

Data Base PL/SQL

Abdalwahab

Qatawneh

Form builder

- Trigger (key next item)

تستخدم (: قبل العنصر إذا ارنا التعامل مع قيمته
تستخدم (:=) من اجل إعطاء قيم للعناصر او المتغيرات
تستخدم (=) من اجل المقارنة

The screenshot shows a form builder interface with a canvas on the left and a PL/SQL Editor on the right. The canvas contains three items: a text item labeled 'T1', a date item labeled 'D1' with a 'sysdate' button, and another text item labeled 'D2' with a 'My Name' button. A blue arrow points from the 'D1' item to the PL/SQL Editor. The PL/SQL Editor shows a trigger named 'KEY-NEXT-ITEM' of type 'Trigger' on object 'BLOCK8'. The trigger code is:

```
:T1:=sysdate+20;  
message('Welcome');  
message(' ');
```

Annotations with arrows point from the code to the form preview:

- An arrow from `:T1:=sysdate+20;` points to a box labeled "إعطاء قيمة للعنصر" (Assigning value to the item).
- An arrow from `message('Welcome');` points to a box labeled "Print 'Welcome'" (Print "Welcome").
- An arrow from `message(' ');` points to the form preview.

The form preview shows a window titled 'Forms' with a red bell icon and the text 'welcome'. Below the text is an 'OK' button. The status bar at the bottom indicates 'Not Modified' and 'Success'.

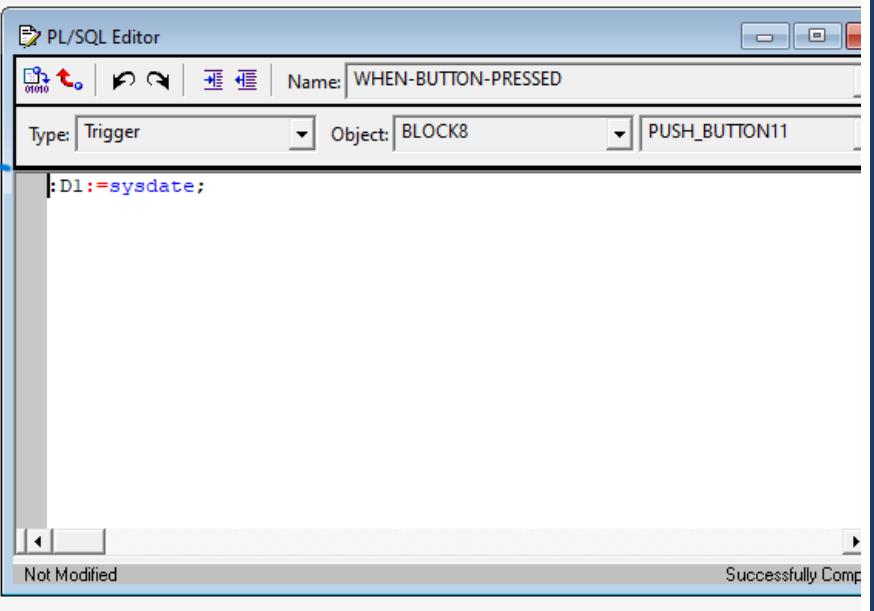
إعطاء تاريخ اليوم
للعنصر عند الانتقال
للعنصر الثاني

Node:

D -> Display-item

T -> Text-item

Trigger (when button pressed)



T1

D1 sysdate

D2 My Name

إعطاء تاريخ اليوم للعنصر
عند الضغط على

Button

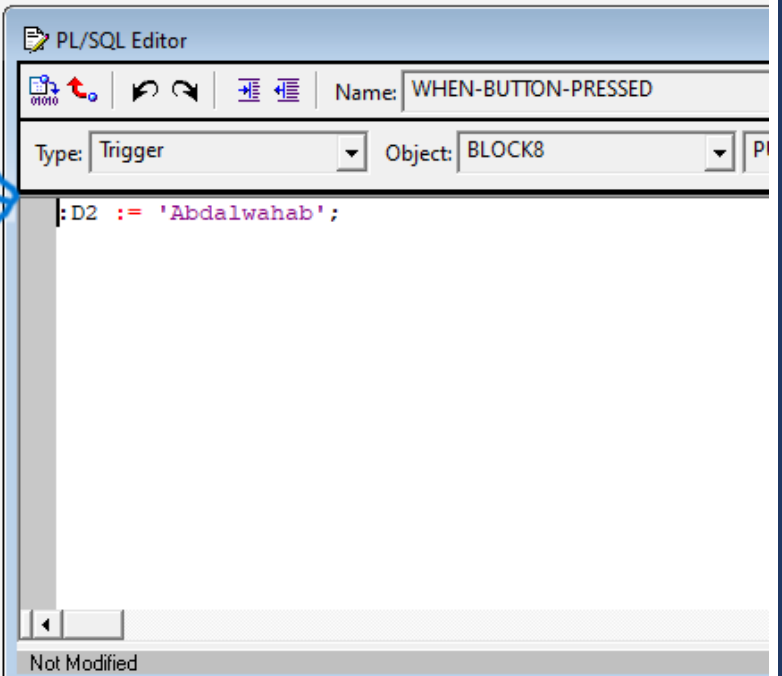
PL/SQL Editor

Name: WHEN-BUTTON-PRESSED

Type: Trigger Object: BLOCK8 PUSH_BUTTON11

:D1:=sysdate;

Not Modified Successfully Comp



E1: CANVAS9 (BLOCK8)

CANVAS9 Block: <Null>

9

B I U

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

T1

D1 sysdate

D2 My Name

PL/SQL Editor

Name: WHEN-BUTTON-PRESSED

Type: Trigger Object: BLOCK8

:D2 := 'Abdalwahab';

Not Modified

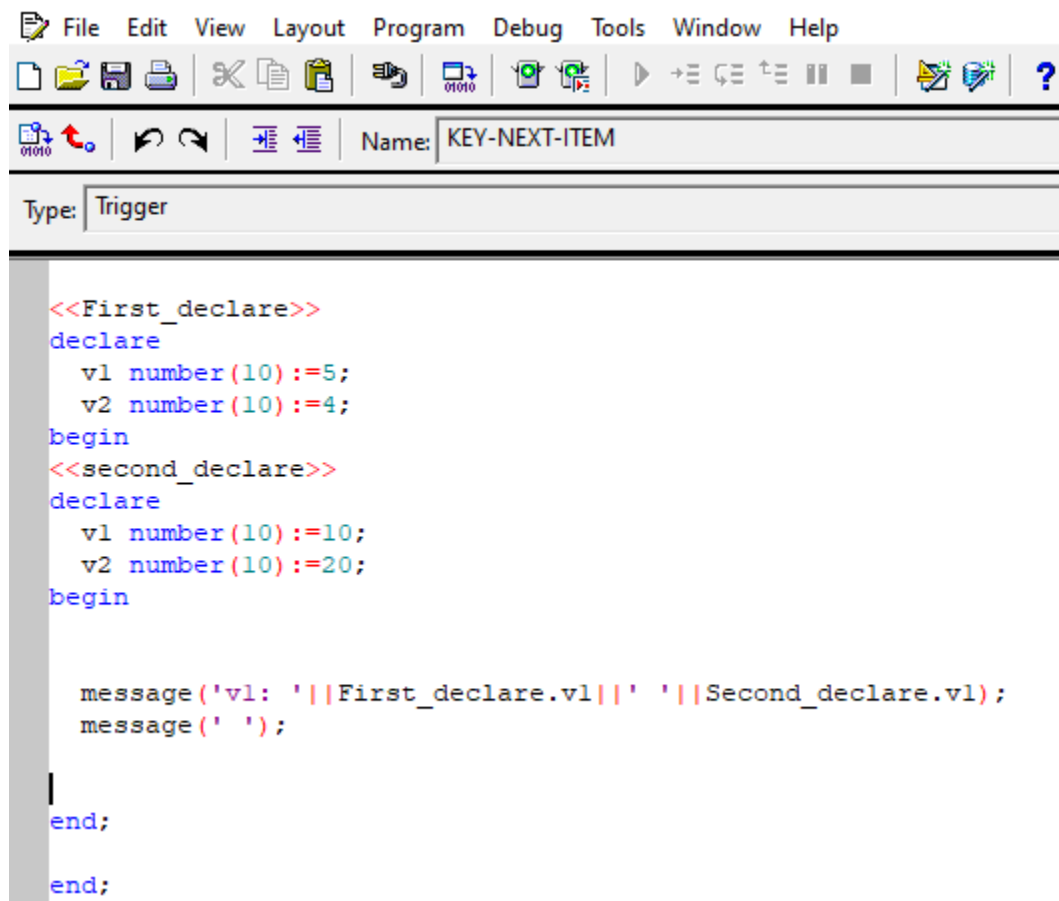
The screenshot shows a PL/SQL Editor window with the following code:

```
declare
  n1 number(10) :=120;
  v1 varchar(40) := 'My Name is Abdalwahab';
  d1 date := '01-mar-222';
begin
  :D1 := n1;
  :D2 := v1;
  :D3 := d1;
end;
```

The editor window has a title bar "PL/SQL Editor" and a menu bar. The "Name" field is set to "WHEN-BUTTON-PRESSED". The "Type" is set to "Trigger", the "Object" is "BLOCK3", and the "Push Button" is "PUSH_BUTTON7". A blue arrow points from the "input" button to the "Name" field. The status bar at the bottom indicates "Not Modified" and "Successfully Compiled".

- Declare: used to define variables

Ex: Nested block



The screenshot shows a PL/SQL IDE window titled 'KEY-NEXT-ITEM'. The 'Type' is set to 'Trigger'. The code defines two nested blocks: 'First_declare' and 'second_declare'. 'First_declare' declares variables 'v1' and 'v2' with initial values 5 and 4. 'second_declare' declares 'v1' and 'v2' with initial values 10 and 20. Both blocks end with 'end;' statements. The code is as follows:

```
<<First_declare>>
declare
  v1 number(10) :=5;
  v2 number(10) :=4;
begin
  <<second_declare>>
  declare
    v1 number(10) :=10;
    v2 number(10) :=20;
  begin

    message('v1: ' || First_declare.v1 || ' ' || Second_declare.v1);
    message(' ');
  |
end;

end;
```

```
declare
v1 number(10) :=22;
v2 number(10) :=44;
begin
declare
v1 number(10) :=10;
v2 number(10) :=20;
begin
message('first v1='||v1||'v2='||v2 );--10 / 20
end;
message('second v1='||v1||'v2='||v2 );-- 22 / 44
message(' ');
end;
```

Message:

Forms

first v1=10v2=20

OK

Forms

second v1=22v2=44

OK

Type: Trigger

```

<<anyname>>
declare

v1 number(10):=22;

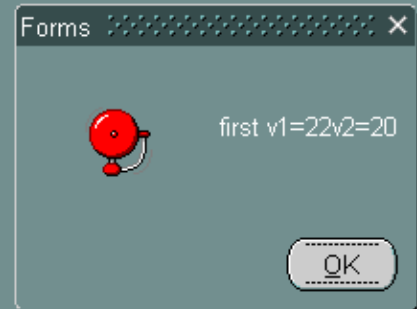
v2 number(10):=44;

begin
<<SECON_D>>
declare
  v1 number(10):=10;
  v2 number(10):=20;
begin

  message('first v1='||anyname.v1||'v2='||v2 );--10 / 20

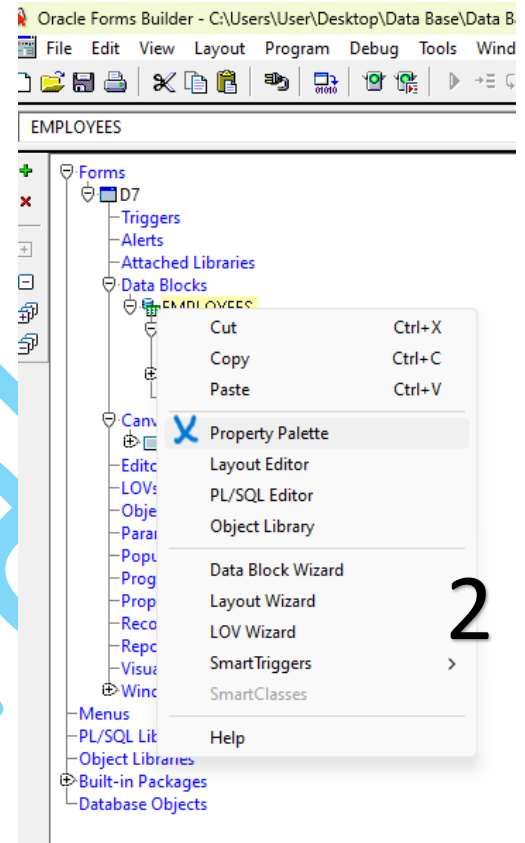
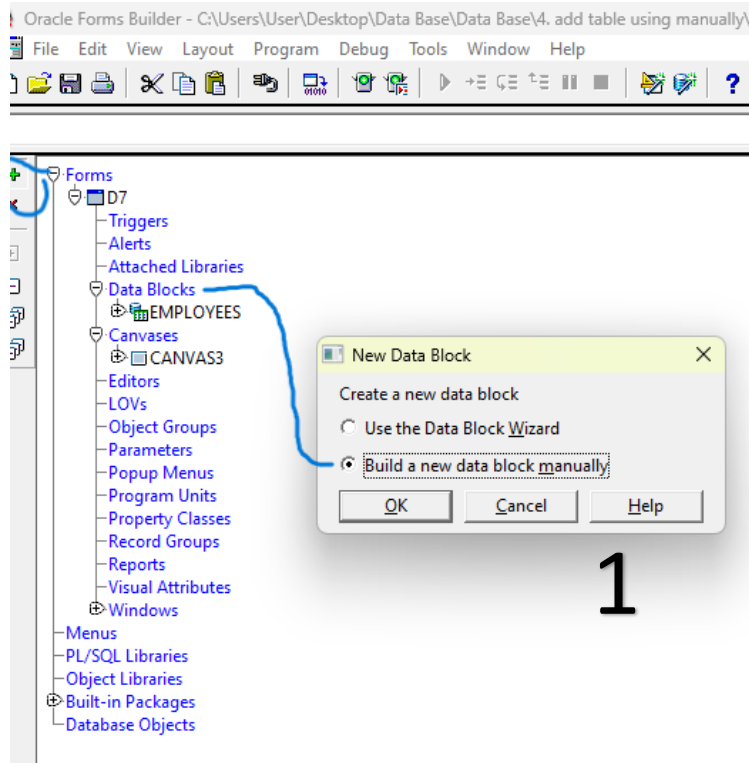
end;
  message('second v1='||v1||'v2='||v2 );-- 22 / 44
  message(' ');

end;
  
```



- Create new data block (build new data block manually)
- Connect items and blocks to databases.

1. Connect the BLOCK with the TABLE

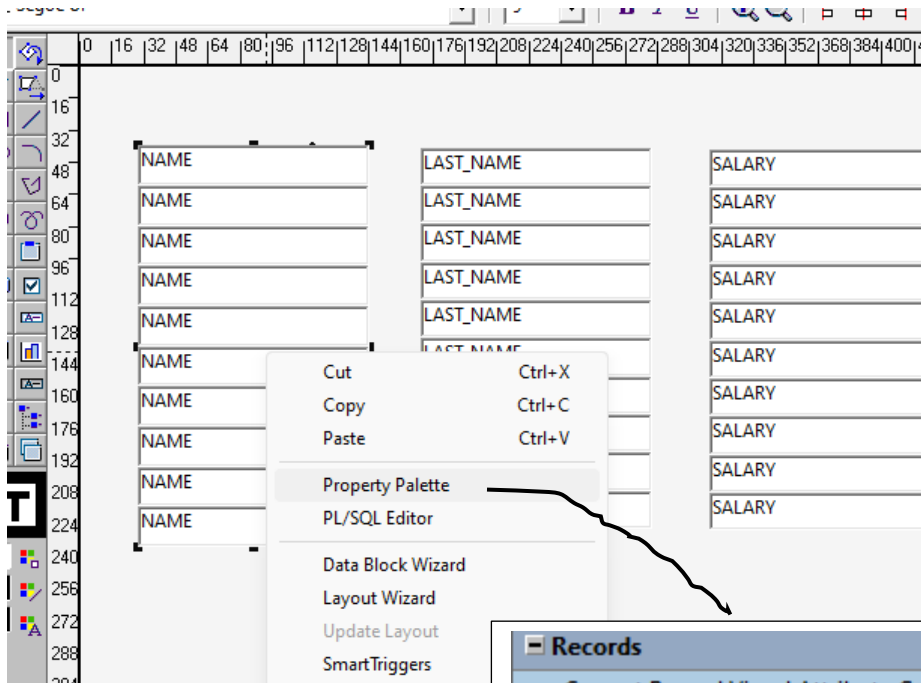


Records	
Current Record Visual Attribute Group	<Null>
Query Array Size	0
Number of Records Buffered	0
Number of Records Displayed	10
Query All Records	No
Record Orientation	Vertical
Single Record	No
Database	

- Table selection

Database	
Database Data Block	Yes
Enforce Primary Key	No
Query Allowed	Yes
Query Data Source Type	Table
Query Data Source Name	employees
Query Data Source Columns	
Query Data Source Arguments	
Alias	

2. Connect the ITEM with the COLUMN



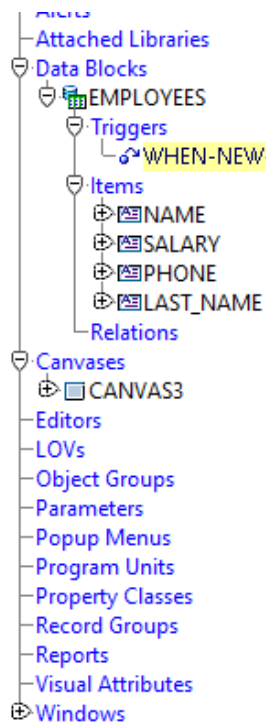
Records	
Current Record Visual Attribute Group	<Null>
Distance Between Records	0
Number of Items Displayed	0
Database	
Database Item	Yes
Column Name	first_name
Primary Key	No
Query Only	No
Query Allowed	Yes
Query Length	0
Case Insensitive Query	No
Insert Allowed	Yes
Update Allowed	Yes
Update Only if NULL	No

ملاحظة: هذا عدد قيم العنصر

ويظهر في المثال ان كل عنصر له 10 قيم السبب ان في البداية يأخذ القيمة من البلوك:

***Number of Records Display: 10**

ويمكن التعديل من هنا بشرط ان لا تتجاوز القيمة المختارة في البلوك



Function
استرجاع القيم

PL/SQL Editor

Name: WHEN-NEW-BLOCK-IN

Type: Trigger Object: EMPLOYEES (Data Block I)

```
execute_query;
message('open first block');
message(' ');
```

يتم تنفيذ trigger عند الدخول الى البلوك

Not Modified Successfully Compiled

• Run

Oracle Developer Forms Runtime - Web

tion Edit Query Block Record Field Help Window



WINDOW1

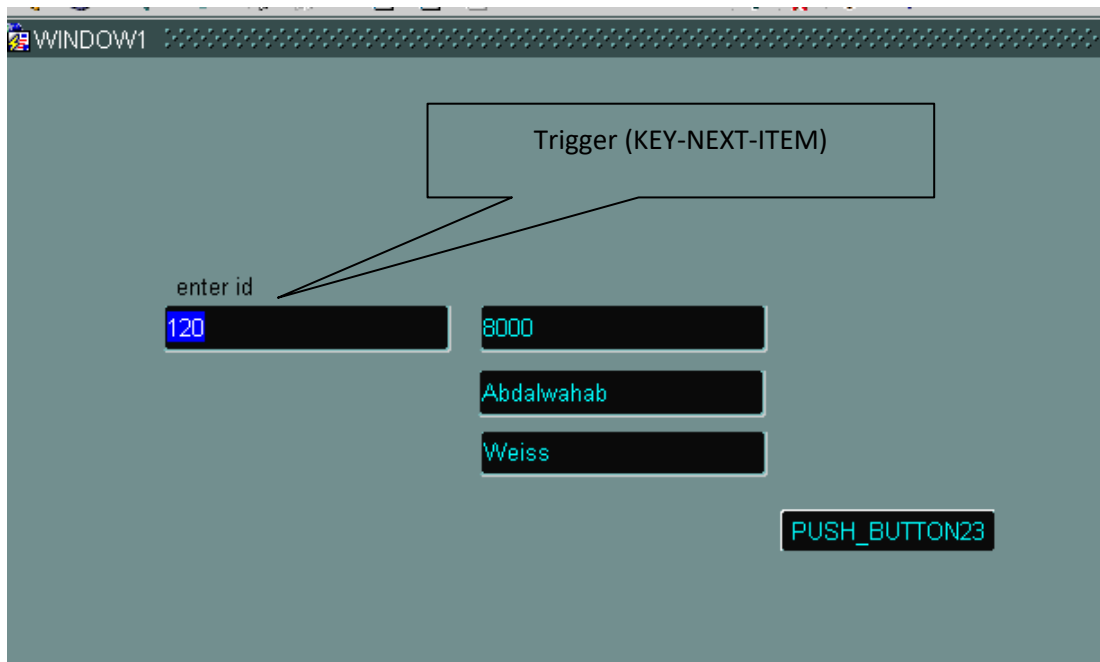
Steven	King	24000	515.123.4567
Neena	Kochhar	17000	515.123.4568
Lex	De Haan	17000	515.123.4569
Alexander	Hunold	9000	590.423.4567
Bruce	Ernst	6000	590.423.4568
David	Austin	4800	590.423.4569
Valli	Pataballa	4800	590.423.4560
Diana	Lorentz	4200	590.423.5567
Nancy	Greenberg	12008	515.124.4569
Daniel	Faviet	9000	515.124.4169

Forms

open first block

OK

Ex: Display the salary and the first name through the employee_id



1. First way:

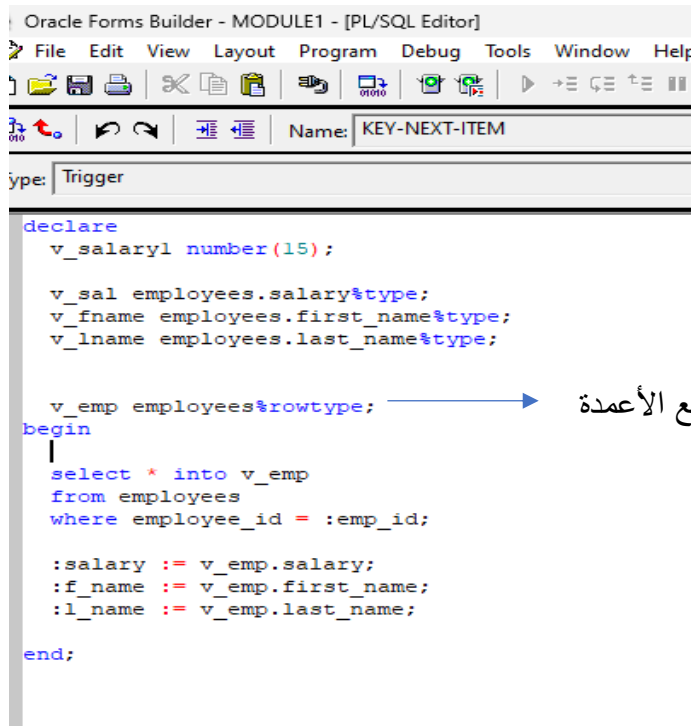
```
declare
  v_salary1 number(15);
  v_sal employees.salary%type;
  v_fname employees.first_name%type;
  v_lname employees.last_name%type;

  v_emp employees%rowtype;
begin
  select salary,first_name,last_name
  into v_sal,v_fname,v_lname
  from employees
  where employee_id = :emp_id;

  :salary := v_sal;
  :f_name := v_fname;
  :l_name := v_lname;
end;
```

تعريف متغيرات بشكل ديناميكي مع قاعدة البيانات

2.Second way:



```
Oracle Forms Builder - MODULE1 - [PL/SQL Editor]
File Edit View Layout Program Debug Tools Window Help
Name: KEY-NEXT-ITEM
Type: Trigger

declare
  v_salary1 number(15);

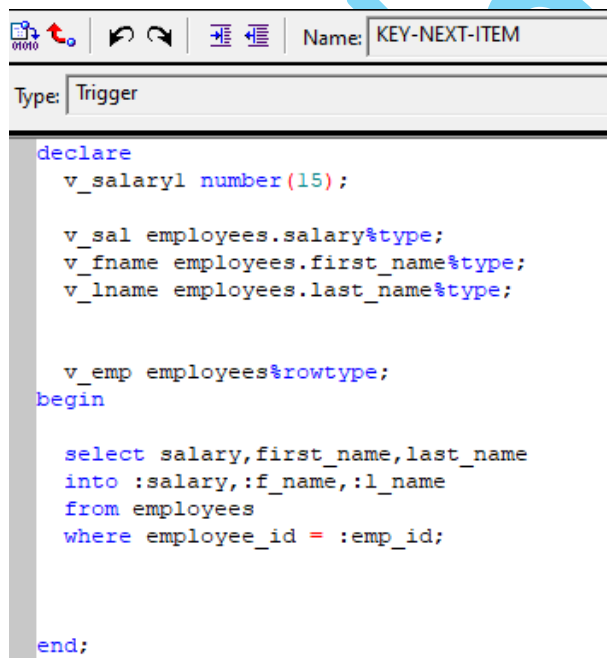
  v_sal employees.salary%type;
  v_fname employees.first_name%type;
  v_lname employees.last_name%type;

  v_emp employees%rowtype;
begin
  select * into v_emp
  from employees
  where employee_id = :emp_id;

  :salary := v_emp.salary;
  :f_name := v_emp.first_name;
  :l_name := v_emp.last_name;
end;
```

تعريف متغير يأخذ جميع الأعمدة

3. The third way



```
Name: KEY-NEXT-ITEM
Type: Trigger

declare
  v_salary1 number(15);

  v_sal employees.salary%type;
  v_fname employees.first_name%type;
  v_lname employees.last_name%type;

  v_emp employees%rowtype;
begin
  select salary,first_name,last_name
  into :salary,:f_name,:l_name
  from employees
  where employee_id = :emp_id;

end;
```

إعطاء قيم بشكل مباشر للعناصر

Ex:

Oracle Forms Builder - C:\Users\User\Desktop\Data Base\Data Base\5. EX all employees with departments\MODULE1.fmb

File Edit View Layout Program Debug Tools Window Help

Canvas: CANVAS3 Block: <Null>

Segoe UI 9 B I U

0 16 32 48 64 80 96 112 128 144 160 176 192 208 224 240 256 272 288 304 320 336 352 368 384 400 416 432 448 464 480 496 512 528 544 560 576 592 608 624 640 656 672 688 704

T1

D1

PL/SQL Editor

Name: KEY-NEXT-ITEM

Type: Trigger Object: BLOCK2 T1

```
declare
v_sal employees.salary%type;
v_fname employees.first_name%type;

v_emp employees%rowtype;

v_dept departments.department_name%type;
begin

select *
into v_emp
from employees
where employee_id =:T1;

select department_name
into v_dept
from departments
where department_id = v_emp.department_id;

:D1 := v_dept;

message('salary: '||v_emp.salary || ', First Name: '||v_emp.first_name);
message('Department Name: '|| v_dept);
message(' ');

end;
```

Not Modified Not Compiled

Ex:

ANVAS3 Block: <Null>

18 B I U [Icons] [Icons]

0 16 32 48 64 80 96 112 128 144 160 176 192 208 224 240 256 272 288 304 320 336 352 368 384 400 416 432 448 464 480 496 512 528 544 560 576 592 608 624 640 656 672 688 704

Enter ID Salary

T1 D2

D1 Salary

PL/SQL Editor

Name: KEY-NEXT-ITEM

Type: Trigger Object: BLOCK2 T1

```
declare
v_sal employees.salary%type;
v_Fname employees.first_name%type;
begin

select salary, First_name
into v_sal , v_Fname
from employees
where employee_id = :T1;

:D2 := v_sal;
message('salary: ' || v_sal || ' First Name: ' || v_Fname);
message(' ');

end;
```

Not Modified Not Compiled

Enter ID Salary

T1 D2

D1 Salary

PL/SQL Editor

Name: WHEN-BUTTON-PRESSED

Type: Trigger Object: BLOCK2 PUSH_BUTTON6

```
declare
v_sal employees.salary%type;
begin
select salary
into v_sal
from employees
where employee_id=:T1;

:D1 :=v_sal;

end;
```

Not Modified Not Compiled

Ex:

EMP_ID

Salary T1

Fisrt_Name T2

Phone T3

PL/SQL Editor

Name: WHEN-BUTTO

Type: Trigger Object: BLOCK2 D1

```
declare
  v_sal employees.salary%type;
begin
  select salary
  into v_sal
  from employees
  where employee_id=:emp_id;

  :T1 :=v_sal;
end;
```

Not Modified Not Compiled

EMP_ID

Salary T1

Fisrt_Name T2

Phone T3

PL/SQL Editor

Name: WHEN-BUTTO

Type: Trigger Object: BLOCK2 PUSH_BU

```
declare
  v_fname employees.first_name%type;
begin
  select first_name
  into v_fname
  from employees
  where employee_id=:emp_id;

  :T2 :=v_fname;
end;
```

Not Modified Not Compiled

EMP_ID

Salary T1

Fisrt_Name T2

Phone T3

PL/SQL Editor

Name: WHEN-BUTTO

Type: Trigger Object: BLOCK2 PUSH_BU

```
declare
  phone employees.phone_number%type;
begin
  select phone_number
  into phone
  from employees
  where employee_id=:emp_id;

  :T3 :=phone;
end;
```

Not Modified Not Compiled

If statement:

The screenshot shows a web application interface on the left and a PL/SQL Editor window on the right. The web application has input fields for 'Enter id' (containing 'T1'), 'Name' (containing 'D1'), 'salary' (containing 'D2'), and 'new salary' (containing 'D3'). There are buttons for 'rollback', 'update and commit', and 'update'. A blue arrow points from the 'update' button to the PL/SQL Editor. The PL/SQL Editor window is titled 'PL/SQL Editor' and shows a trigger named 'KEY-NEXT-ITEM' of type 'Trigger' on object 'BLOCK2' at 'T1'. The code in the editor is as follows:

```
declare
begin
    select salary, First_name
    into :D2, :D1
    from employees
    where employee_id=:T1;

    if :D2<2000 then
        :D3 := :D2 *1.25;
    elsif :D2 between 2000 and 3000 then
        :D3 := :D2*1.15;
    else
        :D3:=:D2*1.01;
    end if;

end;
```

The status bar at the bottom of the editor shows 'Not Modified' and 'Not Compiled'.

- Update/ Commit/ Rollback

The screenshot shows the same web application interface as before, but with the 'Update' button circled in blue. The PL/SQL Editor window is titled 'PL/SQL Editor' and shows a trigger named 'WHEN-BUTTON-PRESSED' of type 'Trigger' on object 'BLOCK2' at 'PUSH_BUTTON'. The code in the editor is as follows:

```
update employees
set salary=:D3
where employee_id=:T1;

clear_item;
```

The status bar at the bottom of the editor shows 'Not Modified' and 'Not Compiled'. The text 'تحديث القيمة' (Update the value) is written in Arabic next to the code.

Enter id
T1

Name D1

salary D2

new salary D3

Update

rollback

commit

update and commit

PL/SQL Editor

Type: Trigger

Obj

commit;

Not Modified

حفظ التحديث

T1

Name D1

salary D2

new salary D3

Update

rollback

commit

update and commit

PL/SQL Editor

Name: V

Type: Trigger

Object: BLOC

rollback;

Not Modified

الغاء التحديث

يجب ان تكون قبل الحفظ

Enter id
T1

Name D1

salary D2

new salary D3

Update

rollback

commit

update and commit

PL/SQL Editor

Name: WHEN-BUTTON-PRESSED

Type: Trigger

Object: BLOCK2

P

begin

update employees

set salary= :D3

where employee_id= :t1;

commit;

Not Modified

في حال كتابتها في هذه الحالة نستغني عن

Rollback

Ex:

PL/SQL Editor

Name: POST-TEXT-ITEM

Type: Trigger Object: BLOCK2 T1

```
declare
begin
    select salary, first_name
    into :osal, :i_fname
    from employees
    where employee_id=:T1;

    if :osal < 2000 then
        :nsal := :osal * 1.25;
    elsif :osal between 2000 and 3000 then
        :nsal := :osal * 1.15;
    else
        :nsal := :osal * 1.01;
    end if;

    end;
```

Not Modified Not Compiled

PL/SQL Editor

Name: WHEN-BUTTON-PRESSED

Type: Trigger Object: BLOCK2 PUSH_BUTTON8

```
declare
begin
    update employees
    set salary =:nsal
    where employee_id=:T1;

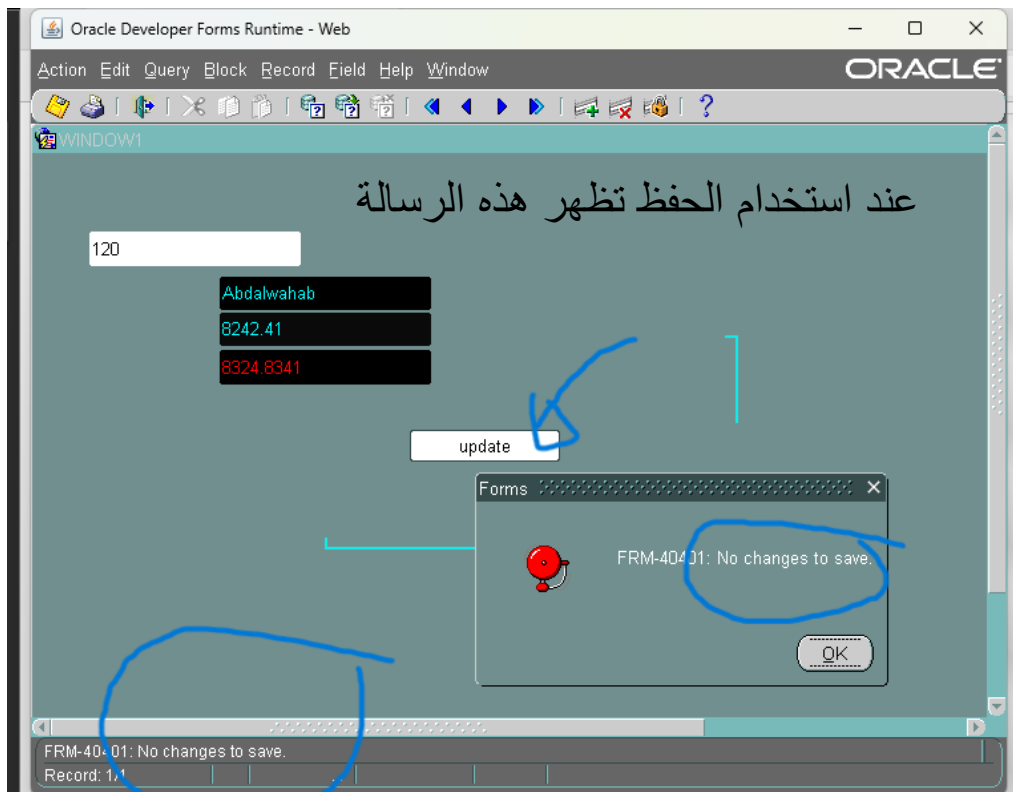
    commit;

    message('Salary has been updated');
    message(' ');

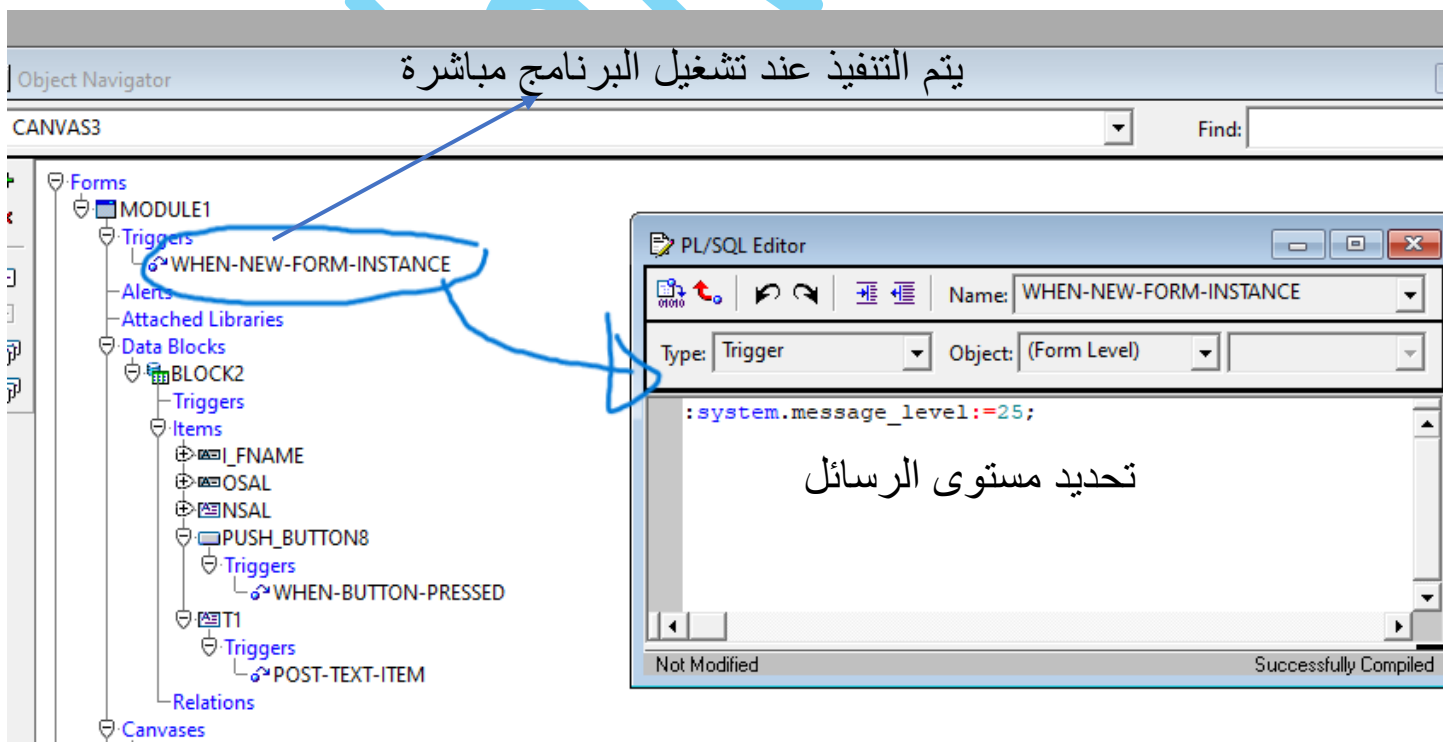
    end;
```

Not Modified Not Compiled

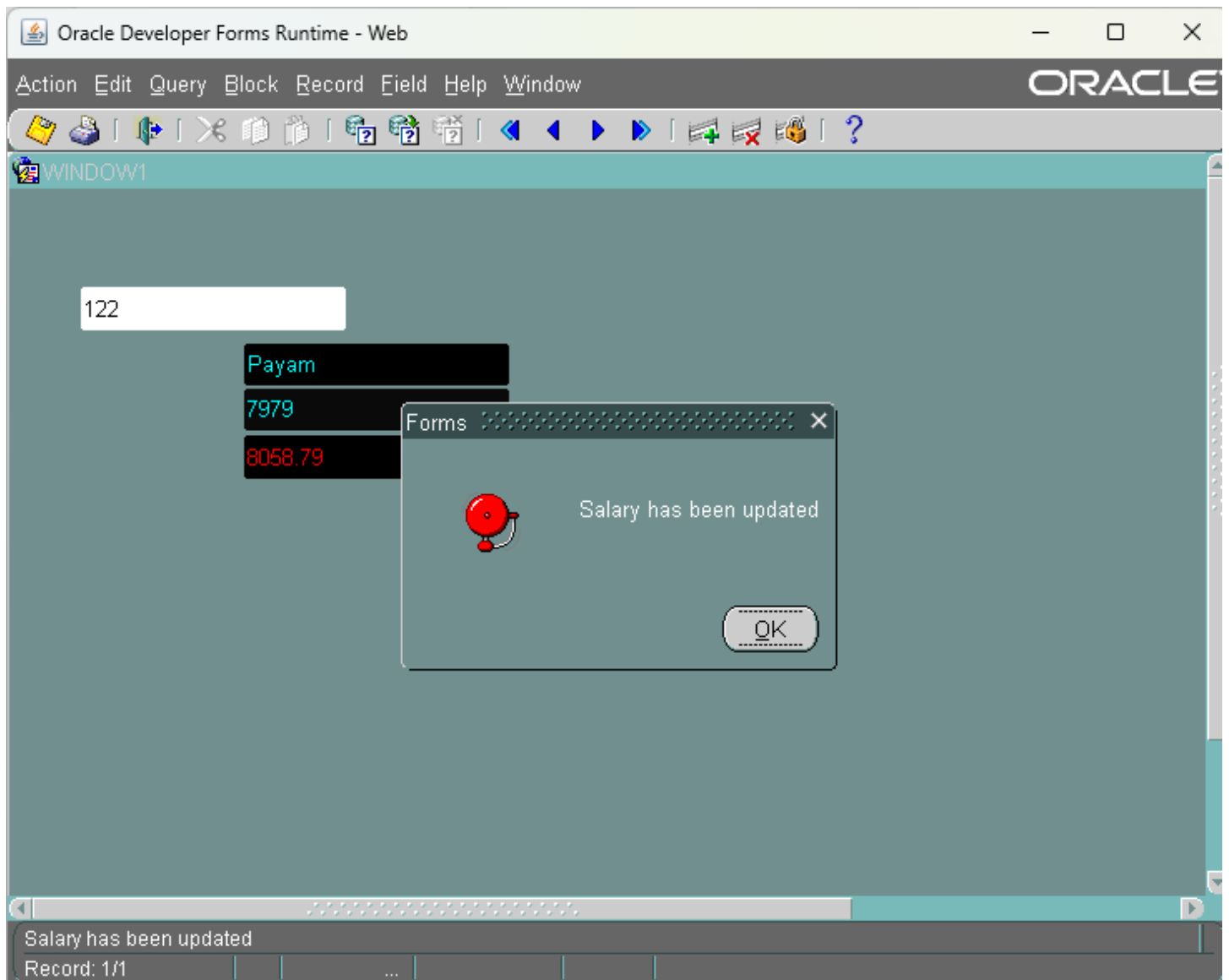
Run:



Solution: للتخلص من الرسائل غير المرغوب فيها نحدد مستوى الرسائل



After:



Ex (BLOCKS):

The image illustrates the implementation of a salary calculation logic using Oracle Forms blocks and PL/SQL triggers.

Main Form Canvas (BLOCK2):

- Fields: Enter ID (T1), First Name (D1), salary (D2), new salary (D3).
- Buttons: Update, save.
- Triggers: KEY-NEXT-ITEM (on T1), WHEN-BUTTON (on Update), WHEN-BUTTON-PRESSED (on save).

PL/SQL Editor - KEY-NEXT-ITEM Trigger:

```
declare
  num number(10) := 0;
begin
  select salary, first_name
  into num, :D1
  from employees
  where employee_id = :T1;

  :D2 := num;

  if num < 2000 then
    :D3 := num * 1.25;
  elsif num between 2000 and 3000 then
    :D3 := num * 1.15;
  else
    :D3 := num * 1.05;
  end if;
end;
```

PL/SQL Editor - WHEN-BUTTON Trigger:

```
commit;
```

PL/SQL Editor - WHEN-BUTTON-PRESSED Trigger:

```
declare
begin
  update employees
  set salary = :D3
  where employee_id = :T1;

  --go_item('T2');
end;
```

Arrows indicate the flow of data and control: from the KEY-NEXT-ITEM trigger to the WHEN-BUTTON trigger, and from the WHEN-BUTTON-PRESSED trigger to the WHEN-BUTTON trigger.

The **Time** value is given when entering block 17.

Using when new block instance

Canvas: CANVAS3 Block: BLOCK17

Enter ID T1

First Name D1

salary D2

new salary D3

save

TIME

T2

1: CANVAS3 (BLOCK2)

ANVAS3 Block: BLOCK2

Enter ID T1

First Name D1

salary D2

new salary D3

Update

TIME

T2

save

Oracle Developer Forms Design View

Object Navigator: WHEN-NEW-BLOCK-INSTANCE

- Triggers
 - Items
 - D1
 - D2
 - T1
 - Triggers
 - KEY-NEXT-ITEM
 - D3
 - PUSH_BUTTON9
 - Triggers
 - WHEN-BUTTON-PRESSED
 - PUSH_BUTTON10
 - Triggers
 - WHEN-BUTTON-PRESSED
 - TIME
 - Relations
 - BLOCK17
 - Triggers
 - WHEN-NEW-BLOCK-INSTANCE

PL/SQL Editor: WHEN-NEW-BLOCK-INSTANCE

```
declare
begin
  :Time := sysdate;
end;
```

Form Fields:

 - Enter ID: T1
 - First Name: D1
 - salary: D2
 - new salary: D3
 - Buttons: save, Update

Oracle Developer Forms Runtime - Web

WINDOW1

20-APR-23

Enter ID: [Redacted]

First Name: [Redacted]

salary: [Redacted]

new salary: [Redacted]

Buttons: save, Update

Annotation: عند الضغط يعطي القيمة (When pressed, it gives the value)

Go_item/go_block

T1	Block 2
T2	Block 3
T3	Block 4

which block are you in now?

D1	Block 2
----	---------

go first block go second block go third block

WHEN-BUTTON-PRESSED

Writing

`go_block('block4');`

or

`go_item('T3');`

WHEN-BUTTON-PRESSED

Writing

`go_block('block3');`

or

`go_item('T2');`

which block are you in now?

D1

Block 2

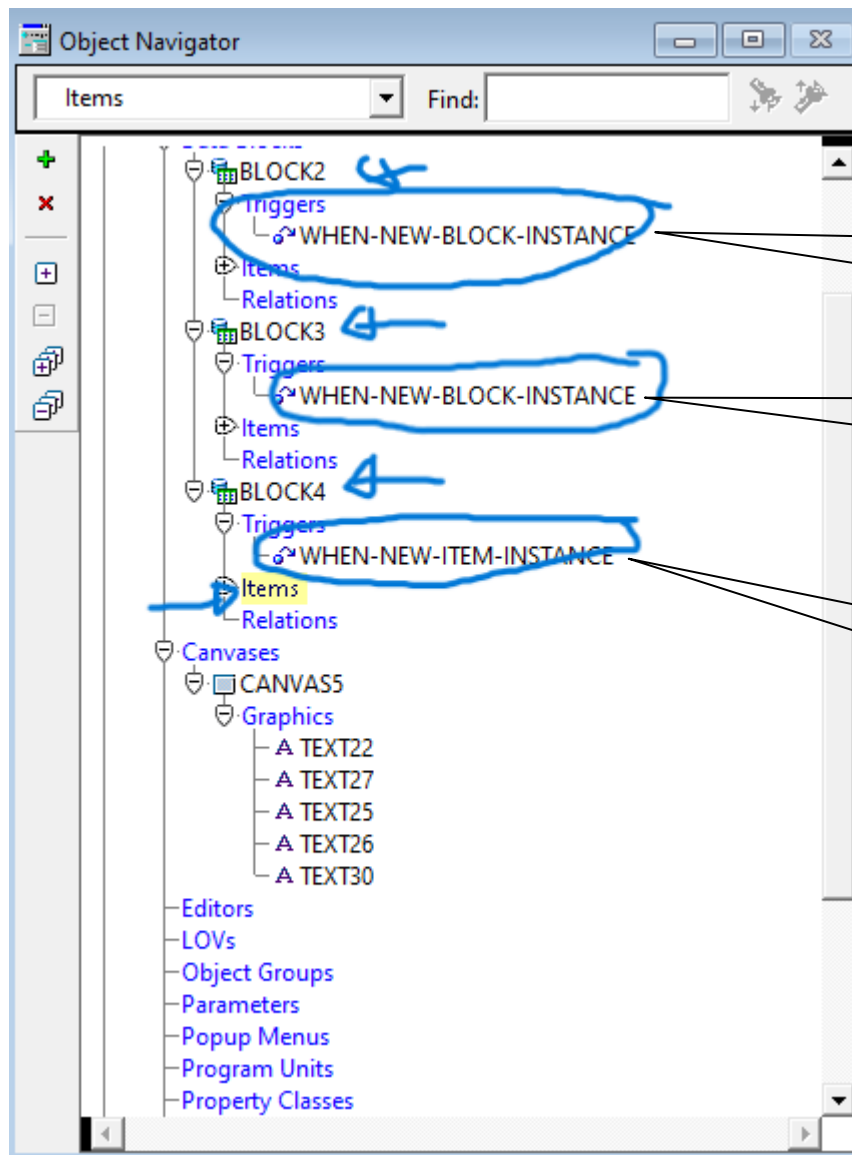
تحديد العنصر في اي بلوك

عند الدخول الى البلوك يعطي اسمه لي:

D1

Trigger:

1. When-new-**block**-instance
2. When-new-**item**-instance
3. When-new-**form**-instance
4. When-new-**record**-instance



عند الدخول الى البلوك 2 اعطي اسمه

--Pl/sql editor

:D1 := 'BLOCK2';

عند الدخول الى البلوك 3 اعطي اسمه

--Pl/sql editor

:D1 := 'BLOCK3';

عند الدخول الى البلوك 4 اعطي اسمه

--Pl/sql editor

:D1 := 'BLOCK4';

File (8.go item)

Ex:

which block are you now?
SHOWBLOCK

enter employee id
EMP_ID

Salary
OSAL

enter new salary
NEW_SAL

Update back

save

PL/SQL Editor

Name: KEY-NEXT-ITEM

Type: Trigger Object: BLOCK2 EMP_ID

```
declare
begin
  select salary
  into :osal
  from employees
  where employee_id=:emp_id;
end;
```

which block are you now?
SHOWBLOCK

enter employee id
EMP_ID

Salary
OSAL

enter new salary
NEW_SAL

Update back

save

PL/SQL Editor

Name: WHEN-BUTTON-PRESSED

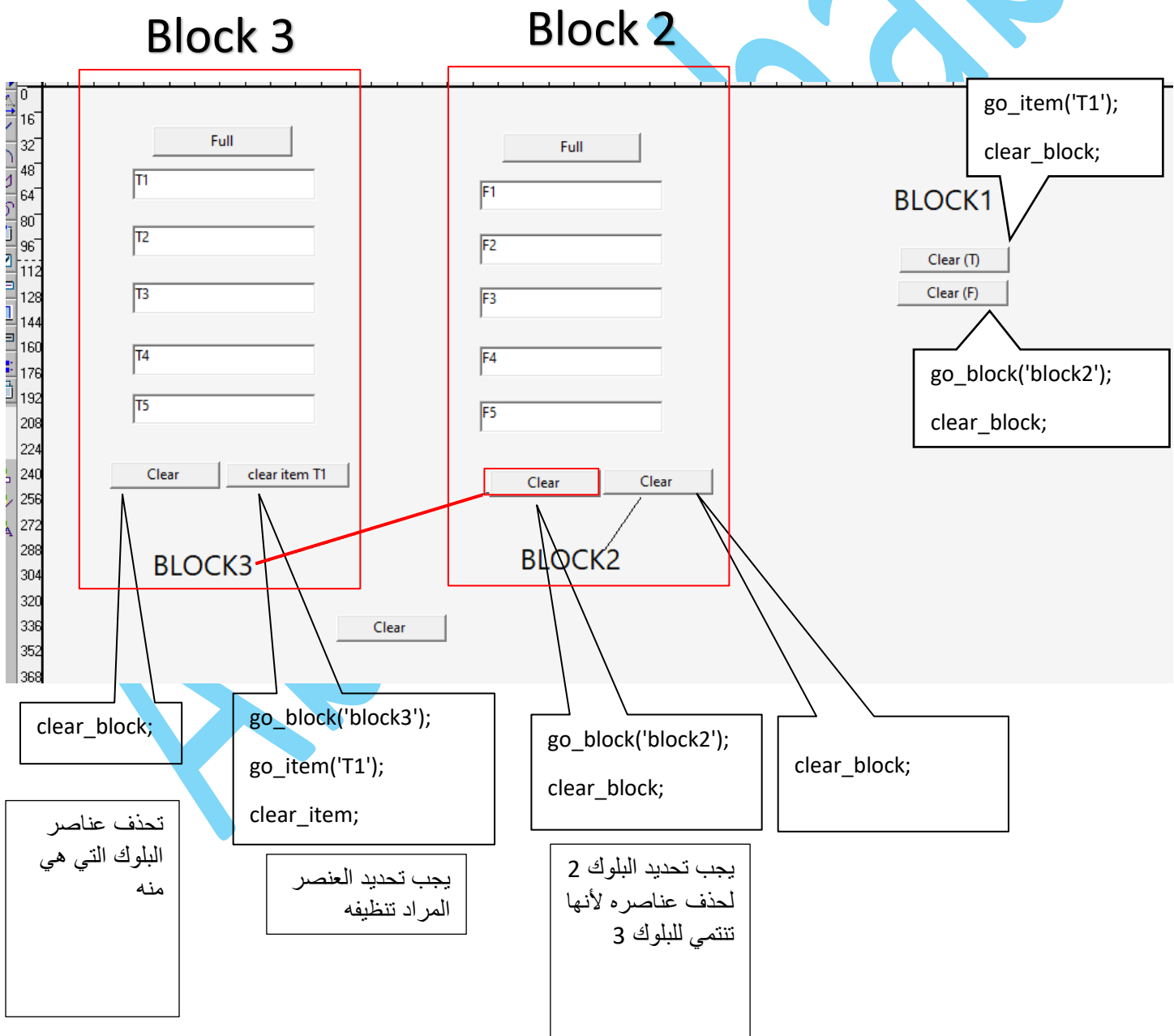
Type: Trigger Object: BLOCK4 PUSH_BUTTON

```
declare
begin
  update employees
  set salary = :new_sal
  where employee_id=:emp_id;

  MESSAGE('salary was updated');
  message(' ');
end;
```

Clear

1. Clear block
2. Clear item
3. Clear form
4. Clear list
5. Clear record



Example:

Oracle Forms Builder - C:\Users\User\Desktop\Data Base\Data Base\10. ex update&clear&goltem\ex.fmb

File Edit View Layout Program Debug Tools Window Help

Canvas: CANVAS3 Block: <Null>

Segoe UI

SHOWBLOCK

Suggestions
122 120

Enter employee id
T1

salary :_SAL
First Name :_FNAME

Enter new Name
NEW_NAME save

Clear Form

Clear_form;

تنظيف جميع
العناصر من جميع
البلوكات

PL/SQL Editor

Name: KEY-NEXT-ITEM

Type: Trigger Object: BLOCK2 T1

```
declare
begin
select first_name, salary
into :i_fname, :i_sal
from employees
where employee_id =:T1;

end;
```

Not Modified Not Compiled

egoe UI

9

SHOWBLOCK

Suggestions
122 120

Enter employee id
T1

salary
_SAL

First Name
_FNAME

Enter new Name
NEW_NAME

save

Clear Form

PL/SQL Editor

Name: WHEN-BUTTON-PRESSED

Type: Trigger Object: BLOCK2 PUSH_BUTTON13

```
T1 := 120;  
go_item('T1');
```

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

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

SHOWBLOCK

Suggestions
122 120

Enter employee id
T1

salary |_SAL
First Name |_FNAME

Commit;

Enter new Name
NEW_NAME

save

Clear Form

Clear_form;

PL/SQL Editor

Name: KEY-NEXT-ITEM

Type: Trigger Object: BLOCK2 NEW_NAME

```
declare
begin
  update employees
  set first_name =:new_name
  where employee_id =:T1;

  message('Updated the first name');
  message(' ');

end;
```

Not Modified

The screenshot displays the Oracle Forms Builder interface. On the left, a form is visible with a 'Suggestions' list containing '122' and '120'. Below this is a text field 'Enter employee id' with 'T1' entered. Further down are fields for 'salary' (containing '_SAL') and 'First Name' (containing '_FNAME'). To the right of these are fields for 'Enter new Name' (containing 'NEW_NAME') and a 'save' button. A 'Clear Form' button is also present. In the center, a 'PL/SQL Editor' window is open, showing a trigger named 'WHEN-NEW-BLOCK-INST...' with the following code:

```
showblock := 'Block(2)';
```

The 'Object Navigator' on the right shows the hierarchy: Forms > EX > Triggers > BLOCK2 > WHEN-NEW-BLOCK-INSTANCE. The 'WHEN-NEW-BLOCK-INSTANCE' trigger is highlighted with a blue circle.

Example:

The screenshot shows the Oracle Forms Builder interface for a form named 'MODULE1.fmb'. The 'Canvas' is set to 'CANVAS3' and the 'Block' is '<Null>'. The form contains a 'Segoe UI' text field with '120' and '122' entered. Below this is a 'CLEAR' button. To the right are five text fields labeled 'D2', 'D3', 'D4', and 'D5', with a 'CLEAR' button at the bottom right. In the center, a 'PL/SQL Editor' window is open, showing a trigger named 'WHEN-BUTTON-PRESSED' with the following code:

```
:T1:=120;  
go_item('T1');
```

The 'Object Navigator' on the right shows the hierarchy: Forms > EX > Triggers > BLOCK2 > WHEN-BUTTON-PRESSED. The 'WHEN-BUTTON-PRESSED' trigger is highlighted with a blue circle.

Oracle Forms Builder - C:\Users\User\Desktop\Data Base\Data Base\10.example\MODULE1.fmb

Edit View Layout Program Debug Tools Window Help

Canvas3 Block: <Null>

9 B I U

120 T1 D2
122 CLEAR D3
D4
D5 CLEAR

SHOWBLOCK

PL/SQL Editor

Name: KEY-NEXT-ITEM

Type: Trigger Object: BLOCK2 T1

```
declare
v_emp employees%rowtype;
begin
select *
into v_emp
from employees
where employee_id =:T1;

:D2 :=v_emp.First_Name;
:D3 :=v_emp.Last_Name;
:D4 :=v_emp.salary;
:D5 :=v_emp.phone_number;

message('salary :'||v_sal);
message(' ');
```

Oracle Forms Builder - C:\Users\User\Desktop\Data Base\Data Base\10.example\MODULE1.fmb

Edit View Layout Program Debug Tools Window Help

Canvas3 Block: <Null>

9 B I U

120 T1 D2
122 CLEAR D3
D4
D5 CLEAR

SHOWBLOCK

PL/SQL Editor

Name: WHEN-BUTTON-PRESSED

Type: Trigger Object: BLOCK2 PUSH_BUTTON1

```
go_item('T1');
clear_item;
go_item('T1');
```

Edit View Layout Program Debug Tools Window Help

Canvas: CANVAS3 Block: <Null>

Page: 9

Canvas UI

120 T1 CLEAR D2 D3 D4 D5 CLEAR

SHOWBLOCK

Same block

PL/SQL Editor

Name: WHEN-BUTTON-PRESSED

Type: Trigger Object: BLOCK14 PUSH_BUTTON

```
clear_block;
go_item('T1');
```

Not Modified Not Compiled

Oracle Forms Builder - C:\Users\User\Desktop\Data Base\Data Base\10.example\MODULE1.fmb

Edit View Layout Program Debug Tools Window Help

Canvas: CANVAS3 Block: <Null>

Page: 9

Canvas UI

120 T1 CLEAR D2 D3 D4 D5 CLEAR

SHOWBLOCK

Object Navigator

WHEN-NEW-BLOCK-INSTANCE

Find:

Forms

- MODULE1
 - Triggers
 - Alerts
 - Attached Libraries
 - Data Blocks
 - BLOCK2
 - Triggers
 - WHEN-NEW-BLOCK-INSTANCE
 - BLOCK14
 - Triggers
 - WHEN-NEW-BLOCK-INSTANCE
 - Canvases
 - CANVAS3

PL/SQL Editor

Name: WHEN-NEW-BLOCK-INST

Type: Trigger Object: BLOCK2 (Data Block L)

```
:showblock := 'Block(2)';
```


Builder - C:\Users\User\Desktop\Data Base\Data Base\10.example\MODULE1.fmb

Layout Program Debug Tools Window Help

Block: <Null>

9 B I U

120 T1 D2
122 CLEAR D3
D4
D5 CLEAR

SHOWBLOCK

PL/SQL Editor

Name: WHEN-NEW-BLOCK-INST

Type: Trigger Object: BLOCK14 (Data Block L)

```
showblock := 'block(14)';
```

Not Modified Successfully Compiled

Object Navigator

WHEN-NEW-BLOCK-INSTAN Find:

- Forms
 - MODULE1
 - Triggers
 - Alerts
 - Attached Libraries
 - Data Blocks
 - BLOCK2
 - Triggers
 - WHEN-NEW-BLOCK-INSTANCE
 - BLOCK14
 - Triggers
 - WHEN-NEW-BLOCK-INSTANCE
 - Items
 - Relations
 - Canvases
 - CANVAS3
 - Graphics
 - Editors
 - LOVs
 - Object Groups
 - Parameters
 - Popup Menus
 - Program Units

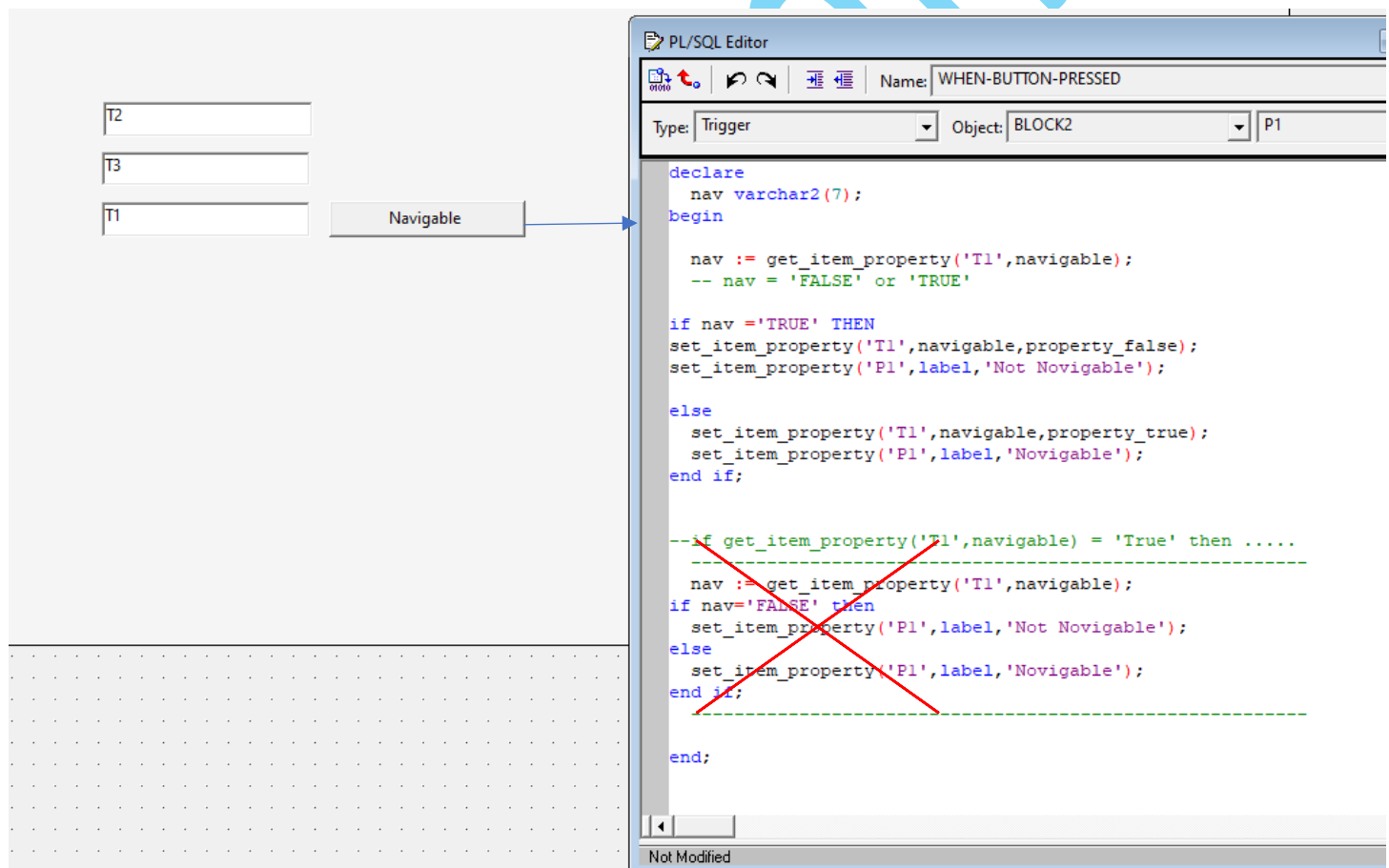
Set/get.

Set_item_property ('Name-item', property-name, value) تعديل القيمة

Get_item_property ('Name-item', property-name) استرجاع القيمة

Yes/no	Value (set)	Return (Get)
Yes	Property_true	'TRUE'
No	Property_false	'FALSE'

Ex:



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

- Title Bar:** PL/SQL Editor
- Toolbar:** Includes icons for file operations, execution, and navigation.
- Properties:** Name: WHEN-BUTTON-PRESSED, Type: Trigger, Object: BLOCK2, P1.
- Code:**

```
declare
  nav varchar2(7);
begin
  nav := get_item_property('T1',navigable);
  -- nav = 'FALSE' or 'TRUE'

  if nav = 'TRUE' THEN
    set_item_property('T1',navigable,property_false);
    set_item_property('P1',label,'Not Navigable');
  else
    set_item_property('T1',navigable,property_true);
    set_item_property('P1',label,'Navigable');
  end if;

  --if get_item_property('T1',navigable) = 'True' then ....
  nav := get_item_property('T1',navigable);
  if nav='FALSE' then
    set_item_property('P1',label,'Not Navigable');
  else
    set_item_property('P1',label,'Navigable');
  end if;

end;
```
- Status Bar:** Not Modified

The UI on the left shows three text boxes labeled T2, T3, and T1. A button labeled 'Navigable' is positioned next to T1. A blue arrow points from the 'Navigable' button to the trigger code.

Visible/ enable/ navigable.

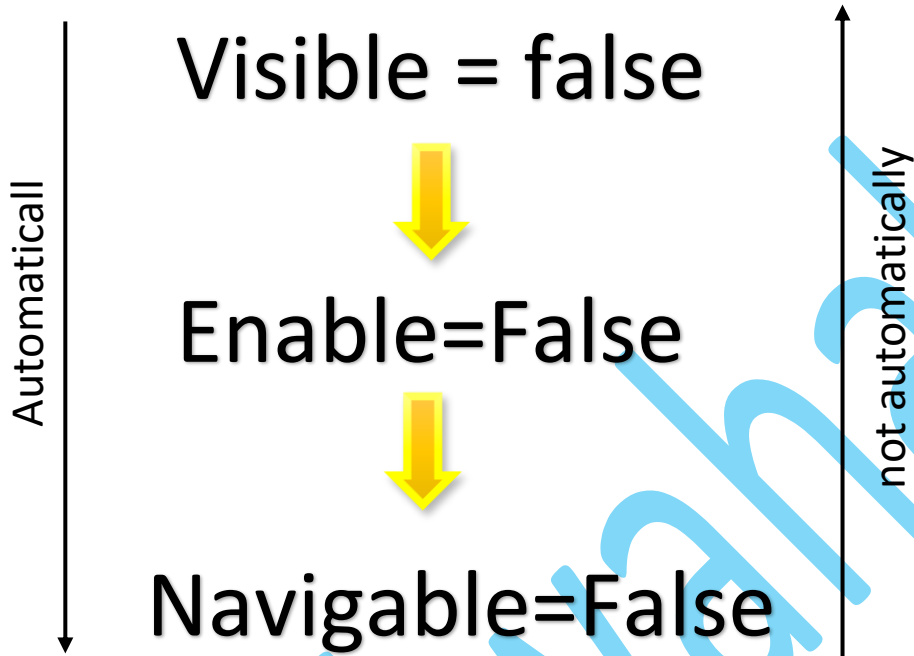


Diagram illustrating the implementation of the 'Visible = false' condition in a PL/SQL trigger:

The diagram shows a form with three text input fields labeled T1, T2, and T3. Below the fields are two sets of buttons: 'single' (Enable, Disable) and 'multiple' (Disable T1, Disable T2, Disable T3). A blue arrow points from the 'Disable' button in the 'single' group to the PL/SQL Editor window.

The PL/SQL Editor window displays the following code:

```
PL/SQL Editor
Name: WHEN-BUTTON-PRESSED
Type: Trigger
Object: BLOCK4
P2

set_item_property('T1',enabled,property_false);
set_item_property('T2',enabled,property_false);
set_item_property('T3',enabled,property_false);
```

A blue callout box indicates: Navigable is false

The Navigable must be modified

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

- Name:** WHEN-BUTTON-PRESSED
- Type:** Trigger
- Object:** BLOCK4
- P1:**

```
set_item_property('T1',enabled,property_true);
set_item_property('T1',navigable,property_true);

set_item_property('T2',enabled,property_true);
set_item_property('T2',navigable,property_true);

set_item_property('T3',enabled,property_true);
set_item_property('T3',navigable,property_true);
```

The UI on the left shows a form with three text input fields labeled T1, T2, and T3. Above these fields are two buttons: 'Enable' and 'Disable'. A blue arrow points from the 'Enable' button to the trigger code. Below the input fields are three buttons labeled 'Disable T1', 'Disable T2', and 'Disable T3'.

Abdallah

PUSH_BUTTON8

T1

T2

T3

single

Enable Disable

multiple

Disable T1 Disable T2 Disable T3

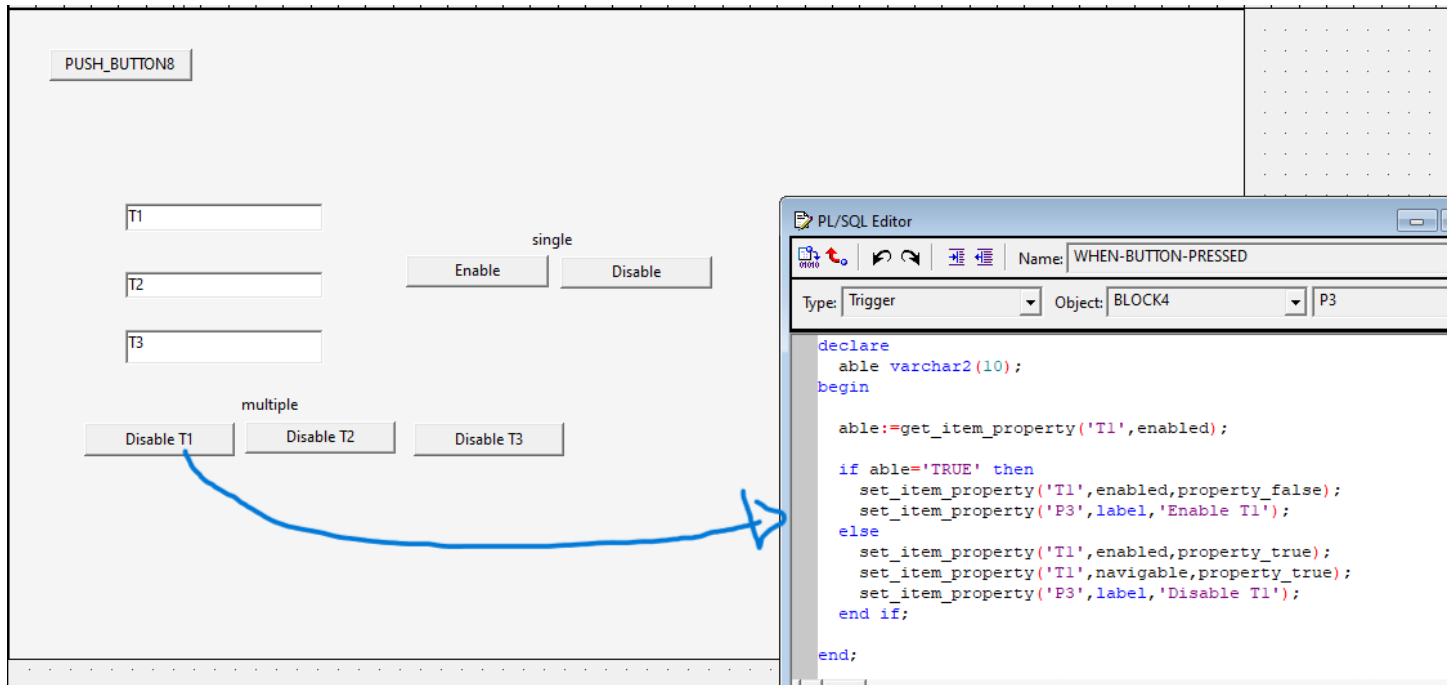
PL/SQL Editor

Name: WHEN-BUTTON-PRESSED

Type: Trigger Object: BLOCK4 P3

```
declare
  able varchar2(10);
begin
  able:=get_item_property('T1',enabled);

  if able='TRUE' then
    set_item_property('T1',enabled,property_false);
    set_item_property('P3',label,'Enable T1');
  else
    set_item_property('T1',enabled,property_true);
    set_item_property('T1',navigable,property_true);
    set_item_property('P3',label,'Disable T1');
  end if;
end;
```



PUSH_BUTTON8

T1

T2

T3

single

Enable Disable

multiple

Disable T1 Disable T2 Disable T3

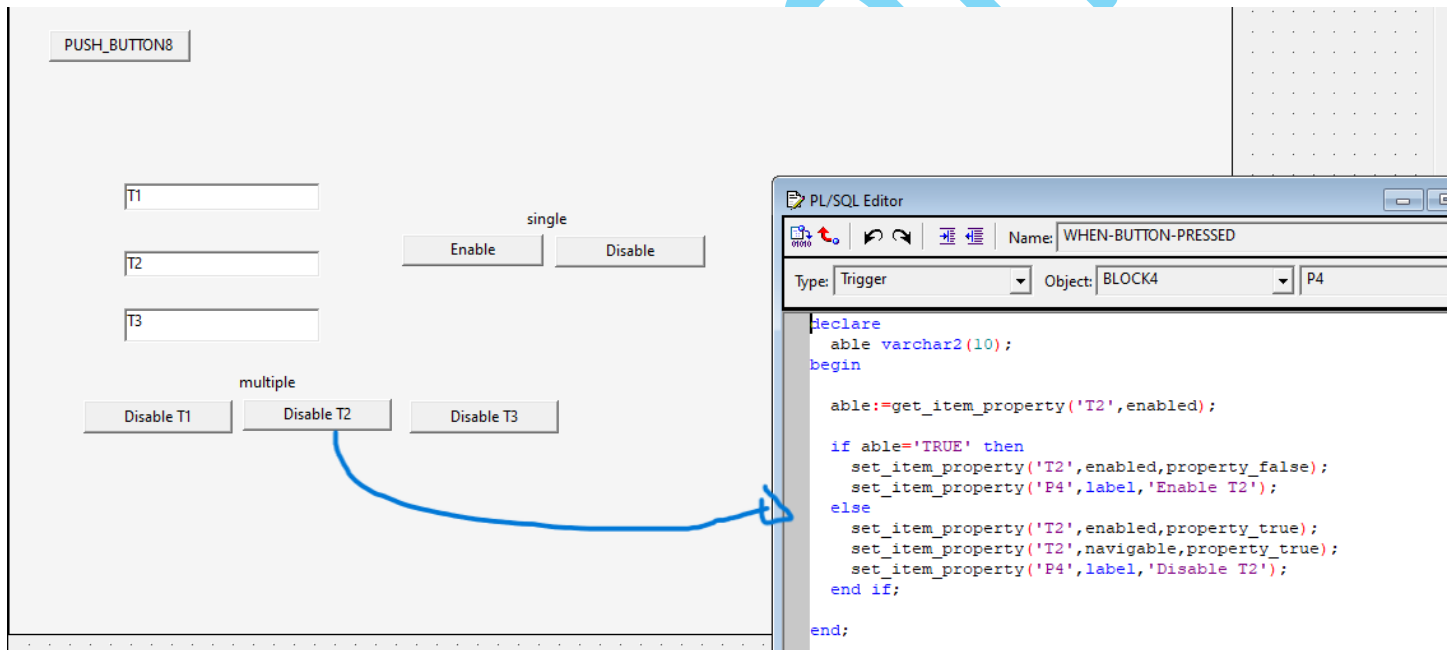
PL/SQL Editor

Name: WHEN-BUTTON-PRESSED

Type: Trigger Object: BLOCK4 P4

```
declare
  able varchar2(10);
begin
  able:=get_item_property('T2',enabled);

  if able='TRUE' then
    set_item_property('T2',enabled,property_false);
    set_item_property('P4',label,'Enable T2');
  else
    set_item_property('T2',enabled,property_true);
    set_item_property('T2',navigable,property_true);
    set_item_property('P4',label,'Disable T2');
  end if;
end;
```



0 16 32 48 64 80 96 112 128 144 160 176 192 208 224 240 256 272 288 304 320 336 352 368 384 400 416 432 448 464 480 496 512 528 544 560 576 592 608 624 640 656 672 688 704 720 736 752 768 784 800 816 8

PUSH_BUTTON8

T1

T2

T3

single

Enable Disable

multiple

Disable T1 Disable T2 Disable T3

PL/SQL Editor

Name: WHEN-BUTTON-PRESSED

Type: Trigger Object: BLOCK4 P5

```
declare
begin
if get_item_property('T3',enabled)='TRUE' then
set_item_property('T3',enabled,property_false);
set_item_property('P5',label,'Enable T3');
else
set_item_property('T3',enabled,property_true);
set_item_property('T3',navigable,property_true);
set_item_property('P5',label,'Disable T3');
end if;
end;
```

File (Example Doctor)

Exception

1. No-data-found.
2. Value-error

The screenshot displays the Oracle APEX development environment. On the left, the 'Canvas' editor shows a form with two input fields: 'enter ID' (containing 'T1') and 'salary' (containing 'D1'). A blue line connects the 'enter ID' field to the 'salary' field, indicating a data flow or validation rule. On the right, the 'Property Palette' and 'PL/SQL Editor' are visible.

Property Palette (Item: D1):

Property	Value
Navigation	
Previous Navigation Item	<Null>
Next Navigation Item	<Null>
Data	
Data Type	Number
Data Length Semantics	Null
Maximum Length	5
Initial Value	
Format Mask	
Copy Value from Item	
Synchronize with Item	<Null>
Calculation	

PL/SQL Editor:

Name: KEY-NEXT-ITEM
Type: Trigger
Object: BLOCK2
Trigger: T1

```
declare
  v_sal number(3);
begin
  --Welcome, Abdalwahab Qataweh
  --test 1 and 120
  select salary into v_sal
  from employees where employee_id=:T1;

  :D1:=v_sal;
exception
  when no_data_found then
    message('No Data Found');message(' ');
  when value_error then
    message('Value Error');message(' ');
end;
```

Nested exception:

The screenshot displays the Oracle Developer environment. On the left, the 'Canvas3 (BLOCK2)' window shows a form with two input fields: 'enter ID T1' and 'salary D1'. A blue arrow points from the 'enter ID T1' field to the PL/SQL Editor window on the right. The PL/SQL Editor window shows a trigger named 'KEY-NEXT-ITEM' of type 'Trigger' attached to 'BLOCK2' and 'T1'. The trigger code is as follows:

```
declare
    v_sal number(10);
begin
    --Welcome, Abdalwahab Qatawneh

    select salary into v_sal
    from employees where employee_id=:T1;

    :D1:=v_sal;

    --create table employee as(select *from employees);-> SQL/plus
    --then insert new employees
exception
    when no_data_found then

begin
    select salary into v_sal
    from employee
    where employee_id=:T1;
exception
    when no_data_found then
        message('this Employee not found');message(' ');
        clear_form;go_item('T1');
    end;

    when value_error then
        message('Value Error');message(' ');
        clear_form;go_item('T1');
    end;
end;
```

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

3. Too_many_rows

The screenshot displays the PL/SQL Developer interface. On the left, the 'PL/SQL Editor' window shows a trigger named 'T1' on 'BLOCK2'. The trigger code is as follows:

```
declare
begin
select First_Name into :D1
from employees where salary=:T1;

--ex too_many_rows -> 4800
--ex single salary ->130000
exception
when too_many_rows then
select count(*), sum(salary)
into :D2,:D3
from employees where salary=:T1;

end;
```

On the right, the 'SQL Worksheet' window shows the execution results of the trigger. The 'enter salary' field is set to 'T1', and the 'First Name' field is set to 'D1'. The 'too many row' message is displayed. The 'Count' field is set to 'D2' and the 'Sum' field is set to 'D3'. The formula 'sum = count * salary' is shown. The 'SQL Worksheet' window also displays a list of rows with columns 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.

Multiple-select to single-select.

Canvas2 (BLOCK4)

INVAS2

Block: <Null>

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

enter id

EMP_ID

FIRST_NAME

T_NAME

enter salary

SALARY

COUNT1

SUM1

Clear

PL/SQL Editor

Name: KEY-NEXT-ITEM

Type: Trigger

Object: BLOCK4

EMP_ID

```
declare
table_name varchar2(30);
begin
select first_name into :first_name
from employees where employee_id=:emp_id;

:t_name:='(EMPLOYEES) 1';

exception
when no_data_found then

begin
select first_name into :first_name
from employee where employee_id=:emp_id;
:t_name:='(EMPLOYEE) 2';
-- or using table_name

exception
when no_data_found then
message('Not found the ID');message(' ');
clear_block;go_item('emp_id');:t_name:='NULL';

end;

end;
```

Canvas2 (BLOCK4)

INVAS2

Block: <Null>

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

enter id

EMP_ID

FIRST_NAME

T_NAME

enter salary

SALARY

COUNT1

SUM1

Clear

PL/SQL Editor

Name: KEY-NEXT-ITEM

Type: Trigger

Object: BLOCK4

SALARY

```
begin

select employee_id,First_name
into :emp_id, :first_name
from employees
where salary =:salary;
--salary = 130000

exception
when no_data_found then
message('Not found this salary');message(' ');

when too_many_rows then

select count(*), sum(salary)
into :count1,:sum1
from employees where salary=:salary;
--salary=4800

-- count(*)or count(salary)or count(employee_id) ..... anythi

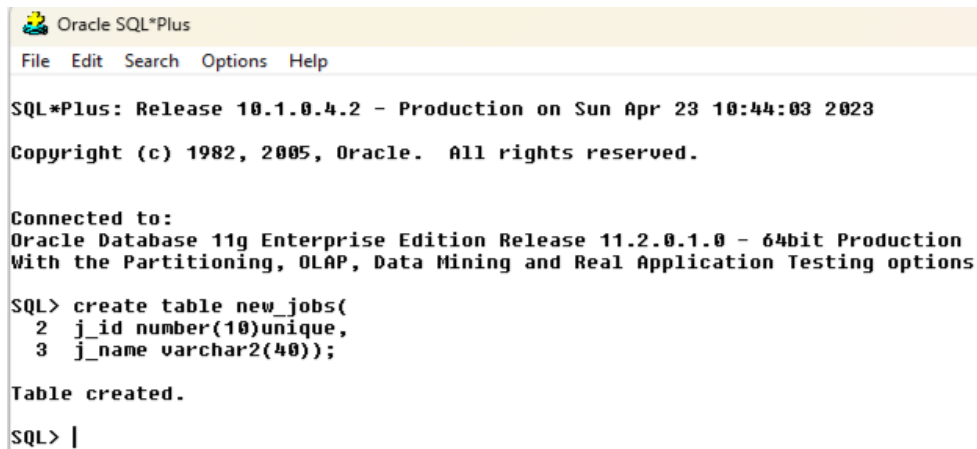
end;
```

Not Modified

Not C

Insert into:

➤ Create a table.



Oracle SQL*Plus

File Edit Search Options Help

SQL*Plus: Release 10.1.0.4.2 - Production on Sun Apr 23 10:44:03 2023

Copyright (c) 1982, 2005, Oracle. All rights reserved.

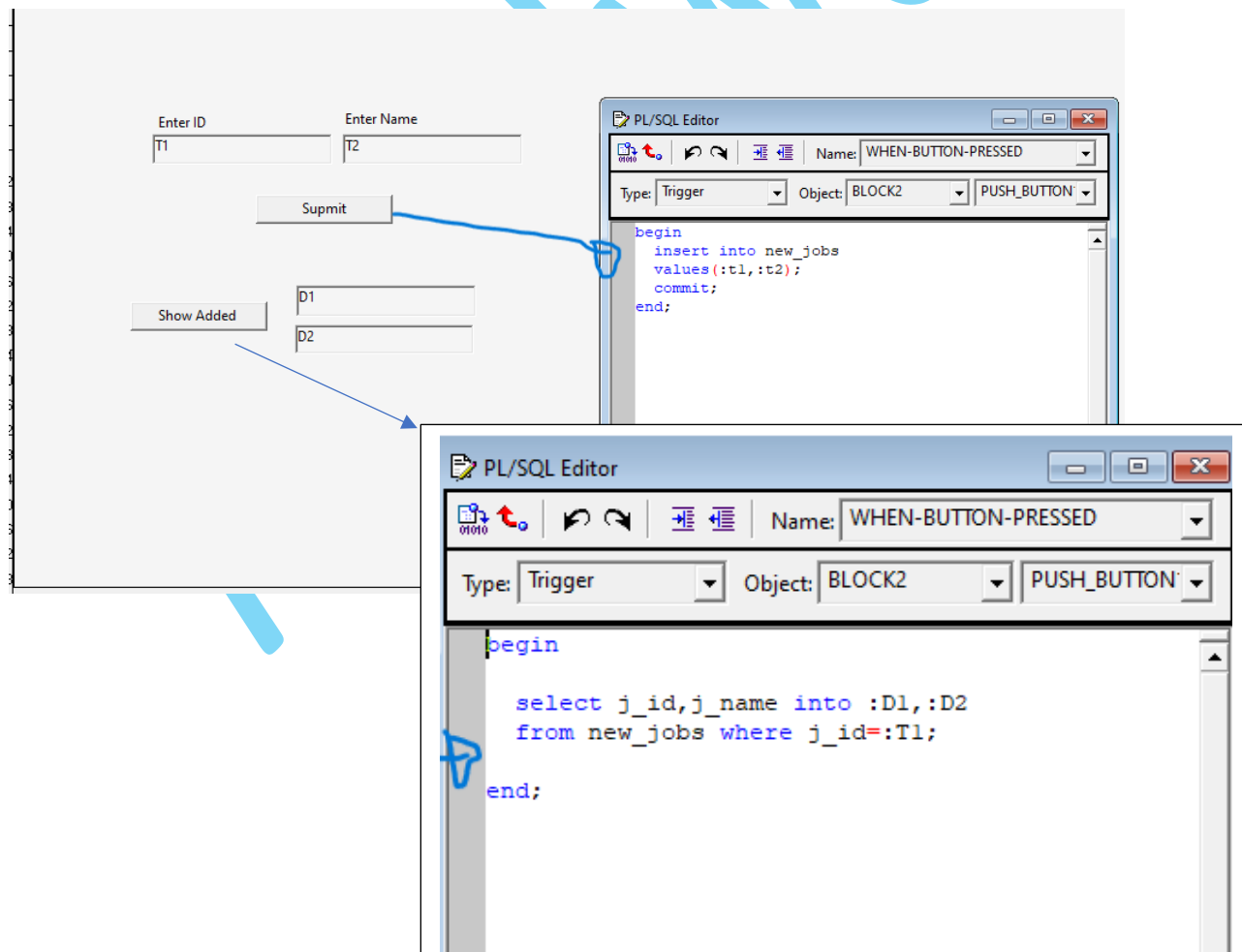
Connected to:
Oracle Database 11g Enterprise Edition Release 11.2.0.1.0 - 64bit Production
With the Partitioning, OLAP, Data Mining and Real Application Testing options

```
SQL> create table new_jobs(
  2   j_id number(10)unique,
  3   j_name varchar2(40));
```

Table created.

SQL> |

➤ Insert



The image shows a web application interface on the left and two PL/SQL Editor windows on the right. The web application has two input fields: "Enter ID" with value "T1" and "Enter Name" with value "T2". Below these is a "Submit" button. Further down are two more input fields: "D1" and "D2", with a "Show Added" button below them. A blue arrow points from the "Submit" button to the first PL/SQL Editor window. Another blue arrow points from the "Show Added" button to the second PL/SQL Editor window.

PL/SQL Editor (Top):

- Name: WHEN-BUTTON-PRESSED
- Type: Trigger
- Object: BLOCK2
- PUSH_BUTTON

```
begin
  insert into new_jobs
  values (:t1,:t2);
  commit;
end;
```

PL/SQL Editor (Bottom):

- Name: WHEN-BUTTON-PRESSED
- Type: Trigger
- Object: BLOCK2
- PUSH_BUTTON

```
begin

  select j_id,j_name into :D1,:D2
  from new_jobs where j_id=:T1;

end;
```

- Dup_val_on_index

- First Solution

The screenshot displays a web form on the left and a PL/SQL Editor window on the right. The form has two input fields: 'Enter ID' with value 'T1' and 'Enter Name' with value 'T2'. Below them is a 'Submit' button. Further down, there is a 'Show Added' button and two output fields: 'D1' and 'D2'. A blue arrow points from the 'Submit' button to the PL/SQL Editor. The editor window is titled 'PL/SQL Editor' and shows a trigger named 'WHEN-BUTTON-PRESSED' of type 'Trigger' attached to 'BLOCK2' at 'PUSH_BUTTON1:'. The trigger code is as follows:

```
begin
  insert into new_jobs
  values (:t1, :t2);
  commit;

exception
  when dup_val_on_index then
    message('The number has already been added');
    message(' ');
    go_item('T1');
end;
```

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

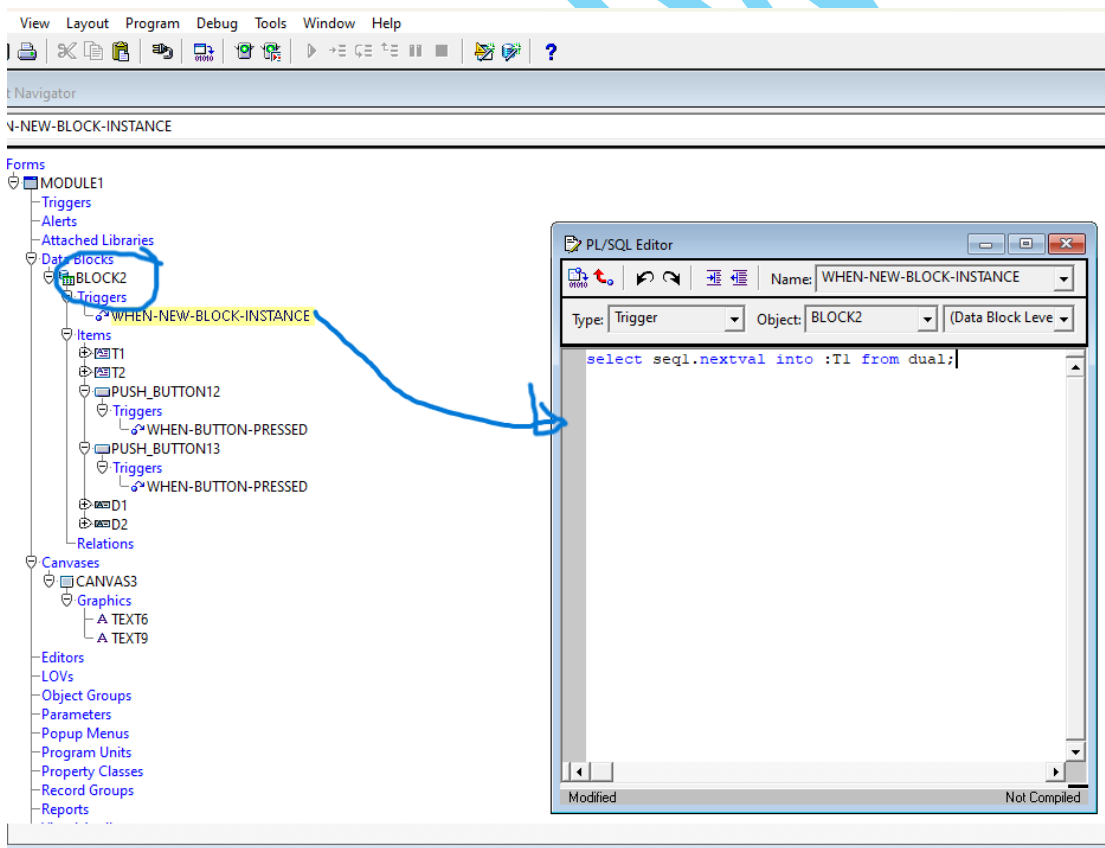
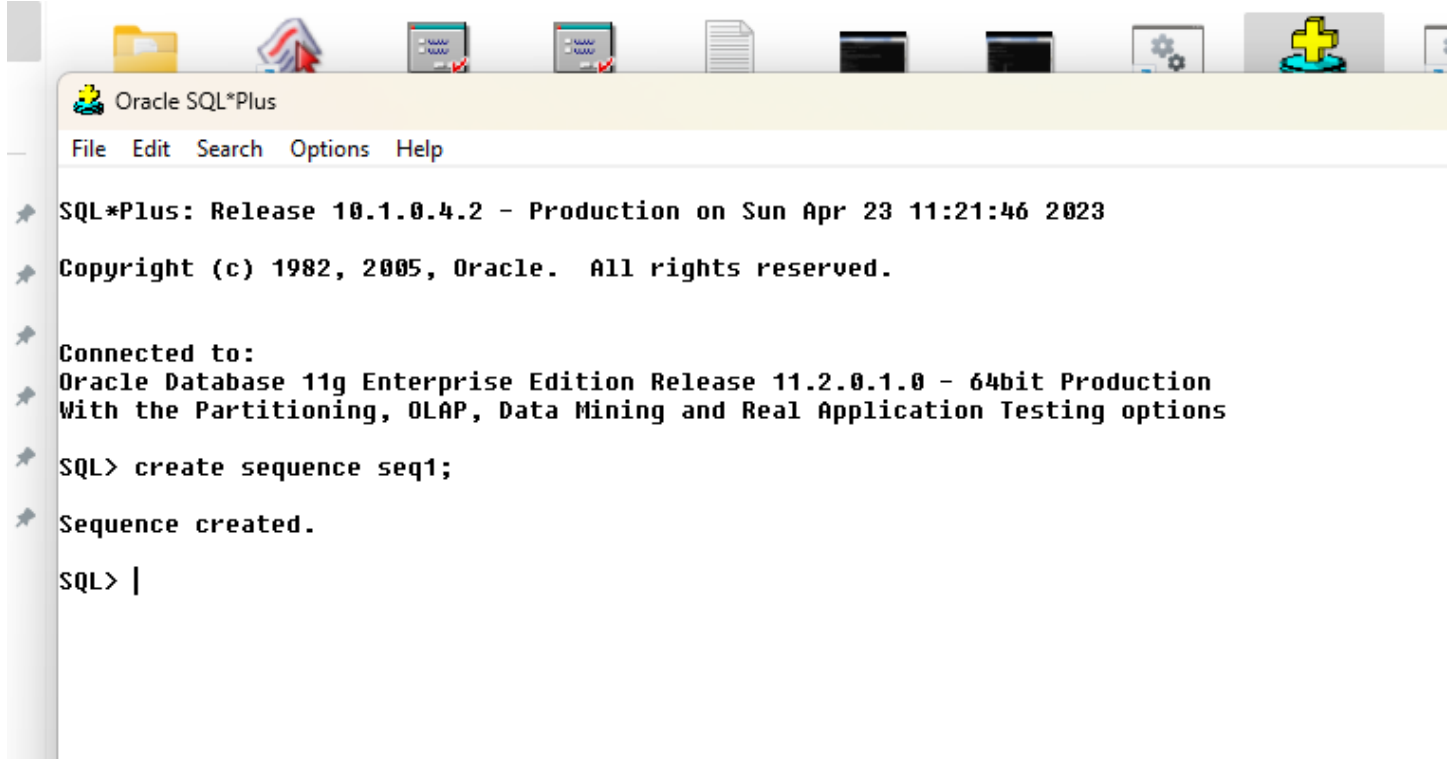
- Second Solution

The screenshot displays the same web form as the first solution. The PL/SQL Editor window on the right shows the same trigger code, but with a modification: the line `:T1:=:T1+1;` is added before the `insert into new_jobs` statement. This line is highlighted in blue. The status bar at the bottom of the editor indicates 'Modified' and 'Not Compiled'.

```
begin
  insert into new_jobs
  values (:t1, :t2);
  commit;

exception
  when dup_val_on_index then
    :T1:=:T1+1;
    insert into new_jobs
    values (:T1, :T2); commit;
end;
```

- Third solution (using sequence)



Enter ID

T1

Enter Name

T2

Submit

Show Added

D1

D2

Property Palette

Item: PUSH_BUTTON12

Item Type	Push Button
Subclass Information	
Comments	
Help Book Topic	
Functional	
Enabled	Yes
Label	Submit
Access Key	
Implementation Class	
Iconic	No
Icon Filename	
Default Button	No
Popup Menu	<Null>

Is object enabled or mouse-manipulable?

PL/SQL Editor

Name: WHEN-BUTTON-PRESSED
Type: Trigger Object: BLOCK2 PUSH_BUTTON12

```
begin
  insert into new_jobs
  values (:t1, :t2);
  commit;

exception
  when dup_val_on_index then
    select seq1.nextval into :T1 from dual;

    insert into new_jobs
    values (:T1, :T2); commit;
end;
```

Not Modified

Successfully Compiled

Enter ID

3

Enter Name

Submit

Show Added

Abdalwahab Mostafa Qatawneh