.Anatella FILE STRUCTURE

1. Introduction

This document provides a description about the structure of a ".anatella" file. An Anatella file is simply an XML file which is a very well known and well documented type of file for which the documentation can be found here : XML Doc.

In the present document you will find a description of all XML tags used by each Anatella box. Each tag or attribute corresponds to a parameter used by an Anatella box.

2. Global structure

The following structure is common for all anatella files

The root node is named 'ANATELLA' and has 6 child nodes :

Name	Description
GlobalParameters	User-defined Graph Global parameters (found in edit tab). This typically includes: • the ODBC links, • the OleDB links, • the "global parameters" names & default values.
WorkingSpace	The <WorkingSpace> node can be omitted: it just contains information that are related to the Anatella window position and to the way the different panels inside the Anatella interface are positioned. If omitted, the .anatella file will still open inside the visual editor with the default visual layout.
ACTIONS	Contains one child per anatella box placed inside the graph
CONNECTORS	Contains one child per connection between two boxes
TEXTANNOTATIONS	Contains one child per comment placed inside the graph
GROUPBOXANNOTATIONS	Contains one child per groupbox annotation placed inside the graph

```
<?xml version="1.0" encoding="utf-8"?>
<ANATELLA version='2.36'>
    <GlobalParameters>.</GlobalParameters>
    <WorkingSpace>...</WorkingSpace>
    <ACTIONS>.. </ACTIONS>
    <CONNECTORS>..</CONNECTORS>
    <TEXTANNOTATIONS>..</TEXTANNOTATIONS>
    <GROUPBOXANNOTATIONS>..</GROUPBOXANNOTATIONS>
</ANATELLA>
```

3. Node ACTIONS

The “x” and the “y” attributes can be omitted for every box, this will result in a graph where every box are placed on the same position but the graph is usable and can still be executed without worries.

The “module” and the “v” attributes can never be omitted and are required to have the same exact value as written inside this document.

All the XML Texts and the XML Attributes must be “XMLEncoded”: i.e. Do the following “*string replaces*” in this specific order:

Replace This:	…with that:	Notes
&	&amp;	ampersand
<	&lt;	less than
'	&apos;	apostrophe
"	&quot;	quotation mark

3.1. ReadCSV

Simple example:

```
<readCSV idx='1' x='0' y='0' sep=',' allowCRInField='1' fileName='D:/census-income.csv'/>
```

Complete example:

```
<readCSV keyHD='#1' idx='#2' x='#3' y='#4' sep='#5' firstLineContainsColumnNames='#6' textqualifier='#7' allowCRInField='#8' dropLastColumnIfEmptyName='#14' fillInEOL='#18' acceptIncompleteRows='#16' dropEmptyLine='#17' trim='#22' amazonDamaged='#23' damagedFile='#19' parsing='#21' encoding='#12' iconvDiscardUnconvertible='#13' bufferSize='#24' fixedNColumn='#15' outputFilenames='#10' abortIfMissing='#11' timeOut='#20' fileName='#9'/>
```

3.1.1. Standard Attributes

ID	Name	Value Type	Description	Default Value
1	keyHD	int	Number generated by anatella and used by anatella for caching purpose (used only for GUI)	
2	idx	int	ID of current box used for connections	
3	x	int	Number that defines the horizontal position of the box in the graph (used only for GUI)	0
4	y	int	Number that defines the vertical position of the box in the graph (used only for GUI)	0
5	sep	string	‘Delimiters’	,
6	firstLineContainsColumnNames	0,1		1
7	textqualifier	string		“
8	allowCRInField	0,1		0
9	fileName	string		

3.1.2. Advanced Attributes

ID	Name	Value Type	Description	Default Value
10	outputFilenames	0,1		0
11	abortIfMissing	0,1		1
12	encoding	string	name of the encoding chosen in the dropdown list	Auto
13	iconvDiscard Unconvertible	0,1	Allow (Discard) Invalid Characters	1
14	dropLastColumn IfEmptyName	0,1		1
15	fixedNColumn	int		1
16	acceptIncompleteRows	0,1	Accept lines of irregular sizes	1
17	dropEmptyLine	0,1		1
18	fillInEOL	[0,1,2]	Drop-down for 'Undefined columns in incomplete lines contain'	0
19	damagedFile	[0..3]		0
20	timeOut	int	Block 'Error Management'	0
21	parsing	0,1		0
22	trim	0,1		1
23	amazonDamaged	0,1		0
24	bufferSize	int		0

3.2. Aggregate

The Anatella aggregate box is the equivalent of the SQL comment "SELECT … FROM … GROUP BY …".

The node <GroupBy> in the XML file defines the "GroupBy" variables. Each selected column is written using a node <v> … </v> (see examples below).

The same is true for the node <OutputVars> but in addition to the attribute "name", there will be a series of attributes that will depend on which aggregates computation has been selected to be in output (sum, mean, countDistinct, min, etc).

Structural example:

```
<aggregate>
 <OneAggregate>
  <GroupBy><v/>…<v/></GroupBy>
  <OutputVars><v/>…<v/></OutputVars>
 </OneAggregate>
</aggregate>
```

Aggregate can also have multiple 'OneAggregate' like so (when using multiple 'OneAggregate', you need to have `inRamAlgo='1'`) :

```
<aggregate inRAMAlgo='1'>
 <OneAggregate>…</OneAggregate>
 …
 <OneAggregate>…</OneAggregate>
</aggregate>
```

Each different “OneAggregate” creates a different output table. This means for example that, if you have 3 “OneAggregate”, you’ll have 3 output pins (with 3 output tables).

<u>Simple example:</u>

GroupBy on variables ‘age’ and ’gender’ with output column:

mean of variable ‘wage per hour’, mean of variable ‘income’, sum of variable ‘income’

in SQL:

```
SELECT
      avg([wage per hour]) as [wage per hour_mean],
      avg([income]) as [income_mean],
      sum([income]) as [income_sum]
FROM [my_table]
GROUP BY [age],[gender]
```

in Anatella:

```
<aggregate idx='1' x='0' y='0' inRAMAlgo='1'>
 <OneAggregate>
  <GroupBy>
   <v name='age'/>
   <v name='gender'/>
  </GroupBy>
  <OutputVars>
   <v name='wage per hour' mean='1'/>
   <v name='income' mean='1' sum='1'/>
  </OutputVars>
 </OneAggregate>
</aggregate>
```

<u>Complete example:</u>

```
<aggregate keyHD='#1' idx='#2' x='#3' y='#4' inRAMAlgo='#5' omitSuffix='#23'
>
 <OneAggregate counts='#6'>
  <GroupBy>
   <v name='#7' type='#8'/>
   <v name='#7' type='#8'/>
  </GroupBy>
  <OutputVars>
   <v name='#7' countNonNull='#9' countNonEmpty='#10' imin='#11' imax='#12' sm
in='#13' smax='#14' first='#15' last='#16' sum='#17' mean='#18' stdDev='#19' c
ountDistinct='#20' mostCommonModality='#21' countMostCommonModality='#22'/>
   <v name='#7' countNonNull='#9' countNonEmpty='#10' imin='#11' imax='#12' sm
in='#13' smax='#14' first='#15' last='#16' sum='#17' mean='#18' stdDev='#19' c
ountDistinct='#20' mostCommonModality='#21' countMostCommonModality='#22'/>
  </OutputVars>
 </OneAggregate>
 <OneAggregate>
  <GroupBy>
   <v name='#7' type='#8'/>
  </GroupBy>
  <OutputVars>
  </OutputVars>
 </OneAggregate>
</aggregate>
```

3.2.1. Standard Attributes

ID	Name	Value Type	Description	Default Value
1	keyHD	int	Number generated by anatella and used by anatella for caching purpose (used only for GUI)	
2	idx	int	ID of current box used for connections	
3	x	int	Number that defines the horizontal position of the box in the graph (used only for GUI)	0
4	y	int	Number that defines the vertical position of the box in the graph (used only for GUI)	0
5	inRAMAlgo	0,1	Checkbox Use in-RAM algorithm, if you want, either: * more than one <OneAggregate> * to avoid sorting the input table … then inRAMAlgo=1	1
6	counts	0,1	Checkbox Output column: counts	0
7	name	string	Attribute of node <v>, name of the column	0
8	type	int	Attribute of node <v>, type of the column (optional: you can disregard)	0
9	countNonNull	0,1	checkbox for output column: self-explanatory	0
10	countNonEmpty	0,1	checkbox for output column: self-explanatory	0
11	imin	0,1	checkbox for output column: minimum of column when the column is a number	0
12	imax	0,1	checkbox for output column: maximum of column when the column is a number	0
13	smin	0,1	checkbox for output column: minimum of column when the column is a string	0
14	smax	0,1	checkbox for output column: maximum of column when the column is a string	0
15	first	0,1	checkbox for output column: first value inside the column (for each different "GroupBy")	0
16	last	0,1	checkbox for output column: last value inside the column (for each different "GroupBy")	0
17	sum	0,1	checkbox for output column: self-explanatory	0
18	mean	0,1	checkbox for output column: self-explanatory	0
19	stdDev	0,1	checkbox for output column: : self-explanatory (standard deviation)	0
20	countDistinct	0,1	checkbox for output column: self-explanatory	0
21	mostCommonModality	0,1	checkbox for output column: the most common modality (for each different "GroupBy")	0
22	countMostCommonModality	0,1	checkbox for output column: the count of the most common modality (for each different "GroupBy")	0
23	omitSuffix	0,1	Advanced Parameters	0

3.3. FilterRows

The FilterRows box lets you select columns to be filtered. Those columns are placed in the tag <InputVars> which contains child nodes with tag <InputVar> defining the selected column.

In addition to that, <FilterRows> has the child node <Expression>. The text encapsulated by this node contains the filter expression of the box.

Simple example: Filter on column age with filter expression : age<60

```
<FilterRows module='MathParser' idx='1' x='480' y='-12'>
 <Expression>age>60</Expression>
 <InputVars>
    <InputVar column='age'/>
 </InputVars>
</FilterRows>
```

Complete example:

```
<FilterRows keyHD='#1' module='#5' idx='#2' x='#3' y='#4' truncate='#6'
 addN='#22' abortWhenError='#24' useJIT='#23' addNow='#15' addNowS='#11' seed=
'#21' elapsedTimeUnit='#16' dateFormatType='#12' dateFormat='#13'
 isUTC='#14' emptyETToNull='#18' referenceTime='#17' castToET1='#19' castToET2
='#20'>
 <Expression>#25</Expression>
 <InputVars>
    <InputVar column='#7'/>
    <InputVar column='#7' label='#8' isNumber='#9' value='#10'
/>
 </InputVars>
</FilterRows>
```

3.3.1. Standard Attributes

ID	Name	Value Type	Description	Default Value
1	keyHD	int	Number generated by anatella and used by anatella for caching purpose (used only for GUI)	
2	idx	int	ID of current box used for connections	
3	x	int	Number that defines the horizontal position of the box in the graph (used only for GUI)	0
4	y	int	Number that defines the vertical position of the box in the graph (used only for GUI)	0
5	module	0,1	constante value : must be: 'MathParser'	none
6	truncate	0,1	Checkbox in 'Expression' panel : Truncate table on first failed row	0
7	column	string	In selected columns : column name	
8	label	string	In selected columns : 'Variable Name' column	
9	isNumber	0,1	In selected columns : checkbox 'ToNumber'	1
10	value	string	In selected columns : 'Current Value' column	
25	Expression.text	string	Filter expression	

3.3.1. Advanced Attributes

ID	Name	Value Type	Description	Default Value
11	addNowS	0,1		0
12	dateFormatType	[0..6]		0
13	dateFormat	string		
14	isUTC	0,1		1
15	addNow	int		0
16	elapsedTimeUnit	[0..4]		2
17	referenceTime	0,1		0
18	emptyETToNull	0,1	CheckBox : ‘Convert Empty String to Null ET’	1
19	castToET1	1,2		1
20	castToET2	1,2		1
21	seed	0,1		0
22	addN	0,1		1
23	useJIT	string		1
24	abortWhenError	0,1		1

3.4. Sort

The Sort box lets you select columns to be sorted. Those columns are placed in a <field> node directly as children of the node <sort>.

Simple example: Sort on column ‘age’

```
<sort idx='1' x='432' y='24' action='S'>
  <field type='0'>age</field>
</sort>
```

Complete example:

```
<sort keyHD='#1'
 idx='#2' x='#3' y='#4' action='#5' tapeSize='#9' maxNTape='#11' useSpecificDi
r='#15' crc='#19' compressTapes='#14' tapeDir='#16' nThreads='#12' ncpu='#10'
doDedup='#17' debugLevel='#13' checkNumeric='#18'>
  <field type='#6' dateFormat='#7'>#8</field>
  <field type='#6'>#8</field>
</sort>
```

3.4.1. Standard Attributes

ID	Name	Value Type	Description	Default Value
1	keyHD	int	Number generated by anatella and used by anatella for caching purpose (used only for GUI)	
2	idx	int	ID of current box used for connections	
3	x	int	Number that defines the horizontal position of the box in the graph (used only for GUI)	0

4	y	int	Number that defines the vertical position of the box in the graph (used only for GUI)	0
5	action	string	Sort Task drop down	0
6	type	string	Selected column sort type	
7	dateFormat	string	Selected column data format	
8	field.text	string	Selected column name	

3.4.2. Advanced Attributes

ID	Name	Value Type	Description	Default Value
9	tapeSize	int	Memory Buffer Size	100
10	ncpu	int	Number of Threads for sorting	1
11	maxNTape	int	Maximum number of Tape Files	1000
12	nThreads	int	Number of Threads for compression	-1
13	debugLevel	int		1
14	compressTapes	0,1		1
15	useSpecificDir	0,1	Radio Button 'Use the directory:'	0
16	tapeDir	string	Directory path	
17	doDedup	0,1	Checkbox in 'Special usage'	0
18	checkNumeric	0,1	Checkbox 'Abort when sorting in Numerical…'	1
19	crc	0,1	Checkbox 'Verify all Tape Files with a CRC check'	1

3.5. RegularExpressionTest

Simple example: Regex '^[0-9]{3,15}$' applied to column 'class of worker'

```
<RegularExpressionTest module='DefaultActions' idx='17' x='408' y='36' result=
'' regexp='^[0-9]{3,15}$' column='class of worker'/>
```

Complete example:

```
<RegularExpressionTest keyHD='#1'
 module='#5' idx='#2' x='#3' y='#4' nullIsFalse='#12' action='#6' result='#11'
 dynamicRE='#8' regexp='#9' recol='#10' column='#7'/>
```

3.5.1. Standard Attributes

ID	Name	Value Type	Description	Default Value
1	keyHD	int	Number generated by anatella and used by anatella for caching purpose (used only for GUI)	
2	idx	int	ID of current box used for connections	
3	x	int	Number that defines the horizontal position of the box in the graph (used only for GUI)	0
4	y	int	Number that defines the vertical position of the box in the graph (used only for GUI)	0
5	module	string	Constant value : must be: 'DefaultActions'	none
6	action	0,1	Drop-down filter or add bool column	0

7	column	string	Column to test	
8	dynamicRE	0,1	Drop-down ‘Regular Expression is’	0
9	regexp	string	Regular expression text when option 0 is chosen for parameter ‘dynamicRE’	^[0-9a-zA-Z]+@[0-9a-zA-Z\-]+[\.]{1}[0-9a-zA-Z\-]+[\.]?[0-9a-zA-Z]$
10	recol	string	Input column name selected when option 1 is chosen for parameter ‘dynamicRE’	
11	result		Result column name when option 1 is selected for parameter ‘action’	
12	nullIsFalse	0,1	Checkbox ‘NULL returns false’	1

3.6. Append

The Append box lets you select columns that will be used to append the input tables. Those columns are placed in a <c> node directly as children of the node <Append>.

Simple example: Append input tables based on columns ‘age’ and ‘class of worker’

```
<Append keyHD='161847461101504584' module='DefaultActions' idx='1' x='0' y='0'
>
  <c>age</c>
  <c>class of worker</c>
</Append>
```

Complete example:

```
<Append keyHD='#1' module='#5' idx='#2' x='#3' y='#4' output='#6' check='#7'>
  <c>#8</c>
  <c>#8</c>
</Append>
```

3.6.1. Standard Attributes

ID	Name	Value Type	Description	Default Value
1	keyHD	int	Number generated by anatella and used by anatella for caching purpose (used only for GUI)	
2	idx	int	ID of current box used for connections	
3	x	int	Number that defines the horizontal position of the box in the graph (used only for GUI)	0
4	y	int	Number that defines the vertical position of the box in the graph (used only for GUI)	0
5	module	string	Constant value : must be: ‘DefaultActions’	none
6	output	[0..6]	Drop-down ‘Output Columns’	3
7	check	0,1	Checkbox ‘check all input tables before start’	1
8	c.text	string	Name of selected column	

3.7. FilterOnKey

The FilterOnKey box lets you select many columns for the master table (pin 0) and one column for the reference table (pin 1). The columns selected from master table are placed in a <c> node directly as children of the node <FilterOnKey>. The column selected from reference table is placed as an attribute of node <FilterOnKey>.

Simple example:

```
<FilterOnKey module='DefaultActions' idx='1' x='0' y='0' slaveKey='Key2'>
  <c>Key1</c>
</FilterOnKey>
```

Complete example:

```
<FilterOnKey keyHD='#1' module='#5' idx='#2' x='#3' y='#4' checkUniquePKInSlav
es='#7' truncate='#8' slaveKey='#6'>
  <c>#9</c>
  <c>#9</c>
  <c>#9</c>
</FilterOnKey>
```

3.7.1. Standard Attributes

ID	Name	Value Type	Description	Default Value
1	keyHD	int	Number generated by anatella and used by anatella for caching purpose (used only for GUI)	
2	idx	int	ID of current box used for connections	
3	x	int	Number that defines the horizontal position of the box in the graph (used only for GUI)	0
4	y	int	Number that defines the vertical position of the box in the graph (used only for GUI)	0
5	module	string	Constant value : must be: 'DefaultActions'	none
6	slaveKey	string	Name of selected column in Reference Table (pin 1)	
7	checkUniquePKInSlaves	0,1	Checkbox	1
8	truncate	0,1	Checkbox	0
9	c.text	string	Name of selected column in Master Table (pin 0)	

3.8. ReplaceStrings

The ReplaceStrings box lets you select many columns to be processed. Each parameter tab corresponds to a child node :

'Replace Whole Column' : <Global>
'Replace partial string' : <String>
'Regular Exp' : <RegExp>
'String (Pre)Processing' : <Processing>

For each of those tabs, different set of columns can be selected in the 'Replacements' panel. For each line in 'Replacements' panel, many columns can be selected. One line is thus represented by one <Group> child node of the appropriate parent tab node as long as the group of columns selected is different from other groups (in the case of two lines with same set of columns selected, no new group will be added but a new <r> child node will appear in <replacements> node). Each <Group> has 2 child nodes : <vars> and <replacements>.

The <vars> node contains one child node per selected column for the current row in the 'Replacements panel'. Each selected columns is placed inside a <c> node.

The <replacements> node contains one child node <r> per replacement for a given common group of columns. Each <r> node contains one <s> and one <d> node (before and after).

The last tab 'String (Pre)Processing' is formed a bit differently and is simpler because there are no <Group> tag. Instead, selected columns are directly placed as child nodes of <Processing> within a <c> node.

<u>Simple example:</u> Replace in column 'age' every occurrence of '60' by '61'

```
<ReplaceStrings keyHD='161847665701504801' module='DefaultActions' idx='1' x='
0' y='0'>
      <Global>
            <Group>
                  <vars>
                        <c>age</c>
                  </vars>
                  <replacements>
                        <r>
                              <s>60</s>
                              <d>61</d>
                        </r>
                  </replacements>
            </Group>
      </Global>
      <String>
      </String>
      <RegExp>
      </RegExp>
      <Processing>
      </Processing>
</ReplaceStrings>
```

<u>Complete example:</u>

```
<ReplaceStrings keyHD='#1' module='#5' idx='#2' x='#3' y='#4'>
      <Global>
            <Group>
                  <vars>
                        <c>#6</c>
                  </vars>
                  <replacements>
                        <r option='#11'>
                              <s>#7</s>
```

```
                <d>#8</d>
            </r>
        </replacements>
    </Group>
    <Group>
        <vars>
            <c>#6</c>
            <c>#6</c>
        </vars>
        <replacements>
            <r>
                <s isNull='#9'/>
                <s></s>
                <d isNull='#10'/>
                <d></d>
            </r>
        </replacements>
    </Group>
</Global>
<String>
    <Group>
        <vars>
            <c>#6</c>
        </vars>
        <replacements>
            <r>
                <s>#7</s>
                <d>#8</d>
            </r>
        </replacements>
    </Group>
    <Group>
        <vars>
            <c>#6</c>
        </vars>
        <replacements>
            <r>
                <s>#7</s>
                <d>#8</d>
            </r>
        </replacements>
    </Group>
</String>
<RegExp sep='#12'>
    <Group>
        <vars>
            <c>#6</c>
        </vars>
        <replacements>
            <r>
                <s>#7</s>
                <d>#8</d>
            </r>
         </replacements>
    </Group>
    <Group>
```

```
            <vars>
                <c>#6</c>
            </vars>
            <replacements>
                <r option='#11'>
                    <s>#7</s>
                    <d>#8</d>
                </r>
            </replacements>
        </Group>
    </RegExp>
    <Processing trim='#15' punctuation='#13' simplify='#14' capitalize='#16'
lowercase='#17' unAccent='#18' titleCase='#20' caseFolded='#21' replaceNonLati
n1='#19' removeFirst='#22' removeLast='#23' truncate='#24'>
        <c>#6</c>
    </Processing>
</ReplaceStrings>
```

3.8.1. Standard Attributes

ID	Name	Value Type	Description	Default Value
1	keyHD	int	Number generated by anatella and used by anatella for caching purpose (used only for GUI)	
2	idx	int	ID of current box used for connections	
3	x	int	Number that defines the horizontal position of the box in the graph (used only for GUI)	0
4	y	int	Number that defines the vertical position of the box in the graph (used only for GUI)	0
5	module	string	Constant value : must be: 'DefaultActions'	none
6	c.text	string	Name of selected columns	
7	s.text	string	'Before' value common for 'Replace Whole Column', 'Replace partial string', 'Regular Exp' tabs	Before
8	d.text	string	'After' value common for 'Replace Whole Column', 'Replace partial string', 'Regular Exp' tabs	After
9	s.isNull	0,1	'Before is Null' in 'Replace Whole Column' tab	0
10	d.isNull	0,1	'After is Null' in 'Replace Whole Column' tab	0
11	r.option	int, [0..9]	'Option' value in 'Replace Whole Column' and 'Regular Exp' tab	0
12	sep	string	'(separator:)' value in 'Regular Exp' tab	,
13	punctuation			0
14	simplify			0
15	trim			0
16	capitalize		to Upper case (mutually exclusive with 'lowercase')	0
17	lowercase		mutually exclusive with 'capitalize'	0
18	unAccent			0
19	replaceNonLatin1			0
20	titleCase			0
21	caseFolded			0
22	removeFirst			0
23	removeLast			0

24	truncate			0

3.9. Calculator

The <Calculator> node is composed of two child nodes : <InputVars> and <OutputVars>.

<InputVars> has one child node <InputVar/> per selected input column.

<OutputVars> has one child node <OutputVar> per output column defined. The text value of this node is the expression value.

Simple example: Add one to column age

```
<Calculator keyHD='1618479618268000162' module='MathParser' idx='1' x='0' y='0'
>
  <InputVars>
    <InputVar column='age'/>
  </InputVars>
  <OutputVars>
    <OutputVar name='NextAge'>age+1</OutputVar>
  </OutputVars>
</Calculator>
```

Complete example:

```
<Calculator keyHD='#1' module='#5' idx='#2' x='#3' y='#4' castToKey1='#16' cas
tToKey2='#17' castToKey3='#18' addN='#30' abortWhenError='#32' useJIT='#31' ad
dNow='#23' addNowS='#19' seed='#29' elapsedTimeUnit='#24' dateFormatType='#20'
dateFormat='#21' isUTC='#22' emptyETToNull='#26' referenceTime='#25' castToET1
='#27' castToET2='#28'>
  <InputVars>
    <InputVar column='#6'/>
    <InputVar column='#6' label='#8' isNumber='#9' value='#10'/>
  </InputVars>
  <OutputVars>
    <OutputVar name='#7'>
      #15
    </OutputVar>
    <OutputVar name='#7' meta='#11' update='#12' isInput='#13' notes='#14'>
      #15
    </OutputVar>
  </OutputVars>
</Calculator>
```

3.9.1. Standard Attributes

ID	Name	Value Type	Description	Default Value
1	keyHD	int	Number generated by anatella and used by anatella for caching purpose (used only for GUI)	
2	idx	int	ID of current box used for connections	
3	x	int	Number that defines the horizontal position of the box in the graph (used only for GUI)	0
4	y	int	Number that defines the vertical position of the box in the graph (used only for GUI)	0
5	module	string	Constant value : must be: 'MathParser'	none
6	column	string	Name of the selected input column (left panel)	
7	name	string	Name of the output column (right panel)	
8	label	string	'Variable Name' in selected input columns	same name as selected column
9	isNumber	0,1	Checkbox 'Cast To Number' in selected input columns	1
10	value	string	'Current Value' in selected input columns	0
11	meta	['F','K', 'U']	Drop-down 'Meta-type' of output column	F
12	update	0,1	Drop-down 'New Var' or 'Update Var'	0
13	isInput	0,1	Checbox 'is Input Var.' below expression	0
14	notes	string	'Notes' below expression	
15	Outputvar.text	string	Expression to be computed	

3.9.2. Advanced Attributes

ID	Name	Value Type	Description	Default Value
16	castToKey1	[0..2]	Radio button 'because negative'	2
17	castToKey2	[0..2]	Radio button 'becausegreater then 4,…'	2
18	castToKey3	[0..2]	Radio button 'because not-an-integer-number'	0
19	addNowS	0,1	Extra Parameters	0
20	dateFormatType	[0..6]	Extra Parameters	0
21	dateFormat	string	Extra Parameters	
22	isUTC	0,1	Extra Parameters	1
23	addNow	int	Extra Parameters	0
24	elapsedTimeUnit	[0..4]	Extra Parameters	2
25	referenceTime	0,1	Extra Parameters	0
26	emptyETToNull	0,1	CheckBox : Convert Empty String to Null ET	1
27	castToET1	1,2	Extra Parameters	1
28	castToET2	1,2	Extra Parameters	1
29	seed	0,1	Extra Parameters	0
30	addN	0,1	Extra Parameters	1
31	useJIT	string	Extra Parameters	1
32	abortWhenError	0,1	Extra Parameters	1

3.10. WriteCSV

The writeCSV box has advanced parameters where the user can define a subset of columns to be written. About the XML structure, this is represented by adding one child node <c> for each selected column.

Simple example:

```
<writeCSV idx='1' x='0' y='0' sep=',' filename=':/myFile.csv'/>
```

Complete example:

```
<writeCSV keyHD='#1' idx='#2' x='#3' y='#4' sep='#6' useComma='#16' firstLineC
ontainsColumnNames='#8' eol='#17' forceQualifier='#18' noNans='#14' textqualif
ier='#7' fillInNull='#13' encoding='#12' floatToString='#15' split='#19' segme
ntSize='#20' splitVar='#21' writeAllRows='#9' testCol='#10' testValue='#11' fi
lename='#5' checkNameCollision='#22' createDir='#23' writeAllColumns='#24'>
  <c>#25</c>
  <c>#25</c>
  <c>#25</c>
</writeCSV>
```

3.10.1. Standard Attributes

ID	Name	Value Type	Description	Default Value
1	keyHD	int	Number generated by anatella and used by anatella for caching purpose (used only for GUI)	
2	idx	int	ID of current box used for connections	
3	x	int	Number that defines the horizontal position of the box in the graph (used only for GUI)	0
4	y	int	Number that defines the vertical position of the box in the graph (used only for GUI)	0
5	filename	string		
6	sep	string	Delimiters	,
7	textqualifier	string		"
8	firstLineContainsColumnNames	0,1		1

3.10.2. Advanced Attributes

ID	Name	Value Type	Description	Default Value
9	writeAllRows	0,1	Radio button choice ‘Write…’	1
10	testCol	string	name of selected column for option ‘Write only the rows where’	
11	testValue	string	value of ‘is equal to’	
12	encoding	[0..2]	Drop-down ‘Character encoding’	0
13	fillInNull	string	‘String that represents the “null” value’	

14	noNans	0,1	CheckBox 'replace Nan,Inf,..'	1
15	floatToString	string	Drop-down 'Float to String Cast'	%g
16	useComma	0,1	Checkbox 'use "comma" as decimal sep..'	0
17	eol	[0..2]	Drop-down 'End-of-Line marker'	0
18	forceQualifier	[0..2]	Drop-down 'Text-Qualifier'	0
19	split	[0..3]	'File splitting' block : radio button choice. Values follows the following order (top to bottom radio button choice) : [0,3,1,2]	0
20	segmentSize		Value of third choice in 'File splitting block'	
21	splitVar		Name of selected column for split	
22	checkNameCollision		First CheckBox in Advanced Parameters	1
23	createDir		Second CheckBox in Advanced Parameters	0
24	writeAllColumns		RadioButton	1
25	c.text		Name of selected column to write in output	

3.11. ReadGel

Simple example:

```
<readGel idx='1' x='0' y='0' fileName=':/myFile.gel_anatella'/>
```

Complete example:

```
<readGel keyHD='#1' idx='#2' x='#3' y='#4' fileNameFromPin='#5' accurateProgre
ssBar='#9' abortIfMissing='#7' allowMoreColumns='#8' timeOut='#10' fileName='#
6'/>
```

3.11.1. Standard Attributes

ID	Name	Value Type	Description	Default Value
1	keyHD	int	Number generated by anatella and used by anatella for caching purpose (used only for GUI)	
2	idx	int	ID of current box used for connections	
3	x	int	Number that defines the horizontal position of the box in the graph (used only for GUI)	0
4	y	int	Number that defines the vertical position of the box in the graph (used only for GUI)	0
5	fileNameFromPin	0,1	Drop-down	0
6	fileName	string		
7	abortIfMissing	0,1		1

3.11.2. Advanced Attributes

ID	Name	Value Type	Description	Default Value
8	allowMoreColumns	0,1	First Checkbox in advanced parameters	0

9	accurateProgressBar	0,1	Second Checkbox in advanced parameters	1
10	timeOut	int	Value in 'Error Management' (seconds)	0

3.12. WriteGel

The writeGel box has advanced parameters where the user can define a subset of columns to be written and/or a subset of columns to avoid writing. About the XML structure, this is represented by adding <write> and <skip> child nodes to the <writeGel> node. Each selected columns is within a <c> node.

Simple example:

```
<writeGel idx='1' x='0' y='0' file=':/myFileOut.gel_anatella'/>
```

Complete example:

```
<writeGel keyHD='#1' idx='#2' x='#3' y='#4'
nThread='#6' writeAllColumns='#11' checkNameCollision='#8' algoC='#9' createDi
r='#10' split='#15' advancedNaming='#18' segmentSize='#16' splitVar='#17' writ
eAllRows='#12' testCol='#13' testValue='#14' deleteTimeOut='#19' file='#5'>
      <write>
            <c>#7</c>
            <c>#7</c>
            <c>#7</c>
      </write>
      <skip>
            <c>#7</c>
            <c>#7</c>
      </skip>
</writeGel>
```

3.12.1. Standard Attributes

ID	Name	Value Type	Description	Default Value
1	keyHD	int	Number generated by anatella and used by anatella for caching purpose (used only for GUI)	
2	idx	int	ID of current box used for connections	
3	x	int	Number that defines the horizontal position of the box in the graph (used only for GUI)	0
4	y	int	Number that defines the vertical position of the box in the graph (used only for GUI)	0
5	file	string		
6	nThread	int	Number of compression Threads	-1

3.12.2. Advanced Attributes

ID	Name	Value Type	Description	Default Value
7	c.text	string	Name of selected columns (for write or skip)	
8	checkNameCollision	0,1	First CheckBox in Advanced Parameters (check for column name collision)	1
9	algoC	0,1	First CheckBox in Advanced Parameters (Compress Data)	1
10	createDir	0,1	Third CheckBox in Advanced Parameters (Create output directory if it does not exists)	0
11	writeAllColumns	0,1	RadioButton	1
12	writeAllRows	0,1	Radio button choice in block 'Row Selection'	1
13	testCol	string	name of selected column for option 'Write only the rows where'	
14	testValue	string	value of 'is equal to'	
15	split	[0..2]	Radio button choice in block 'Anatella Gel file splitting'	0
16	segmentSize	int	Value when option 'Split into several anatella Gel files. Each file is maximum' is selected	100
17	splitVar	string	Selected column for option 'Split into several anatella Gel files based on the variable' (last choice of radio button in 'Anatella Gel file splitting' block)	
18	advancedNaming	0,1	Checkbox appearing when option 'Split into several anatella Gel files based on the variable' is selected (last choice of radio button in 'Anatella Gel file splitting' block)	0
19	deleteTimeOut	int	Value in 'Error Management' block	1

3.13. MultiJoin

The MultiJoin box lets you define multiple joins. Each join corresponds to one new line in the box editor. Each join(each line) is simply represented as a new child node <Join> of the node <MultiJoin>.

For a given join, a subset of columns of the slave table can be selected. Each selected column is represented with <c> nodes that are children of the node <Join>.

Simple example: One Join on 'Key1' and 'Key2'

```
<MultiJoin module='DefaultActions' idx='1' x='0' y='0'>
  <Join pin='1' mainKey='Key1' slaveKey='Key2' prefix=''>
    </Join>
</MultiJoin>
```

Complete example:

```
<MultiJoin keyHD='#1' module='#5' idx='#2' x='#3' y='#4' dataBlockSize='#6' ch
eckUniquePKInSlaves='#7'>
  <Join pin='#8' mainKey='#9' slaveKey='#10' prefix='#11' copyAll='#12' inner=
'#13'>
    <c>#14</c>
    <c>#14</c>
    <c>#14</c>
  </Join>
  <Join pin='#8' mainKey='#9' slaveKey='#10' prefix='#11'>
  </Join>
</MultiJoin>
```

3.13.1. Standard Attributes

ID	Name	Value Type	Description	Default Value
1	keyHD	int	Number generated by anatella and used by anatella for caching purpose (used only for GUI)	
2	idx	int	ID of current box used for connections	
3	x	int	Number that defines the horizontal position of the box in the graph (used only for GUI)	0
4	y	int	Number that defines the vertical position of the box in the graph (used only for GUI)	0
5	module	string	Constant value: must be: 'DefaultActions'	none
6	dataBlockSize	int		10240
7	checkUniquePKInSlaves	0,1		1
8	pin	int	'Pin of Slave Table'	1
9	mainKey	string	'Primary Key in Master Table'	
10	slaveKey	string	'Primaery Key in Slave Table'	
11	prefix	string	'Column Name Prefix'	
12	copyAll	0,1	'Include All columns of Slave Table'	1
13	inner	0,1	'Drop Row If Join Fails (inner join)'	0
14	c.text	string	Column name selected in 'Selected Slave Table Column'	

3.14. Join

The Join box lets you define columns used as keys for the join. The key column from master table (pin 0) is placed as an attribute of node <Join>. Key columns from slave tables (pin!=0) are added as child node <c> of node <SlaveKeys> which is a child of <Join>.

Prefixes can also be defined. Contrary to the selected key columns, the prefix for the master table is placed in a <c> node as well as prefixes for slave tables. Those <c> node are children of a <Prefixes> node which is a child of the <Join> node.

Simple example:

```
<Join module='DefaultActions' idx='1' x='0' y='0' joinMode='1' keyA='Key1' new
='1'>
```

```
  <SlaveKeys>
    <c>KeySlave1</c>
    <c>KeySlave2</c>
  </SlaveKeys>
  <Prefixes>
    <c>prefixMaster_</c>
    <c>prefixSlave1_</c>
    <c>prefixSlave2_</c>
  </Prefixes>
</Join>
```

Complete example:

```
<Join keyHD='#1' module='#5' idx='#2' x='#3' y='#4' joinMode='#6' keyA='#7' ne
w='1'>
  <SlaveKeys>
    <c>#8</c>
    <c>#8</c>
  </SlaveKeys>
  <Prefixes>
    <c>#9</c>
    <c>#9</c>
    <c>#9</c>
  </Prefixes>
</Join>
```

3.14.1. Standard Attributes

ID	Name	Value Type	Description	Default Value
1	keyHD	int	Number generated by anatella and used by anatella for caching purpose (used only for GUI)	
2	idx	int	ID of current box used for connections	
3	x	int	Number that defines the horizontal position of the box in the graph (used only for GUI)	0
4	y	int	Number that defines the vertical position of the box in the graph (used only for GUI)	0
5	module	string	Constant value: must be: 'DefaultActions'	none
6	joinMode	[0..11]	Drop-down 'Type of join'	0
7	keyA	string	Name of selected column for Master Table (pin 0)	
8	SlaveKeys.c.text	string	Name of the selected column coming from a slave table.	
9	Prefixes.c.text	string	The first child node of <Prefixes> is the prefix for the master table. The others are prefixes for slave tables following the order of the child nodes of <SlaveKeys>	
10	closeB	0,1	Checkbox at the bottom when joinMode=[0,1,3,6,7,9]	0

3.15. SelectColumns

The SelectColumns box lets you define columns to be kept or dropped. Each column selected is represented by a child node <c> of the node <SelectColumns>.

Simple example: Selection of 3 columns

```
<SelectColumns module='DefaultActions' idx='1' x='0' y='0' keep='1'>
 <c>age</c>
 <c>class of worker</c>
 <c>detailed industry code</c>
</SelectColumns>
```

Complete example:

```
<SelectColumns keyHD='#1' module='#5' idx='#2' x='#3' y='#4' keep='#6'>
 <c>#7</c>
 <c>#7</c>
 <c>#7</c>
</SelectColumns>
```

3.15.1. Standard Attributes

ID	Name	Value Type	Description	Default Value
1	keyHD	int	Number generated by anatella and used by anatella for caching purpose (used only for GUI)	
2	idx	int	ID of current box used for connections	
3	x	int	Number that defines the horizontal position of the box in the graph (used only for GUI)	0
4	y	int	Number that defines the vertical position of the box in the graph (used only for GUI)	0
5	module	string	Constant value: must be: ‘DefaultActions’	none
6	keep	0,1	Bottom radio button (Keep OR Drop)	1
7	c.text	string	Name of selected columns	

3.16. ColumnRename

The ColumnRename box lets you select columns to be renamed. In the standard options tab, the only information are column names before and after. This is represented by <ColumnRename> having child nodes <before> and <after>. For each rename (each line in the box editor) a new child <c> is added to <before> and <after>.

In addition to that, the tab “Quick Rename” lets you define multiple columns. <ColumnRename> will thus have a third child <QuickRename> where selected columns are added with <c> child nodes as usual.

Simple example: Rename the column “age” to “ageRenamed”

```
<ColumnRename module='DefaultActions' idx='1' x='0' y='0'>
   <before>
```

```
            <c>age</c>
    </before>
    <after>
            <c>ageRenamed</c>
    </after>
    <QuickRename ></QuickRename>
</ColumnRename>
```

Complete example:

```
<ColumnRename keyHD='#1' module='#5' idx='#2' x='#3' y='#4' addMissingColumns=
'#6' caseMode='#7'>
    <before>
            <c>#8</c>
            <c>#8</c>
    </before>
    <after>
            <c>#9</c>
            <c>#9</c>
    </after>
    <QuickRename  prefix='#11' suffix='#12' search='#13' replace='#14'>
            <c>#10</c>
            <c>#10</c>
    </QuickRename>
</ColumnRename>
```

3.16.1. Standard Attributes

ID	Name	Value Type	Description	Default Value
1	keyHD	int	Number generated by anatella and used by anatella for caching purpose (used only for GUI)	
2	idx	int	ID of current box used for connections	
3	x	int	Number that defines the horizontal position of the box in the graph (used only for GUI)	0
4	y	int	Number that defines the vertical position of the box in the graph (used only for GUI)	0
5	module	string	Constant value: must be: 'DefaultActions'	none
6	addMissingColumns	0,1	Bottom Checkbox 'Create NULL column if column is missing'	1
7	before.c.text	string	'Before' value	
8	after.c.text	string	'After' value	

3.16.2. Advanced Attributes

ID	Name	Value Type	Description	Default Value
9	caseMode	[0..3]	Drop-down 'Change case'	0
10	QuickRename.c.text	string	Selected column name	
11	prefix	string		
12	suffix	string		

13	search	string		
14	replace	string		

3.17. ChangeDataType

The ChangeDataType box has 5 tabs, one tab for each type transformation. The representation of the parameters remains very simple.

<ChangeDataType> has one or several children per tab depending of selected parameters:
Tab “to Float” : <toDouble>, <toNULLDouble>
Tab “to Key” : <toKey>
Tab “to Elapsed Time” : <toElpasedTime>
Tab “to String” : < toStringFromFloat>, < toStringFromET>
Tab “Other Parameters” : <toZeroDouble>
For each of those nodes, each selected column will be represented by a child node <c>. Every other parameter than column selection is present as an attribute of <ChangeDataType>.

Simple example: Type of column ‘age’ changed from string to key

```
<ChangeDataType keyHD='161849094919120394' idx='1' x='0' y='0'>
   <toKey>
         <c>age</c>
   </toKey>
</ChangeDataType>
```

Complete example:

```
<ChangeDataType keyHD='#1' idx='#2' x='#3' y='#4' castToKey1='#18' castToKey2=
'#19' castToKey3='#20' castToKey4='#21' toDoubleFailed='#6' strictToDouble='#8
' strictToKey='#14' castToET1='#31' castToET2='#32' floatToString='#34' decima
lSepIn='#7' decimalSepOut='#35' dateFormatTypeOut='#38' elapsedTimeUnitIn='#26
' elapsedTimeUnitOut='#40' dateFormatIn='#23' referenceTimeIn='#25' dateFormat
Out='#37' referenceTimeOut='#39' UTCString='#41' updateSortFlag='#44' emptyDou
bleToNull='#9' emptyKeyToNull='#15' emptyETToNull='#28' naDoubleToNull='#10' n
aKeyToNull='#16' naETToNull='#29' dotDoubleToNull='#11' dotKeyToNull='#17' dot
ETToNull='#30' nullStringToZero='#43' UTCKey='#27' dateFormatTypeIn='#24'>
   <toKey>
         <c>#13</c>
         <c>#13</c>
   </toKey>
   <toDouble>
         <c>#5</c>
         <c>#5</c>
   </toDouble>
   <toNULLDouble>
         <c>#12</c>
         <c>#12</c>
   </toNULLDouble>
   <toZeroDouble>
         <c>#42</c>
         <c>#42</c>
   </toZeroDouble>
   <toElapsedTime>
```

```
            <c>#22</c>
            <c>#22</c>
      </toElapsedTime>
      <toStringFromFloat>
            <c>#33</c>
            <c>#33</c>
      </toStringFromFloat>
      <toStringFromET>
            <c>#36</c>
            <c>#36</c>
      </toStringFromET>
</ChangeDataType>
```

3.17.1. Standard Attributes

ID	Name	Value Type	Description	Default Value
1	keyHD	int	Number generated by anatella and used by anatella for caching purpose (used only for GUI)	
2	idx	int	ID of current box used for connections	
3	x	int	Number that defines the horizontal position of the box in the graph (used only for GUI)	0
4	y	int	Number that defines the vertical position of the box in the graph (used only for GUI)	0

3.17.2. Tab “to Float” Attributes

ID	Name	Value Type	Description	Default Value
5	toDouble.c.text	string	Name of column selected for ‘to Float’	
6	toDoubleFailed	[0..3]	Radio Button “If convertion fails” actual order : [0,1,3,2]	0
7	decimalSepIn	[0..2]	Drop down ‘Decimal Separator’	0
8	strictToDouble	0,1		0
9	emptyDoubleToNull	0,1		1
10	naDoubleToNull	0,1		0
11	dotDoubleToNull	0,1		0
12	toNULLDouble.c.text	string	Name of column selected for ‘Change Floating(nan,inf,-in) to NULL’	

3.17.3. Tab “to Key” Attributes

ID	Name	Value Type	Description	Default Value
13	toKey.c.text	string	Name of column selected for ‘to Float’	
14	strictToKey	0,1		0
15	emptyKeyToNull	0,1		1
16	naKeyToNull	0,1		0
17	dotKeyToNull	0,1		0
18	castToKey1	[0..2]	Radio Button : ‘because not-a-number’	2

			actual value order of radio buttons [1,2,0]	
19	castToKey2	[0..2]	Radio Button : 'because negative' actual value order of radio buttons [1,2,0]	2
20	castToKey3	[0..2]	Radio Button : 'because greater than..' actual value order of radio buttons [1,2,0]	2
21	castToKey4	[0..2]	Radio Button : 'because not-an-integer-number' actual value order of radio buttons [1,2,0]	0

3.17.3. Tab "to Elapsed Time" Attributes

ID	Name	Value Type	Description	Default Value
22	toElapsedTime.c.text	string	Name of column selected for 'to Elapsed Time'	
23	dateFormatIn	string	Date format text when 'dateFormatTypeIn'=0 is selected ('User-specified format')	
24	dateFormatTypeIn	[0..6]	Drop-down 'Date Format'	0
25	referenceTimeIn	string	'Reference Time'	
26	elapsedTimeUnitIn	[0..4]	Drop-down 'Elapsed Time Unit'	2
27	UTCKey	0,1	Checkbox 'Time zone is UTC'	1
28	emptyETToNull	0,1		1
29	naETToNull	0,1		0
30	dotETToNull	0,1		0
31	castToET1	[0..2]	First Radio Button in block 'If conversion fails'	0
32	castToET2	[0..2]	Second Radio Button in block 'If conversion fails'	0

3.17.3. Tab "to String" Attributes

ID	Name	Value Type	Description	Default Value
33	toStringFromFloat.c.text	string	Name of column selected for 'to String from Float or Key'	
34	floatToString	string	Drop-down 'Float to String Cast'	%g
35	decimalSepOut	[0..2]	Drop-down 'Decimal Separator'	0
36	toStringFromET.c.text	string	Name of column selected for 'to String from Elapsed Time'	
37	dateFormatOut	string	Date format text when 'dateFormatTypeOut'=0 is selected ('User-specified format')	
38	dateFormatTypeOut	[0..6]	Drop-down 'Date Format'	0
39	referenceTimeOut	string	'Reference Time'	
40	elapsedTimeUnitOut	[0..4]	Drop-down 'Elapsed Time Unit'	2
41	UTCString	0,1	Checkbox 'Time zone is UTC'	1

3.17.4. Tab "Other Parameters" Attributes

ID	Name	Value Type	Description	Default Value
42	toZeroDouble.c.text	string	Name of column selected for 'Change NULL to zero'	

43	nullStringToZero	0,1	Drop-down order is [1,0]	1
44	updateSortFlag	0,1	Checkbox	1

3.18. ReadExcel

The ReadExcel box lets the user choose columns in the ‘Advanced Parameters’ tab. If columns are selected, <ReadExcel> will have the child node <ColDates> that will have one child node <c> per selected column.

Simple example:

```
<readExcel module='WriteReport' idx='1' x='0' y='0' fileName=':/myFile.xlsx'/>
```

Complete example:

```
<readExcel keyHD='#1' module='#5' idx='#2' x='#3' y='#4' bufferSize='#24' firs
tLineContainsColumnNames='#6' dropEmptyLine='#23' outputFilenames='#25' sheetM
ode='#7' columnMode='#10' rowMode='#15' sheetIdx='#8' sheetName='#9' columnSta
rt1='#11' columnStart2='#13' columnEnd='#12' columnNumber='#14' rowStart1='#16
' rowStart2='#17' rowStart3='#19' rowEnd='#18' rowNumber='#20' ecc='#26' dateF
ormat='#22' fileName=':/myFile.xlsx'>
   <ColDates>
         <c>#21</c>
         <c>#21</c>
   </ColDates>
</readExcel>
```

3.18.1. Standard Attributes

ID	Name	Value Type	Description	Default Value
1	keyHD	int	Number generated by anatella and used by anatella for caching purpose (used only for GUI)	
2	idx	int	ID of current box used for connections	
3	x	int	Number that defines the horizontal position of the box in the graph (used only for GUI)	0
4	y	int	Number that defines the vertical position of the box in the graph (used only for GUI)	0
5	module	string	Constant value : must be: ‘WriteReport’	none
6	firstLineContainsColumnNames	0,1	Checkbox	1
7	sheetMode	0,1	Radio button ‘Select Sheet’ index	0
8	sheetIdx	int	‘Sheet number’	1
9	sheetName	string	‘Sheet with name’	Sheet1
10	columnMode	[0..2]	Radio button ‘Column Range’ index	0
11	columnStart1	string	First ‘starting on column’	A
12	columnEnd	string	‘ending at column’	Z
13	columnStart2	string	Second ‘starting on column’	A
14	columnNumber	int	Number Chooser ‘columns in total’	100
15	rowMode	int	Radio Button ‘Row Range’ index	0
16	rowStart1	int	Second radio button choice	1
17	rowStart2	int	Third radio button choice, param 1	1

18	rowEnd	int	Third radio button choice, param 2	10000
19	rowStart3	int	Fourth radio button choice, param 1	1
20	rowNumber	int	Fourth radio button choice, param 2	10000

3.18.2. Advanced Attributes

ID	Name	Value Type	Description	Default Value
21	ColDates.c.text	string	Name of selected column	
22	dateFormat	string		yyyyMMdd hh:mm:ss
23	dropEmptyLine	0,1	Checkbox ‘drop empty rows’	0
24	bufferSize	int	‘Minimum Input Buffer Size’	1
25	outputFilenames	0,1		0
26	ecc	[0..4]	Drop-Down ‘Replace content of Error Cells with’	0

3.19. WriteExcel

The WriteExcel box lets the user choose columns to write in the ‘Standard Parameters’ tab and other columns in the ‘Advanced Parameters’.

If columns are selected in ‘Standard Parameters’, <writeExcel> will have the child node <ToWrite> that will have one child node <c> per selected column.

If columns are selected in ‘Advanced Parameters’, <writeExcel> will have the child node <ToDate> that will have one child node <c> per selected column.

Simple example:

```
<writeExcel module='WriteReport' idx='1' x='0' y='0' fileNameDestination=':/my
OutFile.xslsx'/>
```

Complete example:

```
<writeExcel keyHD='#1' module='#5' idx='#2' x='#3' y='#4' fileNameDestination=
'#9' checkNameCollision='#18' firstLineContainsColumnNames='#10' forceRecomput
e='#17' writeAllColumns='#15' writeAllRows='#26' createDir='#19' testCol='#27'
 testValue='#28' noNans='#24' floatToString='#25' sheetName='#14' dateFormatTy
pe='#21' dateFormat='#22' dateFormatExcel='#23' createIfNotExists='#7'
fileNameSource='#8' startRow='#11' sheetMode='#12' sheetIdx='#13'>
   <ToWrite>
         <c>#16</c>
         <c>#16</c>
         <c>detailed occupation code</c>
         <c>education</c>
   </ToWrite>
   <ToDate>
         <c>#20</c>
         <c>#20</c>
   </ToDate>
</writeExcel>
```

3.19.1. Standard Attributes

ID	Name	Value Type	Description	Default Value
1	keyHD	int	Number generated by anatella and used by anatella for caching purpose (used only for GUI)	
2	idx	int	ID of current box used for connections	
3	x	int	Number that defines the horizontal position of the box in the graph (used only for GUI)	0
4	y	int	Number that defines the vertical position of the box in the graph (used only for GUI)	0
5	module	string	Constant value : must be: 'WriteReport'	none
6	mode	[0..4]		0
7	createIfNotExists	0,1		1
8	fileNameSource	string	Appears only for 'mode' = [3,4]	
9	fileNameDestination	string		
10	firstLineContainsColumnNames	0,1		1
11	startRow	int	Value in block 'Select Row'. Appears only for 'mode'=[2,4]	1
12	sheetMode	0,1	Radio Button 'Select Sheet'	1
13	sheetIdx	int	First Radio Button choice in block 'Select Sheet'	1
14	sheetName	string	Second Radio Button choice in block 'Select Sheet'	Sheet
15	writeAllColumns	0,1	RadioButton 'Column Selection' (index order of radio button : [1,0])	1
16	ToWrite.c.text	string	Name of the selected column in 'Column Selection'	

3.19.2. Advanced Attributes

ID	Name	Value Type	Description	Default Value
17	forceRecompute	0,1		1
18	checkNameCollision	0,1		1
19	createDir	0,1	Third CheckBox in Advanced Parameters (Create output directory if it does not exists)	0
20	ToDate.c.text	string	Name of selected column in 'Output File Format' block	
21	dateFormatType	[0..6]	Drop-down 'Date Format in Anatella'	0
22	dateFormat	string	Value 'Date Format in Anatella'	
23	dateFormatExcel	[0..11]		0
24	noNans	0,1	Checkbox 'replace Nan,Inf,-Inf with empty cell'	1
25	floatToString	string	Drop-down 'Float to String Cast'	%.16g
26	writeAllRows	0,1	Radio button choice in block 'Row Selection'	1
27	testCol	string	name of selected column for option 'Write only the rows where'	
28	testValue	string	value of 'is equal to'	

3.20. RunToFinishLine

Simple example:

```
<RunToFinishLine idx='1' x='0' y='0'/>
```

Complete example:

```
<RunToFinishLine keyHD='#1' idx='#2' x='#3' y='#4' nPinIn='#5' pinSeparator='#
6'/>
```

3.20.1. Standard Attributes

ID	Name	Value Type	Description	Default Value
1	keyHD	int	Number generated by anatella and used by anatella for caching purpose (used only for GUI)	
2	idx	int	ID of current box used for connections	
3	x	int	Number that defines the horizontal position of the box in the graph (used only for GUI)	0
4	y	int	Number that defines the vertical position of the box in the graph (used only for GUI)	0
5	nPinIn	int	'Total number of input PIN's'	1
6	pinSeparator	int	'Vertical distance..'	12

3.21. InsertKey

Simple example:

```
<InsertKey module='DefaultActions' idx='1' x='0' y='0' startCount='0' columnNa
me='Key'/>
```

Complete example:

```
<InsertKey keyHD='#1' module='#5' idx='#2' x='#3' y='#4' insertKey='#6' keyTyp
e='#7' startCount='#9' columnName='#8'>
  <c name='#10' value='#11' type='#12'/>
  <c name='#10' value='#11' type='#12'/>
</InsertKey>
```

3.21.1. Standard Attributes

ID	Name	Value Type	Description	Default Value
1	keyHD	int	Number generated by anatella and used by anatella for caching purpose (used only for GUI)	
2	idx	int	ID of current box used for connections	
3	x	int	Number that defines the horizontal position of the box in the graph (used only for GUI)	0

4	y	int	Number that defines the vertical position of the box in the graph (used only for GUI)	0
5	module	string	Constant value : must be: 'DefaultActions'	none
6	insertKey	0,1	Checkbox 'Add a key column'	1
7	keyType	0,1	Checkbox 'Meta-type..is "Key"'	1
8	columnName	string	'Name of the Column..'	Key
9	startCount	int	'First key value'	0
10	c.name	string	Block 'Constant' : 'Column Name'	Name
11	c.value	string	Block 'Constant' : 'Value'	value
12	c.type	[U,F,K]	Block 'Constant' : 'Meta Type'	U

3.22. InMemoryJoin

Simple example:

```
<InMemoryJoin keyHD='161855858021748307' module='DefaultActions' idx='1' x='0'
 y='0'  mainKey='Key1' slaveKey='Key2' prefixMain='prefixA_' prefixSlave='pref
ixB_'/>
```

Complete example:

```
<InMemoryJoin keyHD='#1' module='#5' idx='#2' x='#3' y='#4' jm='#6' mainKey='#
7' slaveKey='#9' prefixMain='#8' prefixSlave='#10'/>
```

3.22.1. Standard Attributes

ID	Name	Value Type	Description	Default Value
1	keyHD	int	Number generated by anatella and used by anatella for caching purpose (used only for GUI)	
2	idx	int	ID of current box used for connections	
3	x	int	Number that defines the horizontal position of the box in the graph (used only for GUI)	0
4	y	int	Number that defines the vertical position of the box in the graph (used only for GUI)	0
5	module	string	Constant value : must be: 'DefaultActions'	none
6	jm	0,1	Drop-down 'Type of join'	0
7	mainKey	string	Name of selected column for 'Master Key in ..'	
8	prefixMain	string	'Column-Name Prefix for Master Table (A)'	
9	slaveKey	string	Name of selected column for 'Slave Key (B) on Pin 1'	
10	prefixSlave	string	'Column-Name Prefix for Slave Table (B)'	

3.23. Split

The Split box lets you define columns to be splitted. Each selected column is represented by a new child node <c> of the node <Split>.

Simple example: Split on columns 'colA' and 'colB'

```
<Split keyHD='161855890521748753' module='DefaultActions' idx='1' x='0' y='0'>
  <c>colA</c>
  <c>colB</c>
</Split>
```

Complete example:

```
<Split keyHD='#1' module='#5' idx='#2' x='#3' y='#4' sep='#7' suffix='#11' use
Suffix='#10' trim='#12' fixed='#8' number='#9'>
  <c>#6</c>
  <c>#6</c>
</Split>
```

3.23.1. Standard Attributes

ID	Name	Value Type	Description	Default Value
1	keyHD	int	Number generated by anatella and used by anatella for caching purpose (used only for GUI)	
2	idx	int	ID of current box used for connections	
3	x	int	Number that defines the horizontal position of the box in the graph (used only for GUI)	0
4	y	int	Number that defines the vertical position of the box in the graph (used only for GUI)	0
5	module	string	Constant value : must be: 'DefaultActions'	none
6	c.text	string	Name of selected columns to split	
7	sep	string	'Delimiter'	'space' choice
8	fixed	0,1	Radio button block 'Number of New Columns'	0
9	number	int	'Fixed :' (accessible only if number column selected <2)	10
10	useSuffix	0,1	Radio button block 'New Column Names' Actual index order : [1,0]	1
11	suffix	string		_S
12	trim	0,1	Bottom checkbox	1

3.24. NaiveDeduplicate

The NaiveDeduplicate box lets you define key columns. Each selected column is represented by a new child node <c> of the node <NaiveDeduplicate>.

Simple example: Naïve deduplicate on key columns 'keyA' and 'keyB'

```
<NaiveDeduplicate keyHD='161855957721748602' module='DefaultActions' idx='1' x
='0' y='0'>
  <c>keyA</c>
  <c>keyB</c>
</NaiveDeduplicate>
```

Complete example:

```
<NaiveDeduplicate keyHD='#1' module='#5' idx='#2' x='#3' y='#4' n='#7' counter
Name='#8' checkSort='#9' outputDuplicates='#10' allowMultithread='#11'>
  <c>#6</c>
  <c>#6</c>
</NaiveDeduplicate>
```

3.24.1. Standard Attributes

ID	Name	Value Type	Description	Default Value
1	keyHD	int	Number generated by anatella and used by anatella for caching purpose (used only for GUI)	
2	idx	int	ID of current box used for connections	
3	x	int	Number that defines the horizontal position of the box in the graph (used only for GUI)	0
4	y	int	Number that defines the vertical position of the box in the graph (used only for GUI)	0
5	module	string	Constant value : must be: 'DefaultActions'	none
6	c.text	string	Name of selected columns for 'Primary Key(s)'	
7	n	int	'Number of rows for..'	1
8	counterName	string	'Column Name of the duplicate counter (optional)'	
9	checkSort	0,1	First checkbox	1
10	outputDuplicates	0,1	Second checkbox	0
11	allowMultithread	0,1	Third checkbox	0

3.25. Flatten

The Flatten box lets you define columns to transpose and columns to keep. For each of those columns set there is a corresponding child node of <Flatten> : <ToTranspose> and <ToKeep>. For each selected columns a new child node <c> is added to either <ToTranspose> or <ToKeep>.

Simple example:

```
<Flatten module='DefaultActions' idx='1' x='0' y='0' key='keyA' category='catC
ol'>
    <ToTranspose>
        <c>colToTranspose1</c>
        <c> colToTranspose12</c>
    </ToTranspose>
    <ToKeep>
        <c>colToKeep1</c>
        <c> colToKeep2</c>
        <c> colToKeep3</c>
    </ToKeep>
</Flatten>
```

Complete example:

```
<Flatten keyHD='#1' module='#5' idx='#2' x='#3' y='#4' alternativeNaming='#10'
 checkSort='#13' allowMultithread='#14' useDifferentNulls='#11' outputFullyDef
inedOnly='#12' key='#6' category='#8'>
    <ToTranspose>
        <c>#7</c>
        <c>#7</c>
    </ToTranspose>
    <ToKeep>
        <c>#9</c>
        <c>#9</c>
        <c>#9</c>
    </ToKeep>
</Flatten>
```

3.25.1. Standard Attributes

ID	Name	Value Type	Description	Default Value
1	keyHD	int	Number generated by anatella and used by anatella for caching purpose (used only for GUI)	
2	idx	int	ID of current box used for connections	
3	x	int	Number that defines the horizontal position of the box in the graph (used only for GUI)	0
4	y	int	Number that defines the vertical position of the box in the graph (used only for GUI)	0
5	module	string	Constant value : must be: 'DefaultActions'	none
6	key	string	Name of selected column 'Key (Partitioning) column'	
7	ToTranspose.c.text	string	Name of selected column 'Columns to transpose'	
8	category	string	Name of selected column 'Category column'	
9	ToKeep.c.text	string	Name of selected column 'Columns to keep without any transformation'	
10	alternativeNaming	0,1	Checkbox	0
11	useDifferentNulls	0,1	Checkbox	0

3.25.2. Advanced Attributes

ID	Name	Value Type	Description	Default Value
12	outputFullyDefinedOnly	0,1	First checkbox	0
13	checkSort	0,1	Second checkbox	1
14	allowMultithread	0,1	Third checkbox	0

3.26. ParallelRun

The parallelRun box lets the user add anatella graphs to be run. Each line represents one anatella graph, in the XML this is represented by one node <RunData> child of <parallelRun>. For each graph, the user can define the name, parameters and comments. Each of those parameters is represented by a child node of node <RunData> : <anatellaGraph>, <parameters> and <comments>.

Simple example:

```
<parallelRun idx='1' x='0' y='0' maxConcurrentRun='2' abort='1'>
    <RunData>
        <anatellaGraph>:/GraphToParallelRun.anatella</anatellaGraph>
        <parameters>param1</parameters>
        <comments>myComment</comments>
    </RunData>
</parallelRun>
```

Complete example:

```
<parallelRun keyHD='#1' idx='#2' x='#3' y='#4' maxConcurrentRun='#5' abort='#1
0' displayLogSubGraphs='#13' nRetry='#11' getPeakMem='#14' image='#12'
>
    <RunData>
        <anatellaGraph>#7</anatellaGraph>
        <parameters>#8</parameters>
        <comments>#9</comments>
    </RunData>
    <RunData run='#6'>
        <anatellaGraph>#7</anatellaGraph>
        <parameters>#8</parameters>
        <comments>#9</comments>
    </RunData>
</parallelRun>
```

3.26.1. Standard Attributes

ID	Name	Value Type	Description	Default Value
1	keyHD	int	Number generated by anatella and used by anatella for caching purpose (used only for GUI)	
2	idx	int	ID of current box used for connections	
3	x	int	Number that defines the horizontal position of the box in the graph (used only for GUI)	0
4	y	int	Number that defines the vertical position of the box in the graph (used only for GUI)	0
5	maxConcurrentRun	int		2
6	RunData.run	0,1	Checkbox 'Run?'	1
7	RunData.anatellaGraph.text	string	'Anatella Graph To Run'	
8	RunData.parameters.text	string	'Parameters'	
9	RunData.comments.text	string	'Comments'	

3.26.2. Advanced Attributes

ID	Name	Value Type	Description	Default Value
10	abort	[0..3]	Drop-down ‘Abort condition’	1
11	nRetry	int	‘Number of “Retries”’	5
12	image	string	‘Icon’	
13	displayLogSubGraphs	0,1	First checkbox	0
14	getPeakMem	0,1	Second checkbox	0

3.27. ListFiles

Simple example:

```
<ListFiles module='DefaultActions' idx='1' v='2.37' x='0' y='0'/>
```

Complete example:

```
<ListFiles keyHD='#1' module='#5' idx='#2' v='#6' x='#3' y='#4' filter='#7'
directory='#8' dtFormat='#11' sort='#9' type='#10' nothing='#14' getSize='#12'
recurse='#13'/>
```

3.27.1. Standard Attributes

ID	Name	Value Type	Description	Default Value
1	keyHD	int	Number generated by anatella and used by anatella for caching purpose (used only for GUI)	
2	idx	int	ID of current box used for connections	
3	x	int	Number that defines the horizontal position of the box in the graph (used only for GUI)	0
4	y	int	Number that defines the vertical position of the box in the graph (used only for GUI)	0
5	module	string	Constant value : must be: ‘DefaultActions’	none
6	v	string	Constant value : must be: ‘2.37’	2.08
7	filter	string	‘File Filter’	*
8	directory	string	‘Directory to list’	:/
9	sort	[0..9]	Drop-down ‘Output’	0
10	type	[0..2]	Drop-down ‘Output type’	0
11	dtFormat	string	‘Date Time Format’	yyyyMMdd hh:mm:ss
12	getSize	0,1	Checkbox	0
13	recurse	0,1	Checkbox	0
14	nothing	[0..2]	Drop-down ‘If no files are found’	0

3.28. Generic template

The ‘Generic template’ lets the user defines its own javascript box. Some built-in anatella box are derived from this box. The user has the possibility to add custom box parameters, image, description, javascript code and breakpoints in that code. To enable all the tabs, the user must switch to expert user mode (which can be achieved by clicking the icon in the top bar of anatella).

Custom parameters appear in the XML as <Parameter> nodes children of the <Parameters> node which is a child of the <Generic> node.

For each parameter, the user can define a value for this parameter; the value is placed as the ‘text’ of the <Parameter> node.

In some cases, meta-parameters can be defined for a given parameter. When this happens, the value is no longer simply placed as ‘text’ of the <Parameter> node but instead it will have two new children: <meta> and <data>. The text of the node <meta> will contain the meta-parameters and the text of the node <data> will contain the value of the parameter.

The description, image, javascript code and breakpoints are each represented by a child node of the node <Generic>, respectively : <Description>, <EmbeddedIcon>,<Script>,<BreakPoints>.

The <BreakPoints> node will have one child node <OneBK> per breakpoint.

Note for anatella box that derives from this template: (see 3.29, 3.30, 3.31, 3.36)

All parameters and attributes that are shown in the ‘Simple example’ section of concerned boxes must remain unchanged.

<u>Simple example:</u>

```
<Generic idx='1' x='0' y='0' id='template'
deprecated='1'
longName='this operator simply &quot;pass by&quot; the rows without doing anyt
hing'
image=''
author='Frank Vanden Berghen'
revision='0.01'
pdfDestination='5_24_2_template'>
    <Description>
      SOME TEXT DESCRIPTION
    </Description>
    <Parameters>
        <Parameter id='name' text='a short description' type='combobox'>
        <meta>
            &lt;i&gt;+&lt;/i&gt;&lt;i&gt;-&lt;/i&gt;
        </meta>
        <data>0</data>
        </Parameter>
        <Parameter id='name2' text='a short description2' type='string'>
            myString
        </Parameter>
    </Parameters>
    <Script>
      SOME SCRIPT
    </Script>
</Generic>
```

Complete example:

```
<Generic keyHD='#1' idx='#2' v='#44' x='#3' y='#4'
castDoubleToKey1='#33'
castDoubleToKey2='#34'
castDoubleToKey3='#35'
castStringToDouble='#28'
castStringToKey1='#29'
castStringToKey2='#30'
castStringToKey3='#31'
castStringToKey4='#32'
libs='#27'
id='#5'
orignalId='#9b'
deprecated='#47'
longName='#10'
crypted='#38'
nPinIn='#19'
nPinOut='#23'
isVariableNPinIn='#22'
isVariableNPinOut='#26'
nPinInMin='#20'
nPinInMax='#21'
nPinOutMin='#24'
nPinOutMax='#25'
pinSeparator='#42'
abortOnMissingColumn='#45'
shape='#41'
color='#39'
keywords='#13'
image='#14'
author='#12'
revision='#11'
meta='#43'
tags='#46'
pdf='#37'
pdfDestination='#36'>
    <Description>
        #15
    </Description>
    <EmbeddedIcon>
        #40
    </EmbeddedIcon>
     <Parameters>
        <Parameter id='#8' text='#6' type='#9'>
            #7
        </Parameter>
        <Parameter id='#8' text='#6' type='#9' ep='#7c'>
             <meta>#7a</meta>
            <data>#7b</data>
        </Parameter>
    </Parameters>
    <Script>
        #18
    </Script>
    <BreakPoints>
        <OneBK enabled='#16' lineNumber='#17' ignoreCount='0'/>
        <OneBK enabled='#16' lineNumber='#17' ignoreCount='0'/>
    </BreakPoints>
</Generic>
```

3.28.1. Standard Attributes

ID	Name	Value Type	Description	Default Value
1	keyHD	int	Number generated by anatella and used by anatella for caching purpose (used only for GUI)	
2	idx	int	ID of current box used for connections	
3	x	int	Number that defines the horizontal position of the box in the graph (used only for GUI)	0
4	y	int	Number that defines the vertical position of the box in the graph (used only for GUI)	0
5	id	string	'Script name' common for tab 'Parameters' and tab 'Description'	
6	Parameters.Parameter.text (XML node attribute)	string	Parameter : 'Description' column	a short description
7	Parameters.Parameter.text (XML node text)	variable	Parameter : 'Value' column If column 'Meta-Parameters' defined (e.g. with type 'combobox' selected) : <meta>(#7a) node is placed here along with a <data>(#7b) node containing the value of the 'Value' column. <meta> and <data> nodes are children of <Parameter> node.	value
7c	ep	int	When the parameter on this row is of the "password" type, then, by default, the value of the parameter is saved as an encrypted string, unless ep='0'	1
8	Parameters.Parameter.id	string	Parameter : 'id in code' column	name
9	Parameters.Parameter.type	string	Parameter : 'type'column	string
9b	orignalId	string	Constant value : must be: 'template' Appears when script name(#5) is changed	none

3.28.2. Tab "Description" Attributes

ID	Name	Value Type	Description	Default Value
10	longName	string	'Script short description'	this operator simply &quot;pass by&quot; the

				rows without doing anything
11	revision	int		0.01
12	author	string		Frank Vanden Berghen
13	keywords	string		
14	image	string	Path to the box image selected	
15	Description.text	string	‘Long description’	This operator simply "pass by" the rows from the input pin to the output pin without doing any treatment. This operator is simply a "Basic Template" that you can can "tweak" to create your own, complex operator.

3.28.3. Tab “Code” Attributes

ID	Name	Value Type	Description	Default Value
16	BreakPoints.OneBK .enabled	0,1	Checkbox in column ‘Active’ in ‘BreakPoints’ table. (you must set a breakpoint somewhere in the code by clicking on the left of a row number to make a new breakpoint)	1
17	BreakPoints.OneBK .lineNumber	int	Column ‘LineNumber’ in ‘BreakPoints’ table. Line number where the breakpoint is placed	
18	Script.text	string	Code text	

3.28.4. Tab “Configuration” Attributes

ID	Name	Value Type	Description	Defaul t Value
19	nPinIn	int	Input PIN’s : ‘Default Number’	1
20	nPinInMin	int	Input PIN’s : ‘Minimum Number’	1
21	nPinInMax	int	Input PIN’s : ‘Maximum Number’	0
22	isVariableNPinIn	0,1	Input PIN’s : Checkbox ‘number of PIN is variable’	0
23	nPinOut	int	Output PIN’s : ‘Default Number’	1
24	nPinOutMin	int	Output PIN’s : ‘Minimum Number’	1
25	nPinOutMax	int	Output PIN’s : ‘Maximum Number’	0
26	isVariableNPinOut	0,1	Output PIN’s : Checkbox ‘number of PIN is variable’	0
27	libs	string	‘Anatella Javascript Extra Libraries’	
28	castStringToDouble	[7,8,9,0]	‘Cast to “Float” From “UnKnown/String” type’. Actual radio button indexes : [7,8,9,0]	9
29	castStringToKey1	[10,11,0]	‘Cast to “Key” from “UnKnown/String” type. ‘If not-a-number’ radio button actual order : [10,11,0]	11
30	castStringToKey2	[12,13,0]	‘Cast to “Key” from “UnKnown/String” type.	13

			'If negative' radio button actual order : [12,13,0]	
31	castStringToKey3	[14,15,0]	'Cast to "Key" from "UnKnown/String" type. 'If greater than..' radio button actual order : [14,15,0]	15
32	castStringToKey4	[16,17,0]	'Cast to "Key" from "UnKnown/String" type. 'If not-an-integer-number' radio button actual order : [16,17,0]	0
33	castDoubleToKey1	[3,4,0]	'Cast to "Key" Type from "Float" type'. 'If negative' radio button actual order : [3,4,0]	4
34	castDoubleToKey2	[5,6,0]	'Cast to "Key" Type from "Float" type'. 'If greater than..' radio button actual order : [3,4,0]	6
35	castDoubleToKey3	[1,2,0]	'Cast to "Key" Type from "Float" type'. 'If not-an-integer-number' radio button actual order : [3,4,0]	0
36	pdfDestination	string		5_24_2_template
37	pdf	string	'PDF File'	
38	crypted	0,1	Checkbox 'Encrypt code'	0
39	color	string	'Background Color'	#ffffff
40	EmbeddedIcon.text	string	XML node appears when checkbox 'Embed icon inside .anatella file' is checked. Value is computed by anatella and is taken from selected image in tab 'Description'.	
41	shape	[0..4]		0
42	pinSeparator	int	'Vertical distance between PIN's…'	12
43	meta	string	Composite of parameters in 'Meta-data' block. Drop-down 'Direction' : [n,i,o] Drop-down 'Type' : [f,d,c] Final value is 'Direction'+'Type'+'Meta'. e.g. : Direction = 'Out' ; Type = 'Cloud' ; Meta = 'mymeta' => final value : 'ocmymeta'	nf
44	v	int	'Minimum Anatella version required..'	2.08
45	abortOnMissingColumn	0,1	Drop-down 'When an input column is missing'	1

3.28.5. Tab "Publication" Attributes

ID	Name	Value Type	Description	Default Value
46	tags	string	'Add in categories'	900_User Defined
47	deprecated	0,1	Checkbox 'publish as "deprecated"	1

3.29. Accumulator

Simple example:

```
<Accumulator module='DefaultActions' idx='1' v='2.14' x='0' y='0' colAcc=''/>
```

Complete example:

```
<Accumulator keyHD='#1' module='#5' idx='#2' v='#6' x='#3' y='#4' colAcc='#7'
outColName='#8' startVal='#9' colPartition='#10'/>
```

3.29.1. Standard Attributes

ID	Name	Value Type	Description	Default Value
1	keyHD	int	Number generated by anatella and used by anatella for caching purpose (used only for GUI)	
2	idx	int	ID of current box used for connections	
3	x	int	Number that defines the horizontal position of the box in the graph (used only for GUI)	0
4	y	int	Number that defines the vertical position of the box in the graph (used only for GUI)	0
5	module	string	Constant value : must be: 'DefaultActions'	none
6	v	string	Constant value : must be: '2.14'	2.14
7	colAcc	string	Name of selected column 'Column to accumulate'	
8	outColName	string	'Output column name'	sum
9	startVal	int	'Initial value'	0
10	colPartition	string	Name of selected column 'Partition column'	

3.30. GetMetaData (get_meta-data)

The 'GetMetaData' box is a Generic box, refer to 3.28 for more information.

Example:

```
<Generic idx='1' x='0' y='0' id='get_meta-data'
longName='Get meta-data'
le='1444654687384'
keywords='meta data extract get'
image='/getMetaData.svg'
author='Frank Vanden Berghen'
revision='0.03'
tags='150_Standard'
pdfDestination='5_5_14_get_meta_data'>
    <Description>
```

```
Get meta-data: column name &amp; type + sort meta-data.&lt;br/&gt;&lt;br/&gt;
Do not enable the option &quot;pump all rows from input&quot; unless you know
what you are doing
    </Description>
    <Parameters>
        <Parameter id='readAll' text='pump all rows from input' type='bool'>
            0
        </Parameter>
    </Parameters>
</Generic>
```

3.31. FuzzyJoin

The ‘FuzzyJoin’ box is a Generic box, refer to 3.28 for more information.

Example:

```
<Generic idx='1' x='0' y='0' libs='BagOfWord2' id='fuzzyJoin'
longName='fuzzy join 2 tables'
nPinIn='2'
nPinInMin='2'
le='1516711883018'
keywords='fuzzy join word text jaro winkler damerau leven dice'
image='/fuzzyJoin.svg'
author='Frank Vanden Berghen'
revision='0.08'
tags='350_Text Mining, 100_Join Tables'
pdfDestination='5_4_8_fuzzy_join_javascript_act'>
    <Description></Description>
    <Parameters>
        <Parameter id='idKeyToFind' text='Key in Master Table' type='onecolumn'>
            <meta>0</meta>
            <data></data>
        </Parameter>
        <Parameter id='idRefKey' text='Key in Slave Table' type='onecolumn'>
            <meta>1</meta>
            <data></data>
        </Parameter>
        <Parameter id='idRefId' text='Column to Join in Slave Table' type='manycol
umns'>
            <meta>1</meta>
            <data></data>
        </Parameter>
        <Parameter id='idKNNSize' text='Find k-NN. K=?' type='double'>
            3
        </Parameter>
        <Parameter id='idType' text='Type of similarity' type='combobox'>
            <meta>&lt;i&gt;Damereau LevenStein similarity&lt;/i&gt;
&lt;i&gt;Jaro Winkler similarity&lt;/i&gt;
&lt;i&gt;Dice Coefficient similarity&lt;/i&gt;
&lt;i&gt;Damereau LevenStein distance&lt;/i&gt;</meta>
            <data>2</data>
        </Parameter>
        <Parameter id='idPartition1' text='Parition Var in Main Table (optional)'
type='onecolumn'>
            <meta>0</meta>
            <data></data>
        </Parameter>
        <Parameter id='idPartition2' text='Parition Var in Reference (optional)' t
ype='onecolumn'>
            <meta>1</meta>
            <data></data>
        </Parameter>
        <Parameter id='idPrefix' text='Prefix to add on joined column name (option
al)' type='string'>
        </Parameter>
    </Parameters>
</Generic>
```

3.32. ReadSQLite

The ‘ReadSQLite’ box lets the user select one or multiple SQLite databases and write an SQL query. When more than one database is added, additional databases are placed as child nodes <db> of the node <attach> which is a child of <readSQLite>.

When a direct, static SQL query is defined, the query is placed as text of the node <sql> child of <readSQLite>.

When the SQL query is defined as javascript code, the code is placed as text of the node <javascript> child of <readSQLite>.

Simple example:

```
<readSQLite module='DBConnectors' idx='1' x='0' y='0' fileName=':/db.sqlite'>
    <sql>
        select * from mytable
    </sql>
</readSQLite>
```

Complete example:

```
<readSQLite keyHD='#1' module='#5' idx='#2' x='#3'
y='#4' mode='#6' openMode='1' allowNullOutput='#7' allowError='#17' verbose='#
22' nMaxRunJS='#10' memMax='#20' maxNColumn='#19' lockingMode='#21' firstRowOn
ly='1' timeOut='#18' castModeInt='#25' castModeReal='#31' castFromInt1='#27' c
astFromInt2='#28' castFromInt3='#29' castFromInt4='#30' castFromReal='#32' fil
eName='#14' colSQL='#11' floatToString='#24' spatialite='#23'>
    <sql>
        #8
    </sql>
    <javascript>
        #9
    </javascript>
    <attach>
        <db f='#15' a='#16'/>
        <db f='#15' a='#16'/>
    </attach>
</readSQLite>
```

3.32.1. Standard Attributes

ID	Name	Value Type	Description	Default Value
1	keyHD	int	Number generated by anatella and used by anatella for caching purpose (used only for GUI)	
2	idx	int	ID of current box used for connections	
3	x	int	Number that defines the horizontal position of the box in the graph (used only for GUI)	0
4	y	int	Number that defines the vertical position of the box in the graph (used only for GUI)	0
5	module	string	Constant value : must be: 'DBConnectors'	none
6	mode	[0..3]	Drop-down	0
7	allowNullOutput	0,1	Checkbox 'Allow null SQL result'	1
8	sql.text	string	SQL statement when mode(#6) = 0	

9	javascript.text	string	Javascript statement when mode(#6) = 1	function getSQL(i)
				{
				// if (i==1) return "";
				return "select * from
				barriers";
				}

10	nMaxRunJS	int	'Maximum Number of runs' when mode(#6) = 1	1
11	colSQL	string	Name of selected column when mode(#6) = [2,3]	
12	firstRowOnly	0,1	Checkbox 'Process only the first row..' when mode(#6) = [2,3]	0
13	openMode	[0..3]	Drop-down 'Database Open Mode'	0
14	fileName	string	Name of first database added (with alias = 'main')	
15	attach.db.f	string	Name of additional database added (with alias != 'main')	
16	attach.db.a	string	Alias column value	dbX (where X starts at 1 and is increasing)

3.32.2. Advanced Attributes

ID	Name	Value Type	Description	Default Value
17	allowError	0,1	Checkbox 'Return NULL SQL result if SQLite Error'	0
18	timeOut	int	'Time Out before Failure'	10000
19	maxNColumn	int	'Max number of columns in query'	2000
20	memMax	int	'Max Memory'	200

21	lockingMode	[0..2]	Drop-down ‘Locking Mode’ (enabled when openMode(#13) != 0)	0
22	verbose	0,1	Checkbox	1
23	spatialite	0,1	Checkbox ‘load “mod_spatialite”’	1
24	floatToString	string	Drop-down ‘Float to String Cast’	%.16g
25	castModeInt	[0..2]	Drop-down block ‘Convert from INT’	1
26				
27	castFromInt1	[1,2,0]	When ‘castModeInt’ (#25)=2 : ‘because not-a-number’	2
28	castFromInt2	[1,2,0]	When ‘castModeInt’ (#25)=2 : ‘because negative’	2
29	castFromInt3	[1,2,0]	When ‘castModeInt’ (#25)=2 : ‘because greater than..‘	2
30	castFromInt4	[1,2,0]	When ‘castModeInt’ (#25)=2 : ‘because not-an-integer-number’	0
31	castModeReal	0,1	Drop-down block ‘Convert from REAL’	1
32	castFromReal	[0,1,3,2]	When ‘castModeReal’ (#31)=1	3

3.33. WriteSQLite

The ‘WriteSQLite’ box lets the user select columns to define primary keys and to define a subset of columns to be written.

When primary keys columns are selected the node <writeSQLite> has the child node <keyCols> where each columns is a new child node <c> of the node <keyCols>.

When a subset of columns is selected for write the node <writeSQLite> has the child node <updateCols> where each columns is a new child node <c> of the node <updateCols>.

Simple example:

```
<writeSQLite module='DBConnectors' idx='1' x='0' y='0' fileName=':/myNewSQLite
.sqlite' table='Table1'>
    <keyCols>
        <c>Key</c>
    </keyCols>
    <updateCols></updateCols>
</writeSQLite>
```

Complete example:

```
<writeSQLite keyHD='#1' module='#5' idx='#2' x='#3'
y='#4' operation='#10' dataBaseMode='#7' tableMode='#9' memMax='#20' lockingMo
de='#18' writeAll='#12' commitInterval='#19' writeAllRows='#14' testCol='#15'
testValue='#16' createDir='#17' timeOut='#21' fileName='#6' table='#8'>
    <keyCols>
        <c>#11</c>
        <c>#11</c>
    </keyCols>
    <updateCols>
        <c>#13</c>
        <c>#13</c>
        <c>#13</c>
    </updateCols>
</writeSQLite>
```

3.33.1. Standard Attributes

ID	Name	Value Type	Description	Default Value
1	keyHD	int	Number generated by anatella and used by anatella for caching purpose (used only for GUI)	
2	idx	int	ID of current box used for connections	
3	x	int	Number that defines the horizontal position of the box in the graph (used only for GUI)	0
4	y	int	Number that defines the vertical position of the box in the graph (used only for GUI)	0
5	module	string	Constant value : must be: ‘DBConnectors’	none
6	fileName	string		
7	dataBaseMode	0,1	Drop-down ‘Data base Open Mode’	0
8	table	string	‘Table Name’	Table1
9	tableMode	[0..4]	Drop-down ‘Table Mode’	0
10	operation	[0..3]	Drop-down ‘Operation’	0
11	keyCols.c.text	string	Name of selected columns ‘Primary Keys’	
12	writeAll	0,1	Radio button block ‘Column Selection..’ Actual order [1,0]	1
13	updateCols.c.text	string	Name of selected columns ‘Write only these columns’	
14	writeAllRows	0,1	Radio button block ‘Row Selection’ Actual order [1,0]	1
15	testCol	string	Name of selected column ‘Process only the rows where’	
16	testValue	string	‘is equal to’	

3.33.2. Advanced Attributes

ID	Name	Value Type	Description	Default Value
17	createDir	0,1	Checkbox ‘Create output dir..’	0
18	lockingMode	[0..2]	Drop-down	0
19	commitInterval	int	‘Commit every’	100000
20	memMax	int	‘Max Memory’	200
21	timeOut	int	‘Time Out before Failure’	10000

3.34. SimpleFileOps

The ‘simpleFileOps’ box is a Generic box, refer to 3.28 for more information.

```
<Generic idx='1' x='0' y='0' id='simpleFileOps'
longName='Simple File Operations'
le='1509747159658'
shape='3'
keywords='file copy move rename delete'
image='/fileOps.svg'
author='Frank Vanden Berghen'
revision='0.04'
tags='600_System Tools'>
    <Description>
</Description>
    <Parameters>
        <Parameter id='idMode' text='Mode' type='combobox'>
            <meta>
&lt;i&gt;Copy Files&lt;/i&gt;&lt;i&gt;Move (or Rename) Files&lt;/i&gt;&lt;i&gt
;Delete Files&lt;/i&gt;
            </meta>
            <data>0</data>
        </Parameter>
        <Parameter id='idSource' text='Source File' type='onecolumn'>
            <meta>0</meta>
            <data></data>
        </Parameter>
        <Parameter id='idDestination' text='Destination File' type='onecolumn'
>
            <meta>0</meta>
            <data></data>
        </Parameter>
        <Parameter id='idNAttempts' text='Maximum number of attempts' type='do
uble'>2</Parameter>
        <Parameter id='msToSleep' text='Wait for X milliseconds between each a
ttempt. X=?' type='double'>
            5000
        </Parameter>
        <Parameter id='idAbort' text='Abort if this still fails after all the
attempts' type='bool'>
            0
        </Parameter>
    </Parameters>
    <Script>
    </Script>
</Generic>
```

3.35. BagOfWord

Simple example:

```
<BagOfWord module='BagOfWord2' idx='1' x='0' y='0' column='textCol'/>
```

Complete example:

```
<BagOfWord keyHD='#1' module='#5' idx='#2' x='#3'
y='#4' column='#6' prefix='#7' mode='#10' stemmer='#11' language='#12' tfidf='
#13' minCount='#8' maxWordLen='#9'/>
```

3.35.1. Standard Attributes

ID	Name	Value Type	Description	Default Value
1	keyHD	int	Number generated by anatella and used by anatella for caching purpose (used only for GUI)	
2	idx	int	ID of current box used for connections	
3	x	int	Number that defines the horizontal position of the box in the graph (used only for GUI)	0
4	y	int	Number that defines the vertical position of the box in the graph (used only for GUI)	0
5	module	string	Constant value : must be: ‘BagOfWord2’	none
6	column	string	Name of selected column ‘Column to Bag’	
7	prefix	string	‘Column Name Prefix’	T_
8	minCount	int	‘X=’	42
9	maxWordLen	int	‘Y=’	40
10	mode	[0..2]	Radio button block ‘Stemmer’	1
11	stemmer	string	Drop-down ‘This constant stemmer’	ENGLISH
12	language	string	Name of selected column for ‘The stemmer for the language specified inside the column’	Language1
13	Tfidf	0,1	Checkbox ‘Output TF/IDF ..’	0

3.36. LoopAnatellaGraphs2

The ‘LoopAnatellaGraphs2’ box is a Generic box, refer to 3.28 for more information.

Example:

```
<Generic idx='10' v='2.44' x='-396' y='-108' id='loopAnatellaGraphs2'
  longName='run a Graph in a loop'
  le='1629065646601'
  abortOnMissingColumn='0'
  shape='3'
  keywords='loop run anatella graph meta control process .anatella'
  image='/loop.svg'
  author='Frank Vanden Berghen'
  revision='0.09'
  tags='600_System Tools'
  pdfDestination='5_20_5_loop_anatella_graphs'>
      <Description>
      Run one Anatella graphs in a loop
      </Description>
      <Parameters>
             <Parameter id='idFile' text='Anatella Graph to run inside loop'
             type='onefileopen'>
                   <meta>Anatella files (*.anatella)</meta><data></data>
             </Parameter>
             <Parameter id='idName' text='Global Param Name' type='string'>
                   gpname
```

```
            </Parameter>
            <Parameter id='idValue' text='Global Param Value' type='onecolumn'>
                   <meta>0</meta><data></data>
            </Parameter>
            <Parameter id='idGPV' text='Global Parameters to pass to the graph
            (comma-separated)' type='string'>
            </Parameter>
            <Parameter id='hasCounter' text='add an iteration counter?'
            type='bool'>
                   0
            </Parameter>
            <Parameter id='idCounterName' text='Iteration counter name'
            type='string'>
                   gpcounter
            </Parameter>
            <Parameter id='ncpu' text='run N graph(s) in parallel. N=?'
            type='string'>
                   1
            </Parameter>
            <Parameter id='idSilent' text='Silent Mode?' type='bool'>
                   1
            </Parameter>
            <Parameter id='idVerbosity' text='Message Verbosity inside Log'
            type='combobox'>
                   <meta>&lt;i&gt;Silent&lt;/i&gt;&lt;i&gt;Normal
                   Verbosity&lt;/i&gt;&lt;i&gt;Show the logs of the executed
                   graphs&lt;/i&gt;</meta><data>2</data>
            </Parameter>
            <Parameter id='idAbortAction' text='When run fails:'
            type='combobox'>
                   <meta>&lt;i&gt;Never Abort And No Retry&lt;/i&gt;
                   &lt;i&gt;Never Abort But Retry 5 Times&lt;/i&gt;
                   &lt;i&gt;Abort If Still Failed after 1
                   &quot;Retry&quot;&lt;/i&gt;
                   &lt;i&gt;Abort If Still Warning after 1
                   &quot;Retry&quot;&lt;/i&gt;
                   &lt;i&gt;Abort If Still Failed after 5
                   &quot;Retry&quot;&lt;/i&gt;
                   &lt;i&gt;Abort If Still Warning after 5
                   &quot;Retry&quot;&lt;/i&gt;</meta><data>2</data>
            </Parameter>
      </Parameters>
</Generic>
```

3.37. LoopAnatellaGraphsAdv2

The ‘LoopAnatellaGraphsAdv2’ box is a Generic box, refer to 3.28 for more information.

Example:

```
<Generic idx='11' v='2.44' x='-408' y='24' id='loopAnatellaGraphsAdv2'
  longName='run a Graph in a loop'
  le='1628629722749'
  abortOnMissingColumn='0'
```

```
shape='3'
keywords='loop run anatella graph meta control process .anatella'
image='/loop2.svg'
author='Frank Vanden Berghen'
revision='0.09'
tags='600_System Tools'
pdfDestination='5_20_6_loop_anatella_graphs_adv'>
    <Description>
    Run the different Anatella graphs given in input
    </Description>
    <Parameters>
          <Parameter id='idFile' text='Anatella Graphs to run inside loop'
          type='onecolumn'>
                <meta>0</meta><data>Graph</data>
          </Parameter>
          <Parameter id='idParam' text='Global Loop Params (optional)'
          type='manycolumns'>
                <meta>0</meta><data></data>
          </Parameter>
          <Parameter id='idTest' text='Test Column (optional)'
          type='onecolumn'>
                <meta>0</meta><data></data>
          </Parameter>
          <Parameter id='idGPV' text='Global Parameters to pass to the graph
          (comma-separated)' type='string'>
          </Parameter>
          <Parameter id='hasCounter' text='add an iteration counter?'
          type='bool'>
                0
          </Parameter>
          <Parameter id='idCounterName' text='Iteration counter name'
          type='string'>
                gpcounter
          </Parameter>
          <Parameter id='ncpu' text='run N graph(s) in parallel. N=?'
          type='string'>
                1
          </Parameter>
          <Parameter id='idSilent' text='Silent Mode?' type='bool'>
                1
          </Parameter>
          <Parameter id='idVerbosity' text='Message Verbosity inside Log'
          type='combobox'>
                <meta>&lt;i&gt;Silent&lt;/i&gt;&lt;i&gt;Normal
                Verbosity&lt;/i&gt;&lt;i&gt;Show the logs of the executed
                graphs&lt;/i&gt;</meta><data>2</data>
          </Parameter>
          <Parameter id='idAbortAction' text='When run fails:'
          type='combobox'>
                <meta>&lt;i&gt;Never Abort And No Retry&lt;/i&gt;
                &lt;i&gt;Never Abort But Retry 5 Times&lt;/i&gt;
                &lt;i&gt;Abort If Still Failed after 1
                &quot;Retry&quot;&lt;/i&gt;
                &lt;i&gt;Abort If Still Warning after 1
                &quot;Retry&quot;&lt;/i&gt;
                &lt;i&gt;Abort If Still Failed after 5
                &quot;Retry&quot;&lt;/i&gt;
```

```
                    &lt;i&gt;Abort If Still Warning after 5
                    &quot;Retry&quot;&lt;/i&gt;</meta><data>2</data>
                </Parameter>
            </Parameters>
</Generic>
```

3.38. loadShape

Simple example:

```
<loadShape module='DBConnectors' idx='1' v='2.38' x='0' y='0'/>
```

Complete example:

```
<loadShape keyHD='#1' module='#5' idx='#2' v='#6' x='#3' y='#4' fileName='#7'
outputFileNames='#15' allowError='#16' charset='#8' SRID='#9' reprojection='#14'
castModeInt='#18' castFromInt1='#19' castFromInt2='#20' castFromInt3='#21'
castFromInt4='#22' castToReal='#23' blobMode='#17' encodingMode='#10'
charsetCol='#11' sridMode='#11' SRIDCol='#13'/>
```

3.38.1. Standard Attributes

ID	Name	Value Type	Description	Default Value
1	keyHD	int	Number generated by anatella and used by anatella for caching purpose (used only for GUI)	
2	idx	int	ID of current box used for connections	
3	x	int	Number that defines the horizontal position of the box in the graph (used only for GUI)	0
4	y	int	Number that defines the vertical position of the box in the graph (used only for GUI)	0
5	module	string	Constant value : must be: ‘DBConnectors’	none
6	v	string	Constant value : must be: ‘2.38’	2.38
	mode	0,1	Drop down ‘Constant Filename/From input pin’	Constant Filename
7	fileName	string	if mode==0: file path	
	fileNameCol	string	if mode==1: Name of selected column for ‘From input pin’	filePath

8	charset	string	Drop down
			if mode==0: ‘Character Encoding’
			if mode==1: ‘Character Encoding:Constant’
				CP1252

9	SRID	int	if mode==0: ‘SRID’
			if mode==1: ‘SRID:Constant’
				4326

10	encodingMode	0,1	if mode==1: Radio Button ‘Character Encoding’	0
11	charsetCol	string	if mode==1: Name of selected column for ‘Character Encoding:From Column:’	charset
12	sridMode	0,1	if mode==1: Radio Button ‘SRID’	0
13	SRIDCol	string	if mode==1:	srid

			Name of selected column for ‘SRID:From Column:’	
14	reprojection	0,1	Checkbox ‘Reproject to SRID=4326…’	1

3.38.2. Advanced Attributes

ID	Name	Value Type	Description	Default Value
15	outputFileNames	0,1	Checkbox ‘Output filenames’	0
16	allowError	0,1	Checkbox ‘Skip Errors with a warning’	0
17	blobMode	[0,1,2]	Drop down ‘Geometry (Blob) field’	2
18	castModeInt	0,1	Drop down ‘Convert from the Guessed type “INTEGER”:convert to:’	0
19	castFromInt1	[1,2,0]	if castModeInt==1: Radio button ‘because not-a-number’	2
20	castFromInt2	[1,2,0]	if castModeInt==1: Radio button ‘because negative’	2
21	castFromInt3	[1,2,0]	if castModeInt==1: Radio button ‘because greater than ..’	2
22	castFromInt4	[1,2,0]	if castModeInt==1: Radio button ‘because not-an-integer-number’	0
23	castToReal	[0,1,3,2]	Radio button	3

3.39. joinGIS

The joinGIS box lets the user do set operations on shape files. The user has the possibility to select a set of columns from table on pin 1 to output. If columns are selected, <joinGIS> will have one child node <c> per selected column.

Simple example:

```
<joinGIS module='DBConnectors' idx='1' x='0' y='0'/>
```

Complete example:

```
<joinGIS keyHD='#1' module='#5' idx='#2' x='#3' y='#4' mode='#7' joinType='#6'
outputAllColumns='#17' debug='#19' userInsert='#20' userMakeGeom='#22'
userQuery='#24' geometryA='#10' geometryB='#14' joinColA='#9' joinColB='#13'
SRIDA='#8' prefixA='#11' prefixB='#15' nominalA='#12' nominalB='#16'
userInsertSQL='#21' userMakeGeomSQL='#23' userSQL='#25'>
       <c>#18</c>
       <c>#18</c>
</joinGIS>
```

3.39.1. Standard Attributes

ID	Name	Value Type	Description	Default Value

1	keyHD	int	Number generated by anatella and used by anatella for caching purpose (used only for GUI)	
2	idx	int	ID of current box used for connections	
3	x	int	Number that defines the horizontal position of the box in the graph (used only for GUI)	0
4	y	int	Number that defines the vertical position of the box in the graph (used only for GUI)	0
5	module	string	Constant value : must be: ‘DBConnectors’	none
6	joinType	0,1	Radio button ‘Type of join’	1
7	mode	[0..7]	Drop down ‘There is a “match” when’	0
8	SRIDA	int	‘SRID’	4326
9	joinColA	string	Name of selected column for ‘Table (A) on pin 0: Geometry Join column’	
10	geometryA	[0..4]	Drop down ‘Table (A) on pin 0: Geometry format’	0
11	prefixA	string	‘Table (A) on pin 0: Column-Name Prefix’	
12	nominalA	string	Name of selected column for ‘Table (A) on pin 0: Additional Nominal Join Key(Optional)’	
13	joinColB	string	Name of selected column for ‘Table (B) on pin 1: Geometry Join column’	
14	geometryB	[0..4]	Drop down ‘Table (B) on pin 1: Geometry format’	4
15	prefixB	string	‘Table (B) on pin 1: Column-Name Prefix’	
16	nominalB	string	Name of selected column for ‘Table (B) on pin 1: Additional Nominal Join Key(Optional)’	
17	outputAllColumns	1,0	Radio button ‘Output Column Selection’	1
18	c.text	string	Name of selected columns to output	

3.39.2. Advanced Attributes

ID	Name	Value Type	Description	Default Value
19	debug	0,1	Checkbox ‘Display debug informations …’	0
20	userInsert	0,1	Drop down ‘Command (1) …’	0
21	userInsertSQL	string	Line edit ‘Command (1) …’	
22	userMakeGeom	0,1	Drop down ‘Command (2) …’	0
23	userMakeGeomSQL	string	Line edit ‘Command (2) …’	
24	userQuery	0,1	Drop down ‘Command (3) …’	0
25	userSQL	string	Line edit ‘Command (3) …’	

3.40. inlineTable

The inlineTable box lets the user define a table with custom number of rows, columns, content and column titles. For each column, a new <c> node is added to the node <ColumnNames> child of <inlineTable> containing the name of the column. For each row, a new <r> node is added to the node <Rows> child of the node <inlineTable>. Each <r> node contains as mush children as the number of columns, representing one cell of the table. Each cell is represented with a node <c>, child of the node <r>.

Simple example:

```
<inlineTable idx='1' x='0' y='0'>

	<ColumnNames>
		<c>C1</c>
	</ColumnNames>
	<Rows>
		<r>
			<c></c>
		</r>
	</Rows>
</inlineTable>
```

Complete example:

```
<inlineTable keyHD='#1' idx='#2' x='#3' y='#4' nullValue='#5' trim='#6'>
	<ColumnNames>
		<c>#7</c>
		<c>#7</c>
	</ColumnNames>
	<Rows>
		<r>
			<c>#8</c>
			<c>#8</c>
		</r>
		<r>
			<c>#8</c>
			<c>#8</c>
		</r>
		<r>
			<c>#8</c>
			<c>#8</c>
		</r>
		<r>
			<c>#8</c>
			<c>#8</c>
		</r>
	</Rows>
</inlineTable>
```

3.40.1. Standard Attributes

ID	Name	Value Type	Description	Default Value
1	keyHD	int	Number generated by anatella and used by anatella for caching purpose (used only for GUI)	
2	idx	int	ID of current box used for connections	
3	x	int	Number that defines the horizontal position of the box in the graph (used only for GUI)	0
4	y	int	Number that defines the vertical position of the box in the graph (used only for GUI)	0
5	nullValue	string	'Modality that represent the "NULL"'	null

6	trim	0,1	Checkbox ‘Trim content?’	1
7	ColumnNames.c.text	string	Column name	‘Cx’ with x natural
8	Rows.r.c.text	string	cell value	

3.41. readJSON

The readJSON box lets the user define a JSON structure to extract using levels and XPaths. Each level is represented with a <Level> node child of the root <readJSON> node. For a particular level, each new field to extract is represented with a node <e> child of the <Level> node.

Simple example:

```
<readJSON module='WriteReport' idx='1' x='0' y='0' fileName=''>

      <Level initialPath=''></Level>

</readJSON>
```

Complete example:

```
<readJSON keyHD='#1' module='#5' idx='#2' x='#3' y='#4' bufferSize='#13'
autoFillNRows='#14' fileName='#7' outputFilenames='#16' outputSubTags='#15'
oneJSONPerRow='#9' readTagsFromInPin='#17' colJSON='#8' fromInputPin='#6'>
 <Level initialPath='#10'>
  <e name='#11' path='#12'/>
  <e name='#11' path='#12'/>
 </Level>
 <Level initialPath='#10'>
  <e name='#11' path='#12'/>
  <e name='#11' path='#12'/>
 </Level>
 <Level initialPath='#10'>
  <e name='#11' path='#12'/>
  <e name='#11' path='#12'/>
 </Level>
</readJSON>
```

3.41.1. Standard Attributes

ID	Name	Value Type	Description	Default Value
1	keyHD	int	Number generated by anatella and used by anatella for caching purpose (used only for GUI)	
2	idx	int	ID of current box used for connections	
3	x	int	Number that defines the horizontal position of the box in the graph (used only for GUI)	0
4	y	int	Number that defines the vertical position of the box in the graph (used only for GUI)	0
5	module	string	Constant value : must be: ‘WriteReport’	none
6	fromInputPin	[0,1,2]	Drop down input source	0
7	fileName	string	if ‘fromInputPin’ == 0: name of the input file	

8	colJSON	string	if ‘fromInputPin’ != 0: name of selected column	
9	oneJSONPerRow	0,1	Checkbox ‘One JSON structure per row’	0
10	Level.initialPath	string	‘Iterate on all the subtags located at’	
11	e.name	string	In table ‘Extract the following fields:’ ‘Column Name’	
12	e.path	string	In table ‘Extract the following fields:’ ‘XPath’	

3.41.2. Advanced Attributes

ID	Name	Value Type	Description	Default Value
13	bufferSize	int	‘Minimum Input Buffer Size’	1
14	autoFillNRows	int	‘Number of rows to analyse for Auto Fill-In’	100
15	outputSubTags	0,1	Checkbox ‘Output Subtag names as additional columns’	0
16	outputFilenames	0,1	Checkbox ‘Output Filenames as an additional column’	0
17	readTagsFromInPin	0,1	Checkbox ‘read tags to extract from pin 1’	0

3.42. writeJSON

The writeJSON box lets the user defines the structure of the output JSON using levels. Each level is represented with a <Level> node as a child of the <Levels> node child of the root <writeJSON> nodes. For each level, several items can be defined and are represented with a <Item> node, child of the <Level> node. For some items, several columns can be defined. Those columns are each placed inside a <c> node, child of the <Item> node. Additionally, a prefix and a suffix can be defined and are placed respectively in a <Prefix> and <Suffix> node, child of the root <writeJSON> node.

Simple example:

```
<writeJSON module='WriteReport' idx='1' x='0' y='0' fileName=''>

    <Levels>
        <Level>
            <Item text='{'/>
            <Item type='1'/>
            <Item text='}'/>
        </Level>
    </Levels>
</writeJSON>
```

Complete example:

```
<writeJSON keyHD='#1' module='#5' idx='#2' v='#6' x='#3' y='#4' fileName='#7'
fileMode='#8' checkNameCollision='#9' createDir='#10' writeAllRows='#11'
testCol='#12' testValue='#13' nullString='#15' nullNumber='#17' nan='#18'
floatToString='#16' replaceLF='#14' splitFile='#19' advancedNaming='#20'
splitCol='#21'>
    <Prefix>#22</Prefix>
    <Suffix>#23</Suffix>
    <Levels>
        <Level commaSeparated='#24'>
            <Item text='#27'/>
            <Item type='#28'/>
            <Item type='#28'><c>#30</c><c>#30</c></Item>
            <Item type='#28'/>
            <Item type='#28'><c>#30</c><c>#30</c></Item>
            <Item type='#28' calledLevel='#29'/>
            <Item type='#28'/>
            <Item type='#28'><c>#30</c></Item>
        </Level>
        <Level name='#25' partitionCol='#26'>
            <Item text='#27'/>
            <Item type='#28'/>
        </Level>
        <Level name='#25' partitionCol='#26'>
            <Item type='#28' calledLevel='#29'><c>#30</c><c>#30</c></Item>
        </Level>
    </Levels>
</writeJSON>
```

3.42.1. Standard Attributes

ID	Name	Value Type	Description	Default Value
1	keyHD	int	Number generated by anatella and used by anatella for caching purpose (used only for GUI)	
2	idx	int	ID of current box used for connections	
3	x	int	Number that defines the horizontal position of the box in the graph (used only for GUI)	0
4	y	int	Number that defines the vertical position of the box in the graph (used only for GUI)	0
5	module	string	Constant value : must be: ‘WriteReport’	none
6	v	string	Constant value : must be: ‘2.17’	2.17
7	fileName	string	‘FileName’	
8	fileMode	0,1	Checkbox ‘File Open Mode’	0
9	checkNameCollision	0,1	Checkbox ‘check for column name collision’	1
10	createDir	0,1	Checkbox ‘Create output directory if it does not exists’	0
11	writeAllRows	1,0	Radio button block ‘Row Selection’	1
12	testCol	string	Name of selected column for: ‘Write only the rows where’	
13	testValue	string	Content of line edit for: ‘Write only the rows where … is equal to’	
14	replaceLF	0,1	Block ‘Output File Format’: Checkbox ‘Replace all carriage returns with..’	0
15	nullString	[0,1,2]	Block ‘Output File Format’: drop down ‘For “Null” cells of String Meta-Type’	2
16	floatToString	string	Block ‘Output File Format’: drop down ‘Float to String Cast’	%g
17	nullNumber	0,1	Block ‘Output File Format’: drop down ‘For “Null” cells of Key/Float Meta-Type’	1

18	nan	[0,1,2]	Block ‘Output File Format’: drop down ‘For “NaN, Inf, -Inf” cells’	1
19	splitFile	0,1	Radio button block ‘JSON File Splitting’	0
20	advancedNaming	0,1	if ‘splitFile’ == 1: Block ‘JSON File Splitting’: Checkbox ‘Use advanced naming algorithm..’	0
21	splitCol	string	Block ‘JSON File Splitting’: Name of selected column for ‘Split into several JSON Files based on the variable’	

3.42.2. Levels Attributes

ID	Name	Value Type	Description	Default Value
22	Prefix.text	string	‘Free Text Prefix’	
23	Suffix.text	string	‘Free Text Suffix’ (bottom)	
24	Level.commaSeparated	0,1	Checkbox ‘Repeat (each iteration is separated by a comma “,”)’	1
25	Level.name	text	if at least 2 levels: ‘Name of this level’	‘Lx’ with x natural
26	Level.partitionCol	string	if at least 2 levels: ‘We stay at this level as long as the column’: Name of selected column	
27	Item.text	string	In the table: if ‘Item’ == ‘A simple text’: ‘Parameter(s)’ value	
28	Item.type	[0..7]	In the table: Drop down ‘Item’ if selection != ‘A simple text’	
29	Item.calledLevel	string	In the table: if ‘Item’ == 5: ‘Parameter(s)’: name of selected level	
30	c.text	string	In the table: if ‘Item’ == [2,4,7]: ‘Parameter(s)’: name of selected columns	

3.43. propagateDown

The propagateDown box lets the user define columns to be propagated. Each selected column is placed in a <c> node. All <c> nodes are children of the node <colsPropagate> which is a child of the root node <PropagateDown>.

Simple example:

```
<PropagateDown module='DefaultActions' idx='1' v='2.16' x='0' y='0'/>
```

Complete example:

```
<PropagateDown keyHD='#1' module='#5' idx='#2' v='#6' x='#3' y='#4' rule='#7'
mode='#8' colPartition='#11' checkSort='#12' suffix='#10'>
      <colsPropagate>
```

```
            <c>#9</c>
            <c>#9</c>
      </colsPropagate>
</PropagateDown>
```

3.43.1. Standard Attributes

ID	Name	Value Type	Description	Default Value
1	keyHD	int	Number generated by anatella and used by anatella for caching purpose (used only for GUI)	
2	idx	int	ID of current box used for connections	
3	x	int	Number that defines the horizontal position of the box in the graph (used only for GUI)	0
4	y	int	Number that defines the vertical position of the box in the graph (used only for GUI)	0
5	module	string	Constant value : must be: 'DefaultActions'	none
6	v	string	Constant value : must be: '2.16'	2.16
7	rule	0,1	Drop down 'Propagation rule'	0
8	mode	0,1	Drop down 'Output mode'	1
9	c.text	string	Name of selected columns for 'Columns to propagate'	
10	suffix	string	if mode==0: 'Suffix new columns'	_
11	colPartition	string	Name of selected column for 'Partitioning column'	
12	checkSort	0,1	Checkbox 'Check sort on partitioning column'	1

3.44. R_HistogramGGP2

The 'R_HistogramGGP2' box is a R box which is completely similar to a Generic box except for the start and end tag, refer to 3.28 for more information.

Example:

```
<R idx='1' x='0' y='0' id='R_HistogramGGP2'
  longName='Histograms from Already Aggregated Data'
 crypted='1'
  nPinIn='2'
  isVariableNPinIn='1'
  nPinInMax='19'
  le='1595560549884'
  abortOnMissingColumn='0'
  color='#e1fff0'
  keywords='histogram bar chart plot visualization'
  image='/R_Histogram.svg'
  author='Daniel Soto Zeevaert'
  revision='1.08'
  tags='365_R Visualization'
  pdfDestination='5_10_1_histogram__action_'>
      <Description>
```

```
Creates Many histograms from each aggregate information
available.&lt;br/&gt;
Make sure all computed aggregates are numeric.
</Description>
<Parameters>
        <Parameter id='idMain' text='Plot Title' type='string'>
                Histograms
        </Parameter>
        <Parameter id='idPlotLabels' text='Plot Labels (separated by Comma)'
        type='string'>
        </Parameter>
        <Parameter id='idScaleC' text='Y Axis: Maximum value for COUNT
        (0=auto)' type='double'>
                0
        </Parameter>
        <Parameter id='idScale' text='Y Axis:Maximum Y Axis value for VALUE
        (0=auto)' type='double'>
                0
        </Parameter>
        <Parameter id='idBM' text='Y Axis: Thousands Separator'
        type='string'>
                ,
        </Parameter>
        <Parameter id='idMaxLen' text='LBL: Max length of labels'
        type='double'>
                35
        </Parameter>
        <Parameter id='idSort' text='LBL: Categories Order for Bar Plots
        (String axis only)' type='combobox'>
                <meta>&lt;i&gt;None&lt;/i&gt;&lt;i&gt;Alphabetical&lt;/i&gt
                ;&lt;i&gt;Increasing&lt;/i&gt;&lt;i&gt;Decreasing&lt;/i&gt;
                </meta><data>2</data>
        </Parameter>
        <Parameter id='nBreak' text='LBL: Number of &quot;breaks&quot; (for
        Continuous vars)' type='double'>
                20
        </Parameter>
        <Parameter id='Xangle' text='LBL: Angle for discrete variables'
        type='double'>
                45
        </Parameter>
        <Parameter id='FS' text='CHT: Font Size' type='double'>
                12
        </Parameter>
        <Parameter id='idxH' text='CHT: Orientation Horizontal?'
        type='bool'>
                1
        </Parameter>
        <Parameter id='idxColor1' text='CHT: Color for COUNT ' type='color'>
                #00ff7f
        </Parameter>
        <Parameter id='idxColor2' text='CHT: Color for other variables'
        type='color'>
                #00aaff
        </Parameter>
        <Parameter id='idxColor3' text='CHT: Optional Line Color'
        type='color'>
                #ff4000
        </Parameter>
        <Parameter id='idAlpha' text='CHT: Line Opacity' type='double'>
            0.6
        </Parameter>
        <Parameter id='idxOne' text='CHT: All plots in one Window'
        type='bool'>
            1
        </Parameter>
        <Parameter id='idxColNum' text='CHT: Maximum Number of Charts per
        Row' type='double'>
            1
        </Parameter>
        <Parameter id='idxMargin' text='CHT: Chart Margin' type='double'>
            15
        </Parameter>
        <Parameter id='idxRun' text='IMG: Run Only Mode' type='bool'>
            0
        </Parameter>
        <Parameter id='idxSave' text='IMG: Save Images in PNG?' type='bool'>
            0
        </Parameter>
        <Parameter id='idxDir' text='IMG: PNG Directory' type='onedir'>
            <data>:/</data>
        </Parameter>
        <Parameter id='idW' text='IMG: Multiplot PNG width [Pixel]'
        type='double'>
            1000
        </Parameter>
        <Parameter id='idHt' text='IMG: Multiplot PNG height [Pixel]'
        type='double'>
            400
        </Parameter>
        <Parameter id='myT0' text='whole table' type='allcolumns'>
        </Parameter>
        <Parameter id='myT1' text='' type='allcolumns'>1</Parameter>
        <Parameter id='myT2' text='' type='allcolumns'>2</Parameter>
        <Parameter id='myT3' text='' type='allcolumns'>3</Parameter>
        <Parameter id='myT4' text='' type='allcolumns'>4</Parameter>
        <Parameter id='myT5' text='' type='allcolumns'>5</Parameter>
        <Parameter id='myT6' text='' type='allcolumns'>6</Parameter>
        <Parameter id='myT7' text='' type='allcolumns'>7</Parameter>
        <Parameter id='myT8' text='' type='allcolumns'>8</Parameter>
        <Parameter id='myT9' text='' type='allcolumns'>9</Parameter>
        <Parameter id='myT10' text='' type='allcolumns'>10</Parameter>
        <Parameter id='myT11' text='' type='allcolumns'>11</Parameter>
        <Parameter id='myT12' text='' type='allcolumns'>12</Parameter>
        <Parameter id='myT13' text='' type='allcolumns'>13</Parameter>
        <Parameter id='myT14' text='' type='allcolumns'>14</Parameter>
        <Parameter id='myT15' text='' type='allcolumns'>15</Parameter>
        <Parameter id='myT16' text='' type='allcolumns'>16</Parameter>
        <Parameter id='myT17' text='' type='allcolumns'>17</Parameter>
        <Parameter id='myT18' text='' type='allcolumns'>18</Parameter>
        <Parameter id='myT19' text='' type='allcolumns'>19</Parameter>
    </Parameters>
</R>
```

3.45. readODBC

The readODBC box lets the user make SQL query to a distant database. Depending on the mode, queries can be written directly in the box configuration. In that cas, queries are saved in a <sql> node child of the root node <readODBC>. Queries can also be written with javascript. In that case, queries are saved in a <javascript> node child of the root node <readODBC>.

Simple example:

```
<readODBC module='DBConnectors' idx='1' x='0' y='0'>

</readODBC>
```

Complete example:

```
<readODBC keyHD='#1' module='#5' idx='#2' v='#6' x='#3' y='#4' abortMode='#15'
rowSetBufferSize='#16' allowNullOutput='#12' inLog='#13' useUTF16='#18'
safeModeforBlob='#14' discardUnconvertible='#20' encoding='#19' memMin='#17'
odbcname='#7' mode='#8' colSQL='#11'>
       <sql>#9</sql>
       <javascript>#10</javascript>

</readODBC>
```

3.45.1. Standard Attributes

ID	Name	Value Type	Description	Default Value
1	keyHD	int	Number generated by anatella and used by anatella for caching purpose (used only for GUI)	
2	idx	int	ID of current box used for connections	
3	x	int	Number that defines the horizontal position of the box in the graph (used only for GUI)	0
4	y	int	Number that defines the vertical position of the box in the graph (used only for GUI)	0
5	module	string	Constant value : must be: ‘DBConnectors’	none
6	v	string	Constant value : must be: ‘2.27’	2.27
7	odbcname	string	Drop down ‘Connection Name’	
8	mode	[0,4,1,2,3,5]	Drop down ‘Mode’	0
9	sql.text	string	if mode == [0,4] Text edit	
10	javascript.text	string	if mode == 1 Text edit	function getSQL(i){ return “”;}
11	colSQL	string	if mode == [1,2,3] Text edit	

3.45.2. Advanced Attributes

ID	Name	Value Type	Description	Default Value

12	allowNullOutput	0,1	Checkbox 'Allow null SQL result'	1
13	inLog	0,1	Checkbox 'add SQL statement(s) in log'	0
14	safeModeforBlob	0,1	Checkbox 'Use Slow&Safe mode …'	1
15	abortMode	[0..3]	Drop down 'Abort condition'	2
16	rowSetBufferSize	int	'Fetch buffer row size'	100
17	memMin	int	'Minimum Memory Space for Columns with Strings'	1
18	useUTF16	0,1	Checkbox 'use UTF-16 to request text …'	1
19	encoding	string	if 'useUTF16' == 0: Drop down 'The Database is limited to this encoding'	UTF-8
20	discardUnconvertible	0,1	if 'useUTF16' == 0: Checkbox 'Allow (Discard) Invalid Characters'	1

3.46. upsertODBC

The upsertODBC lets the user connect to a distant database and insert/update data to it. The user can choose a subset of columns to be inserted/updated. Those columns are saved in <c> nodes, children of the node <updateCols> which is a child of the root node <upsertODBC>. The user can choose a name for the table that is saved in a <table> node, child of the root node <upsertODBC>. Additionally, the user can select primary keys columns. Those columns are saved in <c> nodes, children of the node <keyCols> which is a child of the root node <upsertODBC>.

Simple example:

```
<upsertODBC module='DBConnectors' idx='1' v='2.12' x='0' y='0' operation='0'
odbcname=''>

  <table></table>
  <keyCols></keyCols>
  <updateCols></updateCols>
</upsertODBC>
```

Complete example:

```
<upsertODBC keyHD='#1' module='#5' idx='#2' v='#6' x='#3' y='#4'
passThroughAll='#13' rowSetBufferSize='#14' commitInterval='#15'
allowStrangeChars='#16' checkMeta='#25' charEncoding='#19' statusName='#18'
writeAllRows='#20' writeAllColumns='#11' testCol='#21' testValue='#22'
quote='#17' top1='#23' top2='#24' operation='#9' odbcname='#7'>
      <table>#8</table>
      <keyCols>
            <c>#10</c>
            <c>#10</c>
      </keyCols>
      <updateCols>
            <c>#12</c>
            <c>#12</c>
      </updateCols>
</upsertODBC>
```

3.46.1. Standard Attributes

ID	Name	Value Type	Description	Default Value
1	keyHD	int	Number generated by anatella and used by anatella for caching purpose (used only for GUI)	
2	idx	int	ID of current box used for connections	
3	x	int	Number that defines the horizontal position of the box in the graph (used only for GUI)	0
4	y	int	Number that defines the vertical position of the box in the graph (used only for GUI)	0
5	module	string	Constant value : must be: ‘DBConnectors’	none
6	v	string	Constant value : must be: ‘2.12’	2.12
7	odbcname	string	Drop down ‘Connection Name’	
8	table.text	string	‘Table Name’	
9	operation	[0,1,3,2]	Radio button ‘Operation’	0
10	keyCols.c.text	string	If operation == [1,3] Name of selected columns for ‘Primary Keys’	
11	writeAllColumns	0,1	Radio button in block ‘Columns to insert/update’	0
12	updateCols.c.text	string	Name of selected columns for ‘Some columns’	
13	passThroughAll	0,1	Checkbox ‘pass-through to the output pins..’	0

3.46.2. Advanced Attributes

ID	Name	Value Type	Description	Default Value
14	rowSetBufferSize	int	‘BULK operations: Insert’	1
15	commitInterval	int	if radio button in block ‘Transctions-Related Options’ == ‘Group N “Insert’s” inside one transaction. N=?’	1000
16	allowStrangeChars	0,1	Checkbox ‘Allow “strange” characters..’	0
17	quote	string	Drop down ‘Write column names between’	
18	statusName	string	‘Status Column Name	ODBC_Status_1
19	charEncoding	[0,1,2]	Drop down ‘Character Encoding’	0
20	writeAllRows	[1,0]	Radio button in block ‘Row Selection’	1
21	testCol	string	Name of selected column in block ‘Row Selection’	
22	testValue	string	Edit text in block ‘Row Selection’ after ‘is equal to:’	
23	top1	string	First edit text in block ‘Workaround for “SQL-Describe-Param..’	TOP 1
24	top2	string	Second edit text in block ‘Workaround for “SQL-Describe-Param..’	
25	checkMeta	0,1	Checkbox ‘Check that the Anatella Columns with floating-point...’	1

3.47. intervalJoin

The intervalJoin box lets the user select a subset of columns to output. Those columns are saved in <c> nodes, children of the root node <IntervalJoin>.

Simple example:

```
<IntervalJoin module='DefaultActions' idx='1' x='0' y='0' mainLowKey=''
slaveLowKey='' slaveHighKey=''/>
```

Complete example:

```
<IntervalJoin keyHD='#1' module='#5' idx='#2' x='#3' y='#4' mode='#6'
outputRowWhenNoMatch='#7' outputAllColumns='#15' mainLowKey='#8' mainHighKey='#9'
slaveLowKey='#11' slaveHighKey='#12' partitionColumnMaster='#10'
partitionColumnSlave='#14' prefixSlave='#13' outputMode='#17'>
        <c>#16</c>
        <c>#16</c>
</IntervalJoin>
```

3.47.1. Standard Attributes

ID	Name	Value Type	Description	Default Value
1	keyHD	int	Number generated by anatella and used by anatella for caching purpose (used only for GUI)	
2	idx	int	ID of current box used for connections	
3	x	int	Number that defines the horizontal position of the box in the graph (used only for GUI)	0
4	y	int	Number that defines the vertical position of the box in the graph (used only for GUI)	0
5	module	string	Constant value : must be: 'DefaultActions'	none
6	mode	[0,1,2]	Drop down 'Join Mode'	0
7	outputRowWhenNoMatch	0,1	Checkbox 'Output a row from the Master Table when…'	1
8	mainLowKey	string	Name of selected column for block 'Master Table A (Pin 0)' : If mode == 0 : 'Point Value' Else : 'Lower Bound Value'	
9	mainHighKey	string	If mode != 0 : Name of selected column for block 'Master Table A (Pin 0)' : 'Upper Bound Value'	
10	partitionColumnMaster	string	Name of selected column for block 'Master Table A (Pin 0)' : 'Additional Nominal Join Key(optional)'	
11	slaveLowKey	string	Name of selected column for block 'Slave Table B (Pin 1)' : If mode == 2 : 'Point Value' Else : 'Lower Bound Value'	
12	slaveHighKey	string	If mode != 2 : Name of selected column for block 'Slave Table B (Pin 1)' : 'Upper Bound Value'	
13	prefixSlave	string	Block 'Slave Table B (Pin 1)': 'Column-Name Prefix'	
14	partitionColumnSlave	string	Name of selected column for block 'Slave Table B (Pin 1)' : 'Additional Nominal Join Key(optional)'	
15	outputAllColumns	1,0	Radio button in block 'Output Column Selection'	1
16	c.text	string	Name of selected columns in block 'Output Column Selection'	
17	outputMode	[0..4]	Drop down 'Output'	0

3.48. transpose

The ‘transpose’ box is a Generic box, refer to 3.28 for more information.

Example:

```
<Generic idx='1' x='0' y='0' id='transpose'
  longName='transpose the full table'
  le='1404836334296'
  keywords='transpose pivot table'
  image='/transpose.png'
  author='Frank Vanden Berghen'
  revision='0.05'
  tags='800_Other'
  pdfDestination='5_23_2_transpose'>
      <Description>
            Transpose the full table. Only work for SMALL tables since the whole
            table is loaded into RAM.
      </Description>
      <Parameters>
            <Parameter id='firstColContainsColName' text='first Column Contains
            Column Names' type='bool'>
                  0
            </Parameter>
            <Parameter id='hasToCreateColumn' text='Create a column with the
            (old) column names' type='bool'>
                  1
            </Parameter>
            <Parameter id='colPrefix' text='prefix column name' type='string'>
            </Parameter>
      </Parameters>
</Generic>
```

3.49. R_Multiplot

The ‘R_Multiplot’ box is a R box which is completely similar to a Generic box except for the start and end tag, refer to 3.28 for more information.

Example:

```
<R idx='1' x='0' y='0' id='R_Multiplot'
  longName='Multi Plot'
 crypted='1'
  le='1536960584600'
  color='#e1fff0'
  keywords='multiplot chart plot visualization'
  image='/R_multiplot.svg'
  author='Daniel Soto Zeevaert'
  revision='0.06'
  tags='365_R Visualization'
  pdfDestination='5_10_12_multiplot__r_action_'>
      <Description>
```

```
             Use this operator to plot various vector comparison.&lt;br&gt;
             The data must be structured by COLUMNS in which the following order
             is respected:&lt;br&gt;
             ROWNAMES (X axis) | data | data |...| key&lt;br&gt;
             &lt;br&gt;
             To do so, typically use an AGGREGATE, TRANSPOSE, and ADD KEY, in this
             order, and the data will be ready.
      </Description>
             <Parameters>
             <Parameter id='mytable' text='whole table' type='allcolumns'>
             </Parameter>
             <Parameter id='idxMain' text='Chart Title' type='string'>
             </Parameter>
             <Parameter id='idxMin' text='Min Axis' type='double'>
                    1
             </Parameter>
             <Parameter id='idxMax' text='Max Axis' type='double'>
                    2000
             </Parameter>
             <Parameter id='idxMargin' text='Margin (for X labels)'
             type='double'>
                    15
             </Parameter>
             <Parameter id='idxDir' text='PNG Directory' type='onedir'>
                    <data></data>
             </Parameter>
      </Parameters>
</R>
```

3.50. writeColumnarGel

The writeColumnarGel box lets the user define custom column sets. Those sets are saved in <v> nodes, children of the root node <writeColumnarGel>.

Simple example:

```
<writeColumnarGel module='MathParser' idx='1' x='0' y='0' file=''/>
```

Complete example:

```
<writeColumnarGel keyHD='#1' module='#5' idx='#2' v='#6' x='#3' y='#4'
nRowsInBlock='#9' columnStrategy='#19' checkNameCollision='#10' createDir='#11'
split='#15' advancedNaming='#17' splitVar='#16' writeAllRows='#12' testCol='#13'
testValue='#14' deleteTimeOut='#18' file='#7' nThread='#8'>
  <v n='#20' i='#21'/>
  <v n='#20' i='#21'/>
  <v n='#20' i='#21'/>
</writeColumnarGel>
```

3.50.1. Standard Attributes

ID	Name	Value Type	Description	Default Value
1	keyHD	int	Number generated by anatella and used by anatella for caching purpose (used only for GUI)	
2	idx	int	ID of current box used for connections	

3	x	int	Number that defines the horizontal position of the box in the graph (used only for GUI)	0
4	y	int	Number that defines the vertical position of the box in the graph (used only for GUI)	0
5	module	string	Constant value : must be: ‘MathParser’	none
6	v	string	Constant value : must be: ‘2.3’	2.3
7	file	string	‘Anatella Columnar Gel FileName’	
8	nThread	int	if >0 : ‘Number of compression Threads for each Column Set’	-1
9	nRowsInBlock	int	Number of Data Rows inside one data-block	5000

3.50.2. Advanced Attributes

ID	Name	Value Type	Description	Default Value
10	checkNameCollision	0,1	Checkbox ‘check for column name collision’	1
11	createDir	0,1	Checkbox ‘Create output directory if it does not exists’	0
12	writeAllRows	1,0	Radio button in block ‘Row Selection’	1
13	testCol	string	Name of selected columns in block ‘Row Selection’	
14	testValue	string	‘is equal to’ in block ‘Row Selection’	
15	split	[0..3]	Radio button in block ‘Anatella Gel fil splitting’	0
16	splitVar	string	If ‘split’ == 2: Name of selected column for ‘Split into several Anatella Gel files based on the variable’	
17	advancedNaming	0,1	If ‘split’ == 2: Checkbox ‘Use advanced naming algorithm…’	0
18	deleteTimeOut	int	Block ‘Error Management’	1

3.50.3. Column Sets attributes

ID	Name	Value Type	Description	Default Value
19	columnStrategy	[0,1,2]	Radio button	0
20	v.n	text	In the table : ‘Columns’ value	
21	v.i	int	In the table : ‘Sets’ value	0

3.51. JenkinsListJobs

The ‘JenkinsListJobs’ box is a Generic box, refer to 3.28 for more information.

Example:

```
<Generic idx='1' x='0' y='0' id='JenkinsListJobs'
  longName='Jenkins list jobs'
  crypted='1'
  nPinIn='0'
  le='1628675097998'
  keywords='jenkins list jobs'
  image='/jenkinsList.svg'
  author='Raphaël Vander Marcken'
  revision='0.01'
  tags='700_Cloud Services'
  pdfDestination='5_24_2_template'>
      <Description></Description>
             <Parameters>
                    <Parameter id='idURL' text='Jenkins URL' type='string'>
                           localhost:8080
                    </Parameter>
                    <Parameter id='idUser' text='User Login' type='string'>
                    </Parameter>
                    <Parameter id='idPass' text='User Password (or Token)'
             type='password'>
                    </Parameter>
                    <Parameter id='idDebug' text='Debug Display?' type='combobox'>
                           <meta>
                                  &lt;i&gt;No debug&lt;/i&gt;&lt;i&gt;Basic
                                  Debugging Information
                                  printed&lt;/i&gt;&lt;i&gt;Verbose
                                  Debug&lt;/i&gt;
                           </meta>
                           <data>
                                  0
                           </data>
                    </Parameter>
                    <Parameter id='idOptional' text='Optional parameters for cURL'
             type='string'>
                    </Parameter>
                    <Parameter id='nRetry' text='Number of &quot;Connection
             Errors&quot; before Aborting...' type='double'>
                           3
                    </Parameter>
      </Parameters>
</Generic>
```

3.52. JenkinsListBuilds

The ‘JenkinsListBuilds’ box is a Generic box, refer to 3.28 for more information.

Example:

```
<Generic idx='1' x='0' y='0' id='JenkinsListBuilds'
  longName='Jenkins get job builds'
 crypted='1'
  le='1628675103675'
  keywords='jenkins get job builds'
  image='/jenkinsDownload.svg'
  author='Raphaël Vander Marcken'
  revision='0.01'
  tags='700_Cloud Services'
  pdfDestination='5_24_2_template'>
      <Description>

      </Description>
      <Parameters>
            <Parameter id='idJobName' text='Jenkins job name' type='onecolumn'>
                  <meta>0</meta>
                  <data></data>
            </Parameter>
            <Parameter id='idURL' text='Jenkins URL' type='string'>
                  localhost:8080
            </Parameter>
            <Parameter id='idUser' text='User Login' type='string'>
            </Parameter>
            <Parameter id='idPass' text='User Password (or Token)'
            type='password'>
            </Parameter>
            <Parameter id='idDebug' text='Debug Display?' type='combobox'>
                  <meta>
                        &lt;i&gt;No debug&lt;/i&gt;&lt;i&gt;Basic Debugging
                        Information printed&lt;/i&gt;&lt;i&gt;Verbose
                        Debug&lt;/i&gt;
                  </meta>
                  <data>0</data>
            </Parameter>
            <Parameter id='idOptional' text='Optional parameters for cURL'
            type='string'>
            </Parameter>
            <Parameter id='nRetry' text='Number of &quot;Connection Errors&quot;
            before Aborting...' type='double'>
                  3
            </Parameter>
      </Parameters>
</Generic>
```

3.53. Unflatten

The Unflatten box lets you define columns that are common to all output rows. Those columns are placed inside a <common> node, child of the root node <Unflatten>. Each column is placed in a <c> node, child of the <common> node. The box lets you define sets. Each set has a name and columns attached to it. To represent that, we have a <sets> node, child of the root node <Unflatten>. Each set is placed in a <set> node, child of the <sets> node. The set name is an attribute of this <set> node. Each attached columns is placed in a <c> node, child of the <set> node.

Simple example:

```
<Unflatten module='DefaultActions' idx='1' x='0' y='0'>
 <common></common>
 <sets>
 </sets>
</Unflatten>
```

Complete example:

```
<Unflatten keyHD='#1' module='#5' idx='#2' v='#6' x='#3' y='#4' addSetName='#10'
checkSort='#13' setColumnName='#8' newNames='#14'>
 <common>
      <c>#9</c>
      <c>#9</c>
</common>
```

```
 <sets>
  <set name='#11'>
       <c>#12</c>
       <c>#12</c>
  </set>
  <set name='#11'>
       <c>#12</c>
       <c>#12</c>
  </set>
 </sets>
</Unflatten>
```

3.53.1. Standard Attributes

ID	Name	Value Type	Description	Default Value
1	keyHD	int	Number generated by anatella and used by anatella for caching purpose (used only for GUI)	
2	idx	int	ID of current box used for connections	
3	x	int	Number that defines the horizontal position of the box in the graph (used only for GUI)	0
4	y	int	Number that defines the vertical position of the box in the graph (used only for GUI)	0
5	module	string	Constant value : must be: 'DefaultActions'	none
6	v	string	Constant value : must be: '2.44'	2.44
7	key	string	Name of selected column 'Key (Partitioning) column'	
8	setColumnName	string	'Name of column containing the column Set Name'	
9	common.c.text	string	Name of selected columns for 'Common columns to all output rows'	
10	addSetName	0,1	Checkbox 'add a column with the Set Name'	1
11	set.name	string	'Set Name' column when a Set is added	
12	set.c.text	string	Name of selected columns for the column 'Columns in set' when a Set is added	

3.53.2. Advanced Attributes

ID	Name	Value Type	Description	Default Value
13	checkSort	0,1	Checkbox	1
14	newNames	string	'Use this list of columns …'	

3.54. ReadXBF

Simple example:

```
<ReadXBF module='WriteReport' idx='1' x='0' y='0' fileName=''/>
```

Complete example:

```
<ReadXBF keyHD='#1' module='#5' idx='#2' x='#3' y='#4' bufferSize='#10'
encoding='#7' dbtInRam='#9' level7='#8' fileName='#6'/>
```

3.54.1. Standard Attributes

ID	Name	Value Type	Description	Default Value
1	keyHD	int	Number generated by anatella and used by anatella for caching purpose (used only for GUI)	
2	idx	int	ID of current box used for connections	
3	x	int	Number that defines the horizontal position of the box in the graph (used only for GUI)	0
4	y	int	Number that defines the vertical position of the box in the graph (used only for GUI)	0
5	module	string	Constant value : must be: 'WriteReport'	none
6	fileName	string	'FileName'	
7	encoding	[0..2]	Drop-down 'Character encoding'	0
8	level7	0,1	Checkbox 'the DBF file is a dBase level 7..'	0
9	dbtInRam	0,1	Checkbox 'load the full Memo/DBT/..'	0
10	bufferSize	int	'Minimum Input Buffer Size'	1

4. Node CONNECTORS

The <CONNECTORS> node has one child node <Connection> per connection between two boxes.

```
<CONNECTORS>
   <Connection idxSrc='1' idxDest='2'/>
   <Connection idxSrc='2' idxDest='3'/>
</CONNECTORS>
```

Complete example:

```
<CONNECTORS>
   <Connection idxSrc='#1' idxDest='#2'/>
   <Connection idxSrc='#1' idxDest='#2' idxPinOut='#3' idxPinIn='#4'/>
   <Connection idxSrc='#1' idxDest='#2' idxPinOut='#3' idxPinIn='#4'/>
</CONNECTORS>
```

ID	Name	Value Type	Description	Default Value
1	idxSrc	int	Index of source box	
2	idxDest	int	Index of destination box	

3	idxPinOut	int	Index of pin used as source Pins are 0-based indexed	0
4	idxPinIn	int	Index of pin used as destination Pins are 0-based indexed	0

5. Node TEXTANNOTATIONS

The <TEXTANNOTATIONS> node contains the set of text comments in an anatella graph. Each comment is represented by a child node <Annotation> of the node < TEXTANNOTATIONS>.

Simple example:

```
<TEXTANNOTATIONS>
    <Annotation x='-72' y='36' size='14' bold='1'>Comment1</Annotation>
    <Annotation x='24' y='180' size='16' color='#ff0000'>Comment2</Annotation>
</TEXTANNOTATIONS>
```

Complete example:

```
<TEXTANNOTATIONS>
    <Annotation x='#1' y='#2' font='#3' size='#4' bold='#5' italic='#6' underl
ine='#7' color='#8'>
        #9
    </Annotation>
</TEXTANNOTATIONS>
```

ID	Name	Value Type	Description	Default Value
1	x	int	Horizontal position in the graph	
2	y	int	Vertical position in the graph	
3	font	string		MS Shell Dlg 2
4	size	int	Police size of the comment text	14
5	bold	0,1		0
6	italic	0,1		0
7	underline	0,1		0
8	color	string	Color code	#000000
9	Annotation.text	string	The actual text of the comment	

6. Node GROUPBOXANNOTATIONS

The <GROUPBOXANNOTATIONS> node contains the set of colored groupbox in an anatella graph. Each groupbox is represented by a child node <Groupbox> of the node <GROUPBOXANNOTATIONS>.

Simple example:

```
<GROUPBOXANNOTATIONS>
    <Groupbox x='-696' y='0' w='528' h='240' color='#87cefa'/>
    <Groupbox x='-108' y='12' w='264' h='228' color='#7fffd4'/>
</GROUPBOXANNOTATIONS>
```

Complete example:

```
<GROUPBOXANNOTATIONS>
    <Groupbox x='#1' y='#2' w='#3' h='#4' color='#5'/>
    <Groupbox x='#1' y='#2' w='#3' h='#4' color='#5'/>
</GROUPBOXANNOTATIONS>
```

ID	Name	Value Type	Description	Default Value
1	x	int	Horizontal position in the graph	
2	y	int	Vertical position in the graph	
3	w	int	Width	
4	h	int	Height	
5	color	string	Color code	#afeeee

7. Node GlobalParameters

Here is an example of `GlobalParameters` node:

```
<GlobalParameters wDirLoc='1' workingDir='D:/Windows'>
  <Parameters>
    <Par name='GP_Name1' comment='my comments1'>GP_Value1</Par>
    <Par name='GP_Name2' comment='my comments2'>GP_Value2</Par>
  </Parameters>
  <ODBCConnections>
    <odbc name='OdbcName1' link='aa' hasPass='0'/>
    <odbc name='OdbcName2' link='aa' login='cc' ep='1' password='encr.pasw'/>
    <odbc name='OdbcName3' link='aa' login='cc' ep='0' password='my_pass'/>
    <odbc name='OdbcName4' type='1' cstring='bb'/>
    <odbc name='OdbcName5' type='1' ep='1' cstring='NkvinDw=='/>
  </ODBCConnections>
  <OleDBConnections>
    <ole name='OleDBName1' link='bb' hasPass='0'/>
    <ole name='OleDBName2' link='bb' login='cc' ep='1' password='encr.pasw'/>
    <ole name='OleDBName3' link='bb' login='cc' ep='0' password='my_pass'/>
  </OleDBConnections>
</GlobalParameters>
```

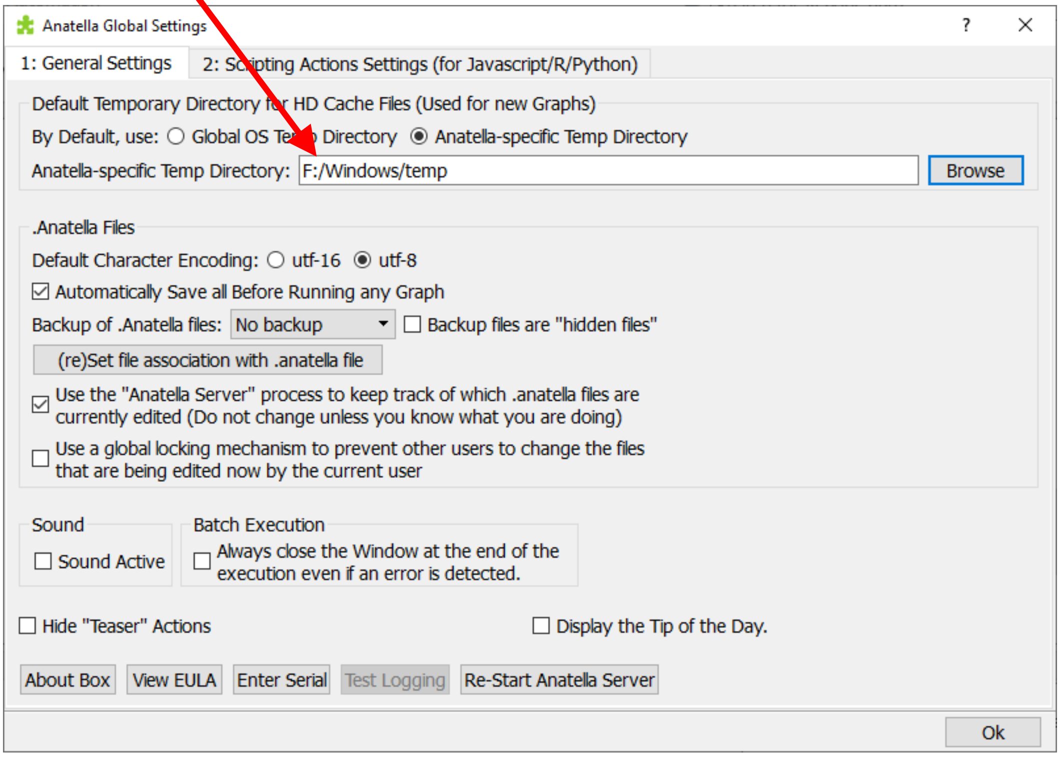
In the example above, the string `'bb'` is usually a very long (connection) string.

The attribute ‘ep’ means “encrypted password”:

- When ‘ep’ equals 1, the attributes ‘password’ and ‘cstring’ are encrypted.
- When ‘ep’ equals 0, the attributes ‘password’ and ‘cstring’ are “in clear”.

The attribute ‘wDirLoc’ defines the location of the temp/working directory.
The default value for the attribute ‘wDirLoc’ is 2.
It’s meaning is:

- when ‘wDirLoc’=0, the temp/ working directory is defined using the windows %TEMP% environmental variable
- when ‘wDirLoc’=1, the temp/ working directory is the globally defined Anatella working dir that is defined here:



Anatella Global Settings

1: General Settings | 2: Scripting Actions Settings (for Javascript/R/Python)

Default Temporary Directory for HD Cache Files (Used for new Graphs)
By Default, use: ○ Global OS Temp Directory ● Anatella-specific Temp Directory
Anatella-specific Temp Directory: F:/Windows/temp

.Anatella Files
Default Character Encoding: ○ utf-16 ● utf-8
☑ Automatically Save all Before Running any Graph
Backup of .Anatella files: No backup ☐ Backup files are "hidden files"
☑ Use the "Anatella Server" process to keep track of which .anatella files are currently edited (Do not change unless you know what you are doing)
☐ Use a global locking mechanism to prevent other users to change the files that are being edited now by the current user

Sound
☐ Sound Active

Batch Execution
☐ Always close the Window at the end of the execution even if an error is detected.

☐ Hide "Teaser" Actions
☐ Display the Tip of the Day.

- when ‘wDirLoc’=2, the temp/ working directory is defined using the ‘workingDir’ attribute.