round, rescale - Rounding Functions

Syntax

```
... round|rescale( val = arg ...  ) ...
```

Effect

The rounding functions expect a decimal floating point number as a main argument **val** and additional arguments that describe how this floating point number is handled. The type of the return value of a rounding function is always **decfloat34**. Within an arithmetic expression, the argument for the decimal floating point number can either be an arithmetic expression or a function. The other arguments must always be specified as numeric data objects.

- Rounding Function
- Rescaling Function

Note

The class CL_ABAP_MATH includes the method NORMALIZE for normalizing a decimal floating point object. The mantissa does not have any trailing zeroes in a normalized floating point number.

Rounding Function

The rounding function **round** can be implemented in operand positions using the following syntax:

Syntax

```
... round( val = arg {dec = n}|{prec = n} [mode = m] ) ...
```

Effect

This function rounds a decimal floating point number declared as an argument for the parameter **val**. A data object specified for **arg** is converted to the data type **decfloat34** before the function is executed, if necessary.

- If the parameter **dec** is given a value, the value entered is rounded to the number of decimal places specified in **n** and returned. **n** expects data objects of the type **i**. The value of these data objects cannot be less than -6144. If a negative value is given, the relevant integer digit is rounded.
- If the parameter **prec** is given a value, the value entered is rounded to the precision specified in **n** and returned. **n** expects data objects of the type **i**. The value of these data objects must be greater than 0.

A rounding can reduce scaling and precision but cannot increase them. If **dec** is specified, the mantissa of the return code does not contain any zeroes after the place where the rounding applies. If **prec** is specified, the input value is returned without being changed, if the specified precision is greater than or equal to the input value.

The parameter **mode** (optional) can be used to set the rounding type. For **m** it is only possible to specify values that exist as ROUND_… constants in class CL_ABAP_MATH. The following table shows the possible rounding rules. If **mode** is not given a value, commercial rounding is used.

Constant	Rounding Rule
ROUND_HALF_UP	The value is rounded up to the next round figure. If the value falls precisely halfway between two rounded values, it is rounded up away from zero (commercial rounding).
ROUND_HALF_DOWN	The value is rounded down to the next round figure. If the value falls precisely halfway between two round values, it is rounded down towards zero.
ROUND_HALF_EVEN	The value is rounded to the next round figure. If the value falls precisely halfway between two rounded values, it is rounded to the value which has an even number in the last place.
ROUND_UP	The value is always rounded away from zero/to the larger absolute value.
ROUND_DOWN	The value is always rounded to zero/to the smaller absolute value.
ROUND_CEILING	The value is always rounded in a positive direction/to the larger value.
ROUND_FLOOR	The value is always rounded in a positive direction/to the larger value.

Example

For a demonstration of the rounding types, see Rounding Function **round**.

Rescaling Function

The rescaling function **rescale** can be implemented in operand positions using the following syntax:

Syntax

```
... rescale( val = arg {dec = n}|{prec = n} [mode = m] ) ...
```

Effect

This function changes the scaling of a decimal floating point number declared as an argument for the parameter **val**. A data object specified for **arg** is converted to the data type **decfloat34** before the function is executed, if necessary.

- If the parameter **dec** is given a value, the value entered is rounded using the scaling specified in **n** and returned. **n** expects data objects of the type **i**. The value of these data objects cannot be less than -6144. If the scaling produces more than 34 places in the mantissa of the return value, a handleable exception is raised.
- If the parameter **prec** is given a value, the value entered is returned with the precision specified in **n** and appropriate scaling and returned. **n** expects data objects of the type **i**. The value of these data objects must be greater than 0 and less than 34.

A rescaling can both reduce and increase scaling and precision. An increase adds zeroes on the right.

The input value is rounded if required. The optional parameter **mod** can be used to specify the rounding rule, as described under the function **round**. The default is commercial rounding.

Examples

The following table shows the results of commercial rounding of the decimal floating point number 1234.56789 (scaling 5, precision 9) with various values for **dec**.

dec	String Representation	Internal Representation	Scaling	Precision
-5	0E+5	0...0000000000E+5	-5	1
-4	0E+4	0...0000000000E+4	-4	1
-3	1E+3	0...0000000001E+3	-3	1
-2	1.2E+3	0...0000000012E+2	-2	2
-1	1.23E+3	0...0000000123E+1	-1	3
0	1235	0...0000001235E+0	0	4
1	1234.6	0...0000012346E-1	1	5
2	1234.57	0...0000123457E-2	2	6
3	1234.568	0...0001234568E-3	3	7
4	1234.5679	0...0012345679E-4	4	8
5	1234.56789	0...0123456789E-5	5	9
6	1234.56789	0...0123456789E-5	5	9

The following table shows the results of commercial rounding of the decimal floating point number 1234.56789 (scaling 5, precision 9) with various values for **prec**.

prec	String Representation	Internal Representation	Scaling	Precision
1	1E+3	0...0000000001E+3	-3	1
2	1.2E+3	0...0000000012E+2	-2	2
3	1.23E+3	0...0000000123E+1	-1	3
4	1235	0...0000001235E+0	0	4
5	1234.6	0...0000012346E-1	1	5
6	1234.57	0...0000123457E-2	2	6

7	1234.568	0...0001234568E-3	3	7
8	1234.5679	0...0012345679E-4	4	8
9	1234.56789	0...0123456789E-5	5	9
10	1234.56789	0...0123456789E-5	5	9

The following table shows the results of rescaling of the decimal floating point number 1234.56789 (scaling 5, precision 9) with various values for **dec**, if commercial rounding is used.

dec	String Representation	Internal Representation	Scaling	Precision
-5	0E+5	0...0000000000000E+5	-5	1
-4	0E+4	0...0000000000000E+4	-4	1
-3	1E+3	0...0000000000001E+3	-3	1
-2	1.2E+3	0...0000000000012E+2	-2	2
-1	1.23E+3	0...0000000000123E+1	-1	3
0	1235	0...0000000001235E+0	0	4
1	1234.6	0...0000000012346E-1	1	5
2	1234.57	0...0000000123457E-2	2	6
3	1234.568	0...0000001234568E-3	3	7
4	1234.5679	0...0000012345679E-4	4	8
5	1234.56789	0...0000123456789E-5	5	9
6	1234.567890	0...0001234567890E-6	6	10
7	1234.5678900	0...0012345678900E-7	7	11
8	1234.56789000	0...0123456789000E-8	8	12

The following table shows the results of rescaling of the decimal floating point number 1234.56789 (scaling 5, precision 9) with various values for **prec**, if commercial rounding is used.

prec	String Representation	Internal Representation	Scaling	Precision
1	1E+3	0...0000000000001E+3	-3	1
2	1.2E+3	0...0000000000012E+2	-2	2
3	1.23E+3	0...0000000000123E+1	-1	3
4	1235	0...00000000001235+0	0	4
5	1234.6	0...0000000012346E-1	1	5
6	1234.57	0...0000000123457E-2	2	6
7	1234.568	0...0000001234568E-3	3	7
8	1234.5679	0...0000012345679E-4	4	8
9	1234.56789	0...0000123456789E-5	5	9
10	1234.567890	0...0001234567890E-6	6	10
11	1234.5678900	0...0012345678900E-7	7	11
12	1234.56789000	0...0123456789000E-8	8	12