case

Last Modified on 25/05/2020 1:42 pm AEST

Returns one of several results based on some condition.

Keyword

```
case value
  when option1 then result1
  when option2 then result2
  ...
  else default
end
```

```
case
  when condition1 then result1
  when condition2 then result2
  ...
  else default
end
```

Arguments

Argument	Data Type
value	any
option	any single-value
condition	bool (true/false)
result	any
default	any

Comments

The **case** keywords provides a convenient way to test for multiple conditions. It can be used in one of two ways:

1. An initial *value* calculation is provided immediately after the case keyword, and that calculation result is then matched to one of several possible options.
2. Or, no initial calculation is provided, but several true/false conditions are provided.

If multiple options match the value, or if multiple conditions result in true, then the first match determines the result value.

If a value search calculation is provided, then it may be a single-value or list calculation, resulting in a single bool (true/false) or list of bools respectively.

It is common that the *options* and *results* are simply hard-coded string values, however any single-value calculation can be used for *option*, *result*, and *default*.

Default 'else' result

Optionally, an **else** keyword can be used to indicate the default result that should be returned if none of the other options were matched. If no no **else** keyword is provided, and none of the options are matched, then the result will be **null**.

Case-insensitive

String comparisons are case insentive for the **case** keyword, and for the ReadiNow platform in general.

Output data type

If the output data types do not match then a common data type that matches all the possible results will be used; for example, this will be a string if one of the results are a string data type.

It is recommended that calculations are written such that all outputs are of the same type.

Nulls

Consistent with the way that nulls are treated in calculations (where a null value represents the idea that the value is unknown or unknowable):

1. If an initial value is provided, and it is null, then none of the options will match, even if they also evaluate to null.
2. If a list of conditions is used, and either side of the condition is null, then the condition will be false.

In either case, the default result will be returned.

Examples

```
case
  when [Input].[Country] = 'Australia' then 'citizen'
  when [Input].[City] = 'Sydney' then 'local'
  when [Input] is null then 'unspecified'
  else 'foreign'
end
```

```
case [Input].[Country]
  when 'Australia' then 'citizen'
  when 'New Zealand' then 'resident'
  else 'foreign'
end
```

```
case [Var]
  when 'Australia' then 'citizen'
  when 'Singapore' then 457
end
```

```
case [All Tasks].[Priority]
   when 'High' then 5
   when 'Low' then 1
end
```

```
/* tax bracket calculator - illustrative only */
case
    when [Gross Income] <= 18200 then 0
    when [Gross Income] <= 37000 then 0.19 * ([Gross Income] - 18200)
    when [Gross Income] <= 90000 then 3572 + 0.325 * ([Gross Income] - 37000)
    when [Gross Income] <= 180000 then 20797 + 0.37 * ([Gross Income] - 90000)
    when [Gross Income] >= 180000 then 54097 + 0.45 * ([Gross Income] - 180000)
end
```