



**In this section the Developer will learn...**

- To learn how to make decisions using 'if' statements

## Conditional Statements – If .. Then Statements

---



'If' statements are the decision makers with the PL/SQL coding, it is their function to decide what processing is to be done according to a value held in a variable or a table column.

An 'if' statement always begins with an 'if' and must always terminate with an 'end if;'

An 'if' is always followed by a condition and then an action, for example..

```
if <condition> then
```

```
    <process>;
```

```
end if;
```

## Conditional Statements – If .. Then Statements

---



The condition can be optionally enclosed by brackets but must always return a true or a false answer.

Conditions can use the same tests that 'where' statements use in an SQL statement, therefore the following conditions are correct ...

*if l\_dummy < 10 then*

*if l\_dummy between 1 and 10 then*

*if l\_dummy not between 1 and 20 then*

*if l\_dummy in (1,4,5,6,7,10,12) then*

*if l\_date between '12-May-2010' and '13-Jun-2010' then*

*if l\_text like '%T%' then*

## Conditional Statements – If .. Then Statements

---



In addition the Developer can use any Oracle function appropriate for the test ...

*if upper(l\_text) = 'ABC' then*

*if add\_months(l\_date, 12) = '12-Dec-2010' then*

*if substr(l\_name, 1, 3) = 'ANN' then*

*if l\_name||l\_surname = 'WILLIAMJENKINS' then*

## Conditional Statements – If .. Then Else Statements

---



In the majority of circumstances, the Developer will need to offer an alternative to the processing performed when the condition is not met, this is the else statement ...

```
if <condition> then
```

```
    <process>;
```

```
else
```

```
    <process2>;
```

```
end if;
```

Note the indentation and the use of lines when programming a condition, this will visually allow the Developer to spot a mistake at a glance ...

## Conditional Statements – If .. Then Else Statements

---



In some circumstances such as a Developer merely creating the 'shell' of the condition for compilation etc, no processing within the condition is necessary, here the Developer uses the 'null' process ...

*if* *<condition>* *then*

*<process>*

*else*

*null;*

*end if;*



The 'elsif' clause allows the Developer to perform a series of tests within the statement allowing different results to perform different processes, each elsif must be followed immediately by a condition ...

```
if l_name = 'John' then  
    <process1>;  
elsif l_name = 'Sue' then  
    <process2>;  
elsif l_name = 'Dave' then  
    <process3>;  
else  
    <process4>;  
end if;
```

Note that the else statement must always be the last one and cannot appear in the middle of the tests for example.

## Conditional Statements – Combining Tests

---



In the same way that a where clause can be made up of several tests against the data, so can a condition consist of a number of different conditions, they can be linked together (like wheres) using either the 'or' or 'and' clause. The syntax is as follows ...

```
if l_name = 'Dave' and l_dob = '24-Jan-1965' then
```

```
    <process>;
```

```
end if;
```

```
if l_town = 'London' or l_age > 70 then
```

```
    <process>;
```

```
end if;
```



## Conditional Statements – Nesting Conditions

---



In some circumstances, the Developer may wish to perform a series of tests and depending on the result perform several more, here the Developer is examining the month of the year and within that will test for a particular date ...

```
if l_month = 'jan' then
```

```
    if l_day = '1' then
```

```
        <process1>;
```

```
    elsif l_day = '2' then
```

```
        <process2>;
```

```
    end if;
```

```
end if;
```

Note the indentation for clarity and how both sets are finalised.

## Conditional Statements – Boolean Conditions

---



In all Conditional Statements the test is a boolean one, either true or false, the Developer however can use boolean variables directly ...

*declare*

*l\_dummy boolean := true;*

*begin*

*if l\_dummy then*

To test for false boolean ...

*if not l\_dummy then*