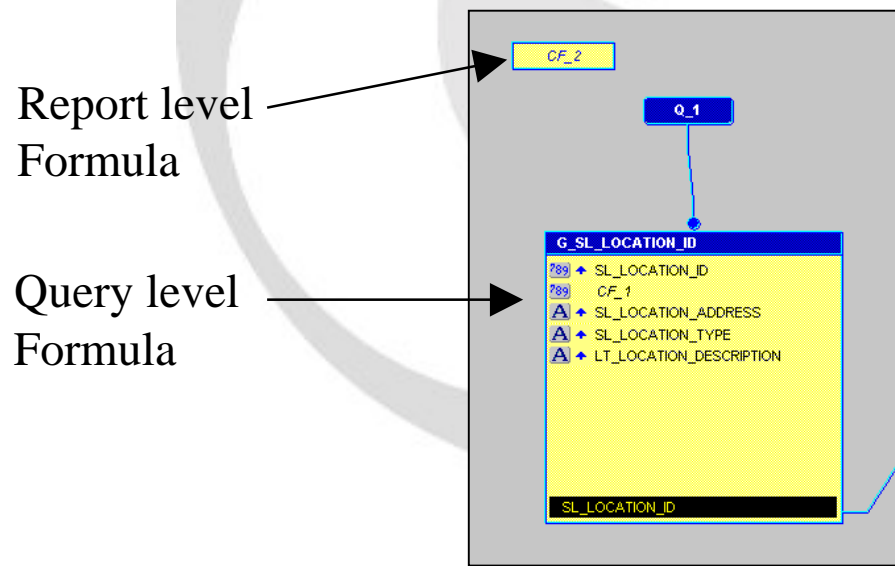




Formula Columns can be placed both within a column listing or independently within the GUI window.

When placed within the query the Formula will automatically be initialised when the record changes, when outside a query the Formula will continue to accrue values for the entire report.





The Formula is in effect a PL/SQL function which can be reached via the Formula's property, this in effect can perform a calculation and 'return' that value to the Formula variable ...

The screenshot shows a window titled "LOCATION: Program Unit - PU_006". The window has a menu bar with "Compile", "Revert", "New...", "Delete", "Close", and "Help". Below the menu bar, there are dropdown menus for "Type: Object Level", "Object: Column", and "CF_2". A text field labeled "Name:" contains the word "Formula". The main area of the window is a text editor containing the following PL/SQL code:

```
function CF_2Formula return Number is
begin
  return(:salary + :bonus);
end;
```

At the bottom of the window, there are two status indicators: "Modified" on the left and "Not Compiled" on the right.

The PL/SQL editor is examined in full detail later in this section.



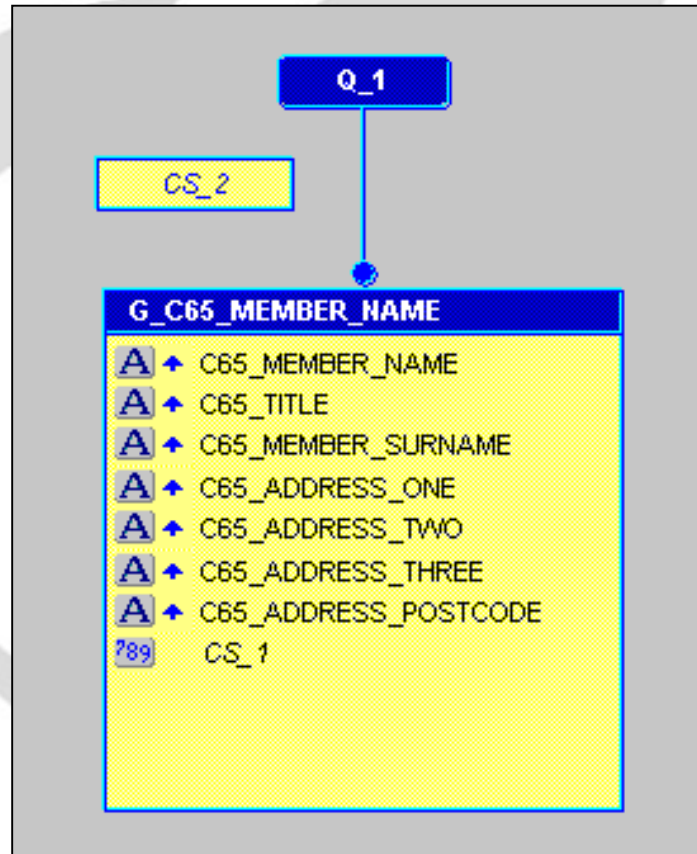
- For efficiency, the use of a formula column should be limited to instances where it is not possible to use a SELECT statement to perform the calculation.
- Formula columns should not be used to set a parameter's value. A discussion of parameters is given later in the course.



- By its very nature a Summary Column should not be placed at the same level as the column(s) it is actually ‘summing’ instead it should be positioned in the group above or at Report level.
- The reset level property will initialise the value whenever a record changes in that group.
- As in the case of Formulas, Summaries should not be created if either the SQL query or the built-in Report totalling can be used.



- Summary Columns can be placed both within a column listing or independently within the GUI window.





- Summary Columns can be created for all datatypes, however Summaries have greater scope when applied to numeric columns.
- The Summary types are
 - Sum
 - Average
 - Minimum
 - Maximum
 - Count
 - First
 - Last
 - % of Total
 - Std Deviation
 - Variance

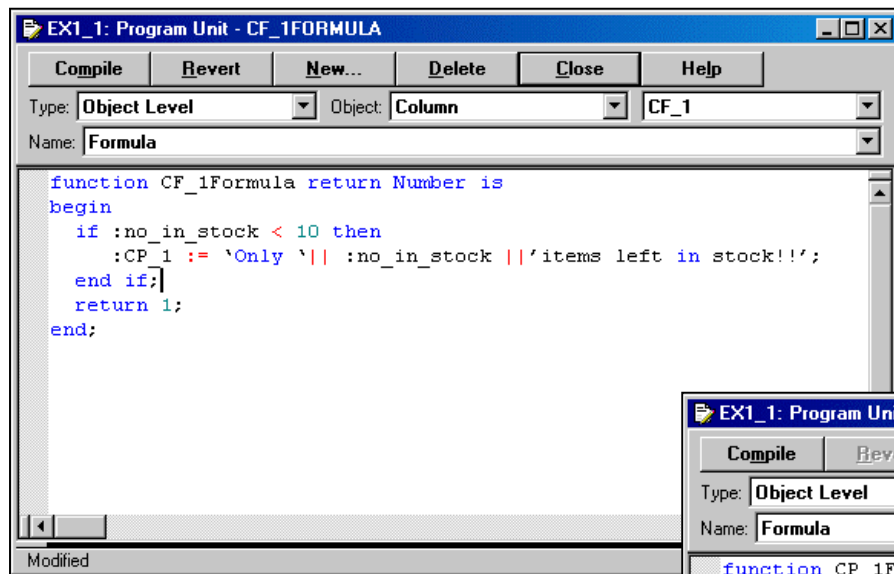


Placeholder columns are storage areas in the Data Model, they can be populated by a Before Report trigger but are generally populated by a Formula.

Although Placeholders allow PL/SQL to be written within them, they cannot be 'self-populated', their value must be set by an external Program Unit, any attempts at 'self-population' will result in a runtime compilation error. However the Formula assigned to the Placeholder must return a value.

To complicate matters, if a Placeholder is defined at Report level then it can only be populated by a Formula at Report Level or the above trigger. In many ways using Parameters to perform a Placeholders functionality is a better option.

Data Model Interface - Placeholder Columns



EX1_1: Program Unit - CF_1FORMULA

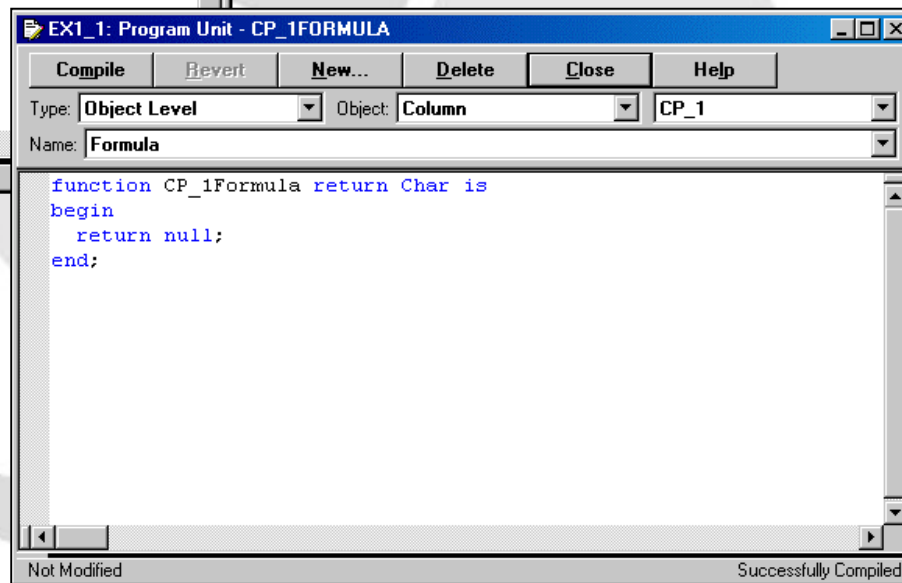
Compile Revert New... Delete Close Help

Type: Object Level Object: Column CF_1

Name: Formula

```
function CF_1Formula return Number is
begin
  if :no_in_stock < 10 then
    :CP_1 := 'Only ' || :no_in_stock || 'items left in stock!';
  end if;
  return 1;
end;
```

Modified



EX1_1: Program Unit - CP_1FORMULA

Compile Revert New... Delete Close Help

Type: Object Level Object: Column CP_1

Name: Formula

```
function CP_1Formula return Char is
begin
  return null;
end;
```

Not Modified

Successfully Compiled