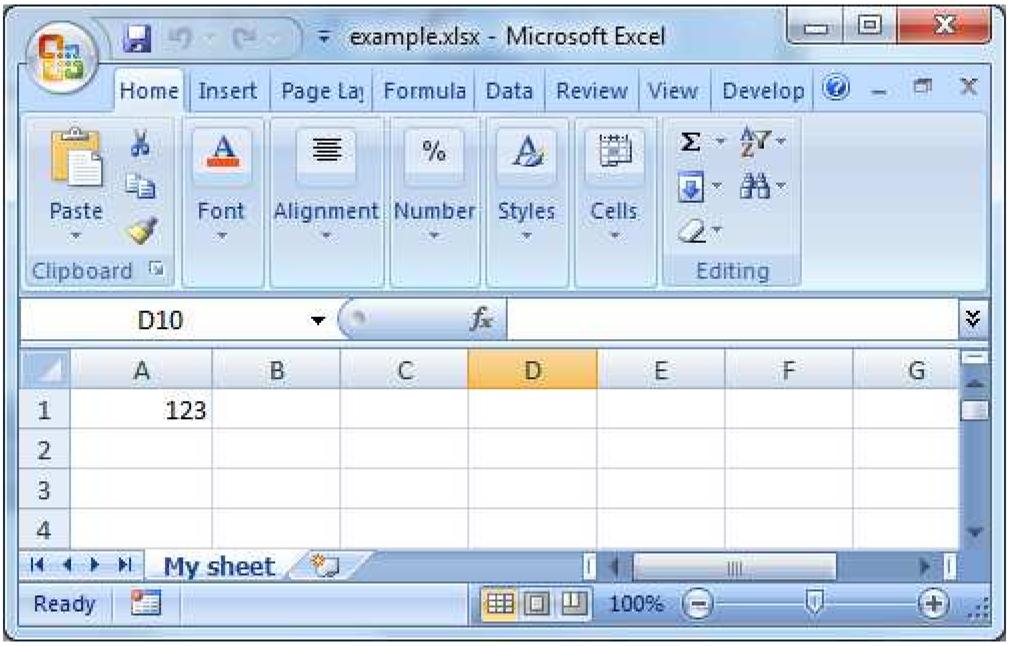
Output:



Example:

```
DECLARE
     doc_id PLS_INTEGER;
     sheet_id PLS_INTEGER;
     row_id PLS_INTEGER;
BEGIN
     doc_id := ORA_EXCEL.new_document;
     sheet_id := ORA_EXCEL.add_sheet('My sheet', doc_id);
     row_id := ORA_EXCEL.add_row(doc_id, sheet_id);
     ORA_EXCEL.set_row_height(15, doc_id, sheet_id, row_id);
     ORA_EXCEL.set_cell_value('A', 123, doc_id, sheet_id, row_id);
     ORA_EXCEL.save_to_file('EXPORT_DIR', 'example.xlsx', doc_id);
END;
```

Output:



Procedure ORA_EXCEL.set_cell_font

Description:

ORA_EXCEL.set_cell_font(cell_name varchar, font_name varchar2 [, font_size pls_integer, doc_id pls_integer, sheet_id pls_integer, row_id pls_integer])

Set cell font family and size

Mandatory parameters:

- cell_name – cell name
- font_name – font family

Optional parameters:

- font_size – font size
- doc_id – document id
- sheet_id – sheet id
- row_id – row id

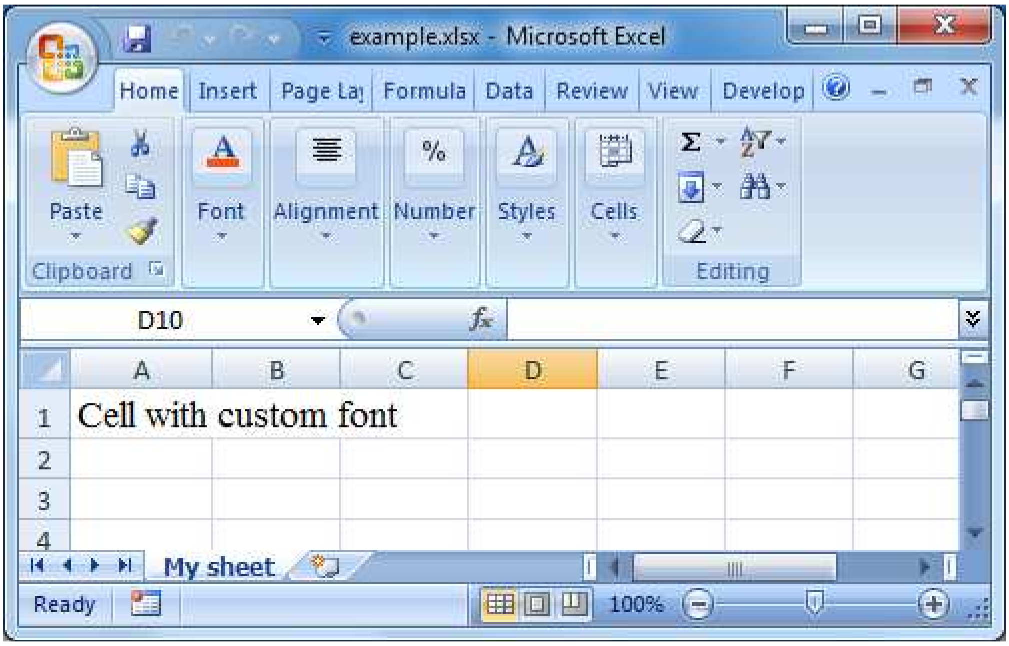
Returns:

Procedure, does not return any value

Example:

```
BEGIN
     ORA_EXCEL.new_document;
     ORA_EXCEL.add_sheet('My sheet');
     ORA_EXCEL.add_row;
     ORA_EXCEL.set_cell_value('A', 'Cell with custom font');
     ORA_EXCEL.set_cell_font('A', 'Times New Roman', 14);
     ORA_EXCEL.save_to_file('EXPORT_DIR', 'example.xlsx');
END;
```

Output:

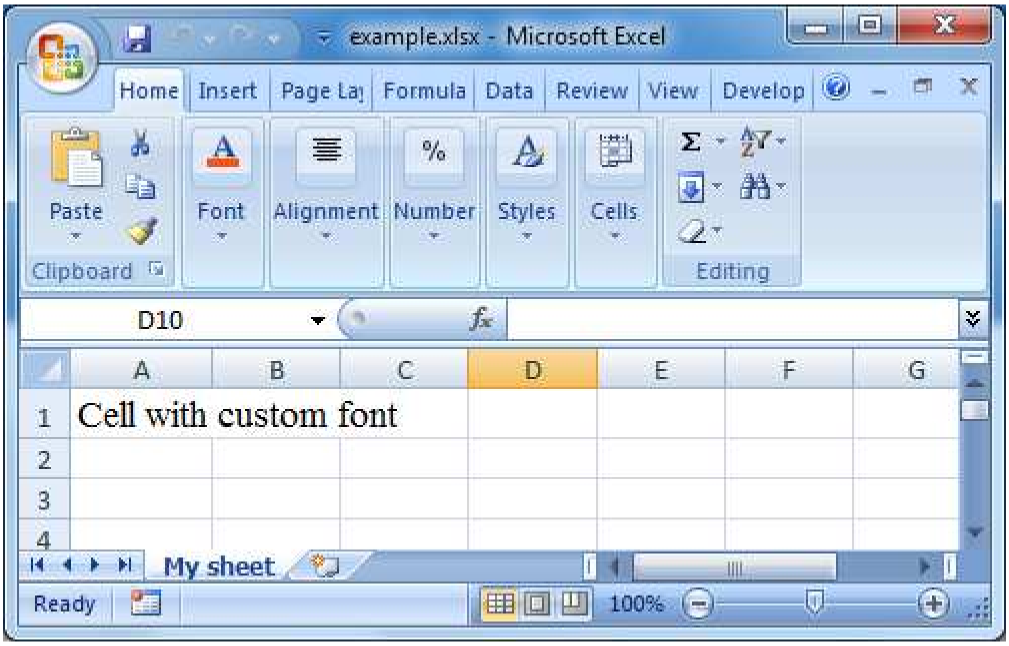


Cell with custom font

Example:

```
DECLARE
     doc_id PLS_INTEGER;
     sheet_id PLS_INTEGER;
     row_id PLS_INTEGER;
BEGIN
     doc_id := ORA_EXCEL.new_document;
     sheet_id := ORA_EXCEL.add_sheet('My sheet', doc_id);
     row_id := ORA_EXCEL.add_row(doc_id, sheet_id);
     ORA_EXCEL.set_cell_value('A', 'Cell with custom font', doc_id,
     sheet_id, row_id);
     ORA_EXCEL.set_cell_font('A', 'Times New Roman', 14);
     ORA_EXCEL.save_to_file('EXPORT_DIR', 'example.xlsx', doc_id);
END;
```

Output:



Cell with custom font

Procedure ORA_EXCEL.set_default_font

Description:

ORA_EXCEL.set_default_font(font_name varchar2, font_size pls_integer [, doc_id pls_integer])

Set default font family and size for whole document

Mandatory parameters:

- font_name – font family
- font_size – font size

Optional parameters:

- doc_id – document id
- sheet_id – sheet id
- row_id – row id

Returns:

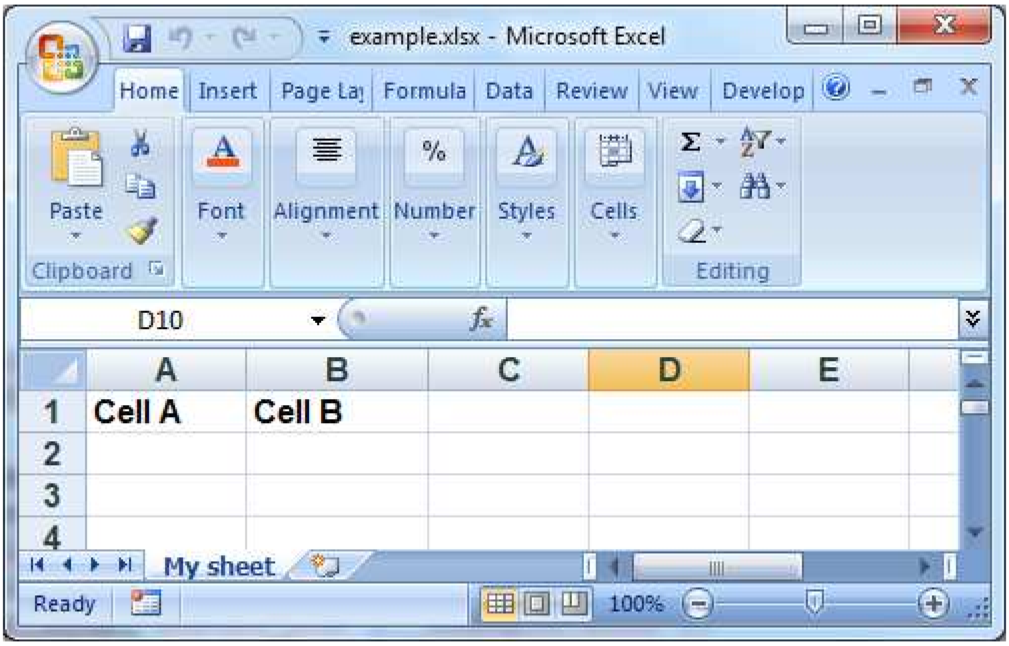
Procedure, does not return any value

Example:

```
BEGIN
      ORA_EXCEL.new_document;
      ORA_EXCEL.set_default_font('Arial bold', 12);

      ORA_EXCEL.add_sheet('My sheet');
      ORA_EXCEL.add_row;
      ORA_EXCEL.set_cell_value('A', 'Cell A');
      ORA_EXCEL.set_cell_value('B', 'Cell B');
```

```
     ORA_EXCEL.save_to_file('EXPORT_DIR', 'example.xlsx');
END;
```

Output:

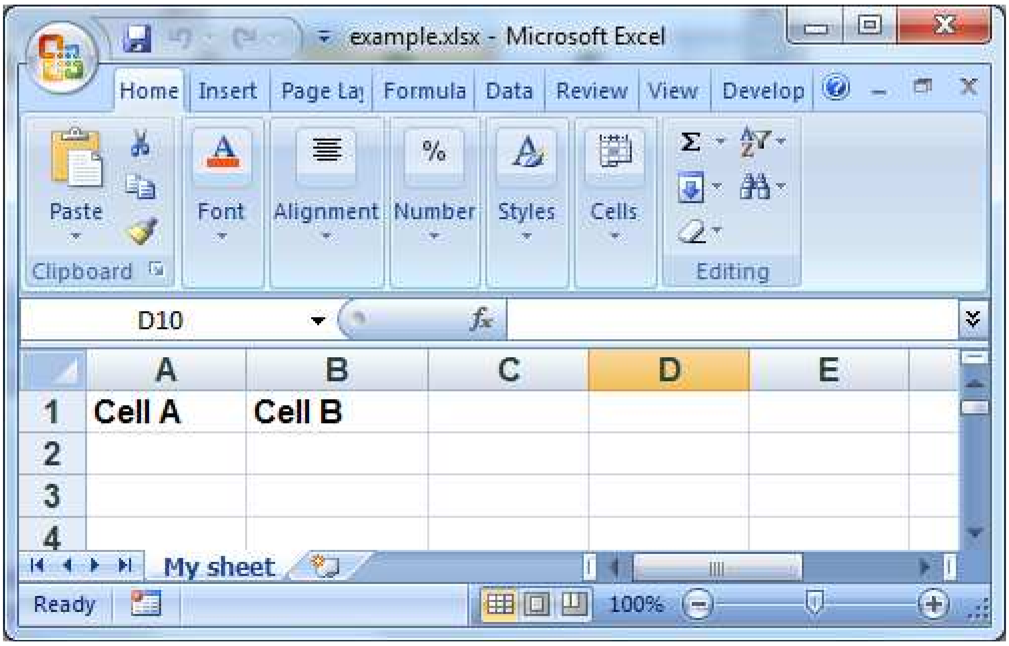


	A	B	C	D	E
1	Cell A	Cell B			
2					
3					
4					

Example:

```
DECLARE
     doc_id PLS_INTEGER;
     sheet_id PLS_INTEGER;
     row_id PLS_INTEGER;
BEGIN
     doc_id := ORA_EXCEL.new_document;
     ORA_EXCEL.set_default_font('Arial bold', 12);

     sheet_id := ORA_EXCEL.add_sheet('My sheet', doc_id);
     row_id := ORA_EXCEL.add_row(doc_id, sheet_id);
     ORA_EXCEL.set_cell_value('A', 'Cell A', doc_id, sheet_id, row_id);
     ORA_EXCEL.set_cell_value('B', 'Cell B', doc_id, sheet_id, row_id);
     ORA_EXCEL.save_to_file('EXPORT_DIR', 'example.xlsx', doc_id);
END;
```

Output:



	A	B	C	D	E
1	Cell A	Cell B			
2					
3					
4					

Procedure ORA_EXCEL.set_cell_bold

Description:

ORA_EXCEL.set_cell_bold(name varchar2 [, doc_id pls_integer, sheet_id pls_integer, row_id pls_integer])

Set the cell text bold

Mandatory parameters:

- name – cell name

Optional parameters:

- doc_id – document id
- sheet_id – sheet id
- row_id – row id

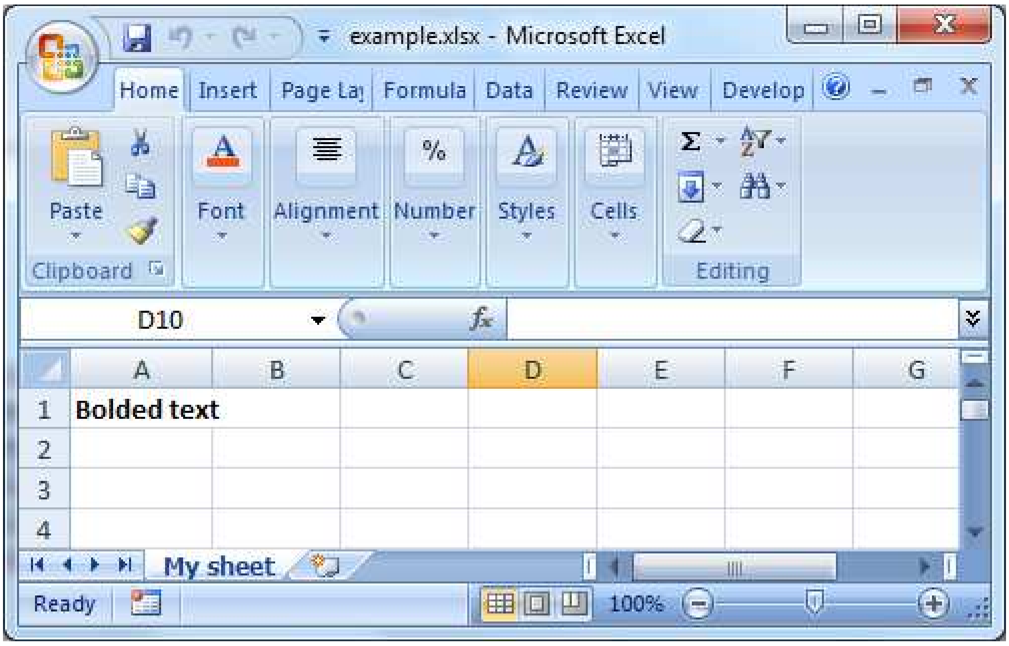
Returns:

Procedure, does not return any value

Example:

```
BEGIN
      ORA_EXCEL.new_document;
      ORA_EXCEL.add_sheet('My sheet');
      ORA_EXCEL.add_row;
      ORA_EXCEL.set_cell_value('A', 'Bolded text');
      ORA_EXCEL.set_cell_bold('A');
      ORA_EXCEL.save_to_file('EXPORT_DIR', 'example.xlsx');
END;
```

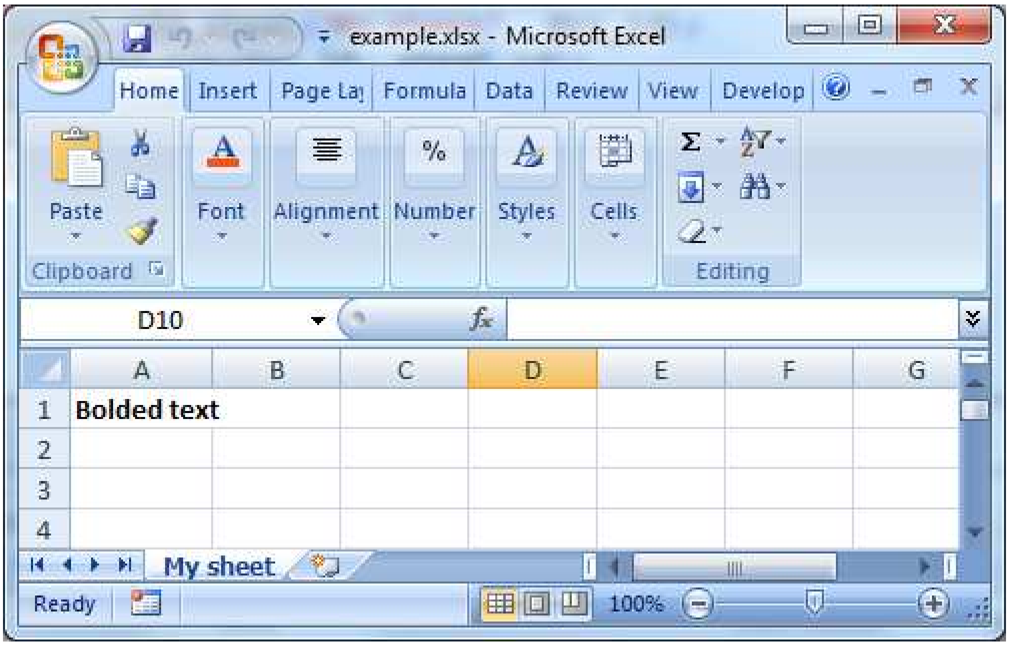
Output:



Example:

```
DECLARE
     doc_id PLS_INTEGER;
     sheet_id PLS_INTEGER;
     row_id PLS_INTEGER;
BEGIN
     doc_id := ORA_EXCEL.new_document;
     sheet_id := ORA_EXCEL.add_sheet('My sheet', doc_id);
     row_id := ORA_EXCEL.add_row(doc_id, sheet_id);
     ORA_EXCEL.set_cell_value('A', 'Bolded text', doc_id, sheet_id, row_id);
     ORA_EXCEL.set_cell_bold('A', doc_id, sheet_id, row_id);
     ORA_EXCEL.save_to_file('EXPORT_DIR', 'example.xlsx', doc_id);
END;
```

Output:



Bolded text

Procedure ORA_EXCEL.set_cell_italic

Description:

ORA_EXCEL.set_cell_italic(name varchar2 [, doc_id pls_integer, sheet_id pls_integer, row_id pls_integer])

Set the cell text italic

Mandatory parameters:

- name – cell name

Optional parameters:

- doc_id – document id
- sheet_id – sheet id
- row_id – row id

Returns:

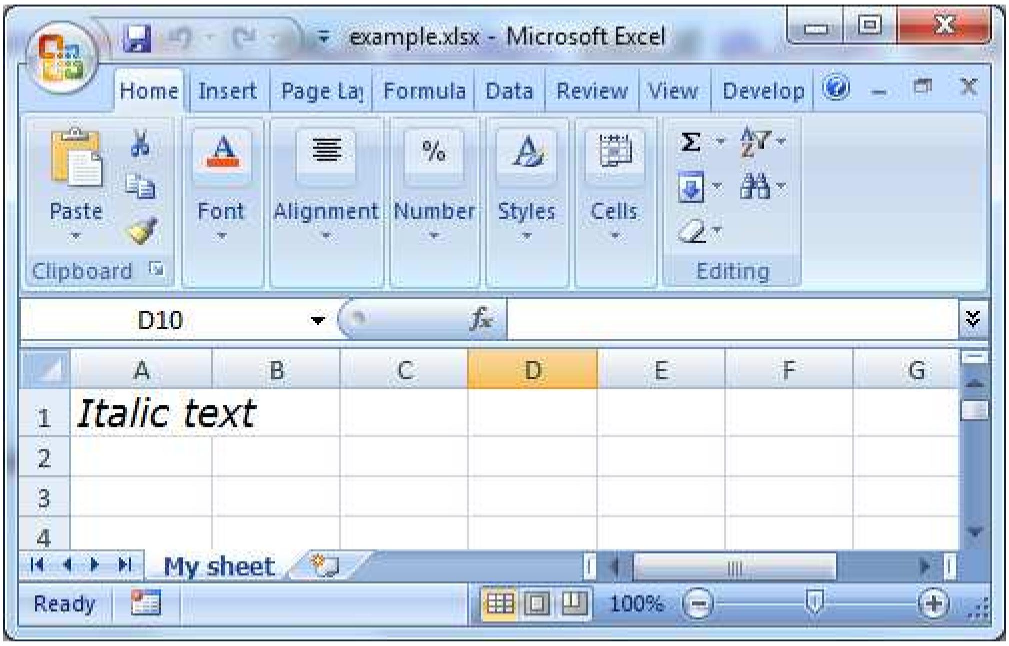
Procedure, does not return any value

Example:

```
BEGIN
     ORA_EXCEL.new_document;
     ORA_EXCEL.add_sheet('My sheet');
     ORA_EXCEL.add_row;
     ORA_EXCEL.set_cell_value('A', 'Italic text');
     ORA_EXCEL.set_cell_font('A', 'Verdana', 14);
     ORA_EXCEL.set_cell_italic('A');
     ORA_EXCEL.save_to_file('EXPORT_DIR', 'example.xlsx');
```

```
END;
```

Output:

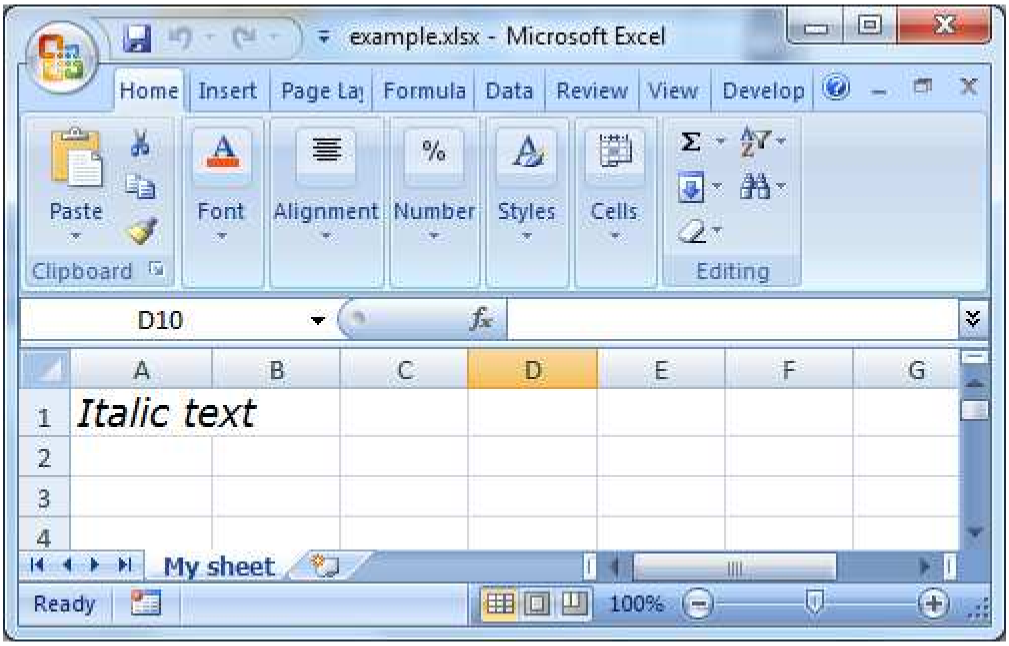


	A	B	C	D	E	F	G
1	*Italic text*						
2							
3							
4							

Example:

```
DECLARE
      doc_id  PLS_INTEGER;
      sheet_id  PLS_INTEGER;
      row_id  PLS_INTEGER;
BEGIN
      doc_id  := ORA_EXCEL.new_document;
      sheet_id  := ORA_EXCEL.add_sheet('My sheet', doc_id);
      row_id  := ORA_EXCEL.add_row(doc_id, sheet_id);
      ORA_EXCEL.set_cell_value('A', 'Italic text', doc_id, sheet_id, row_id);
      ORA_EXCEL.set_cell_font('A', 'Verdana', 14, doc_id, sheet_id, row_id);
      ORA_EXCEL.set_cell_italic('A', doc_id, sheet_id, row_id);
      ORA_EXCEL.save_to_file('EXPORT_DIR', 'example.xlsx', doc_id);
END;
```

Output:



Italic text

Procedure ORA_EXCEL.set_cell_underline

Description:

```
ORA_EXCEL.set_cell_underline(name varchar2 [, doc_id pls_integer, sheet_id pls_integer, row_id pls_integer])
```

Set the cell text underline

Mandatory parameters:

- name – cell name

Optional parameters:

- doc_id – document id
- sheet_id – sheet id
- row_id – row id

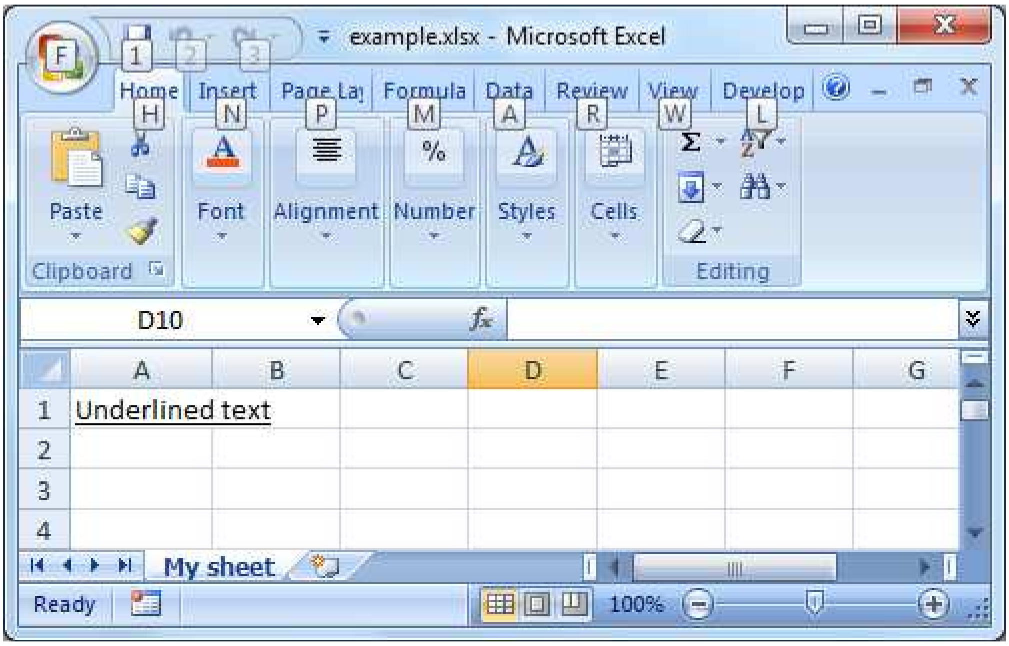
Returns:

Procedure, does not return any value

Example:

```
BEGIN
        ORA_EXCEL.new_document;
        ORA_EXCEL.add_sheet('My sheet');
        ORA_EXCEL.add_row;
        ORA_EXCEL.set_cell_value('A', 'Underlined text');
        ORA_EXCEL.set_cell_underline('A');
        ORA_EXCEL.save_to_file('EXPORT_DIR', 'example.xlsx');
END;
```

Output:

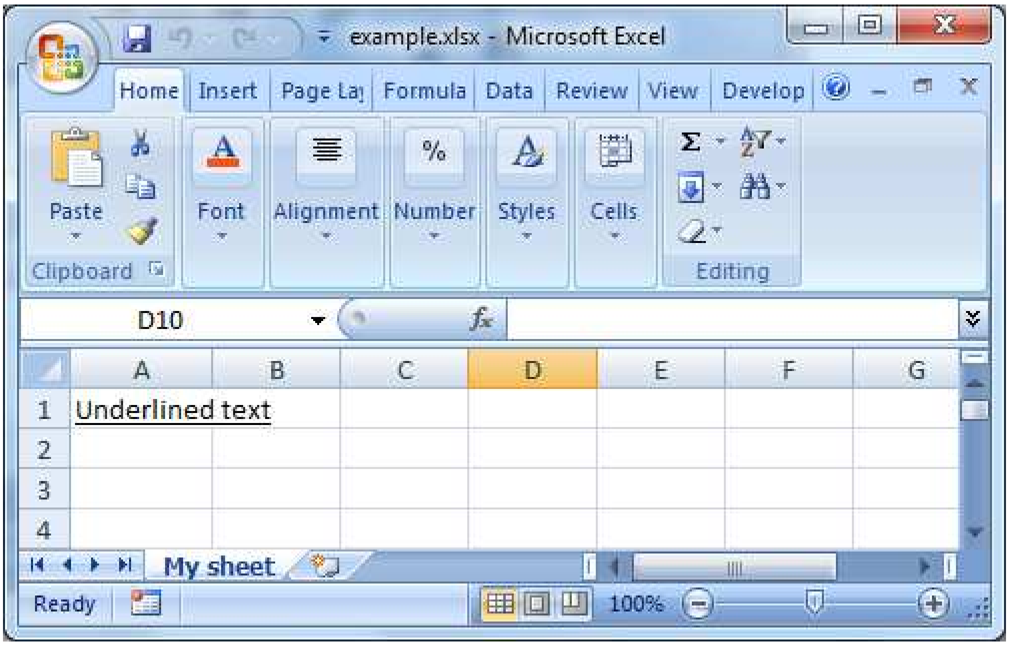


	A	B	C	D	E	F	G
1	Underlined text						
2							
3							
4							

Example:

```
DECLARE
     doc_id PLS_INTEGER;
     sheet_id PLS_INTEGER;
     row_id PLS_INTEGER;
BEGIN
     doc_id := ORA_EXCEL.new_document;
     sheet_id := ORA_EXCEL.add_sheet('My sheet', doc_id);
     row_id := ORA_EXCEL.add_row(doc_id, sheet_id);
     ORA_EXCEL.set_cell_value('A', 'Underlined text', doc_id, sheet_id, row_id);
     ORA_EXCEL.set_cell_underline('A', doc_id, sheet_id, row_id);
     ORA_EXCEL.save_to_file('EXPORT_DIR', 'example.xlsx', doc_id);
END;
```

Output:



Underlined text

Procedure ORA_EXCEL.set_cell_color

Description:

ORA_EXCEL.set_cell_color(name varchar2, color varchar2 [, doc_id pls_integer, sheet_id pls_integer, row_id pls_integer])

Set the text color

Mandatory parameters:

- name – cell name
- color – RGB color in hex format (for example FF0000 – red)

Optional parameters:

- doc_id – document id
- sheet_id – sheet id
- row_id – row id

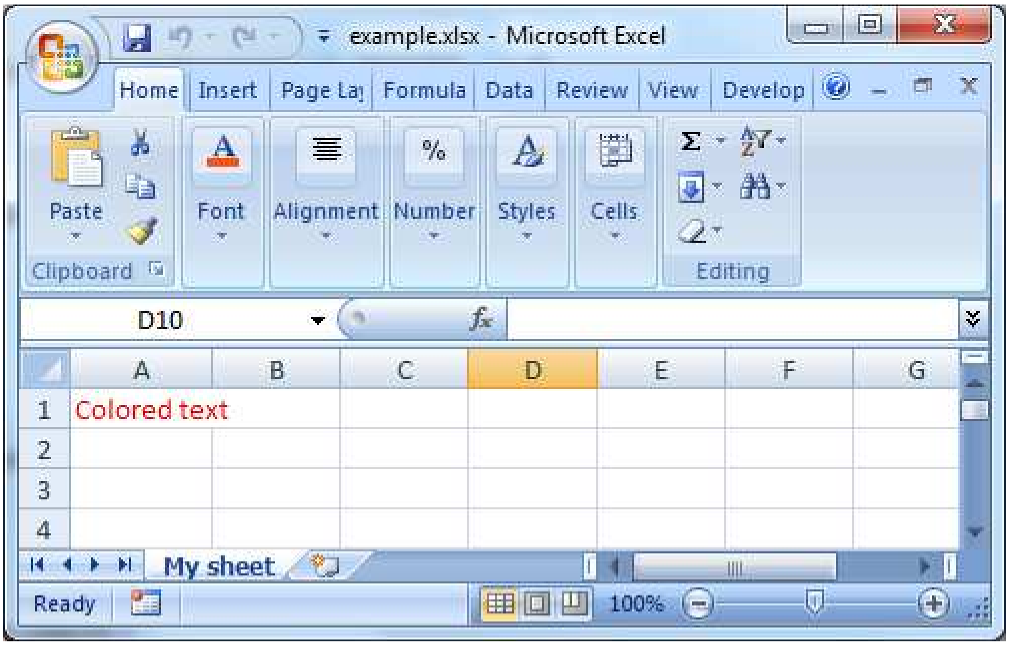
Returns:

Procedure, does not return any value

Example:

```
BEGIN
     ORA_EXCEL.new_document;
     ORA_EXCEL.add_sheet('My sheet');
     ORA_EXCEL.add_row;
     ORA_EXCEL.set_cell_value('A', 'Colored text');
     ORA_EXCEL.set_cell_color('A', 'FF0000');
     ORA_EXCEL.save_to_file('EXPORT_DIR', 'example.xlsx');
END;
```

Output:

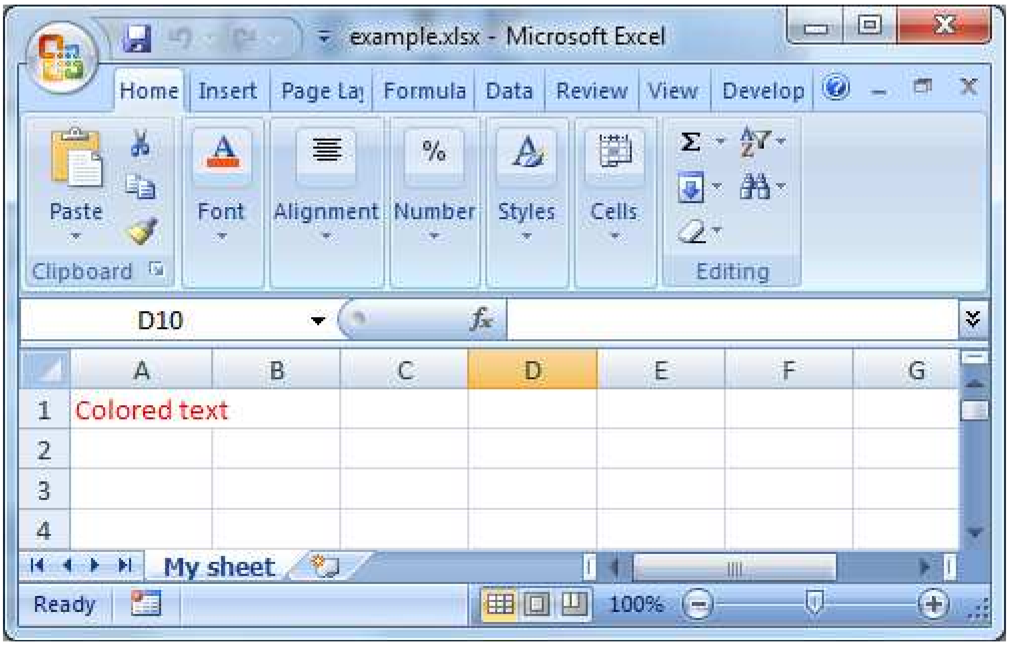


Colored text

Example:

```
DECLARE
     doc_id PLS_INTEGER;
     sheet_id PLS_INTEGER;
     row_id PLS_INTEGER;
BEGIN
     doc_id := ORA_EXCEL.new_document;
     sheet_id := ORA_EXCEL.add_sheet('My sheet', doc_id);
     row_id := ORA_EXCEL.add_row(doc_id, sheet_id);
     ORA_EXCEL.set_cell_value('A', 'Colored text', doc_id, sheet_id, row_id);
     ORA_EXCEL.set_cell_color('A', 'FF0000', doc_id, sheet_id, row_id);
     ORA_EXCEL.save_to_file('EXPORT_DIR', 'example.xlsx', doc_id);
END;
```

Output:



A
Colored text

Procedure ORA_EXCEL.set_cell_bg_color

Description:

ORA_EXCEL.set_cell_bg_color(name varchar2, color varchar2 [, doc_id pls_integer, sheet_id pls_integer, row_id pls_integer])

Color the background of cell

Mandatory parameters:

- name – cell name
- color – RGB color in hex format (for example FF0000 – red)

Optional parameters:

- doc_id – document id
- sheet_id – sheet id
- row_id – row id

Returns:

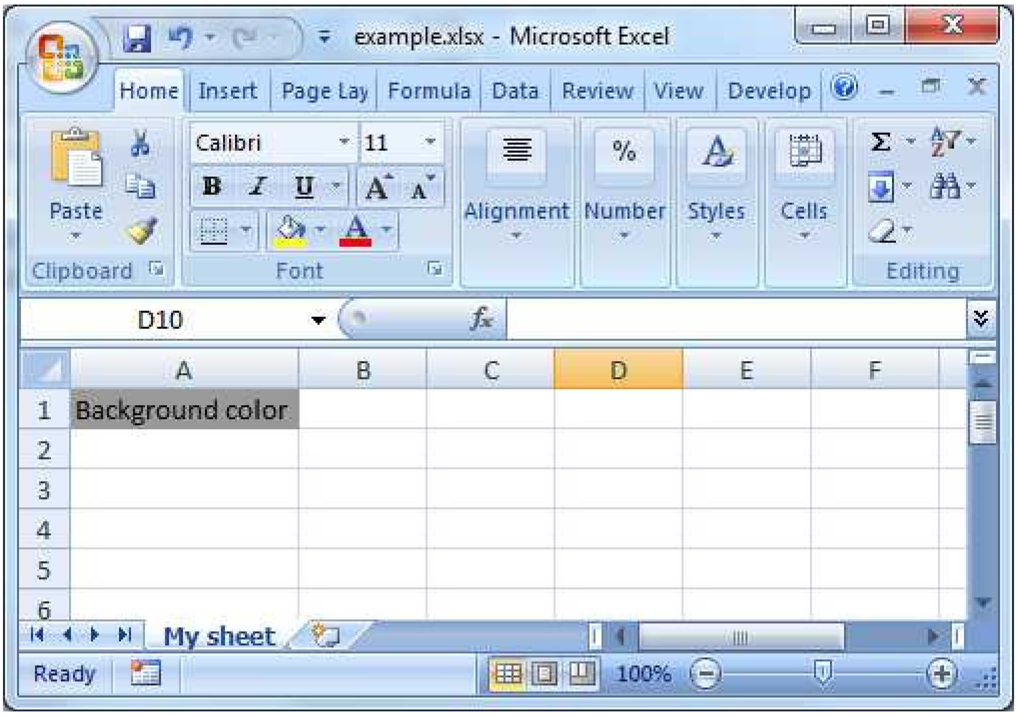
Procedure, does not return any value

Example:

```
BEGIN
     ORA_EXCEL.new_document;
     ORA_EXCEL.add_sheet('My sheet');
     ORA_EXCEL.add_row;
     ORA_EXCEL.set_cell_value('A', 'Background color');
     ORA_EXCEL.set_cell_bg_color('A', '999999');
     ORA_EXCEL.save_to_file('EXPORT_DIR', 'example.xlsx');
```

```
END;
```

Output:

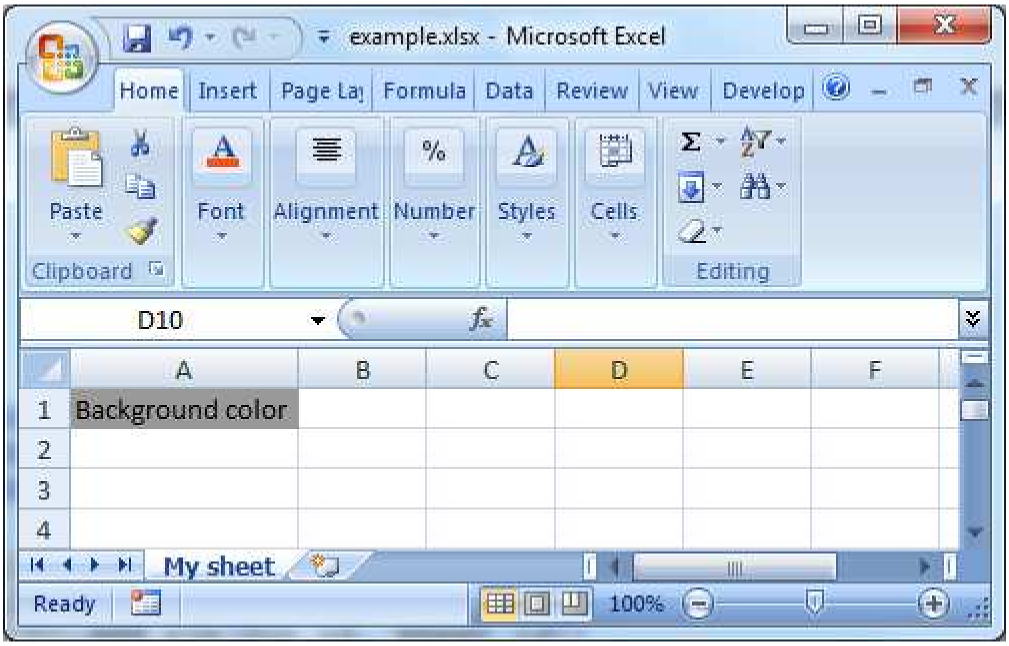


Background color

Example:

```
DECLARE
      doc_id PLS_INTEGER;
      sheet_id PLS_INTEGER;
      row_id PLS_INTEGER;
BEGIN
      doc_id := ORA_EXCEL.new_document;
      sheet_id := ORA_EXCEL.add_sheet('My sheet', doc_id);
      row_id := ORA_EXCEL.add_row(doc_id, sheet_id);
      ORA_EXCEL.set_cell_value('A', 'Background color', doc_id, sheet_id, row_id);
      ORA_EXCEL.set_cell_bg_color('A', '999999', doc_id, sheet_id, row_id);
      ORA_EXCEL.save_to_file('EXPORT_DIR', 'example.xlsx', doc_id);
END;
```

Output:



	A	B	C	D	E	F
1	Background color					
2						
3						
4						

Procedure ORA_EXCEL.set_cell_align_left

Description:

ORA_EXCEL.set_cell_align_left(name varchar2 [, doc_id pls_integer, sheet_id pls_integer, row_id pls_integer])

Align text left

Mandatory parameters:

- name – cell name

Optional parameters:

- doc_id – document id
- sheet_id – sheet id
- row_id – row id

Returns:

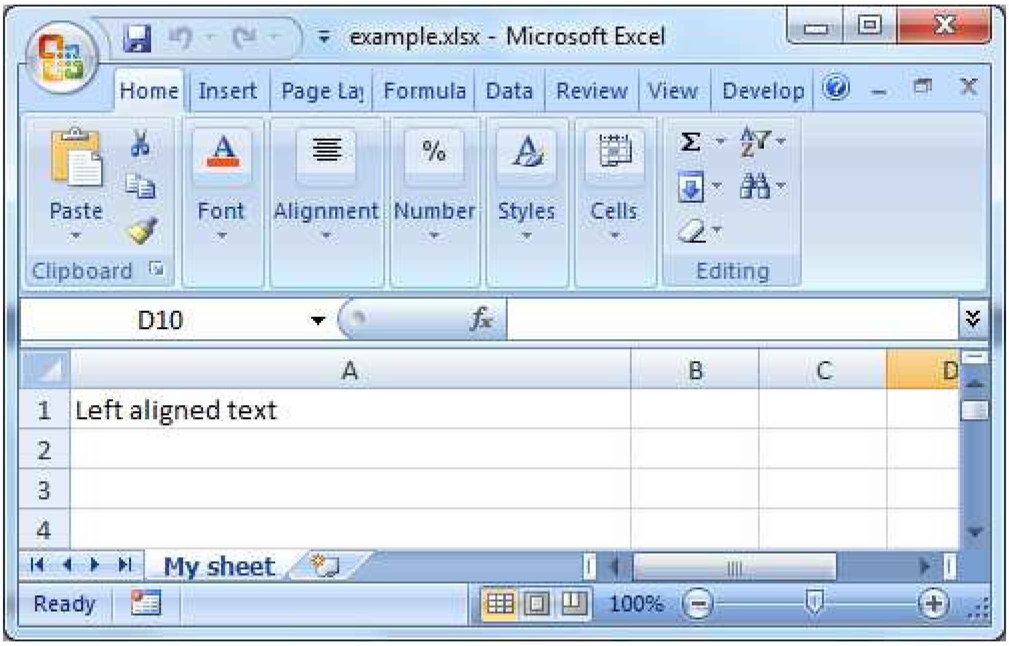
Procedure, does not return any value

Example:

```
BEGIN
      ORA_EXCEL.new_document;
      ORA_EXCEL.add_sheet('My sheet');
      ORA_EXCEL.add_row;
      ORA_EXCEL.set_cell_value('A', 'Left aligned text');
      ORA_EXCEL.set_column_width('A', 40);
      ORA_EXCEL.set_cell_align_left('A');
```

```
    ORA_EXCEL.save_to_file('EXPORT_DIR', 'example.xlsx');
END;
```

Output:

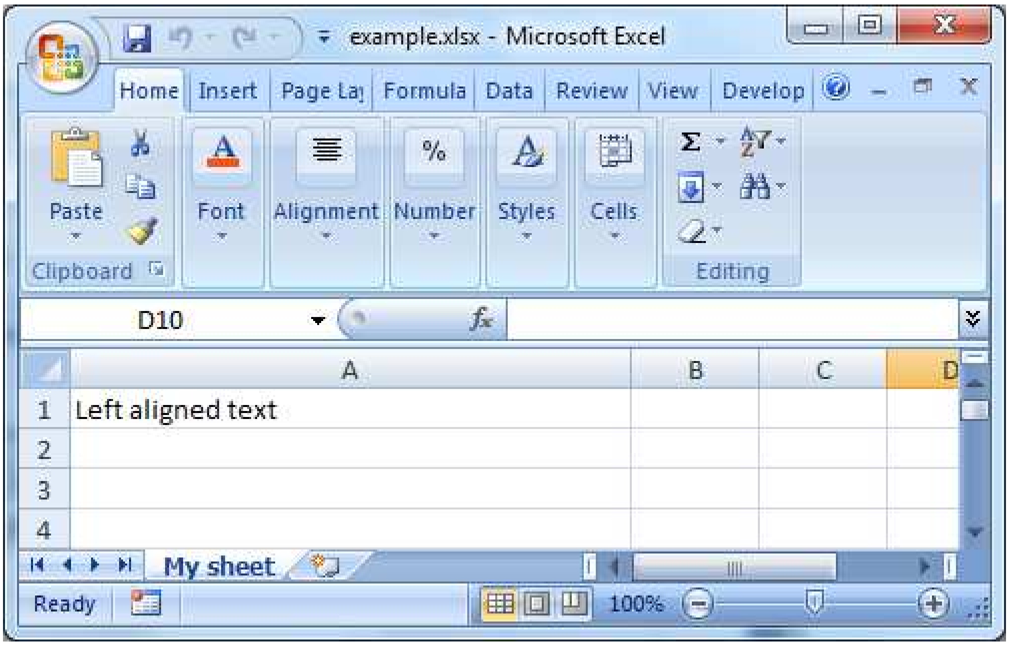


Left aligned text

Example:

```
DECLARE
    doc_id PLS_INTEGER;
    sheet_id PLS_INTEGER;
    row_id PLS_INTEGER;
BEGIN
    doc_id := ORA_EXCEL.new_document;
    sheet_id := ORA_EXCEL.add_sheet('My sheet', doc_id);
    row_id := ORA_EXCEL.add_row(doc_id, sheet_id);
    ORA_EXCEL.set_cell_value('A', 'Left aligned text', doc_id, sheet_id, row_id);
    ORA_EXCEL.set_column_width('A', 40);
    ORA_EXCEL.set_cell_align_left('A', doc_id, sheet_id, row_id);
    ORA_EXCEL.save_to_file('EXPORT_DIR', 'example.xlsx', doc_id);
END;
```

Output:



	A	B	C	D
1	Left aligned text			
2				
3				
4				

Procedure ORA_EXCEL.set_cell_align_right

Description:

ORA_EXCEL.set_cell_align_right(name varchar2 [, doc_id pls_integer, sheet_id pls_integer, row_id pls_integer])

Align text right

Mandatory parameters:

- name – cell name

Optional parameters:

- doc_id – document id
- sheet_id – sheet id
- row_id – row id

Returns:

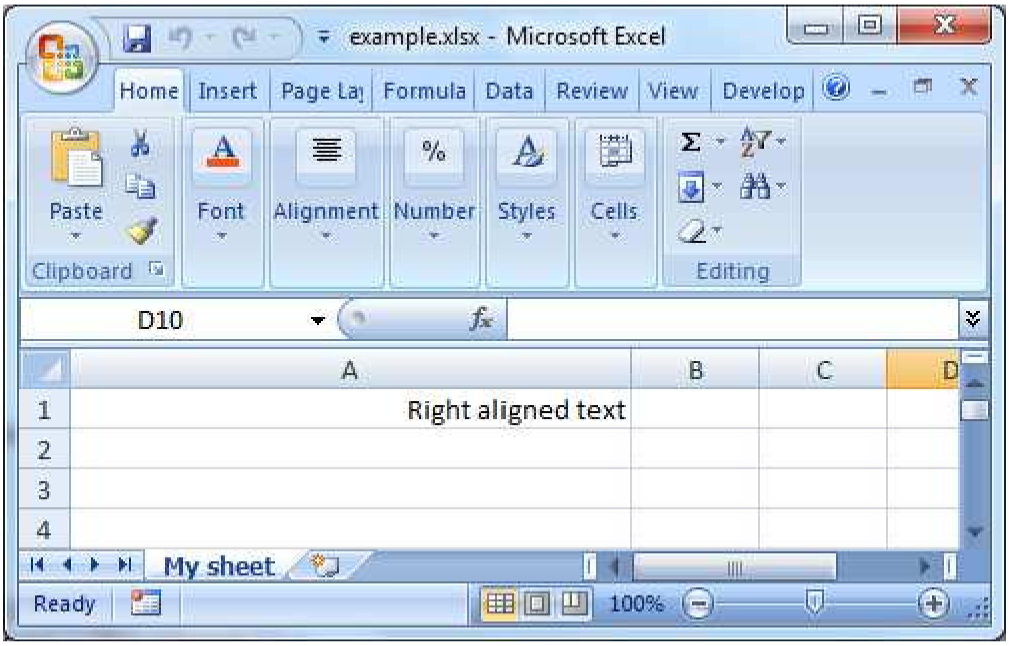
Procedure, does not return any value

Example:

```
BEGIN
     ORA_EXCEL.new_document;
     ORA_EXCEL.add_sheet('My sheet');
     ORA_EXCEL.add_row;
     ORA_EXCEL.set_cell_value('A', 'Right aligned text');
     ORA_EXCEL.set_column_width('A', 40);
     ORA_EXCEL.set_cell_align_right('A');
     ORA_EXCEL.save_to_file('EXPORT_DIR', 'example.xlsx');
```

```
END;
```

Output:

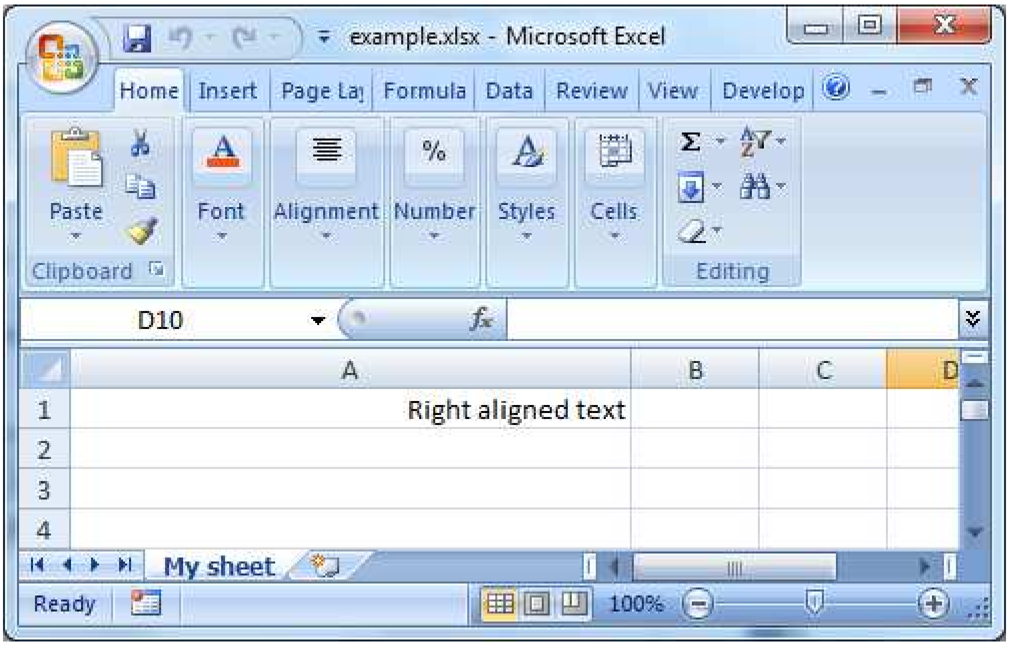


Right aligned text

Example:

```
DECLARE
     doc_id PLS_INTEGER;
     sheet_id PLS_INTEGER;
     row_id PLS_INTEGER;
BEGIN
     doc_id := ORA_EXCEL.new_document;
     sheet_id := ORA_EXCEL.add_sheet('My sheet', doc_id);
     row_id := ORA_EXCEL.add_row(doc_id, sheet_id);
     ORA_EXCEL.set_cell_value('A', 'Right aligned text', doc_id, sheet_id,
row_id);
     ORA_EXCEL.set_column_width('A', 40);
     ORA_EXCEL.set_cell_align_right('A', doc_id, sheet_id, row_id);
     ORA_EXCEL.save_to_file('EXPORT_DIR', 'example.xlsx', doc_id);
END;
```

Output:



Procedure ORA_EXCEL.set_cell_align_center

Description:

ORA_EXCEL.set_cell_align_center(name varchar2 [, doc_id pls_integer, sheet_id pls_integer, row_id pls_integer])

Center text

Mandatory parameters:

- name – cell name

Optional parameters:

- doc_id – document id
- sheet_id – sheet id
- row_id – row id

Returns:

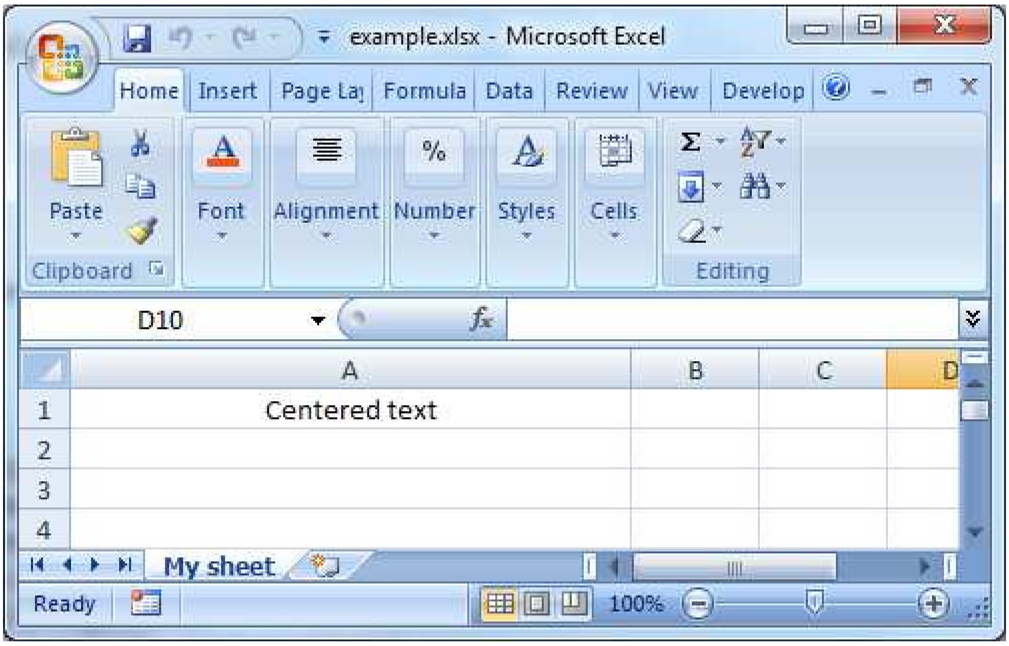
Procedure, does not return any value

Example:

```
BEGIN
     ORA_EXCEL.new_document;
     ORA_EXCEL.add_sheet('My sheet');
     ORA_EXCEL.add_row;
     ORA_EXCEL.set_cell_value('A', 'Centered text');
     ORA_EXCEL.set_column_width('A', 40);
     ORA_EXCEL.set_cell_align_center('A');
     ORA_EXCEL.save_to_file('EXPORT_DIR', 'example.xlsx');
```

```
END;
```

Output:

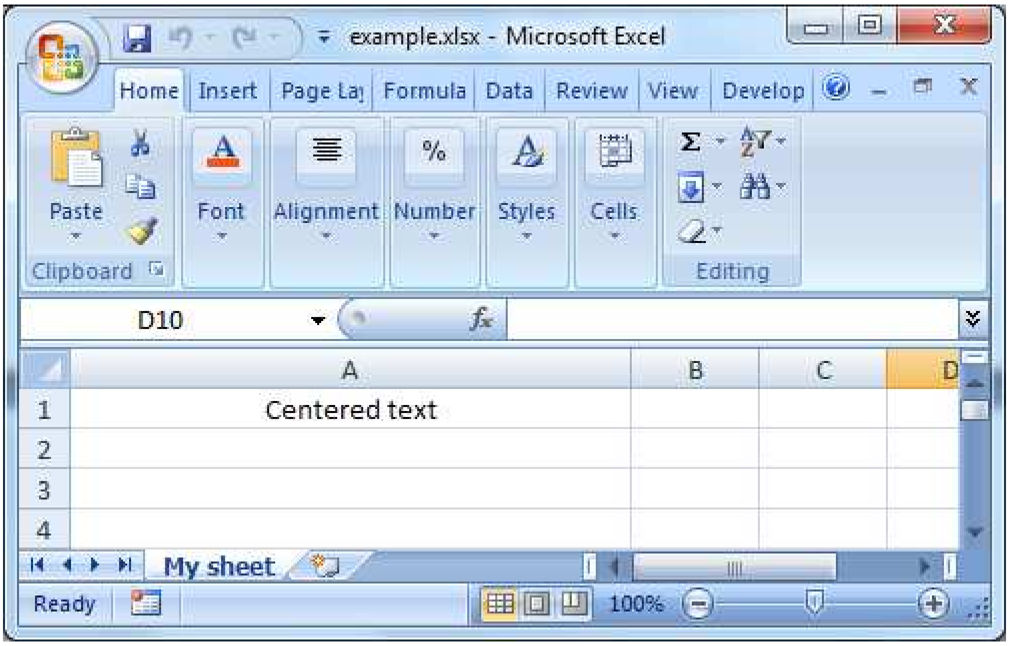


Centered text

Example:

```
DECLARE
     doc_id PLS_INTEGER;
     sheet_id PLS_INTEGER;
     row_id PLS_INTEGER;
BEGIN
     doc_id := ORA_EXCEL.new_document;
     sheet_id := ORA_EXCEL.add_sheet('My sheet', doc_id);
     row_id := ORA_EXCEL.add_row(doc_id, sheet_id);
     ORA_EXCEL.set_cell_value('A', 'Centered text', doc_id, sheet_id, row_id);
     ORA_EXCEL.set_column_width('A', 40);
     ORA_EXCEL.set_cell_align_center('A', doc_id, sheet_id, row_id);
     ORA_EXCEL.save_to_file('EXPORT_DIR', 'example.xlsx', doc_id);
END;
```

Output:



Centered text

Procedure ORA_EXCEL.set_cell_vert_align_top

Description:

ORA_EXCEL.set_cell_vert_align_top(name varchar2 [, doc_id pls_integer, sheet_id pls_integer, row_id pls_integer])

Align text to the top of the cell

Mandatory parameters:

- name – cell name

Optional parameters:

- doc_id – document id
- sheet_id – sheet id
- row_id – row id

Returns:

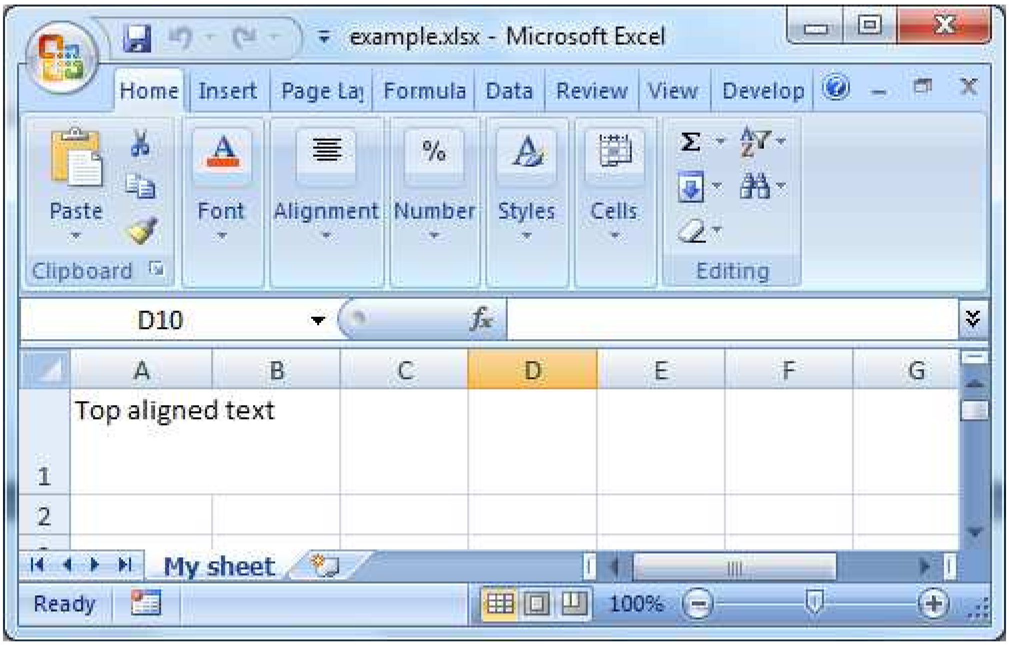
Procedure, does not return any value

Example:

```
BEGIN
     ORA_EXCEL.new_document;
     ORA_EXCEL.add_sheet('My sheet');
     ORA_EXCEL.add_row;
     ORA_EXCEL.set_row_height(40);
     ORA_EXCEL.set_cell_value('A', 'Top aligned text');
     ORA_EXCEL.set_cell_vert_align_top('A');
     ORA_EXCEL.save_to_file('EXPORT_DIR', 'example.xlsx');
```

```
END;
```

Output:

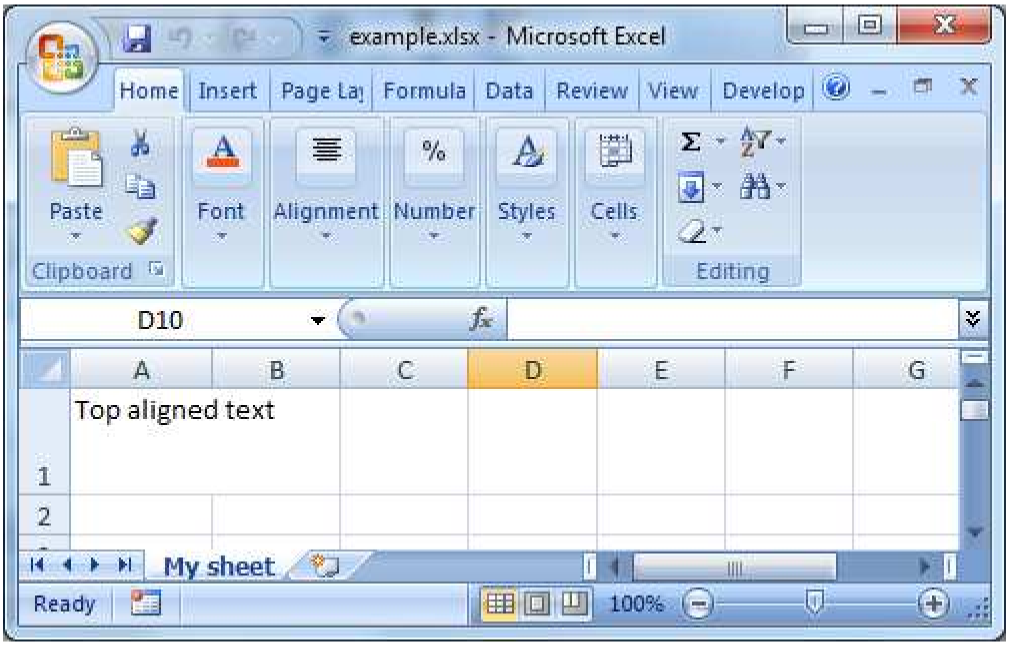


Top aligned text

Example:

```
DECLARE
     doc_id PLS_INTEGER;
     sheet_id PLS_INTEGER;
     row_id PLS_INTEGER;
BEGIN
     doc_id := ORA_EXCEL.new_document;
     sheet_id := ORA_EXCEL.add_sheet('My sheet', doc_id);
     row_id := ORA_EXCEL.add_row(doc_id, sheet_id);
     ORA_EXCEL.set_row_height(40, doc_id, sheet_id, row_id);
     ORA_EXCEL.set_cell_value('A', 'Top aligned text', doc_id, sheet_id, row_id);
     ORA_EXCEL.set_cell_vert_align_top('A', doc_id, sheet_id, row_id);
     ORA_EXCEL.save_to_file('EXPORT_DIR', 'example.xlsx', doc_id);
END;
```

Output:



Top aligned text

Procedure ORA_EXCEL.set_cell_vert_align_middle

Description:

ORA_EXCEL.set_cell_vert_align_middle(name varchar2 [, doc_id pls_integer, sheet_id pls_integer, row_id pls_integer])

Align text so that is centered between the top and bottom of the cell

Mandatory parameters:

- name – cell name

Optional parameters:

- doc_id – document id
- sheet_id – sheet id
- row_id – row id

Returns:

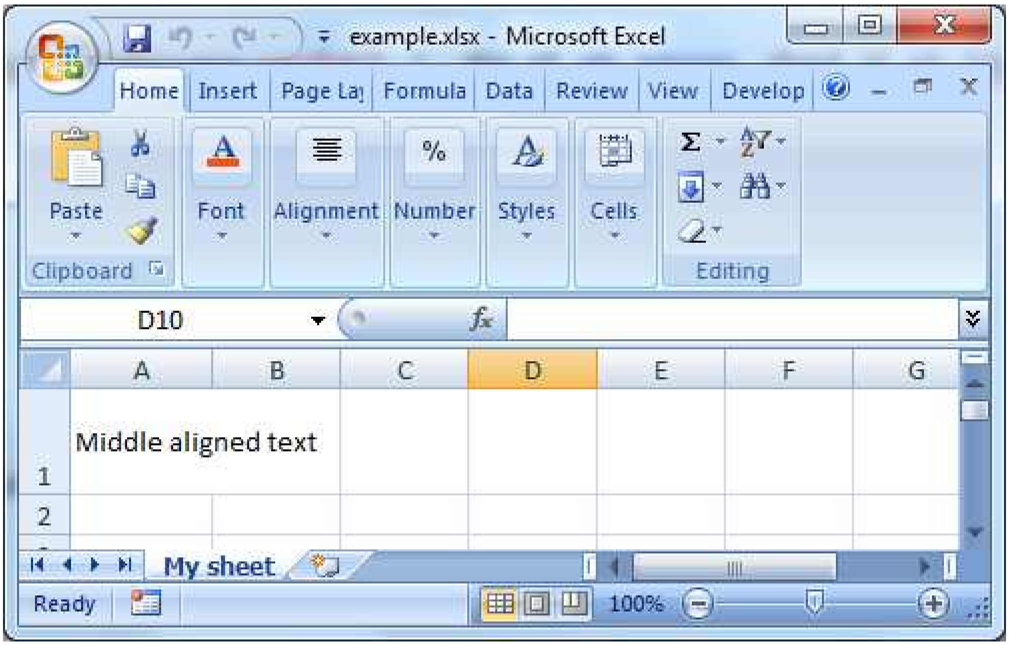
Procedure, does not return any value

Example:

```
BEGIN
     ORA_EXCEL.new_document;
     ORA_EXCEL.add_sheet('My sheet');
     ORA_EXCEL.add_row;
     ORA_EXCEL.set_row_height(40);
     ORA_EXCEL.set_cell_value('A', 'Middle aligned text');
     ORA_EXCEL.set_cell_vert_align_middle('A');
     ORA_EXCEL.save_to_file('EXPORT_DIR', 'example.xlsx');
```

```
END;
```

Output:

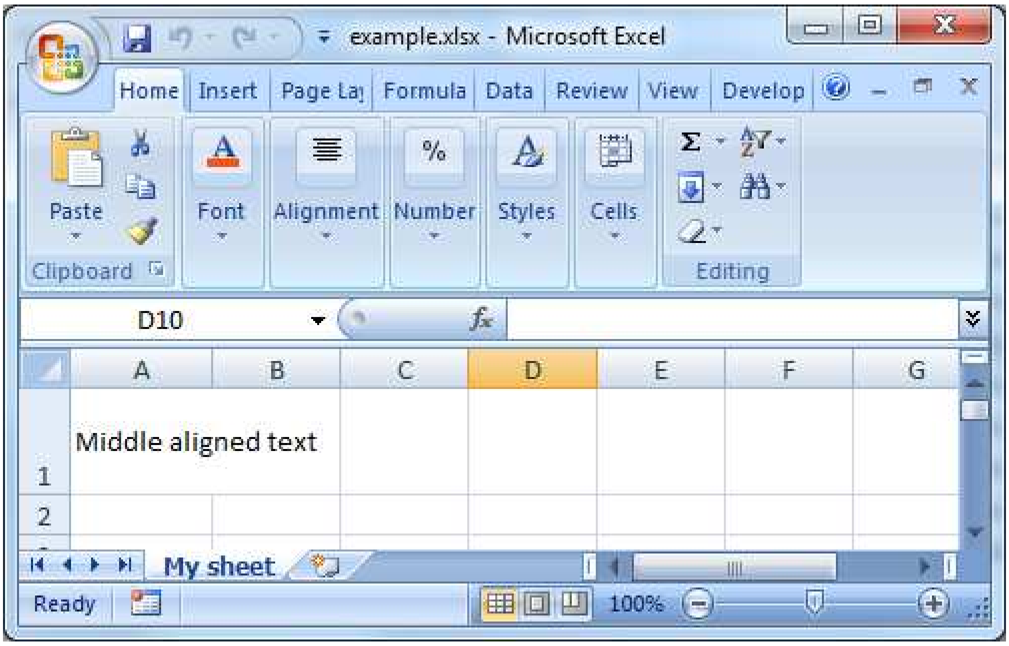


Middle aligned text

Example:

```
DECLARE
      doc_id PLS_INTEGER;
      sheet_id PLS_INTEGER;
      row_id PLS_INTEGER;
BEGIN
      doc_id := ORA_EXCEL.new_document;
      sheet_id := ORA_EXCEL.add_sheet('My sheet', doc_id);
      row_id := ORA_EXCEL.add_row(doc_id, sheet_id);
      ORA_EXCEL.set_row_height(40, doc_id, sheet_id, row_id);
      ORA_EXCEL.set_cell_value('A', 'Middle aligned text', doc_id, sheet_id,
row_id);
      ORA_EXCEL.set_cell_vert_align_middle('A', doc_id, sheet_id, row_id);
      ORA_EXCEL.save_to_file('EXPORT_DIR', 'example.xlsx', doc_id);
END;
```

Output:



Middle aligned text

Procedure ORA_EXCEL.set_cell_vert_align_bottom

Description:

ORA_EXCEL.set_cell_vert_align_bottom(name varchar2 [, doc_id pls_integer, sheet_id pls_integer, row_id pls_integer])

Align text to the bottom of the cell

Mandatory parameters:

- name – cell name

Optional parameters:

- doc_id – document id
- sheet_id – sheet id
- row_id – row id

Returns:

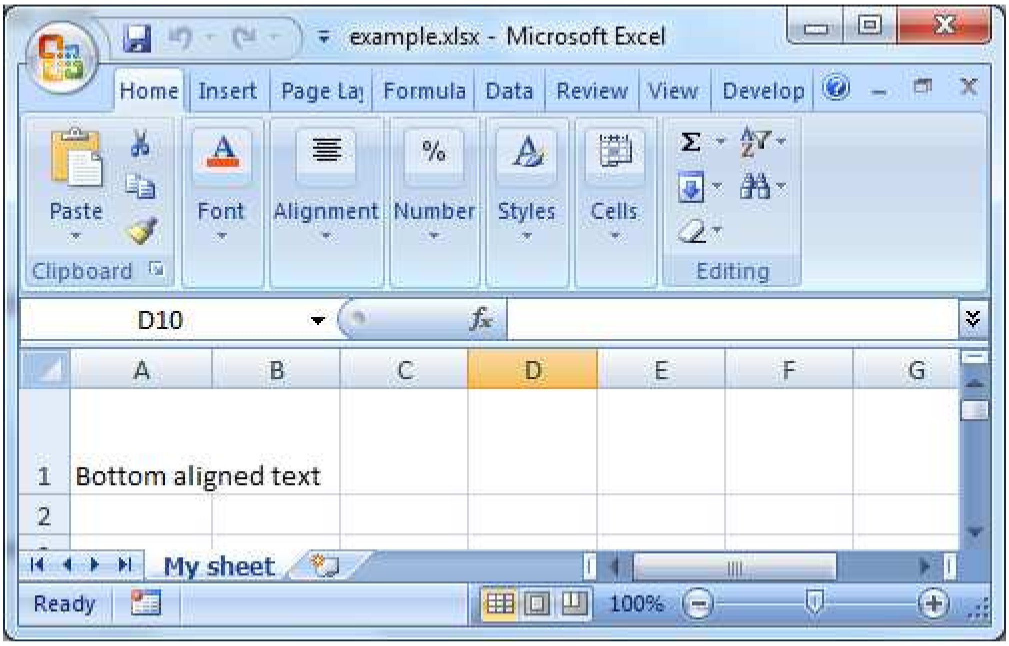
Procedure, does not return any value

Example:

```
BEGIN
    ORA_EXCEL.new_document;
    ORA_EXCEL.add_sheet('My sheet');
    ORA_EXCEL.add_row;
    ORA_EXCEL.set_row_height(40);
    ORA_EXCEL.set_cell_value('A', 'Bottom aligned text');
    ORA_EXCEL.set_cell_vert_align_bottom('A');
    ORA_EXCEL.save_to_file('EXPORT_DIR', 'example.xlsx');
```

```
END;
```

Output:

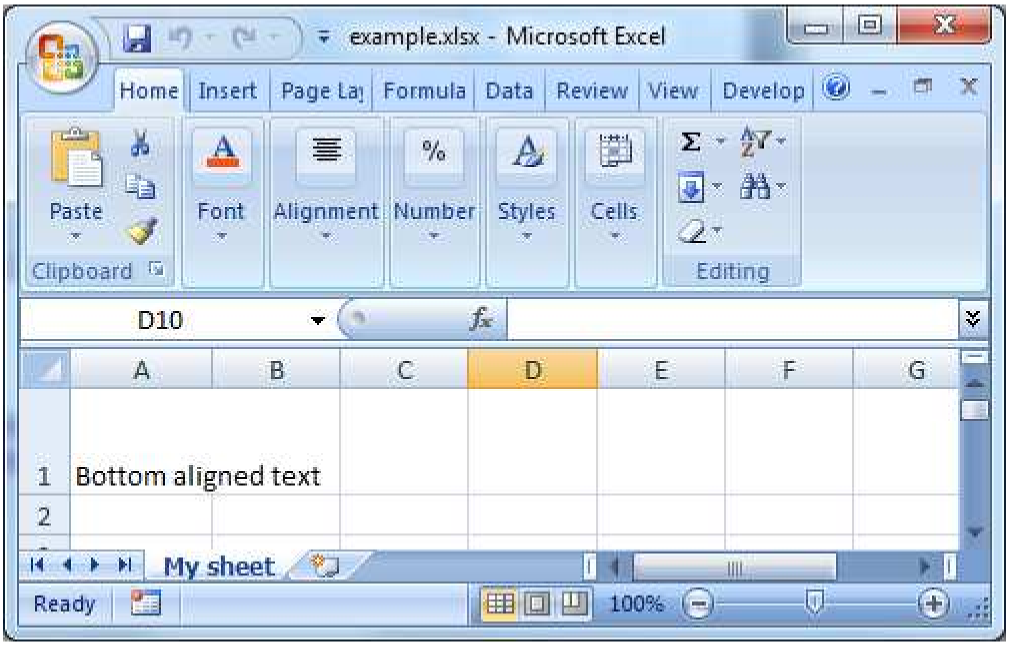


Bottom aligned text

Example:

```
DECLARE
     doc_id PLS_INTEGER;
     sheet_id PLS_INTEGER;
     row_id PLS_INTEGER;
BEGIN
     doc_id := ORA_EXCEL.new_document;
     sheet_id := ORA_EXCEL.add_sheet('My sheet', doc_id);
     row_id := ORA_EXCEL.add_row(doc_id, sheet_id);
     ORA_EXCEL.set_row_height(40, doc_id, sheet_id, row_id);
     ORA_EXCEL.set_cell_value('A', 'Bottom aligned text', doc_id, sheet_id,
row_id);
     ORA_EXCEL.set_cell_vert_align_bottom('A', doc_id, sheet_id, row_id);
     ORA_EXCEL.save_to_file('EXPORT_DIR', 'example.xlsx', doc_id);
END;
```

Output:



Bottom aligned text

Procedure ORA_EXCEL.set_column_width

Description:

ORA_EXCEL.set_column_width(name varchar2, width number [, doc_id pls_integer, sheet_id pls_integer, row_id pls_integer])

Set column width

Mandatory parameters:

- name – cell name
- width – column width

Optional parameters:

- doc_id – document id
- sheet_id – sheet id
- row_id – row id

Returns:

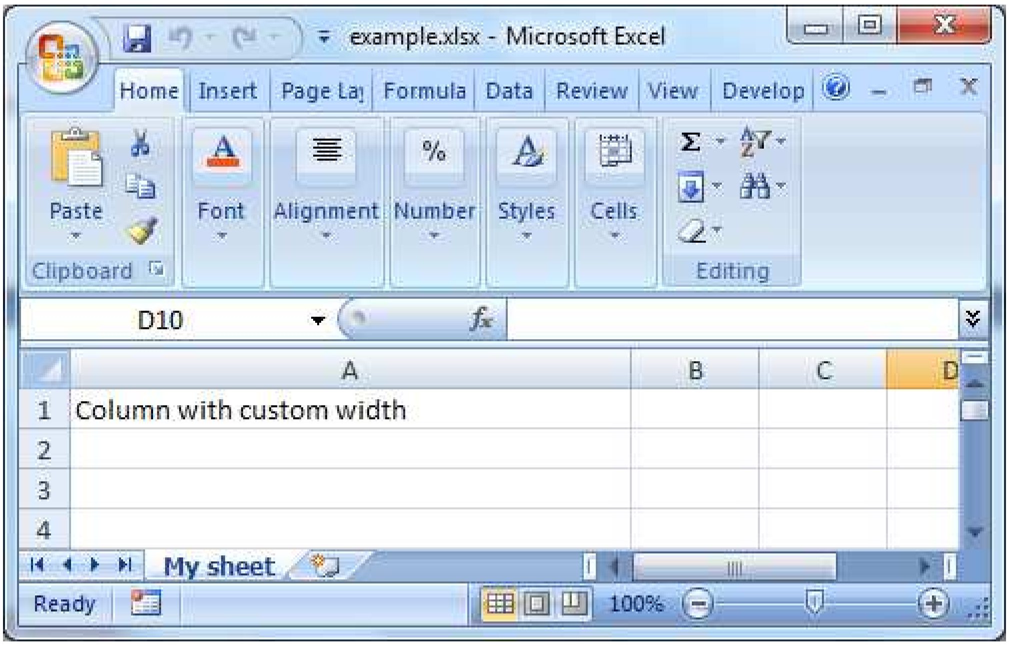
Procedure, does not return any value

Example:

```
BEGIN
      ORA_EXCEL.new_document;
      ORA_EXCEL.add_sheet('My sheet');
      ORA_EXCEL.add_row;
      ORA_EXCEL.set_cell_value('A', 'Column with custom width');
      ORA_EXCEL.set_column_width('A', 40);
```

```
    ORA_EXCEL.save_to_file('EXPORT_DIR', 'example.xlsx');
END;
```

Output:

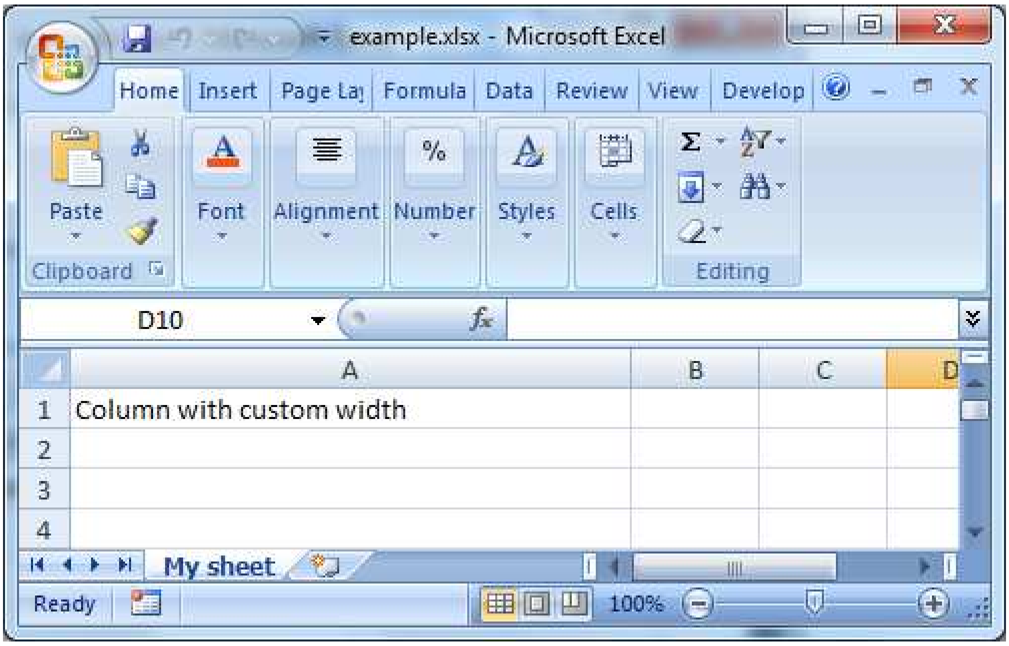


Column with custom width

Example:

```
DECLARE
    doc_id PLS_INTEGER;
    sheet_id PLS_INTEGER;
    row_id PLS_INTEGER;
BEGIN
    doc_id := ORA_EXCEL.new_document;
    sheet_id := ORA_EXCEL.add_sheet('My sheet', doc_id);
    row_id := ORA_EXCEL.add_row(doc_id, sheet_id);
    ORA_EXCEL.set_cell_value('A', 'Column with custom width', doc_id, sheet_id,
    row_id);
    ORA_EXCEL.set_column_width('A', 40, doc_id, sheet_id);
    ORA_EXCEL.save_to_file('EXPORT_DIR', 'example.xlsx', doc_id);
END;
```

Output:



	A	B	C	D
1	Column with custom width			
2				
3				
4				

Procedure ORA_EXCEL.set_cell_border_top

Description:

ORA_EXCEL.set_cell_border_top(name varchar2 [, style varchar2, color VARCHAR2, doc_id pls_integer, sheet_id pls_integer, row_id pls_integer])

Set top cell border style and color

Mandatory parameters:

- name – cell name

Optional parameters:

- style – border style (allowed values: thin, thick, double)
- color – RGB border color in hex format (for example FF0000 – red)
- doc_id – document id
- sheet_id – sheet id
- row_id – row id

Returns:

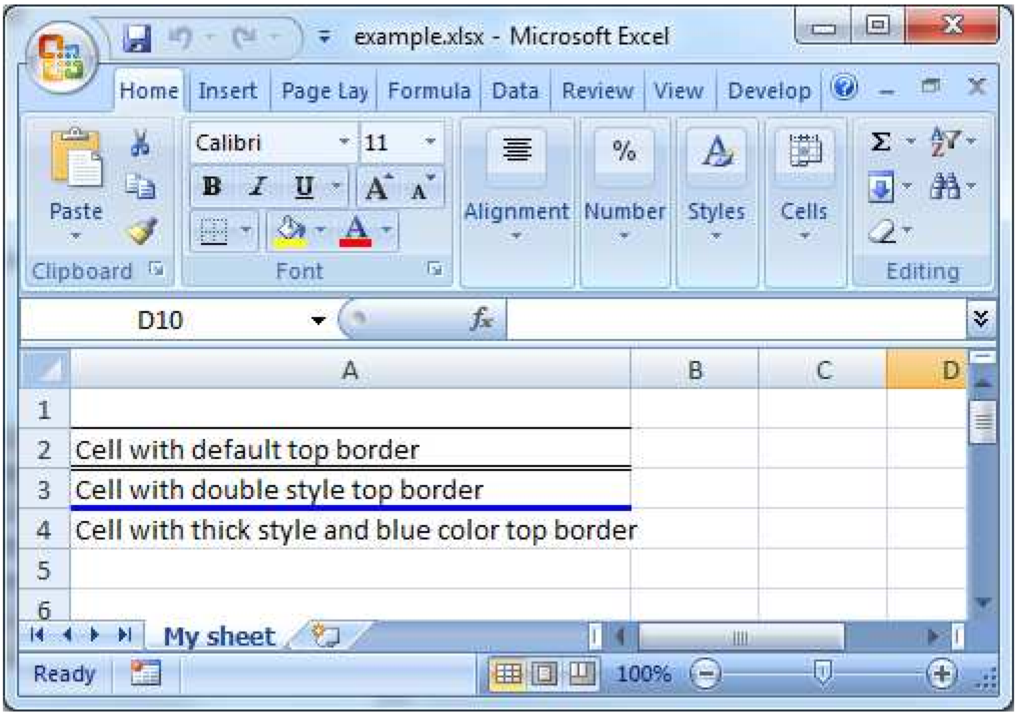
Procedure, does not return any value

Example:

```
BEGIN
     ORA_EXCEL.new_document;
     ORA_EXCEL.add_sheet('My sheet');
     ORA_EXCEL.set_column_width('A', 40);
     ORA_EXCEL.add_row;
```

```
    ORA_EXCEL.add_row;
    ORA_EXCEL.set_cell_value('A', 'Cell with default top border');
    ORA_EXCEL.set_cell_border_top('A');

    ORA_EXCEL.add_row;
    ORA_EXCEL.set_cell_value('A', 'Cell with double style top border');
    ORA_EXCEL.set_cell_border_top('A', 'double');

    ORA_EXCEL.add_row;
    ORA_EXCEL.set_cell_value('A', 'Cell with thick style and blue top border');
    ORA_EXCEL.set_cell_border_top('A', 'thick', '0000FF');
    ORA_EXCEL.save_to_file('EXPORT_DIR', 'example.xlsx');
END;
```

Output:

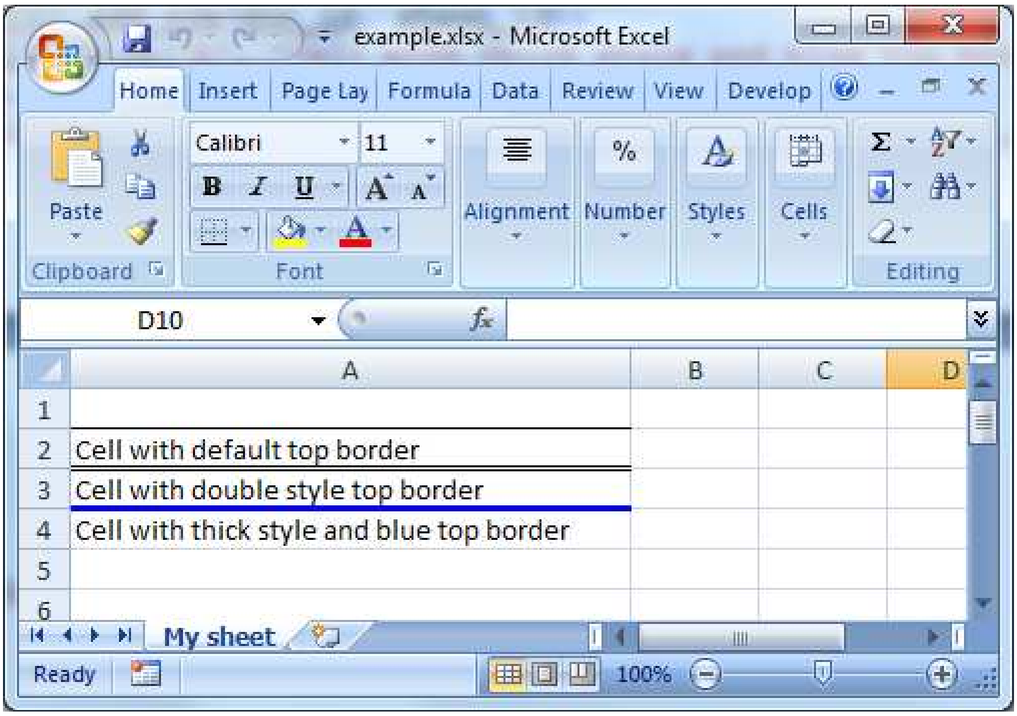


Cell with default top border
Cell with double style top border
Cell with thick style and blue color top border

Example:

```
DECLARE
    doc_id PLS_INTEGER;
    sheet_id PLS_INTEGER;
    row_id PLS_INTEGER;
BEGIN
    doc_id := ORA_EXCEL.new_document;
    sheet_id := ORA_EXCEL.add_sheet('My sheet', doc_id);
    ORA_EXCEL.set_column_width('A', 40, doc_id, sheet_id);

    row_id := ORA_EXCEL.add_row(doc_id, sheet_id);
    row_id := ORA_EXCEL.add_row(doc_id, sheet_id);
    ORA_EXCEL.set_cell_value('A', 'Cell with default top border',
    doc_id, sheet_id, row_id);
    ORA_EXCEL.set_cell_border_top('A', NULL, NULL, doc_id, sheet_id, row_id);

    row_id := ORA_EXCEL.add_row(doc_id, sheet_id);
    ORA_EXCEL.set_cell_value('A', 'Cell with double style top border', doc_id,
    sheet_id, row_id);
    ORA_EXCEL.set_cell_border_top('A', 'double', NULL, doc_id, sheet_id,
    row_id);

    row_id := ORA_EXCEL.add_row(doc_id, sheet_id);
    ORA_EXCEL.set_cell_value('A', 'Cell with thick style and blue top border',
    doc_id, sheet_id, row_id);
    ORA_EXCEL.set_cell_border_top('A', 'thick', '0000FF', doc_id, sheet_id,
    row_id);
    ORA_EXCEL.save_to_file('EXPORT_DIR', 'example.xlsx', doc_id);
END;
```

Output:



example.xlsx - Microsoft Excel

D10

	A	B	C	D
1				
2	Cell with default top border			
3	Cell with double style top border			
4	Cell with thick style and blue top border			
5				
6				

Procedure ORA_EXCEL.set_cell_border_bottom

Description:

ORA_EXCEL.set_cell_border_bottom(name varchar2 [, style varchar2, color VARCHAR2, doc_id pls_integer, sheet_id pls_integer, row_id pls_integer])

Set bottom cell border style and color

Mandatory parameters:

- name – cell name

Optional parameters:

- style – border style (allowed values: thin, thick, double)
- color – RGB border color in hex format (for example FF0000 – red)
- doc_id – document id
- sheet_id – sheet id
- row_id – row id

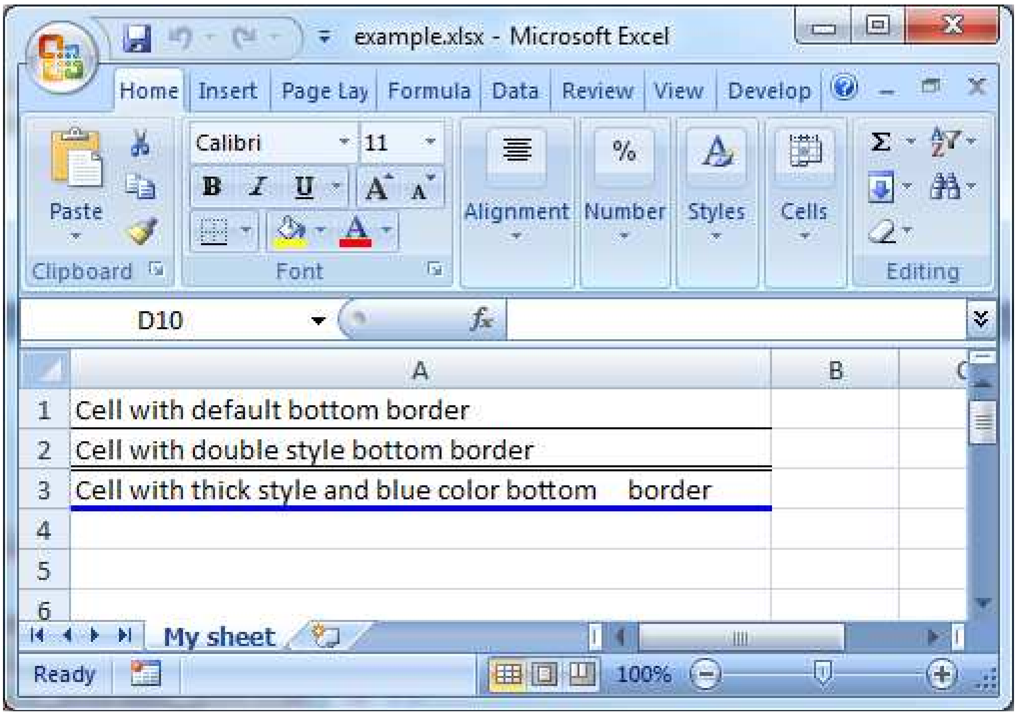
Returns:

Procedure, does not return any value

Example:

```
BEGIN
	ORA_EXCEL.new_document;
	ORA_EXCEL.add_sheet('My sheet');
	ORA_EXCEL.set_column_width('A', 40);

	ORA_EXCEL.add_row;
	ORA_EXCEL.set_cell_value('A', 'Cell with default bottom border');
	ORA_EXCEL.set_cell_border_bottom('A');

	ORA_EXCEL.add_row;
	ORA_EXCEL.set_cell_value('A', 'Cell with double style bottom border');
	ORA_EXCEL.set_cell_border_bottom('A', 'double');

	ORA_EXCEL.add_row;
	ORA_EXCEL.set_cell_value('A', 'Cell with thick style and blue color bottom 
	border');
	ORA_EXCEL.set_cell_border_bottom('A', 'thick', '0000FF');
	ORA_EXCEL.save_to_file('EXPORT_DIR', 'example.xlsx');
END;
```

Output:



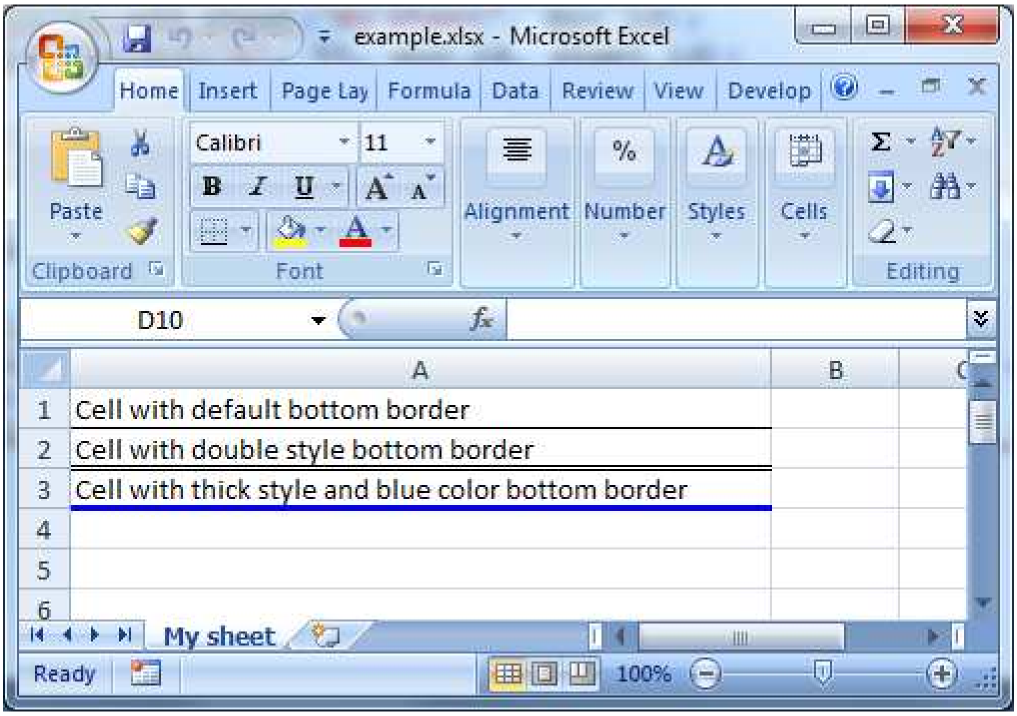
	A
1	Cell with default bottom border
2	Cell with double style bottom border
3	Cell with thick style and blue color bottom border
4	
5	
6	

Example:

```
DECLARE
	doc_id PLS_INTEGER;
	sheet_id PLS_INTEGER;
	row_id PLS_INTEGER;
BEGIN
	doc_id := ORA_EXCEL.new_document;
	sheet_id := ORA_EXCEL.add_sheet('My sheet', doc_id);
```

```
    ORA_EXCEL.set_column_width('A', 50, doc_id, sheet_id);

    row_id := ORA_EXCEL.add_row(doc_id, sheet_id);
    ORA_EXCEL.set_cell_value('A', 'Cell with default bottom border', doc_id,
    sheet_id, row_id);
    ORA_EXCEL.set_cell_border_bottom('A', NULL, NULL, doc_id, sheet_id, row_id);

    row_id := ORA_EXCEL.add_row(doc_id, sheet_id);
    ORA_EXCEL.set_cell_value('A', 'Cell with double style bottom border', doc_id,
    sheet_id, row_id);
    ORA_EXCEL.set_cell_border_bottom('A', 'double', NULL, doc_id, sheet_id,
    row_id);

    row_id := ORA_EXCEL.add_row(doc_id, sheet_id);
    ORA_EXCEL.set_cell_value('A', 'Cell with thick style and blue color bottom
    border', doc_id, sheet_id, row_id);
    ORA_EXCEL.set_cell_border_bottom('A', 'thick', '0000FF', doc_id, sheet_id,
    row_id);
    ORA_EXCEL.save_to_file('EXPORT_DIR', 'example.xlsx', doc_id);
END;
```

Output:



	A
1	Cell with default bottom border
2	Cell with double style bottom border
3	Cell with thick style and blue color bottom border

Procedure ORA_EXCEL.set_cell_border_left

Description:

ORA_EXCEL.set_cell_border_left(name varchar2 [, style varchar2, color VARCHAR2, doc_id pls_integer, sheet_id pls_integer, row_id pls_integer])

Set left cell border style and color

Mandatory parameters:

- name – cell name

Optional parameters:

- style – border style (allowed values: thin, thick, double)
- color – RGB border color in hex format (for example FF0000 – red)
- doc_id – document id
- sheet_id – sheet id
- row_id – row id

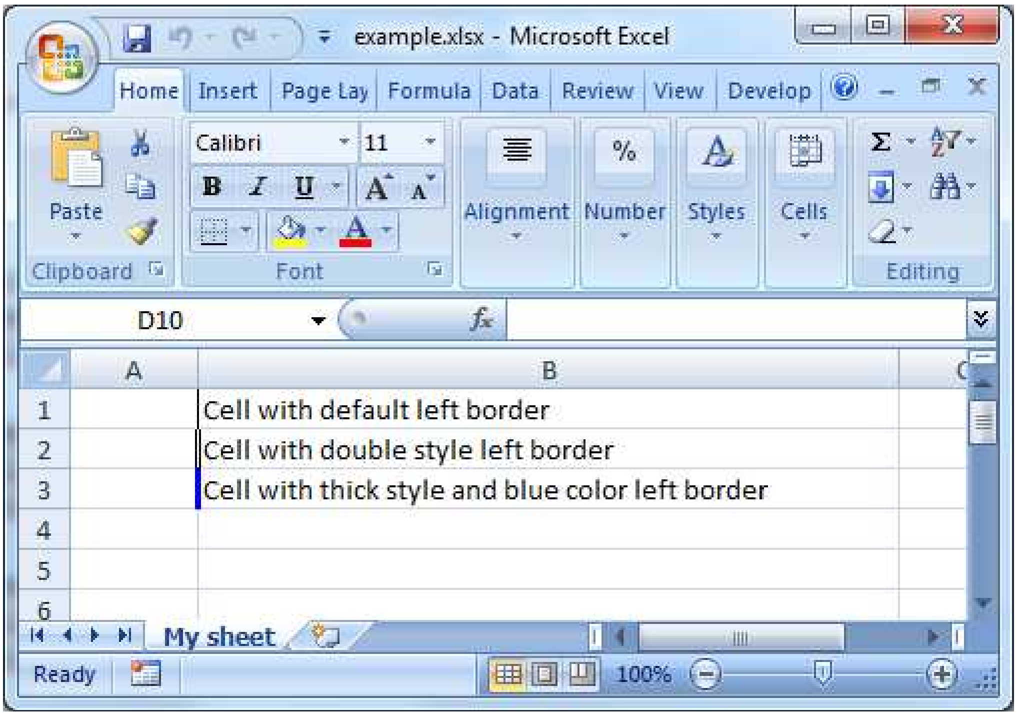
Returns:

Procedure, does not return any value

Example:

```
BEGIN
     ORA_EXCEL.new_document;
     ORA_EXCEL.add_sheet('My sheet');
     ORA_EXCEL.set_column_width('B', 50);

     ORA_EXCEL.add_row;
     ORA_EXCEL.set_cell_value('B', 'Cell with default left border');
     ORA_EXCEL.set_cell_border_left('B');

     ORA_EXCEL.add_row;
     ORA_EXCEL.set_cell_value('B', 'Cell with double style left border');
     ORA_EXCEL.set_cell_border_left('B', 'double');

     ORA_EXCEL.add_row;
     ORA_EXCEL.set_cell_value('B', 'Cell with thick style and blue color left
border');
     ORA_EXCEL.set_cell_border_left('B', 'thick', '0000FF');
     ORA_EXCEL.save_to_file('EXPORT_DIR', 'example.xlsx');
END;
```

Output:

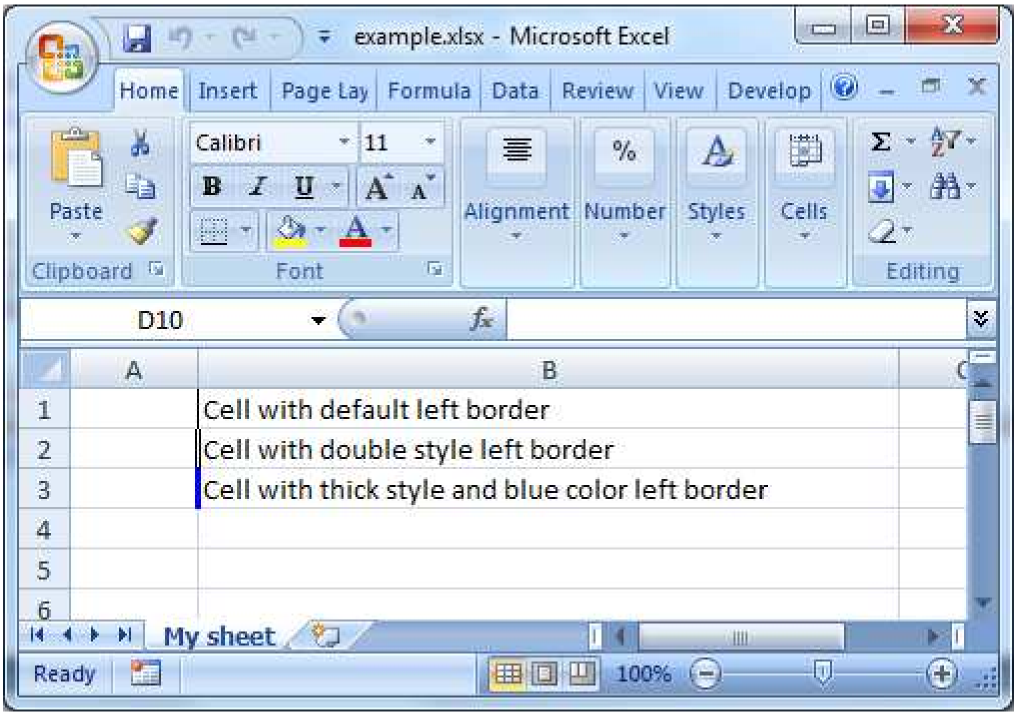


Cell with default left border
Cell with double style left border
Cell with thick style and blue color left border

Example:

```
DECLARE
     doc_id PLS_INTEGER;
     sheet_id PLS_INTEGER;
     row_id PLS_INTEGER;
BEGIN
     doc_id := ORA_EXCEL.new_document;
     sheet_id := ORA_EXCEL.add_sheet('My sheet', doc_id);
     ORA_EXCEL.set_column_width('B', 50, doc_id, sheet_id);

     row_id := ORA_EXCEL.add_row(doc_id, sheet_id);
     ORA_EXCEL.set_cell_value('B', 'Cell with default left border', doc_id,
     sheet_id, row_id);
     ORA_EXCEL.set_cell_border_left('B', NULL, NULL, doc_id, sheet_id, row_id);

     row_id := ORA_EXCEL.add_row(doc_id, sheet_id);
     ORA_EXCEL.set_cell_value('B', 'Cell with double style left border', doc_id,
     sheet_id, row_id);
     ORA_EXCEL.set_cell_border_left('B', 'double', NULL, doc_id, sheet_id,
     row_id);

     row_id := ORA_EXCEL.add_row(doc_id, sheet_id);
     ORA_EXCEL.set_cell_value('B', 'Cell with thick style and blue color left
     border', doc_id, sheet_id, row_id);
     ORA_EXCEL.set_cell_border_left('B', 'thick', '0000FF', doc_id, sheet_id,
     row_id);
     ORA_EXCEL.save_to_file('EXPORT_DIR', 'example.xlsx', doc_id);
END;
```

Output:



	A	B
1		Cell with default left border
2		Cell with double style left border
3		Cell with thick style and blue color left border
4		
5		
6		

Procedure ORA_EXCEL.set_cell_border_right

Description:

ORA_EXCEL.set_cell_border_right(name varchar2 [, style varchar2, color VARCHAR2, doc_id pls_integer, sheet_id pls_integer, row_id pls_integer])

Set right cell border style and color

Mandatory parameters:

- name – cell name

Optional parameters:

- style – border style (allowed values: thin, thick, double)
- color – RGB border color in hex format (for example FF0000 – red)
- doc_id – document id
- sheet_id – sheet id
- row_id – row id

Returns:

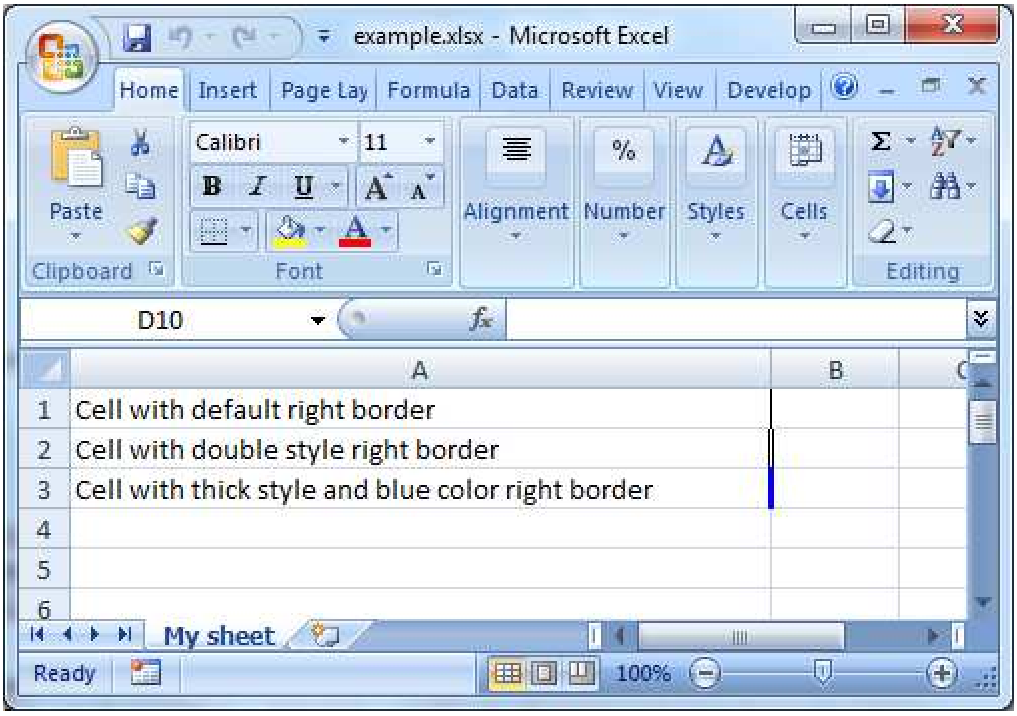
Procedure, does not return any value

Example:

```
BEGIN
     ORA_EXCEL.new_document;
     ORA_EXCEL.add_sheet('My sheet');
     ORA_EXCEL.set_column_width('A', 50);
```

```
    ORA_EXCEL.add_row;
    ORA_EXCEL.set_cell_value('A', 'Cell with default right border');
    ORA_EXCEL.set_cell_border_right('A');

    ORA_EXCEL.add_row;
    ORA_EXCEL.set_cell_value('A', 'Cell with double style right border');
    ORA_EXCEL.set_cell_border_right('A', 'double');

    ORA_EXCEL.add_row;
    ORA_EXCEL.set_cell_value('A', 'Cell with thick style and blue color right border');
    ORA_EXCEL.set_cell_border_right('A', 'thick', '0000FF');
    ORA_EXCEL.save_to_file('EXPORT_DIR', 'example.xlsx');
END;
```

Output:

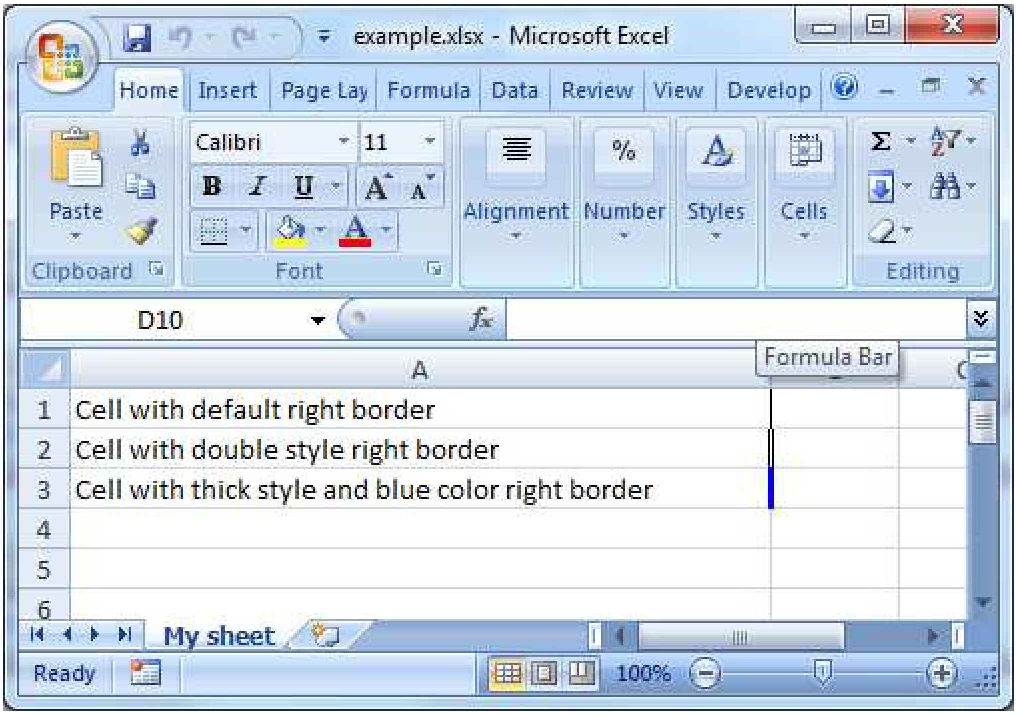


Cell with default right border
Cell with double style right border
Cell with thick style and blue color right border

Example:

```
DECLARE
     doc_id PLS_INTEGER;
     sheet_id PLS_INTEGER;
     row_id PLS_INTEGER;
BEGIN
     doc_id := ORA_EXCEL.new_document;
     sheet_id := ORA_EXCEL.add_sheet('My sheet', doc_id);
     ORA_EXCEL.set_column_width('A', 50, doc_id, sheet_id);

     row_id := ORA_EXCEL.add_row(doc_id, sheet_id);
     ORA_EXCEL.set_cell_value('A', 'Cell with default right border', doc_id,
     sheet_id, row_id);
     ORA_EXCEL.set_cell_border_right('A', NULL, NULL, doc_id, sheet_id, row_id);

     row_id := ORA_EXCEL.add_row(doc_id, sheet_id);
     ORA_EXCEL.set_cell_value('A', 'Cell with double style right border', doc_id,
     sheet_id, row_id);
     ORA_EXCEL.set_cell_border_right('A', 'double', NULL, doc_id, sheet_id,
     row_id);

     row_id := ORA_EXCEL.add_row(doc_id, sheet_id);
     ORA_EXCEL.set_cell_value('A', 'Cell with thick style and blue color right
     border', doc_id, sheet_id, row_id);
     ORA_EXCEL.set_cell_border_right('A', 'thick', '0000FF', doc_id, sheet_id,
     row_id);
     ORA_EXCEL.save_to_file('EXPORT_DIR', 'example.xlsx', doc_id);
END;
```

Output:



	A
1	Cell with default right border
2	Cell with double style right border
3	Cell with thick style and blue color right border
4	
5	
6	

Procedure ORA_EXCEL.set_cell_border

Description:

ORA_EXCEL.set_cell_border(name varchar2 [, style varchar2, color varchar2, doc_id pls_integer, sheet_id pls_integer, row_id pls_integer])

Set all cell borders (top, bottom, left and right) style and color

Mandatory parameters:

- name – cell name

Optional parameters:

- style – border style (allowed values: thin, thick, double)
- color – RGB border color in hex format (for example FF0000 – red)
- doc_id – document id
- sheet_id – sheet id
- row_id – row id

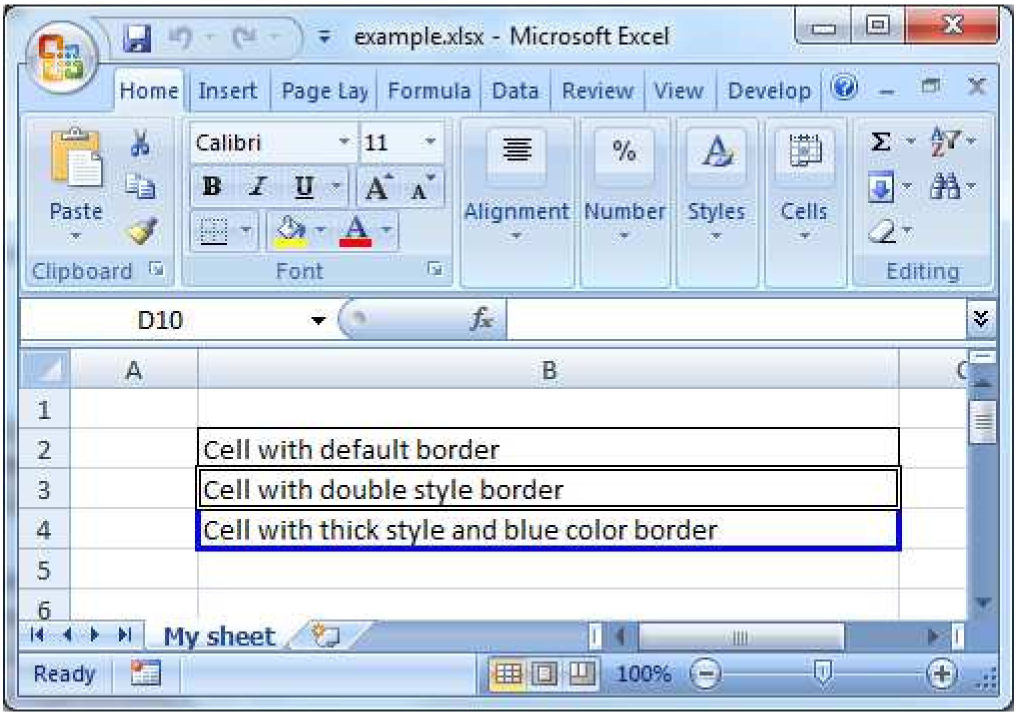
Returns:

Procedure, does not return any value

Example:

```
BEGIN
     ORA_EXCEL.new_document;
     ORA_EXCEL.add_sheet('My sheet');
     ORA_EXCEL.set_column_width('B', 50);

     ORA_EXCEL.add_row;
     ORA_EXCEL.add_row;
     ORA_EXCEL.set_cell_value('B', 'Cell with default border');
     ORA_EXCEL.set_cell_border('B');

     ORA_EXCEL.add_row;
     ORA_EXCEL.set_cell_value('B', 'Cell with double style border');
     ORA_EXCEL.set_cell_border('B', 'double');

     ORA_EXCEL.add_row;
     ORA_EXCEL.set_cell_value('B', 'Cell with thick style and blue color border');
     ORA_EXCEL.set_cell_border('B', 'thick', '0000FF');
     ORA_EXCEL.save_to_file('EXPORT_DIR', 'example.xlsx');
END;
```

Output:



Cell with default border

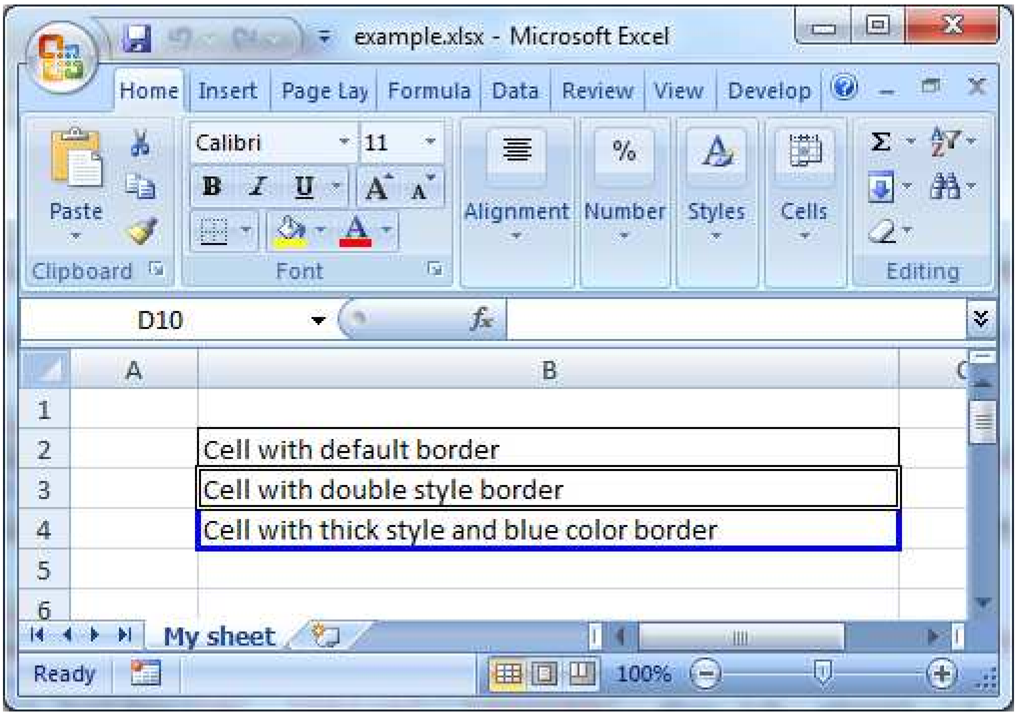
Cell with double style border

Cell with thick style and blue color border

Example:

```
DECLARE
     doc_id PLS_INTEGER;
     sheet_id PLS_INTEGER;
     row_id PLS_INTEGER;
BEGIN
     doc_id := ORA_EXCEL.new_document;
     sheet_id := ORA_EXCEL.add_sheet('My sheet', doc_id);
     ORA_EXCEL.set_column_width('B', 50, doc_id, sheet_id);

     row_id := ORA_EXCEL.add_row(doc_id, sheet_id);
     row_id := ORA_EXCEL.add_row(doc_id, sheet_id);
     ORA_EXCEL.set_cell_value('B', 'Cell with default border', doc_id, sheet_id,
     row_id);
     ORA_EXCEL.set_cell_border('B', NULL, NULL, doc_id, sheet_id, row_id);

     row_id := ORA_EXCEL.add_row(doc_id, sheet_id);
     ORA_EXCEL.set_cell_value('B', 'Cell with double style border', doc_id,
     sheet_id, row_id);
     ORA_EXCEL.set_cell_border('B', 'double', NULL, doc_id, sheet_id, row_id);

     row_id := ORA_EXCEL.add_row(doc_id, sheet_id);
     ORA_EXCEL.set_cell_value('B', 'Cell with thick style and blue color border',
     doc_id, sheet_id, row_id);
     ORA_EXCEL.set_cell_border('B', 'thick', '0000FF', doc_id, sheet_id, row_id);
     ORA_EXCEL.save_to_file('EXPORT_DIR', 'example.xlsx', doc_id);
END;
```

Output:



Cell with default border

Cell with double style border

Cell with thick style and blue color border

Procedure ORA_EXCEL.set_cell_wrap_text

Description:

ORA_EXCEL.set_cell_wrap_text (name varchar2, [, doc_id pls_integer, sheet_id pls_integer, row_id pls_integer])

Make all content visible within a cell by displaying it on multiple lines

Mandatory parameters:

- name – cell name

Optional parameters:

- doc_id – document id
- sheet_id – sheet id
- row_id – row id

Returns:

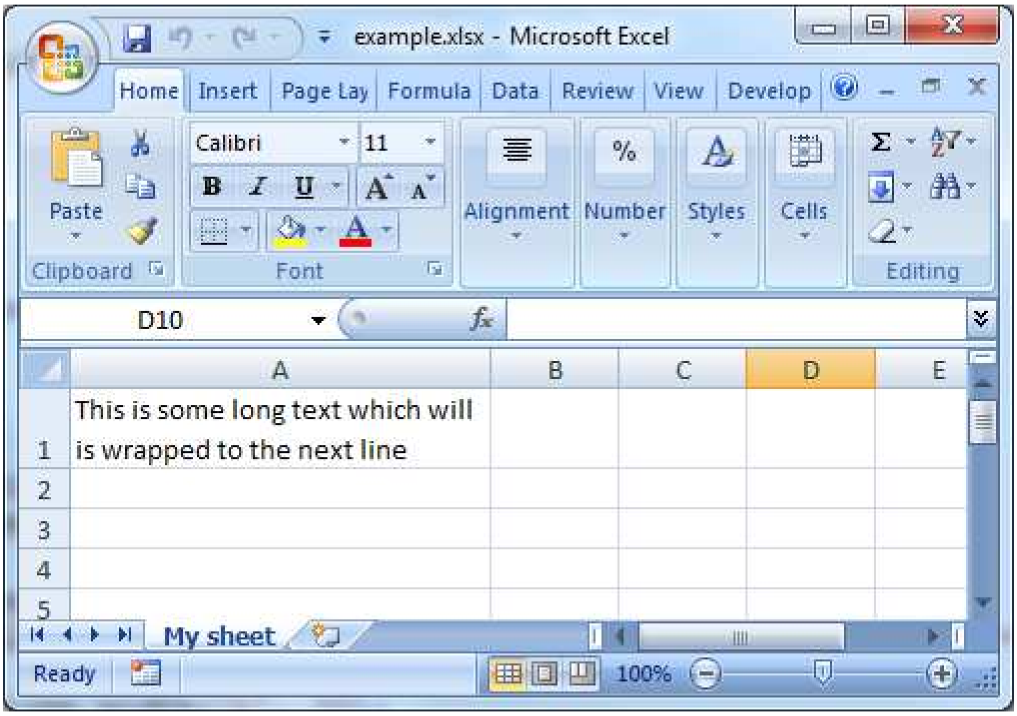
Procedure, does not return any value

Example:

```
BEGIN
      ORA_EXCEL.new_document;
      ORA_EXCEL.add_sheet('My sheet');

      ORA_EXCEL.add_row;
      ORA_EXCEL.set_cell_value('A', 'This is some long text which will is wrapped
      to the next line');
```

```
    ORA_EXCEL.set_column_width('A', 30);
    ORA_EXCEL.set_cell_wrap_text('A');
    ORA_EXCEL.save_to_file('EXPORT_DIR', 'example.xlsx');
END;
```

Output:

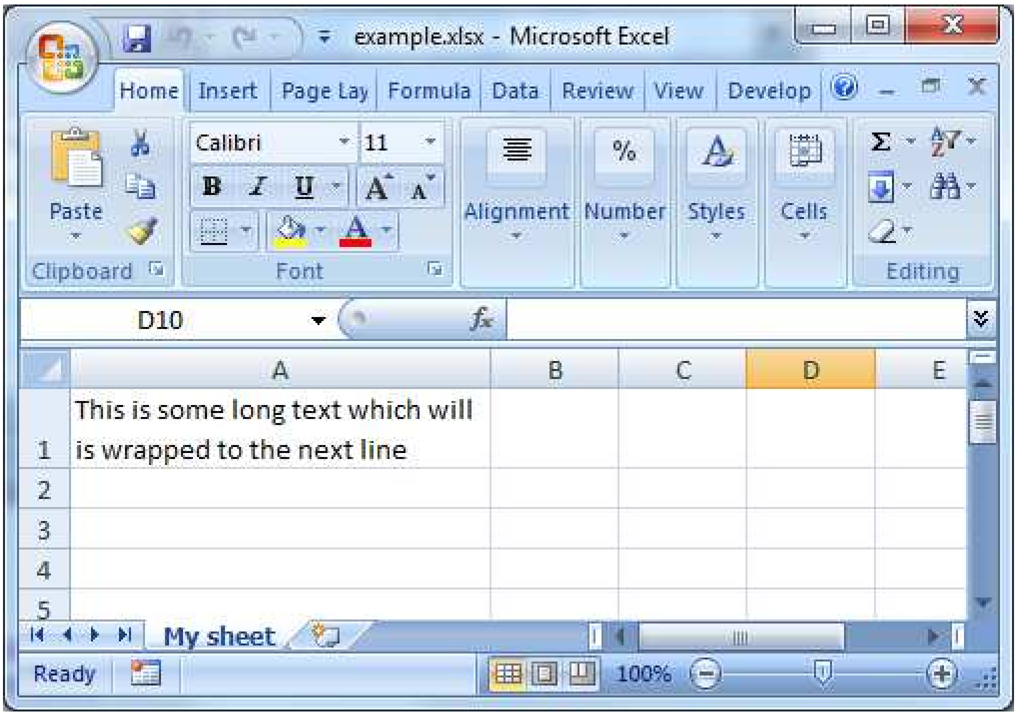


This is some long text which will is wrapped to the next line

Example:

```
DECLARE
    doc_id PLS_INTEGER;
    sheet_id PLS_INTEGER;
    row_id PLS_INTEGER;
BEGIN
    doc_id := ORA_EXCEL.new_document;
    sheet_id := ORA_EXCEL.add_sheet('My sheet', doc_id);

    row_id := ORA_EXCEL.add_row(doc_id, sheet_id);
    ORA_EXCEL.set_cell_value('A', 'This is some long text which will is wrapped
    to the next line', doc_id, sheet_id, row_id);
    ORA_EXCEL.set_column_width('A', 30, doc_id, sheet_id);
    ORA_EXCEL.set_cell_wrap_text('A', doc_id, sheet_id, row_id);
    ORA_EXCEL.save_to_file('EXPORT_DIR', 'example.xlsx', doc_id);
END;
```

Output:



This is some long text which will
is wrapped to the next line

Procedure ORA_EXCEL.merge_cells

Description:

ORA_EXCEL.merge_cells(cell_from varchar2, cell_to varchar2 [, doc_id pls_integer, sheet_id pls_integer, row_id pls_integer])

Join horizontal cells into one larger cell

Mandatory parameters:

- cell_from – begining cell to join
- cell_to – ending cell to join

Optional parameters:

- doc_id – document id
- sheet_id – sheet id
- row_id – row id

Returns:

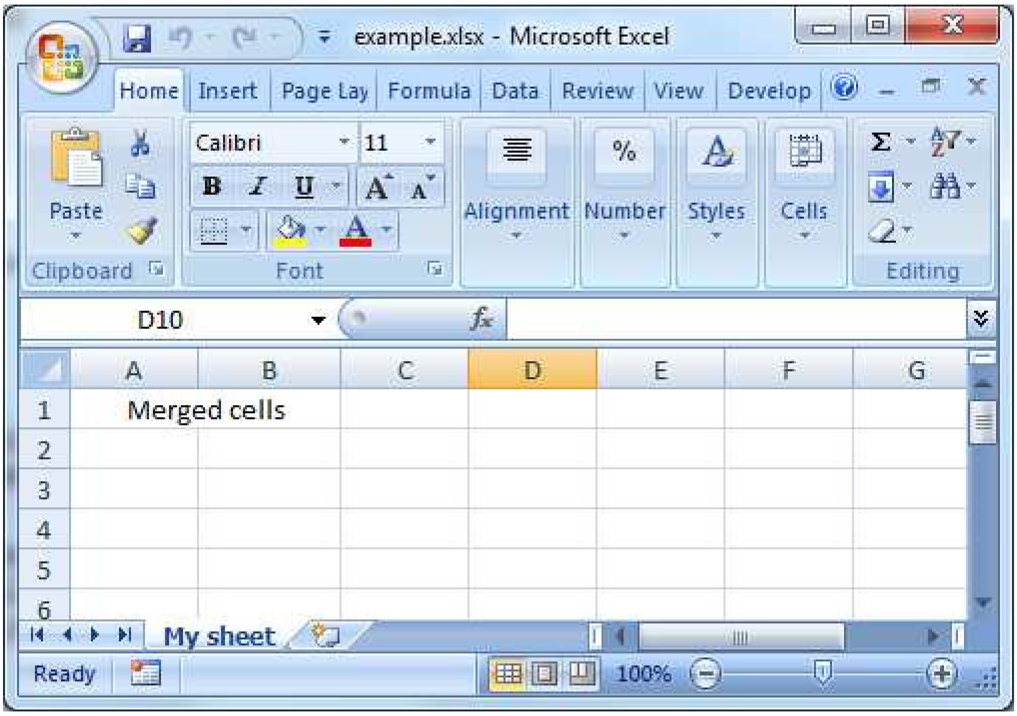
Procedure, does not return any value

Example:

```
BEGIN
    ORA_EXCEL.new_document;
    ORA_EXCEL.add_sheet('My sheet');
```

```
     ORA_EXCEL.add_row;
     ORA_EXCEL.set_cell_value('A', 'Merged cells');
     ORA_EXCEL.merge_cells('A', 'B');
     ORA_EXCEL.set_cell_align_center('A');
     ORA_EXCEL.save_to_file('EXPORT_DIR', 'example.xlsx');
END;
```

Output:

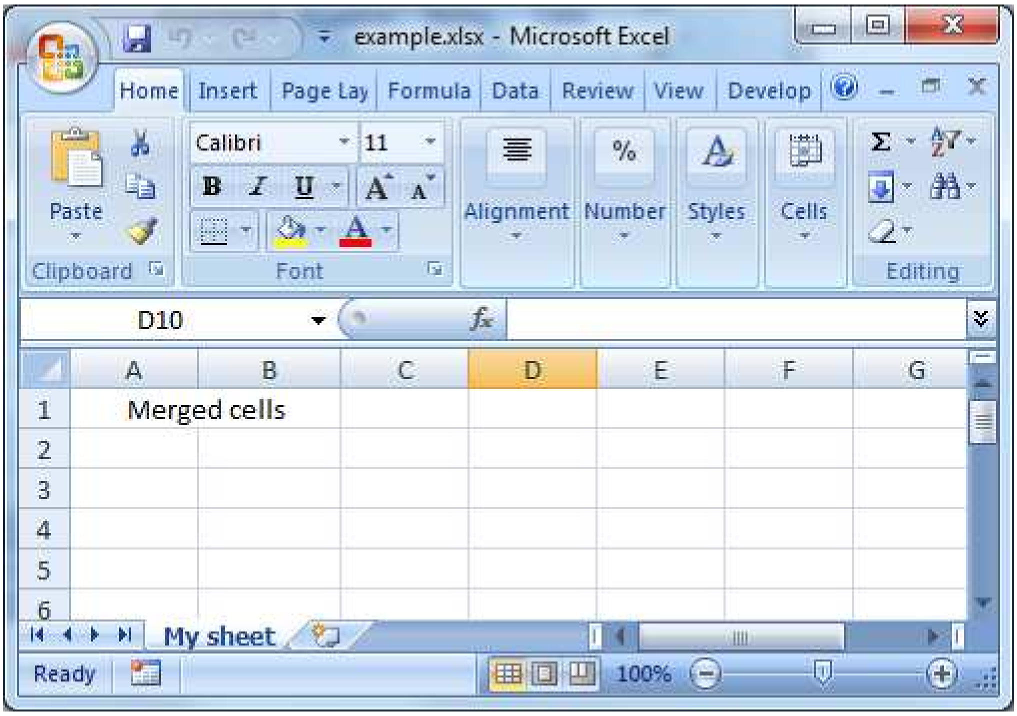


Merged cells

Example:

```
DECLARE
     doc_id PLS_INTEGER;
     sheet_id PLS_INTEGER;
     row_id PLS_INTEGER;
BEGIN
     doc_id := ORA_EXCEL.new_document;
     sheet_id := ORA_EXCEL.add_sheet('My sheet', doc_id);

     row_id := ORA_EXCEL.add_row(doc_id, sheet_id);
     ORA_EXCEL.set_cell_value('A', 'Merged cells', doc_id, sheet_id, row_id);
     ORA_EXCEL.merge_cells('A', 'B', doc_id, sheet_id, row_id);
     ORA_EXCEL.set_cell_align_center('A', doc_id, sheet_id, row_id);
     ORA_EXCEL.save_to_file('EXPORT_DIR', 'example.xlsx', doc_id);
END;
```

Output:



Merged cells

Procedure ORA_EXCEL.merge_rows

Description:

ORA_EXCEL.merge_rows (name varchar2, num_rows pls_integer [, doc_id pls_integer, sheet_id pls_integer, row_id pls_integer])

Join vertical cells into one larger cell

Mandatory parameters:

- name – name of begining cell
- num_rows – number of rows to merge

Optional parameters:

- doc_id – document id
- sheet_id – sheet id
- row_id – row id

Returns:

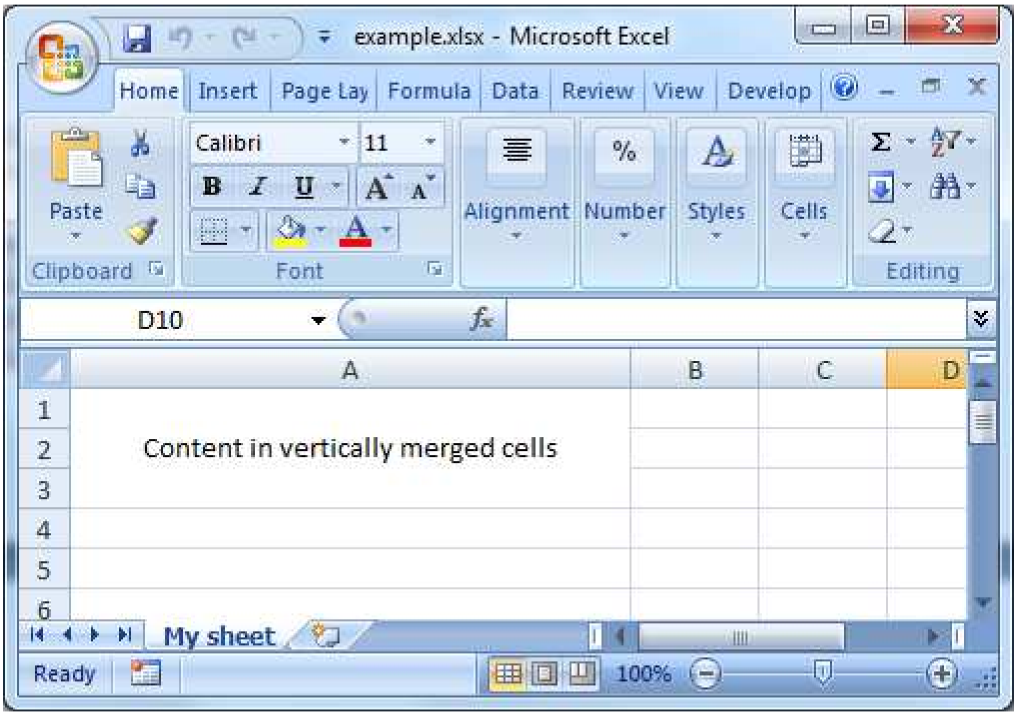
Procedure, does not return any value

Example:

```
BEGIN
      ORA_EXCEL.new_document;
      ORA_EXCEL.add_sheet('My sheet');
      ORA_EXCEL.set_column_width('A', 40);
      ORA_EXCEL.add_row;
```

```
     ORA_EXCEL.set_cell_value('A', 'Content in vertically merged cells');
     ORA_EXCEL.merge_rows('A', 2);
     ORA_EXCEL.set_cell_vert_align_middle('A');
     ORA_EXCEL.set_cell_align_center('A');
     ORA_EXCEL.save_to_file('EXPORT_DIR', 'example.xlsx');
END;
```

Example:

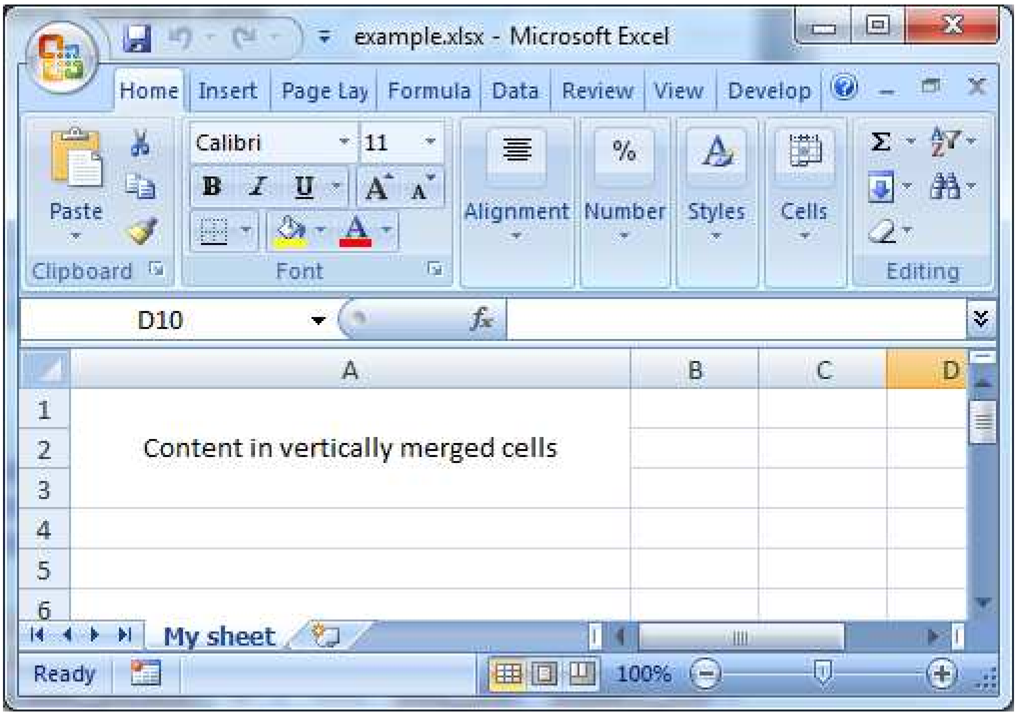


Content in vertically merged cells

Example:

```
DECLARE
     doc_id PLS_INTEGER;
     sheet_id PLS_INTEGER;
     row_id PLS_INTEGER;
BEGIN
     doc_id := ORA_EXCEL.new_document;
     sheet_id := ORA_EXCEL.add_sheet('My sheet', doc_id);
     ORA_EXCEL.set_column_width('A', 40, doc_id, sheet_id);

     row_id := ORA_EXCEL.add_row(doc_id, sheet_id);
     ORA_EXCEL.set_cell_value('A', 'Content in vertically merged cells', doc_id,
     sheet_id, row_id);
     ORA_EXCEL.merge_rows('A', 2, doc_id, sheet_id, row_id);
     ORA_EXCEL.set_cell_vert_align_middle('A', doc_id, sheet_id, row_id);
     ORA_EXCEL.set_cell_align_center('A', doc_id, sheet_id, row_id);
     ORA_EXCEL.save_to_file('EXPORT_DIR', 'example.xlsx', doc_id);
END;
```

Output:



Content in vertically merged cells

Procedure ORA_EXCEL.set_cell_format

Description:

ORA_EXCEL.set_cell_format (cell_name varchar2, format varchar2 [, doc_id pls_integer, sheet_id pls_integer, row_id pls_integer])

Set format of the cell content

Mandatory parameters:

- cell_name – cell name
- format – format string

Optional parameters:

- doc_id – document id
- sheet_id – sheet id
- row_id – row id

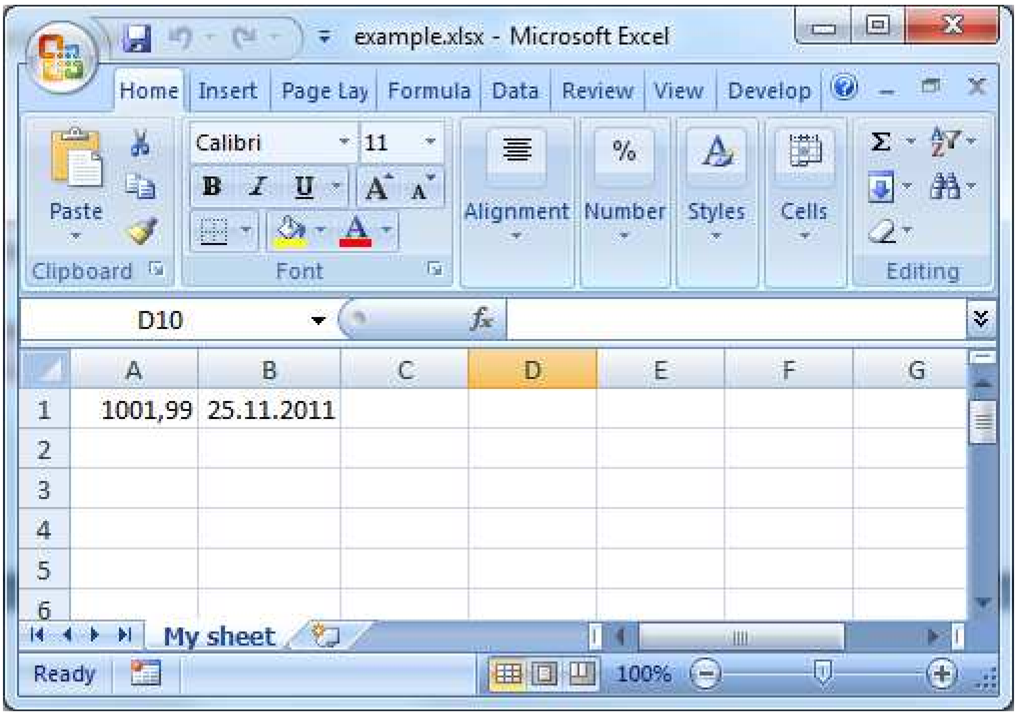
Returns:

Procedure, does not return any value

Example:

```
BEGIN
     ORA_EXCEL.new_document;
     ORA_EXCEL.add_sheet('My sheet');
     ORA_EXCEL.add_row;
     ORA_EXCEL.set_cell_value('A', 1001.99);
     ORA_EXCEL.set_cell_format('A', '0.00');

     ORA_EXCEL.set_cell_value('B', SYSDATE);
     ORA_EXCEL.set_cell_format('B', 'd.m.yyyy');

     ORA_EXCEL.save_to_file('EXPORT_DIR', 'example.xlsx');
END;
```

Output:



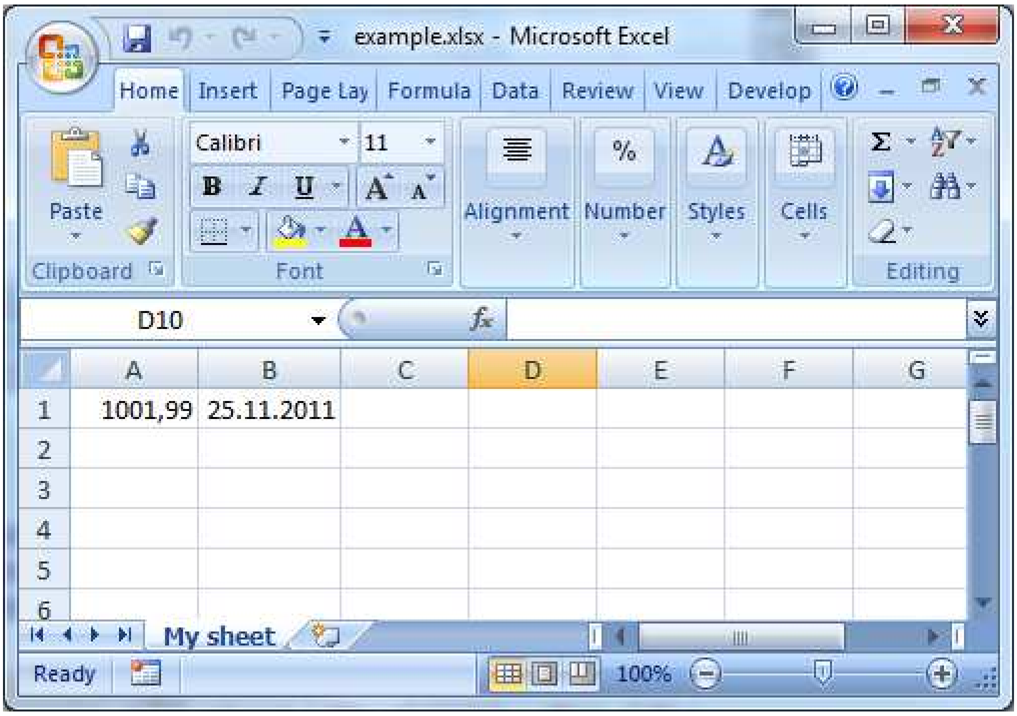
D10

	A	B	C	D	E	F	G
1	1001,99	25.11.2011					
2							
3							
4							
5							
6							

Exmaple:

```
DECLARE
     doc_id PLS_INTEGER;
     sheet_id PLS_INTEGER;
     row_id PLS_INTEGER;
BEGIN
     doc_id := ORA_EXCEL.new_document;
     sheet_id := ORA_EXCEL.add_sheet('My sheet', doc_id);

     row_id := ORA_EXCEL.add_row(doc_id, sheet_id);
     ORA_EXCEL.set_cell_value('A', 1001.99, doc_id, sheet_id, row_id);
     ORA_EXCEL.set_cell_format('A', '0.00', doc_id, sheet_id, row_id);

     ORA_EXCEL.set_cell_value('B', SYSDATE, doc_id, sheet_id, row_id);
     ORA_EXCEL.set_cell_format('B', 'd.m.yyyy', doc_id, sheet_id, row_id);
     ORA_EXCEL.save_to_file('EXPORT_DIR', 'example.xlsx', doc_id);
END;
```

Output:



	A	B	C	D	E	F	G
1	1001,99	25.11.2011					
2							
3							
4							
5							
6							

Procedure ORA_EXCEL.query_to_sheet

Description:

ORA_EXCEL.query_to_sheet(query varchar2 [, show_column_names boolean default true, doc_id pls_integer default current_doc_id, sheet_id pls_integer default current_sheet_id])

Get result from query and write it to sheet

Mandatory parameters:

- query – query string

Optional parameters:

- show_column_names – boolean parameter, true – show column names, false – do not show column names on first row
- doc_id – document id
- sheet_id – sheet id
- row_id – row id

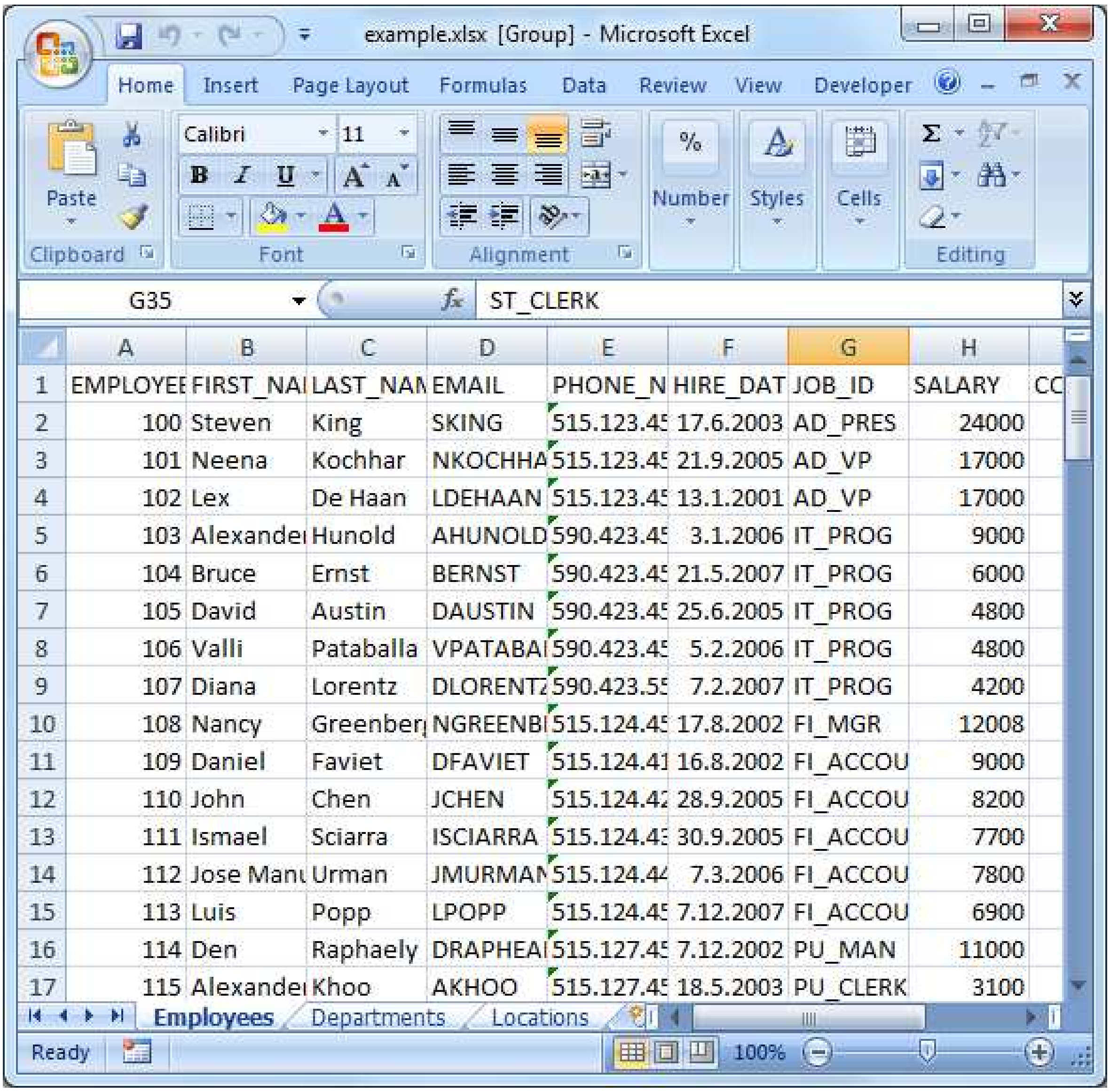
Returns:

Procedure, does not return any value

Example:

```
BEGIN
     ORA_EXCEL.new_document;

     ORA_EXCEL.add_sheet('Employees');
     ORA_EXCEL.query_to_sheet('select * from employees');

     ORA_EXCEL.add_sheet('Departments');
     ORA_EXCEL.query_to_sheet('select * from departments');

     ORA_EXCEL.add_sheet('Locations');
     ORA_EXCEL.query_to_sheet('select * from locations');

     ORA_EXCEL.save_to_file('EXPORT_DIR', example.xlsx');
END;
```

Output:



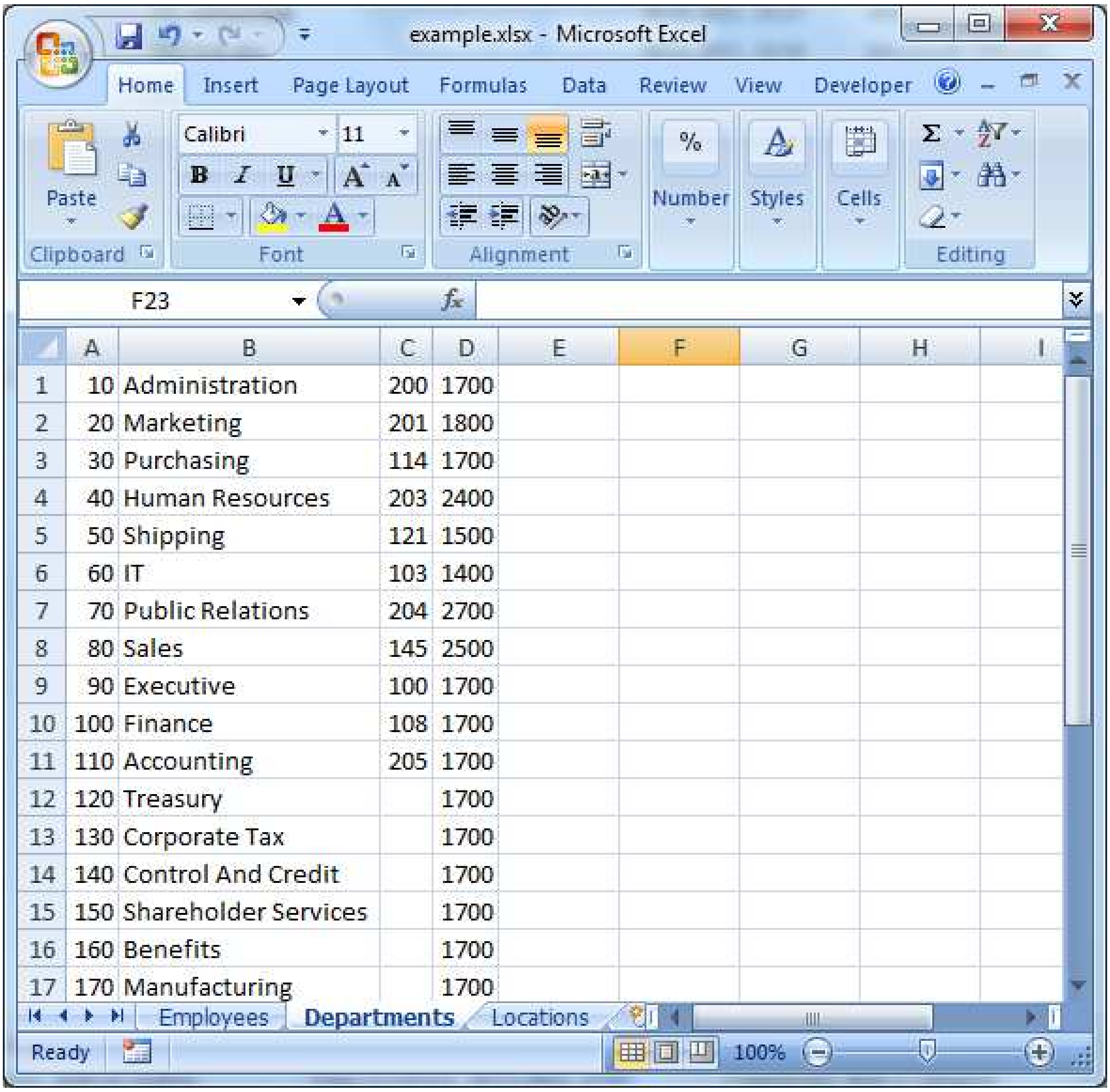
G35 | ST_CLERK

	A	B	C	D	E	F	G	H	
1	EMPLOYEE	FIRST_NA	LAST_NAM	EMAIL	PHONE_N	HIRE_DAT	JOB_ID	SALARY	CC
2	100	Steven	King	SKING	515.123.45	17.6.2003	AD_PRES	24000	
3	101	Neena	Kochhar	NKOCHHA	515.123.45	21.9.2005	AD_VP	17000	
4	102	Lex	De Haan	LDEHAAN	515.123.45	13.1.2001	AD_VP	17000	
5	103	Alexande	Hunold	AHUNOLD	590.423.45	3.1.2006	IT_PROG	9000	
6	104	Bruce	Ernst	BERNST	590.423.45	21.5.2007	IT_PROG	6000	
7	105	David	Austin	DAUSTIN	590.423.45	25.6.2005	IT_PROG	4800	
8	106	Valli	Pataballa	VPATABA	590.423.45	5.2.2006	IT_PROG	4800	
9	107	Diana	Lorentz	DLORENTZ	590.423.55	7.2.2007	IT_PROG	4200	
10	108	Nancy	Greenber	NGREENB	515.124.45	17.8.2002	FI_MGR	12008	
11	109	Daniel	Faviet	DFAVIET	515.124.41	16.8.2002	FI_ACCOU	9000	
12	110	John	Chen	JCHEN	515.124.42	28.9.2005	FI_ACCOU	8200	
13	111	Ismael	Sciarra	ISCIARRA	515.124.43	30.9.2005	FI_ACCOU	7700	
14	112	Jose Manu	Urman	JMURMAN	515.124.44	7.3.2006	FI_ACCOU	7800	
15	113	Luis	Popp	LPOPP	515.124.45	7.12.2007	FI_ACCOU	6900	
16	114	Den	Raphaely	DRAPHEA	515.127.45	7.12.2002	PU_MAN	11000	
17	115	Alexande	Khoo	AKHOO	515.127.45	18.5.2003	PU_CLERK	3100	

Exmaple:

```
DECLARE
     doc_id PLS_INTEGER;
     sheet_id PLS_INTEGER;
     row_id PLS_INTEGER;
BEGIN
     doc_id := ORA_EXCEL.new_document;

     sheet_id := ORA_EXCEL.add_sheet('Employees');
     ORA_EXCEL.query_to_sheet('select * from employees', TRUE, doc_id, sheet_id);

     sheet_id := ORA_EXCEL.add_sheet('Departments');
     ORA_EXCEL.query_to_sheet('select * from departments', FALSE, doc_id,
     sheet_id);

     sheet_id := ORA_EXCEL.add_sheet('Locations');
     ORA_EXCEL.query_to_sheet('select * from locations', TRUE, doc_id, sheet_id);

     ORA_EXCEL.save_to_file('EXPORT_DIR', 'example.xlsx');
END;
```

Output:



	A	B	C	D
1	10	Administration	200	1700
2	20	Marketing	201	1800
3	30	Purchasing	114	1700
4	40	Human Resources	203	2400
5	50	Shipping	121	1500
6	60	IT	103	1400
7	70	Public Relations	204	2700
8	80	Sales	145	2500
9	90	Executive	100	1700
10	100	Finance	108	1700
11	110	Accounting	205	1700
12	120	Treasury		1700
13	130	Corporate Tax		1700
14	140	Control And Credit		1700
15	150	Shareholder Services		1700
16	160	Benefits		1700
17	170	Manufacturing		1700

Procedure ORA_EXCEL. set_cell_formula

Description:

```
ORA_EXCEL.set_cell_formula(name VARCHAR2,
                           formula VARCHAR2,
                           [doc_id PLS_INTEGER DEFAULT current_doc_id,
                           sheet_id PLS_INTEGER DEFAULT current_sheet_id,
                           row_id PLS_INTEGER DEFAULT current_row_id])
```

Sets cell formula

Mandatory parameters:

- name - name of the cell where value will be added
- formula - formula that will be used to calculate cell value

Optional parameters:

- doc_id – document id
- sheet_id - id of sheet
- row_id - id of row on which height will be adjusted

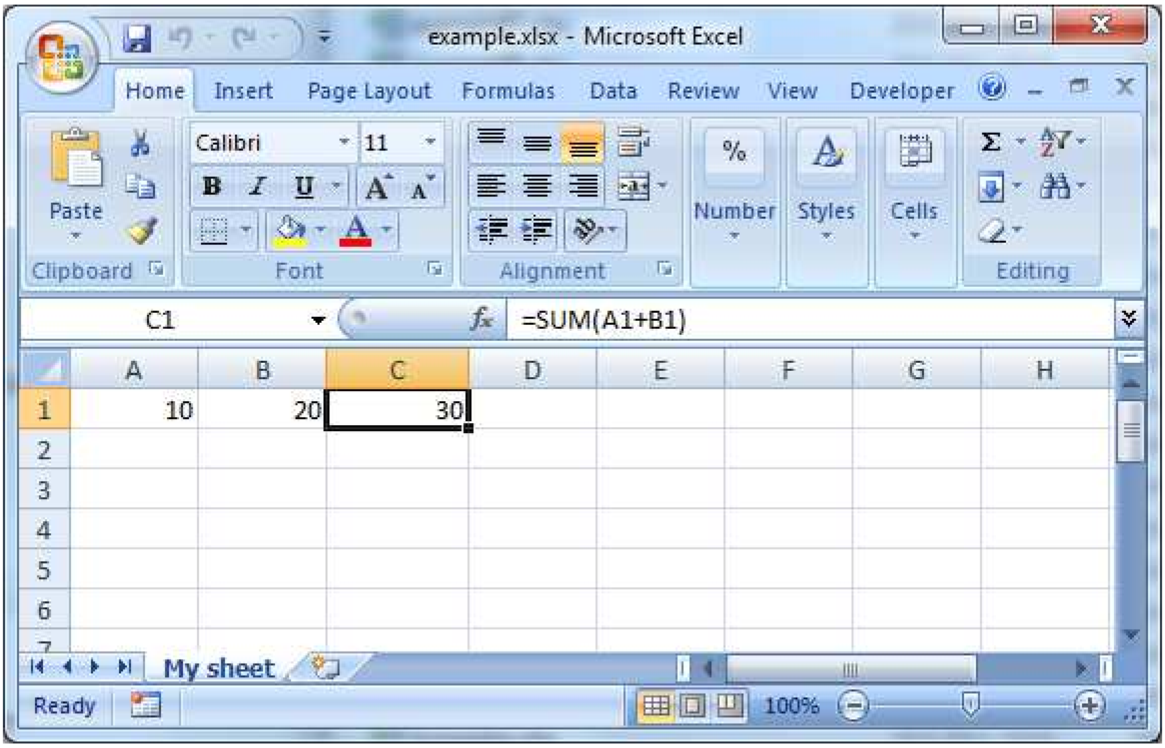
Returns:

Procedure, does not return any value

Example:

```
BEGIN
     ORA_EXCEL.new_document;
     ORA_EXCEL.add_sheet('My sheet');

     ORA_EXCEL.add_row;
     ORA_EXCEL.set_cell_value('A', 10);
     ORA_EXCEL.set_cell_value('B', 20);
     ORA_EXCEL.set_cell_value('C', 10 + 20);
     ORA_EXCEL.set_cell_formula('C', 'SUM(A1+B1)');
     -- Use ORA_EXCEL.current_row_id for current rown number

     ORA_EXCEL.save_to_file('EXPORT_DIR', 'example.xlsx');
END;
```

Output:



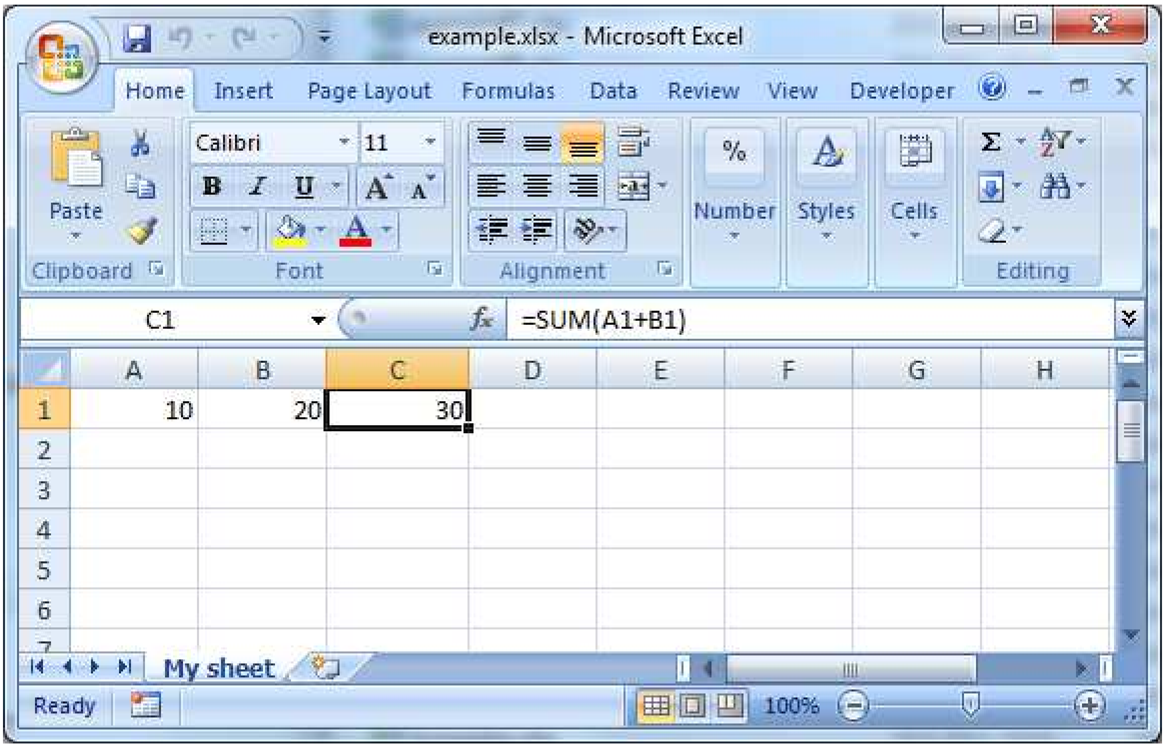
C1 =SUM(A1+B1)

	A	B	C	D	E	F	G	H
1	10	20	30					
2								
3								
4								
5								
6								

Example:

```
DECLARE
     doc_id PLS_INTEGER;
     sheet_id PLS_INTEGER;
     row_id PLS_INTEGER;
BEGIN
     doc_id := ORA_EXCEL.new_document;
     sheet_id := ORA_EXCEL.add_sheet('My sheet', doc_id);

     row_id := ORA_EXCEL.add_row(doc_id, sheet_id);
     ORA_EXCEL.set_cell_value('A', 10, doc_id, sheet_id, row_id);
     ORA_EXCEL.set_cell_value('B', 20, doc_id, sheet_id, row_id);
     ORA_EXCEL.set_cell_value('C', 10 + 20, doc_id, sheet_id, row_id);
     ORA_EXCEL.set_cell_formula('C', 'SUM(A1+B1)');
     -- Use ORA_EXCEL.current_row_id for current rown number

     ORA_EXCEL.save_to_file('EXPORT_DIR', 'example.xlsx', doc_id);
END;
```

Output:



C1 =SUM(A1+B1)

	A	B	C	D	E	F	G	H
1	10	20	30					
2								
3								
4								
5								
6								

Procedure ORA_EXCEL. set_cell_rotate_text

Description:

```
ORA_EXCEL. set_cell_rotate_text(name VARCHAR2,
                                 degrees INTEGER,
                                 [doc_id PLS_INTEGER DEFAULT current_doc_id,
                                 sheet_id PLS_INTEGER DEFAULT current_sheet_id,
                                 row_id PLS_INTEGER DEFAULT current_row_id])
```

Rotates text to a diagonal angle

Mandatory parameters:

- name - name of cell content will be centered
- degrees - degree from 90 to 180 which will be used to rotate text

Optional parameters:

- doc_id – document id
- sheet_id - id of sheet
- row_id - id of row on which height will be adjusted

Returns:

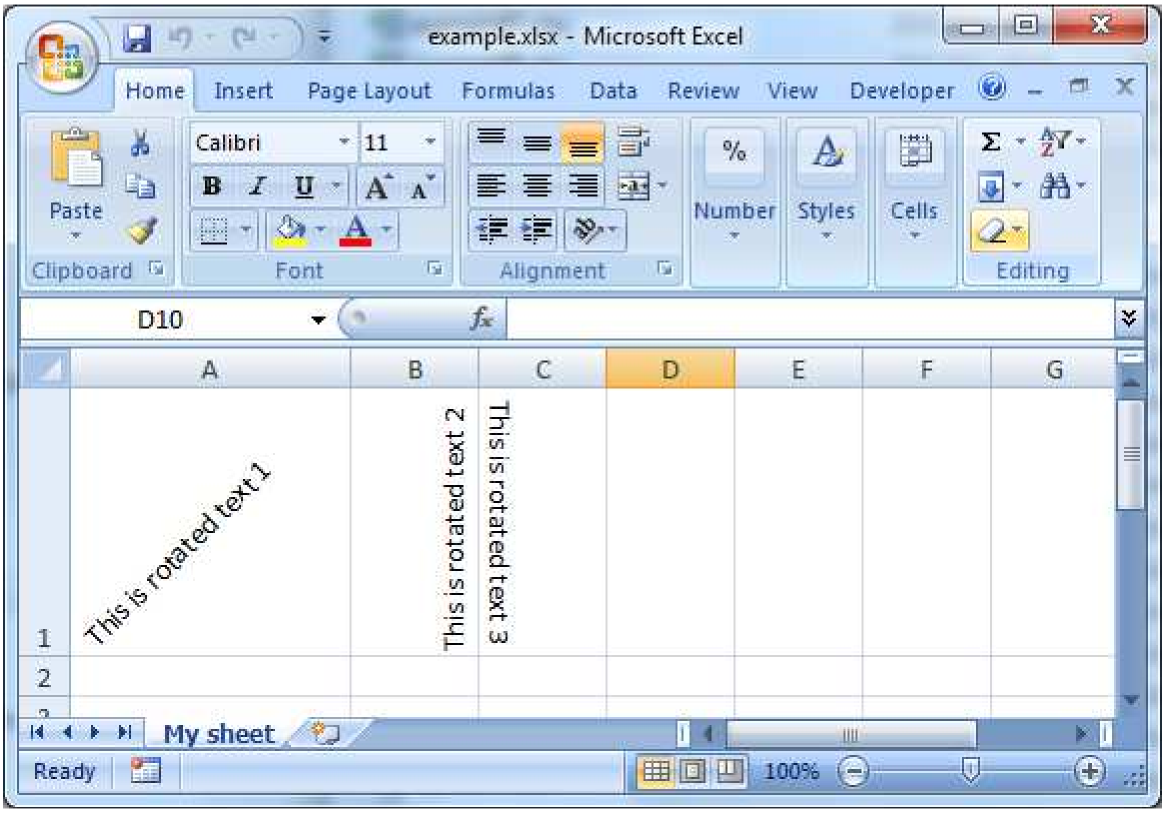
Procedure, does not return any value

Example:

```
BEGIN
     ORA_EXCEL.new_document;
     ORA_EXCEL.add_sheet('My sheet');
```

```
    ORA_EXCEL.add_row;
    ORA_EXCEL.set_cell_value('A', 'This is rotated text');
    ORA_EXCEL.set_cell_rotate_text('A', 45);
    ORA_EXCEL.set_column_width('A', 20);

    ORA_EXCEL.set_cell_value('B', 'This is rotated text');
    ORA_EXCEL.set_cell_rotate_text('B', 90);

    ORA_EXCEL.set_cell_value('C', 'This is rotated text');
    ORA_EXCEL.set_cell_rotate_text('c', 180);

    ORA_EXCEL.save_to_file('EXPORT_DIR', 'example.xlsx');

END;
```

Output:



This is rotated text 1

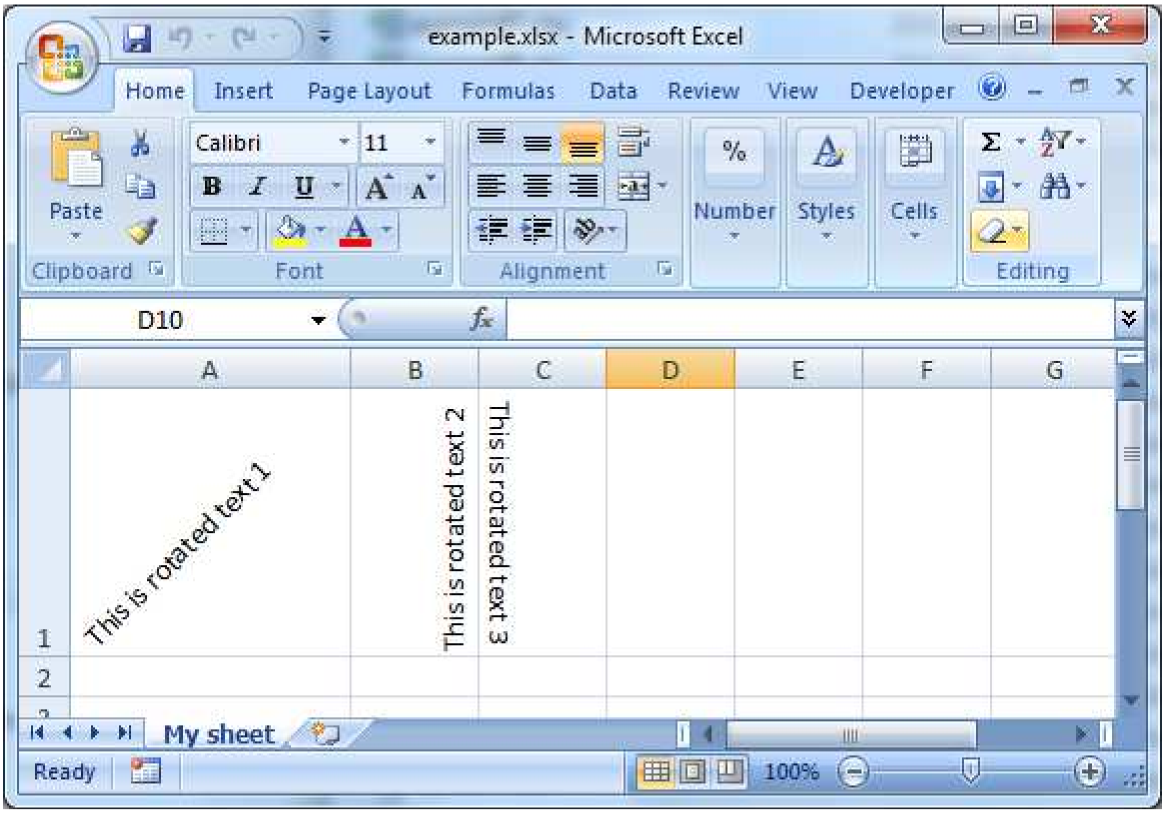
This is rotated text 2

This is rotated text 3

Example:

```
DECLARE
    doc_id PLS_INTEGER;
    sheet_id PLS_INTEGER;
    row_id PLS_INTEGER;
BEGIN
    doc_id := ORA_EXCEL.new_document;
    sheet_id := ORA_EXCEL.add_sheet('My sheet', doc_id);

    row_id := ORA_EXCEL.add_row(doc_id, sheet_id);
    ORA_EXCEL.set_cell_value('A', 'This is rotated text 1',
                             doc_id, sheet_id, row_id);
    ORA_EXCEL.set_cell_rotate_text('A', 45, doc_id, sheet_id, row_id);
    ORA_EXCEL.set_column_width('A', 20, doc_id, sheet_id);
    ORA_EXCEL.set_cell_value('B', 'This is rotated text 2',
                             doc_id, sheet_id, row_id);
    ORA_EXCEL.set_cell_rotate_text('B', 90, doc_id, sheet_id, row_id);

    ORA_EXCEL.set_cell_value('C', 'This is rotated text 3',
                             doc_id, sheet_id, row_id);
    ORA_EXCEL.set_cell_rotate_text('c', 180, doc_id, sheet_id, row_id);

    ORA_EXCEL.save_to_file('EXPORT_DIR', 'example.xlsx', doc_id);

END;
```

Output:



This is rotated text 1

This is rotated text 2

This is rotated text 3

Procedure ORA_EXCEL. set_sheet_margins

Description:

```
ORA_EXCEL. set_sheet_margins(left_margin NUMBER,
                             right_margin NUMBER,
                              top_margin NUMBER,
                              bottom_margin NUMBER,
                              header_margin NUMBER,
                              footer_margin NUMBER,
                              [sheet_id PLS_INTEGER DEFAULT current_sheet_id])
```

Rotates text to a diagonal angle

Mandatory parameters:

- left_margin - margin size on the left side of sheet

- right_margin - margin size on the right side of sheet
- top_margin - margin size on the top side of sheet
- bottom_margin - margin size on the bottom side of sheet
- header_margin - margin size on the header side of sheet
- footer_margin - margin size on the footer side of sheet

Optional parameters:

- sheet_id - id of sheet

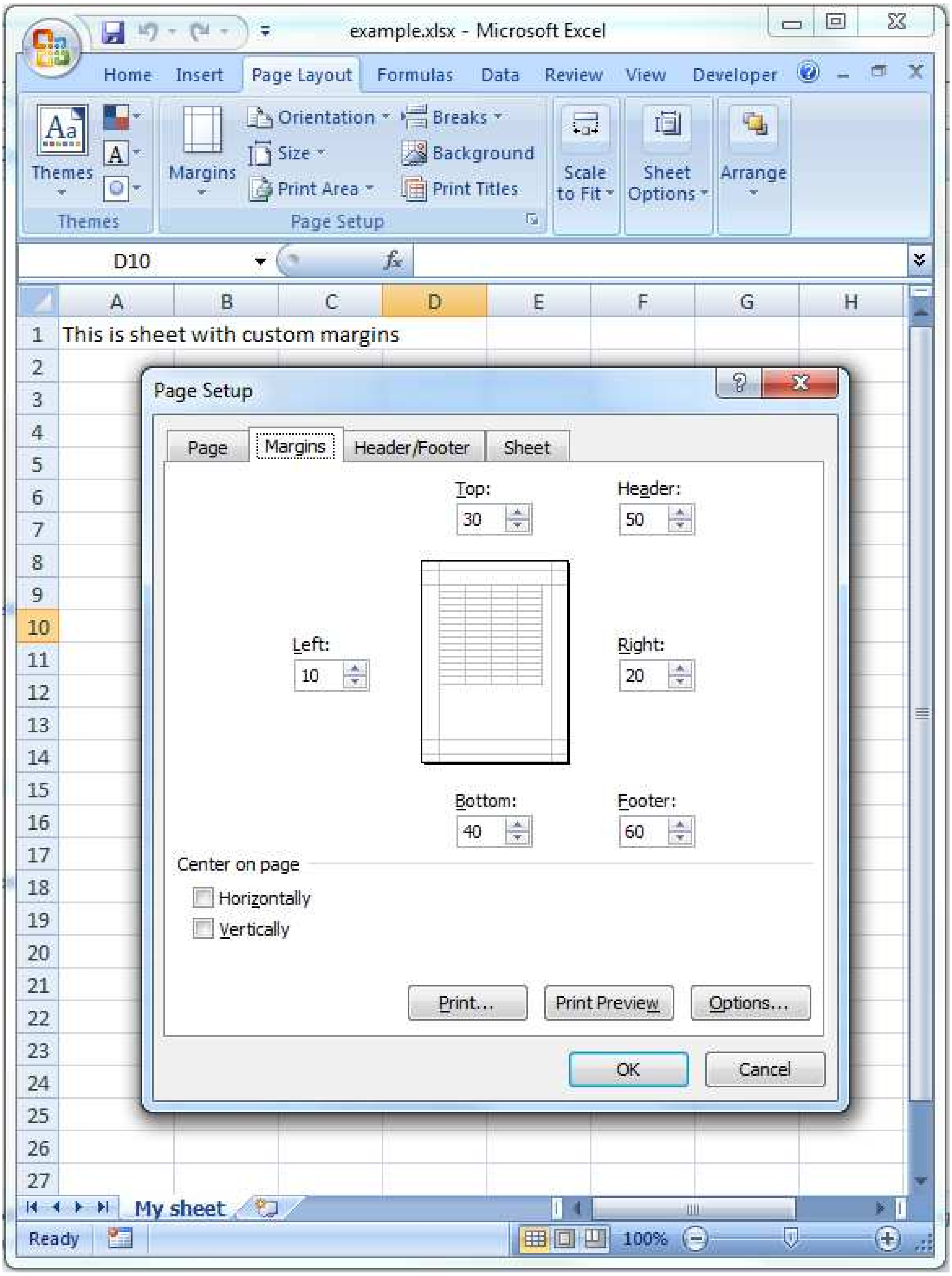
Returns:

Procedure, does not return any value

Example:

```
BEGIN
     ORA_EXCEL.new_document;
     ORA_EXCEL.add_sheet('My sheet');

     ORA_EXCEL.add_row;
     ORA_EXCEL.set_cell_value('A', 'This is sheet with custom margins');
     ORA_EXCEL.set_sheet_margins(left_margin => 10,
                                 right_margin => 20,
                                 top_margin => 30,
                                 bottom_margin => 40,
                                 header_margin => 50,
                                 footer_margin => 60);

     ORA_EXCEL.save_to_file('EXPORT_DIR', 'example.xlsx');

END;
```

Output:



This is sheet with custom margins

Top: 30

Header: 50

Left: 10

Right: 20

Bottom: 40

Footer: 60

Center on page

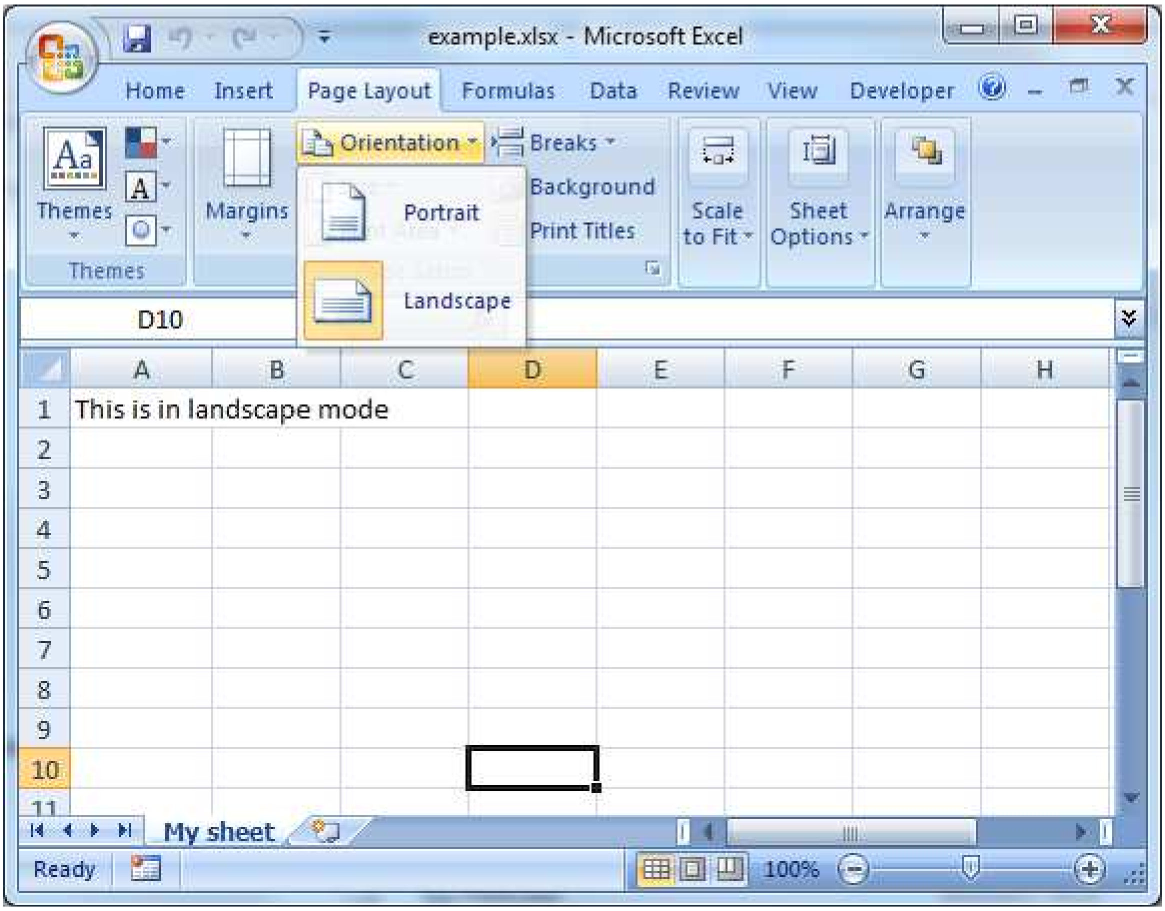
☐ Horizontally

☐ Vertically

Example:

```
DECLARE
     doc_id PLS_INTEGER;
     sheet_id PLS_INTEGER;
     row_id PLS_INTEGER;
BEGIN
     doc_id := ORA_EXCEL.new_document;
     sheet_id := ORA_EXCEL.add_sheet('My sheet', doc_id);

     row_id := ORA_EXCEL.add_row(doc_id, sheet_id);
     ORA_EXCEL.set_cell_value('A', 'This is sheet with custom margins',
                                    doc_id, sheet_id, row_id);
     ORA_EXCEL.set_sheet_margins(left_margin => 10,
                                       right_margin => 20,
                                       top_margin => 30,
                                       bottom_margin => 40,
                                       header_margin => 50,
                                       footer_margin => 60,
                                       sheet_id => sheet_id);

     ORA_EXCEL.save_to_file('EXPORT_DIR', 'example.xlsx', doc_id);

END;
```

Output:



This is in landscape mode

Procedure ORA_EXCEL. set_sheet_landscape

Description:

```
ORA_EXCEL. set_sheet_landscape(sheet_id PLS_INTEGER DEFAULT current_sheet_id)
```

Sets sheet orientation to landscape

Mandatory parameters:

- -

Optional parameters:

- sheet_id - id of sheet

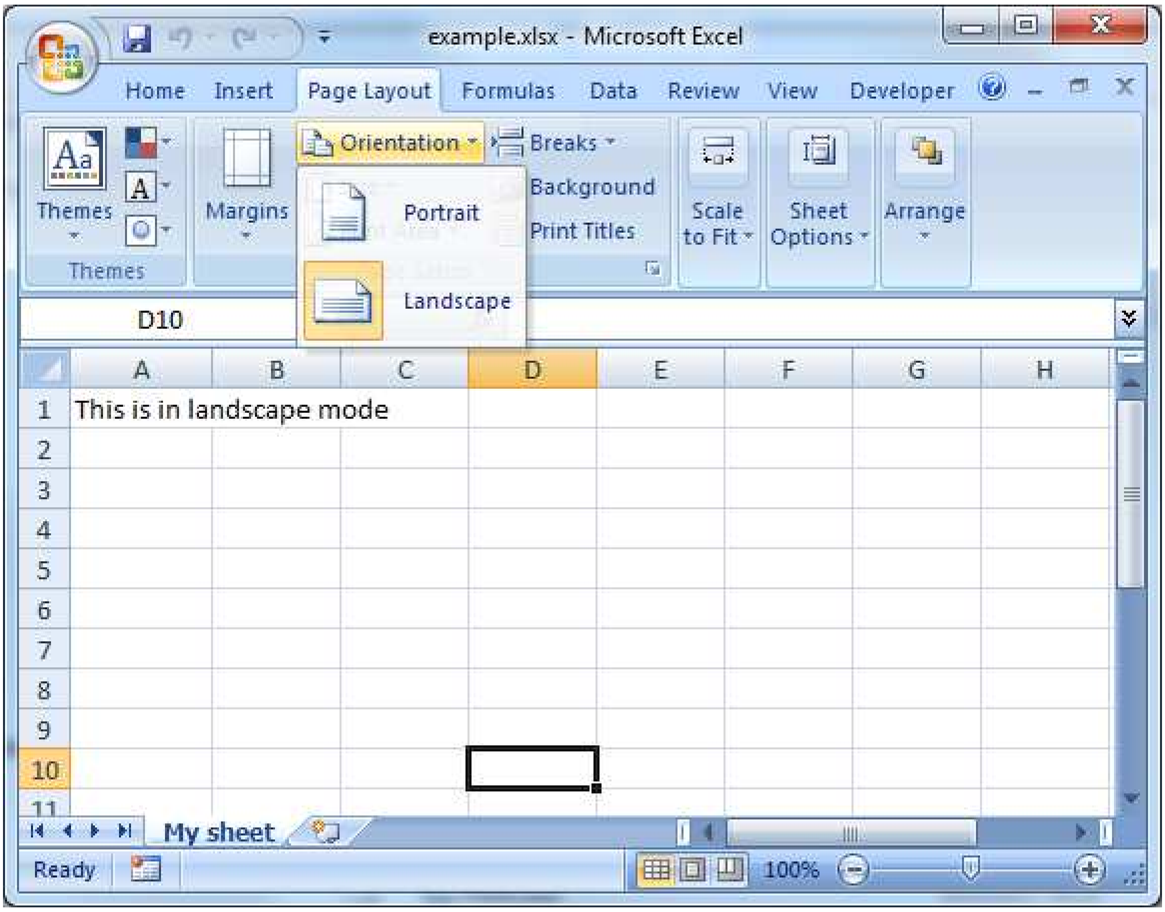
Returns:

Procedure, does not return any value

Example:

```
BEGIN
     ORA_EXCEL.new_document;

     ORA_EXCEL.add_sheet('My sheet');
     ORA_EXCEL.set_sheet_landscape;

     ORA_EXCEL.add_row;
     ORA_EXCEL.set_cell_value('A', 'This is in landscape mode');

     ORA_EXCEL.save_to_file('EXPORT_DIR', 'example.xlsx');

END;
```

Output:



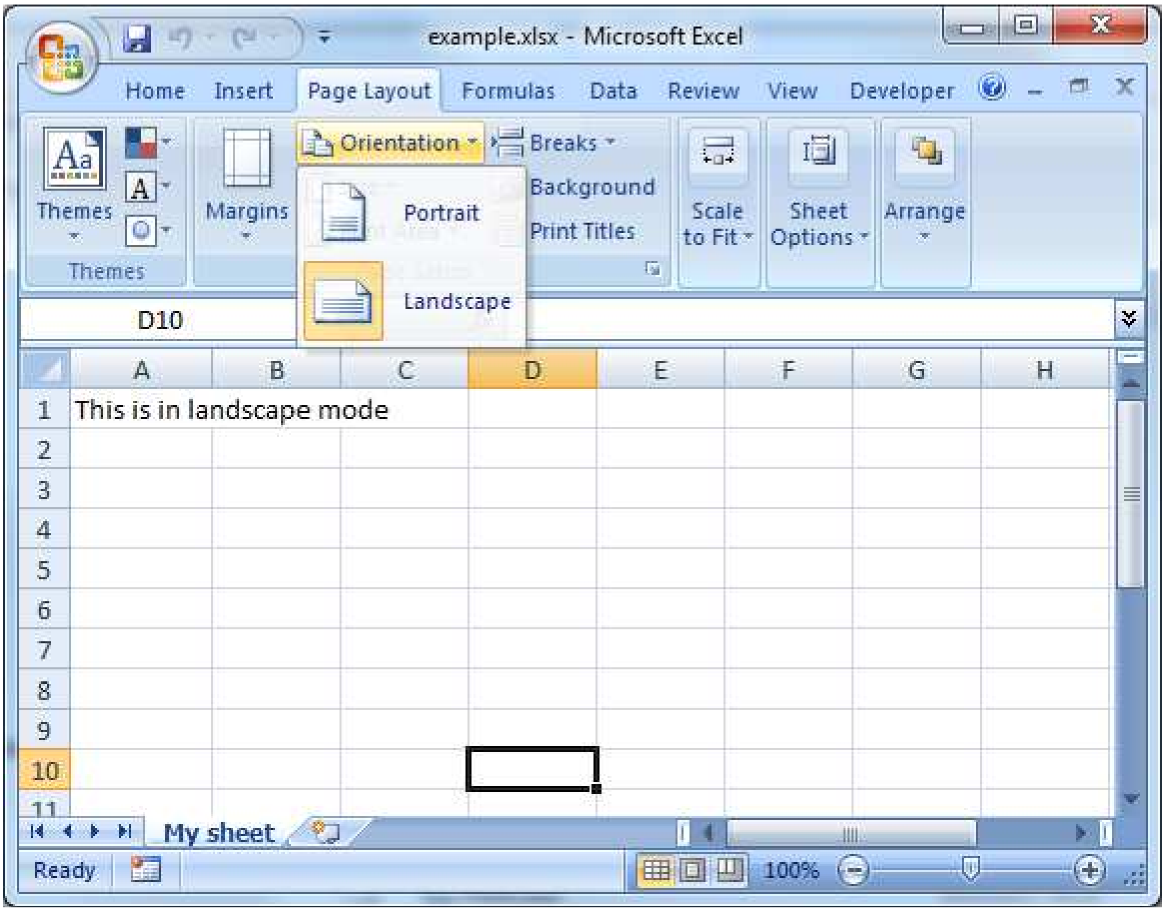
This is in landscape mode

Example:

```
DECLARE
```

```
    doc_id PLS_INTEGER;
    sheet_id PLS_INTEGER;
    row_id PLS_INTEGER;
BEGIN
    doc_id := ORA_EXCEL.new_document;
    sheet_id := ORA_EXCEL.add_sheet('My sheet', doc_id);
    ORA_EXCEL.set_sheet_landscape(sheet_id);

    row_id := ORA_EXCEL.add_row(doc_id, sheet_id);
    ORA_EXCEL.set_cell_value('A', 'This is in landscape mode',
                                           doc_id, sheet_id, row_id);

    ORA_EXCEL.save_to_file('EXPORT_DIR', 'example.xlsx');


END;
```

Output:



	A
1	This is in landscape mode

Procedure ORA_EXCEL. set_sheet_paper_size

Description:

```
ORA_EXCEL. set_sheet_paper_size(paper_size INTEGER,
                                  sheet_id PLS_INTEGER DEFAULT current_sheet_id)
```

Sets sheet paper size

Mandatory parameters:

- paper_size - paper size of sheet

Paper size values:
1 - Letter (8-1/2 in. x 11 in.)
2 - Letter Small (8-1/2 in. x 11 in.)
3 - Tabloid (11 in. x 17 in.)
4 - Ledger (17 in. x 11 in.)
5 - Legal (8-1/2 in. x 14 in.)
6 - Statement (5-1/2 in. x 8-1/2 in.)
7 - Executive (7-1/2 in. x 10-1/2 in.)
8 - A3 (297 mm x 420 mm)
9 - A4 (210 mm x 297 mm)
10 - A4 Small (210 mm x 297 mm)
11 - A5 (148 mm x 210 mm)
12 - B4 (250 mm x 354 mm)
13 - A5 (148 mm x 210 mm)
14 - Folio (8-1/2 in. x 13 in.)
15 - Quarto (215 mm x 275 mm)
16 - 10 in. x 14 in.
17 - 11 in. x 17 in.
18 - Note (8-1/2 in. x 11 in.)
19 - Envelope #9 (3-7/8 in. x 8-7/8 in.)
20 - Envelope #10 (4-1/8 in. x 9-1/2 in.)
21 - Envelope #11 (4-1/2 in. x 10-3/8 in.)
22 - Envelope #12 (4-1/2 in. x 11 in.)
23 - Envelope #14 (5 in. x 11-1/2 in.)
24 - C size sheet
25 - D size sheet
26 - E size sheet
27 - Envelope DL (110 mm x 220 mm)
28 - Envelope C5 (162 mm x 229 mm)
29 - Envelope C3 (324 mm x 458 mm)
30 - Envelope C4 (229 mm x 324 mm)
31 - Envelope C6 (114 mm x 162 mm)
32 - Envelope C65 (114 mm x 229 mm)
33 - Envelope B4 (250 mm x 353 mm)
34 - Envelope B5 (176 mm x 250 mm)
35 - Envelope B6 (176 mm x 125 mm)
36 - Envelope (110 mm x 230 mm)
37 - Envelope Monarch (3-7/8 in. x 7-1/2 in.)
38 - Envelope (3-5/8 in. x 6-1/2 in.)
39 - U.S. Standard Fanfold (14-7/8 in. x 11 in.)
40 - German Legal Fanfold (8-1/2 in. x 13 in.)
41 - German Legal Fanfold (8-1/2 in. x 13 in.)

Optional parameters:

- sheet_id - id of sheet

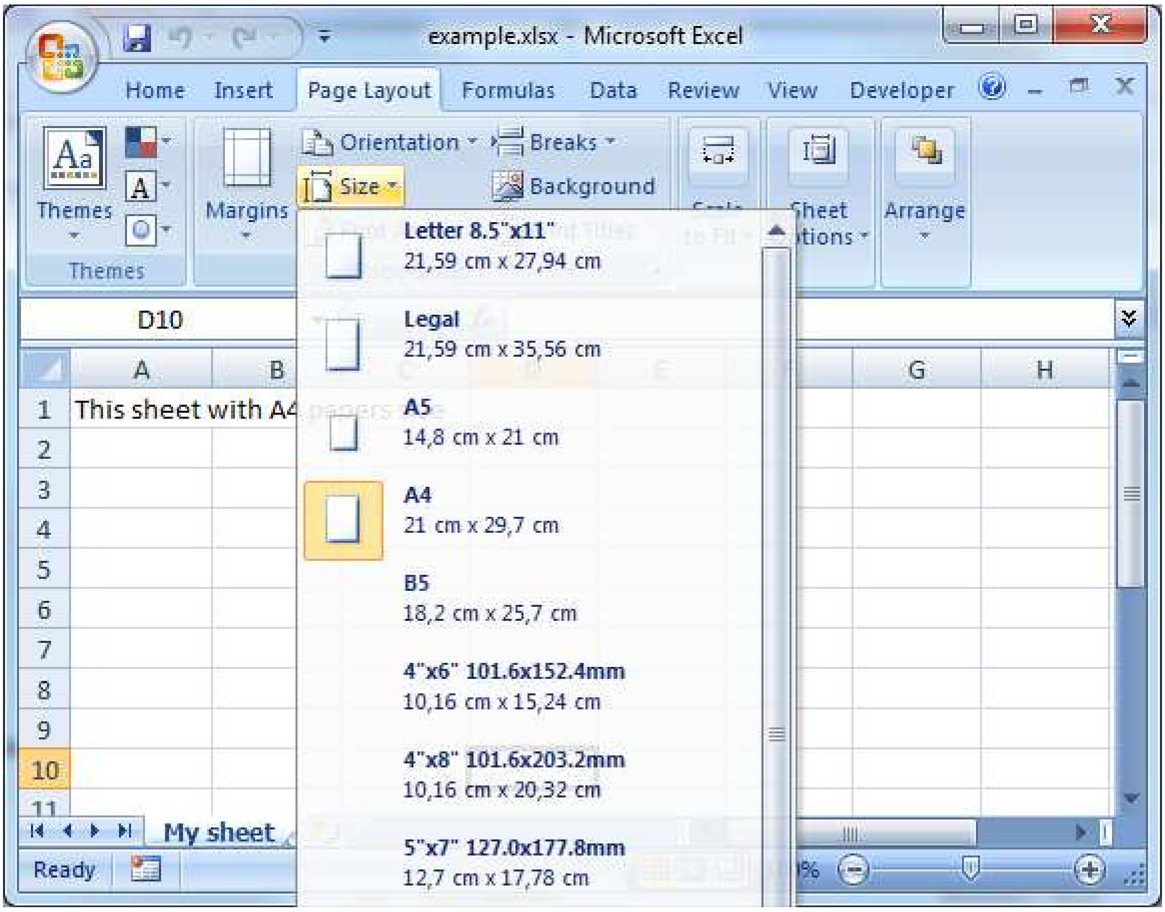
Returns:

Procedure, does not return any value

Example:

```
BEGIN
      ORA_EXCEL.new_document;

      ORA_EXCEL.add_sheet('My sheet');
      ORA_EXCEL.set_sheet_paper_size(9);

      ORA_EXCEL.add_row;
      ORA_EXCEL.set_cell_value('A', 'This sheet with A4 papers size');

      ORA_EXCEL.save_to_file('EXPORT_DIR', 'example.xlsx');

END;
```

Output:

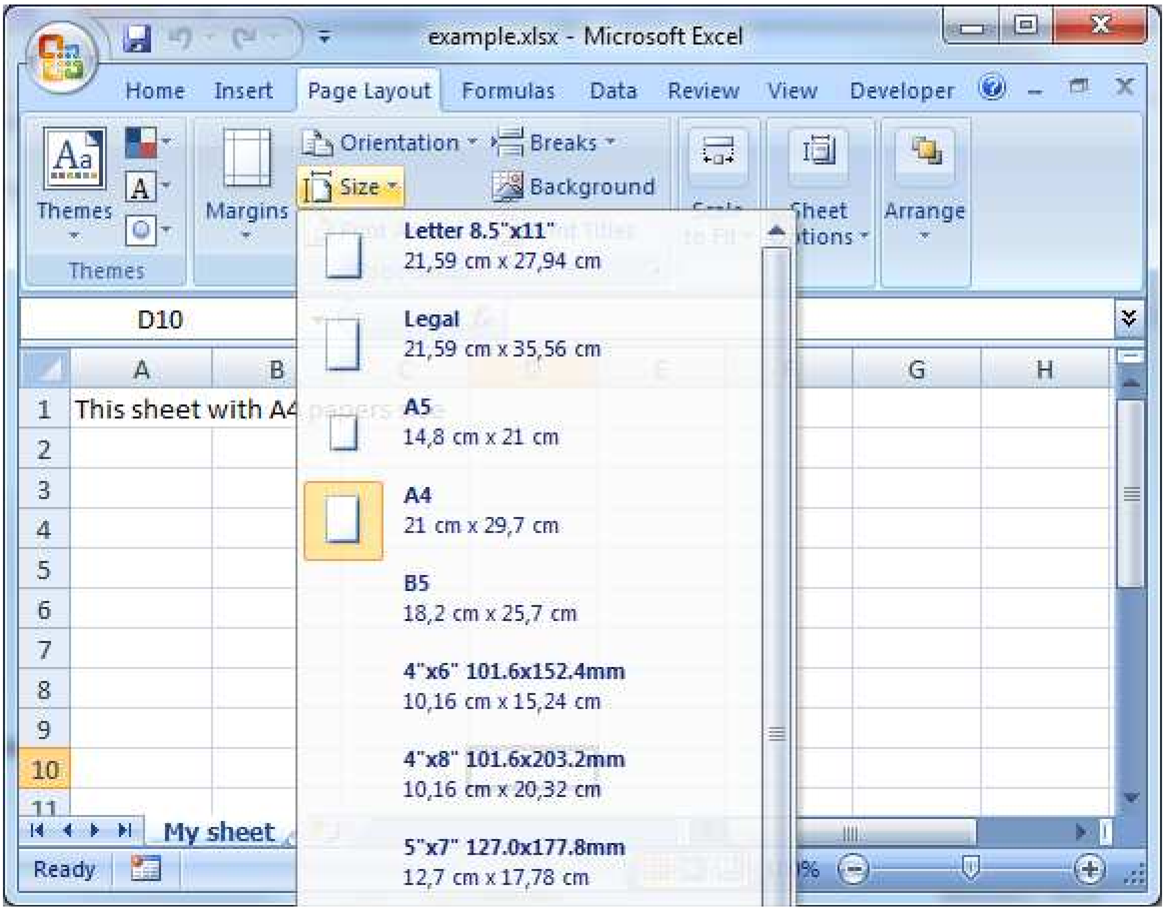


Example:

```
DECLARE
      doc_id PLS_INTEGER;
      sheet_id PLS_INTEGER;
      row_id PLS_INTEGER;
BEGIN
```

```
    doc_id := ORA_EXCEL.new_document;
    sheet_id := ORA_EXCEL.add_sheet('My sheet', doc_id);
    ORA_EXCEL.set_sheet_paper_size(9, sheet_id);

    row_id := ORA_EXCEL.add_row(doc_id, sheet_id);
    ORA_EXCEL.set_cell_value('A', 'This sheet with A4 papers size',
                               doc_id, sheet_id, row_id);

    ORA_EXCEL.save_to_file('EXPORT_DIR', 'example.xlsx', doc_id);

END;
```

Output:



Procedure ORA_EXCEL. set_sheet_header_text

Description:

```
ORA_EXCEL. set_sheet_header_text(header_text VARCHAR2,
                                 sheet_id PLS_INTEGER DEFAULT current_sheet_id)
```

Sets sheet header text

Mandatory parameters:

- header_text - text that will be displayed on sheets header, limited to 1000 characters

Optional parameters:

- sheet_id - id of sheet

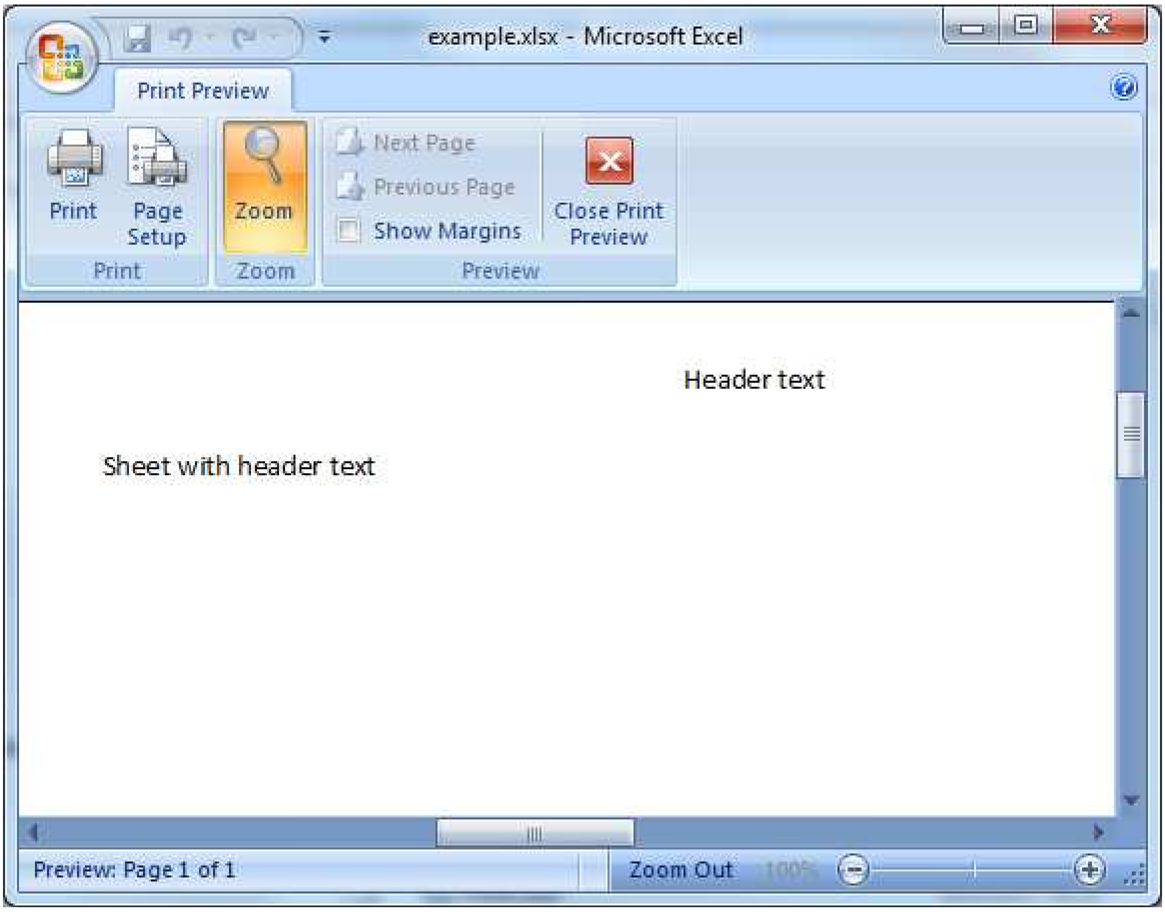
Returns:

Procedure, does not return any value

Example:

```
BEGIN
	ORA_EXCEL.new_document;

	ORA_EXCEL.add_sheet('My sheet');
	ORA_EXCEL.set_sheet_header_text('Header text');

	ORA_EXCEL.add_row;
	ORA_EXCEL.set_cell_value('A', 'Sheet with header text');

	ORA_EXCEL.save_to_file('EXPORT_DIR', 'example.xlsx');
END;
```

Output:



Header text

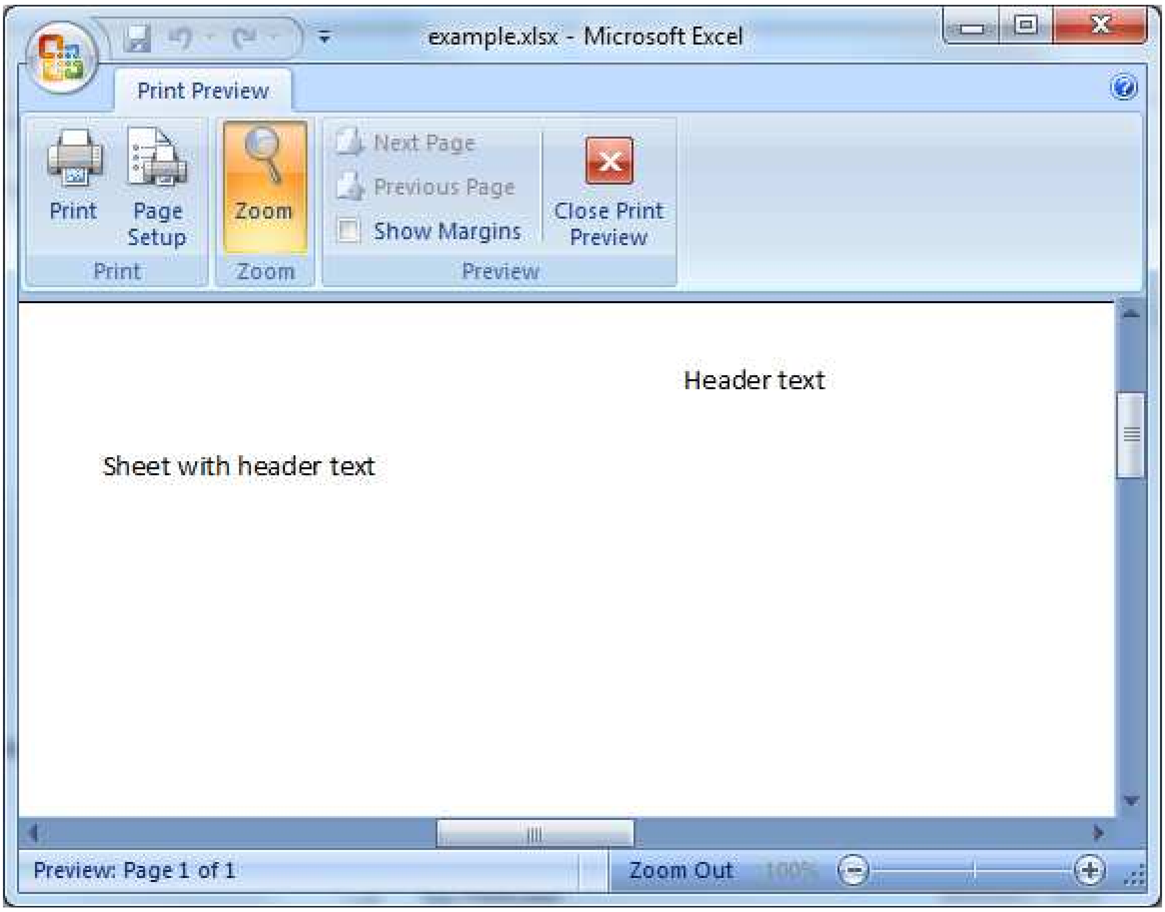
Sheet with header text

Example:

```
DECLARE
	doc_id PLS_INTEGER;
	sheet_id PLS_INTEGER;
	row_id PLS_INTEGER;
BEGIN
	doc_id := ORA_EXCEL.new_document;
```

```
    sheet_id := ORA_EXCEL.add_sheet('My sheet', doc_id);
    ORA_EXCEL.set_sheet_header_text('Header text', sheet_id);

    row_id := ORA_EXCEL.add_row(doc_id, sheet_id);
    ORA_EXCEL.set_cell_value('A', 'Sheet with header text',
                                   doc_id, sheet_id, row_id);

    ORA_EXCEL.save_to_file('EXPORT_DIR', 'example.xlsx', doc_id);
END;
```

Output:



Header text

Sheet with header text

Procedure ORA_EXCEL. set_sheet_footer_text

Description:

```
ORA_EXCEL. set_sheet_footer_text(header_text VARCHAR2,
                                  sheet_id PLS_INTEGER DEFAULT current_sheet_id)
```

Sets sheet foter text

Mandatory parameters:

- footer_text - text that will be displayed on sheets footer, limited to 1000 characters

Optional parameters:

- sheet_id - id of sheet

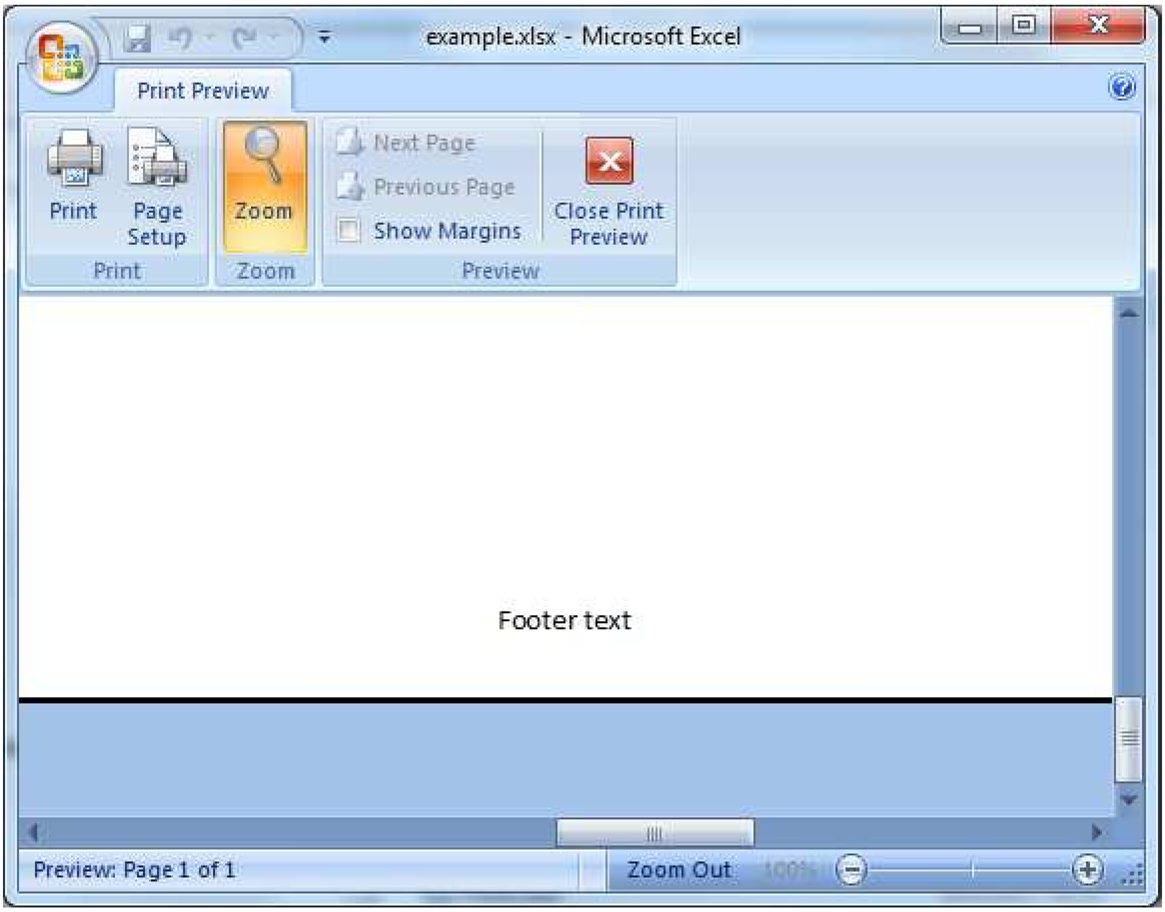
Returns:

Procedure, does not return any value

Example:

```
BEGIN
     ORA_EXCEL.new_document;

     ORA_EXCEL.add_sheet('My sheet');
     ORA_EXCEL.set_sheet_footer_text(Footer text');

     ORA_EXCEL.add_row;
     ORA_EXCEL.set_cell_value('A', 'Sheet with footer text');

     ORA_EXCEL.save_to_file('EXPORT_DIR', 'example.xlsx');
END;
```

Output:



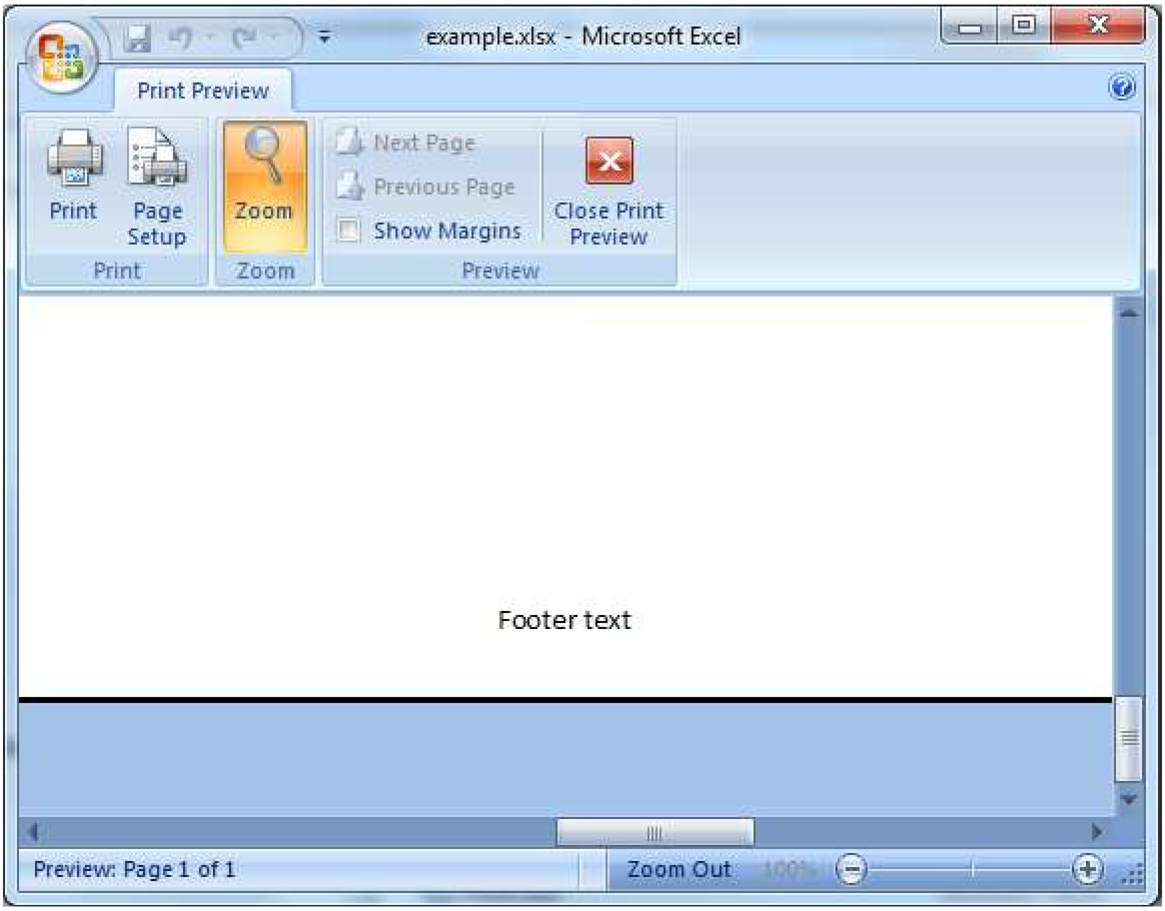
Footer text

Example:

```
DECLARE
     doc_id PLS_INTEGER;
     sheet_id PLS_INTEGER;
     row_id PLS_INTEGER;
BEGIN
     doc_id := ORA_EXCEL.new_document;
     sheet_id := ORA_EXCEL.add_sheet('My sheet', doc_id);
     ORA_EXCEL.set_sheet_footer_text('Footer text', sheet_id);

     row_id := ORA_EXCEL.add_row(doc_id, sheet_id);
```

```
    ORA_EXCEL.set_cell_value('A', 'Sheet with footer text',
                                  doc_id, sheet_id, row_id);

    ORA_EXCEL.save_to_file('EXPORT_DIR', 'example.xlsx', doc_id);
END;
```

Output:



Procedure ORA_EXCEL. set_cell_hyperlink

Description:

```
ORA_EXCEL.set_cell_hyperlink(name VARCHAR2,
                             hyperlink VARCHAR2,
                             doc_id PLS_INTEGER DEFAULT current_doc_id,
                             sheet_id PLS_INTEGER DEFAULT current_sheet_id,
                             row_id PLS_INTEGER DEFAULT current_row_id)
```

Sets hyperlink for cell

Mandatory parameters:

- name - cell name
- hyperlink - hyperlink that will be set on cell

Optional parameters:

- doc_id - id of dpcument

- sheet_id - id of sheet
- row_id - id of row

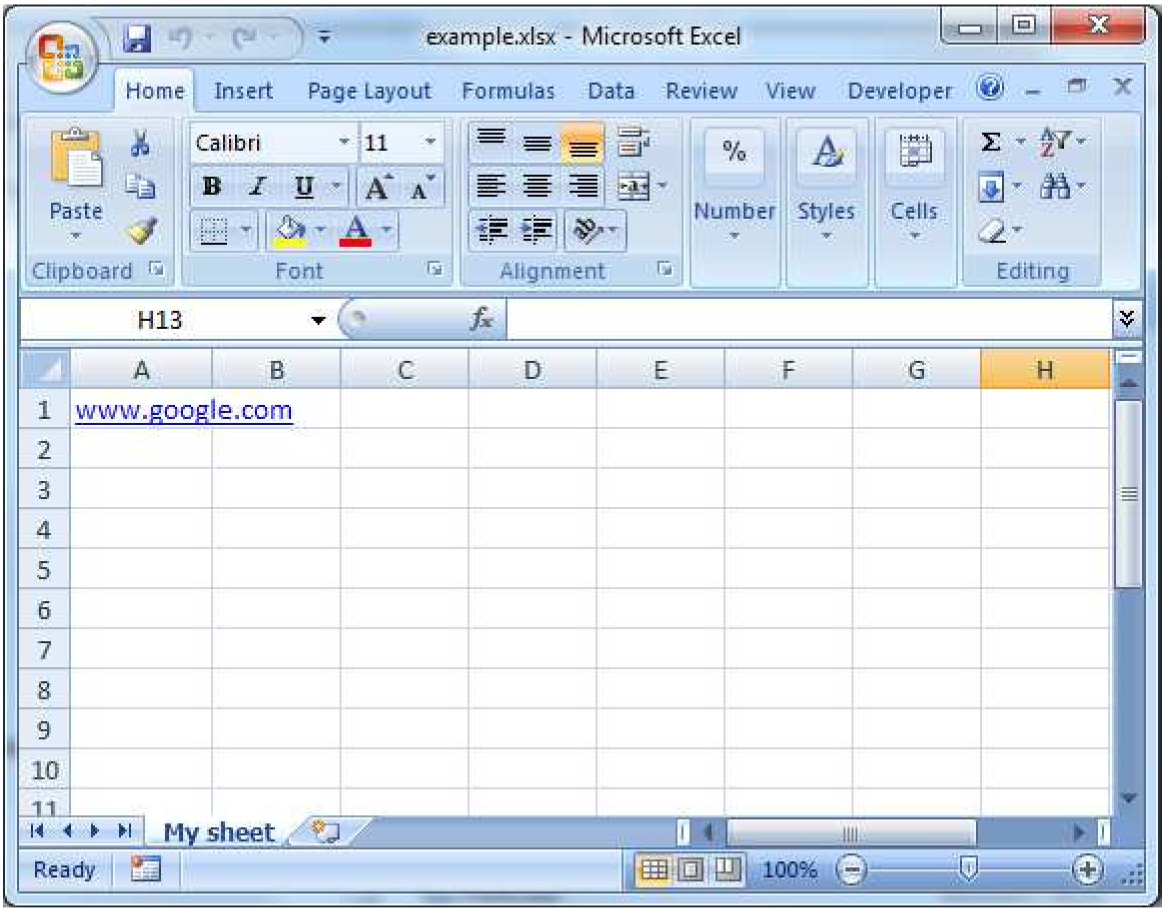
Returns:

Procedure, does not return any value

Example:

```
BEGIN
     ORA_EXCEL.new_document;

     ORA_EXCEL.add_sheet('My sheet');


     ORA_EXCEL.add_row;
     ORA_EXCEL.set_cell_value('A', 'www.google.com');
     ORA_EXCEL.set_cell_hyperlink('A', 'http://www.google.com');

     ORA_EXCEL.save_to_file('EXPORT_DIR', 'example.xlsx');

END;
```

Output:



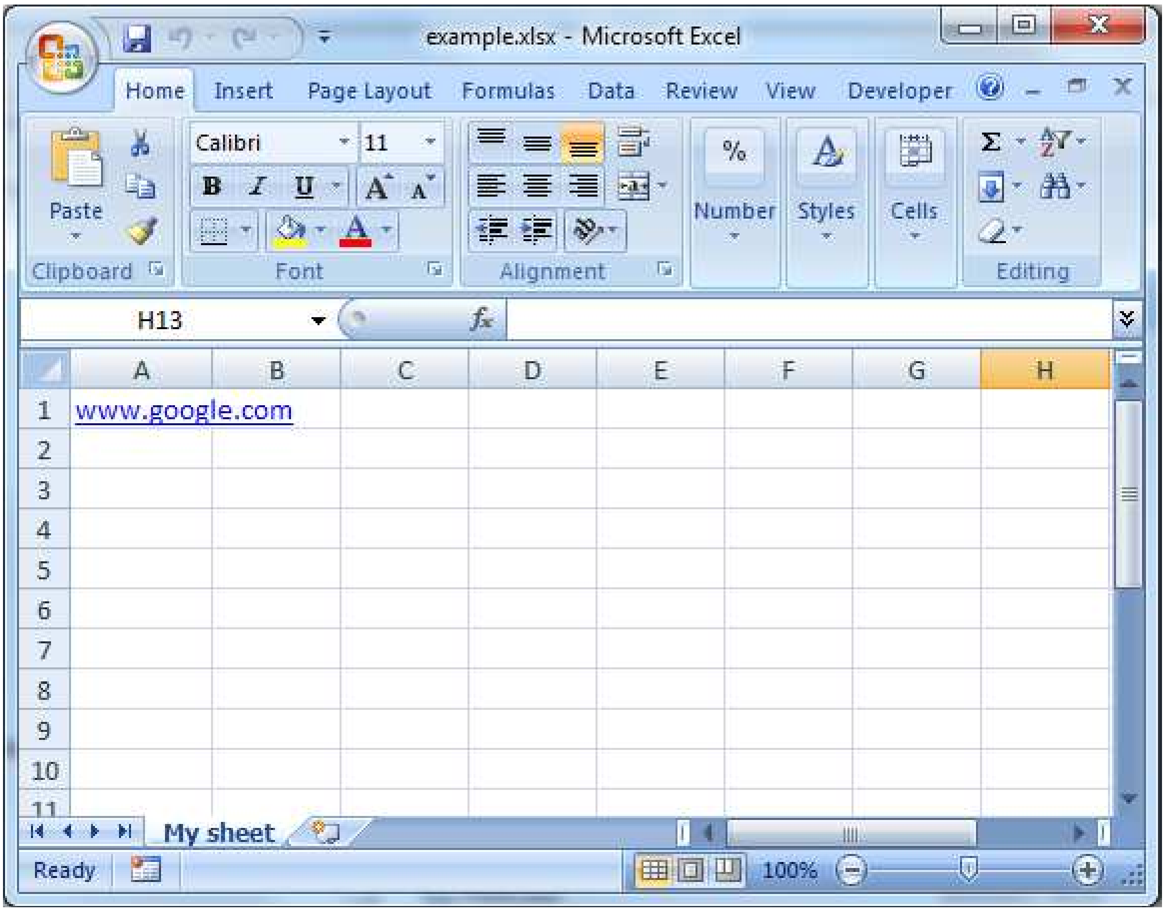
www.google.com

Example:

```
DECLARE
     doc_id PLS_INTEGER;
     sheet_id PLS_INTEGER;
     row_id PLS_INTEGER;
```

```
BEGIN
     doc_id := ORA_EXCEL.new_document;
     sheet_id := ORA_EXCEL.add_sheet('My sheet', doc_id);

     row_id := ORA_EXCEL.add_row(doc_id, sheet_id);
     ORA_EXCEL.set_cell_value('A', 'www.google.com',
                                   doc_id, sheet_id, row_id);
     ORA_EXCEL.set_cell_hyperlink('A', 'http://www.google.com',
                                   doc_id, sheet_id, row_id);


     ORA_EXCEL.save_to_file('EXPORT_DIR', 'example.xlsx', doc_id);

END;
```

Output:



Procedure ORA_EXCEL. set_cell_indent_left

Description:

```
ORA_EXCEL.set_cell_indent_left(name VARCHAR2,
                               indent INTEGER,
                               [doc_id PLS_INTEGER DEFAULT current_doc_id,
                               sheet_id PLS_INTEGER DEFAULT current_sheet_id,
                               row_id PLS_INTEGER DEFAULT current_row_id])
```

Sets left indent within the cell

Mandatory parameters:

- name - name of cell content will be indented from the left side
- indent - number of indent from left site of cell

Optional parameters:

- doc_id - id of dpcument
- sheet_id - id of sheet
- row_id - id of row

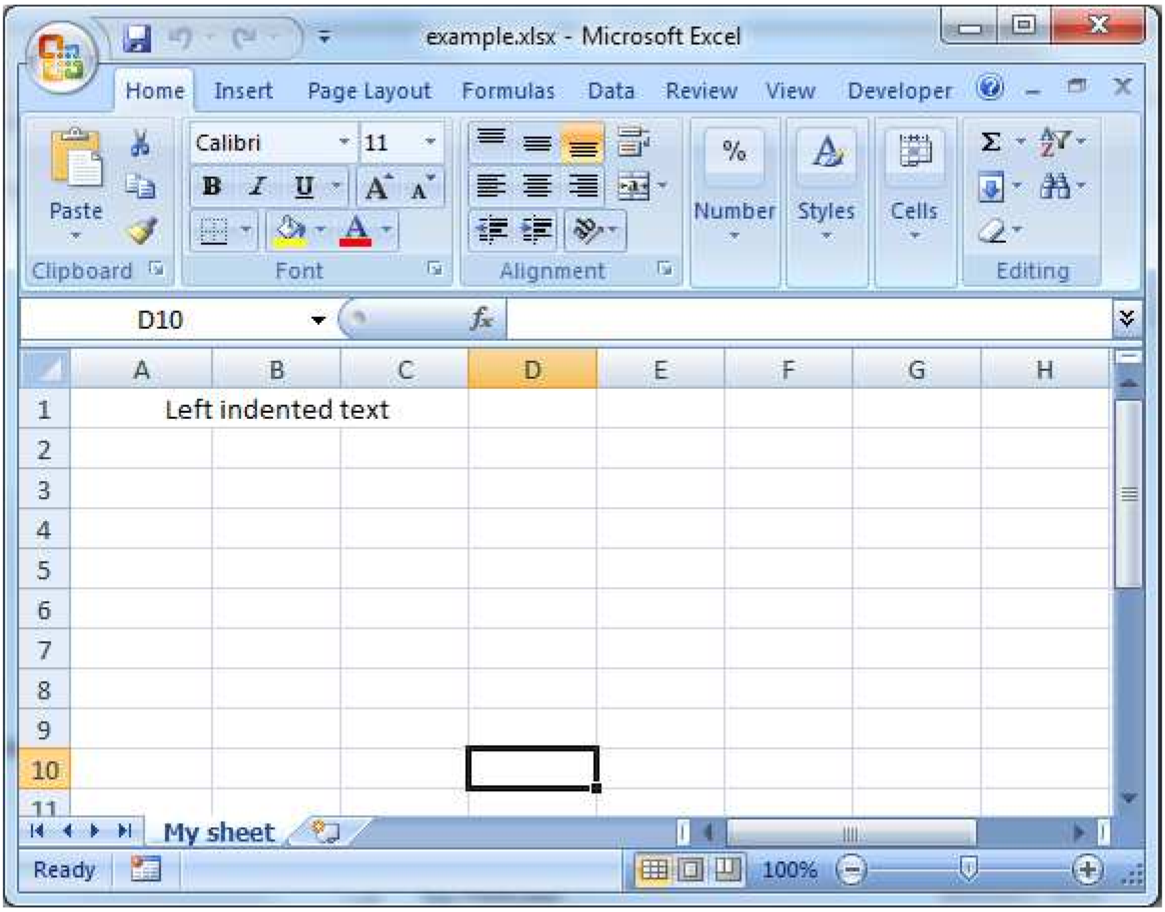
Returns:

Procedure, does not return any value

Example:

```
BEGIN
    ORA_EXCEL.new_document;

    ORA_EXCEL.add_sheet('My sheet');

    ORA_EXCEL.add_row;
    ORA_EXCEL.set_cell_value('A', 'Left indented text');
    ORA_EXCEL.set_cell_indent_left('A', 5);

    ORA_EXCEL.save_to_file('EXPORT_DIR', 'example.xlsx');

END;
```

Output:

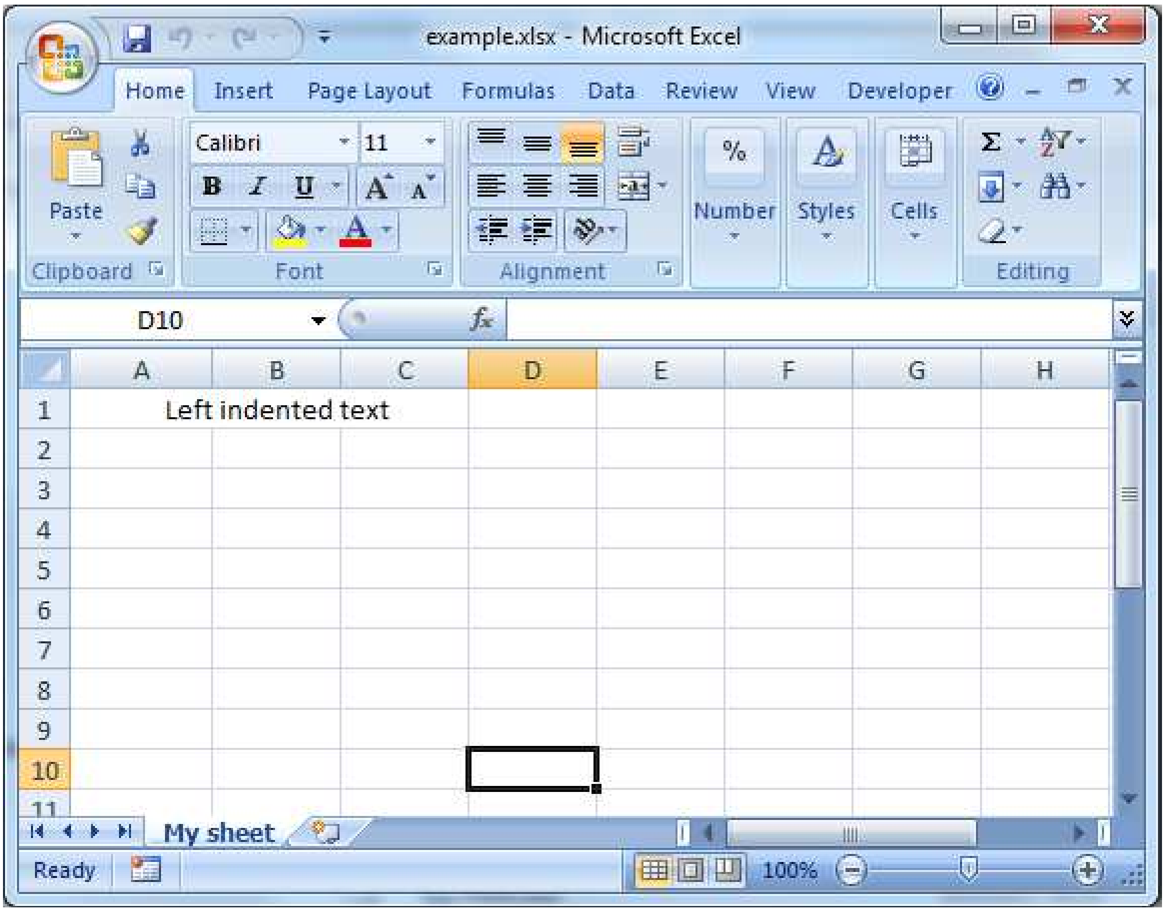


Left indented text

Example:

```
DECLARE
     doc_id PLS_INTEGER;
     sheet_id PLS_INTEGER;
     row_id PLS_INTEGER;
BEGIN
     doc_id := ORA_EXCEL.new_document;
     sheet_id := ORA_EXCEL.add_sheet('My sheet', doc_id);

     row_id := ORA_EXCEL.add_row(doc_id, sheet_id);
     ORA_EXCEL.set_cell_value('A', 'Left indented text',
                               doc_id, sheet_id, row_id);
     ORA_EXCEL.set_cell_indent_left('A', 5,
                                      doc_id, sheet_id, row_id);

     ORA_EXCEL.save_to_file('EXPORT_DIR', 'example.xlsx', doc_id);

END;
```

Output:



Left indented text

Procedure ORA_EXCEL. set_cell_indent_right

Description:

```
ORA_EXCEL.set_cell_indent_right(name VARCHAR2,
                                indent INTEGER,
[doc_id PLS_INTEGER DEFAULT current_doc_id,
sheet_id PLS_INTEGER DEFAULT current_sheet_id,
row_id PLS_INTEGER DEFAULT current_row_id])
```

Sets right indent within the cell

Mandatory parameters:

- name - name of cell content will be indented from the right side
- indent - number of indent from right site of cell

Optional parameters:

- doc_id - id of dpcument
- sheet_id - id of sheet
- row_id - id of row

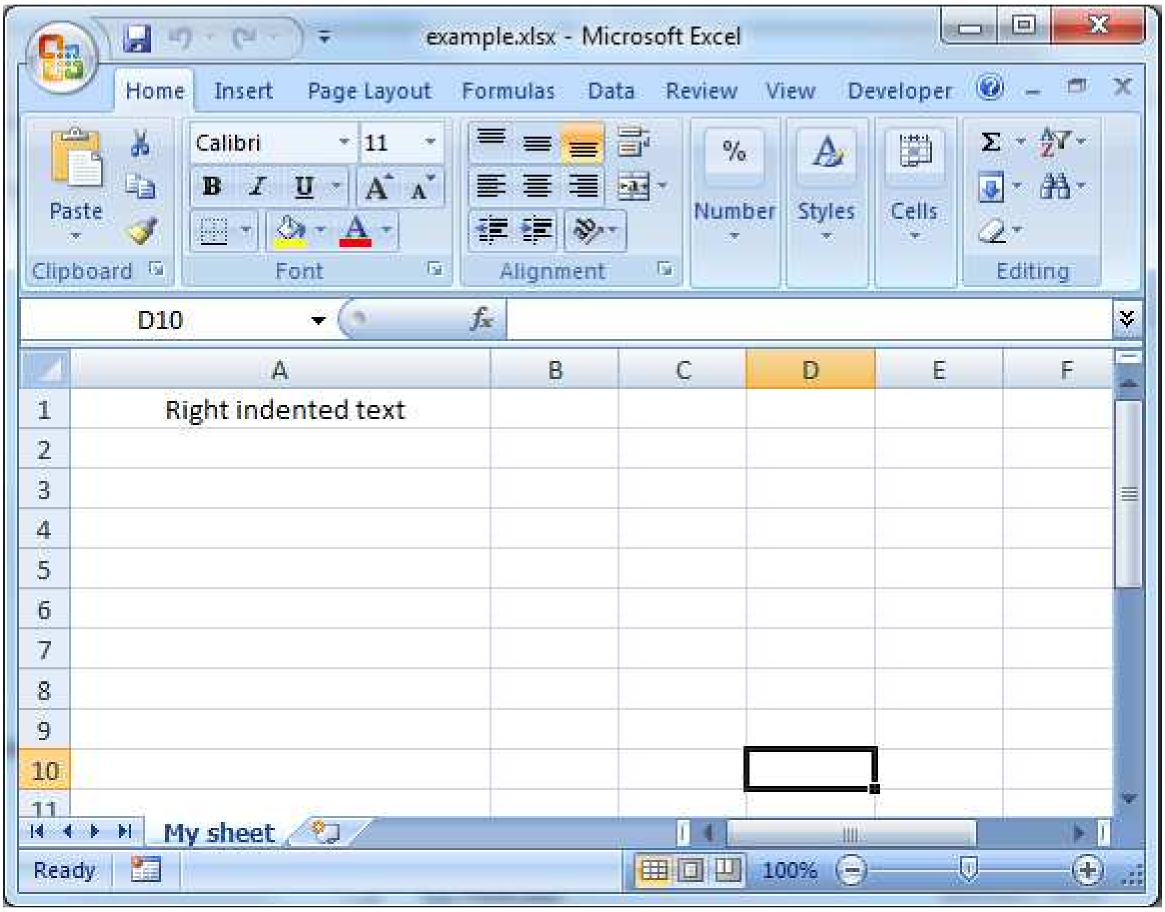
Returns:

Procedure, does not return any value

Example:

```
BEGIN
    ORA_EXCEL.new_document;

    ORA_EXCEL.add_sheet('My sheet');

    ORA_EXCEL.add_row;
    ORA_EXCEL.set_cell_value('A', 'Right indented text');
    ORA_EXCEL.set_cell_indent_right('A', 5);

    ORA_EXCEL.save_to_file('EXPORT_DIR', 'example.xlsx');

END;
```

Output:

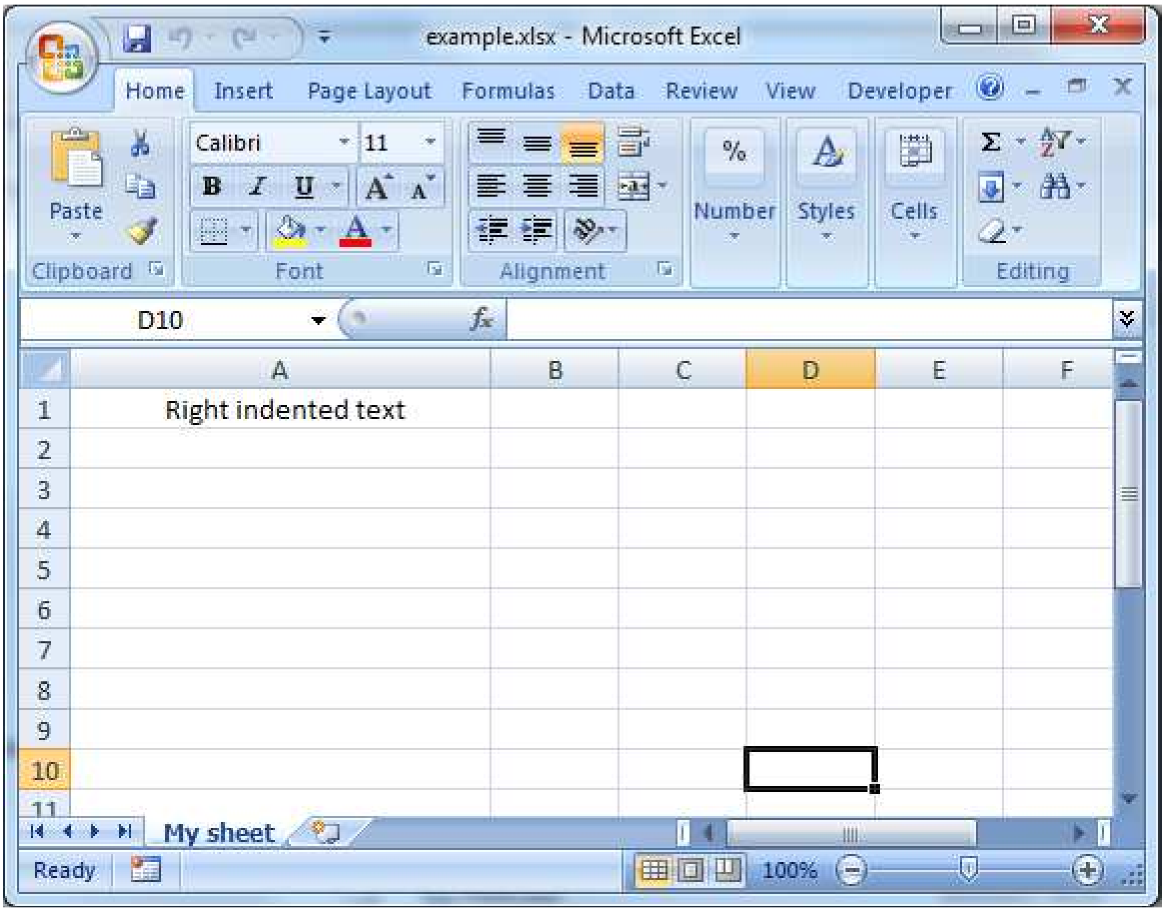


Right indented text

Example:

```
DECLARE
     doc_id PLS_INTEGER;
     sheet_id PLS_INTEGER;
     row_id PLS_INTEGER;
BEGIN
     doc_id := ORA_EXCEL.new_document;
     sheet_id := ORA_EXCEL.add_sheet('My sheet', doc_id);

     row_id := ORA_EXCEL.add_row(doc_id, sheet_id);
     ORA_EXCEL.set_cell_value('A', 'Right indented text',
                               doc_id, sheet_id, row_id);
     ORA_EXCEL.set_cell_indent_right('A', 5, doc_id, sheet_id, row_id);

     ORA_EXCEL.save_to_file('EXPORT_DIR', 'example.xlsx', doc_id);

END;
```

Output:



Right indented text

Procedure ORA_EXCEL.set_cell_comment

Description:

```
ORA_EXCEL. set_cell_comment(name VARCHAR2,
                             autohr VARCHAR2,
                             comment_text VARCHAR2,
                             comment_box_width NUMBER DEFAULT 100,
                             comment_box_height NUMBER DEFAULT 60,
                             [doc_id PLS_INTEGER DEFAULT current_doc_id,
                             sheet_id PLS_INTEGER DEFAULT current_sheet_id,
                             row_id PLS_INTEGER DEFAULT current_row_id)
```

Sets right indent within the cell

Mandatory parameters:

- name - cell name
- author - name of the autor of the comment
- comment_text - comment text for the cell
- comment_box_width - width of comment box
- comment_box_height - height of comment box

Optional parameters:

- doc_id - id of dpcument
- sheet_id - id of sheet
- row_id - id of row

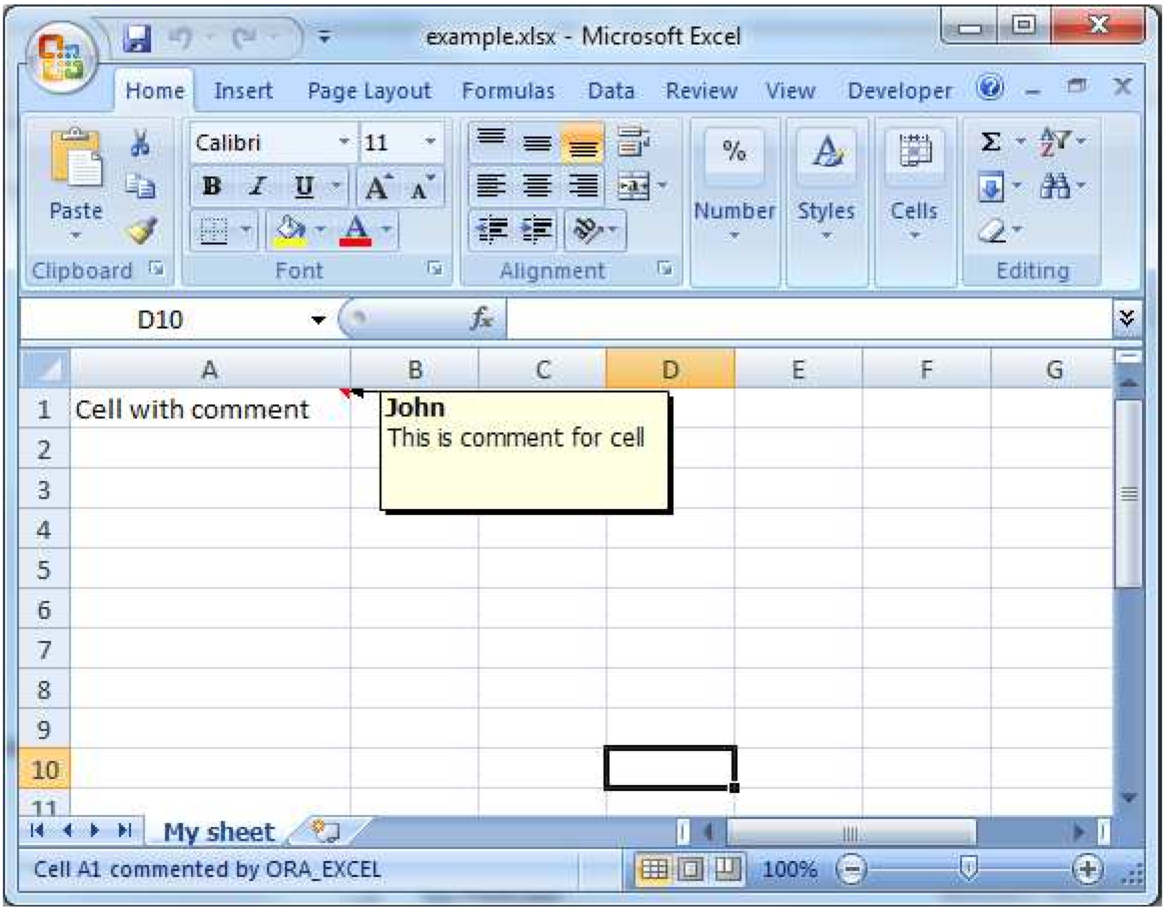
Returns:

Procedure, does not return any value

Example:

```
BEGIN
     ORA_EXCEL.new_document;

     ORA_EXCEL.add_sheet('My sheet');


     ORA_EXCEL.add_row;
     ORA_EXCEL.set_cell_value('A', 'Cell with comment');
     ORA_EXCEL.set_column_width('A', 20);
     ORA_EXCEL.set_cell_comment('A', 'John', 'This is comment for cell', 100, 50);

     ORA_EXCEL.save_to_file('EXPORT_DIR', 'example.xlsx');

END;
```

Output:



Cell with comment

John
This is comment for cell

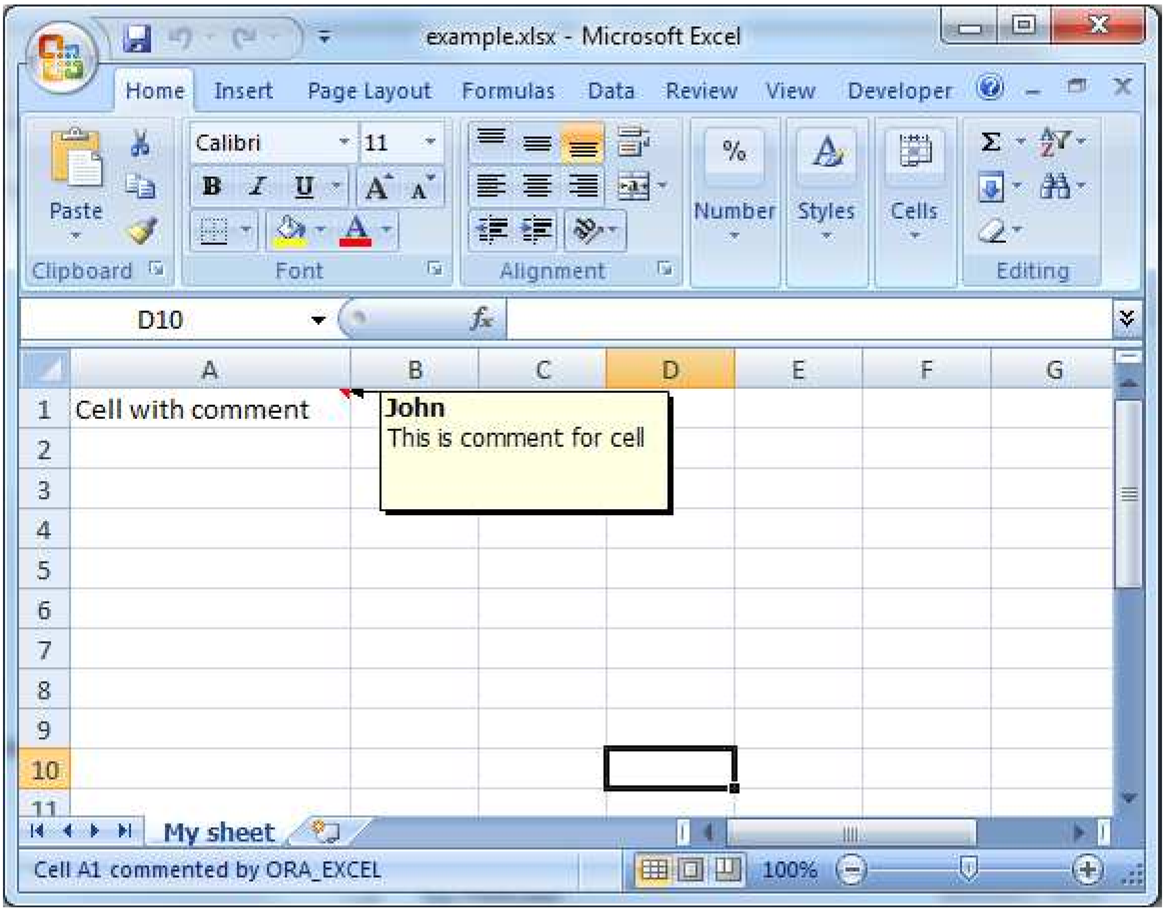
Cell A1 commented by ORA_EXCEL

Example:

```
DECLARE
```

```
    doc_id PLS_INTEGER;
    sheet_id PLS_INTEGER;
    row_id PLS_INTEGER;
BEGIN
    doc_id := ORA_EXCEL.new_document;
    sheet_id := ORA_EXCEL.add_sheet('My sheet', doc_id);

    row_id := ORA_EXCEL.add_row(doc_id, sheet_id);
    ORA_EXCEL.set_cell_value('A', 'Cell with comment', doc_id, sheet_id, row_id);
    ORA_EXCEL.set_column_width('A', 20, doc_id, sheet_id);
    ORA_EXCEL.set_cell_comment('A', 'John', 'This is comment for cell', 100, 50,
                               doc_id, sheet_id, row_id);

    ORA_EXCEL.save_to_file('EXPORT_DIR', 'example.xlsx', doc_id);

END;
```

Output:



example.xlsx - Microsoft Excel

D10

	A	B	C	D	E	F	G
1	Cell with comment						

John
This is comment for cell

Cell A1 commented by ORA_EXCEL

Procedure ORA_EXCEL. hide_column

Description:

```
ORA_EXCEL. hide_column(name VARCHAR2,
                       doc_id PLS_INTEGER DEFAULT current_doc_id,
                       sheet_id PLS_INTEGER DEFAULT current_sheet_id)
```

Hides column

Mandatory parameters:

- name - name of the column which will be hidden

Optional parameters:

- doc_id - id of dpcument
- sheet_id - id of sheet

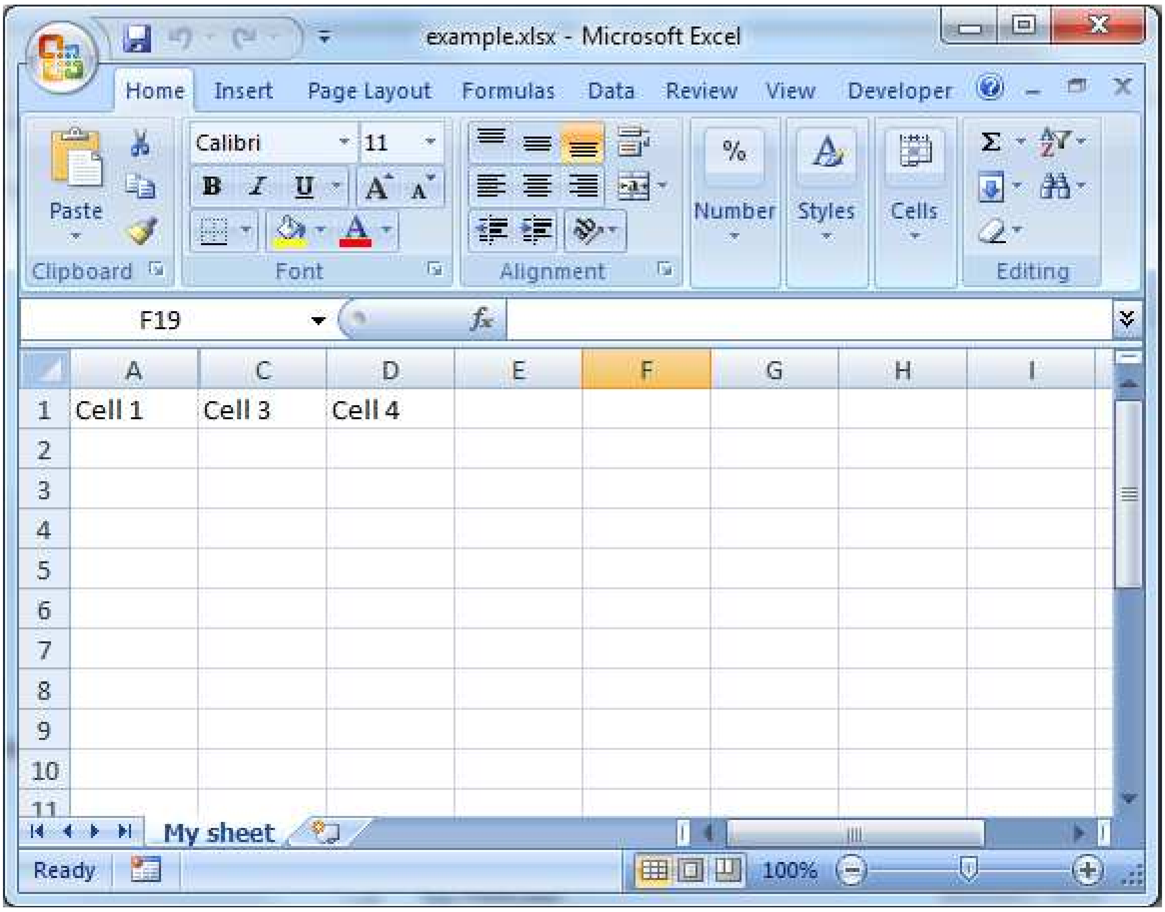
Returns:

Procedure, does not return any value

Example:

```
BEGIN
    ORA_EXCEL.new_document;

    ORA_EXCEL.add_sheet('My sheet');


    ORA_EXCEL.add_row;
    ORA_EXCEL.set_cell_value('A', 'Cell 1');
    ORA_EXCEL.set_cell_value('B', 'Cell 2');
    ORA_EXCEL.hide_column('B');
    ORA_EXCEL.set_cell_value('C', 'Cell 3');
    ORA_EXCEL.set_cell_value('D', 'Cell 4');

    ORA_EXCEL.save_to_file('EXPORT_DIR', 'example.xlsx');

END;
```

Output:

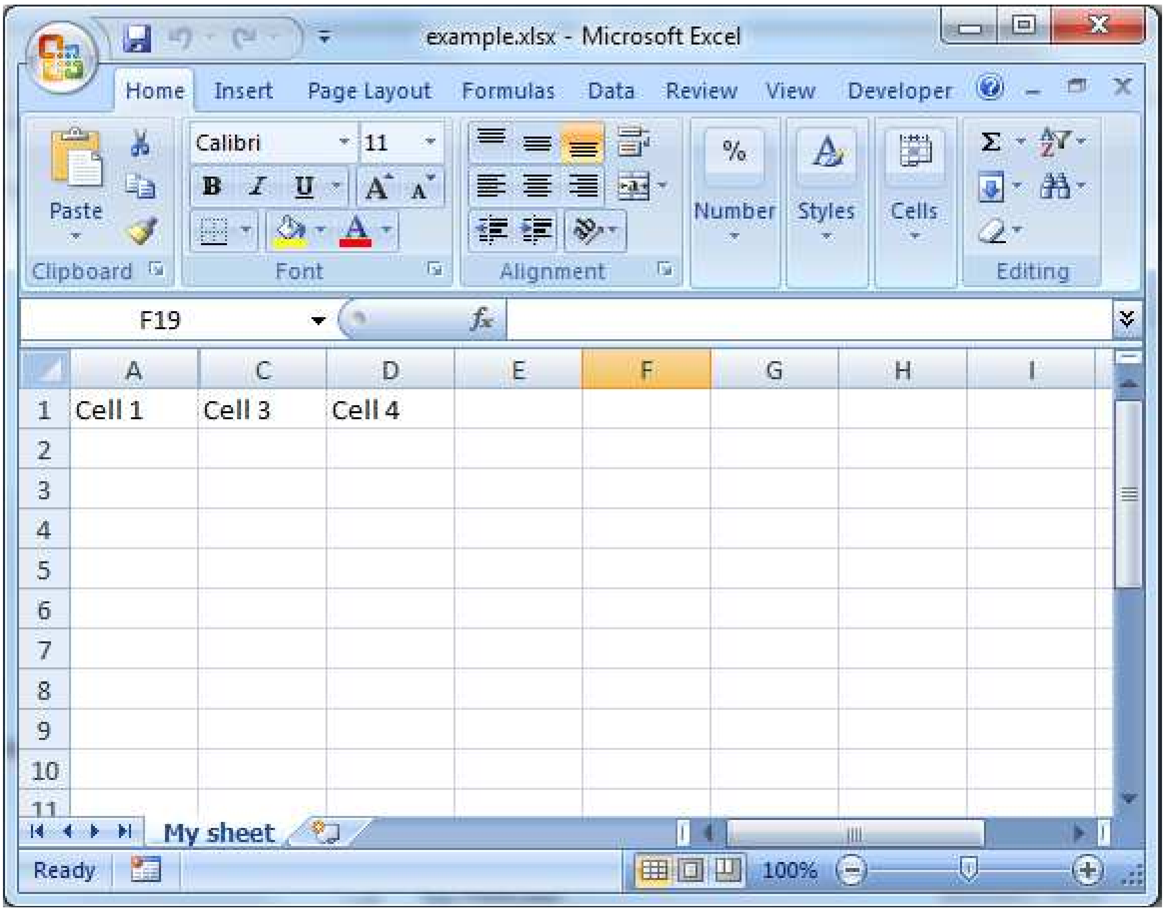


	A	C	D	E	F	G	H	I
1	Cell 1	Cell 3	Cell 4					

Example:

```
DECLARE
     doc_id PLS_INTEGER;
     sheet_id PLS_INTEGER;
     row_id PLS_INTEGER;
BEGIN
     doc_id := ORA_EXCEL.new_document;
     sheet_id := ORA_EXCEL.add_sheet('My sheet', doc_id);

     row_id := ORA_EXCEL.add_row(doc_id, sheet_id);
     ORA_EXCEL.set_cell_value('A', 'Cell 1', doc_id, sheet_id, row_id);
     ORA_EXCEL.set_cell_value('B', 'Cell 2', doc_id, sheet_id, row_id);
     ORA_EXCEL.hide_column('B', doc_id, sheet_id);
     ORA_EXCEL.set_cell_value('C', 'Cell 3', doc_id, sheet_id, row_id);
     ORA_EXCEL.set_cell_value('D', 'Cell 4', doc_id, sheet_id, row_id);

     ORA_EXCEL.save_to_file('EXPORT_DIR', 'example.xlsx', doc_id);

END;
```

Output:



	A	C	D	E	F	G	H	I
1	Cell 1	Cell 3	Cell 4					

Procedure ORA_EXCEL.hide_row

Description:

```
ORA_EXCEL.hide_row(doc_id PLS_INTEGER DEFAULT current_doc_id,
                   sheet_id PLS_INTEGER DEFAULT current_sheet_id,
                   row_id PLS_INTEGER DEFAULT current_row_id)
```

Hides row

Mandatory parameters:

- row_id - unique identificator of row which will be hidden

Optional parameters:

- doc_id - id of dpcument
- sheet_id - id of sheet

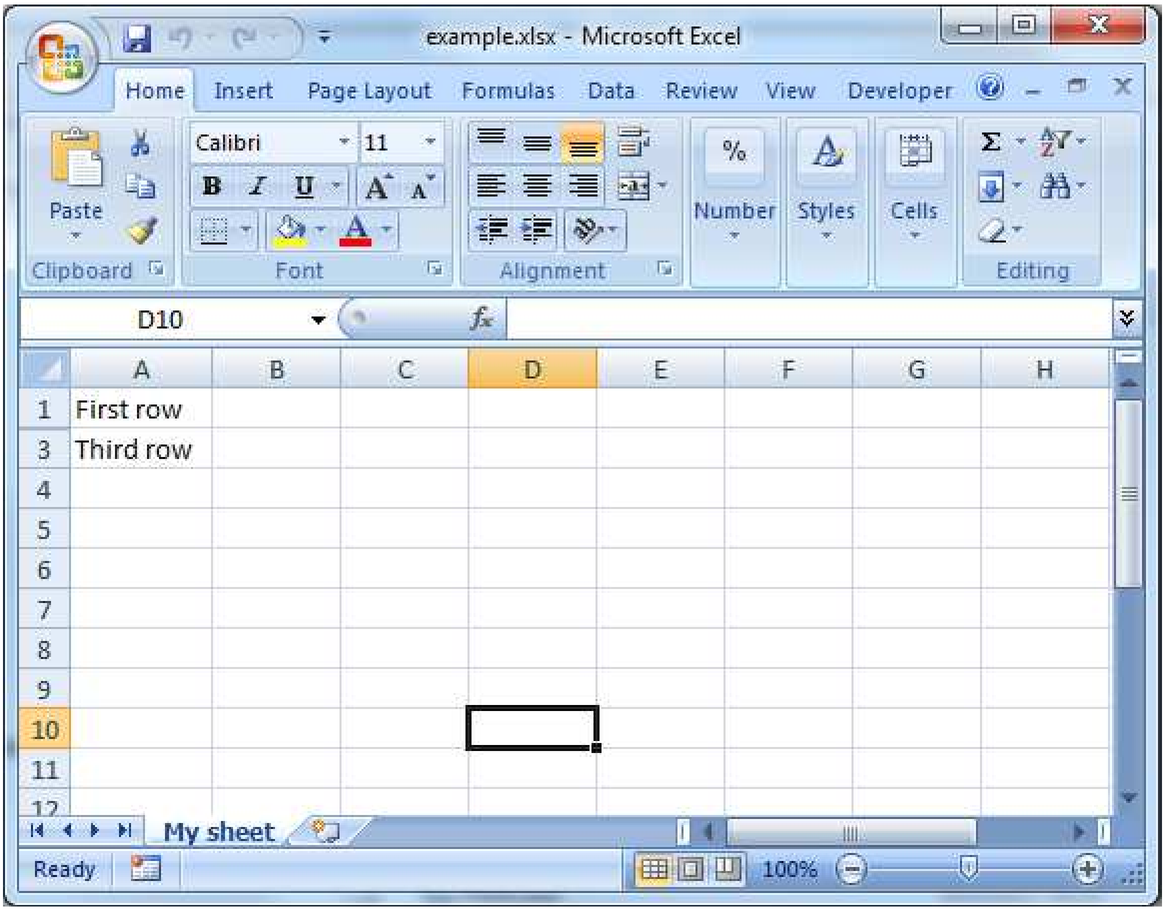
Returns:

Procedure, does not return any value

Example:

```
BEGIN
    ORA_EXCEL.new_document;
    ORA_EXCEL.add_sheet('My sheet');


    ORA_EXCEL.add_row;
    ORA_EXCEL.set_cell_value('A', 'First row');

    ORA_EXCEL.add_row;
    ORA_EXCEL.set_cell_value('A', 'Second row');
    ORA_EXCEL.hide_row;

    ORA_EXCEL.add_row;
    ORA_EXCEL.set_cell_value('A', 'Third row');

    ORA_EXCEL.save_to_file('EXPORT_DIR', 'example.xlsx');

END;
```

Output:

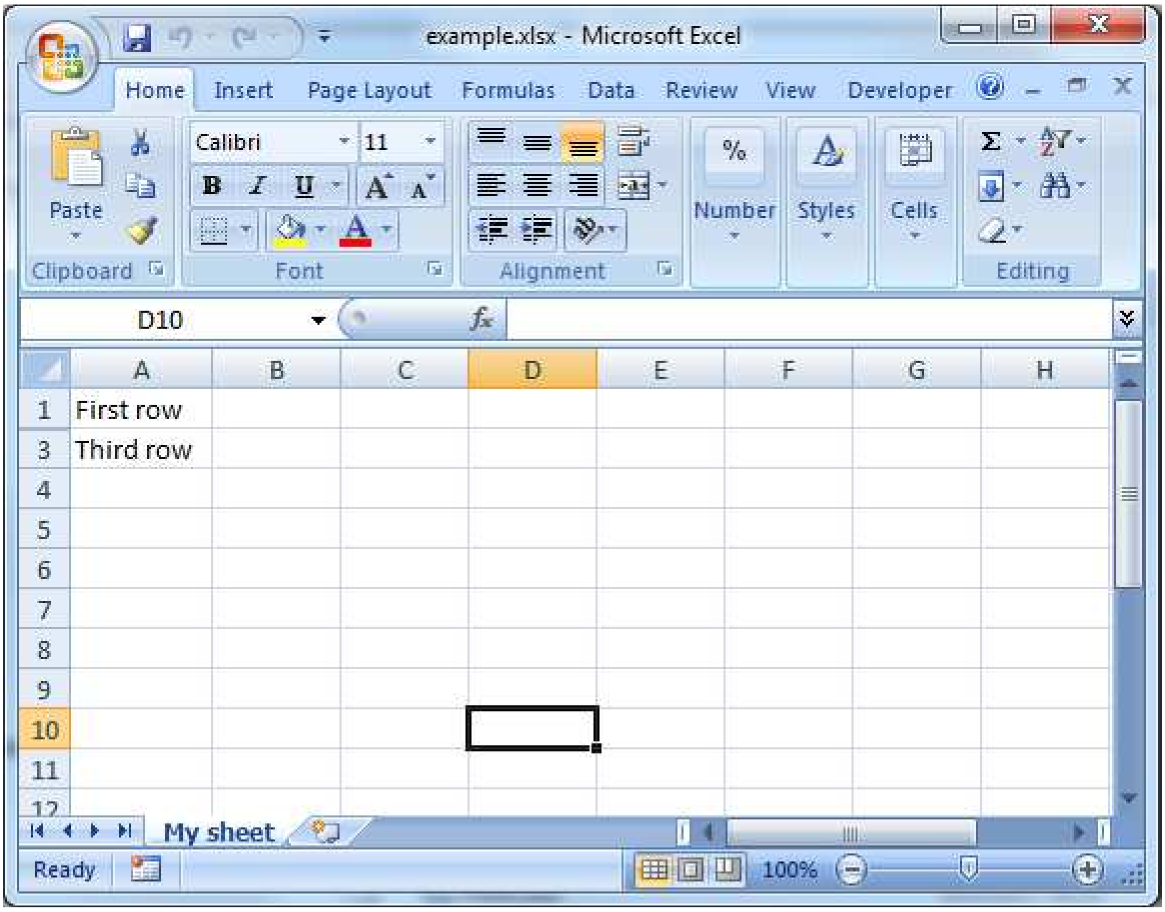


	A	B	C	D	E	F	G	H
1	First row							
3	Third row							
4								
5								
6								
7								
8								
9								
10								
11								
12								

Example:

```
DECLARE
    doc_id PLS_INTEGER;
    sheet_id PLS_INTEGER;
    row_id PLS_INTEGER;
BEGIN
    doc_id := ORA_EXCEL.new_document;
    sheet_id := ORA_EXCEL.add_sheet('My sheet', doc_id);
    row_id := ORA_EXCEL.add_row(doc_id, sheet_id);
    ORA_EXCEL.set_cell_value('A', 'First row', doc_id, sheet_id, row_id);

    row_id := ORA_EXCEL.add_row(doc_id, sheet_id);
    ORA_EXCEL.set_cell_value('A', 'Second row', doc_id, sheet_id, row_id);
    ORA_EXCEL.hide_row(doc_id, sheet_id, row_id);

    row_id := ORA_EXCEL.add_row(doc_id, sheet_id);
    ORA_EXCEL.set_cell_value('A', 'Third row', doc_id, sheet_id, row_id);

    ORA_EXCEL.save_to_file('EXPORT_DIR', 'example.xlsx', doc_id);

END;
```

Output:



	A	B	C	D	E	F	G	H
1	First row							
3	Third row							
4								
5								
6								
7								
8								
9								
10								
11								
12								

Procedure ORA_EXCEL.set_cells_filter

Description:

```
ORA_EXCEL. set_cells_filter(cell_from VARCHAR2,
                             cell_to VARCHAR2,
                             [doc_id PLS_INTEGER DEFAULT current_doc_id,
                             sheet_id PLS_INTEGER DEFAULT current_sheet_id])
```

Sets column auto filter between defined columns range

Mandatory parameters:

- cell_from - cell name with row number from which auto filter will start, example: A1
- cell_to - cell name with row number where auto filter will end, example: A5

Optional parameters:

- doc_id - id of dpcument
- sheet_id - id of sheet

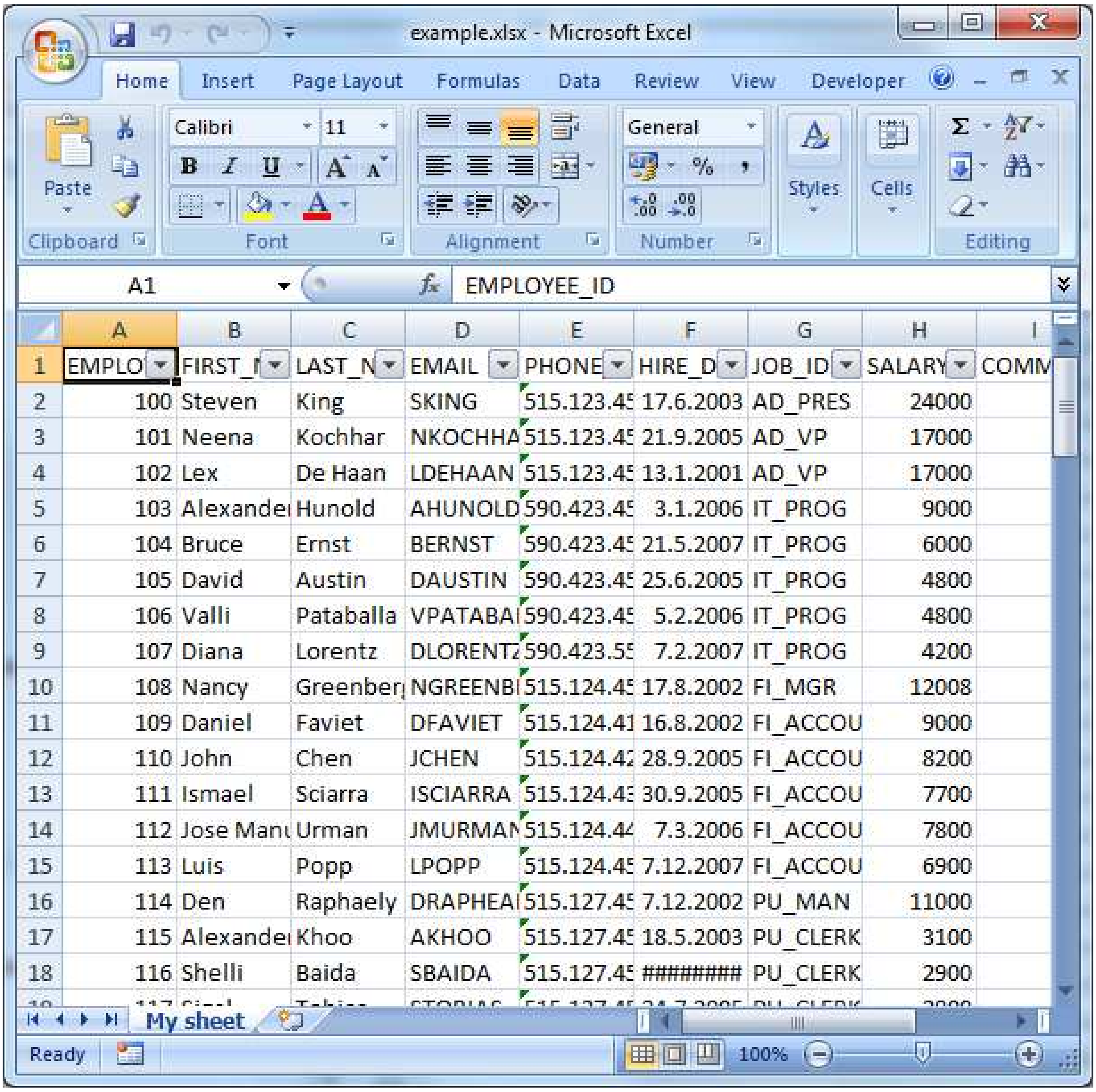
Returns:

Procedure, does not return any value

Example:

```
BEGIN
     ORA_EXCEL.new_document;

     ORA_EXCEL.add_sheet('My sheet');
     ORA_EXCEL.query_to_sheet('select * from employees');
     ORA_EXCEL.set_cells_filter('A1', 'K1');

     ORA_EXCEL.save_to_file('EXPORT_DIR', 'example.xlsx');

END;
```

Output:

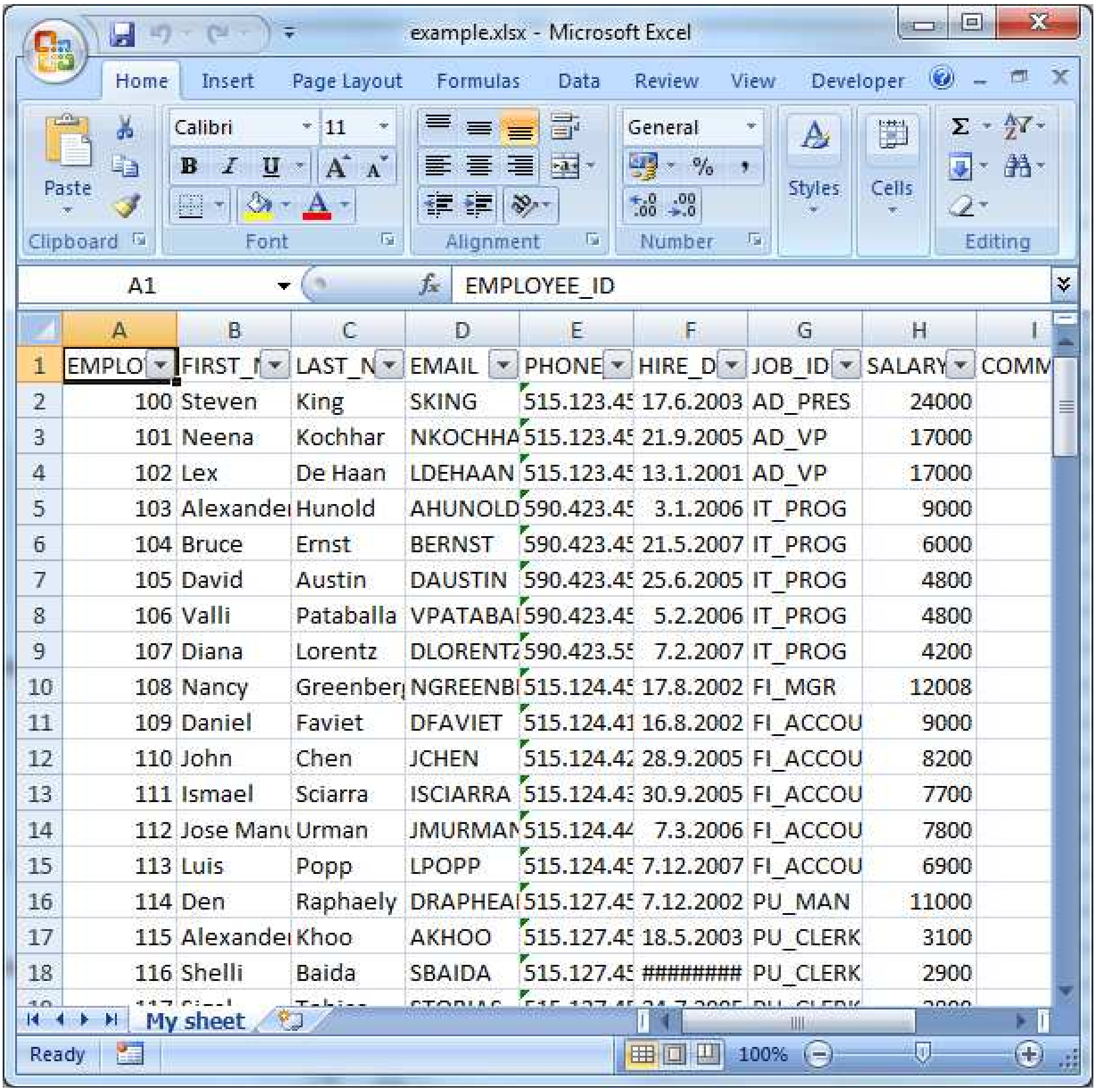


	A	B	C	D	E	F	G	H	I
1	EMPLO	FIRST_	LAST_N	EMAIL	PHONE	HIRE_D	JOB_ID	SALARY	COMM
2	100	Steven	King	SKING	515.123.45	17.6.2003	AD_PRES	24000	
3	101	Neena	Kochhar	NKOCHHA	515.123.45	21.9.2005	AD_VP	17000	
4	102	Lex	De Haan	LDEHAAN	515.123.45	13.1.2001	AD_VP	17000	
5	103	Alexande	Hunold	AHUNOLD	590.423.45	3.1.2006	IT_PROG	9000	
6	104	Bruce	Ernst	BERNST	590.423.45	21.5.2007	IT_PROG	6000	
7	105	David	Austin	DAUSTIN	590.423.45	25.6.2005	IT_PROG	4800	
8	106	Valli	Pataballa	VPATABA	590.423.45	5.2.2006	IT_PROG	4800	
9	107	Diana	Lorentz	DLORENT	590.423.55	7.2.2007	IT_PROG	4200	
10	108	Nancy	Greenber	NGREENB	515.124.45	17.8.2002	FI_MGR	12008	
11	109	Daniel	Faviet	DFAVIET	515.124.41	16.8.2002	FI_ACCOU	9000	
12	110	John	Chen	JCHEN	515.124.42	28.9.2005	FI_ACCOU	8200	
13	111	Ismael	Sciarra	ISCIARRA	515.124.43	30.9.2005	FI_ACCOU	7700	
14	112	Jose Man	Urman	JMURMAN	515.124.44	7.3.2006	FI_ACCOU	7800	
15	113	Luis	Popp	LPOPP	515.124.45	7.12.2007	FI_ACCOU	6900	
16	114	Den	Raphaely	DRAPHEA	515.127.45	7.12.2002	PU_MAN	11000	
17	115	Alexande	Khoo	AKHOO	515.127.45	18.5.2003	PU_CLERK	3100	
18	116	Shelli	Baida	SBAIDA	515.127.45	########	PU_CLERK	2900	

Example:

```
DECLARE
     doc_id PLS_INTEGER;
     sheet_id PLS_INTEGER;
BEGIN
     doc_id := ORA_EXCEL.new_document;
     sheet_id := ORA_EXCEL.add_sheet('My sheet', doc_id);

     ORA_EXCEL.query_to_sheet('select * from employees', TRUE, doc_id, sheet_id);
     ORA_EXCEL.set_cells_filter('A1', 'K1', doc_id, sheet_id);

     ORA_EXCEL.save_to_file('EXPORT_DIR', 'example.xlsx', doc_id);

END;
```

Output:



A1 *fx* EMPLOYEE_ID

	A	B	C	D	E	F	G	H	I
1	EMPLO	FIRST_N	LAST_N	EMAIL	PHONE	HIRE_D	JOB_ID	SALARY	COMM
2	100	Steven	King	SKING	515.123.45	17.6.2003	AD_PRES	24000	
3	101	Neena	Kochhar	NKOCHHA	515.123.45	21.9.2005	AD_VP	17000	
4	102	Lex	De Haan	LDEHAAN	515.123.45	13.1.2001	AD_VP	17000	
5	103	Alexande	Hunold	AHUNOLD	590.423.45	3.1.2006	IT_PROG	9000	
6	104	Bruce	Ernst	BERNST	590.423.45	21.5.2007	IT_PROG	6000	
7	105	David	Austin	DAUSTIN	590.423.45	25.6.2005	IT_PROG	4800	
8	106	Valli	Pataballa	VPATABA	590.423.45	5.2.2006	IT_PROG	4800	
9	107	Diana	Lorentz	DLORENTZ	590.423.55	7.2.2007	IT_PROG	4200	
10	108	Nancy	Greenber	NGREENB	515.124.45	17.8.2002	FI_MGR	12008	
11	109	Daniel	Faviet	DFAVIET	515.124.41	16.8.2002	FI_ACCOU	9000	
12	110	John	Chen	JCHEN	515.124.42	28.9.2005	FI_ACCOU	8200	
13	111	Ismael	Sciarra	ISCIARRA	515.124.43	30.9.2005	FI_ACCOU	7700	
14	112	Jose Manu	Urman	JMURMAN	515.124.44	7.3.2006	FI_ACCOU	7800	
15	113	Luis	Popp	LPOPP	515.124.45	7.12.2007	FI_ACCOU	6900	
16	114	Den	Raphaely	DRAPHEA	515.127.45	7.12.2002	PU_MAN	11000	
17	115	Alexande	Khoo	AKHOO	515.127.45	18.5.2003	PU_CLERK	3100	
18	116	Shelli	Baida	SBAIDA	515.127.45	########	PU_CLERK	2900	

Procedure ORA_EXCEL.save_to_file

Description:

ORA_EXCEL.save_to_file(directory_name varchar2, file_name varchar2 [, doc_id pls_integer])

Save generated Excel document to file

Mandatory parameters:

- directory_name – Oracle directory
- file_name – file name (example my_document.xlsx)

Optional parameters:

- doc_id – document id

- sheet_id – sheet id
- row_id – row id

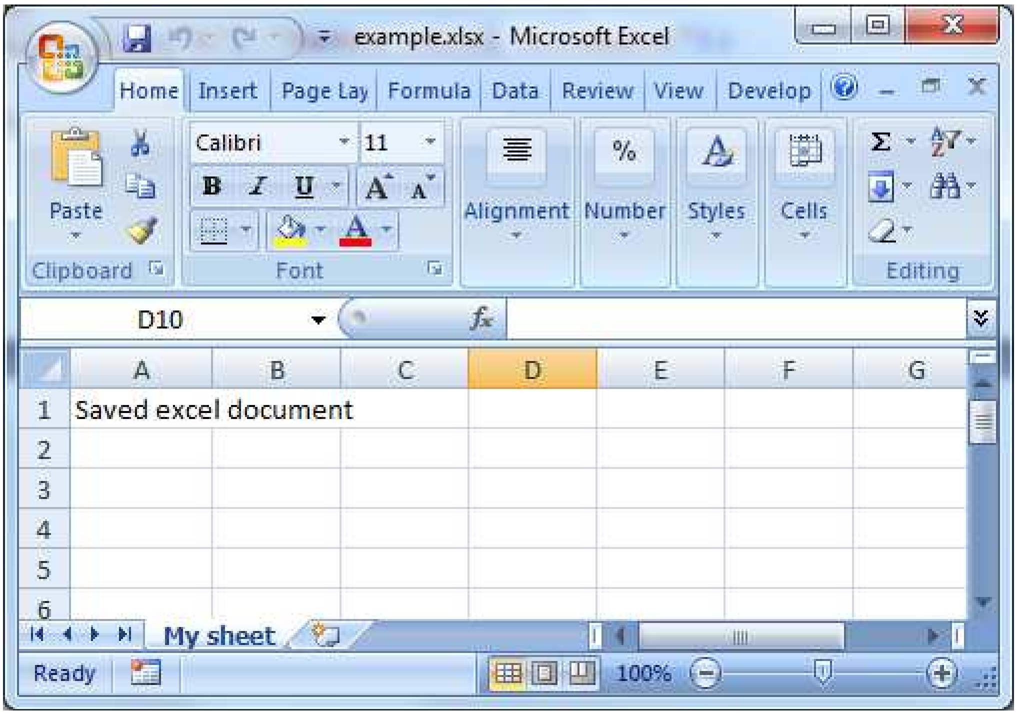
Returns:

Procedure, does not return any value

Example:

```
BEGIN
     ORA_EXCEL.new_document;
     ORA_EXCEL.add_sheet('My sheet');

     ORA_EXCEL.add_row;
     ORA_EXCEL.set_cell_value('A', 'Saved excel document');

     ORA_EXCEL.save_to_file('EXPORT_DIR', 'example.xlsx');
END;
```

Output:



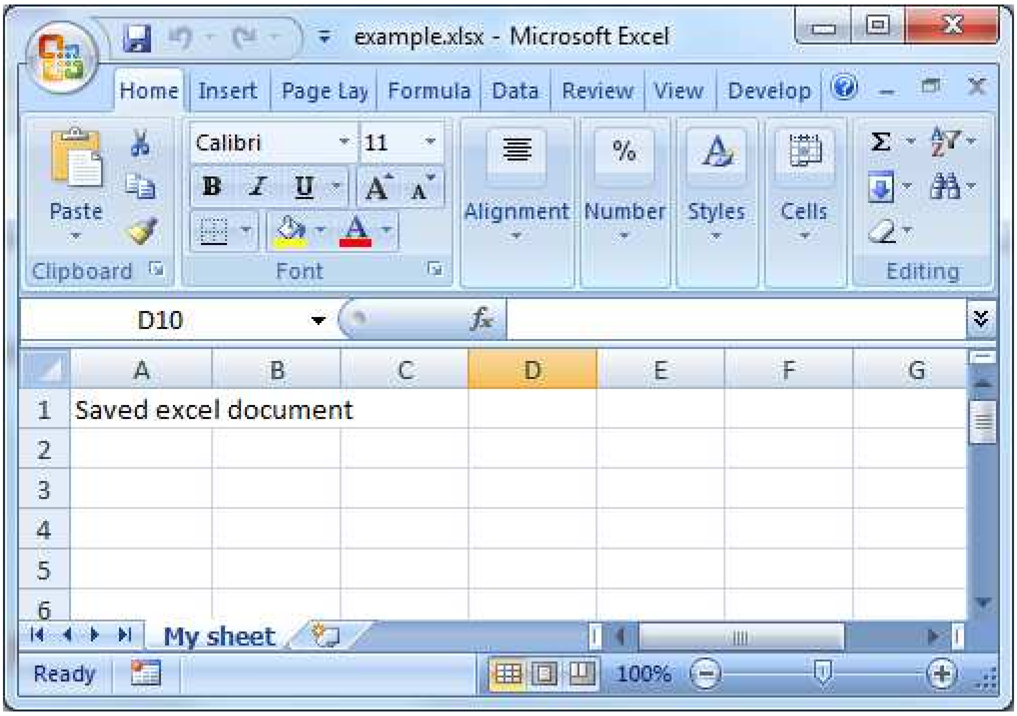
D10

Saved excel document

Example:

```
DECLARE
     doc_id PLS_INTEGER;
     sheet_id PLS_INTEGER;
     row_id PLS_INTEGER;
BEGIN
     doc_id := ORA_EXCEL.new_document;
     sheet_id := ORA_EXCEL.add_sheet('My sheet', doc_id);

     row_id := ORA_EXCEL.add_row(doc_id, sheet_id);
     ORA_EXCEL.set_cell_value('A', 'Saved excel document', doc_id, sheet_id,
row_id);

     ORA_EXCEL.save_to_file('EXPORT_DIR', 'example.xlsx', doc_id);
END;
```

Output:



	A	B	C	D	E	F	G
1	Saved excel document						
2							
3							
4							
5							
6							

Procedure ORA_EXCEL.save_to_blob

Description:

ORA_EXCEL.save_to_blob(blob_file in out blob [, doc_id pls_integer])

Save generated Excel document to PL/SQL BLOB variable type

Mandatory parameters:

- blob_file – PL/SQL BLOB variable where binary content of Excel worksheet will be stored

Optional parameters:

- doc_id – document id

Returns:

Procedure, does not return any value

Example:

```
DECLARE
      generated_excel BLOB;
BEGIN
      ORA_EXCEL.new_document;
      ORA_EXCEL.add_sheet('My sheet');

      ORA_EXCEL.add_row;
      ORA_EXCEL.set_cell_value('A', 'Saved excel document');
```

```
     ORA_EXCEL.save_to_blob(generated_excel);

     -- Excel document is stored into generated_excel variable
END;
```

Example:

```
DECLARE
     doc_id PLS_INTEGER;
     sheet_id PLS_INTEGER;
     row_id PLS_INTEGER;
     generated_excel BLOB;
BEGIN
     doc_id := ORA_EXCEL.new_document;
     sheet_id := ORA_EXCEL.add_sheet('My sheet', doc_id);

     row_id := ORA_EXCEL.add_row(doc_id, sheet_id);
     ORA_EXCEL.set_cell_value('A', 'Saved excel document', doc_id, sheet_id,
     row_id);

     ORA_EXCEL.save_to_blob(generated_excel, doc_id);
     -- Excel document is stored into generated_excel variable
END;
```

Procedure ORA_EXCEL.set_1904_date_system

Description:

ORA_EXCEL.set_1904_date_system

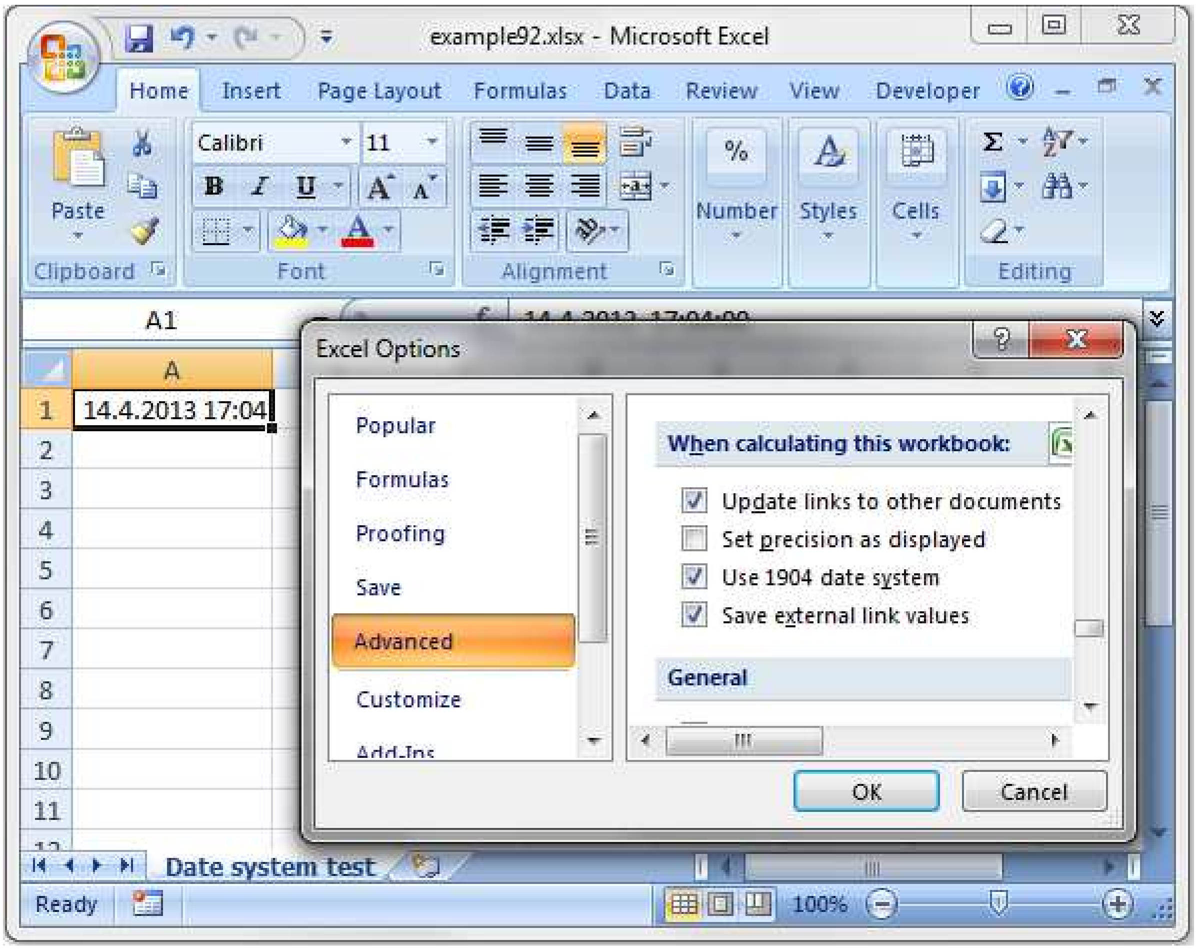
Set generated Excel document date setting to use January 1st of 1904 as start year

Returns:

Procedure, does not return any value

Example:

```
BEGIN

        ORA_EXCEL.new_document;
        ORA_EXCEL.set_1904_date_system;

        ORA_EXCEL.add_sheet('Date system test');

        ORA_EXCEL.add_row;
        ORA_EXCEL.set_cell_value('A', SYSDATE);

        dbms_output.put_line(TO_CHAR(SYSDATE, 'dd.mm.yyyy hh24:mi:ss'));

        ORA_EXCEL.save_to_file('EXPORT_DIR', 'example92.xlsx');

END;
```



A1: 14.4.2013 17:04

When calculating this workbook:

☑ Update links to other documents
☐ Set precision as displayed
☑ Use 1904 date system
☑ Save external link values

General

Procedure ORA_EXCEL.set_1900_date_system

Description:

ORA_EXCEL.set_1904_date_system

Set generated Excel document date setting to use January 1st of 1900 as start year

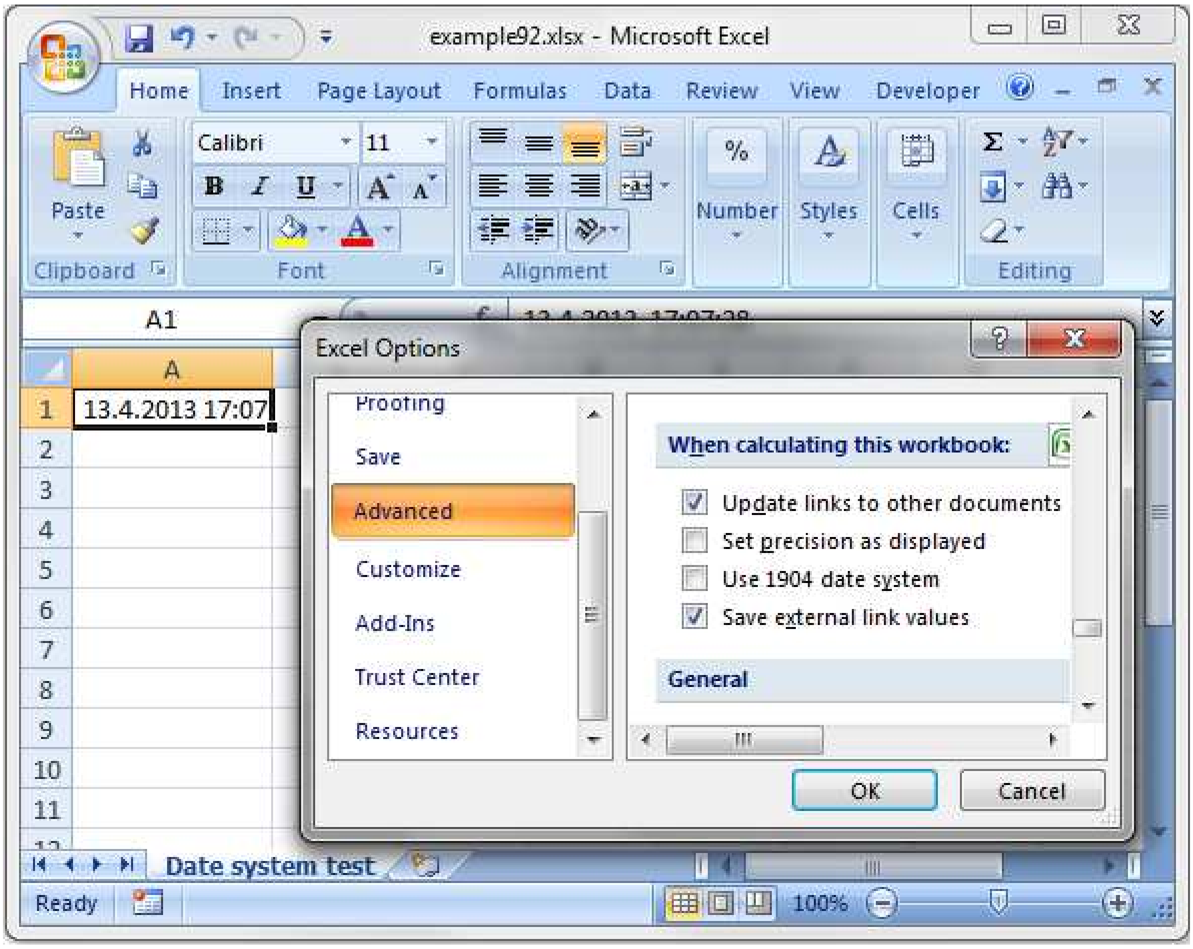
Returns:

Procedure, does not return any value

Example:

```
BEGIN

    ORA_EXCEL.new_document;
    ORA_EXCEL.set_1900_date_system;

    ORA_EXCEL.add_sheet('Date system test');

    ORA_EXCEL.add_row;
    ORA_EXCEL.set_cell_value('A',  SYSDATE);

    dbms_output.put_line(TO_CHAR(SYSDATE,  'dd.mm.yyyy hh24:mi:ss'));

    ORA_EXCEL.save_to_file('EXPORT_DIR',  'example92.xlsx');
```

```
END;
```



When calculating this workbook:

☑ Update links to other documents
☐ Set precision as displayed
☐ Use 1904 date system
☑ Save external link values