



# EIOPA Validations Syntax

---

**Ver 2.7.0**

**The document contains non-binding information, and is subject to substantial further changes**

**LAST UPDATE: 15/07/2022**

# INDEX

- I Modification history ..... 3**
- II Introduction..... 4**
- III Syntax use cases..... 4**
  - III.1 Generic mathematical and logical operators ..... 4*
  - III.2 Syntax specific for EIOPA validations ..... 4*
    - III.2.1 Data type constrains ..... 4
    - III.2.2 'Empty' ..... 5
    - III.2.3 'sNNN' ..... 5
    - III.2.4 Dictionary element reference..... 6
    - III.2.5 'Like'/'not like' ..... 6
    - III.2.6 Conditional validations..... 7
    - III.2.7 'Has to be reported' ..... 7
    - III.2.8 'rNNN' & 'cNNN' ..... 7
    - III.2.9 'Reported' ..... 8
    - III.2.10'Allowed combinations of values' ..... 8
    - III.2.11'Unit of a monetary concept for (...) does not match value of (...) ' ..... 8

## I Modification history

Date	Main change description
30/09/2015	First version of the document
15/07/2016	'rNNN' & 'cNNN' syntax added to the document. Replacing syntax 'for every' with 'not(isfallback)'
1/06/2017	'Reported' added to syntax
1/06/2017	'Allowed combinations of values'
1/06/2017	'Unit of a monetary concept for (...) does not match value of (...)'
1/11/2018	BV4 example for 'Like'/'not like' was replaced with BV6 validation (due to the fact that BV4 validation was removed from the 2.3.0 Hotfix scope)
3/06/2019	Updates to examples due to improvements in business and technical validations
15/07/2020	Updates to examples due to improvements in business and technical validations
15/07/2021	Updates to examples to make it more general

## II Introduction

Aim of this document is to describe syntax, wording and patterns used in definition of business and technical rules for EIOPA XBRL taxonomies. The examples provided are based on Solvency II validations.

## III Syntax use cases

### III.1 Generic mathematical and logical operators

Below table describes basic operators used in business rules

Operator	Meaning
+	Addition
-	Subtraction
*	Multiplication
/	Division
=	Equation
>=	Greater than or equal
<=	Less than or equal
<>	Not equal (other than)
Sum()	Calculates summation of components inside the parenthesis
Max()	Finds maximum value from the components inside the parenthesis
Min()	Finds minimum value from the components inside the parenthesis
Abs()	Returns absolute value from the components inside the parenthesis
And	Both components must be true
Or	At least one component must be true
Exp()	Calculates the exponential function. It requires the expression, numerator and denominator of an exponent inside parentheses separated by a comma

### III.2 Syntax specific for EIOPA validations

Some syntax used in validations is specific for EIOPA project. Below particular case with explanation and examples are provided.

#### III.2.1 Data type constrains

Data type constrain is used to identify applicable patterns for a given reportable fact. In majority of the cases it refers to one of the ISO codification standards, like ISO 4217 for currencies or ISO 8601 for dates.

Examples:

Validation	Explanation
{c0390} - data type constrains [ISO 8601: (yyyy-mm-dd)]	Value in column C0390 must be in line with the ISO 8601 format (yyyy-mm-dd)
{c0080} - data type constrains [ISO 4217: alphabetic code]	Value in column C0080 must be in line with the ISO 4217 format
{c0040} - data type constrains [ISO 3166-1: alpha-2 code; or 'XA'; or 'EU'; or 'AA']	Value in column C0040 must be in line with the ISO 3166-1 alpha-2 code format but additionally possible values are also "XA", "EU" and "AA" which do not belong to the standard
{Z0030} - data type constrains [ISO 4217: alphabetic code; or 'Total/NA']	Value on z-axis Z0030 must be in line with ISO 4217 format and additionally it can be reported as a "Total/NA"

NOTE:

This type of validation is represented in a taxonomy as a reference to list of domain members defined in the dictionary or by data type as an XML attribute, hence technically no XBRL formulas are generated for data type constrains.

### III.2.2 'Empty'

This operator is used to check whether particular reportable element was or was not reported.

Examples:

Validation	Explanation
{er0010, ec0020} <> empty	Cell er0010,ec0020 must not be empty ( <i>information about template code is provided elsewhere</i> )
If {c0290} like '##75' or {c0290} like '##95' then {c0280} = empty	If for a given row in open table (e.g. S.06.02), third and fourth digits from code reported in column c0290 is 75 or 95, then column c0280 must be empty ( <i>information about template code is provided elsewhere</i> )
If {S.15.02, c0060}<>empty then {S.15.01, c0090}<>empty	If there is a row reported in table S.15.02, the row with the same key must be reported in table S.15.01. Column codes are provided as examples of datapoints that must be provided in each table to make sure the validation is executed properly.

### III.2.3 'sNNN'

This expression is used to indicate situations where component of the equation is a summation of cells for a given column, row and/or z-axis. sNNN is always preceded by the Sum() operator.

Examples:

Validation	Explanation
{S.02.01, r0770,c0010}>= sum({S.31.01, c0140,(sNNN)})	Value in cell r0770,c0010 in table S.02.01 must be greater or equal to summation of values from all rows for column c0140 in table S.31.01. <i>Note: If there will be drop down list on z-axis in table S.31.01, it will be multiplied by these options as well</i>
sum({S.29.04, r0110,c0040,(sNNN)}) = {S.29.03, r0200,c0050} + {S.29.03, r0200,c0060}	Summation of values from cell r0100,c0040 for all z-axis combinations in table S.29.04 must be equal to summation of cells r0200,c0050 and r0200,c0060 from table S.29.03

### III.2.4 Dictionary element reference

Since some of the reported facts are components of the dictionary (e.g. *s2c\_SE:x10* which is an domain member from the SE domain and its label is *Undertakings pursuing both life and non-life insurance activity*), they are also used in a number of business rules. In the expressions, these cases are identified by putting relevant dictionary component within the square brackets.

Examples:

Validation	Explanation
If {S.01.02, r0190,c0010} = [s2c_AP:x9] then {S.01.01, r0370,c0010}=[s2c_CN:x1]	If value reported in cell r0190,c0010 in table S.01.02 is <i>s2c_AP:x9</i> ( <i>Use of transitional measure on the risk-free interest rate</i> ), then value of cell r0370,c0010 in table S.01.01 must be <i>s2c_CN:x1</i> ( <i>Reported</i> )
If {z0020}=[s2c_PU:x60] then {z0030}<>empty	If a value reported on z-axis z0020 is <i>s2c_PU:x60</i> (The items reported refer to a RFF/MP), then z-axis z0030 must not be empty

### III.2.5 'Like'/'not like'

This operator provides mechanism to distinguish pattern or a given sign from the reported element. It is used primarily to filter out particular rows from open tables. In case of technical validations (TV) patterns include '^' identifying beginning of a text and '\$' identifying the end of a text.

Examples:

Validation	Explanation
If {c0290} like '##71' or {c0290} like '##75' or {c0290} like '##9#' then {c0270} = empty	If for a given row in open table S.06.02, third digit from code reported in column c0290 is 9, or third and fourth digits are 71 or 75 then column c0270 must be empty ( <i>information about template code is provided elsewhere</i> )
{S.02.01, r0150,c0010}=sum({S.06.02, c0170,(sNNN)}); Filter - {S.06.02, c0290} like '##2#' and {S.06.02, c0090}=[s2c_LB:x91]	Value in cell r0150,c0010 in table S.02.01 must be equal to summation of values from column c0170 in table S.06.02 for a specific rows defined by the filter. Filter takes out only rows, where third digit in column c0290 is 2 and where value reported in column c0090 is "s2c_LB:x91" ( <i>Neither unit-linked nor index-linked</i> )

si1495 like "^LEI/[A-Z0-9]{20}\$" or "^None"	Value reported for metric si1495 must be in line with LEI which is 20 character alphanumeric code, preceded by "LEI/". The only other accepted value is "None"
dim:GO like "^LEI/[A-Z0-9]{20}\$" or "^None"	Value reported for dimension GO must be in line with LEI which is 20 character alphanumeric code, preceded by "LEI/". The only other accepted value is "None"

### III.2.6 Conditional validations

These validations are represented by *If x then y* notation. Often, logical test (x) and the result if true (y) are complex expressions and are using other operators described in this document.

Examples:

Validation	Explanation
If {S.26.01, r0020,c0010} = [s2c_AP:x34] and {S.02.01, r0850,c0010} = empty then {S.26.01, r0110,c0060} = MAX(0,({S.26.01, r0110,c0020}-{S.26.01, r0110,c0030})-({S.26.01, r0110,c0040}-{S.26.01, r0110,c0050}))	If value reported in table S.26.01 cell r0020,c0010 is s2c_AP:x34 ( <i>Simplifications not used</i> ) and cell reported in S.02.01 r0850,c0010 is empty then, for table S.26.01, value in cell r0110,c0060 must be equal to the maximum value of either 0 or result of subtracting r0110,c0030; r0110,c0040 and r0110,c0050 from r0110,c0020

### III.2.7 'Has to be reported'

This expression checks if any cell from the specified "has to be reported" table exists in the report.

Examples:

Validation	Explanation
If {S.01.02, r0150,c0010}=[s2c_PU:x4] or {S.01.02, r0170,c0010}=[s2c_PU:x51] then {SR.01.01} has to be reported	If value in cell r0150,c0010 is s2c_PU:x4 or value in cell r0170,c0010 is s2c_PU:x51 in table S.01.02, then all cells in table SR.01.01 must be reported.

### III.2.8 'rNNN' & 'cNNN'

This expression is used to indicate situation that cross template validation should be executed for a range of rows ('rNNN') or columns ('cNNN') from one of tables that it refers.

Examples:

Validation	Explanation
If {SR.01.01, r0810,c0010}=Reported then {SR.17.01, rNNN,c0180}=sum({SR.17.01, rNNN,(c0020-0170)})	If value in cell r0810,c0010 in table SR.01.01 is s2c_CN:x1, then value reported in SR.17.01 table in column c0180 must be equal to sum of columns c0020-0170 for all the rows specified in the rows range ( <i>information about rows range is provided elsewhere</i> )

### III.2.9 'Reported'

This expression is used to indicate all potential options when particular SR template is reported.

Validation	Explanation
If {SR.01.01, r0870,c0010}=Reported then {SR.26.01, r0210,cNNN}={SR.26.01, r0221,cNNN}+{SR.26.01, r0230,cNNN}+{SR.26.01, r0231,cNNN}+{SR.26.01, r0240,cNNN}	If value in cell r0870,c0010 in table SR.01.01 is <i>s2c_CN:x1 (1 - Reported) or s2c_CN:x60 (16 - Reported due to request of Article 112 of Directive 2009/138/EC) or s2c_CN:x71 (17 - Reported twice due to use of PIM)</i> , then for given scenario in template SR.26.01 for particular column (from range specified as cNNN) value reported in row r0210 must be equal to sum of rows r0221-0240

### III.2.10 'Allowed combinations of values'

This expression is used to indicate possible combinations of integers.

Validation	Explanation
si2468 allows 1 or 9	The only values that can be reported for si2468 are "1" or "9"
si1371 allows combinations of values from 1 to 4	The only values that can be reported for si2468 are "1" or "2" or "3" or "4" or "1,2" or "1,3" or "1,4" or "2,3" or "2,4" or "3,4" or "1,2,3" or "1,2,4" or "1,3,4" or "2,3,4" or "1,2,3,4"

### III.2.11 'Unit of a monetary concept for (...) does not match value of (...)'

This expression is used to indicate possible mismatches between currency in the currency context and currency provided by data model.

Validation	Explanation
Unit of a monetary concept for AF:x0 does not match value of s2md_met:ei1930	Validation verifies if all monetary facts are reported in reporting currency matching the one selected with s2md_met:ei1930
Unit of monetary concept for AF:x1 does not match value of OC dimension	Validation verifies if all monetary facts identified as AF:x1 have the same currency identified with OC dimension and currency context