

# Data Base

Abdalwahab

Qatawneh

## Record:

T3
T3
T3
T3
T3
T3
T3
T3
T3
T3

Record

Record

Record

Record

Record

Record

Record

Record

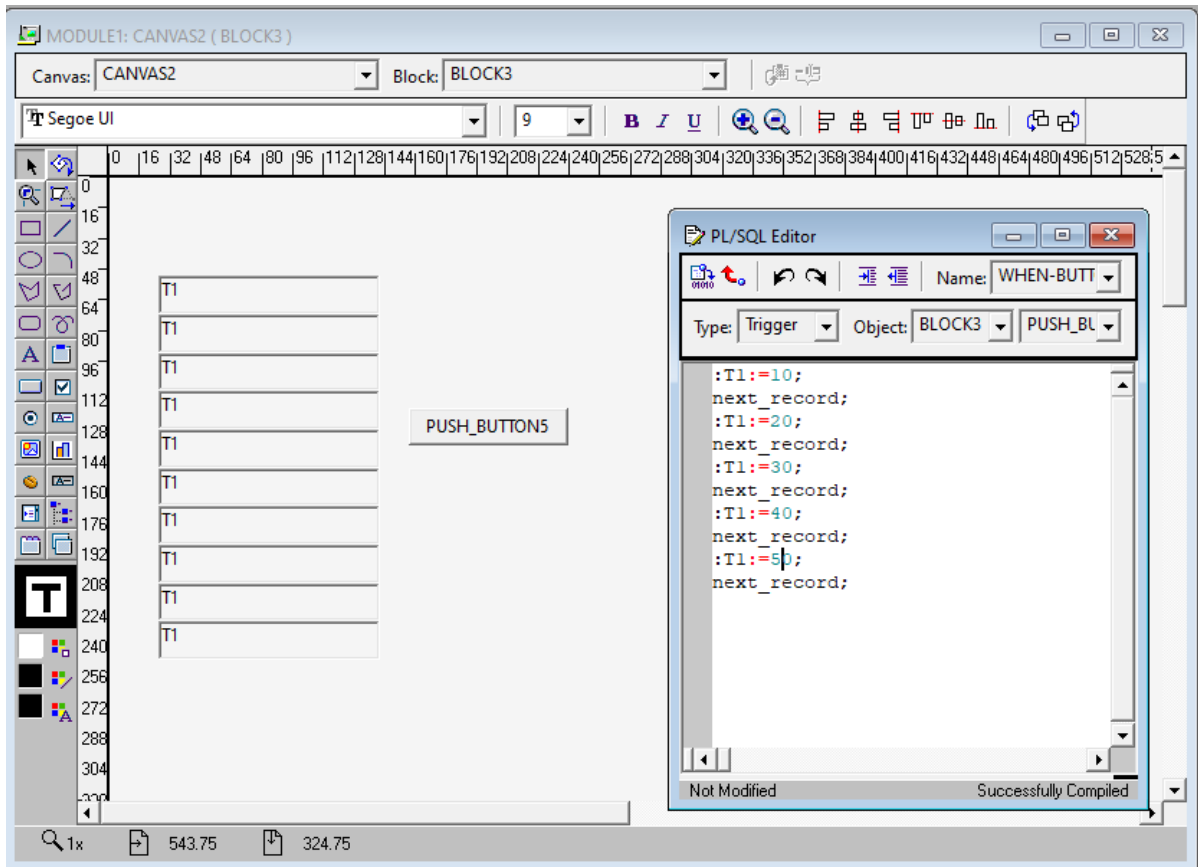
Record

1. Alert
2. LOVs
3. Loops (for, while and loop)
4. List item
5. For select.
6. For cursor.
7. Cursor.
8. Functions.
9. Procedures.

- Next\_record

هو ينقل جميع العناصر الموجودات بنفس البلوك المستخدم فيه next\_record الى record التالي

Example:

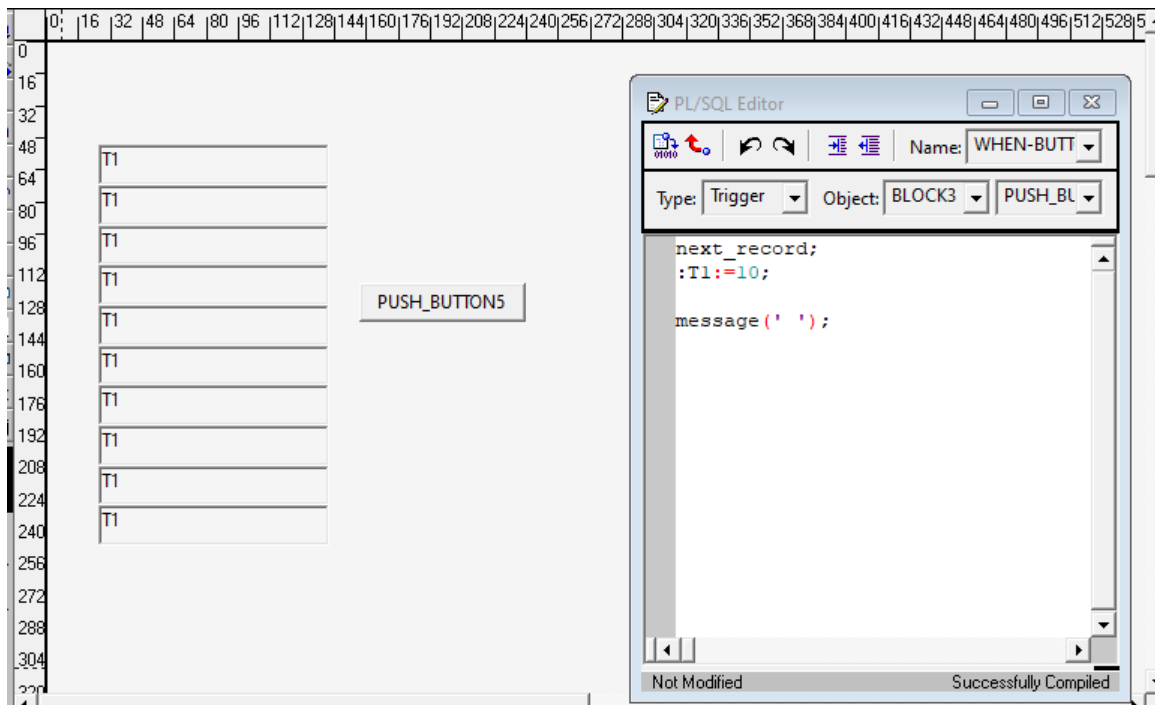


Run:

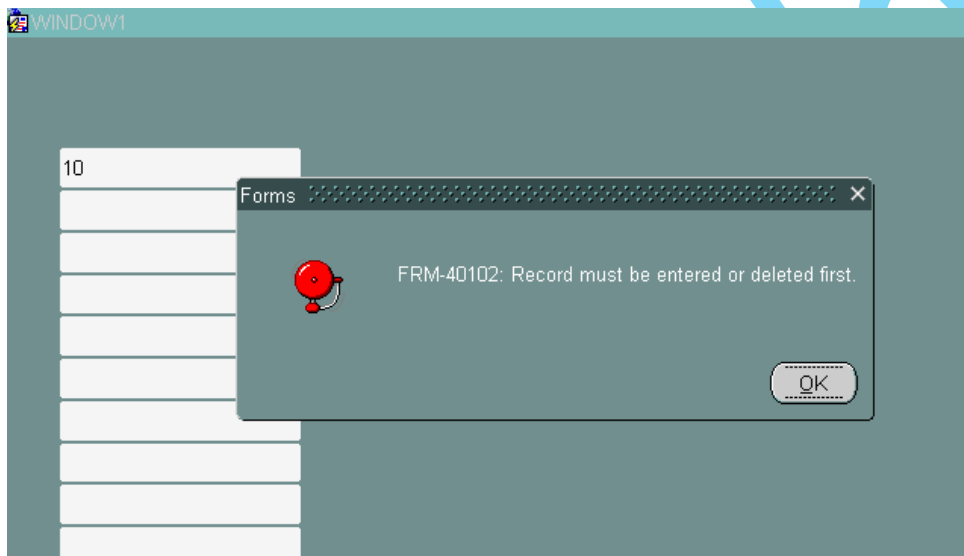
Form Elements:

- 10
- 20
- 30
- 40
- 50
- PUSH\_BUTTON5

When-Mouse-DoubleClick



Run:



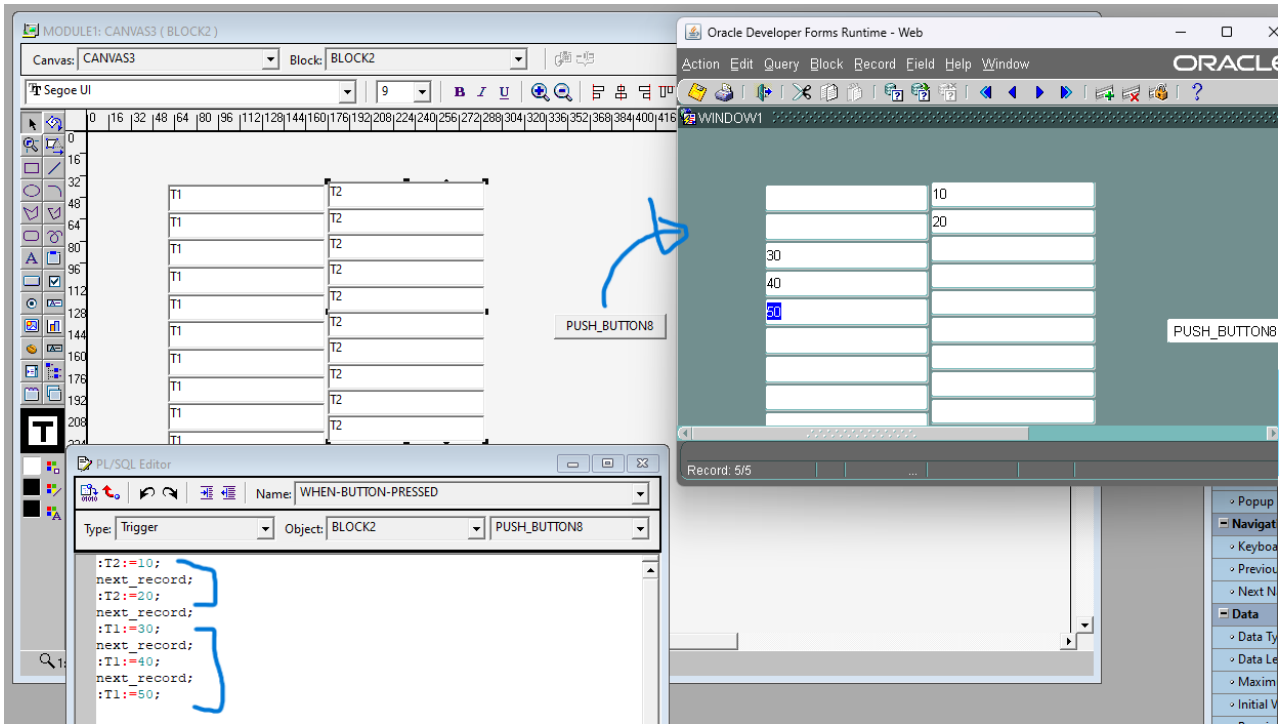
عند عدم استخدام record من ثم الانتقال إلى record next يحذف record الذي لم يتم استعماله



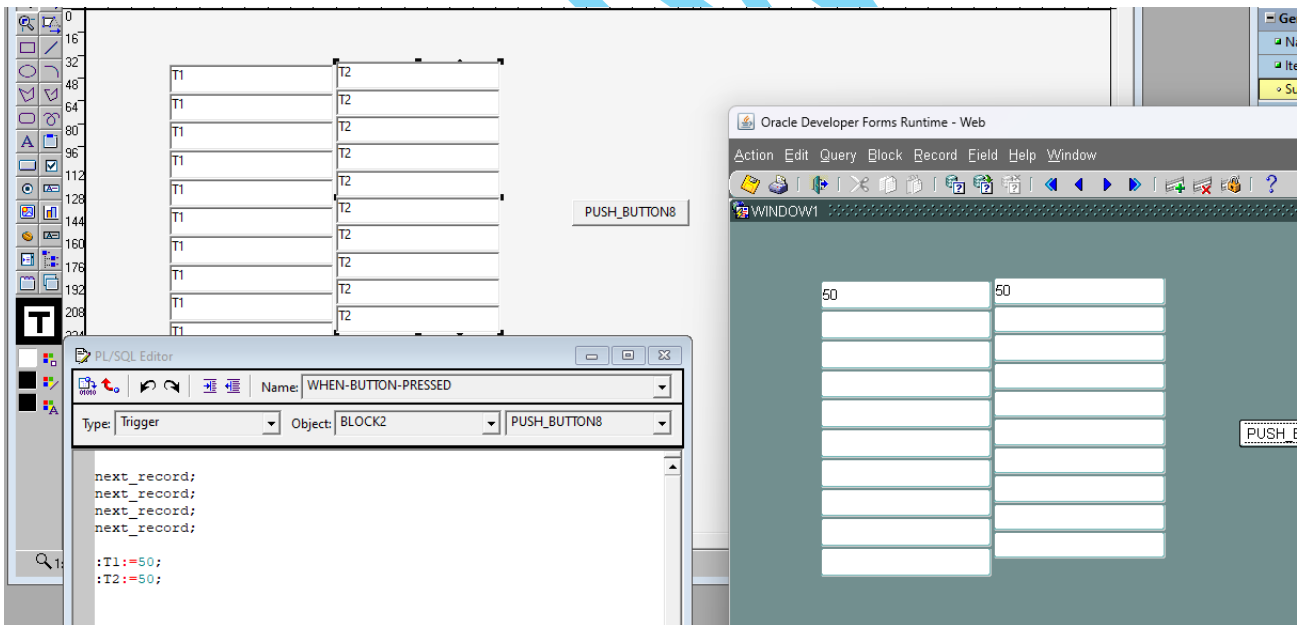
Tow record in same block:



Example:



Example:



Example:

T1	T2
T1	
T1	
T1	
T1	
T1	
T1	
T1	
T1	
T1	

```
PL/SQL Editor
Name: WHEN-MOUSE-DOUBLECLICK
Type: Trigger Object: BLOCK4 T1

:T1:=100;
next_record;
:T1:=100;
next_record;
:T1:=100;
next_record;
:T1:=100;
next_record;
:T1:=100;
next_record;
:T1:=100;
next_record;
:T2:=9000;
```

100	9000
100	
100	
100	
100	
100	

100	
100	
100	
100	
100	
100	

**Solution:**

**Add T2 in another block.**

# LOOP:

1. Basic loop
2. While loop
3. For loop

Loop

--statement

exit when(condition)

End loop;

Example:

The screenshot shows a PL/SQL Editor window with the following code:

```
declare
  counter number(10) := 0;
begin
  :T2 := ' ';

  loop
    counter := counter + 1;

    :T2 := :T2 || counter || ',';

    exit when (counter = 10);
  end loop;
  message('/ ');
end;
```

The status bar at the bottom of the editor window shows "Not Modified" and "Successfully Compiled".

Run:

The screenshot shows a web application interface with a text input field containing the sequence "1,2,3,4,5,6,7,8,9,10," and a button labeled "LOOP".

While loop:

While (condition)

Loop

--statement

End loop;

PL/SQL Editor

Name: WHEN-BUTTON-PRESSED

Type: Trigger Object: BLOCK2 PUSH\_BUTTON

```
declare
  i number(10):=0;
begin
  :T1:=' ';
  while(i<10)
  loop
    i:=i+1;
    :T1:=:T1||i||',';
  end loop;
end;
```

Not Modified Successfully Compiled

LOOP

While loop

loop record

T3

T3

T3

T3

T3

T3

T3

T3

T3

T3

T3

**Run:**

1,2,3,4,5,6,7,8,9,10,

While loop



## Example (using Record):

The screenshot displays a web application interface on the left and a PL/SQL Editor window on the right.

**Web Application Interface:**

- Input fields: T2, T1, and a vertical stack of 10 T3 fields.
- Buttons: "LOOP", "While loop", and "loop record".

**PL/SQL Editor:**

- Title: PL/SQL Editor
- Name: WHEN-BUTTON-PRESSED
- Type: Trigger
- Object: BLOCK2
- Action: PUSH\_BUTTON
- Code:

```
declare
  i number(10) := 0;
begin
  while (i < 10)
  loop
    i := i + 1;

    :T3 := i;
    next_record;

  end loop;
end;
```

Modified Not Compiled

Run:

The screenshot shows the web application after the "loop record" button has been clicked. The T3 input fields are now populated with numbers 1 through 10. The "loop record" button is still visible.

Solution: Same the block (next-item)

## Example:

### 1. Basic loop

```
declare
  counter number(10):=0;
  Result number;
begin
  loop
    counter := counter+1;
    message('8 *'|| counter|| '='|| 8*counter);
    exit when(counter=10);
  end loop;

  message('Hello');
end;
```

### 2. While loop

```
declare
  counter number(10):=0;
  Result number;

  v_text boolean:=true;
begin
  while (counter<10) loop
    counter := counter+1;
    message('8 *'|| counter|| '='|| 8*counter);
  end loop;

  message('Hello');
end;
```

```
declare
  counter number(10):=0;
  Result number;

  v_text boolean:=true;
begin
  while v_text loop
    counter := counter+1;
    message('8 *'|| counter|| '='|| 8*counter);
    if(counter=10)then
      v_text:=false;
    end if;
  end loop;

  message('Hello');
end;
```

\*8 -> record

**For loop:**

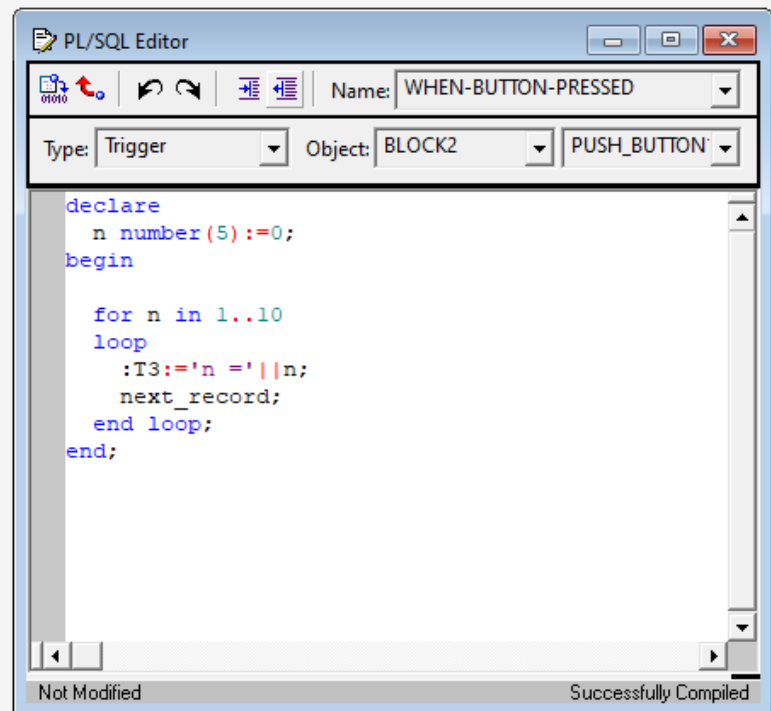
***For N in 1..10***

***Loop***

***End loop;***

T3
T3
T3
T3
T3
T3
T3
T3
T3
T3

loop record



```
PL/SQL Editor
Name: WHEN-BUTTON-PRESSED
Type: Trigger Object: BLOCK2 PUSH_BUTTON

declare
  n number(5) := 0;
begin

  for n in 1..10
  loop
    :T3:='n = ' || n;
    next_record;
  end loop;
end;
```

Not Modified Successfully Compiled

n =1
n =2
n =3
n =4
n =5
n =6
n =7
n =8
n =9
n =10

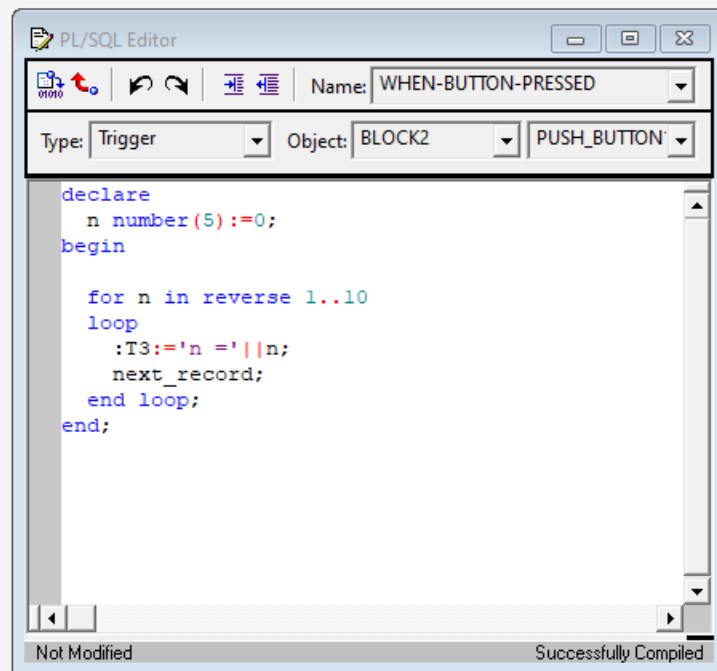
loop record

Option: Reverse:

***You may use Trigger (when-mouse-doubleclick). -> T3***

T3
T3
T3
T3
T3
T3
T3
T3
T3
T3

loop record



```
PL/SQL Editor
Name: WHEN-BUTTON-PRESSED
Type: Trigger
Object: BLOCK2
Event: PUSH_BUTTON

declare
  n number(5) := 0;
begin
  for n in reverse 1..10
  loop
    :T3:='n ='|n;
    next_record;
  end loop;
end;
```

Not Modified Successfully Compiled

n =10

n =9

n =8

n =7

n =6

n =5

n =4

n =3

n =2

n =1

loop record

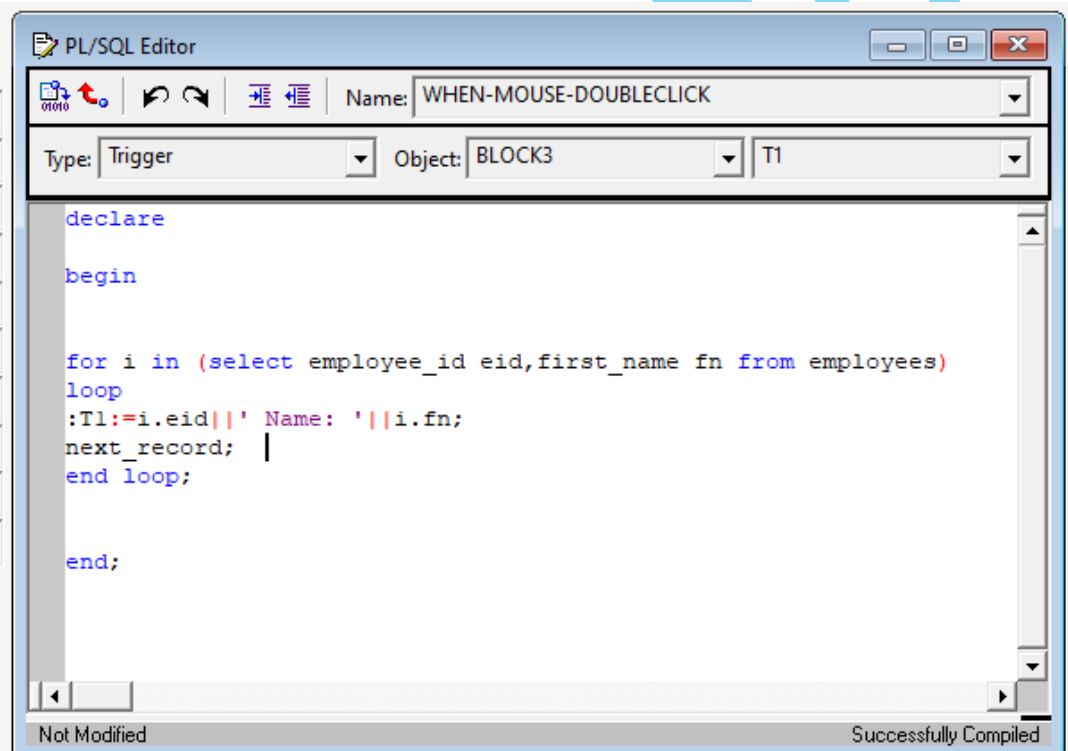
## ➤ For select.

For **index** in (select)

Loop

End loop;

T1
T1
T1
T1
T1
T1
T1
T1
T1
T1



```
PL/SQL Editor
Name: WHEN-MOUSE-DOUBLECLICK
Type: Trigger Object: BLOCK3 T1

declare
begin

for i in (select employee_id eid,first_name fn from employees)
loop
:T1:=i.eid||' Name: '||i.fn;
next_record;
end loop;

end;
```

Not Modified Successfully Compiled

➤ Select column1 **new-name**, column2 **new-name** from table;

## Example:

The screenshot shows a PL/SQL Editor window with the following details:

- Name:** WHEN-BUTTON-PRESSED
- Type:** Trigger
- Object:** BLOCK2
- Event:** PUSH\_BUTTON

The PL/SQL code in the editor is as follows:

```
declare
  counter number(2) := 0;
begin
  for i in (select * from employees)
  loop
    counter := counter + 1;
    :T3 := 'id = ' || i.employee_id;
    next_record;
  end loop;
  exit when (counter = 7);
end;
```

On the left side of the editor, there is a vertical list of text boxes, each labeled "T3". A "loop record" button is positioned to the right of these text boxes.

The screenshot shows a loop record interface with the following details:

- A vertical list of text boxes, each containing an "id" value (e.g., id = 100, id = 101, id = 102, id = 103, id = 104, id = 105, id = 106).
- A "loop record" button positioned to the right of the text boxes.

Limit (Counter)

## Note (I only access to the columns we selected)

- True

PL/SQL Editor

Name: WHEN-BUTTON-PRESSED

Type: Trigger Object: BLOCK2 PUSH\_BUTTON

```
declare
  counter number(2) := 0;
begin
  for i in (select employee_id from employees)
  loop
    counter := counter + 1;

    :T3 := 'id = ' || i.employee_id;

    next_record;

    exit when (counter = 7);
  end loop;
end;
```

Not Modified Successfully Compiled

- False

PL/SQL Editor

Name: WHEN-BUTTON-PRESSED

Type: Trigger Object: BLOCK2 PUSH\_BUTTON

```
counter number(2) := 0;
begin
  for i in (select employee_id from employees)
  loop
    counter := counter + 1;

    :T3 := 'id = ' || i.first_name;
```

Error 302 at line 10, column 18  
component 'FIRST\_NAME' must be declared  
Error 0 at line 10, column 3  
Statement ignored

Modified Compiled with Errors

Example:

16	
32	T1
48	T1
64	T1
80	T1
96	T1
112	T1
128	T1
144	T1
160	T1
176	T1
192	T1
208	T1
224	

Steven 100
Neena 101
Lex 102
Alexander 103
Bruce 104

```
PL/SQL Editor
Name: WHEN-MOUSE-DOUBLECLICK
Type: Trigger Object: BLOCK2 T1

declare
  n number(2) :=0;
begin
  for i in (select first_name FF, employee_id EE from employees order
  loop
    :T1:=i.FF||' '|| i.EE;
  next_record;
  n:=n+1;
  exit when(n=5);
  end loop;

  end;
```

Not Modified Successfully Compiled



## Example(Order By Column)

PL/SQL Editor

Name: WHEN-BUTTON-PRESSED

Type: Trigger Object: BLOCK2 PUSH\_BUTTON18

```
declare
  counter number(2) := 0;
begin
  for anything in (select * from employees order by employee_id)
  loop

    :T3:= 'ID:' || anything.employee_id || ' Name:' || anything.first_name;

    next_record;

    counter:=counter+1;
    exit when(counter=7);
  end loop;
  select first_name into :T3
  from employees where employee_id=120;
end;
```

Not Modified Successfully Compiled

ID:100 Name:Steven

ID:101 Name:Neena

ID:102 Name:Lex

ID:103 Name:Alexander

ID:104 Name:Bruce

ID:105 Name:David

ID:106 Name:Valli

Abdalwahab

loop record

## Problem:

The screenshot shows the Oracle APEX Canvas2 (BLOCK3) interface. The canvas contains a table with columns T1, T2, and T3. A button labeled PUSH\_BUTTON7 is present. The PL/SQL Editor window shows the trigger code for WHEN-BUTTON-PRESSED:

```

begin
:T1:=10; -- Record 1
:T2:=10;
next_record;
:T1:=10; -- Record 2
:T2:=10;
next_record;
:T1:=10; -- Record 3
:T2:=10;
next_record;
:T3:=100; -- Record 4
end;

```

The status bar indicates "Not Modified" and "Successfully Compiled".

The screenshot shows the APEX Canvas2 (BLOCK3) interface after the button press. The table contains the following data:

T1	T2	T3
10	Record 1	10
10	Record 2	10
10	Record 3	10
	Record 4	100
	Record 5	
	.	
	.	
	.	
	.	
	.	
	.	

The button is labeled PUSH\_BUTTON7. The text "نحن الان نقف على السجل الرابع" (We are now at the fourth record) is displayed.

## Problem

WINDOW1

10	10	
10	10	
10	10	

عند استخدام سجل اخر يتغير السجل لجميع العناصر التي هي من نفس البلوك

PUSH\_BUTTON7

Next: Enter record 4

WINDOW1

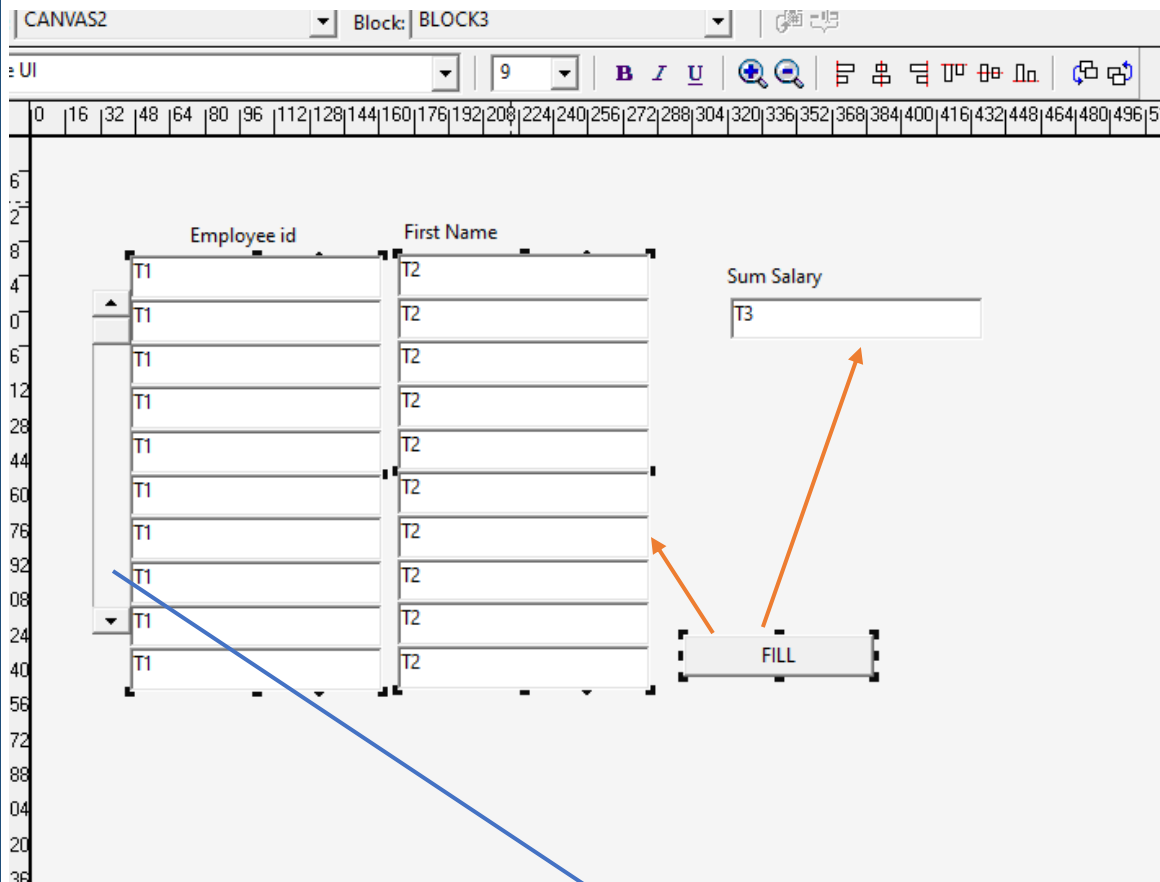
10	10	100
10	10	
10	10	

PUSH\_BUTTON7

Solution:

Add T3 to another block.

Example:



Property Palette

Find:

Data Block: BLOCK3

Precompute summaries

- DML Returning Value: No

Scrollbar

- ▣ Show Scroll Bar: Yes
- ▣ Scroll Bar Canvas: CANVAS2
- ▣ Scroll Bar Tab Page: <Null>
- Scroll Bar Orientation: Vertical
- ▣ Scroll Bar X Position: 29
- ▣ Scroll Bar Y Position: 70
- ▣ Scroll Bar Width: 18
- ▣ Scroll Bar Length: 158
- Reverse Direction: No

Visual Attributes

- Visual Attribute Group: DEFAULT

Internal name of the object.

Object Navigator

BLOCK3

Find:

- Triggers
- Alerts
- Attached Libraries
- Data Blocks
  - BLOCK3
    - Triggers
    - Items
      - T1
      - T2
      - PUSH\_BUTTON10
        - Triggers
          - WHEN-BUTTON-PRESSED
    - Relations
      - BLOCK5
- Triggers
- Items
  - T3
- Relations
- Canvases
  - CANVAS2

Nvl (column, alternative value)

- If an empty value is returned, replace it with the alternative value.
- And it must be named.



```

Name: WHEN-BUTTON-PRESSED
Type: Trigger Object: BLOCK3 PUSH_BUTTON10

declare
sal employees.salary%type:=0;
begin

for indexx in (select employee_id,first_name,nvl(salary,0) salName from employees order by employee_id)
loop
:T1:=indexx.employee_id;
:T2:=indexx.first_name;
next_record;

sal:=sal+indexx.salName;

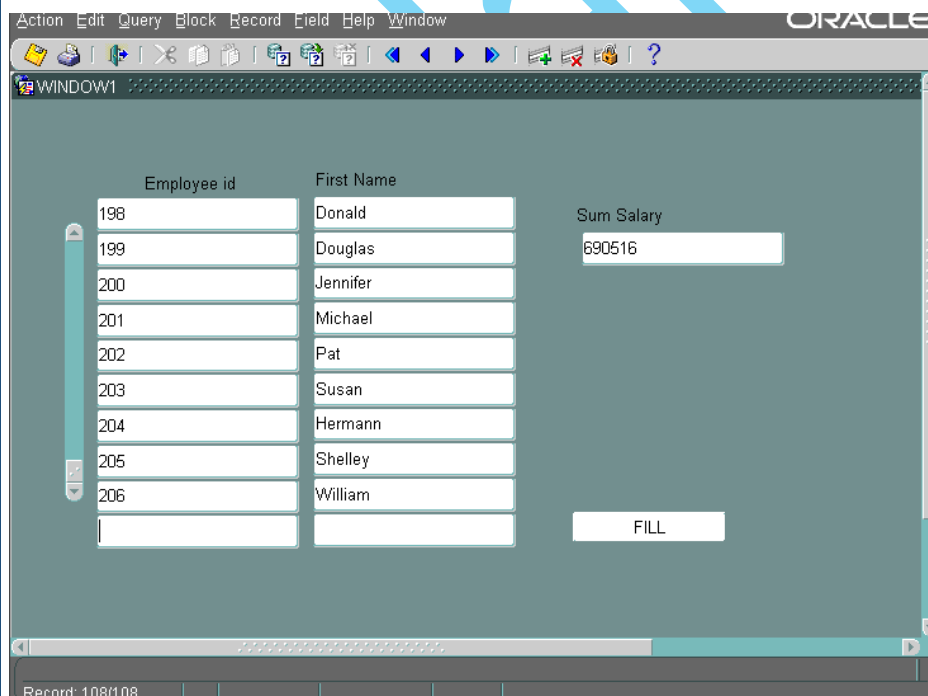
end loop;

:T3:=sal;

end;
```

Not Modified Successfully Col

Run:

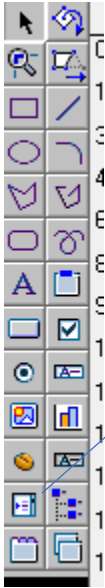


The screenshot shows the Oracle Forms application window titled 'WINDOW1'. It contains a table with three columns: 'Employee id', 'First Name', and 'Sum Salary'. The table lists employees with IDs 198 through 206. The 'Sum Salary' column shows a value of 690516. Below the table is a button labeled 'FILL'. The status bar at the bottom indicates 'Record: 108/108'.

Employee id	First Name	Sum Salary
198	Donald	
199	Douglas	690516
200	Jennifer	
201	Michael	
202	Pat	
203	Susan	
204	Hermann	
205	Shelley	
206	William	

The screenshot shows the Canvas2 application interface. At the top, there is a toolbar with various icons for editing and viewing. Below the toolbar, a table is displayed with two columns: "Employee id" and "First Name". The table contains 10 rows of data, with "T1" in the "Employee id" column and "T2" in the "First Name" column. To the right of the table, there is a "Sum Salary" block with a text input field containing "T3". A blue arrow points from the "Sum Salary" block to a box labeled "B". Below the table, there is a "Block 2" label with a blue arrow pointing to it. At the bottom right, there is a "FILL" button. The interface is designed to allow users to interact with data and perform calculations.

# List



***There are two ways to populate the list:***

- 1. Property palette.***
- 2. Using (Add-list-element).***

***The list consists of:***

- 1. Name***
- 2. Index***
- 3. Label***
- 4. Value***

## 1. Using Property palette.



Property Palette

Find:

Item: L1

General	
Name	L1
Item Type	List Item
Subclass Information	
Comments	
Help Book Topic	
Functional	
Enabled	Yes
Elements in List	
List Style	Poplist

Internal name of the object.



Property Palette

Find:

Item: L1

General	
Name	L1
Item Type	List Item
Subclass Information	
Comments	
Help Book Topic	
Functional	
Enabled	Yes
Elements in List	<a href="#">More...</a>
List Style	Poplist
Mapping of Other Values	
Implementation Class	
Case Restriction	Mixed
Popup Menu	<Null>
Navigation	

Property 6396 Hint



Property Palette

Find:

Item: L1

General	
Name	L1
Item Type	List Item
Subclass Information	
Comments	
Help Book Topic	
Functional	
Enabled	Yes
Elements in List	More...
List Style	Poplist
Mapping of Other Values	
Implementation Class	
Case Restriction	Mixed
Popup Menu	<Null>
Navigation	

Property 6396 Hint

Name

Value

List Elements

List Elements

Abdalwahab

Ali

Heba

List Item Value

1

OK Cancel Help

List Elements

List Elements

Abdalwahab

Ali

Heba

List Item Value

2

OK Cancel Help

List Elements

List Elements

Abdalwahab

Ali

Heba

List Item Value

3

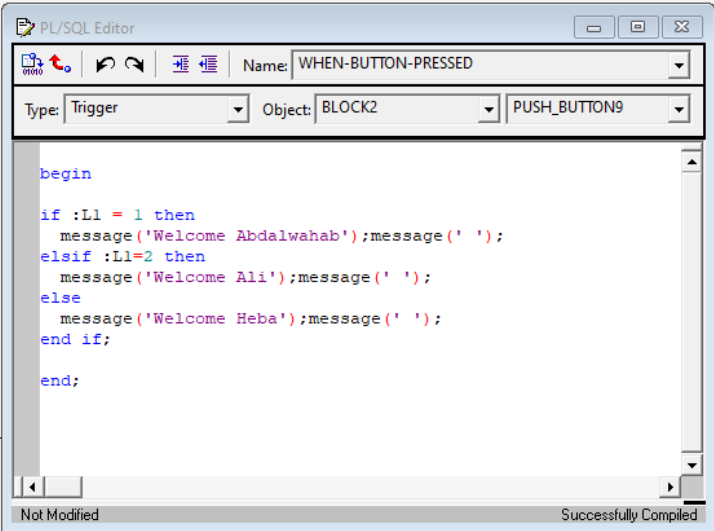
OK Cancel Help

## Value processing:

يمكن التعامل مع قيمة القائمة مثل قيمة اي عنصر اخر

L1

PUSH\_BUTTON9



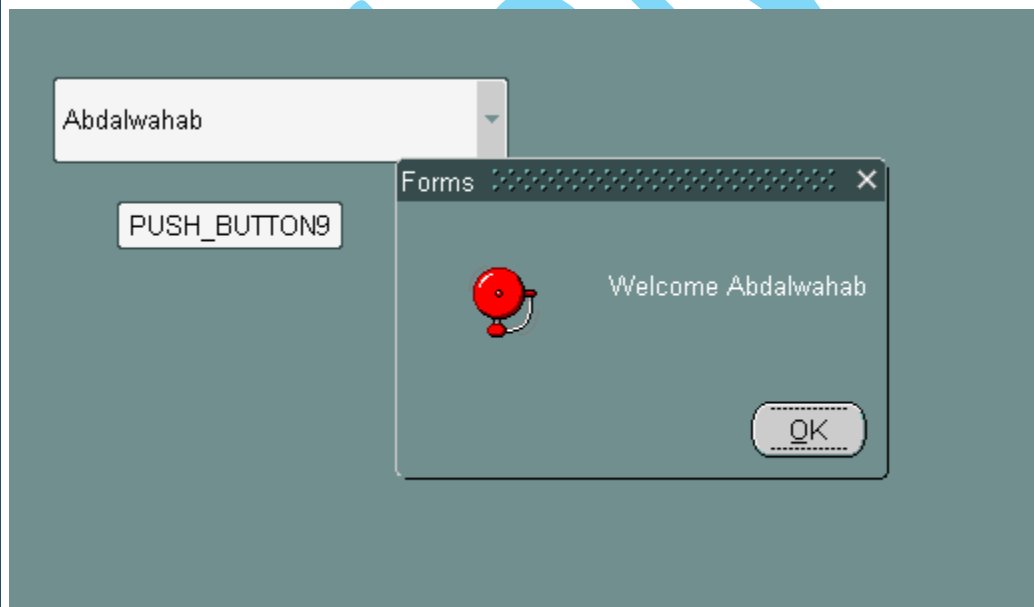
```
begin
if :L1 = 1 then
  message('Welcome Abdalwahab');message(' ');
elsif :L1=2 then
  message('Welcome Ali');message(' ');
else
  message('Welcome Heba');message(' ');
end if;
end;
```

Not Modified Successfully Compiled

Run:

Abdalwahab

PUSH\_BUTTON9

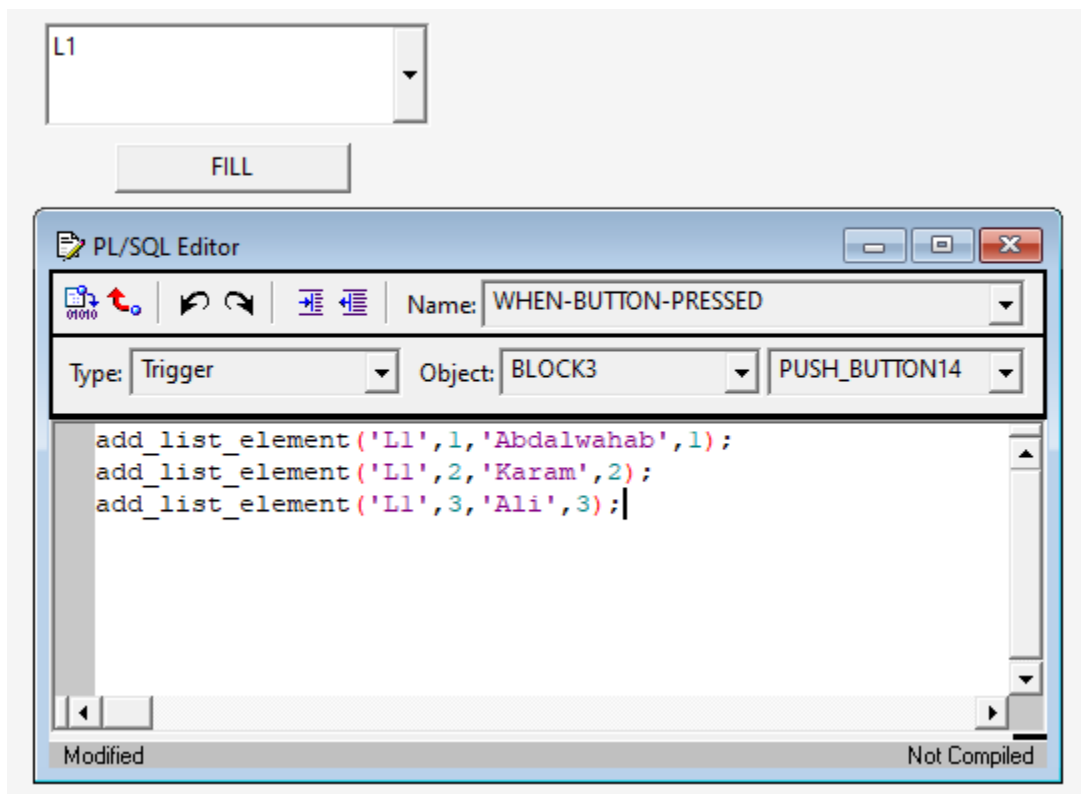


Forms

Welcome Abdalwahab

OK

## 2. Using *add list element*.



Problem:

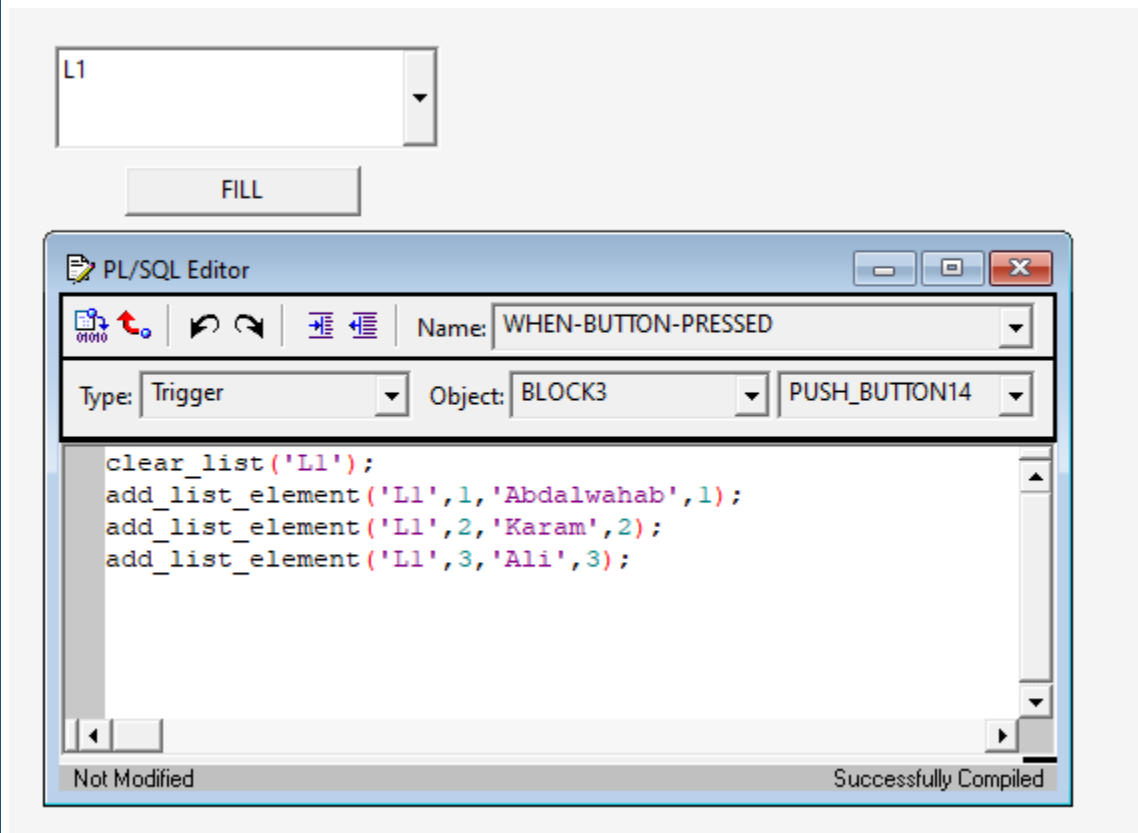
One click:



2 clicks:



## ***Solution (clear\_list('listName'))***



Can be used:

1. When-new-form-instance (Best Use).
2. When-mouse-douBlick.

# Example:

Employee id	First Name
T1	T2
T1	T2
T1	T2
T1	T2
T1	T2
T1	T2
T1	T2
T1	T2
T1	T2

Sum Salary  
T3

Employees  
L1

FILL

```
PL/SQL Editor
Name: WHEN-BUTTON-PRESSED
Type: Trigger
Object: BLOCK3
PUSH_BUTTON10

begin
for c in (select employee_id,first_name,nvl(salary,0) salName from employees order by employee_id)
loop
:T1:=c.employee_id;
:T2:=c.first_name;
next_record;

sal:=sal+c.salName;

i:=i+1;
add_list_element('L1',i,c.first_name,c.employee_id);

end loop;

:T3:=sal;

end;
```

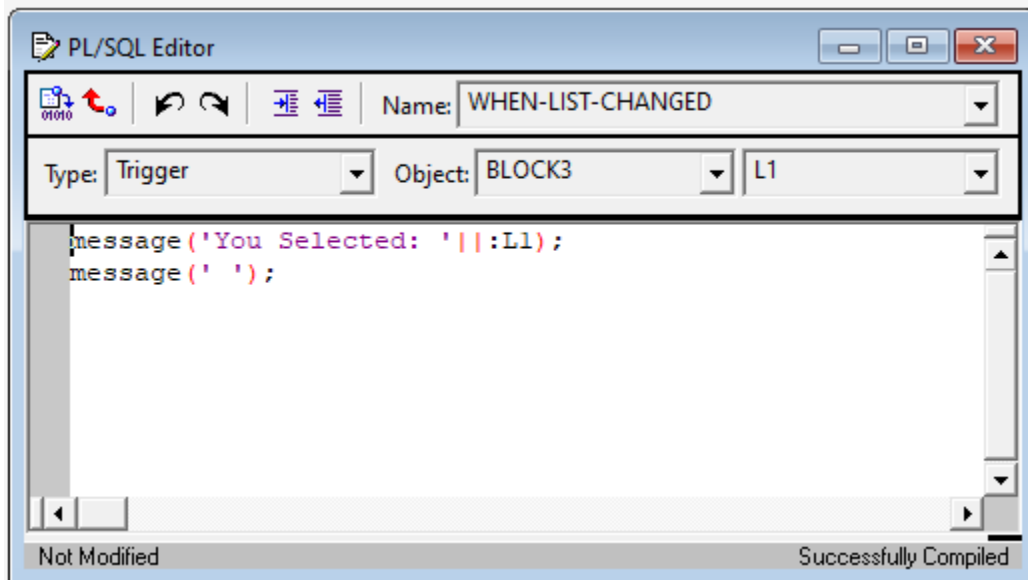
Not Modified Successfully Compiled

## ➤ Trigger(when-list-changed)

عند اختيار قيمة للقائمة يتفعل الحدث

Employees

L1



# Example

you select id

L1

Employee ID

Salary

T1

T2

The screenshot displays the Oracle PL/SQL Developer interface. The Object Navigator on the left shows the hierarchy: Forms > MODULE1 > Triggers > WHEN-NEW-FORM-INSTANCE. The PL/SQL Editor window is open, showing the trigger definition for 'WHEN-NEW-FORM-INSTANCE' at the '(Form Level)'. The trigger is a 'Trigger' type. The code in the editor is as follows:

```
declare
  indexx number:=1;
begin
  for i in (select employee_id eid,first_name fn from employees order by employee_id)
  loop
    add_list_element('L1',indexx,i.fn,i.eid);
    indexx:=indexx+1;
  end loop;
end;
```

The status bar at the bottom indicates 'Modified' and 'Not Compiled'.

you select id

L1

Employee ID

T1

Salary

T2

PL/SQL Editor

Name: WHEN-LIST-CHANGED

Type: Trigger

Object: BLOCK3

L1

```
declare

begin

select salary into :T2
from employees
where employee_id=:L1;

:T1:=:L1;

end;
```

Not Modified

Successfully Compiled

you select id

Lex

Employee ID

102

Salary

17000



## Example:

Employee id	First Name
T1	T2
T1	T2
T1	T2
T1	T2
T1	T2
T1	T2
T1	T2
T1	T2
T1	T2
T1	T2
T1	T2

تعبئة القائمة من سجلات  
عند الضغط على قيمة احدا السجلات تضاف الى القائمة

Double Click  
Add Record

L2

FILL Records

FILL Records

```
PL/SQL Editor
Name: WHEN-BUTTON-PRESSED
Type: Trigger
Object: BLOCK3
PUSH_BUTTON22

declare

begin

  for i in (select employee_id eid, first_name fn, nvl(salary,0) sal,
              department_id from employees)

  loop
    :T1:=i.eid;
    :T2:=i.fn;
    next_record;
  end loop;

end;
```

Modified Not Compiled

0 16 32 48 64 80 96 112 128 144 160 176 192 208 224 240 256 272 288 304 320 336 352 368 384 400 416 432 448 464 480 496 512 528 544 560 576 592

Employee id	First Name
T1	T2
T1	T2
T1	T2
T1	T2
T1	T2
T1	T2
T1	T2
T1	T2
T1	T2
T1	T2

Double Click Add Record L2

FILL Records

PL/SQL Editor

Name: WHEN-MOUSE-DOUBLECLICK

Type: Trigger Object: BLOCK3 T1

```
add_list_element('L2',:T1,:T2,:T1);
```

When mouse double click

Modified Not Compiled

1x 154.50 334.50

Example:

Employee

T1	+
T1	+
T1	+
T1	+
T1	+
T1	+
T1	+
T1	+
T1	+
T1	+

Double Click Add Record

L2

FILL Records

PL/SQL Editor

Name: WHEN-BUTTON-PRESSED

Type: Trigger Object: BLOCK3 PUSH\_BUTTON:

```
add_list_element('L2',1,'List1: '::T1,:T1);
```

Modified Not Compiled

Employee

T1	+
T1	+
T1	+
T1	+
T1	+
T1	+
T1	+
T1	+
T1	+
T1	+

Double Click Add Record

L2

FILL Records

331.50

PL/SQL Editor

Name: WHEN-BUTTON-PRESSED

Type: Trigger Object: BLOCK3 PUSH\_BUTTON22

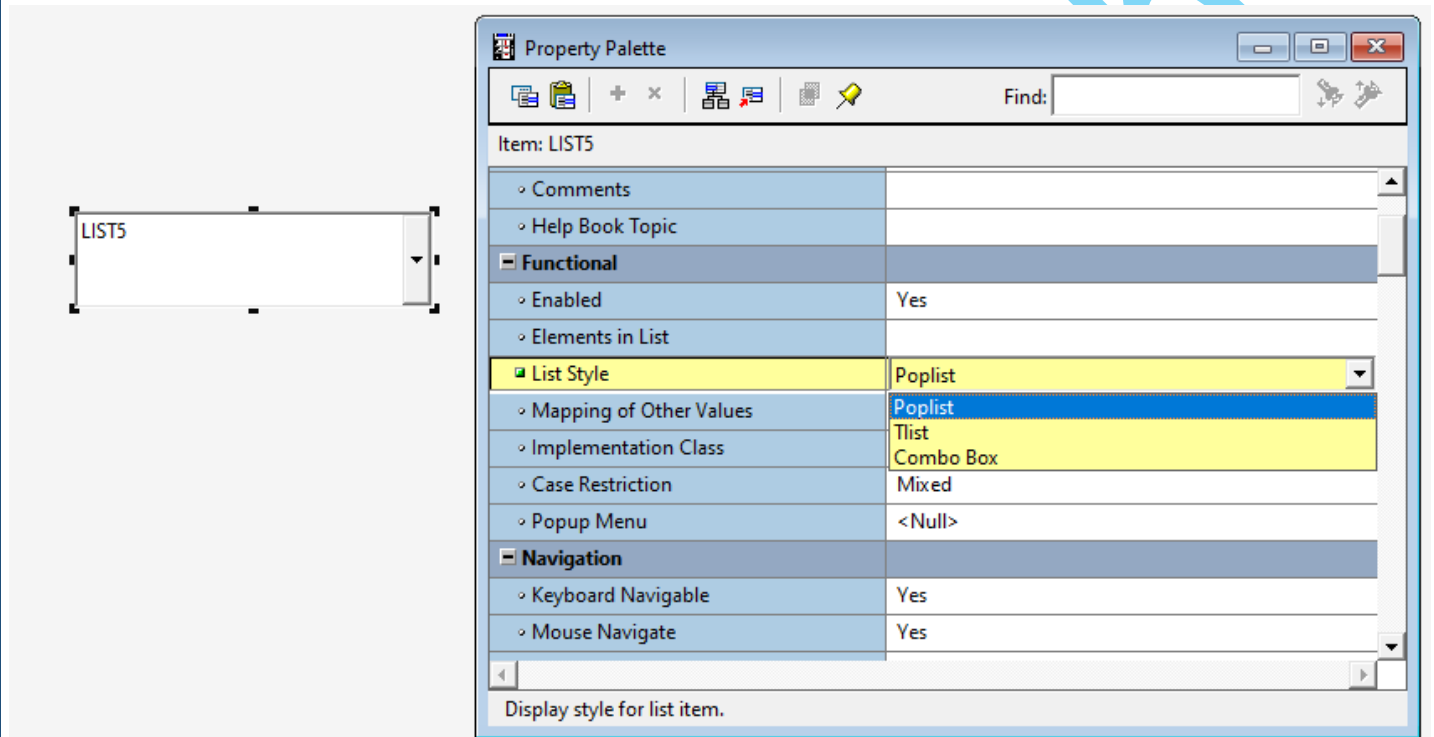
```
declare
begin
  for i in (select employee_id eid, first_name fn, nvl(salary,0) sal,
    department_id from employees order by employee_id)
  loop
    :T1:=i.eid||i.fn;
    next_record;
  end loop;
end;
```

Not Modified Successfully Compiled

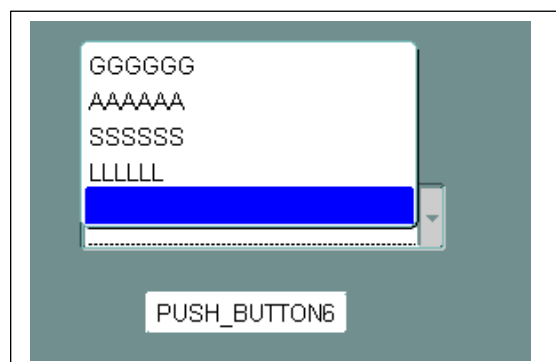
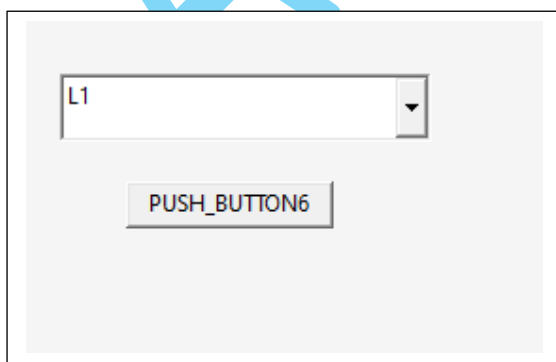
- General
  - Name
  - Subclass Information
  - Comments
- Functional
  - Trigger Style
  - Trigger Text
  - Fire in Enter-Query Mode

## List Types:

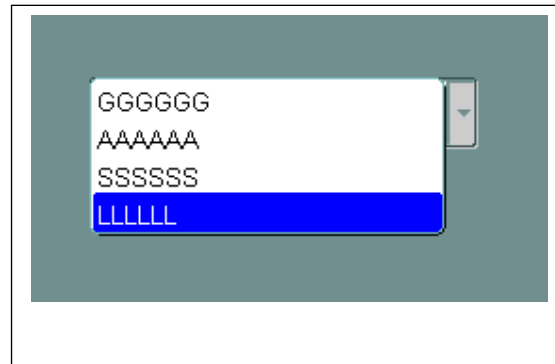
1. Pop-list
2. T-list
3. Combo Box



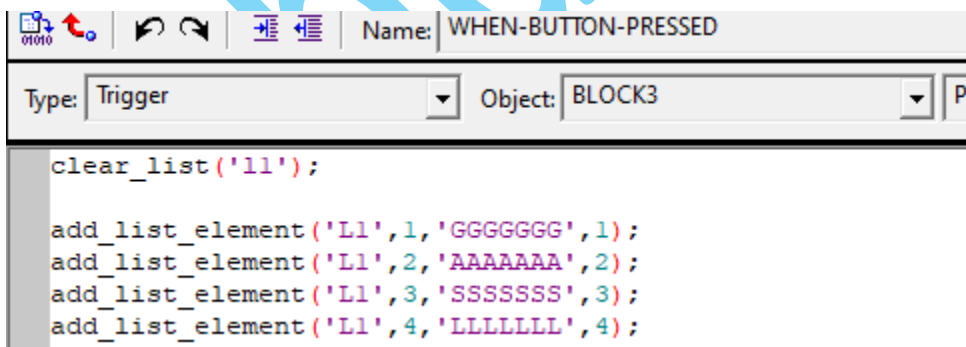
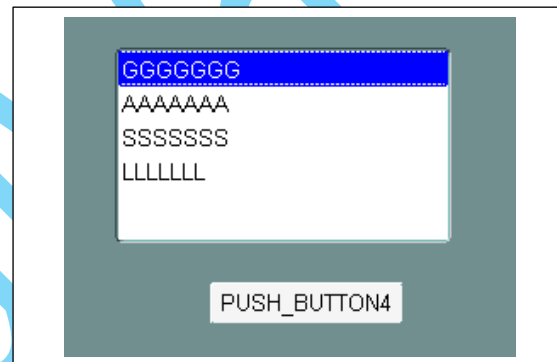
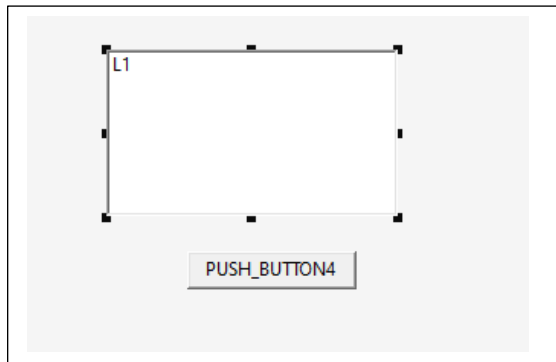
### 1. Pop-list



## 2. Combo Box



## 3. T-list



### For types:

1. **For Numeric.**      For i in x..y
2. **For Select.**      For i in (select)
3. **For Cursor.**

Note:

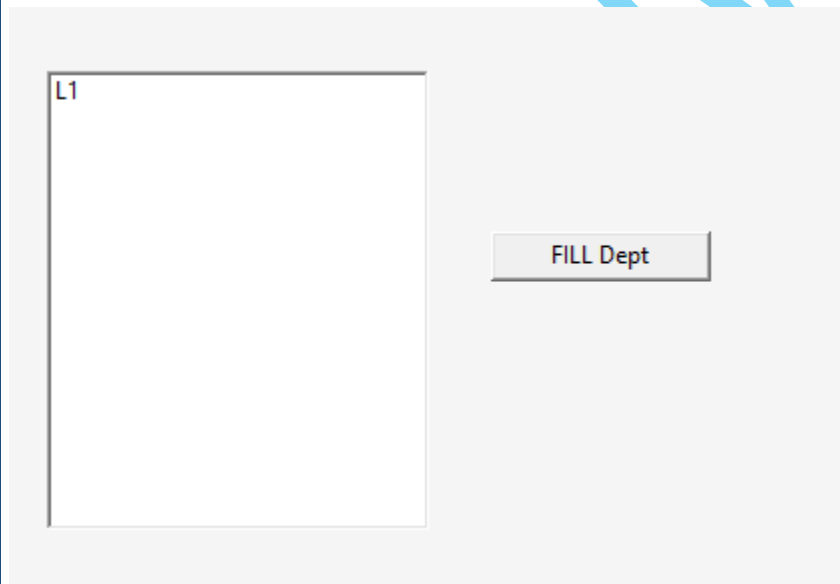
For **x** in (select .....)

**x** => Row Type (of the selected columns).

- **Cursor:** it is used for memory management.
- The **cursor** is defined in the <declare>.

---

## Example



The image shows a software interface with a light gray background. On the left, there is a rectangular box representing a table. The top-left corner of the table contains the text 'L1'. To the right of the table, there is a button with the text 'FILL Dept'.

## ➤ First Way:

```
PL/SQL Editor
Name: WHEN-BUTTON-PRESSED
Type: Trigger Object: BLOCK3 PUSH_BUTTON4

declare
  cursor dept is select department_id did, department_name dn from departments;
begin
  clear_list('L1');

  for i in dept
  loop
    add_list_element('L1', i.did, i.dn, i.did);
  end loop;
end;
```

Open dept

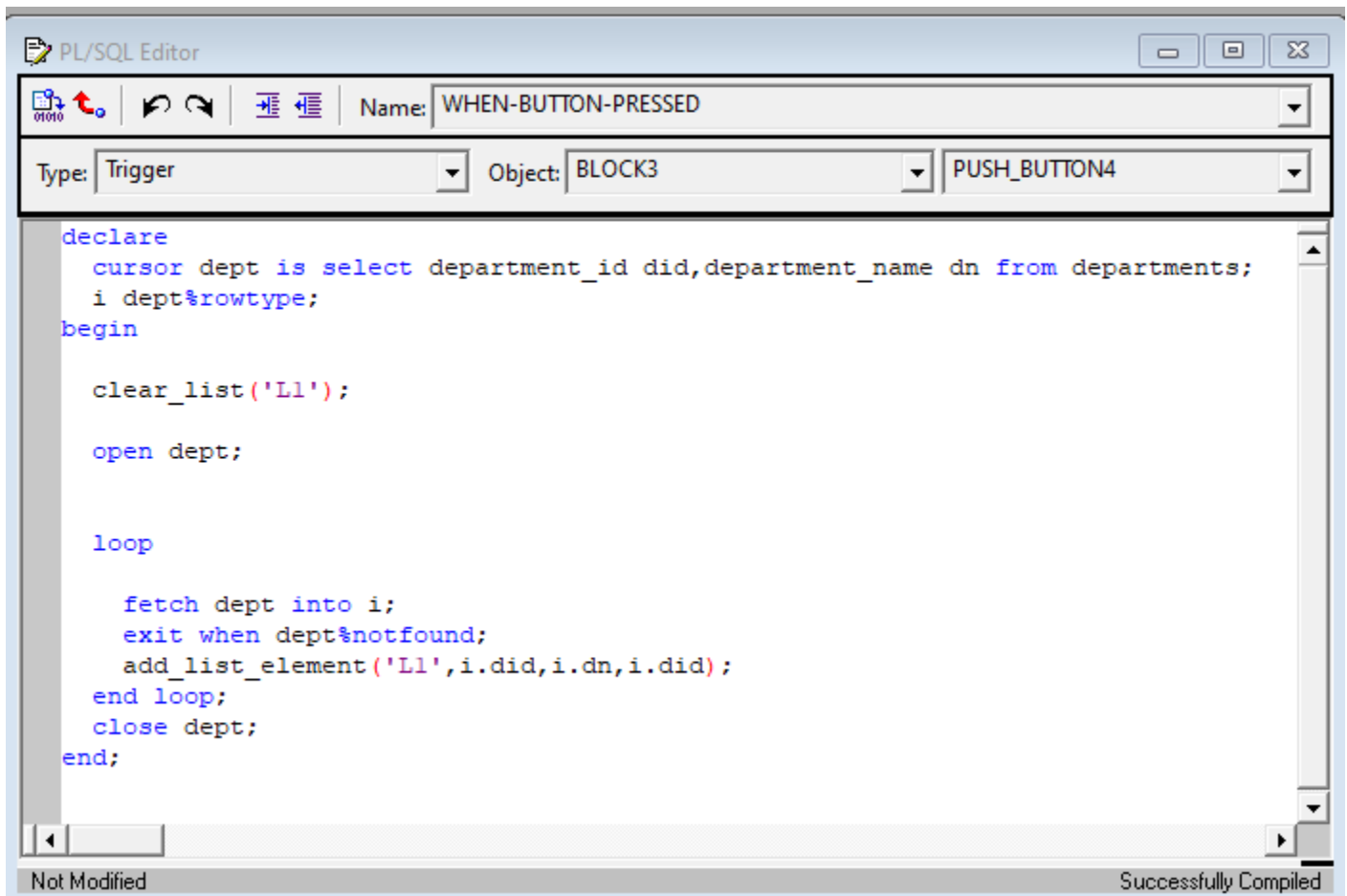
Not Modified Successfully Compiled

Run

Administration  
Marketing  
Purchasing  
Human Resources  
Shipping  
IT  
Public Relations  
Sales  
Executive  
Finance  
Accounting  
Treasury

FILL Dept

## ➤ Second Way



```
PL/SQL Editor
Name: WHEN-BUTTON-PRESSED
Type: Trigger Object: BLOCK3 PUSH_BUTTON4

declare
  cursor dept is select department_id did, department_name dn from departments;
  i dept%rowtype;
begin
  clear_list('L1');

  open dept;

  loop

    fetch dept into i;
    exit when dept%notfound;
    add_list_element('L1',i.did,i.dn,i.did);
  end loop;
  close dept;
end;
```

Not Modified Successfully Compiled

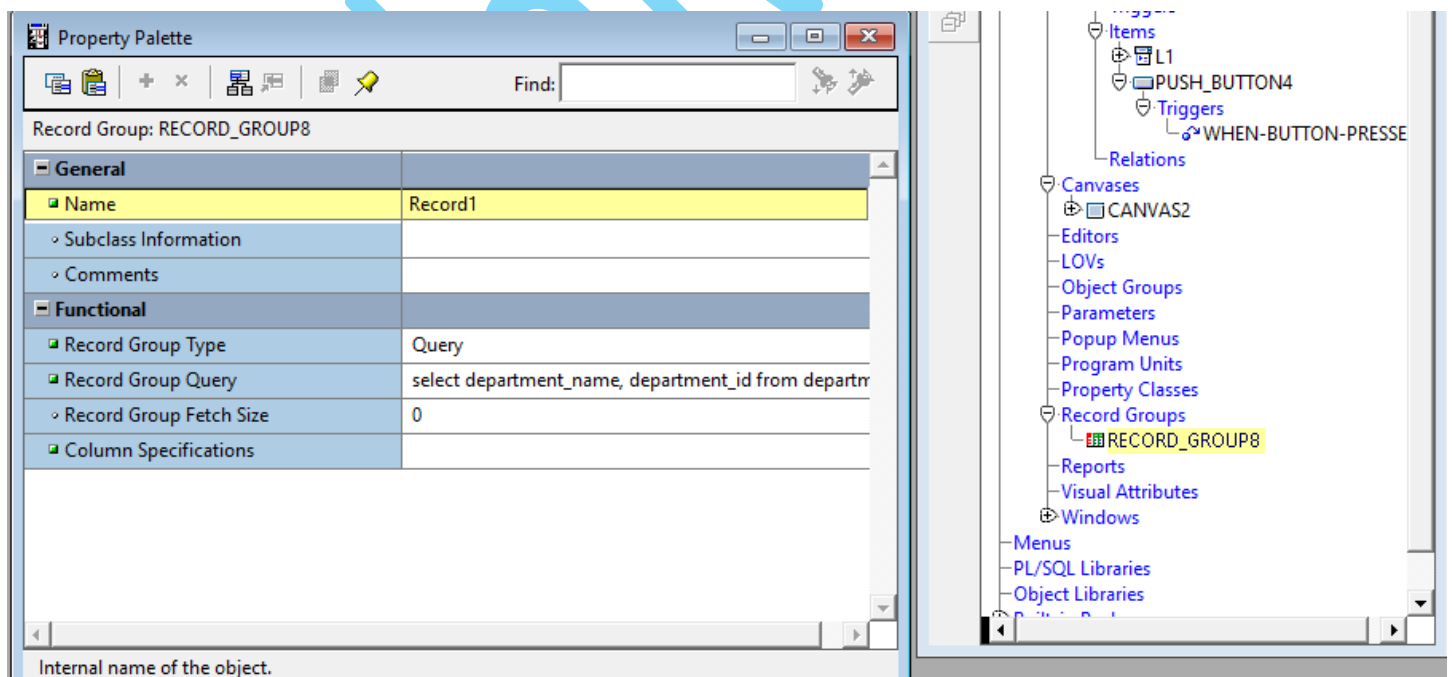
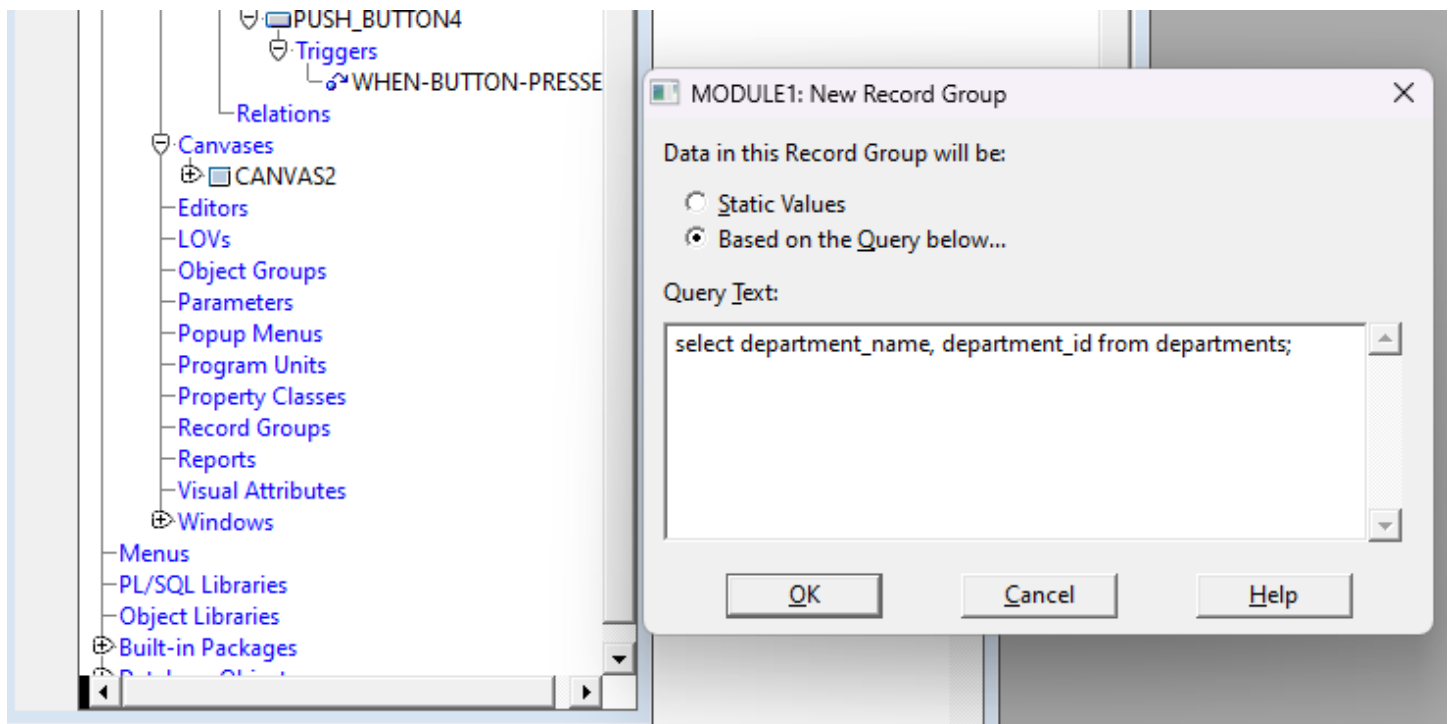
Same Run

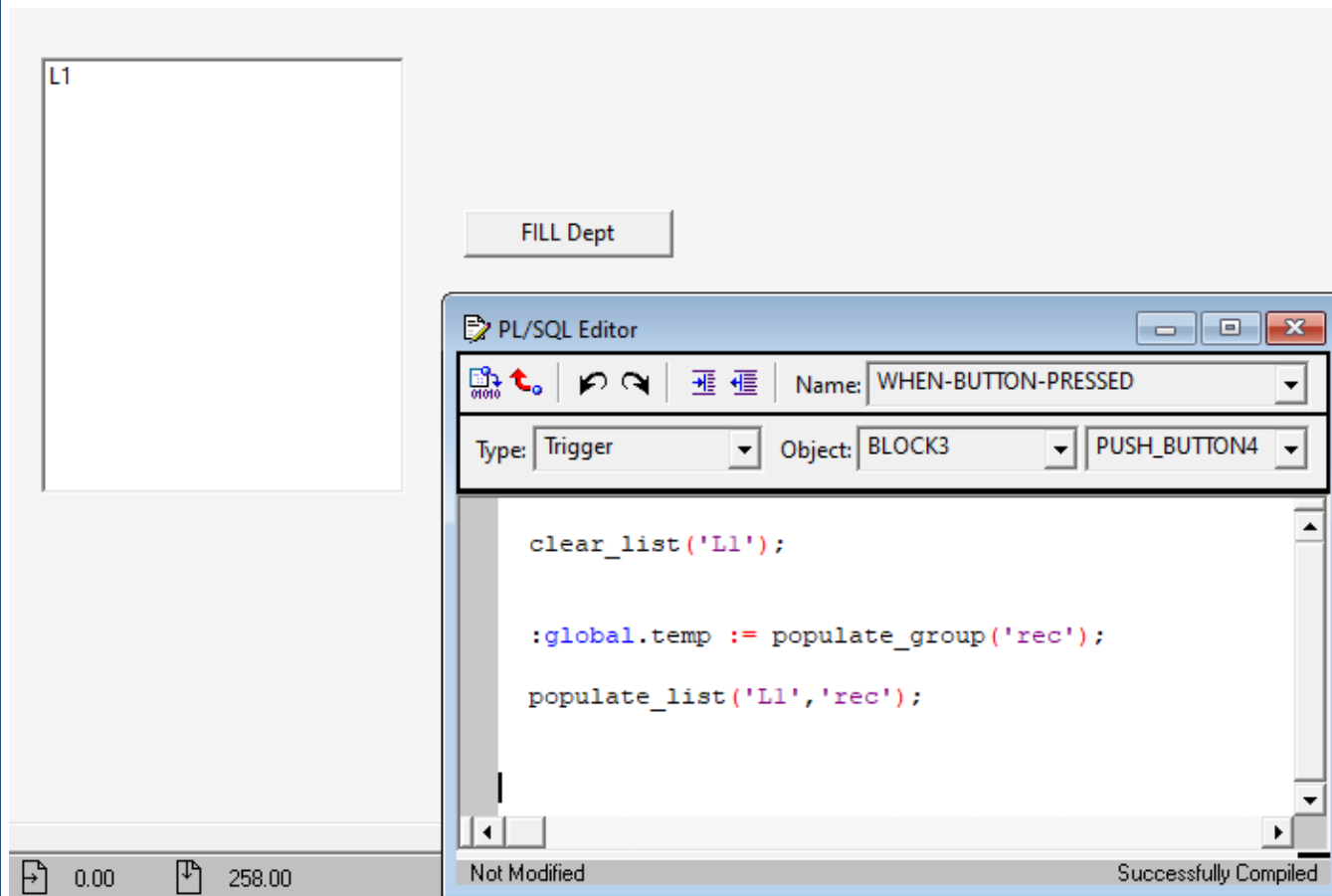
Dept%notFound

- ⇒ If the list is empty.
- ⇒ Must be before adding.



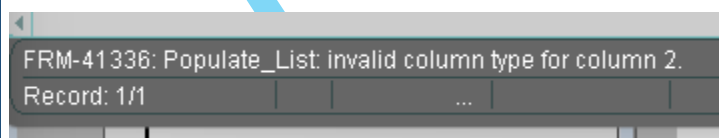
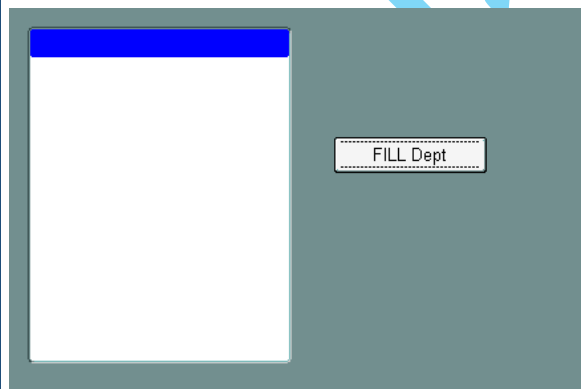
## ➤ Using Record Group



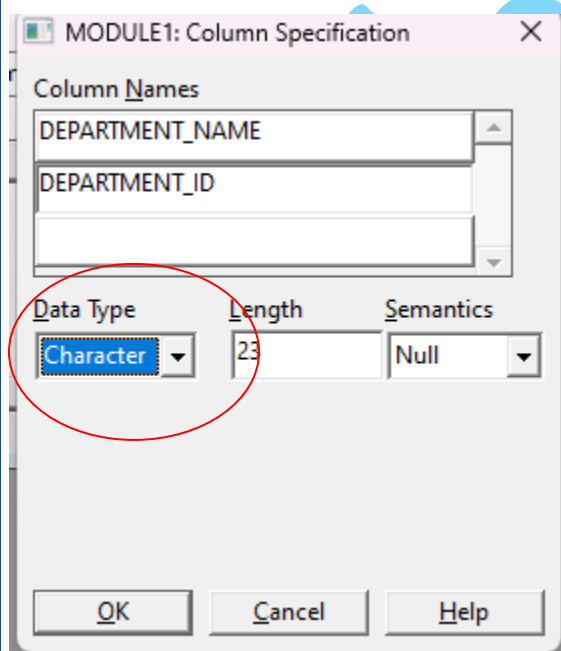
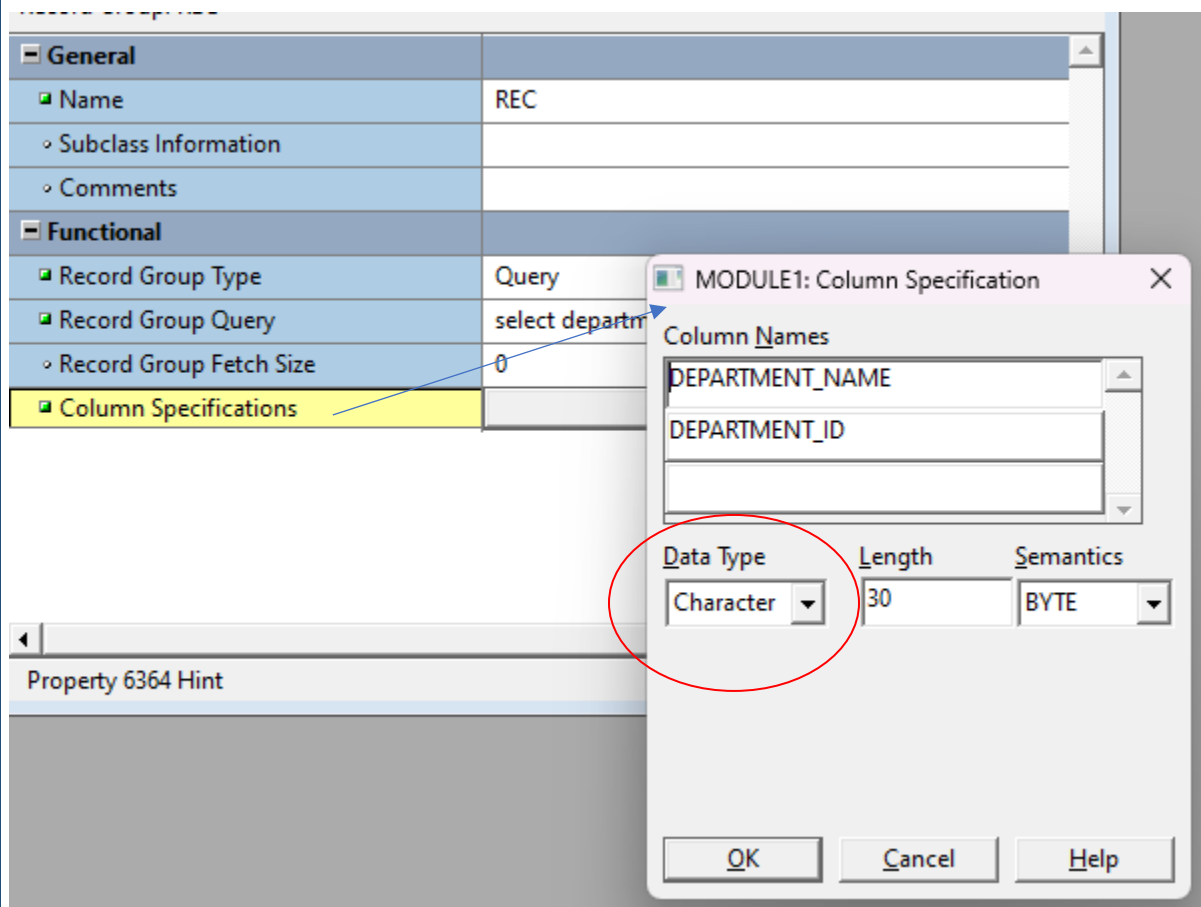


Run:

Problem:



➤ All columns must be Character



When new block instance

يجب ان تكون البيانات مرتبه

Example:

L1	L2
----	----

➤ Create Procedure

The screenshot displays the PL/SQL Editor window with the following content:

**PL/SQL Editor**

Name: FILL\_LIST (Procedure Body)

Type: Program Unit Object:

```
PROCEDURE Fill_list(list_name varchar2) IS  
  
  cursor emp is select first_name fn,  
                    employee_id eid from employees;  
  
  i emp%rowtype;  
  
  counter number:=0;  
  
BEGIN  
  clear_list(list_name);  
  open emp;  
  
  loop  
    counter:=counter+1;  
    exit when emp%notfound;  
    fetch emp into i;  
  
    add_list_element(List_name,counter,i.fn,i.eid);  
  
  end loop;  
  close emp;  
  
END;
```

The right-hand pane shows a tree view with the following items:

- Editors
- LOVs
- Object Groups
- Parameters
- Popup Menus
- Program Units
- Property Classes
- Record Groups
- Reports
- Visual Attributes
- Windows

An arrow points from the "FILL\_LIST (Procedure Body)" item in the tree view to the procedure code in the editor. The status bar at the bottom indicates "Modified" and "Not Compiled".

➤ Next



01010

Name: WHEN-NEW-BLOCK-INSTANCE

Type: Trigger Object: BLOCK3 (Data Block Level)

```
fill_list('L1');  
fill_list('L2');
```

1. Add list using cursor.
2. Add list using (for select).
3. Add list using (for cursor).

## Example Function/procedure:

The screenshot displays the Oracle Developer interface. On the left, the Object Navigator shows a tree structure for a form named 'MODULE1'. The tree includes 'Triggers', 'Alerts' (with 'ALT2'), 'Attached Libraries', 'Data Blocks' (with 'BLOCK4'), 'Items' (with 'PUSH\_BUTTON5' and 'PUSH\_BUTTON6'), 'Relations', 'Canvases' (with 'CANVAS3'), 'Editors', 'LOVs', 'Object Groups', 'Parameters', 'Popup Menus', 'Program Units', 'Property Classes', 'Record Groups', and 'Reports'. The 'CHANGE\_ALERT (Procedure Body)' is highlighted under 'Program Units'. On the right, the PL/SQL Editor shows the code for the 'CHANGE\_ALERT (Procedure Body)'.

```
PROCEDURE change_alert(v1 in varchar2, but1 varchar2, but2 varchar2) IS
var1 varchar2(20);
BEGIN
    set_alert_property('alt2', alert_message_text, v1);
    set_alert_button_property('alt2', 88, label, but1);
    set_alert_button_property('alt2', alert_button2, label, but2);

END;
```

The screenshot displays the Oracle Developer interface. On the left, the Object Navigator shows the same tree structure as the previous screenshot, but now 'GETM3 (Function Body)' is highlighted under 'Program Units'. On the right, the PL/SQL Editor shows the code for the 'GETM3 (Function Body)'.

```
FUNCTION Getm3(v1 in varchar2) RETURN varchar2 IS
var1 varchar2(20);
BEGIN
    message('Your variable is: '||v1);
    var1:=v1||v1;
    return var1;

END;
```

