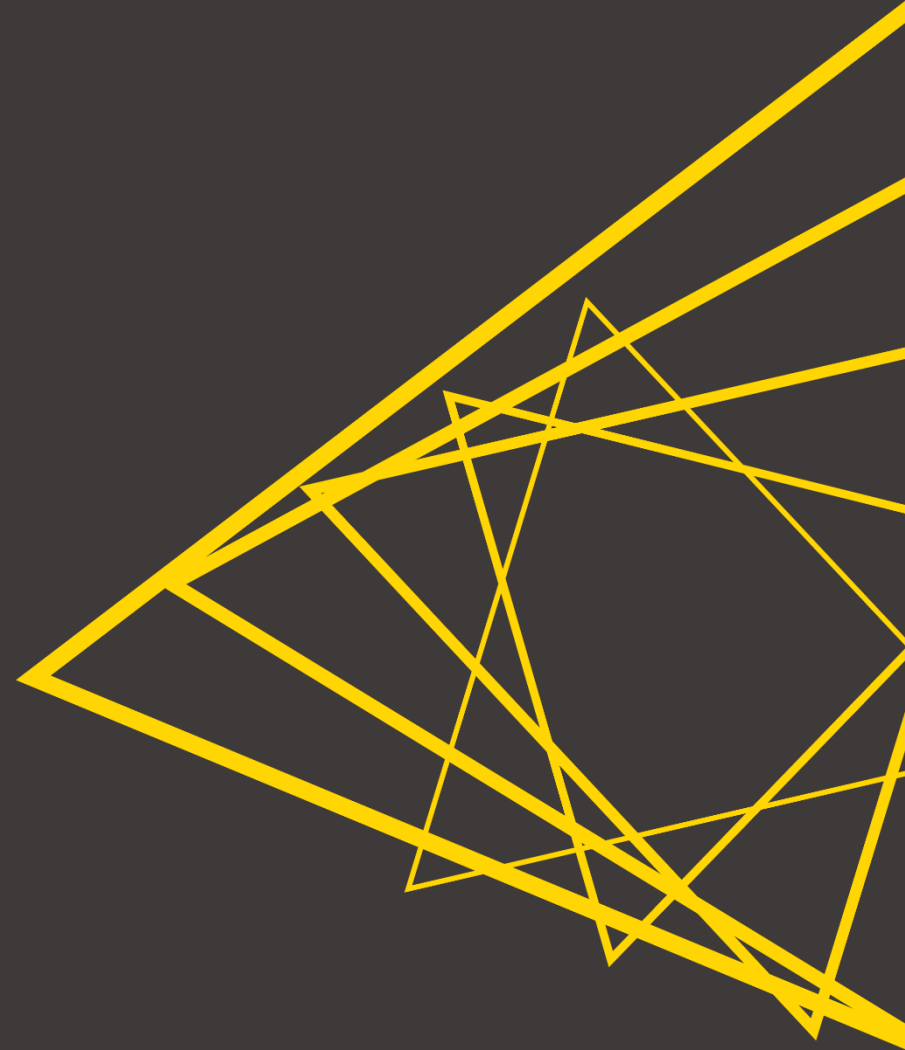


KNIME Training @ Syngenta

Monday, September 11, 2023

Dr. Alice Krebs



Content

Topic	Details
(Chemical) Data Processing	Manipulating Data – Cleaning, Data types, Conditions/Rule Engine, Regex Aggregating Data – GroupBy, Pivoting
Visualization & Output	Visualizing Data Creating Interactive & Composite Views, Components Output & File download, presentations, Smartsheet, Excel
Loops & Flow Variables, Workflow Control	What is a Flow Variable? - Creating and using Flow Variables, Path Flow Variables Repeating Workflow Parts – Loops Switches, Try-Catch, Error handling
KNIME Python Integration	Executing Python code in KNIME Conda Environment Propagation Integrating Jupyter notebooks, Code adaptations to match KNIME Python API
Customized KNIME Nodes in Python	Basics on the example template node
Chemistry Use Cases	Multiparameter Optimization (MPO), Sketcher, Similarity Search, Fingerprints, Clustering

(Chemical) Data Processing



Selected Commercial Extensions for Cheminformatics

- ▼ BioSolveIT Nodes
 - ▶ CoLibri (Chemistry Spaces)
 - ▶ IO
 - Assess Affinity with Hyde in SeeSAR
 - Compute FTrees Similarity
 - Compute FlexS Alignments
 - Compute LeadIT Docking
 - Convert Molecules with Naomi
 - FTrees Query Generator
 - Filter Molecules with Naomi
 - FlexX Docking
 - Generate 3D Coordinates
 - Generate Protomers / Tautomers with Naomi
 - Interactive BioSolveIT Table
 - Interactive SeeSAR Viewer
 - Prepare Receptor with LeadIT
 - Run ReCore Interactively
 - Search FTrees Fragment Space
 - SeeSAR Project Generator
- ▼ ChemAxon / Infocom
 - ▼ JChem
 - ▶ IO
 - ▶ Converter
 - ▶ Marvin
 - ▶ Calculator Plugins
 - ▶ JChem Base
 - ▶ JChem Cartridge
 - ▶ Standardizer
 - ▶ Structure Checker
 - ▶ Name to Structure
 - ▶ Screen
 - ▶ JKlustor
 - ▶ Reactor
 - ▶ Markush Viewer
 - ▶ Metabolizer
 - ▶ Fragmenter
 - ▶ Marvin
- ▼ Cresset
 - ▼ Forge
 - ▶ Models
 - ▶ Project
 - Forge Align
 - Activity Miner
 - FieldTemplater
 - ▼ Spark
 - Spark Fragment Selector
 - Generate Spark Database
 - Spark Database Search
 - ▼ XedTools
 - XedMin
 - XedeX
 - Torch/Forge Molecule Viewer
- ▼ MOE
 - ▶ Input
 - ▶ Output
 - ▶ Convert
 - ▶ Transform
 - ▶ Process
 - ▶ Calculate
 - ▶ QuaSAR
 - ▶ Fingerprints
 - ▶ Simulations
 - ▶ Bioinformatics
 - ▶ Fragment Based Design
 - ▶ CombiChem
 - ▶ Miscellaneous
 - ▶ Pharmacophore
 - ▶ Materials
- ▼ Schrödinger
 - ▶ Readers/Writers
 - ▶ Converters
 - ▶ Ligand Preparation
 - ▶ Property Generation
 - ▶ Cheminformatics
 - ▶ Pharmacophore Modeling
 - ▶ Protein Structure Prediction
 - ▶ Docking and Scoring
 - ▶ Molecular Mechanics
 - ▶ Molecular Dynamics
 - ▶ Quantum Mechanics
 - ▶ Workflows
 - ▶ Filtering
 - ▶ Reporting
 - ▶ Scripting
 - ▶ Tools

Selected Open Source Extensions for Cheminformatics

RDKit

- Converters
- Modifiers
- Calculators
 - RDKit Descriptor Calculation
 - RDKit Calculate Charges
- Geometry
 - RDKit Generate Coords
 - RDKit Optimize Geometry
 - RDKit Add Conformers
 - RDKit Open 3D Alignment
 - RDKit RMSD Filter
- Fingerprints
 - RDKit Fingerprint
 - RDKit Count-Based Fingerprint
 - RDKit Fingerprint Reader
 - RDKit Fingerprint Writer
 - RDKit Diversity Picker
- Fragments
- Searching
- Reactions
- Viewing
 - RDKit Interactive Table
 - RDKit SMILES Headers
 - RDKit Molecule Highlighting
- Experimental

CDK

- 3D
 - 3D Coordinates
 - 3D D-Moments
 - 3D D-Similarity
 - 3D RMSD
 - 3D Viewer
 - 3D WHIM
- AMBIT
- I/O
 - 2D Coordinates
 - Atom Signatures
 - ChemSpider
 - Connectivity
 - Depiction
 - Element Filter
 - Fingerprint Similarity
 - Fingerprints
 - Hydrogen Manipulator
 - Lipinski's Rule-of-Five
 - Mass Calculator
 - Molecular Properties
 - OPSIN
 - SMARTS Query
 - Structure Sketcher
 - Substructure Search
 - Sugar Remover
 - Sum Formula
 - Symmetry
 - XLogP

Erlwood Nodes

- IO
- Structure Data Format Converters
- Structure Similarity
 - Fingerprint Similarity
- Structure Properties
 - Plane of Best Fit Calculator
- Virtual Screening
 - Virtual Screening Metrics
- Evaluation and Ranking
 - Desirability Ranking
 - Pareto Ranking
- SAR Analysis
 - Automated Matched Pairs
 - Free-Wilson Matched Pairs
- Viewers
 - 2D/3D Scatterplot
 - Activity Cliffs Viewer
 - Similarity Viewer
- Testing

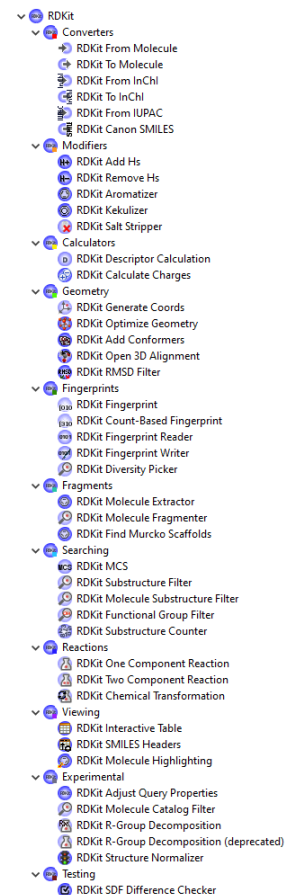
Vernalis

- Databases
- European PubMed Central
 - European PubMed Central Advanced Search
- Fingerprints
- Flow Control
- IO
- Matched Molecular Pairs (MMPs)
 - Filtering
 - Fragmentation
 - Pair Generation
 - Rendering
 - Transforms
 - MMP Calculate Maximum Cuts (RDKit)
 - MMP Fragmentation Type Loop Start
- Uniquify IDs
- Principal Moments of Inertia (PMIs)
- RCSB PDB Tools
 - PDB Connector
 - PDB Connector (XML Query)
 - PDB Connector Combine XML Queries
 - PDB Connector Custom Report
 - PDB Connector Query Only
 - PDB Connector Query Only (XML Query)
 - PDB Connector XML Query Builder
 - PDB Describe Heterogens
 - PDB Downloader (Source)
 - PDB SMILES Query
 - PDB Downloader
- Local PDB Tools
- Sequence Tools
- Speedy SMILES
- Testing
- Miscellaneous

What is RDKit?

- Open source cheminfo library in C++
- Wrappers for KNIME maintained by the open source community
- Useful for:
 - Descriptor calculation
 - Cleaning structures
 - InChI conversion
 - Canonical SMILES
 - Fingerprints
 - Scaffolds/substructures
 - Reaction simulation
 - and more...

<http://www.rdkit.org>



Overview of Types in KNIME

- Basic KNIME data types
 - String, integer, double
- KNIME core chemistry data types:
 - SMILES, SMARTS, SDF, Mol, Mol2
 - Structures in these formats can be rendered in KNIME tables
- Change the rendering by clicking on the column header
 - Available renderers depend on installed extensions

Read table - 5:3 - Table Reader

File Edit Hilite Navigation View

Table "default" - Rows: 121 Spec - Columns: 5 Properties Flow Variables

Row ID	Mol Molecules_Mol	SDF Molecules_SDF	SMI SMILES	ChEMBLID	stdInChIKey
Row0				CHEMBL461	QIAF
Row1				CHEMBL55523	HSRXSKHRSXRCFC-WDSKDSINSA-N
Row2				CHEMBL299...	AJHCSUXXECOXOY-NSHDSACASA-N

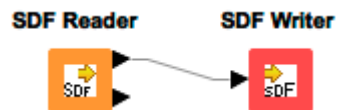
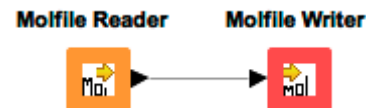
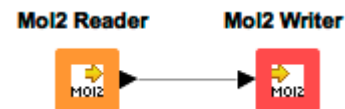
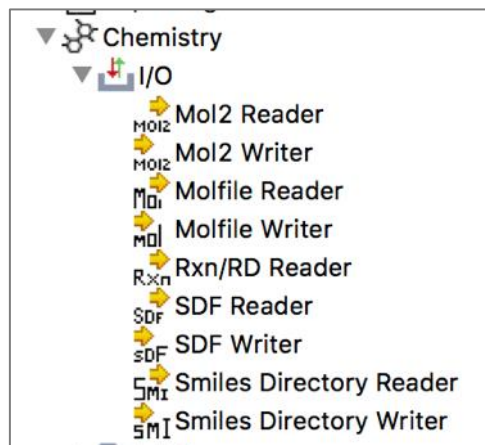
Available Renderers >

- RDKit 2D depiction
- String
- Canvas 2D
- Marvin
- Multi-line String

Nodes for Reading and Writing files

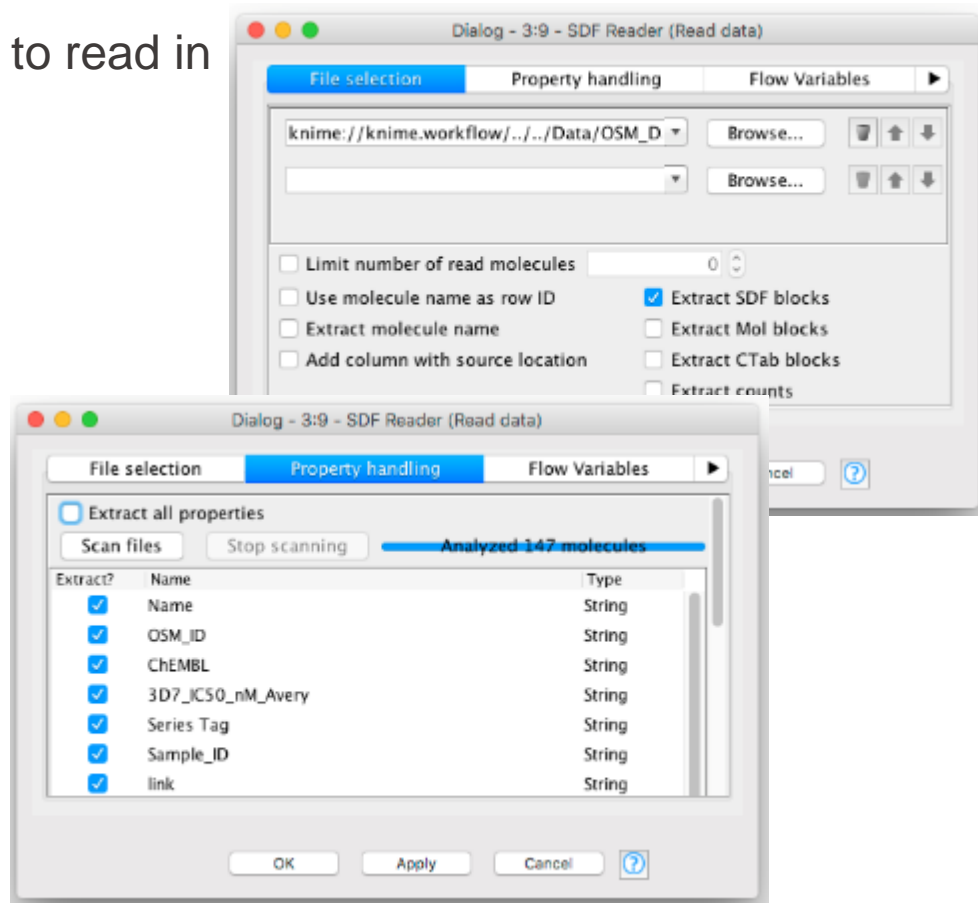
Reader and writers provided for:

- SDF, SMILES, Mol, Mol2



Reading SD files

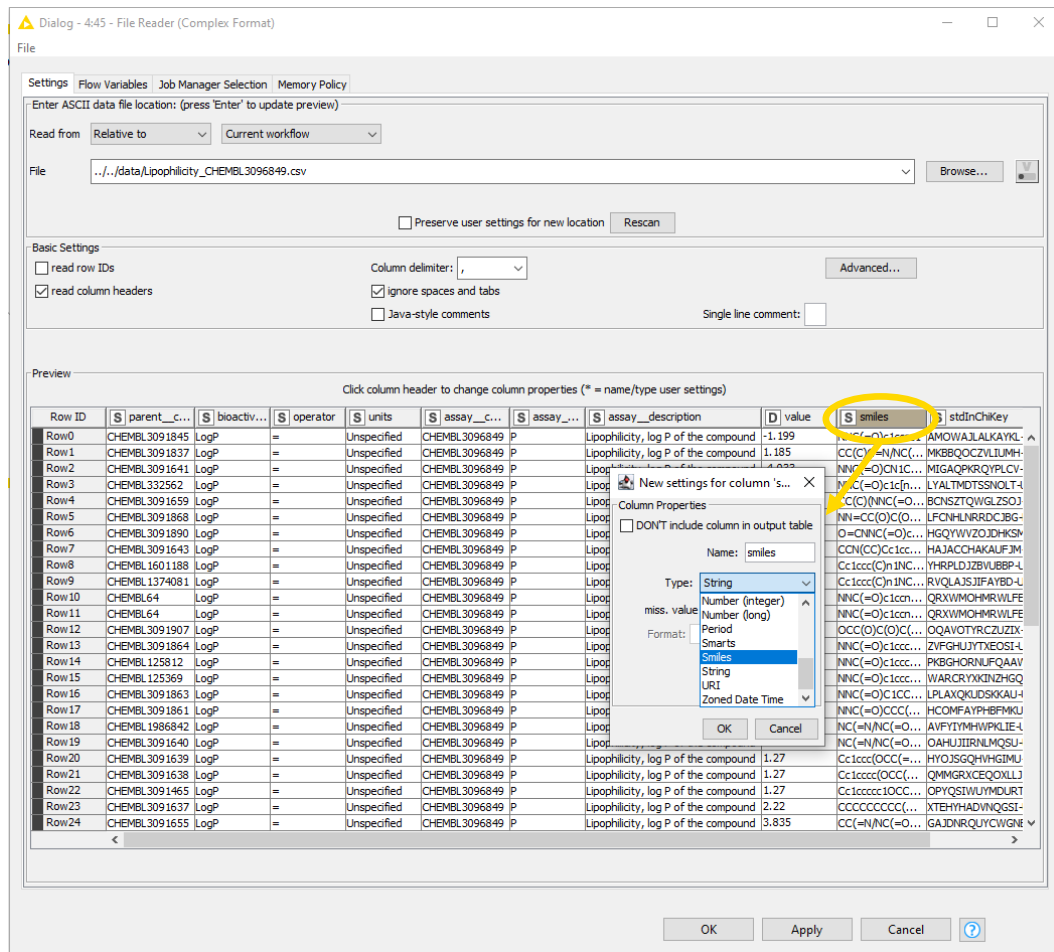
- Configure the property handling tab to read in columns with “other” data



File Reader (Complex Format)

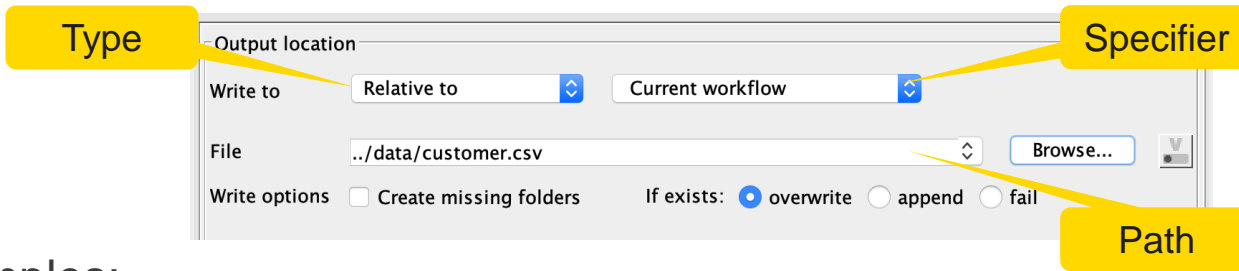
- Allows to directly define data type upon import
- Click on column header to set type

File Reader
(Complex Format)



Common Settings: File Path

- A path consists of three parts:
 - **Type**: Specifies the file system type e.g. local, relative, mountpoint, custome_url or connected.
 - **Specifier**: Optional string with additional file system specific information e.g. relative to which location (knime.workflow)
 - **Path**: Specifies the location within the file system



- Examples:
 - (LOCAL, , C:\Users\username\Desktop)
 - (RELATIVE, knime.workflow, file1.csv)
 - (MOUNTPOINT, MOUNTPOINT_NAME, /path/to/file1.csv)
 - (CONNECTED, amazon-s3:eu-west-1, /mybucket/file1.csv)


Common Settings: 4 Default File Systems

■ Local File System


Input location


Read from:

Mode: ☒ File ☐ Files in folder

File: 



■ Relative to ...


Read from: 

File: 


Current mountpoint
Current workflow data area
Current workflow


■ Mountpoint

Read from:  

File: 

■ Custom URL

Read from: 

URL: 

Workflow Data Area

- Workflows are folders in the workspace folder
- Data area is a folder inside the workflow
- Easy way to share and store data with a workflow

« 2023_06_..._Tr... > Chemistry Basics Visualization			
Name	Date modified	Type	Size
.artifacts	09/06/2023 16:41	File folder	
Color Manager (#34)	09/06/2023 16:41	File folder	
Column Filter (#6)	09/06/2023 16:34	File folder	
Column Filter (#28)	09/06/2023 16:41	File folder	
Column Rename (#32)	09/06/2023 16:41	File folder	
Concatenate (#36)	09/06/2023 16:41	File folder	
data	09/06/2023 16:28	File folder	
Duplicate Row Filter (#37)	09/06/2023 16:41	File folder	
Excel Reader (#44)	01/07/2021 10:57	File folder	
File Reader _Complex Format_ (#53)	09/06/2023 16:34	File folder	
Molecule Type Cast (#19)	09/06/2023 16:41	File folder	
RDKit Canon SMILES (#24)	09/06/2023 16:41	File folder	
RDKit Descriptor Calculation (#5)	09/06/2023 16:41	File folder	
RDKit From Molecule (#41)	09/06/2023 16:41	File folder	
Renderer to Image (#42)	09/06/2023 16:41	File folder	
SDF Reader (#3)	01/07/2021 10:57	File folder	
SDF Writer (#7)	09/06/2023 16:34	File folder	
Table Reader (#47)	09/06/2023 16:34	File folder	
Table Writer (#54)	09/06/2023 16:41	File folder	
Visualize mo (#40)	09/06/2023 16:41	File folder	
.savedWithData	09/06/2023 16:41	SAVEDWITHDATA ...	1 KB
workflow.knime	09/06/2023 16:41	KNIME File	24 KB
workflow.svg	09/06/2023 16:41	Microsoft Edge HT...	131 KB
workflowset.meta	30/06/2021 14:44	META File	2 KB

Excel Reader (XLS)

- Reads .xls and .xlsx file from Microsoft Excel
- Supports reading from multiple sheets

Excel Reader



Read Excel
Sheet Names



Excel Reader

Excel Reader



File system

File path

Sheet specific settings

Preview

Dialog - 0:1 - Excel Reader

File

Settings Transformation Advanced Settings Flow Variables Memory Policy

Input location

Read from: Relative to Current workflow

Mode: ☒ File ☐ Files in folder

File: .././data/Product Data2.xls Browse...

Sheet selection

☒ Select first sheet with data (Product Data.xls_defa...)

☐ Select sheet with name Product Data.xls_defa...

☐ Select sheet at index 0 (Sheet indexes start with 0.)

Column header

☒ Table contains column names in row number 1 (Row numbers start with 1. See "File Content" tab to identify row numbers.)

Row ID

☒ Generate row IDs ☐ Table contains row IDs in column A

Sheet area

☒ Read entire data of the sheet ☐ Read only data in columns from A to and rows from 1 to . (See "File Content" tab to identify columns and rows.)

Preview File Content

Preview with current settings

The suggested column types are based on the first 50 rows only. See "Advanced Settings" tab.

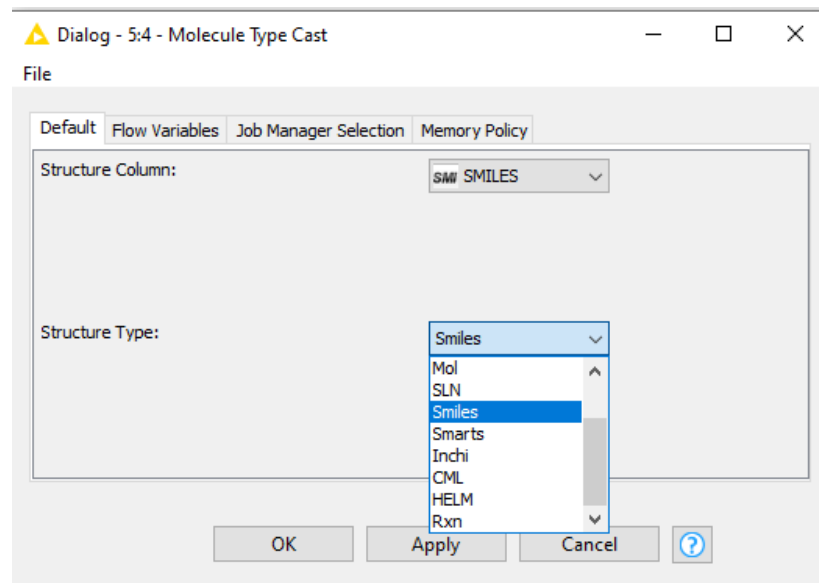
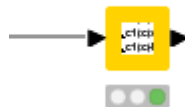
Row ID	Custom...	Products
Row0	11000	Private Investment
Row1	11001	Private Investment
Row2	11002	Private Investment
Row3	11003	Private Investment
Row4	11004	Private Investment

OK Apply Cancel ?

Molecule Type Cast

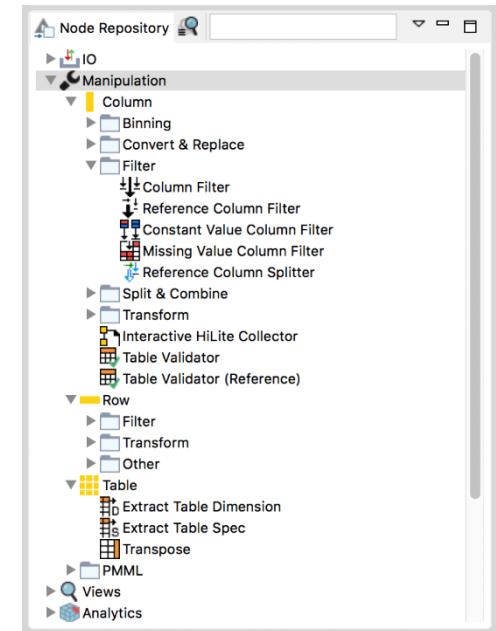
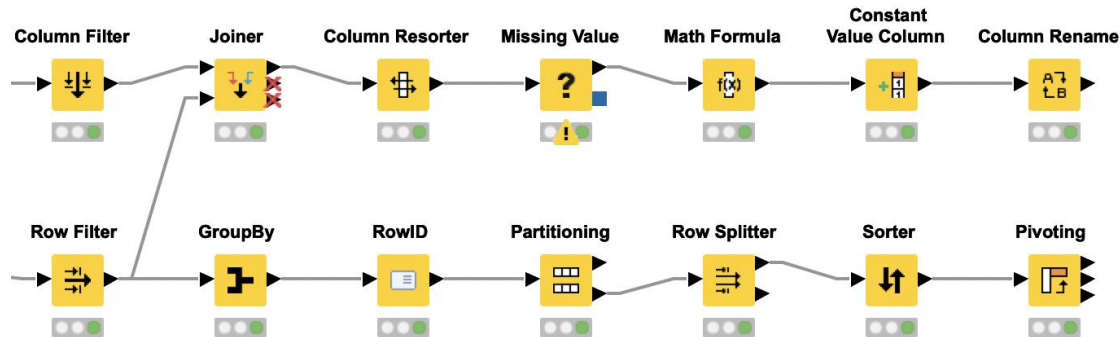
- Chemical data can be read in as string with all other Reader nodes
- Molecule Type Cast node converts of a string column into the chosen molecule type
- Variety of molecule types available:
 - e. g. Mol2, PDB, SDF, CML, HELM, SLN, Smiles, Smarts, or Rxn
- No sanity checking

Molecule Type Cast



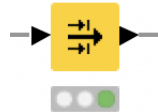
Data Manipulation Nodes

- Yellow color with a variety of input and output ports
- Apply a transformation to input data
- Many, many nodes!



Row Filter

Row Filter



Dialog - 7:28 - Row Filter (Pf3D7_ps_red < 150)

File

Filter Criteria | Flow Variables | Job Manager Selection | Memory Policy

Column value matching

Column to test: **D Pf3D7_ps_red**

☐ filter based on collection elements

Matching criteria

☐ use pattern matching

☐ case sensitive match ☐ contains wild cards

☐ regular expression

☒ use range checking

lower bound: **0.0**

upper bound: **150.0**

☐ only missing values match

☒ Include rows by attribute value

☐ Exclude rows by attribute value

☐ Include rows by number

☐ Exclude rows by number

☐ Include rows by row ID

☐ Exclude rows by row ID

OK Apply Cancel ?

- Row
 - Filter
 - Duplicate Row Filter
 - Filter Apply
 - Filter Apply Row Splitter
 - Filter Definition Merger
 - HiLite Row Splitter
 - Nominal Value Row Filter
 - Nominal Value Row Splitter
 - Numeric Row Splitter
 - Reference Row Filter
 - Reference Row Splitter
 - Row Filter
 - Row Splitter
 - Rule-based Row Filter
 - Rule-based Row Filter (Dictionary)
 - Rule-based Row Splitter
 - Rule-based Row Splitter (Dictionary)

Data Aggregation (Pivoting)

Type	Name	Safety
NSAIDs	paracetamol	irritant
NSAIDs	aspirin	irritant
NSAIDs	ibuprofen	health hazard
NSAID	diclofenac	acute toxic
PPIs	omeprazole	irritant
PPIs	pantoprazole	irritant
SSRIs	fluoxetine	acute toxic
SSRIs	paroxetine	health hazard
SSRIs	citalopram	health hazard
SSRIs	sertraline	health hazard



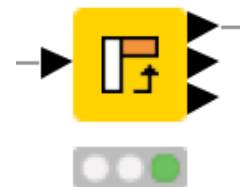
Type	Acute toxic	Health hazard	Irritant
NSAIDs	1	1	2
PPIs	?	?	2
SSRIs	1	3	?

Pivoting Node: **Group** - **Pivot** - **Aggregate**

Pivoting with Two Aggregation Methods

Row ID	S Type	S Safety	S Name
Row0	NSAID	irritant	paracetamol
Row1	NSAID	irritant	aspirin
Row2	NSAID	health hazard	ibuprofen
Row3	NSAID	acute toxic	diclofenac
Row4	PPIs	irritant	omeprazole
Row5	PPIs	irritant	pantoprazole
Row7	SSRIs	acute toxic	fluoxetine
Row8	SSRIs	health hazard	paroxetine
Row9	SSRIs	health hazard	citalopram
Row10	SSRIs	health hazard	sertraline

Pivoting



Row ID	S Type	I acute toxic+Count(Name)	I health hazard+Count(Name)	I irritant+Count(Name)
Row0	NSAID	1	1	2
Row1	PPIs	?	?	2
Row2	SSRIs	1	3	?

Pivoting Node: **Group** - **Pivot** - **Aggregate**

Pivoting

I	mol.	D	standard_value	S	standard_relation	S	standard_units	S	standard_type	D	pchemb.	S	target_chemb.
152260			6.1	=		nM	Ki			8.21		CHEMBL214	
152137			7.8	=		nM	Ki			8.11		CHEMBL214	
152134			120	=		nM	Ki			6.92		CHEMBL214	
152080			210	=		nM	Ki			6.68		CHEMBL214	
52338			47	=		nM	Ki			7.33		CHEMBL214	
40962			2,759.7	=		nM	Ki			5.56		CHEMBL214	
40961			10,000	>		nM	Ki			?		CHEMBL214	
40178			10,000	>		nM	Ki			?		CHEMBL214	
40145			10,000	>		nM	Ki			?		CHEMBL214	
40144			3,967.7	=		nM	Ki			5.4		CHEMBL214	
40143			405.6	=		nM	Ki			6.39		CHEMBL214	
40142			10,000	>		nM	Ki			?		CHEMBL214	
40141			10,000	>		nM	Ki			?		CHEMBL214	
40138			10,000	>		nM	Ki			?		CHEMBL214	
40060			990.7	=		nM	Ki			6		CHEMBL214	
37553			31.1	=		nM	Ki			7.51		CHEMBL214	
37476			5.05	=		nM	Ki			8.3		CHEMBL214	
37323			40.7	=		nM	Ki			7.39		CHEMBL214	
36988			14.4	=		nM	Ki			7.84		CHEMBL214	
36732			170	=		nM	Ki			6.77		CHEMBL214	
36730			10,000	>		nM	Ki			?		CHEMBL214	
36357			355	=		nM	Ki			?		CHEMBL214	
36324			1.8	=		nM	Ki			?		CHEMBL214	
36287			139	=		nM	Ki			?		CHEMBL214	
36144			319	=		nM	Ki			?		CHEMBL214	
36144			48.8	=		nM	Ki			?		CHEMBL214	
35706			778	=		nM	Ki			?		CHEMBL214	
35706			142	=		nM	Ki			?		CHEMBL214	
35672			554	=		nM	Ki			?		CHEMBL214	
35623			10,000	>		nM	Ki			?		CHEMBL214	
35324			1.3	=		nM	Ki			?		CHEMBL214	
33781			80	=		nM	Ki			?		CHEMBL214	
33781			80	=		nM	Ki			?		CHEMBL214	
33681			168	=		nM	Ki			?		CHEMBL214	

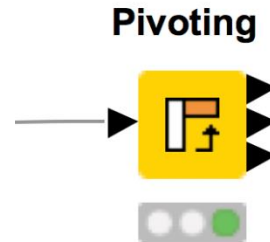
Row ID	I	molregno	S	CHEMBL214+pref_name	S	CHEMBL214+standard_type	D	CHEMBL214+standard_value	S	CHEMBL214+standard_relation	S	CHEMBL214+standard_units	D	CHEMBL214+pchembl_value	S	CHEMBL214+assay_chemb_id
Row0		2214		Serotonin 1a (5-HT1a) receptor	Ki			6,620.9	=	nM		5.18				CHEMBL616124
Row1		2246		Serotonin 1a (5-HT1a) receptor	Ki			6.38	=	nM		8.2				CHEMBL2026267
Row2		2261		Serotonin 1a (5-HT1a) receptor	Ki			6	=	nM		8.22				CHEMBL870197
Row3		3587		Serotonin 1a (5-HT1a) receptor	Ki			12	=	nM		7.92				CHEMBL1030618
Row4		3599		Serotonin 1a (5-HT1a) receptor	Ki			296	=	nM		6.53				CHEMBL615779
Row5		3854		Serotonin 1a (5-HT1a) receptor	Ki			3,600	=	nM		5.44				CHEMBL3366152
Row6		3859		Serotonin 1a (5-HT1a) receptor	Ki			0.17	=	nM		9.77				CHEMBL1034993
Row7		4149		Serotonin 1a (5-HT1a) receptor	Ki			0.6	=	nM		9.22				CHEMBL1046676
Row8		4181		Serotonin 1a (5-HT1a) receptor	Ki			17.2	=	nM		7.76				CHEMBL3820056
Row9		4634		Serotonin 1a (5-HT1a) receptor	Ki			2	=	nM		8.7				CHEMBL839708
Row10		6291		Serotonin 1a (5-HT1a) receptor	Ki			6.7	=	nM		8.17				CHEMBL884575
Row11		6446		Serotonin 1a (5-HT1a) receptor	Ki			24	=	nM		7.62				CHEMBL616370
Row12		7065		Serotonin 1a (5-HT1a) receptor	Ki			190	=	nM		6.72				CHEMBL3226682
Row13		7714		Serotonin 1a (5-HT1a) receptor	Ki			22.4	=	nM		7.65				CHEMBL616120
Row14		8543		Serotonin 1a (5-HT1a) receptor	Ki			344	=	nM		6.46				CHEMBL615460
Row15		9128		Serotonin 1a (5-HT1a) receptor	Ki			7,943	>	nM		?				CHEMBL615975
Row16		10841		Serotonin 1a (5-HT1a) receptor	Ki			4	=	nM		8.4				CHEMBL616370
Row17		12494		Serotonin 1a (5-HT1a) receptor	Ki			10.371	=	nM		9.43				CHEMBL1035001
Row18		13073		Serotonin 1a (5-HT1a) receptor	Ki											

Pivoting Node: **Group** - **Pivot** - **Aggregate**

Pivoting

Performs pivoting on selected columns for grouping and pivoting

- Values of group columns become unique rows
- Values of the pivot columns become unique columns for each set of column combination together with each aggregation
- Many aggregation methods are provided (similar to GroupBy)



Pivoting

Groups ~ Rows

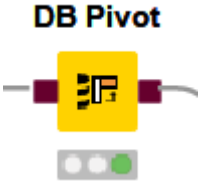
Pivots ~ Columns

Aggregation

Pivot table - 0:35 - Pivoting

Row ID	Category	Online+Sum(OrderedItems)	Onsite+Sum(OrderedItems)
Row0	Clothing	11823	7604
Row1	Electronics	10754	6624
Row2	Home	7180	5109

Data Pivoting



Settings | Description | Flow Variables | Job Manager Selection

Groups | Pivots | Manual Aggregation

Group columns

Available

Pivot columns

Available

Aggregation settings

Available columns

Select

To change multiple columns use right mouse click for context menu.

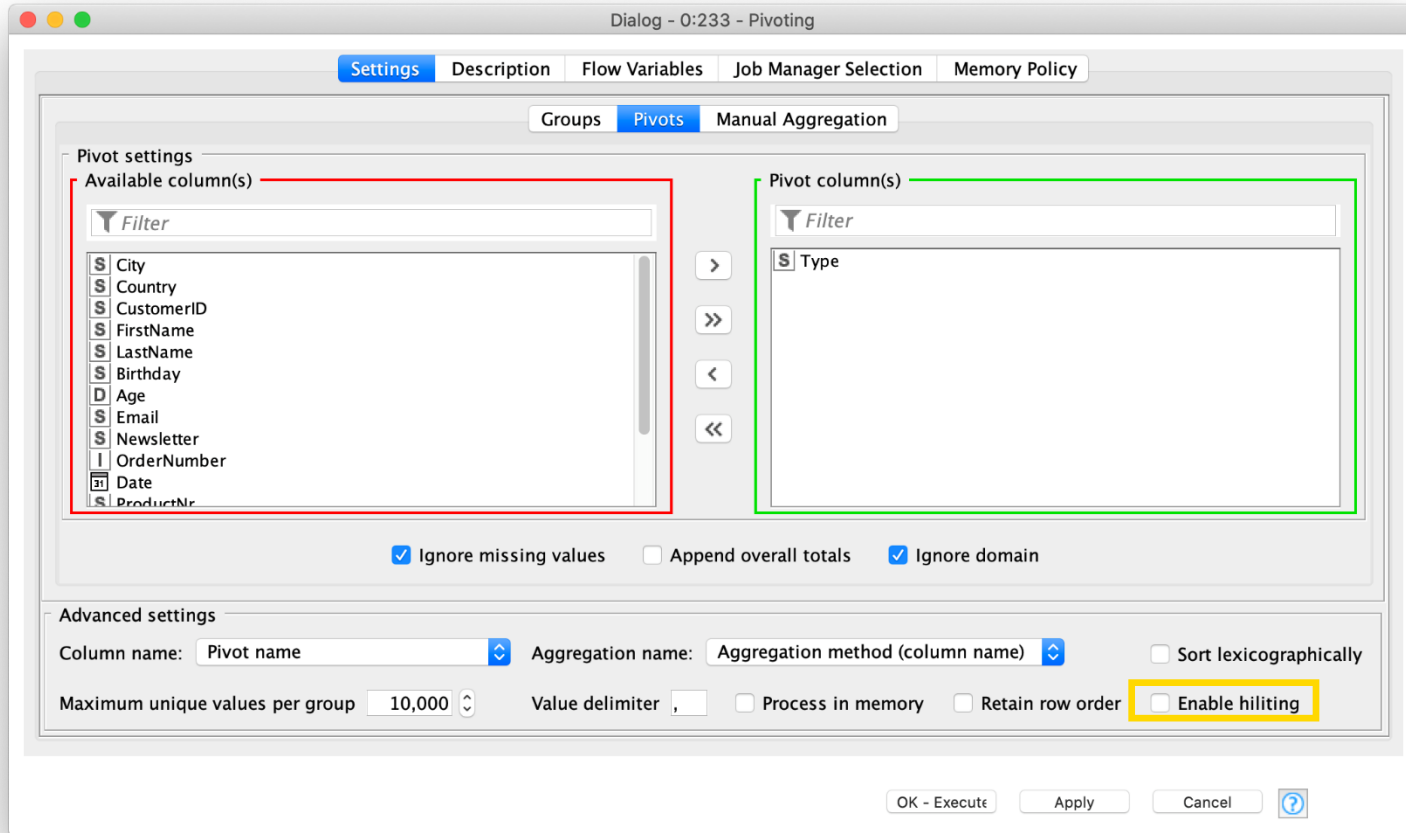
Column	Aggregation (click to change)	Missing	Parameter
S pref_name	Minimum	<input checked="" type="checkbox"/>	
D standard_value	Minimum	<input type="checkbox"/>	
D pchembl_value	Maximum	<input type="checkbox"/>	
S standard_type	Minimum	<input checked="" type="checkbox"/>	
S standard_relation	Minimum	<input checked="" type="checkbox"/>	
S published_units	Minimum	<input checked="" type="checkbox"/>	
S assay_chembl_id	Minimum	<input checked="" type="checkbox"/>	

Advanced settings

Column name: Pivot name+Aggregation name | Aggregation name: Aggregation method (column name) | ☐ Sort lexicographically

Maximum unique values per group: 10,000 | Value delimiter: , | ☐ Process in memory | ☐ Retain row order | ☐ Enable hilling

Enable Hiliting!



Data Aggregation (GroupBy)

Type	Name	Weigt
NSAID	paracetamol	151.17
NSAID	aspirin	180.16
NSAID	ibuprofen	206.29
NSAID	diclofenac	296.15
PPI	omeprazole	345.42
PPI	pantoprazole	383.38
SSRI	fluoxetine	309.33
SSRI	paroxetine	329.37
SSRI	citalopram	324.40
SSRI	sertraline	342.70



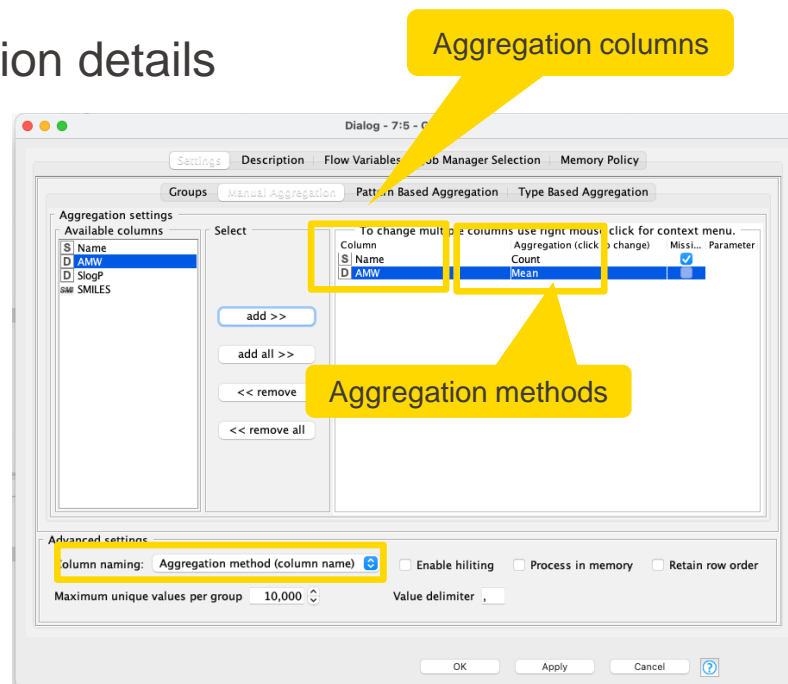
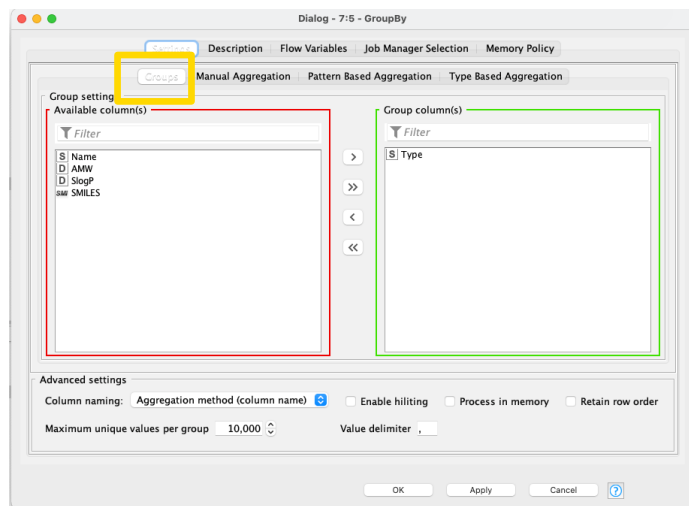
Type	Count(Name)	Mean(Weight)
NSAID	4	208.44
PPI	2	364.40
SSRI	4	326.45

Aggregated on Type (**group**) by
Count (**aggregation method**) and
Mean (**aggregation method**)

GroupBy

Aggregate to summarize data

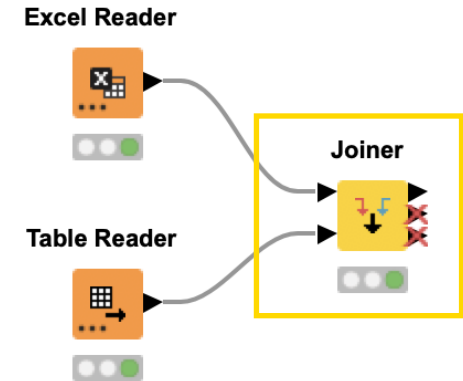
- First tab provides grouping options
- Second tab provides control over aggregation details



YouTube KNIME TV video: <https://youtu.be/bDwF-TOMtWw>

Joiner

- Combines columns from two different tables
 - Top input port: “Left” data table
 - Bottom input port: “Right” data table
- Outputs:
 - Top port: Resulting joined table
 - Middle port: Unmatched rows from the left input table (top input port)
 - Bottom port: Unmatched rows from the right input table (bottom input port)
- By default the two bottom output ports are deactivated



Joiner Configuration – Linking Rows

Values to join on.
Multiple joining columns
are allowed

Select the rows which
should be included in the
joined table

Activate this checkbox to
activate the bottom
output ports

Dialog - 3:8 - Joiner

File

Joiner Settings | Column Selection | Performance | Flow Variables | Job Manager Selection | Memory Policy

Join columns

Match ☒ all of the following ☐ any of the following

Top Input (left' table) Bottom Input (right' table)

+ -

+

Compare values in join columns by ☒ value and type ☐ string representation ☐ making integer types compatible


Include in output

☒ Matching rows

☐ Left unmatched rows

☐ Right unmatched rows

Inner join



Output options

☒ Split join result into multiple tables (top = matching rows, middle = left unmatched rows, bottom = right unmatched rows)

☐ Merge join columns

☐ Hlling enabled

OK Apply Cancel ?

Joining Columns of Data – Inner Join

Left Table

molregno	chembl_id	SMILES
22	CHEMBL1794855	CCCN(CCC)
24	CHEMBL278751	CCN(C)
15	CHEMBL103772	CCCN1CC
10	CHEMBL328107	C1CN(CCN1)

Right Table

molregno	Ki_value	Ki_relation	Ki_unit
17	76.0	=	nM
65	6.56	=	nM
35	100	>	nM
15	8	=	nM
10	95.8	=	nM



Inner Join

molregno	chembl_id	SMILES	Ki_value	Ki_relation	Ki_unit
15	CHEMBL103772	CCCN1CC	8	=	nM
10	CHEMBL328107	C1CN(CCN1)	95.8	=	nM

Joining Columns of Data – Left Outer Join

Left Table

molregno	chembl_id	SMILES
22	CHEMBL1794855	CCCN(CCC)
24	CHEMBL278751	CCN(C)
15	CHEMBL103772	CCCN1CC
10	CHEMBL328107	C1CN(CCN1)

Right Table

molregno	Ki_value	Ki_relation	Ki_unit
17	76.0	=	nM
65	6.56	=	nM
35	100	>	nM
15	8	=	nM
10	95.8	=	nM



Left Outer Join

molregno	chembl_id	SMILES	Ki_value	Ki_relation	Ki_unit
22	CHEMBL1794855	CCCN(CCC)	?	?	?
24	CHEMBL278751	CCN(C)	?	?	?
15	CHEMBL103772	CCCN1CC	8	=	nM
10	CHEMBL328107	C1CN(CCN1)	95.8	=	nM

Joining Columns of Data – Right Outer Join

Left Table

molregno	chembl_id	SMILES
22	CHEMBL1794855	CCCN(CCC)
24	CHEMBL278751	CCN(C)
15	CHEMBL103772	CCCN1CC
10	CHEMBL328107	C1CN(CCN1)

Right Table

molregno	Ki_value	Ki_relation	Ki_unit
17	76.0	=	nM
65	6.56	=	nM
35	100	>	nM
15	8	=	nM
10	95.8	=	nM

Right Outer Join

molregno	chembl_id	SMILES	Ki_value	Ki_relation	Ki_unit
17	?	?	76.0	=	nM
65	?	?	6.56	=	nM
35	?	?	100	>	nM
15	CHEMBL103772	CCCN1CC	8	=	nM
10	CHEMBL328107	C1CN(CCN1)	95.8	=	nM

Joining Columns of Data – Full Outer Join

Left Table

molregno	chembl_id	SMILES
22	CHEMBL1794855	CCCN(CCC)
24	CHEMBL278751	CCN(C)
15	CHEMBL103772	CCCN1CC
10	CHEMBL328107	C1CN(CCN1)

Right Table

molregno	Ki_value	Ki_relation	Ki_unit
17	76.0	=	nM
65	6.56	=	nM
35	100	>	nM
15	8	=	nM
10	95.8	=	nM

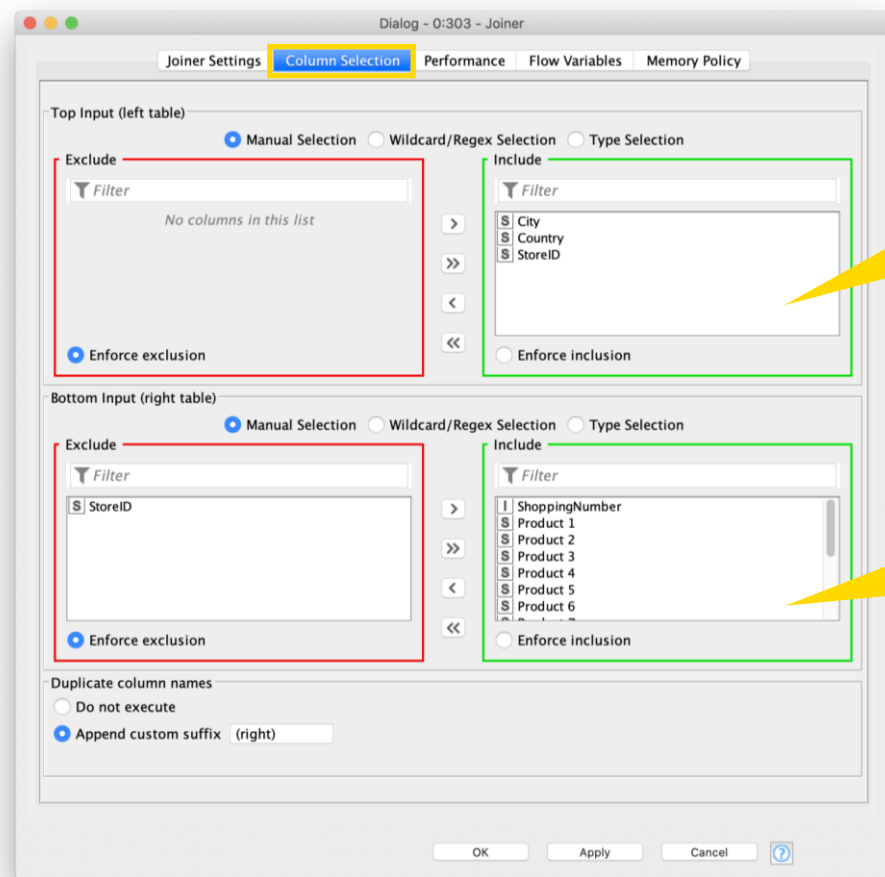
Values missing in the left table

Full Outer Join

molregno	chembl_id	SMILES	Ki_value	Ki_relation	Ki_unit
17	?	?	76.0	=	nM
65	?	?	6.56	=	nM
35	?	?	100	>	nM
15	CHEMBL103772	CCCN1CC	8	=	nM
10	CHEMBL328107	C1CN(CCN1)	95.8	=	nM
22	CHEMBL1794855	CCCN(CCC)	?	?	?
24	CHEMBL278751	CCN(C)	?	?	?

Values missing in the right table

Joiner Configuration – Column Selection

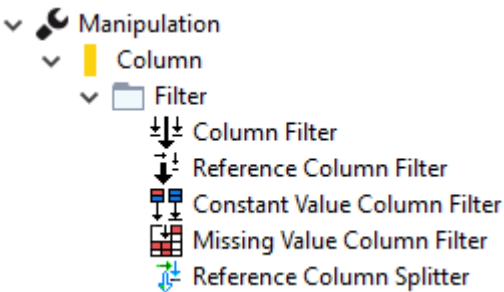
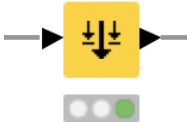


Columns from top table for joined table

Columns from lower table for joined table

Column Filter

Column Filter



Dialog - 7:25 - Column Filter (Remove)

File

Column Filter | Flow Variables | Job Manager Selection | Memory Policy

☒ Manual Selection ☐ Wildcard/Regex Selection ☐ Type Selection

Exclude

Filter

No columns in this list

☒ Enforce exclusion

Include

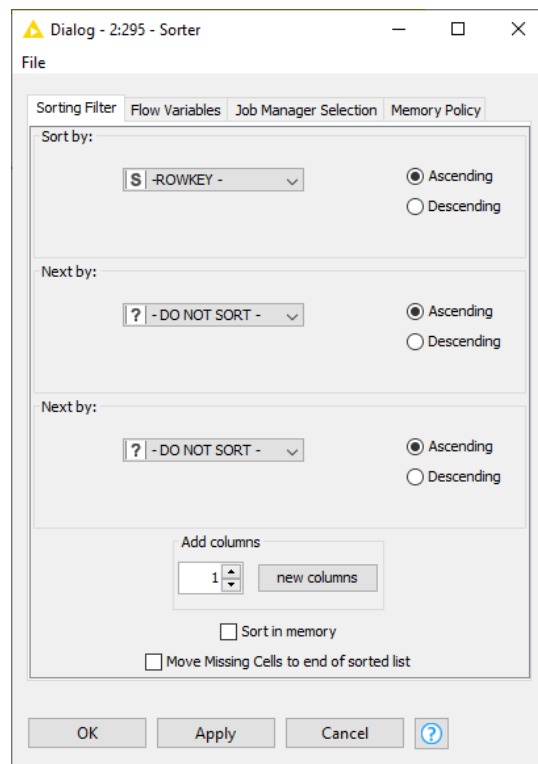
Filter

S Sample
D Pf3D7_ps_green
D Pf3D7_ps_red
S Pf3D7_ps_hit
D Pf3D7_pEC50

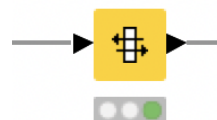
☐ Enforce inclusion

OK Apply Cancel ?

Other Useful Nodes for Row and Column Handling



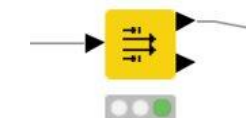
Column Resorter



Column Splitter

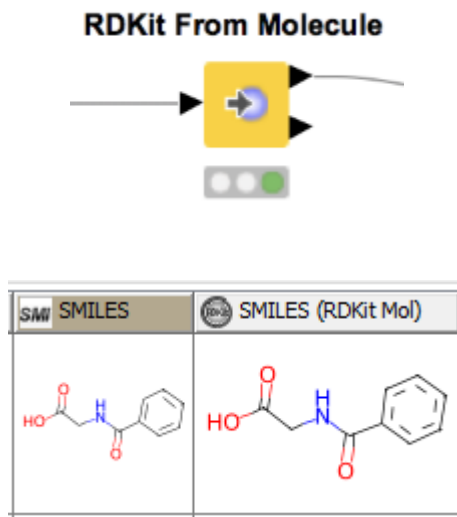


Row Splitter



RDKit From Molecule

- Generates RDKit molecule column from a molecule string representation (SMILES, SDF or SMARTS)
- Can generate 2D coordinates
- Does some “chemical sanity check”



Dialog - 0:27 - RDKit From Molecule

Options | **Advanced** | Flow Variables ▶

Molecule column: SDF Molecule ☐ Treat as query

New column name: Molecule (RDKit Mol)

☐ Remove source column

Error Handling

☒ Send error rows to second output

☐ Insert missing values

☐ Generate error information column

Error Information Column Name:

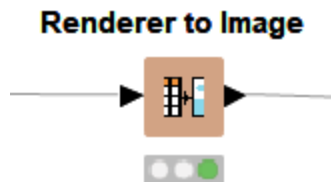
2D Coordinates

☐ Generate 2D Coordinates

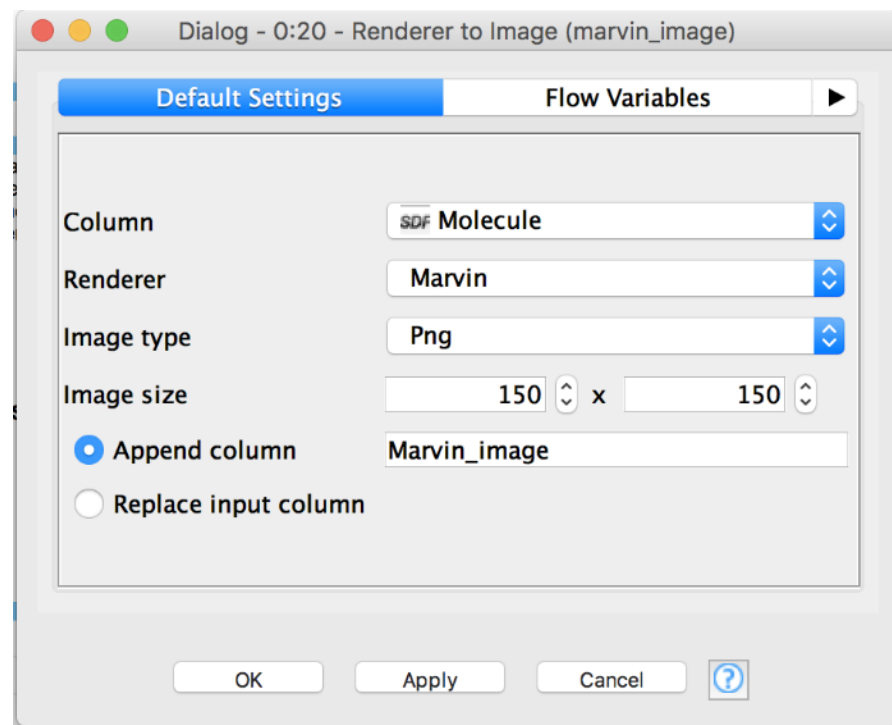
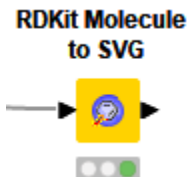
☐ Force Generation

Renderer to Image

- Needed to show rendered molecules in Interactive Views
- Convert chemical structure to an image
- Various image types

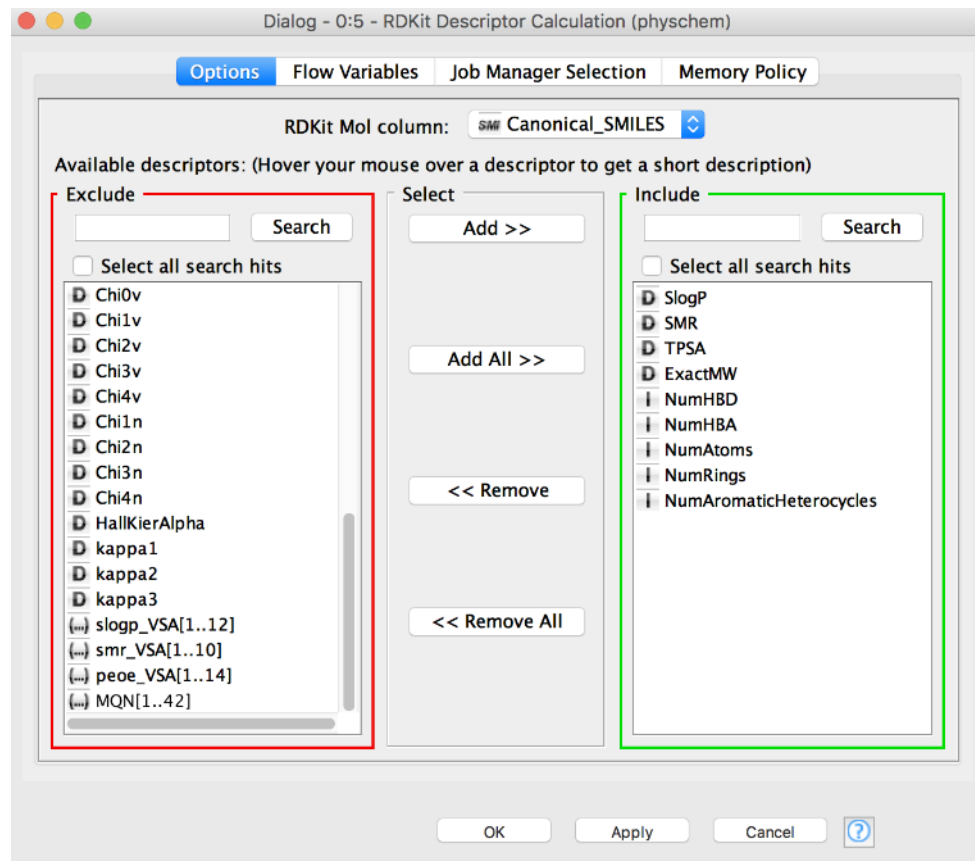
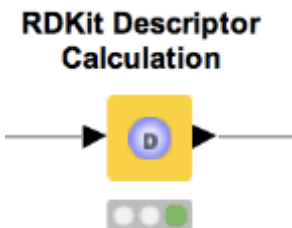


- Alternative:

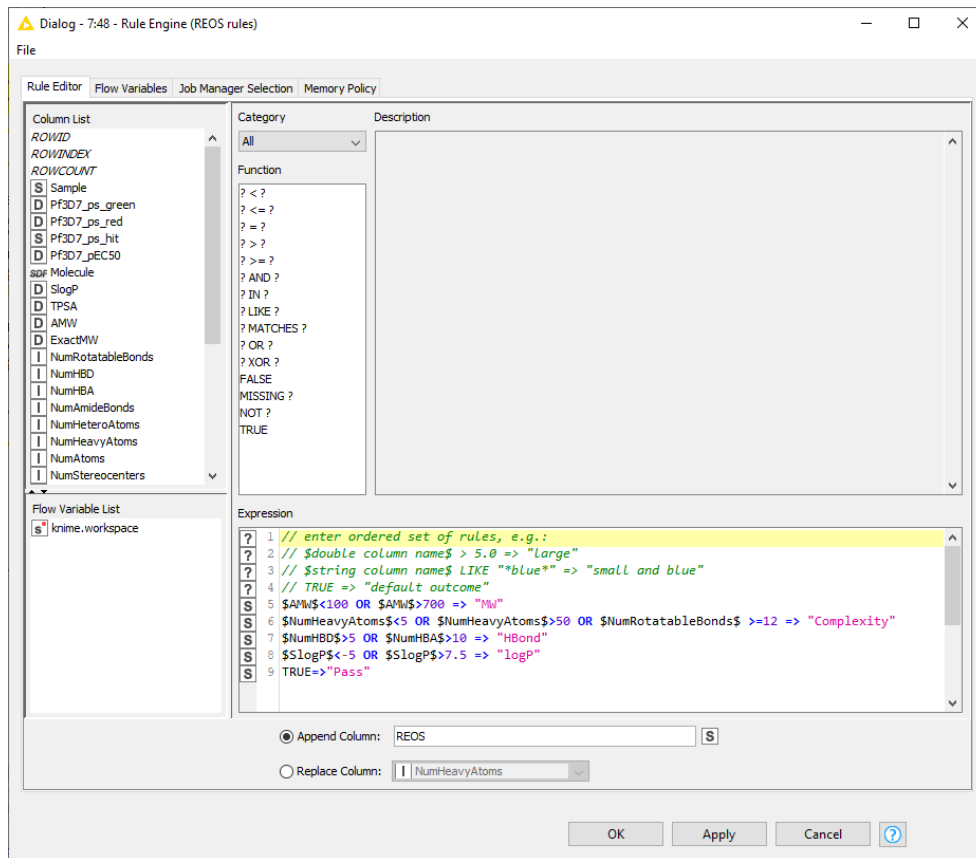
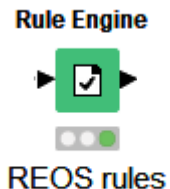


Compute Descriptors

- Not all descriptors from the Python library available in the node



Rule Engine



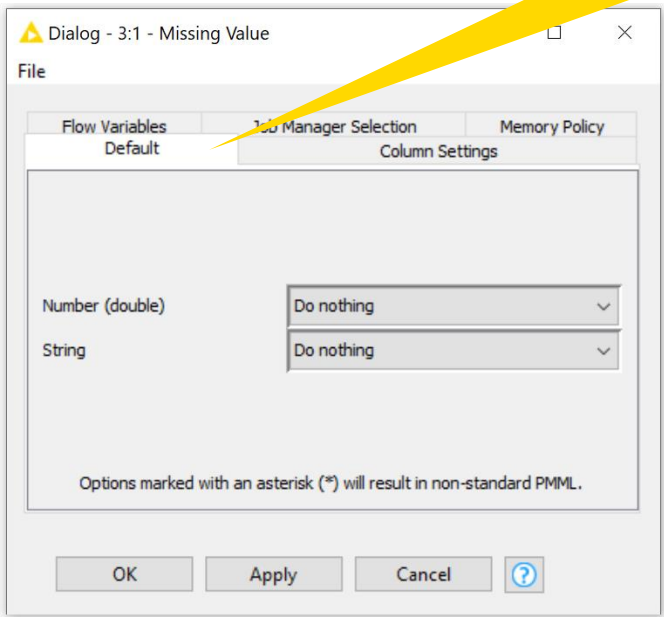
IF => THEN
ELSE

Missing Value

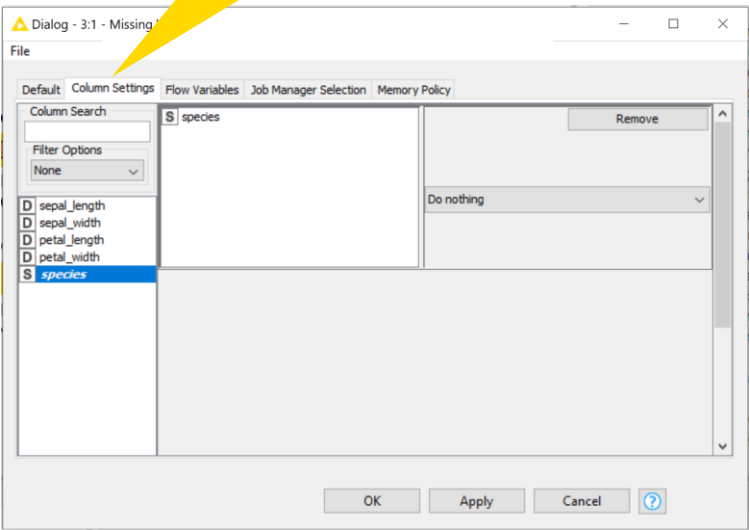
helps to handle missing values



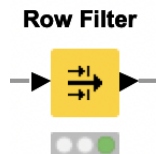
Define default values for all columns of specific type



Define values for every single column



Row Filter



Dialog - 7:28 - Row Filter (Pf3D7_ps_red < 150)

File

Filter Criteria | Flow Variables | Job Manager Selection | Memory Policy

Column value matching

Column to test: D Pf3D7_ps_red

☐ filter based on collection elements

Matching criteria

☐ use pattern matching

☐ case sensitive match ☐ contains wild cards

☐ regular expression

☒ use range checking

lower bound: 0.0

upper bound: 150.0

☐ only missing values match

☒ Include rows by attribute value

☐ Exclude rows by attribute value

☐ Include rows by number

☐ Exclude rows by number

☐ Include rows by row ID

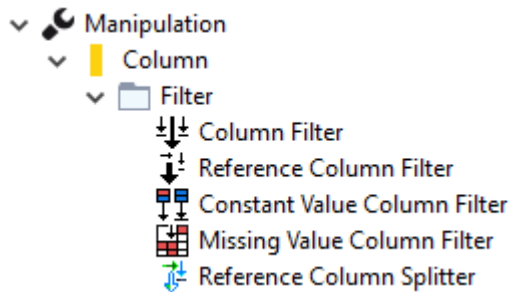
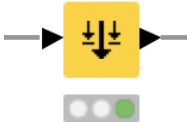
☐ Exclude rows by row ID

OK Apply Cancel ?

- Row
 - Filter
 - Duplicate Row Filter
 - Filter Apply
 - Filter Apply Row Splitter
 - Filter Definition Merger
 - HiLite Row Splitter
 - Nominal Value Row Filter
 - Nominal Value Row Splitter
 - Numeric Row Splitter
 - Reference Row Filter
 - Reference Row Splitter
 - Row Filter
 - Row Splitter
 - Rule-based Row Filter
 - Rule-based Row Filter (Dictionary)
 - Rule-based Row Splitter
 - Rule-based Row Splitter (Dictionary)

Column Filter

Column Filter



Dialog - 7:25 - Column Filter (Remove)

File

Column Filter | Flow Variables | Job Manager Selection | Memory Policy

☒ Manual Selection ☐ Wildcard/Regex Selection ☐ Type Selection

Exclude

Filter

No columns in this list

☒ Enforce exclusion

Include

Filter

S	Sample
D	Pf3D7_ps_green
D	Pf3D7_ps_red
S	Pf3D7_ps_hit
D	Pf3D7_pEC50

☐ Enforce inclusion

OK Apply Cancel ?

Other Useful Nodes for Row and Column Handling



Dialog - 2:295 - Sorter

File

Sorting Filter Flow Variables Job Manager Selection Memory Policy

Sort by:

S - ROWKEY -

Ascending

Descending

Next by:

? - DO NOT SORT -

Ascending

Descending

Next by:

? - DO NOT SORT -

Ascending

Descending

Add columns

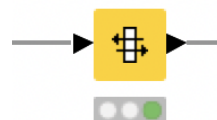
1 new columns

Sort in memory

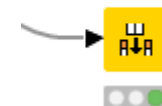
Move Missing Cells to end of sorted list

OK Apply Cancel ?

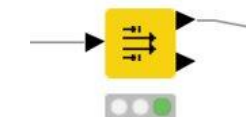
Column Resorter



Column Splitter



Row Splitter



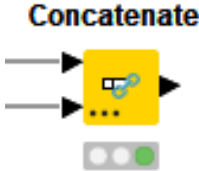
Concatenate Tables

Table A

RowID	Mol Reg No	Chembl ID	Ki value
0	35	CHEMBL15435	100.0
1	15	CHEMBL1794855	8.0

Table B

RowID	Mol Reg No	Chembl ID
0	15	CHEMBL1794855
1	10	CHEMBL278751
2	22	CHEMBL103772



union of columns

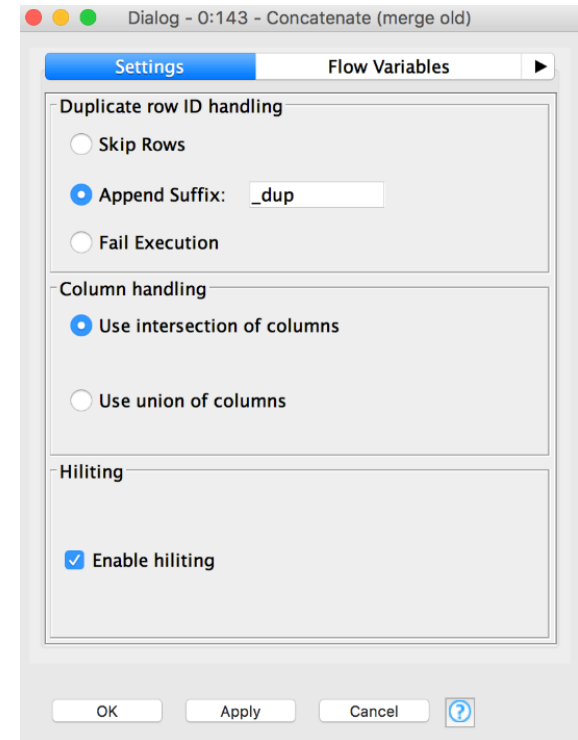
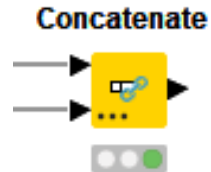
RowID	Mol Reg No	Chembl ID	Ki value
0	35	CHEMBL15435	100.0
1	15	CHEMBL1794855	8.0
0_dup	15	CHEMBL1794855	
1_dup	10	CHEMBL278751	
2	22	CHEMBL103772	

intersection of columns

Concatenate

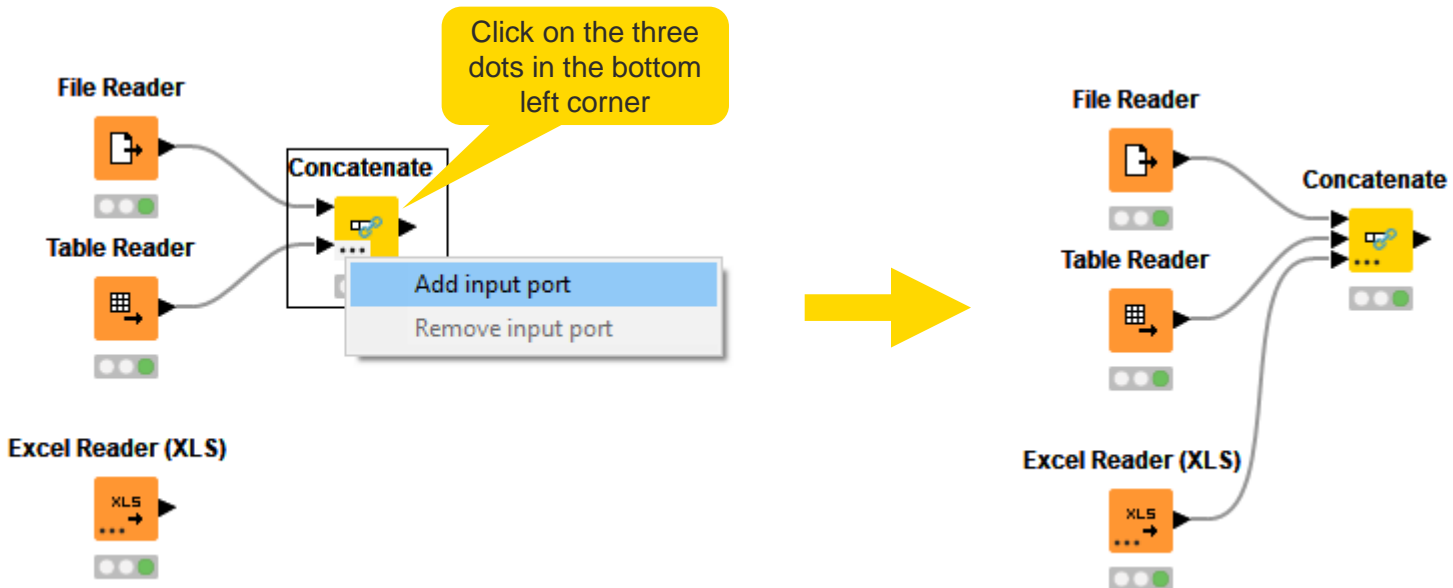
Combine rows from 2 or more tables with shared columns

- Handles duplicate row keys gracefully
- Take the union or intersection of columns



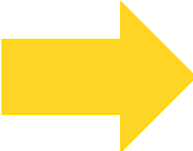
Dynamic Ports

Add and remove node ports based on your needs, e.g. in order to concatenate three or more tables

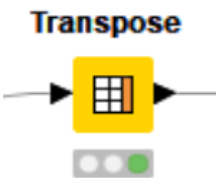


Transpose

Row ID	S column1	S column2	S column3
Row0	1		3
Row1	1		3
Row2	1		3
Row3	1		3



Row ID	S Row0	S Row1	S Row2	S Row3
column1	1	1	1	1
column2	2	2	2	2
column3	3	3	3	3



Dialog - 3:4 - Transpose

File

Options

Flow Variables

Job Manager Selection

Memory Policy

Chunk size (columns): 10

OK

Apply

Cancel

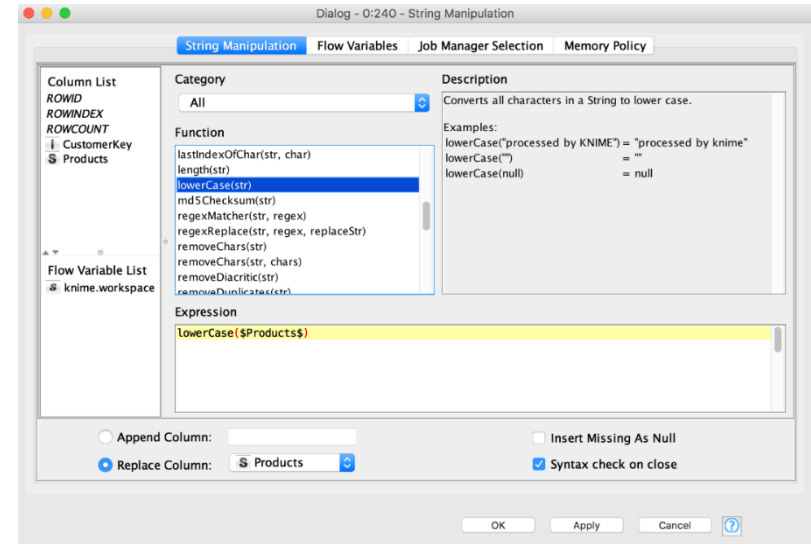
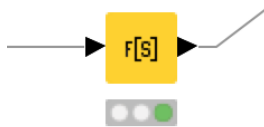
?

String Manipulation

Create and edit values in String columns

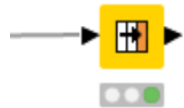
- Clean up capitalization (e.g. Lowercase)
- Replace strings
- Modify existing strings or create new columns

String Manipulation



More Nodes

String Replacer



Dialog - 2:297 - String Replacer

File

Standard settings | Flow Variables | Job Manager Selection | Memory Policy

Target column:

Pattern type: ☒ Wildcard pattern ☐ Regular expression

Pattern:

Replacement text:

Replace ...: ☒ ... whole string ☐ ... all occurrences

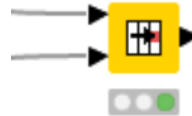
Case sensitive search: ☒

Use backslash as escape character: ☐

Append new column: ☐

OK Apply Cancel ?

Cell Replacer



Dialog - 2:298 - Cell Replacer

File

Options | Flow Variables | Job Manager Selection | Memory Policy

Input table: Target column:

Dictionary table: Input (Lookup): Output (Replacement):

Append/Replace Result Column: ☒ Append new column

If no element matches use: ☐ Input ☒ Missing

Metadata in Output: ☒ Copy metadata from replacement column

OK Apply Cancel ?

Rule Engine

Rule Engine



REOS rules

Dialog - 7:48 - Rule Engine (REOS rules)

File

Rule Editor | Flow Variables | Job Manager Selection | Memory Policy

Column List	Category	Description
ROWID	All	
ROWINDEX		
ROWCOUNT		
Sample	Function	? < ?
Pf3D7_ps_green		? <= ?
Pf3D7_ps_red		? = ?
Pf3D7_ps_hit		? > ?
Pf3D7_pEC50		? >= ?
ssr Molecule		? AND ?
SlogP		? IN ?
TPSA		? LIKE ?
AMW		? MATCHES ?
ExactMW		? OR ?
NumRotatableBonds		? XOR ?
NumHBD		FALSE
NumHBA		MISSING ?
NumAmideBonds		NOT ?
NumHeteroAtoms		TRUE
NumHeavyAtoms		
NumAtoms		
NumStereocenters		

Flow Variable List

knime.workspace

Expression

```
1 // enter ordered set of rules, e.g.:
2 // $double column name$ > 5.0 => "Large"
3 // $string column name$ LIKE "*blue*" => "small and blue"
4 // TRUE => "default outcome"
5 $AMW$<100 OR $AMW$>700 => "MW"
6 $NumHeavyAtoms$<5 OR $NumHeavyAtoms$>50 OR $NumRotatableBonds$ >=12 => "Complexity"
7 $NumHBD$>5 OR $NumHBA$>10 => "HBond"
8 $SlogP$<-5 OR $SlogP$>7.5 => "logP"
9 TRUE=>"Pass"
```

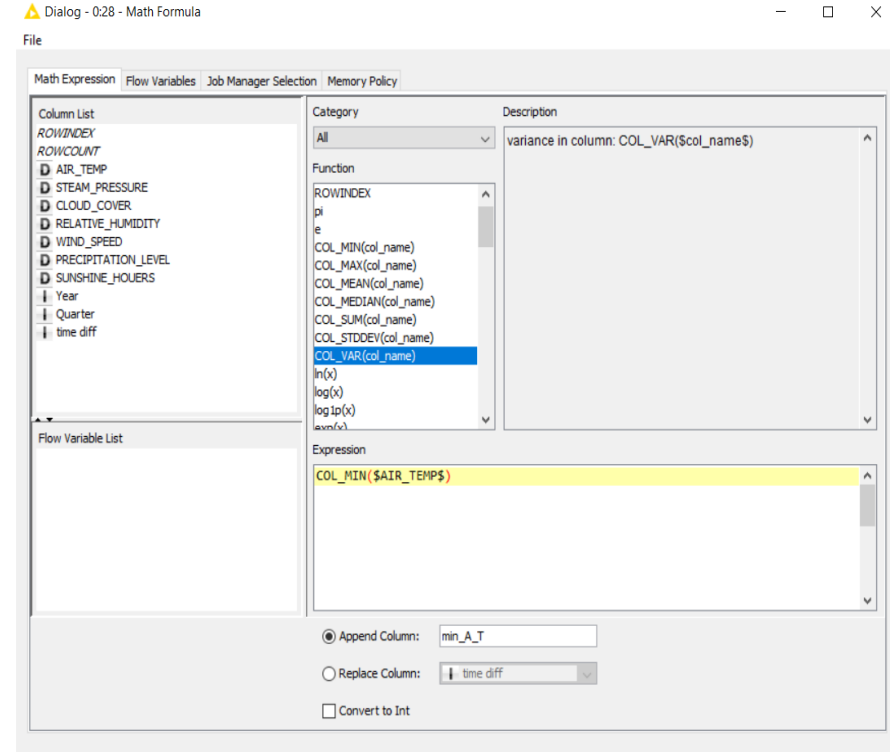
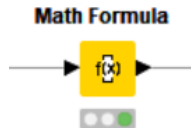
Append Column: REOS

Replace Column: NumHeavyAtoms

OK Apply Cancel ?

Math Formula

- Row-wise calculations
- Some column-wise statistics
- Lots of mathematical functions
- Double click on function, then select column



Column Expression

- Append or modify an arbitrary number of columns using expressions
- Many different functions are available
- No restriction on number of lines per expression allow to write complex expressions
- Part of the KNIME Labs extension

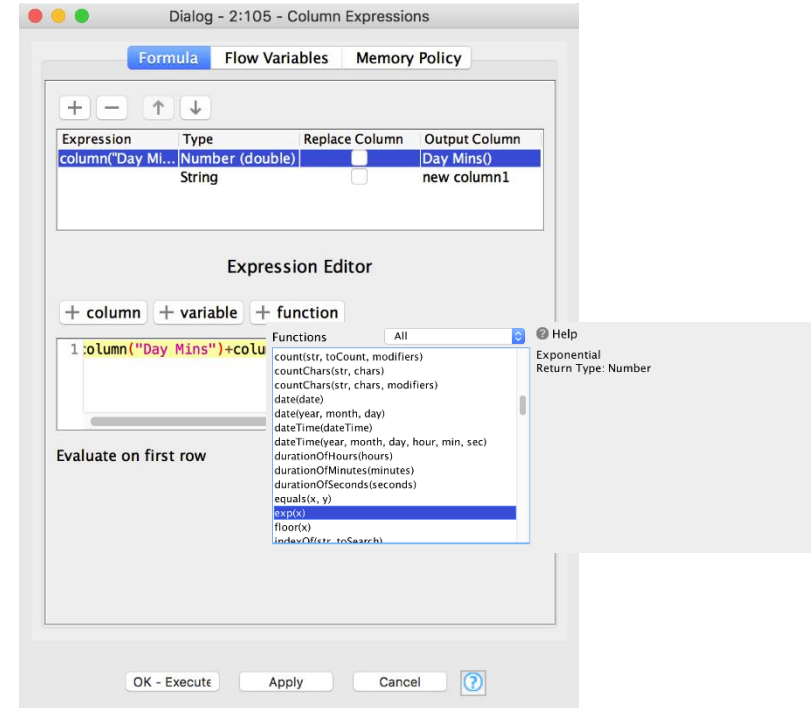
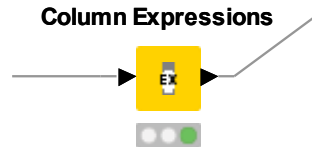
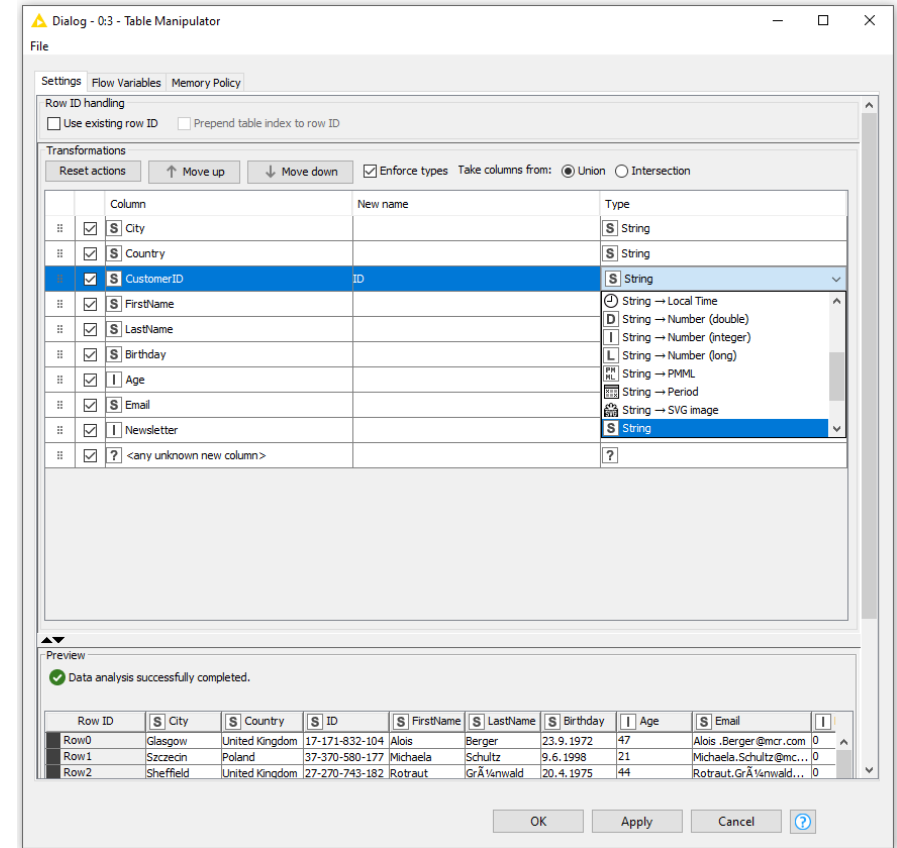
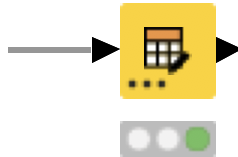


Table Manipulator

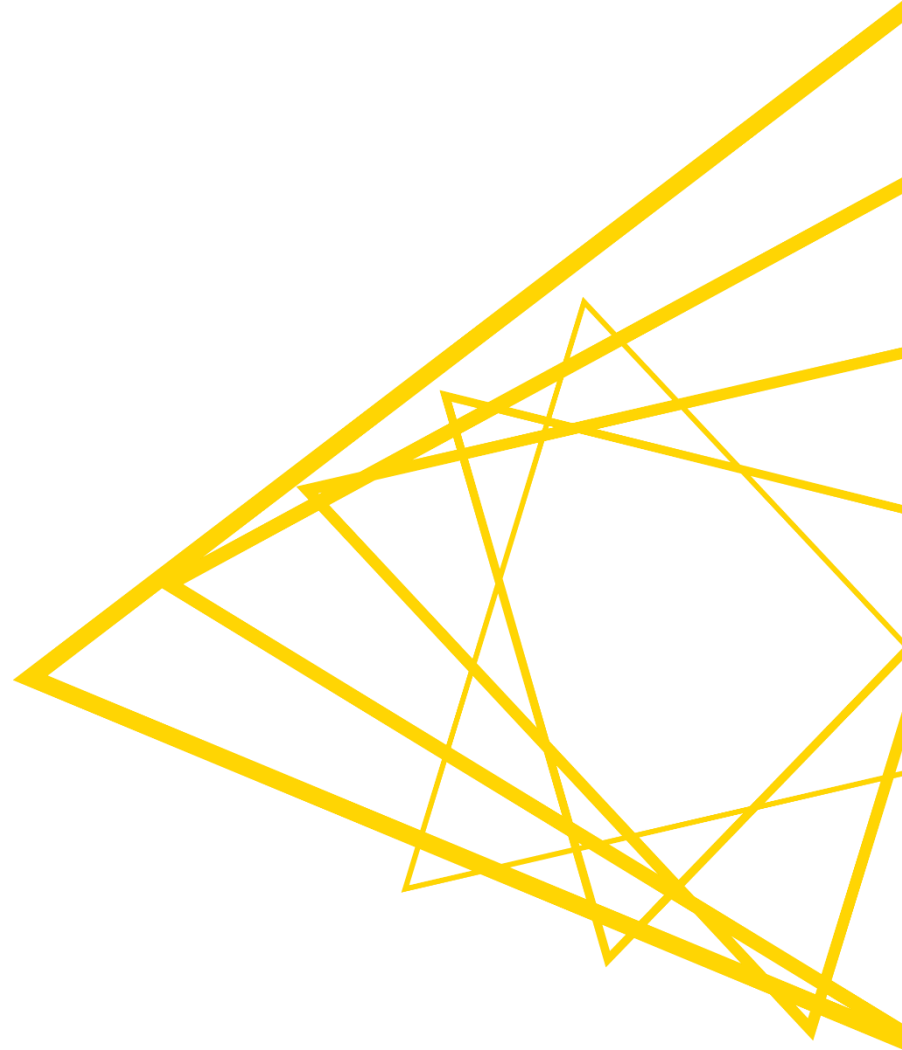
Allows for

- Concatenation of multiple files/tables
- Column filtering
- Column sorting
- Column renaming
- Column type mapping

Table Manipulator



Data visualisation



Data Visualization

- Large selection of easy to use visualization nodes
 - Web-based and interactive
 - Dedicated nodes, no scripting required
- Plotly nodes
 - Similar but integrated from an external library
- R and Python View nodes for highly customizable graphics
 - Require scripting

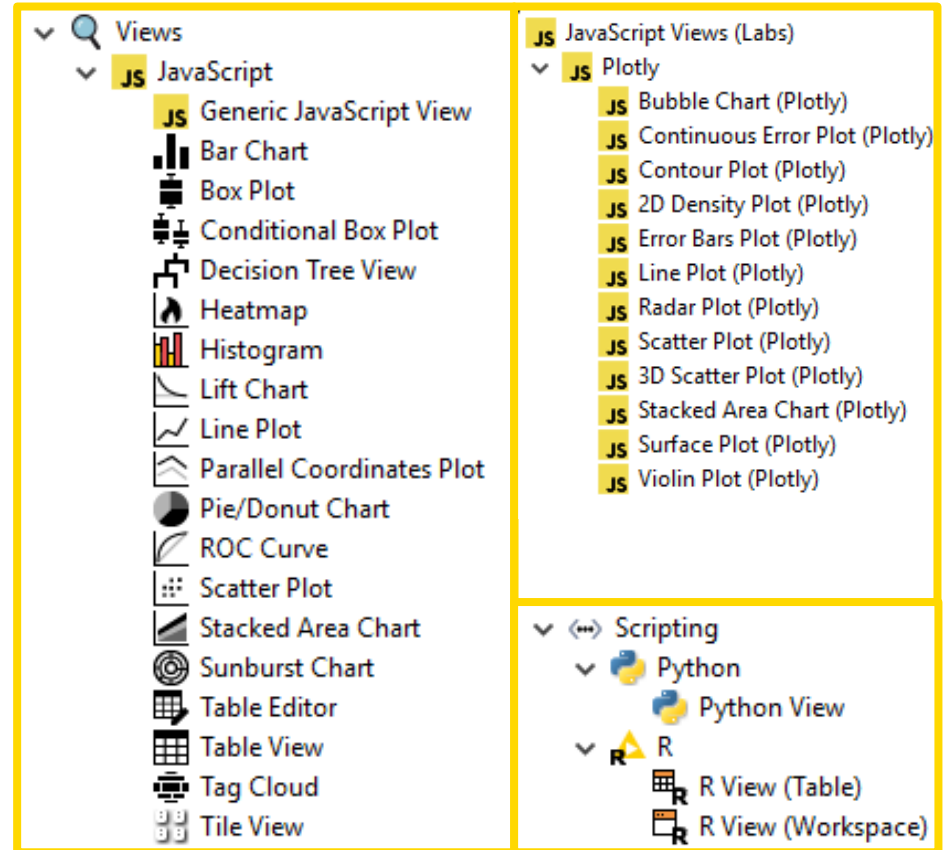
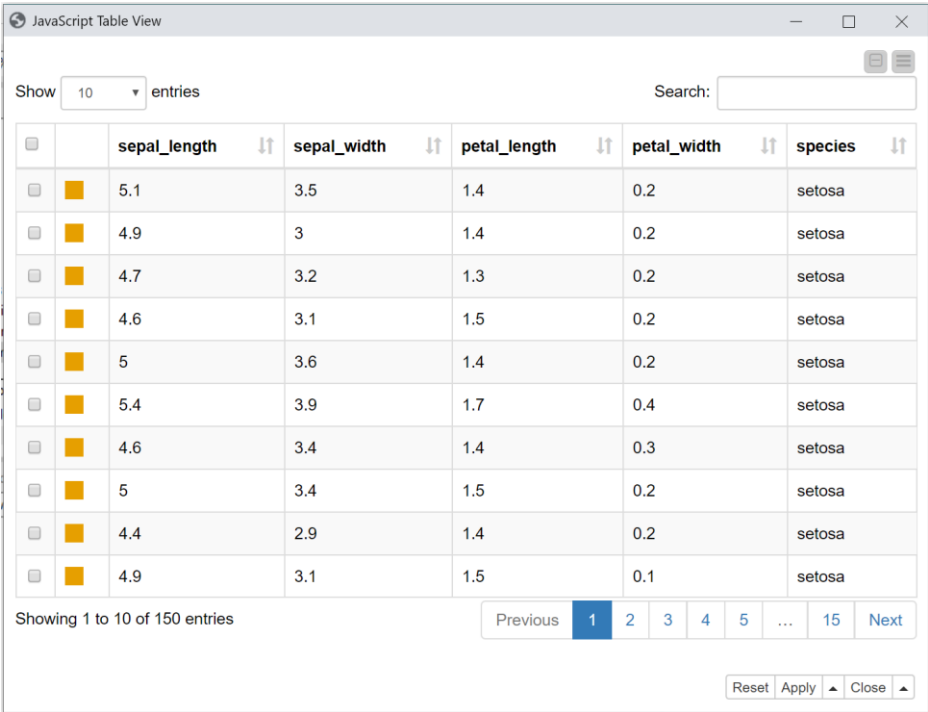
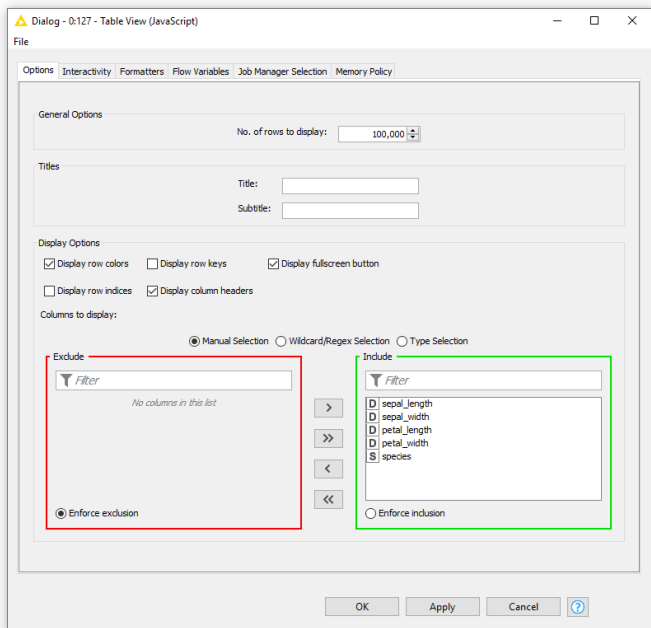
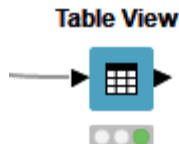
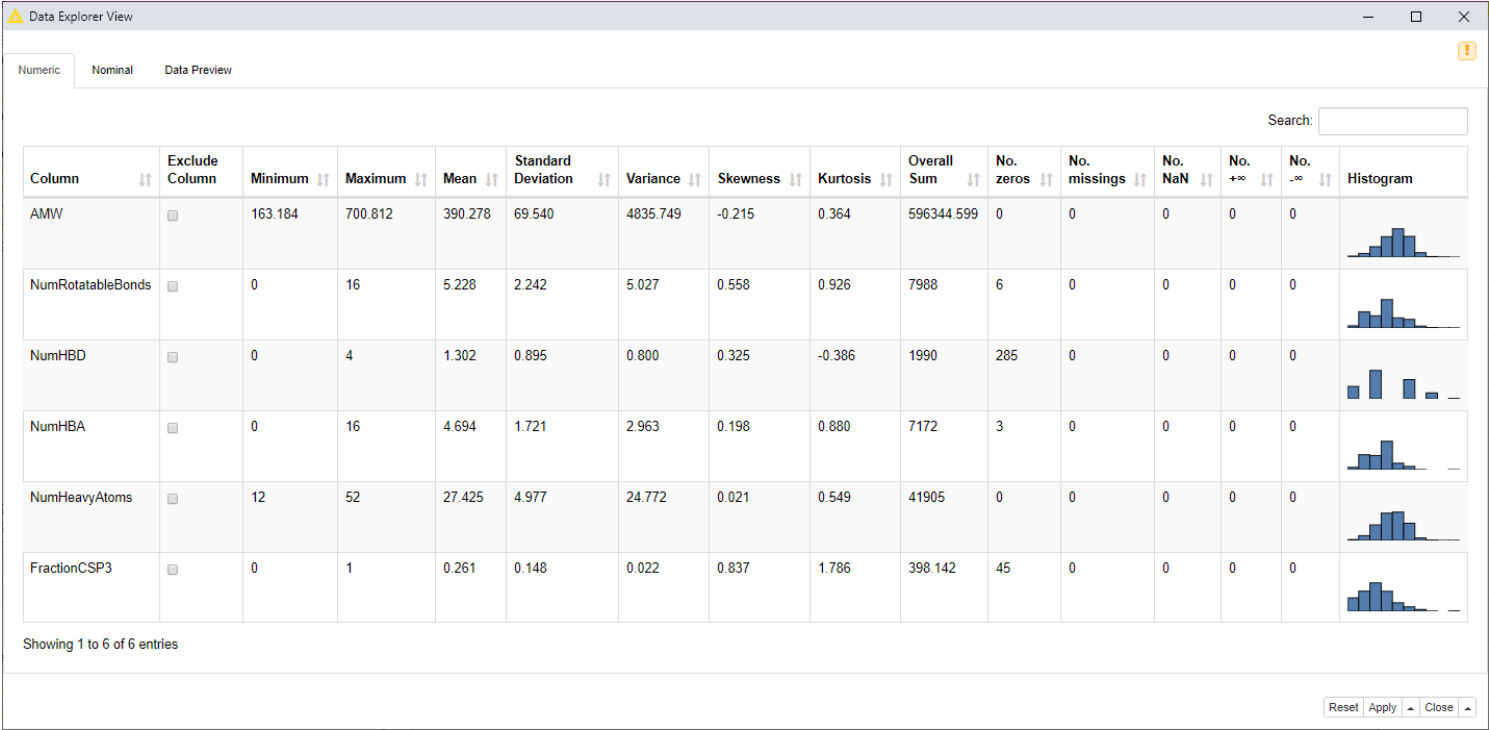
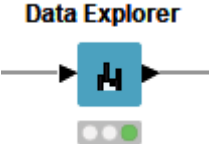


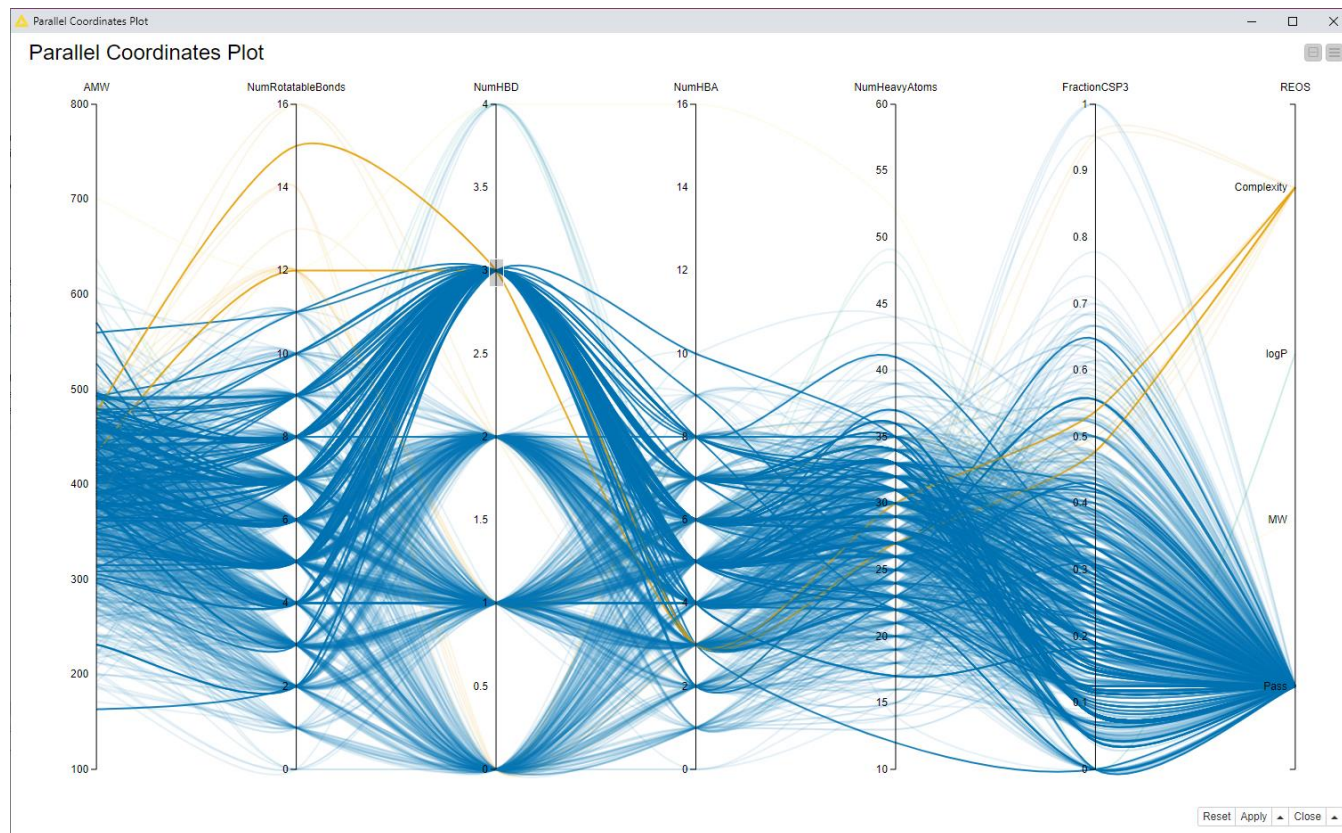
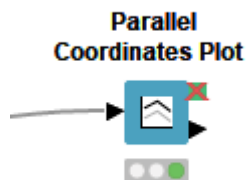
Table View



Data Explorer

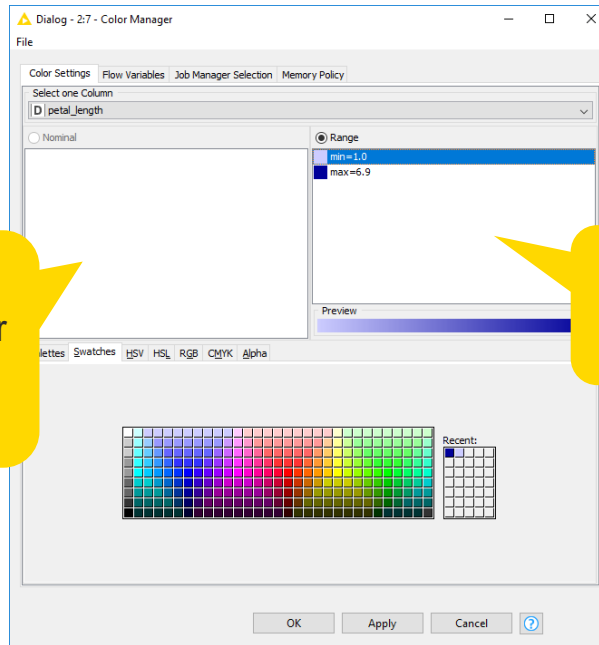


Parallel Coordinates Plot



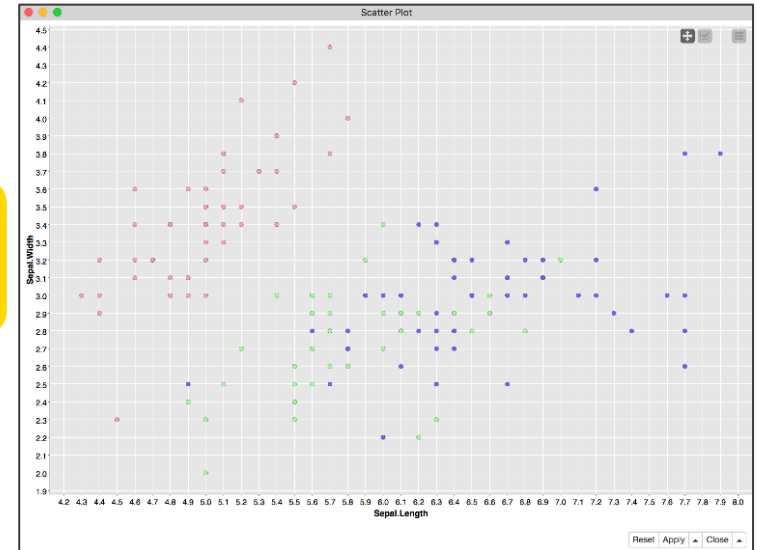
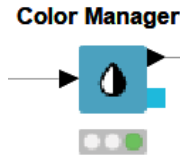
Color Manager

- Color by nominal or continuous values
- Sync colors between views using the color model port and Color Appender node



Discrete colors for nominal values

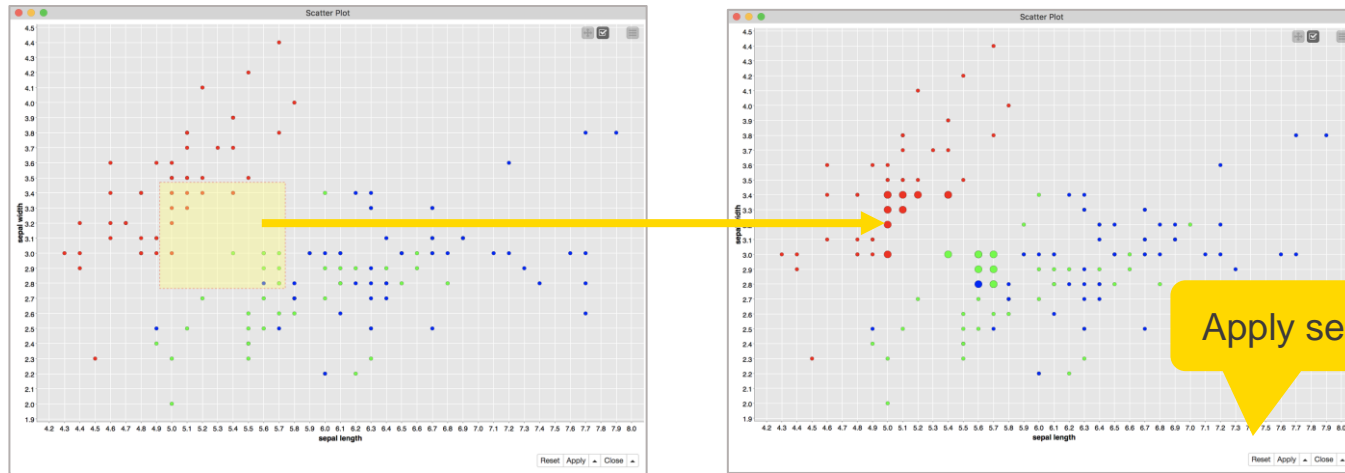
Color range for numerical values



Selection & Filtering in JavaScript Views

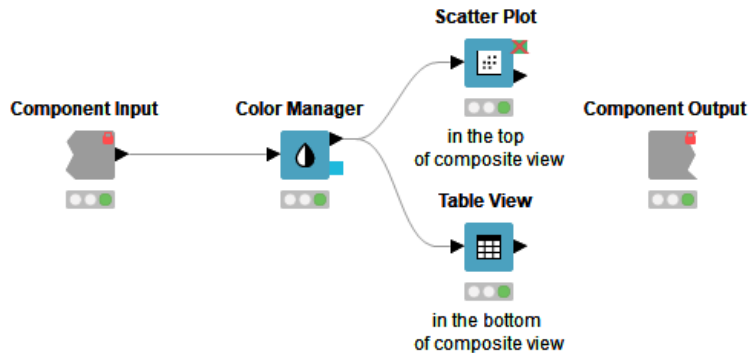
Interactivity allows you to select data points in views

- Selection is propagated to other views.
- Highlight selected rows or filter them
- Click “Apply” to add column to data that indicates selection (true/false) for use in downstream nodes

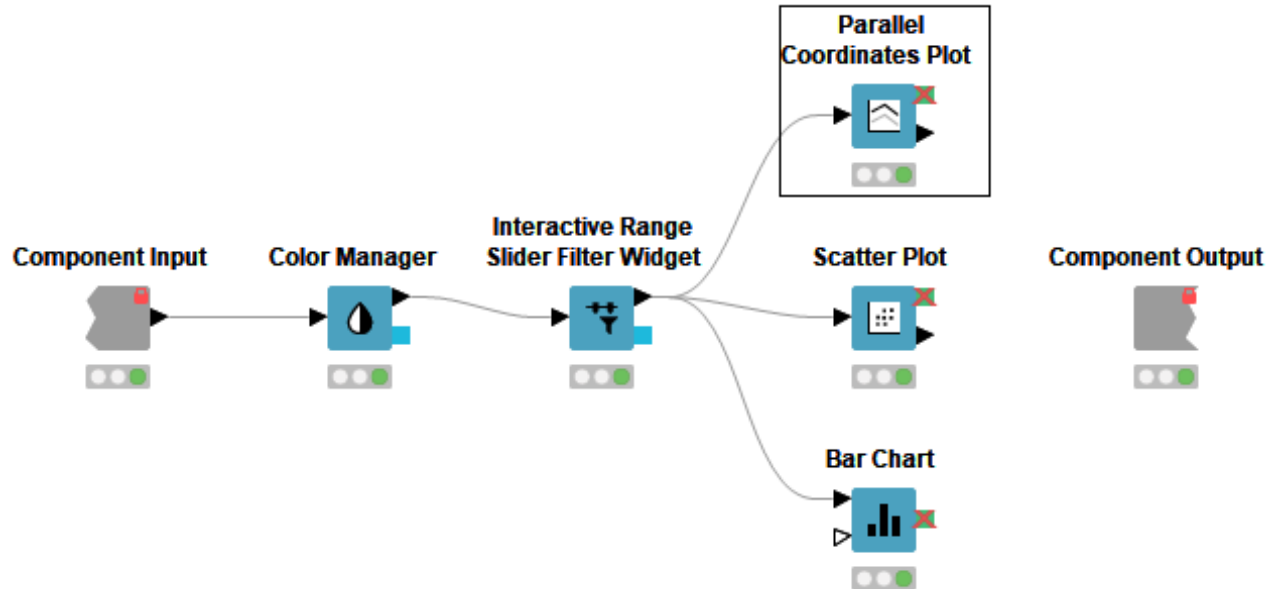


Components – Combined Views

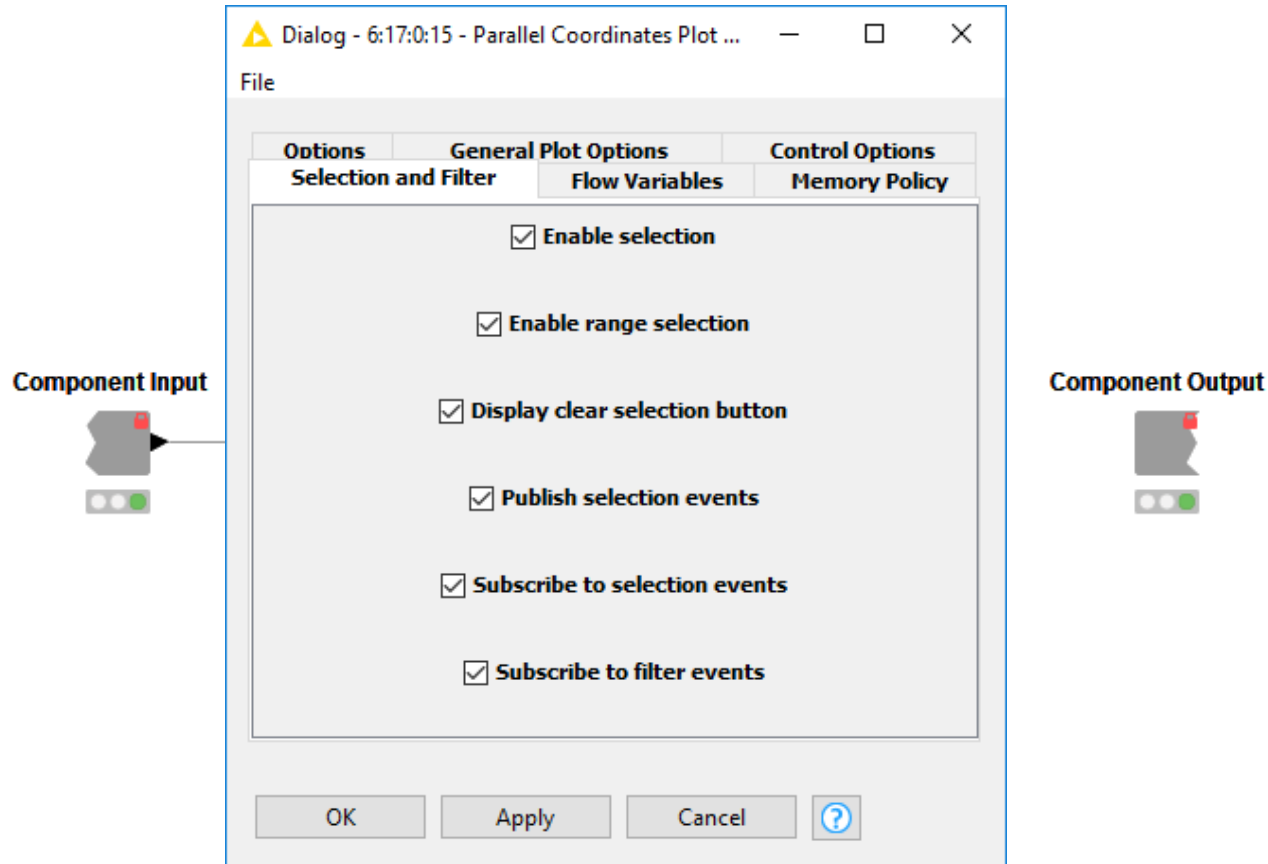
- Multiple JavaScript View nodes can be combined in Components
- Selections are transmitted to all other views
- Also for use on the KNIME WebPortal



Interactivity across Charts: Selection and Filter Events



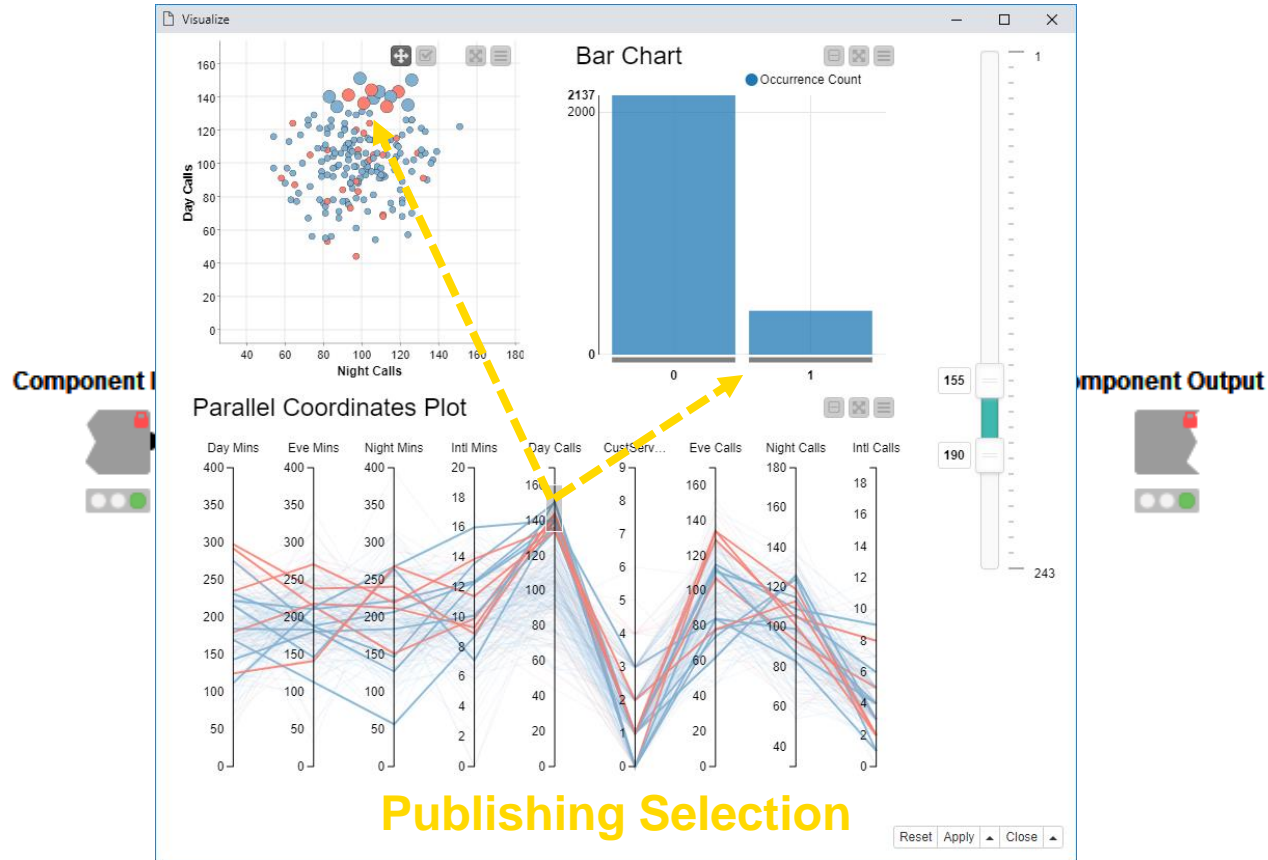
Interactivity across Charts: Selection and Filter Events



Interactivity across Charts: Selection and Filter Events

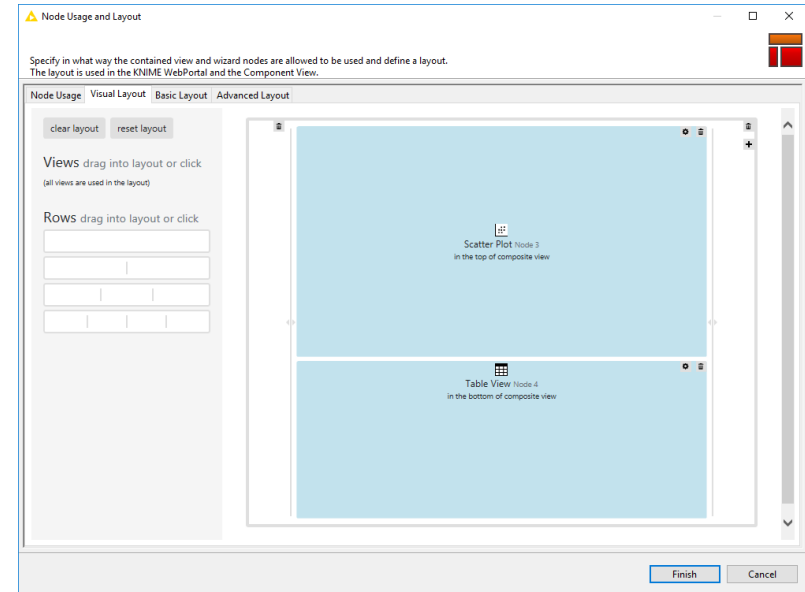
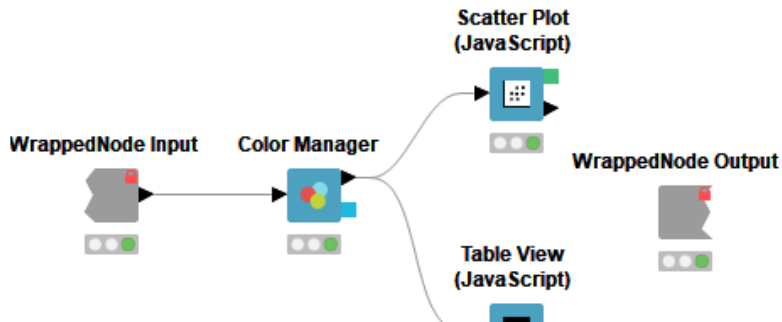
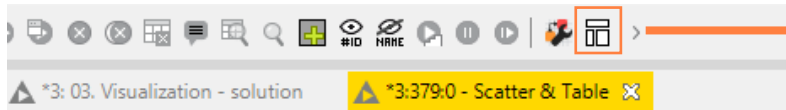


Interactivity across Charts: Selection and Filter Events



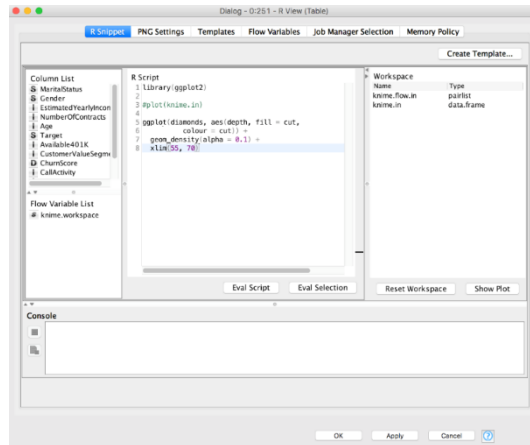
Configure Content and Views Layout

- Click layout button when inside Component to assign views to rows and columns
 - Add views and rows via *drag&drop*
 - Add columns using **+** buttons

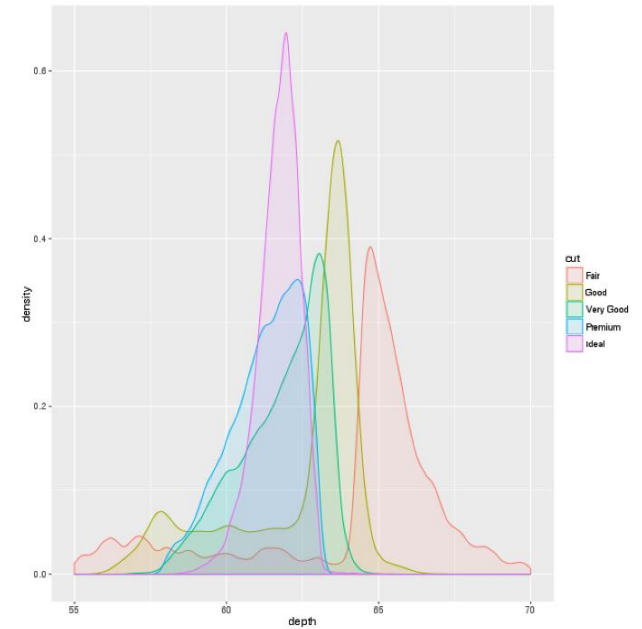


Script-based View Nodes

- R View nodes for greater customizability
 - Use your favorite libraries, e.g. ggplot2
- If you prefer Python: Python View node
- For JS developers: Generic JavaScript View

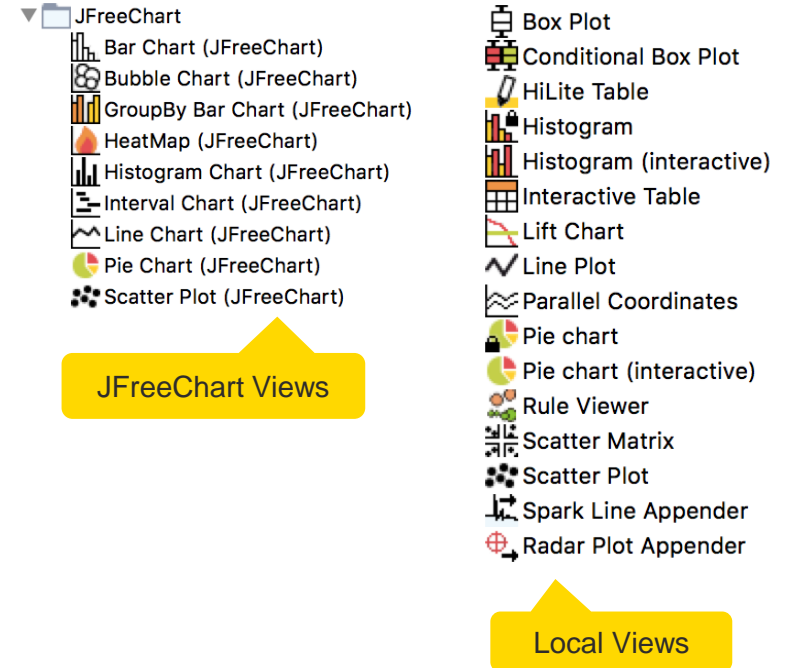


R View (Table)



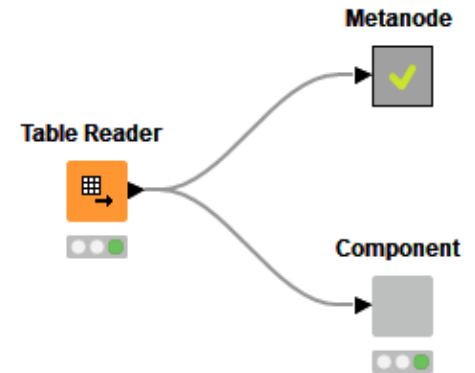
Legacy View Nodes: JFreeChart & KNIME Views

- KNIME provides three types of visualizations
 - **JavaScript Views**
 - JFreeChart Views
 - Local Views
- Active development only for JavaScript Views -> use those!
- JFreeChart and Local Views still useful when visualizing locally



Components

- Components encapsulate functionality for reuse and sharing
- Components main features:
 - Local Flow Variable scope
 - Configurable via Configuration nodes
- Components are the key to advanced functionality in KNIME products:
 - Components corresponds to a KNIME WebPortal page
 - Configurations on a WebPortal page are defined using Widget nodes
 - Possibility to be shared via KNIME Hub



Component Description

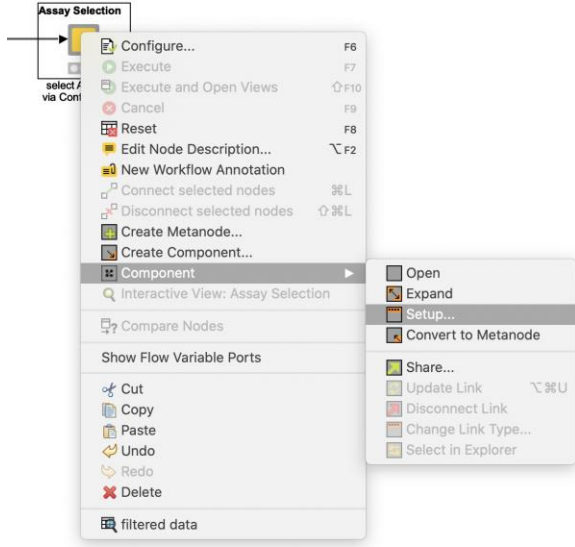
- Make your component look like a KNIME node

The image displays two screenshots of the KNIME software interface, illustrating the process of creating a custom component.

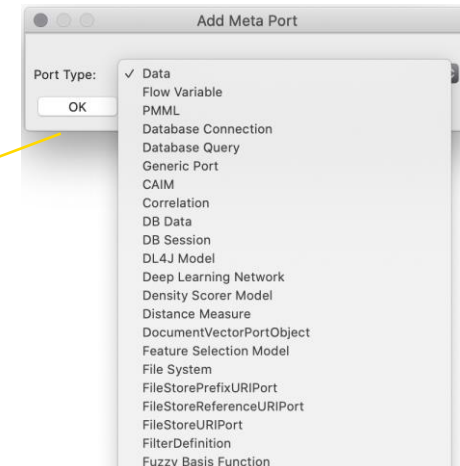
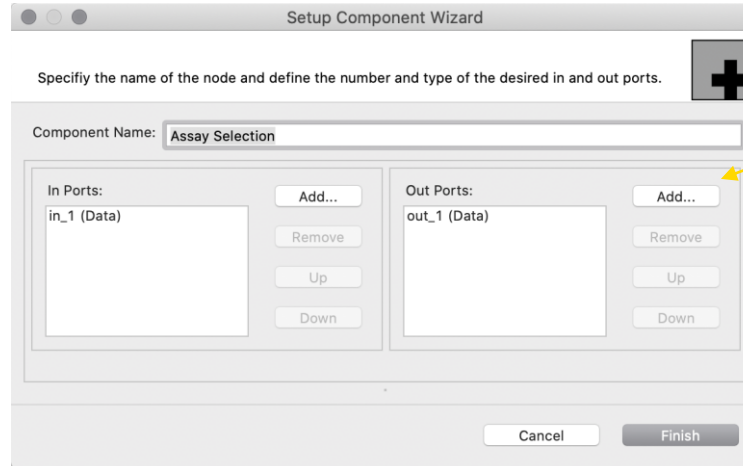
Top Screenshot: Shows the 'Assay Selection' component configuration window. The 'Description' field contains the text: 'This component filters the data by the selected AssayID'. The 'Component Icon' section shows a dropdown menu with options: Learner, Manipulator (selected), Predictor, Sink, Source, and Visualizer. The 'Port #1' field is set to 'all data' with the description 'Data containing records for all AssayIDs'. The 'Output Port #1' field is set to 'filtered data' with the description 'Data containing records for the selected AssayID'. A yellow callout points to the 'Description' field with the text: 'Add description of the component'. Another yellow callout points to the 'Port #1' field with the text: 'Add description of the input and output ports'. A third yellow callout points to the 'Component Icon' dropdown with the text: 'Add background color or icon'.

Bottom Screenshot: Shows the 'Assay Selection' component configuration window. The 'Description' field contains the text: 'This component filters the data by the selected AssayID'. The 'Dialog Options' section shows the 'Assay Selection' dialog with the text: 'Select AssayID to filter data'. The 'Ports' section shows the 'Input Ports' and 'Output Ports' fields. The 'Input Ports' field is set to '0 Data containing records for all AssayIDs'. The 'Output Ports' field is set to '0 Data containing records for the selected AssayID'. A yellow callout points to the 'Description' field with the text: 'Add description of the component'. Another yellow callout points to the 'Port #1' field with the text: 'Add description of the input and output ports'. A third yellow callout points to the 'Component Icon' dropdown with the text: 'Add background color or icon'.

Configure Component Ports

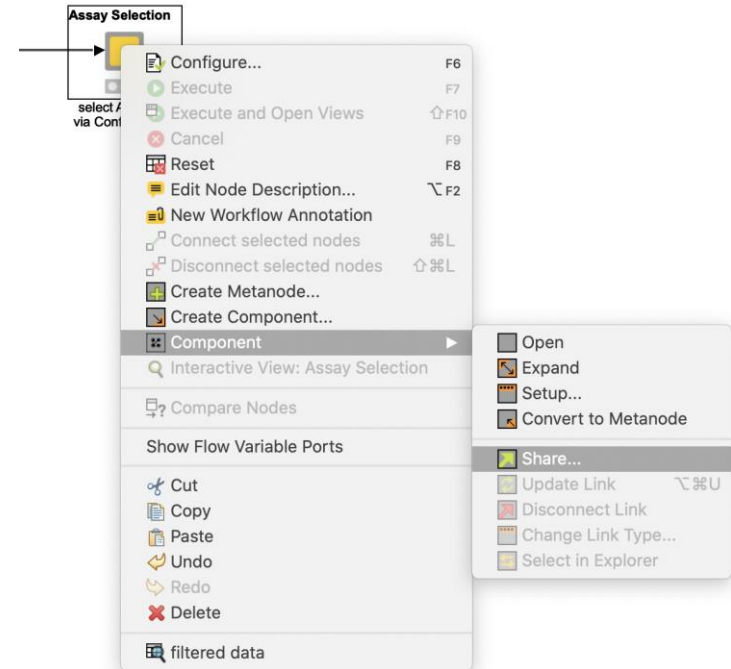


- Add input and output points to Metanodes/Components
- Remove ports to adapt to changes after creation of Metanodes/Components



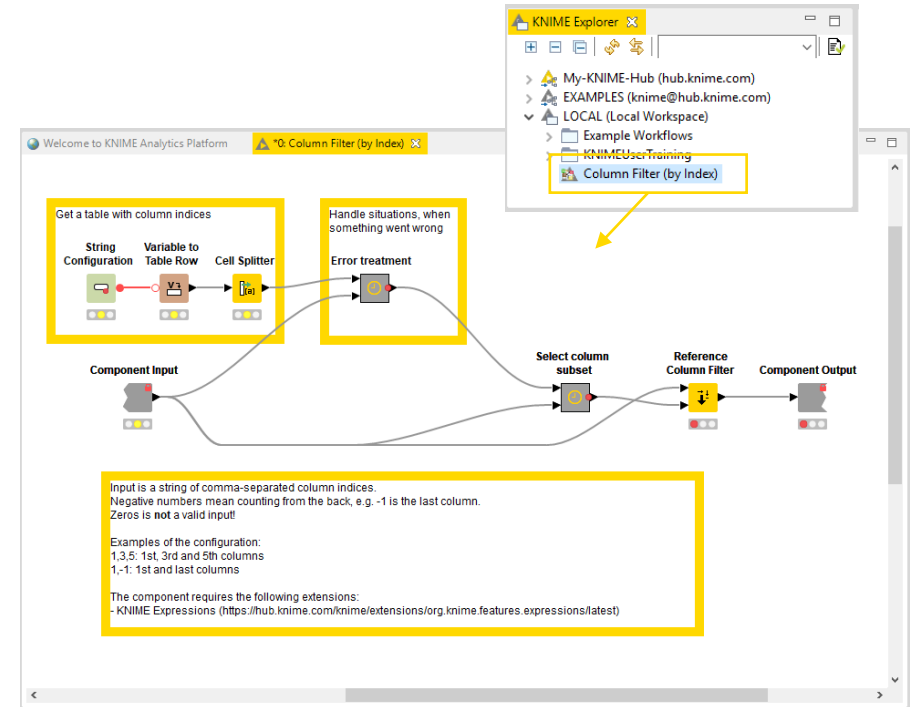
What is a shared component?

- Components can be saved in your KNIME workspace, KNIME Server or the KNIME Hub for later reuse
- To do this, simply right-click any Component and select “Share...”
- Shared Components are read-only instances of a Component
- Public Shared Components are available on EXAMPLES Server and on the KNIME Hub



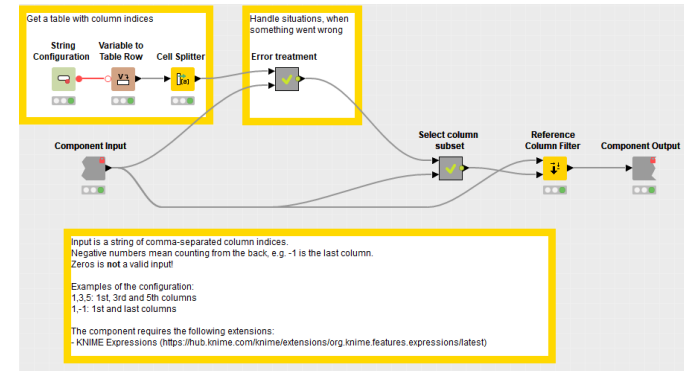
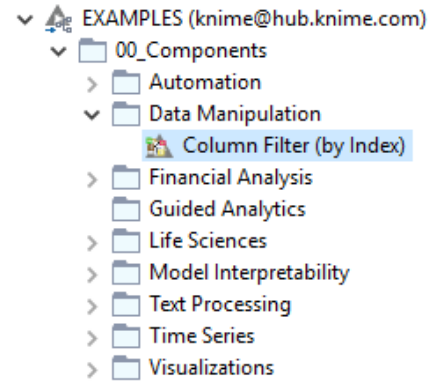
How can you edit a shared component?

- Components can be edited using the Component Editor similar to workflows
- To edit a Component using the Component Editor, double-click the Component in its location in the KNIME Explorer
- To ensure Components are executable when opened in the Component Editor, chose the option “Include input data with component” when sharing it

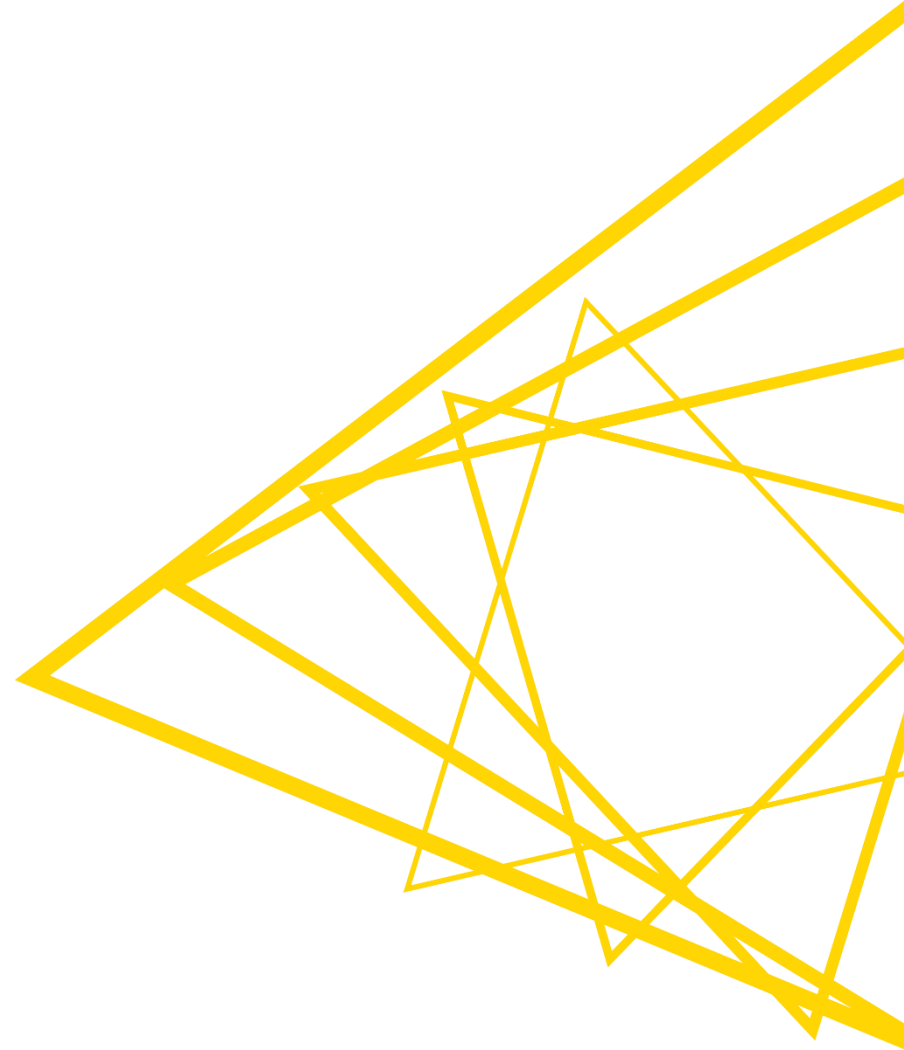


How can you use a shared component?

- To use a Shared Component, drag and drop it to the workflow editor
- Instances of Shared Components can be updated either manually or when workflow is opened
- Shared Components can also be unlinked from its original location, which makes it editable in the workflow directly
- Update Shared Components by overwriting them

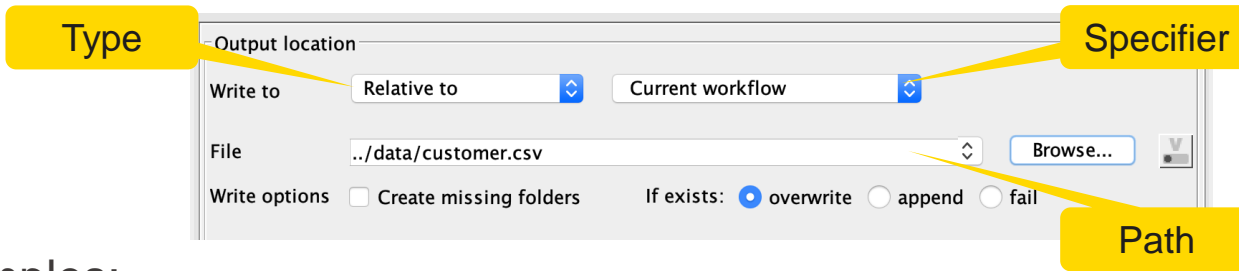


Flow Variables and Loops



Common Settings: File Path

- A path consists of three parts:
 - **Type**: Specifies the file system type e.g. local, relative, mountpoint, custome_url or connected.
 - **Specifier**: Optional string with additional file system specific information e.g. relative to which location (knime.workflow)
 - **Path**: Specifies the location within the file system



- Examples:
 - (LOCAL, , C:\Users\username\Desktop)
 - (RELATIVE, knime.workflow, file1.csv)
 - (MOUNTPOINT, MOUNTPOINT_NAME, /path/to/file1.csv)
 - (CONNECTED, amazon-s3:eu-west-1, /mybucket/file1.csv)


Common Settings: File Path Options

■ Local File System

Input location

Read from:

Mode: ☒ File ☐ Files in folder

File: 


■ Relative to ...

Read from:

Current mountpoint


Current workflow data area

Current workflow

File: 


■ Mountpoint

Read from:

File: 

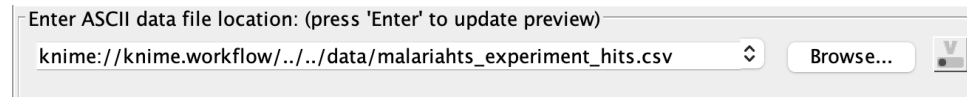
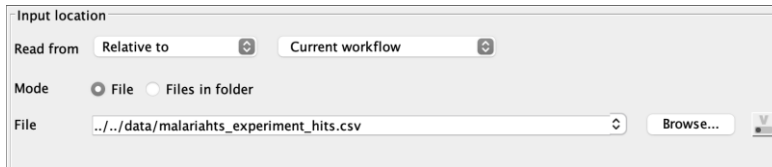
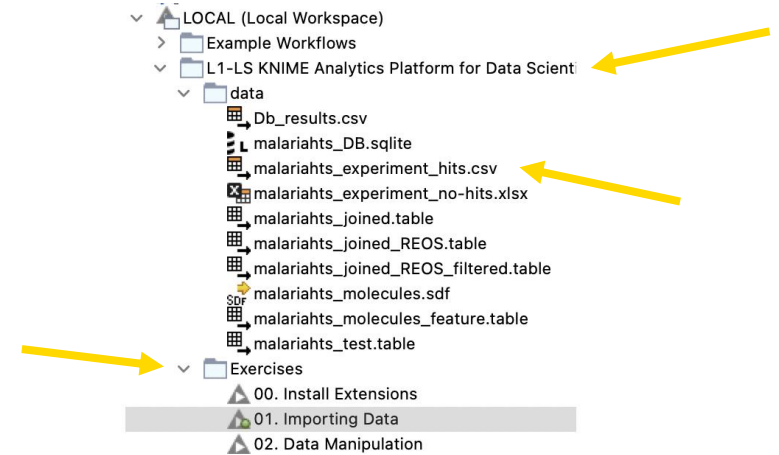
■ Custom URL

Read from:

URL: 

Workflow-Relative File Paths

- Best choice if workflows are to be shared
- Requires matching folder structure within workflow group
 - Independent of environment outside of workflow group
- Example: Path to „Sentiment Analysis.table“
 - Local path:
 - C:\Users\rb\knime-workspace\KNIMEUserTraining\data\malariahts_experiments_hits.csv
 - Workflow relative:



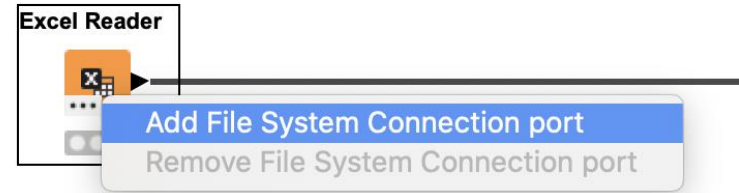
YouTube KNIME TV Channel:

<https://youtu.be/liZsOnhZgzk>

Common Settings: Connecting to other File Systems

- Add file system connection port to connect to another file system

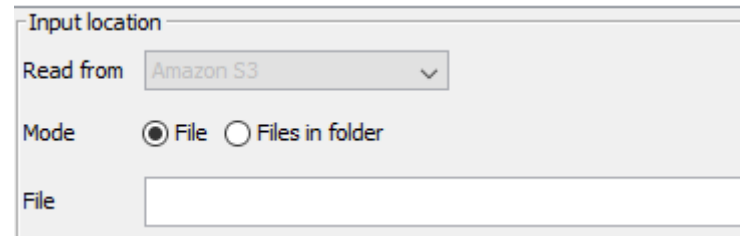
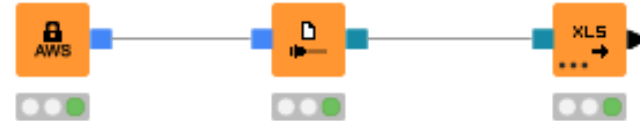
- Click on the three dots on the lower left to add or remove a dynamic port.



- Supported file systems

- Microsoft Azure
- Google
- Amazon
- Databricks
- BigData file systems (hdfs, httpFS, ...)
- On-premise (e.g. ssh, ftp, ...)

Amazon
Authentication Amazon S3 Connector Excel Reader (XLS)



Common Settings: Read Single or Multiple Files

■ Single file

Input location

Read from: Local File System

Mode: ☒ File ☐ Files in folder

File: /Users/kathrinmelcher/Desktop/course_data.csv

Browse...

■ Files in a folder

Input location

Read from: Relative to Current workflow

Mode: ☐ File ☒ Files in folder

Filter options ☐ Include subfolders

Folder: ../../data/

Browse...

Selected 22 of 22 files

- Option to include subfolder
- Option to define filter criteria

Filter options

File filter options

☒ File extension(s) .csv

☐ Case sensitive

☐ File name *

☐ Case sensitive ☒ Wildcard ☐ Regular expression

☐ Include hidden files

Folder filter options

☒ Folder name month

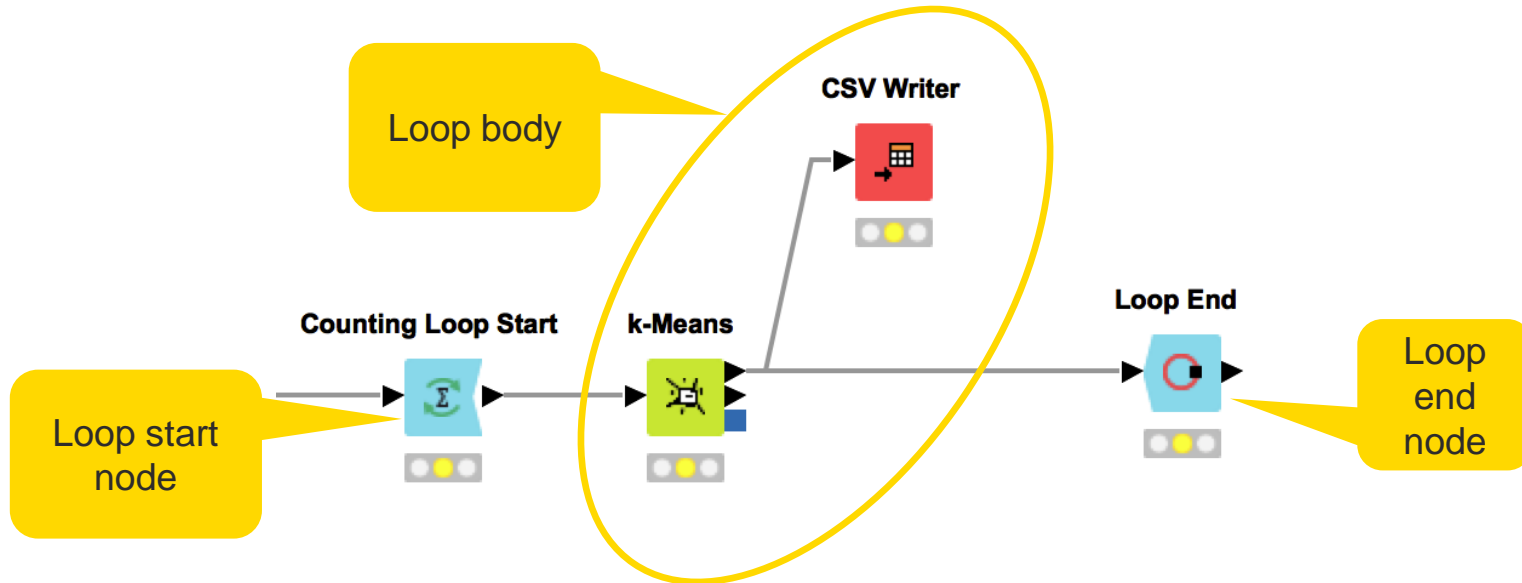
☐ Case sensitive ☒ Wildcard ☐ Regular expression

☐ Include hidden folders

OK Cancel

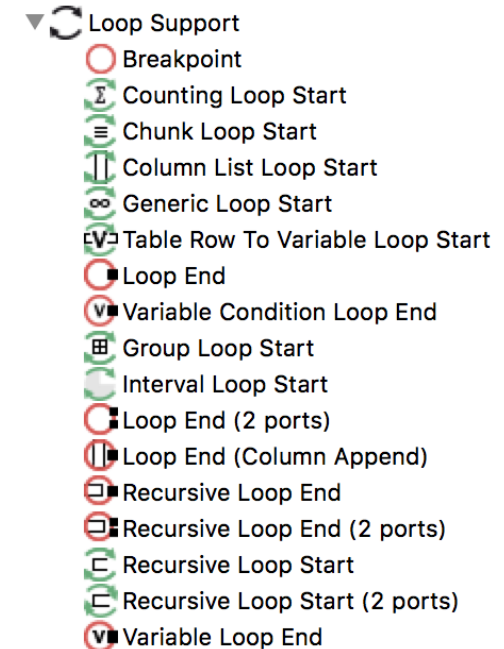
The Loop Block

- A loop block is defined by appropriate loop start and loop end nodes.
- Loop body = nodes in between (including side branches).



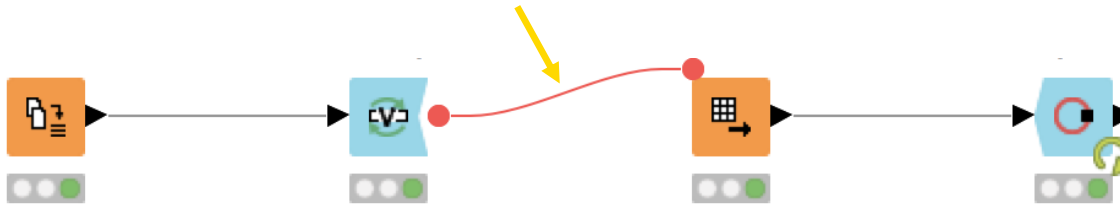
Loop Start and End Nodes

- For different tasks, there are different loop start and end nodes
- Nodes with circular arrows (green) = Start node
- Nodes with a closed circle (red) = End node
- Flow Variables are really helpful to build the loop body

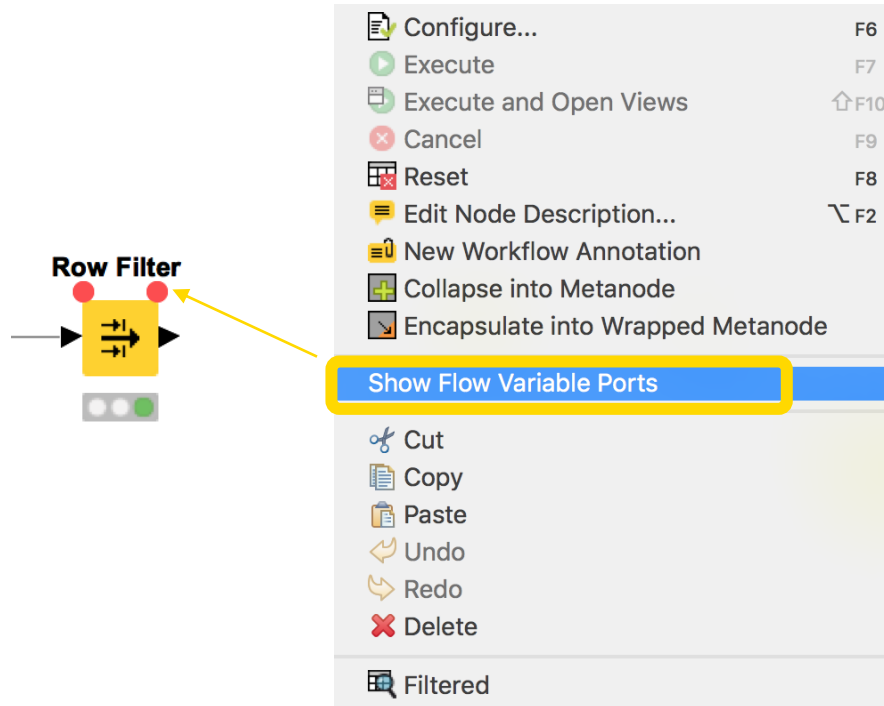


Flow variables

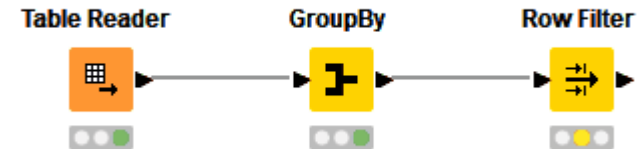
- Flow Variables are workflow parameters that are to overwrite existing node settings
- Especially in loops they can hold different values, depending on the iteration



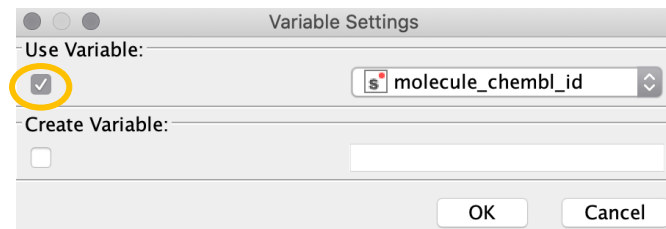
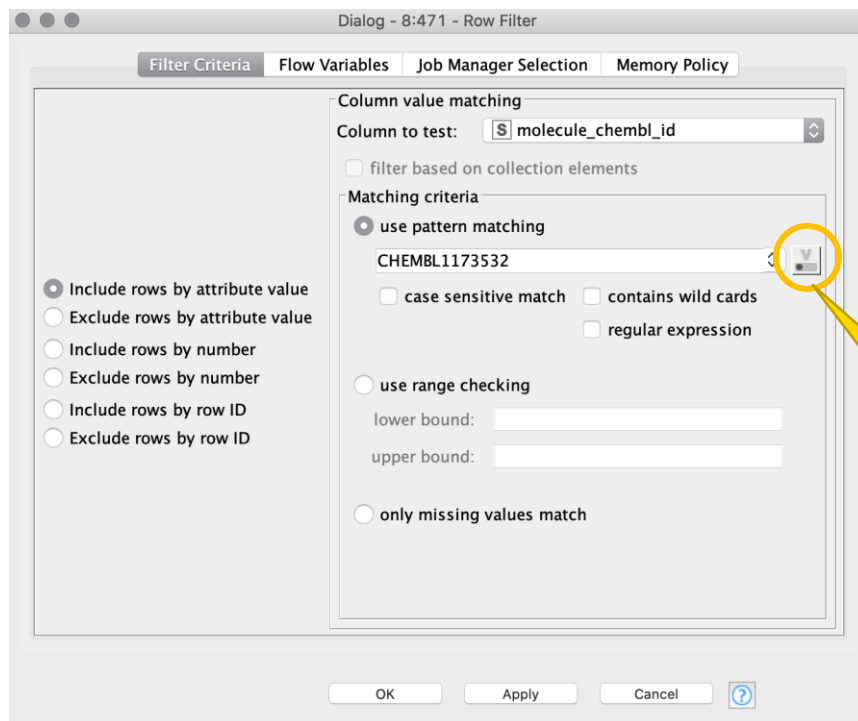
Flow Variable – Ports and connections



... or just drag and drop from one upper corner of a node to another

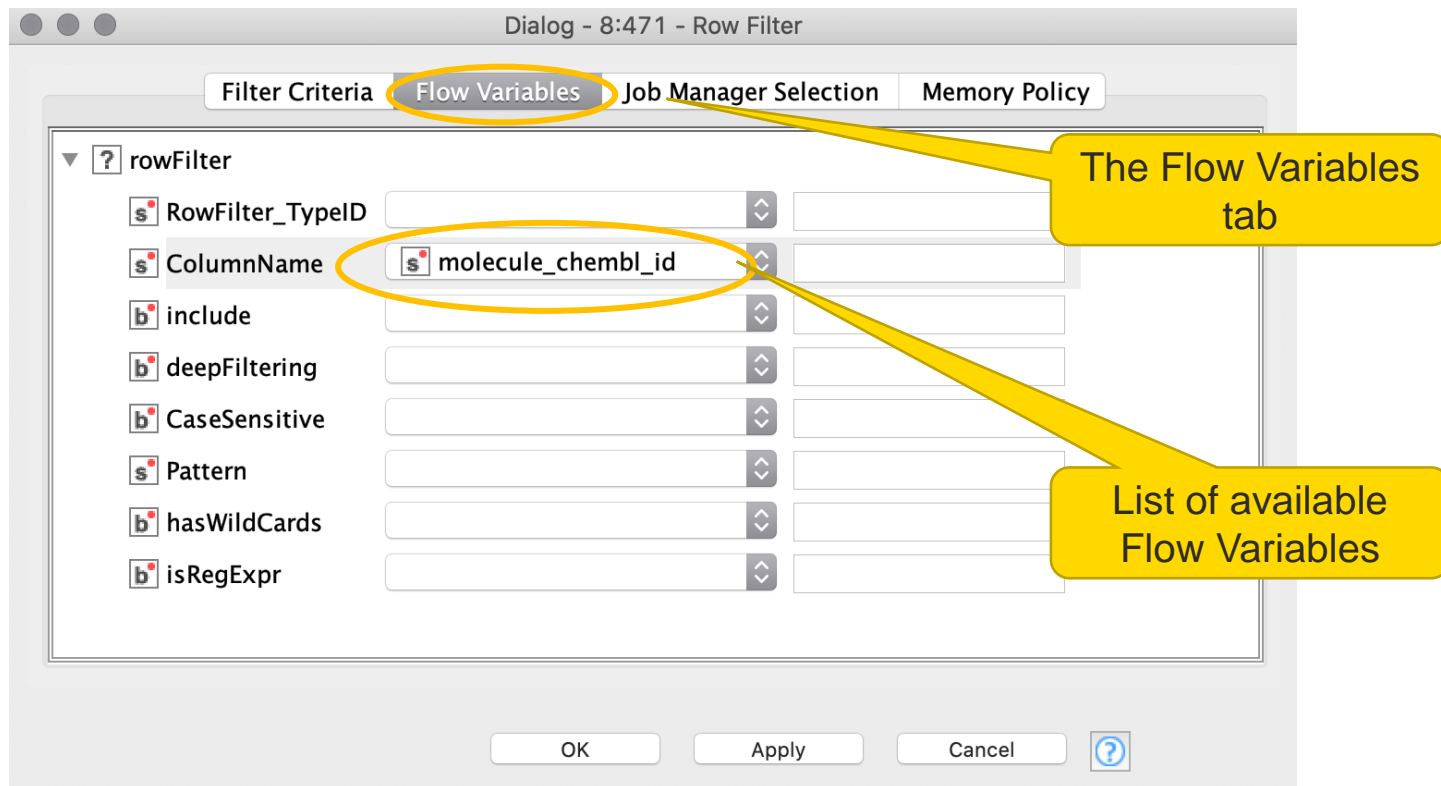


Apply a Flow Variable (Button)

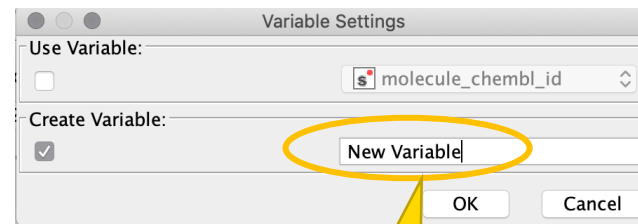
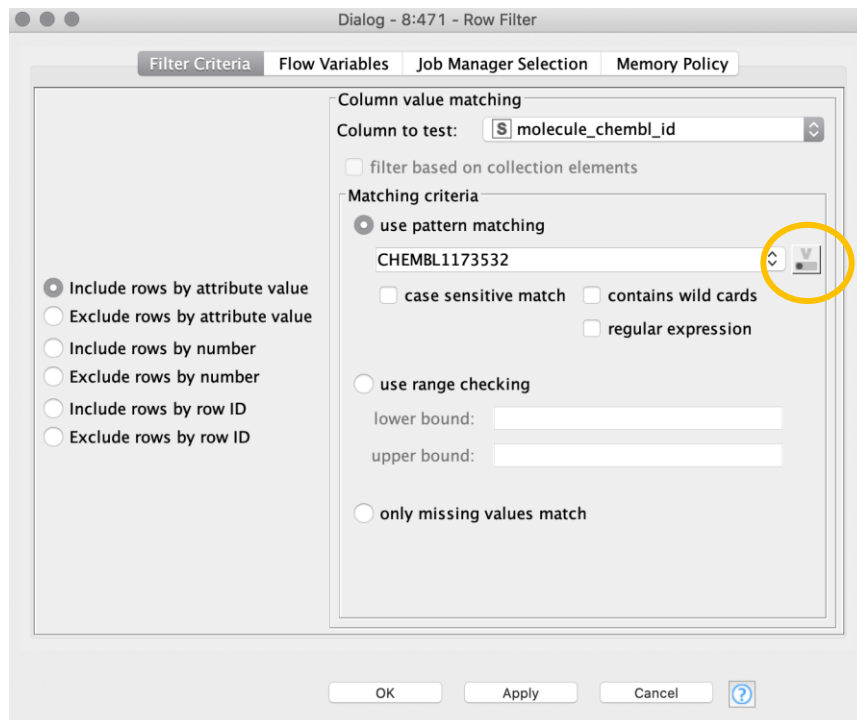


The Flow Variable button

Apply a Flow Variable (Advanced)



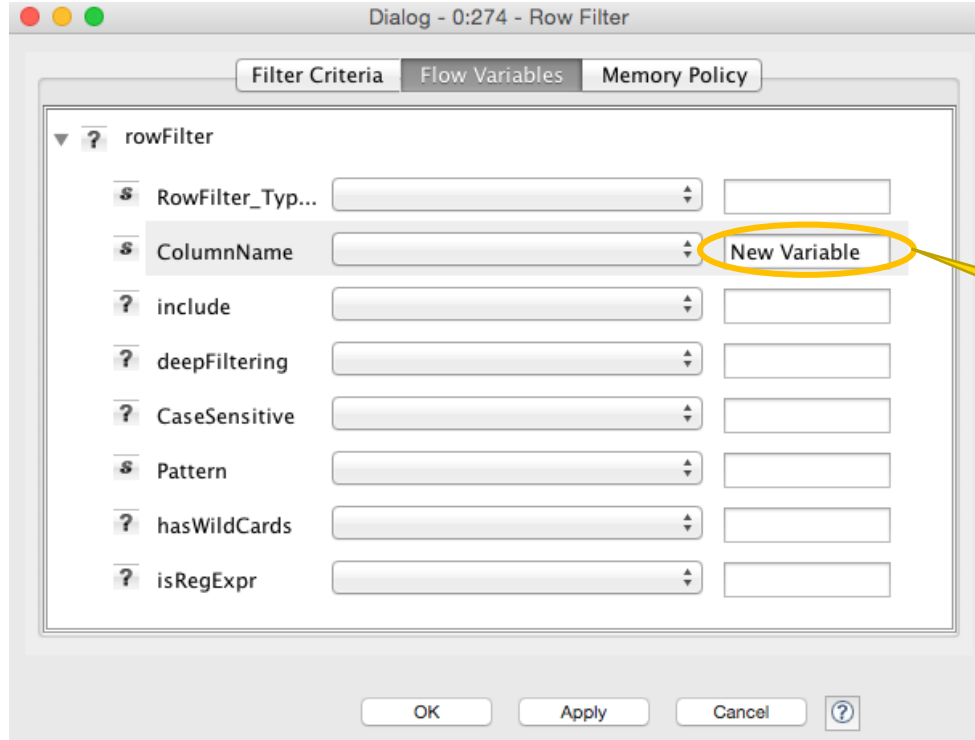
Create a Flow Variable (Button)



Name of the new
Flow Variable

Create a Flow Variable (Advanced)

- Converting a setting value into a Flow Variable



Key Features: Flow Variables

- Flow Variables are workflow parameters used to overwrite existing node settings
- A Flow Variable is carried along workflow branches (parallel branches don't share local Flow Variables)
- Flow Variables can be of type String, Integer, Double, Boolean, Long and Array
- Flow Variables can be created
 1. Using the "Table Row to Variable" node
 2. In the "Flow Variable" tab of any node
 3. Using Configuration and Widget nodes

Example 1: Reading many Excel sheets from one file

- List all sheet names of an Excel file
- Convert sheet name into a flow variable (1 sheet name per iteration)
- In each iteration, read the spreadsheet with the current sheet name
- Close the loop and collect the results

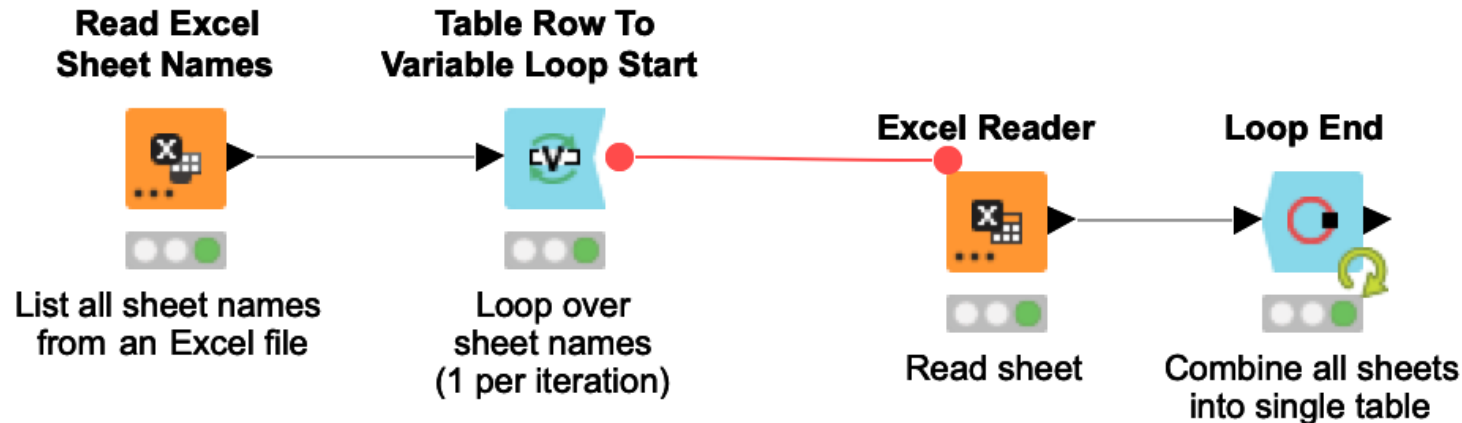
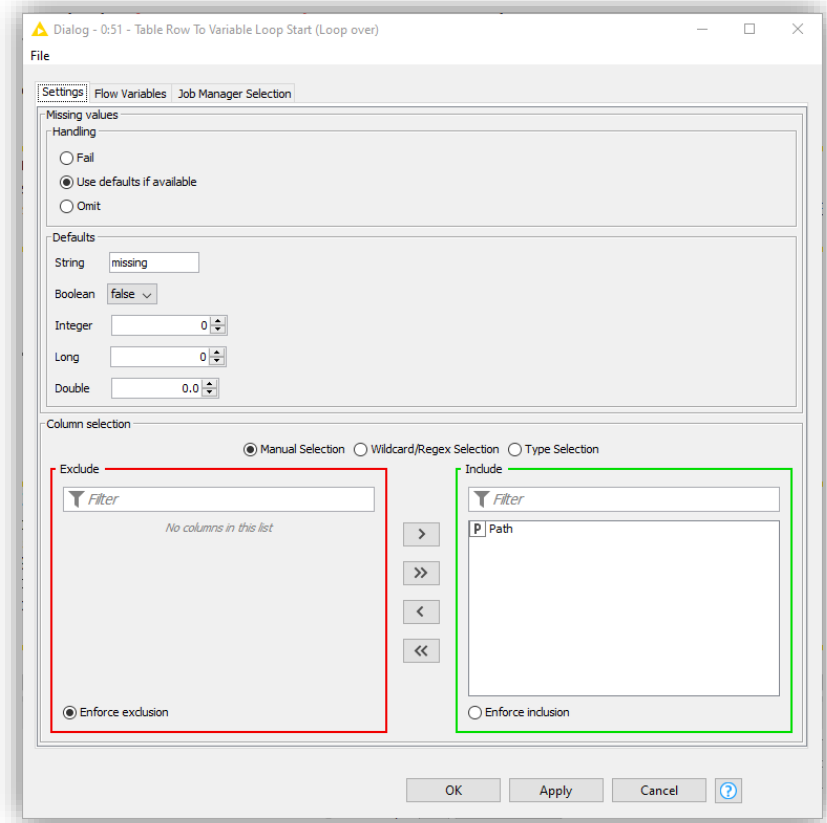
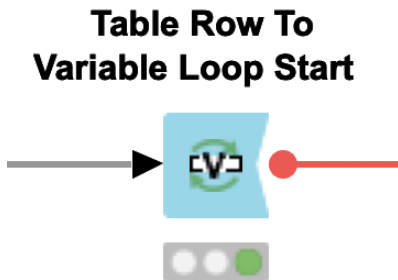


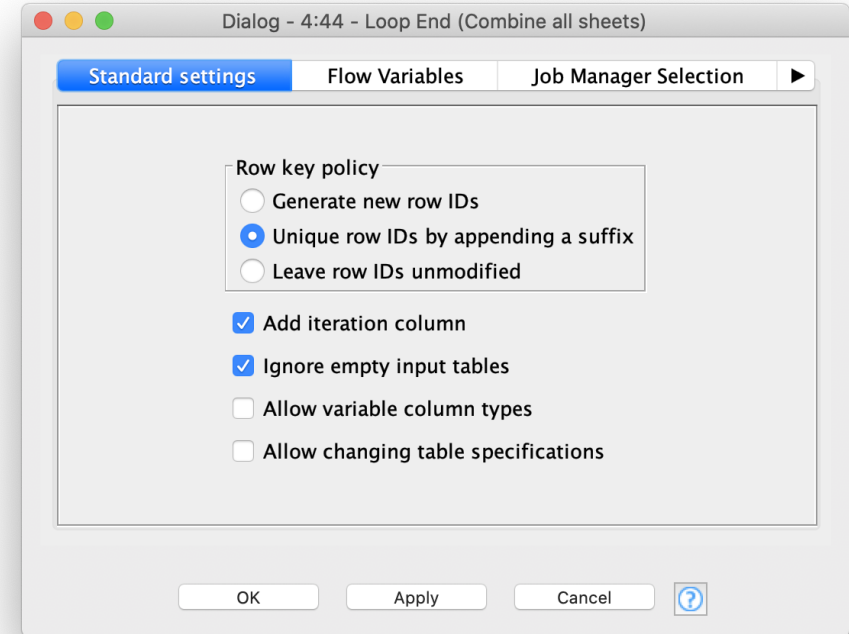
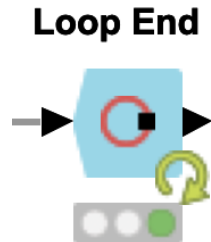
Table Row to Variable Loop Start

- Similar to the Table Row to Variable node
- Each iteration of the loop converts the next row of the input table into Flow Variables
- Injects variables into other nodes to re-execute subflows with a progression of settings



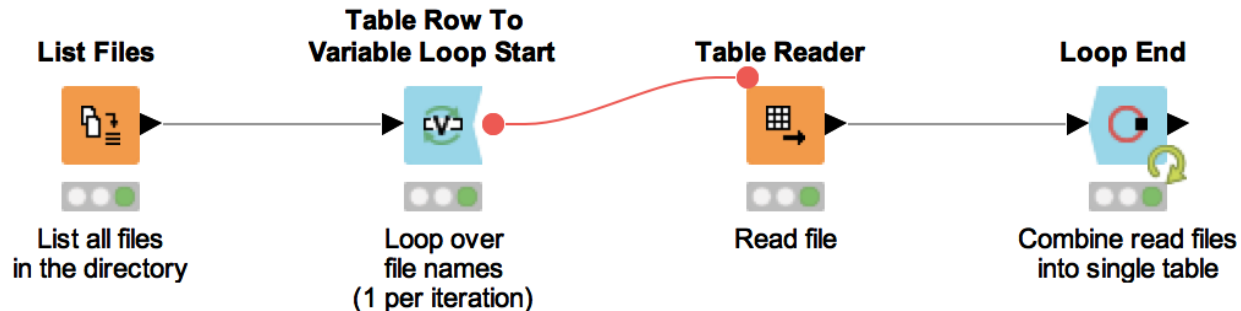
Loop End

- Can be used to end of a loop
- Collects the results of the different iterations by row-wise concatenation of the incoming tables
- Provides options to:
 - Add a column with the iteration number
 - Allow variable column types
 - Allow changing table specifications

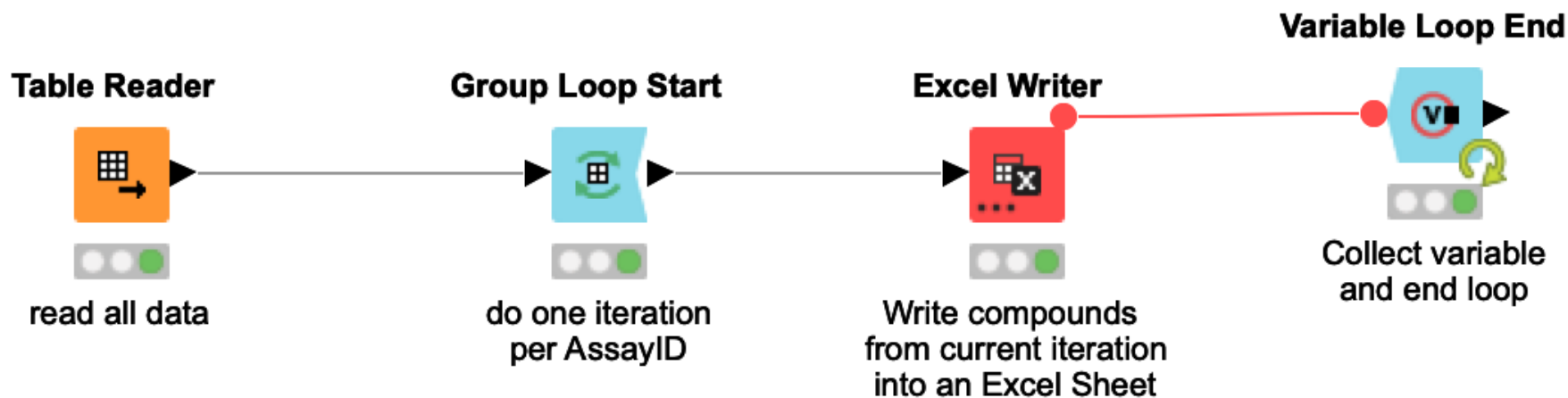


Similar: Reading Many Files

- List all files in a directory
- Convert each file path to a Flow Variable (1 per iteration)
- In each iteration, read the file and collect the results



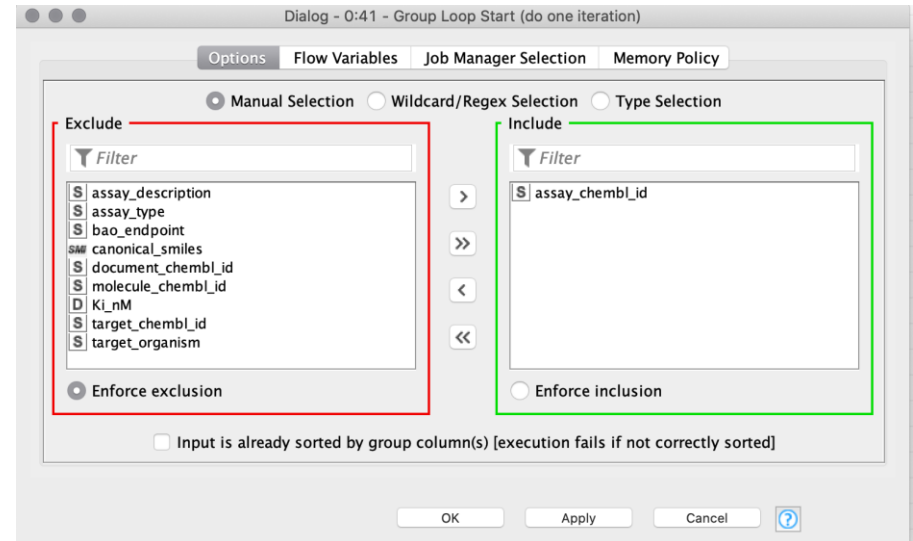
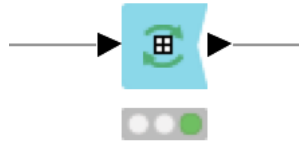
Example 2: Writing to multiple Excel Sheets in same file



Group Loop Start

- Similar to GroupBy except without aggregation tab
- Each iteration of the loop passes the next group of rows
- You implement the aggregation task. It can be anything from a complex calculation to updating a database

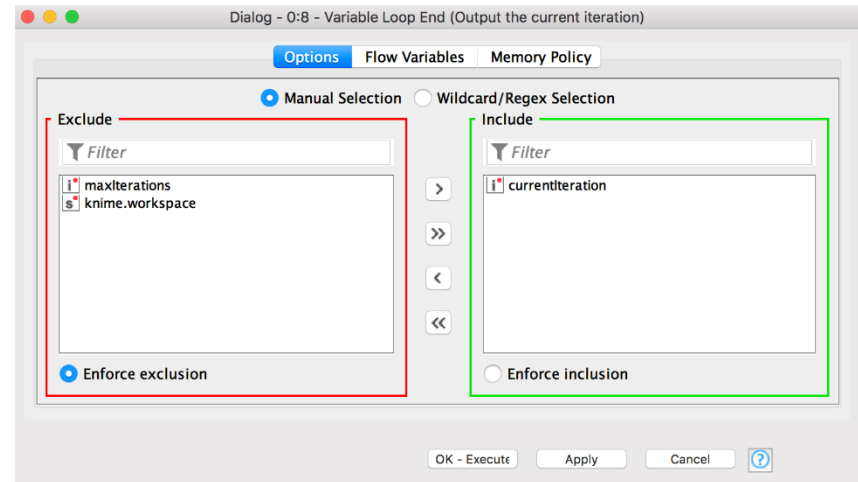
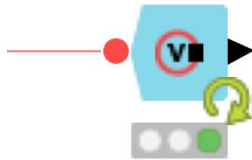
Group Loop Start



Variable Loop End

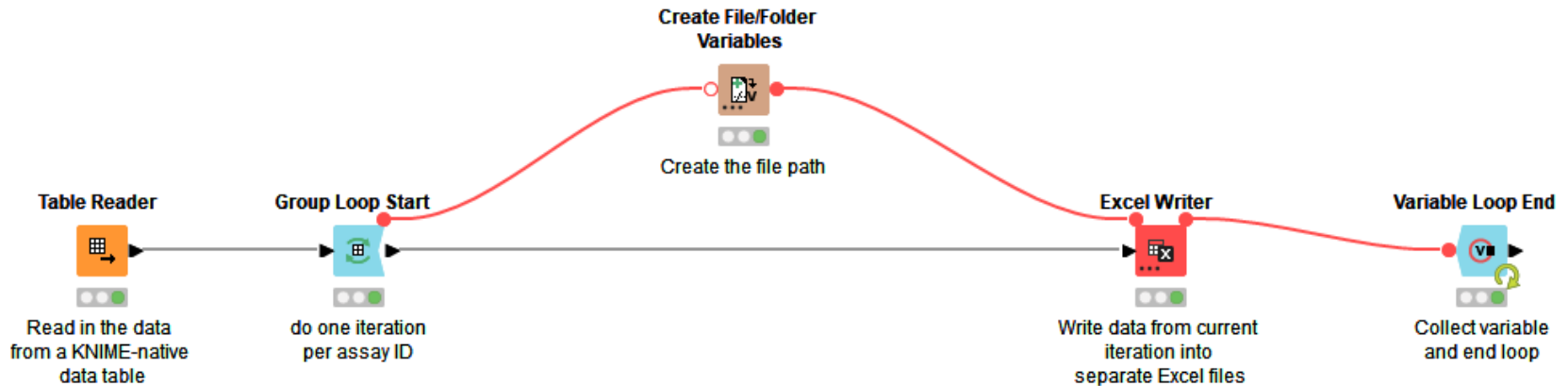
- Collects variables created in the loop
- Closes a loop without aggregating any data
 - Can be used to close loops where all the magic happens inside
 - e.g. in each iteration where the aggregated table is already written

Variable Loop End



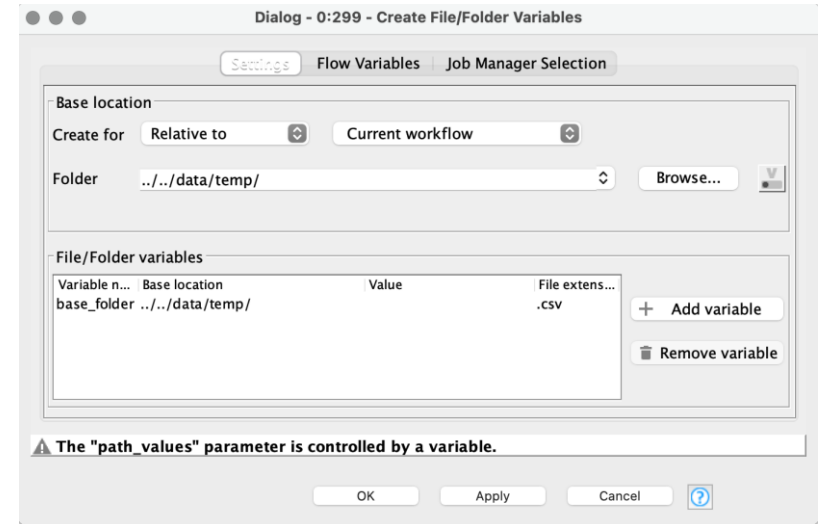
Example 3: Writing into separate files

- Group Loop Start → Variable Loop End
- Group data by specific column values
- Iterate over all groups of data
- Create an appropriate path variable
- Write grouped data to Excel files with new file name

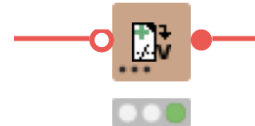


Create File/Folder Variables

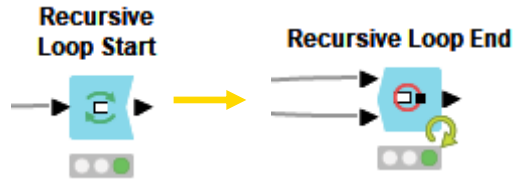
- Creates one or multiple path flow variable(s) pointing to files / folders
- Inputs:
 - Base location
 - Flow variable name(s)
 - Value (file name or path relative to base location)
 - File extension (optional)
- Output variables can be used to control the output location in writer nodes.



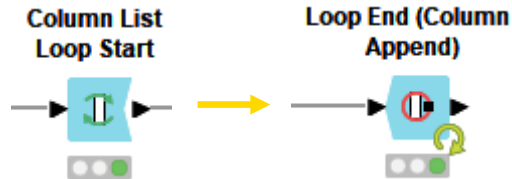
Create File/Folder Variables



Loop node pairs that go often together



The Recursive Loop node pair enables the passing of data tables from the Recursive Loop End back to the Recursive Loop Start.



Takes one column at a time and joins the output (i.e. the column after modification) to the current input table.



Execute the body of the loop until a certain condition on one of the flow variables is met.

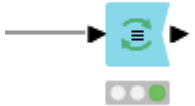
More loop start nodes

Counting Loop Start



Starts a loop which is executed a predefined number of times

Chunk Loop Start



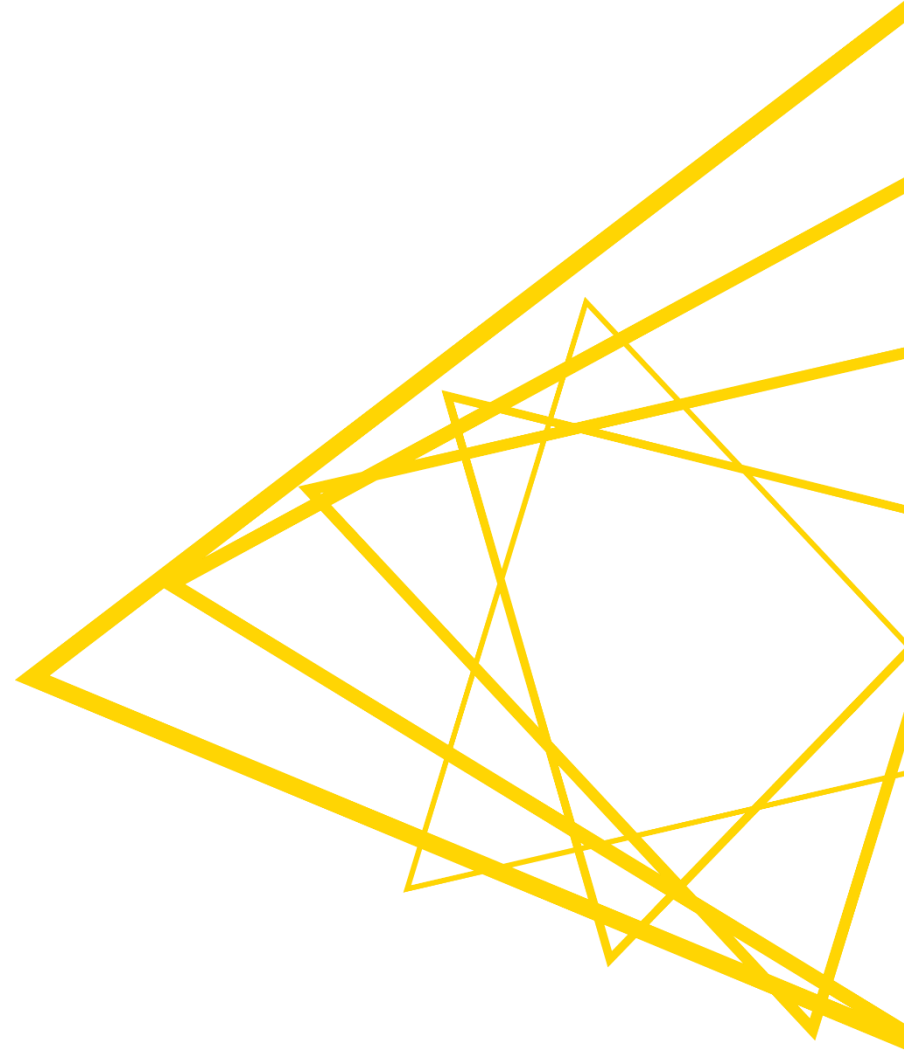
Each iteration processes another chunk of consecutive rows

Examples on the KNIME Hub

- Find more examples for loops on the [KNIME Hub](#)

Home > 06_Control_Structures > 04_Loops			
←			
📁	21_Parameter_optimization_loop		
📄	01_Loop_over_a_set_of_parameter_for_k_means		
📄	02_Example_for_Reading_a_List_of_Files		
📄	03_Looping_over_all_columns_and_manipulation_of_each		
📄	04_Looping_for_Multiple_Target_Prediction		
📄	05_Write_each_row_in_one_file		
📄	06_Writing_a_data_table_column_wise_to_multiple_csv_files		
📄	07_Example_for_Recursive_Looping		
📄	08_Example_for_Recursive_Replacement_of_Strings		
📄	09_Looping_over_Chunks_of_the_Data		
📄	10_Looping_a_fixed_number		
📄	11_Looping_over_Groups_of_the_Data		
📄	12_Using_TableRows_as_FlowVariables_in_Loop		
📄	13_Usage_of_Generic_Loop_Start		
📄	14_Looping_over_defined_Intervals		
📄	15_Collecting_Columns_in_Loop		
📄	16_Collecting_Variables_in_Loop		
📄	17_Usage_of_Breakpoint_in_Loops		
📄	18_Recover_from_Breakpoint_in_Loop		
📄	19_Forecasting_with_TimeDelay_Loop		
📄	20_Time_Series_Prediction_with_a_Recursive_Loop		

Error handling



Workflow Control Structures

■ Try-Catch

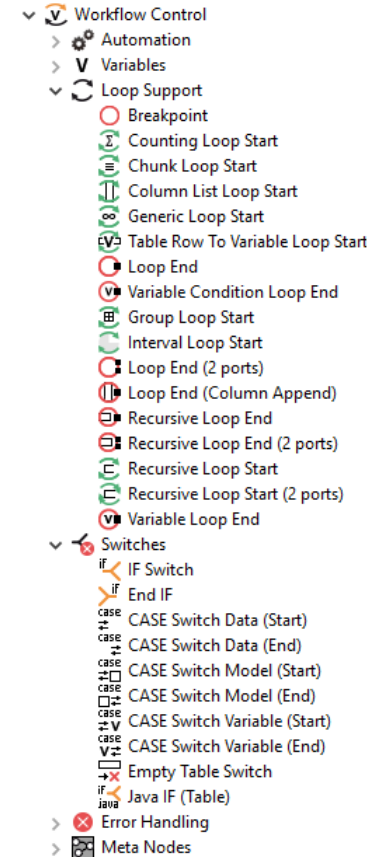
- Handle workflow branches that may fail in execution and you don't know before execution

■ Active Branch Inverter

■ Breakpoint

■ Switches

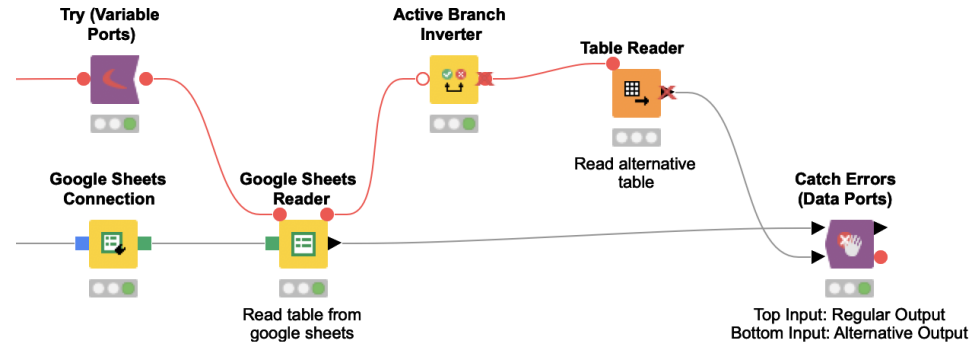
- Direct the path of a workflow by selectively executing one or more workflow branches.



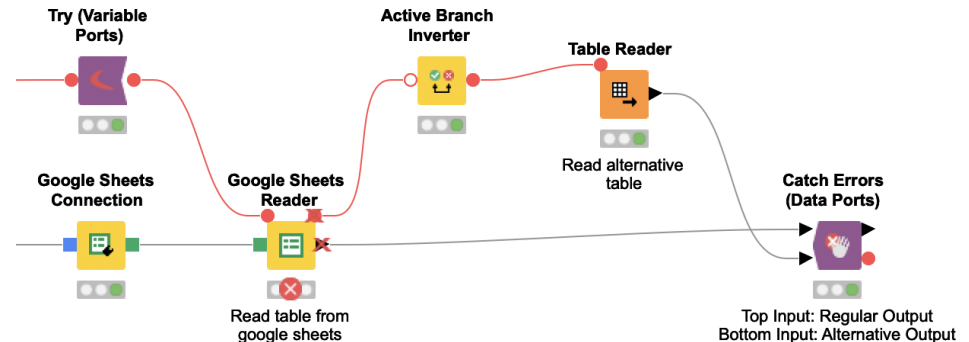
Try-Catch

- A way to catch errors in workflows
- Useful when it is hard to know if a node will execute (for example, when reading from a Google Sheet)
- KNIME tries to execute the nodes, but if it fails will fall back to an alternative branch

Regular Execution



Alternative Execution



Try-Catch

- Try-Catch nodes available for different data types

The original input

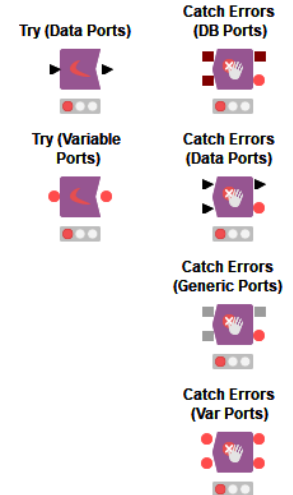
The input to be used when execution on the main branch failed; alternative input

Catch Errors
(Data Ports)

The original input or the alternative input, depending on node failure

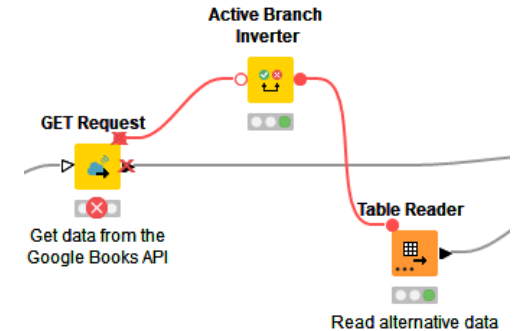
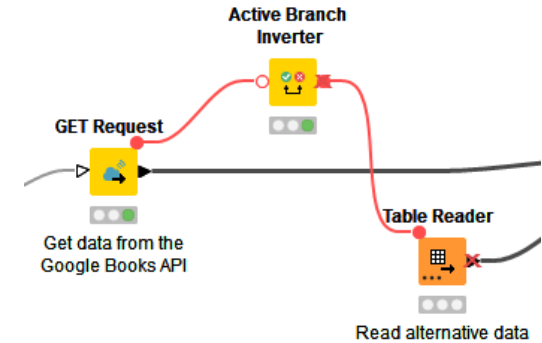
Reasons for Failure (if any)

- Workflow Control
 - Automation
 - Variables
 - Loop Support
 - Switches
- Error Handling
 - Catch Errors (DB Ports)
 - Catch Errors (Data Ports)
 - Catch Errors (Generic Ports)
 - Catch Errors (Var Ports)
 - Try (Data Ports)
 - Try (Variable Ports)



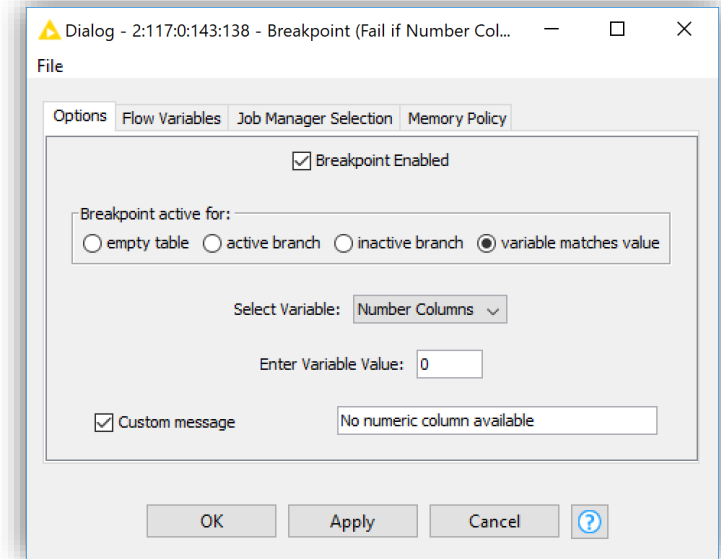
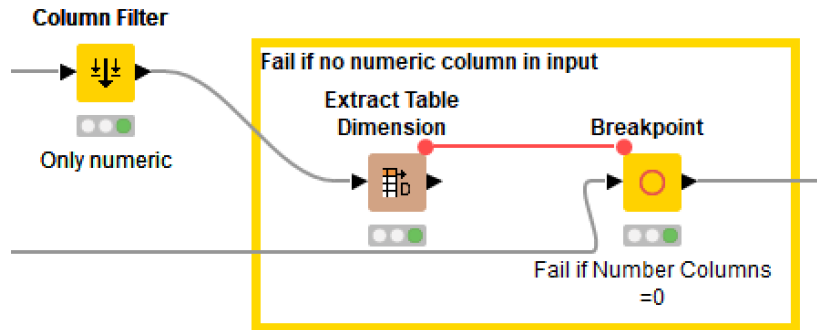
Active Branch Inverter

- Node execution can be triggered via flow variables: if they are connected to upstream nodes by flow variables, they will be automatically be executed
- If the node executes that is connected to the Active Branch Inverter, the alternative branch/node is not active
- If the node that is connected to the Active Branch Inverter does NOT execute, the execution of the alternative branch/node is triggered by flow variable
- like an If-switch for node execution, the condition is the execution of the upstream node



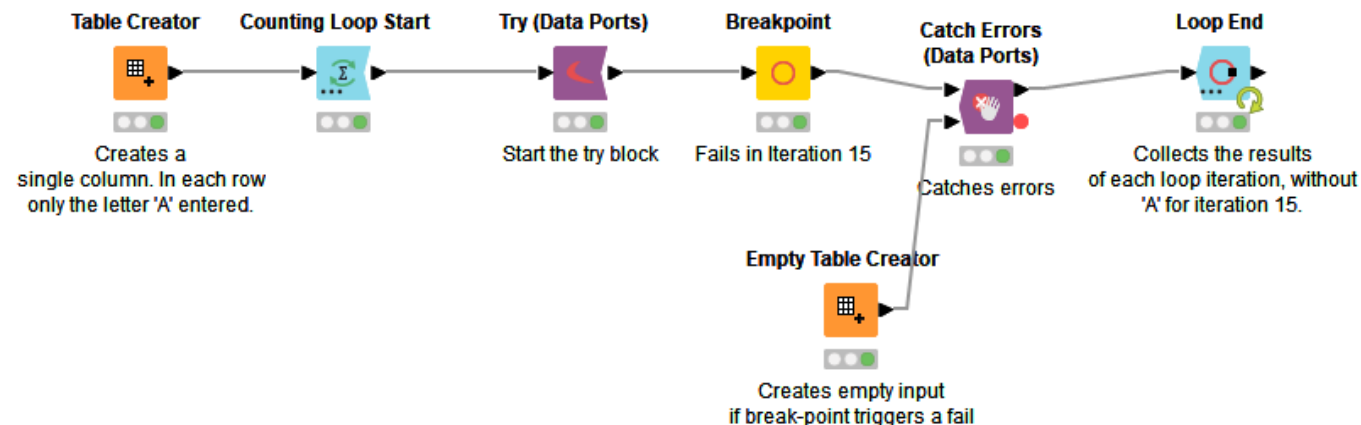
Breakpoint

- Stops execution of a workflow branch
- Useful to stop the execution of a component and provide a custom error message
- Execution stops based on the selected condition:
 - Empty table
 - Active/Inactive branch
 - Flow Variable value



Breakpoint and Try-Catch

- Recover from a Breakpoint using the Try-Catch

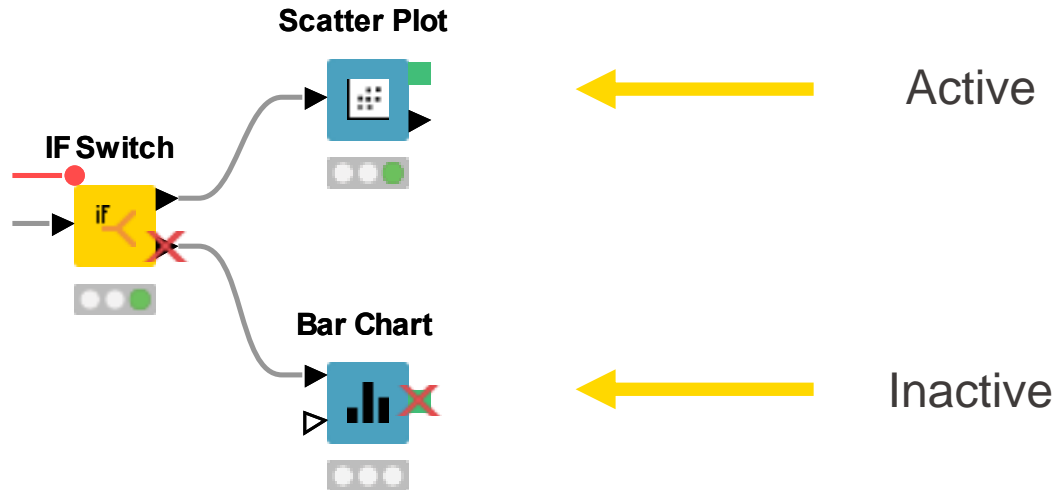


Row ID	Text	Iteration
Row0#0	A	0
Row0#1	A	1
Row0#2	A	2
Row0#3	A	3
Row0#4	A	4
Row0#5	A	5
Row0#6	A	6
Row0#7	A	7
Row0#8	A	8
Row0#9	A	9
Row0#10	A	10
Row0#11	A	11
Row0#12	A	12
Row0#13	A	13
Row0#14	A	14
Row0#15	?	15
Row0#16	A	16
Row0#17	A	17
Row0#18	A	18
Row0#19	A	19

- <https://kni.me/w/yNkGIznDDI8YcVby>

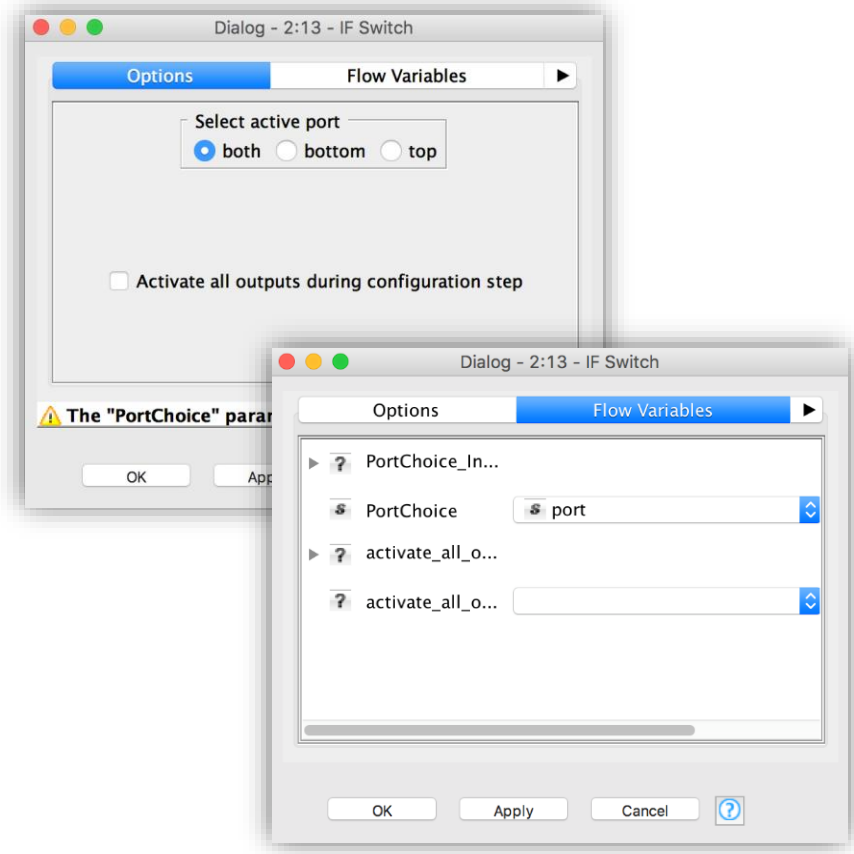
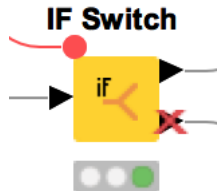
Switches

- A switch allows you to selectively activate branches of a workflow
- Inactive branches are marked with a red x on their output ports. Inactive nodes propagate down stream.



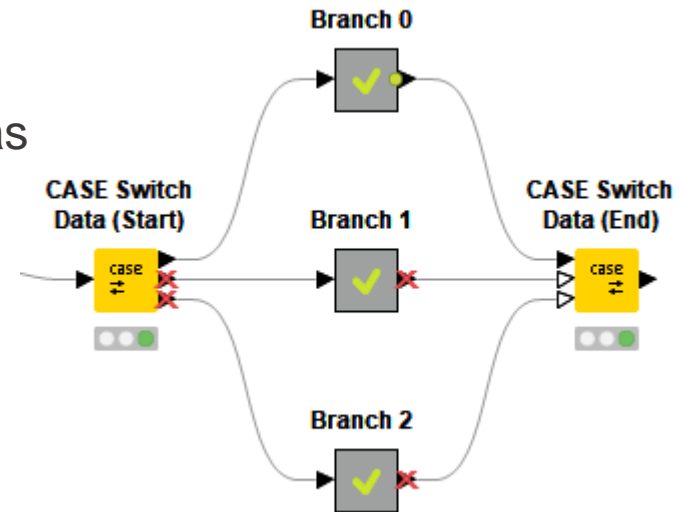
If Switch

- Controls which branches of your workflow are active programmatically
- Controlled with a Flow Variable, setting the value to the literal Strings: “top”, “bottom”, “both”
- May be used in Flow Variables or tables (different nodes)



Case Switch Data

- Similar to If-Switch: Takes data from single input port and passes it to the active output port
- Nodes connected to inactive branches are not executed
- Configure via node dialog, or pass port index as Flow Variable
 - 0, 1, 2 for top, middle, and bottom port
- Case switches also available for Flow Variable and model ports



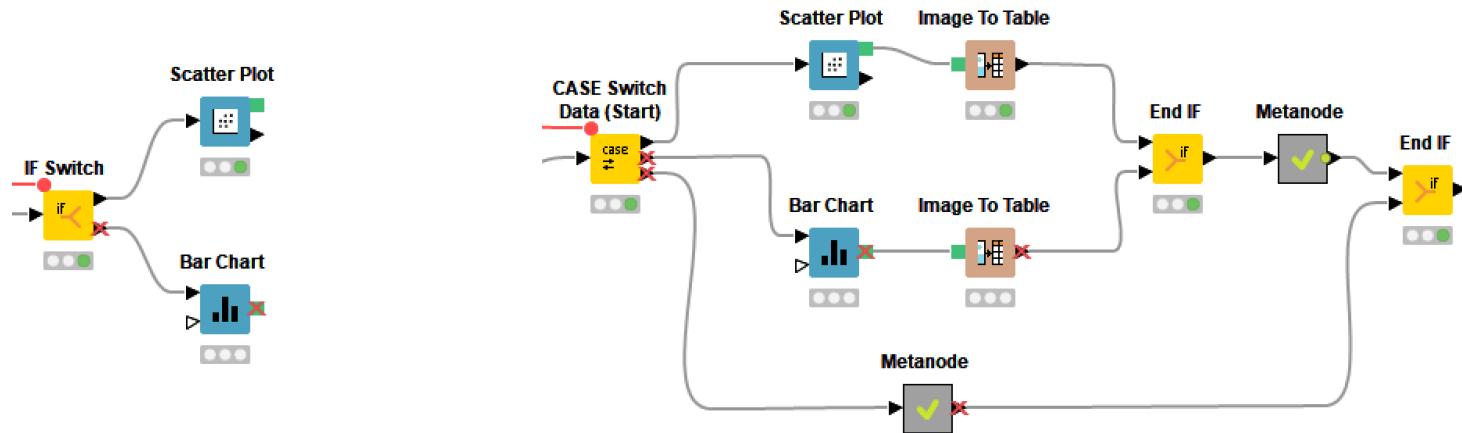
The difference between Loops and Switches

Loops

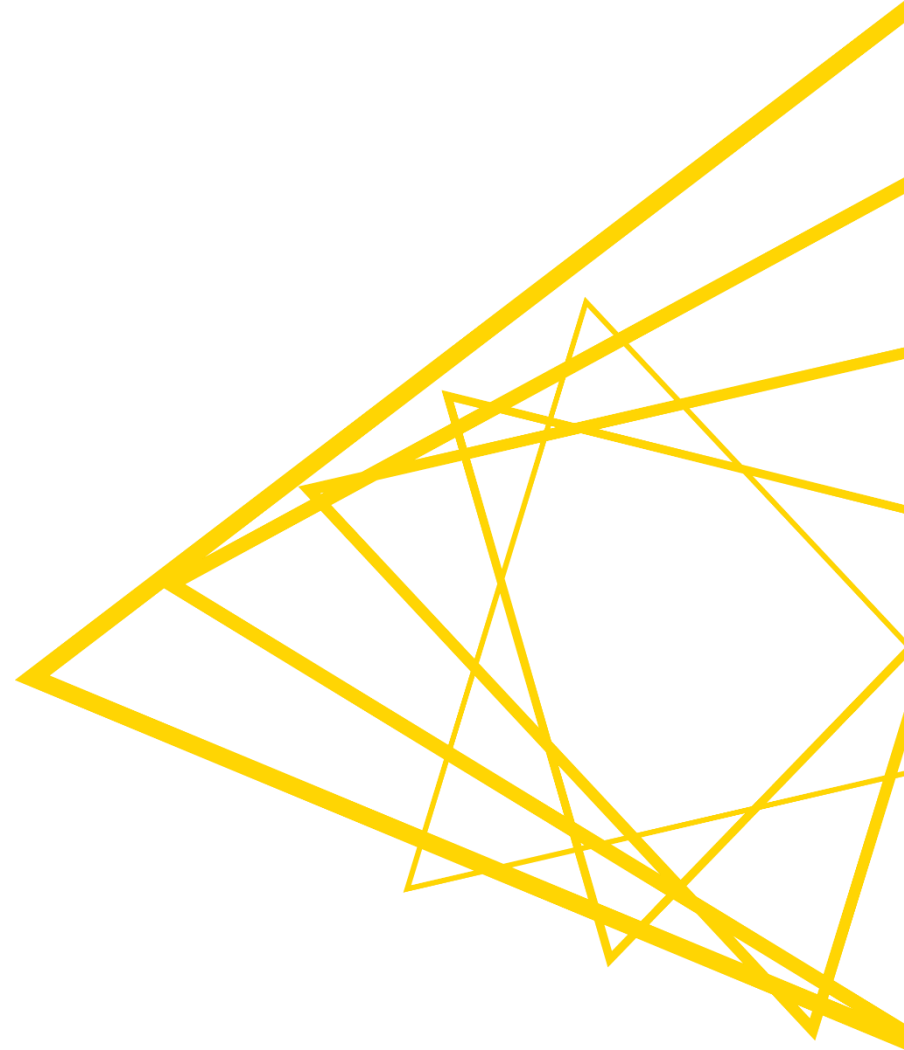
- The Loop Start is connected to the Loop End node, they form a pair.
- A loop iterates over a workflow part.

Switches

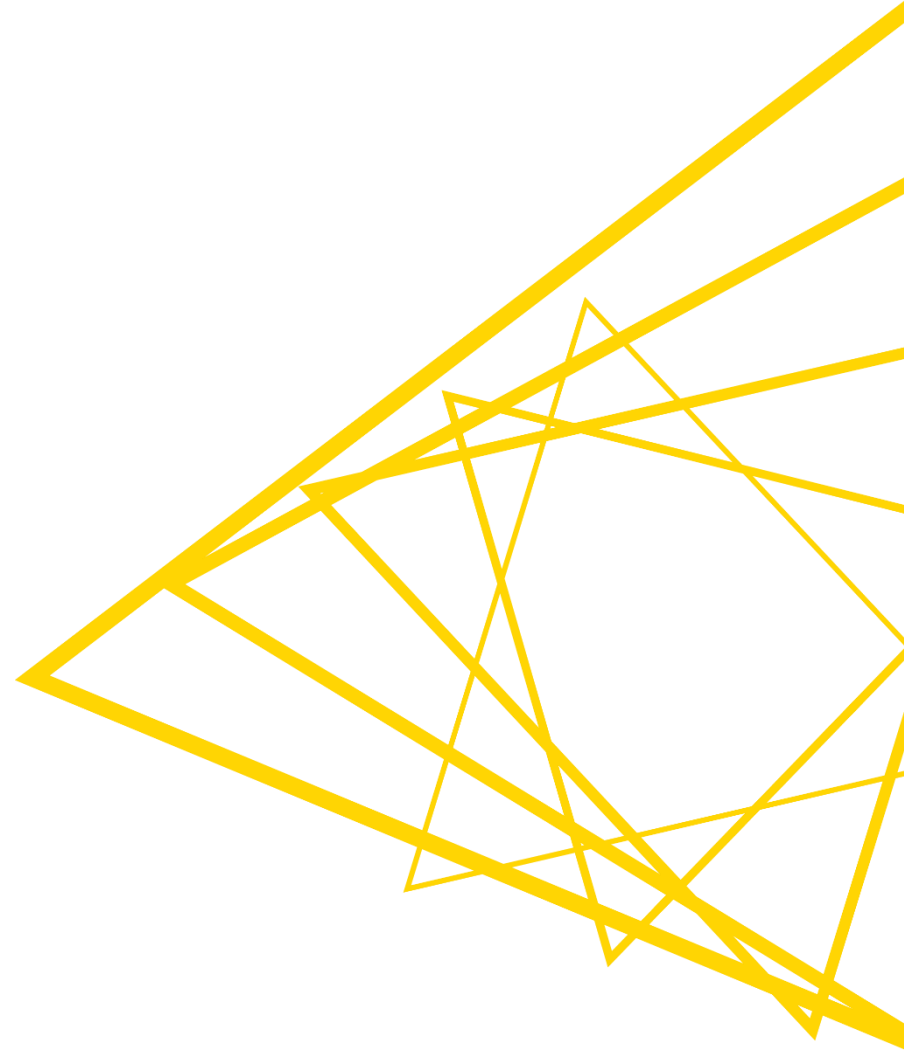
- A Switch Start can be used without a corresponding Switch End. They can also be combined.



Output

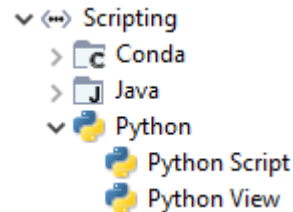


KNIME Python Integration



KNIME Python Integration

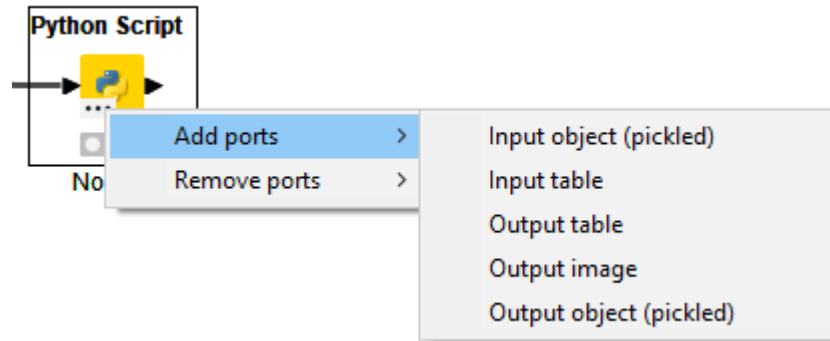
- Install the KNIME Python Integration for the nodes
- Since 4.7 comes with a bundled environment under the hood
- Includes the extensions on the right



```
- beautifulsoup4      # 4.11.1
- cloudpickle         # 2.2.0
- ipython             # 8.8.0
- jedi>=0.18.1        # 0.18.2
- matplotlib-base     # 3.6.2
- markdown            #
- nbformat            # 5.7.1
- nltk                # 3.8.1
- nomkl               # 1.0
- numpy>=1.22         # 1.24.1
- openpyxl            # 3.0.10
- pandas              # 1.5.2
- packaging           # 21.3
- pillow              # 9.4.0
- plotly              # 5.11.0
- py4j                # 0.10.9.7
- pyarrow>=9          # 9.0.0
- python=3.9          # 3.9.15
- python-dateutil     # 2.8.2
- pytz                # 2022.7
- pyyaml              # 6.0
- requests            # 2.28.1
- scikit-learn        # 1.2.0
- scipy               # 1.10.0
- seaborn             # 0.12.2
- statsmodels         # 0.13.5
```

Python Scripting node

- Variable input and output ports
- Click on the three dots to change them



Python Scripting node

- Configuring it opens the code editor
- use the *knime.scripting.iomodule* (imported as *knio* by default) to access the node's input data and populate its output data

The screenshot shows the configuration dialog for the Python Scripting node in KNIME. The dialog has several tabs: Script, Executable Selection, Templates, Flow Variables, Job Manager Selection, and Memory Policy. The 'Script' tab is active, showing a code editor with the following Python code:

```
1 import knime.scripting.io as knio
2
3 # This example script simply outputs the node's input table.
4
5 knio.output_tables[0] = knio.input_tables[0]
6
```

On the left side, there are two panels: 'Input variables' and 'Flow variables'. The 'Input variables' panel shows a tree structure with 'knio.input_tables[0]' expanded, revealing 'column1' and 'column2'. The 'Flow variables' panel shows 'knime.workspace'. On the right side, there is a 'Python workspace' table with columns 'Name', 'Type', and 'Value'. Below it, the 'Output variables' section shows 'knio.output_tables[0]'. At the bottom right, there is a 'Python console' area. The 'Execute script' and 'Execute selected lines' buttons are visible. A status bar at the bottom indicates 'Successfully loaded input data into Python' and a warning message: 'The "python3_command" parameter is controlled by a variable.'.

Input data columns

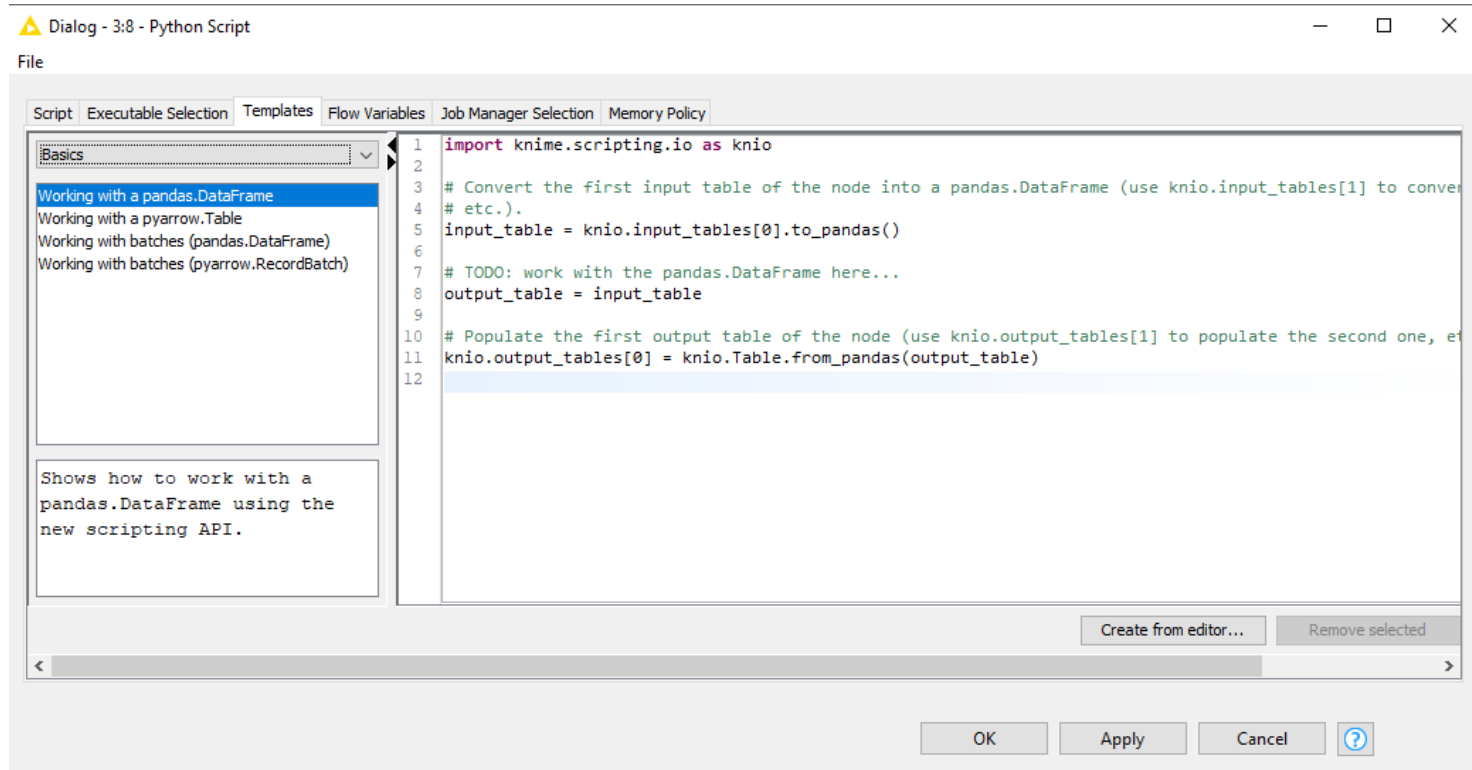
Available flow variables

Python workspace

Python console

Python Scripting node

- Find code templates under the Templates tab



Accessing data

- **tables**
 - `knio.input_tables[i]` and `knio.output_tables[i]`
- **objects**
 - `knio.input_objects[i]` and `knio.output_objects[i]`
- **images**
 - `knio.output_images[i]`
 - must be either a string describing an SVG image or a byte array encoding a PNG image
- **flow variables**
 - `knio.flow_variables['name_of_flow_variable']`
- where `i` is the index of the corresponding table/object/image (starting with 0)

KNIME Tables to Python

- Pandas data frames

- To Pandas df:

- `df = knio.input_tables[0].to_pandas()`

- From Pandas df:

- `knio.output_tables[0] = knio.Table.from_pandas(df)`

- PyArrow tables

- To PyArrow table:

- `table = knio.input_tables[0].to_pyarrow()`

- From PyArrow table

- `knio.output_tables[0] = knio.Table.from_pyarrow(table)`

Porting Scripts from the Legacy Nodes

```
import knime.scripting.io as knio
input_table_1 = knio.input_tables[0].to_pandas()

# the script from the legacy nodes goes here

knio.output_tables[0] = knio.Table.from_pandas(output_table_1)
```

Jupyter Notebook

```
# Path to the folder containing the notebook, e.g. the folder 'data' contained
# in my workflow folder
notebook_directory = "knime://knime.workflow/data/"

# Filename of the notebook
notebook_name = "sum_table.ipynb"

# Load the notebook as a Python module
import knime.scripting.jupyter as knupyter
my_notebook = knupyter.load_notebook(notebook_directory, notebook_name)

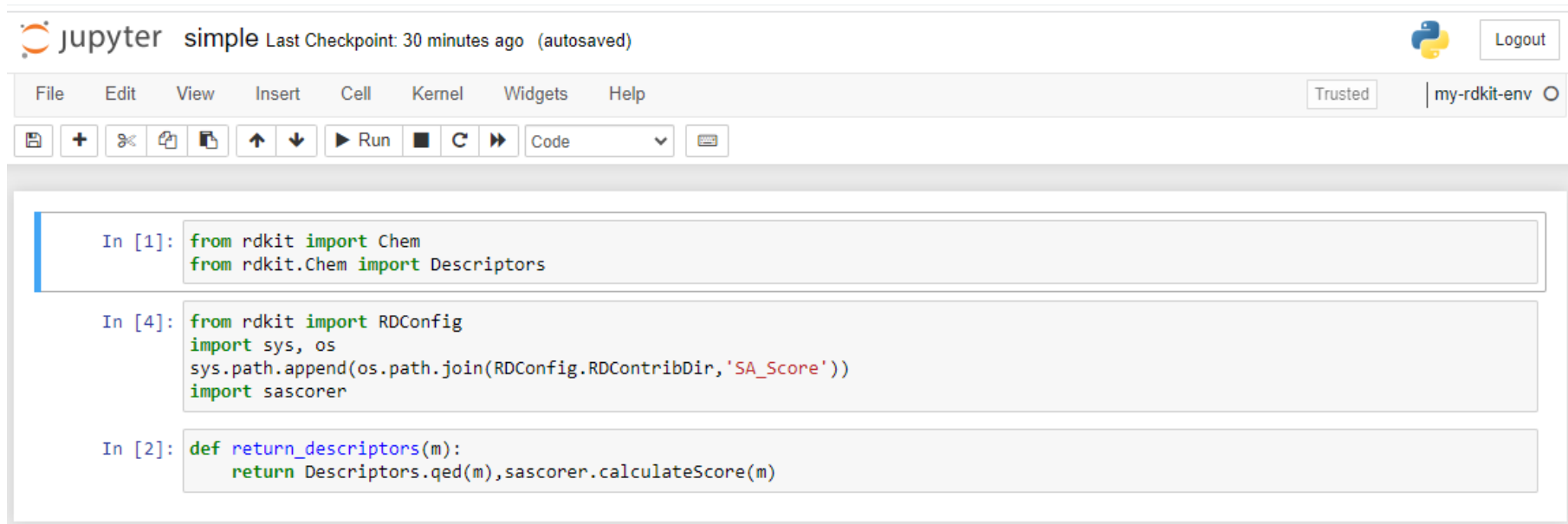
# Print its textual contents
knupyter.print_notebook(notebook_directory, notebook_name)

# Call a function 'sum_each_row' defined in the notebook
output_table = my_notebook.sum_each_row(input_table)
```

https://docs.knime.com/latest/python_installation_guide/index.html#jupyter-notebooks

Jupyter Notebook

- Jupyter notebook of the example



The screenshot displays a Jupyter Notebook interface. At the top, the header shows the Jupyter logo, the text "simple", and "Last Checkpoint: 30 minutes ago (autosaved)". On the right, there is a Python logo and a "Logout" button. Below the header is a menu bar with options: File, Edit, View, Insert, Cell, Kernel, Widgets, and Help. To the right of the menu bar, it says "Trusted" and "my-rdkit-env". Below the menu bar is a toolbar with icons for saving, adding, opening, and closing files, as well as navigation and execution controls. The main area contains three code cells:

```
In [1]: from rdkit import Chem
        from rdkit.Chem import Descriptors
```

```
In [4]: from rdkit import RDConfig
        import sys, os
        sys.path.append(os.path.join(RDConfig.RDContribDir, 'SA_Score'))
        import sascore
```

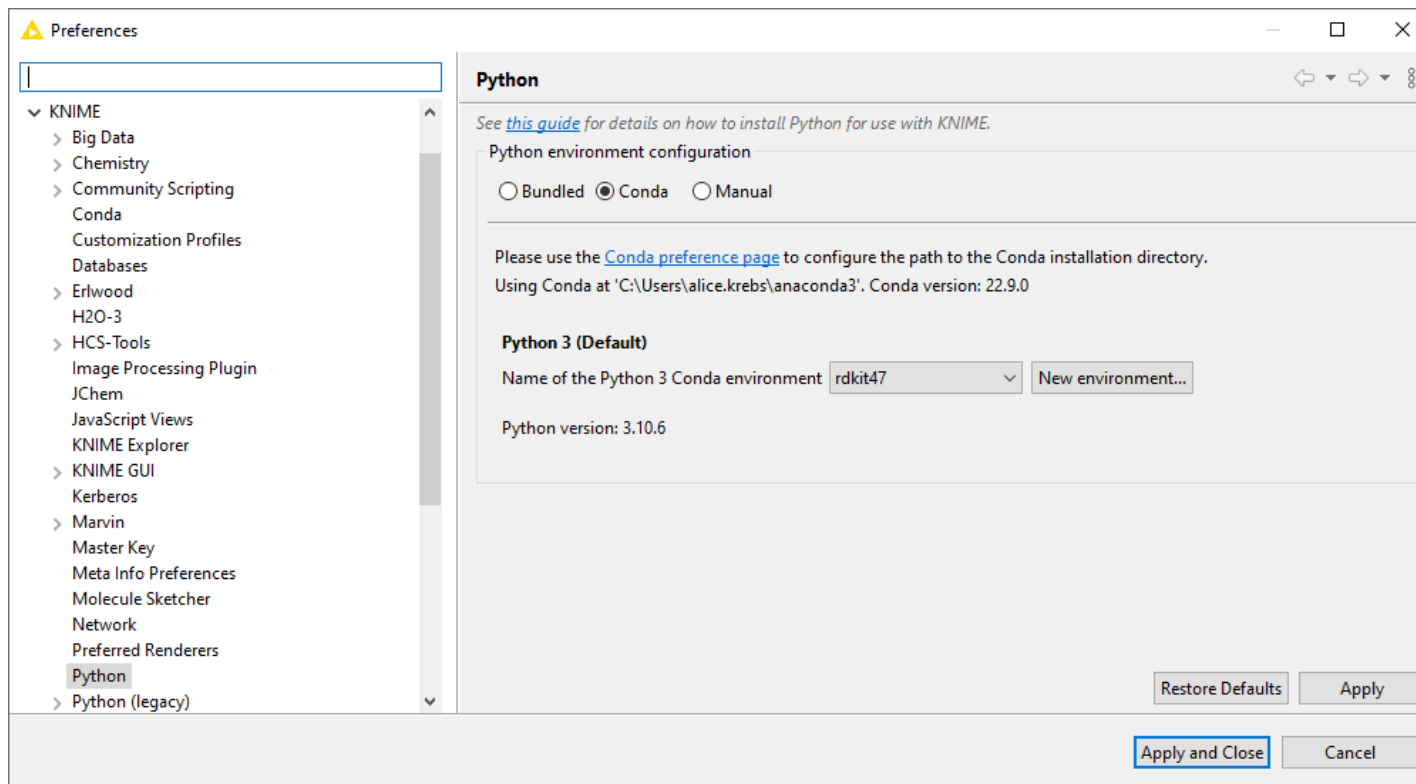
```
In [2]: def return_descriptors(m):
        return Descriptors qed(m), sascore.calculateScore(m)
```

KNIME packages for environments

- *knime-python-base*
 - contains the basic packages which are always needed
- *knime-python-scripting*
 - contains *knime-python-base* and installs additionally the packages used in the Python Script node

Pointing to Environments

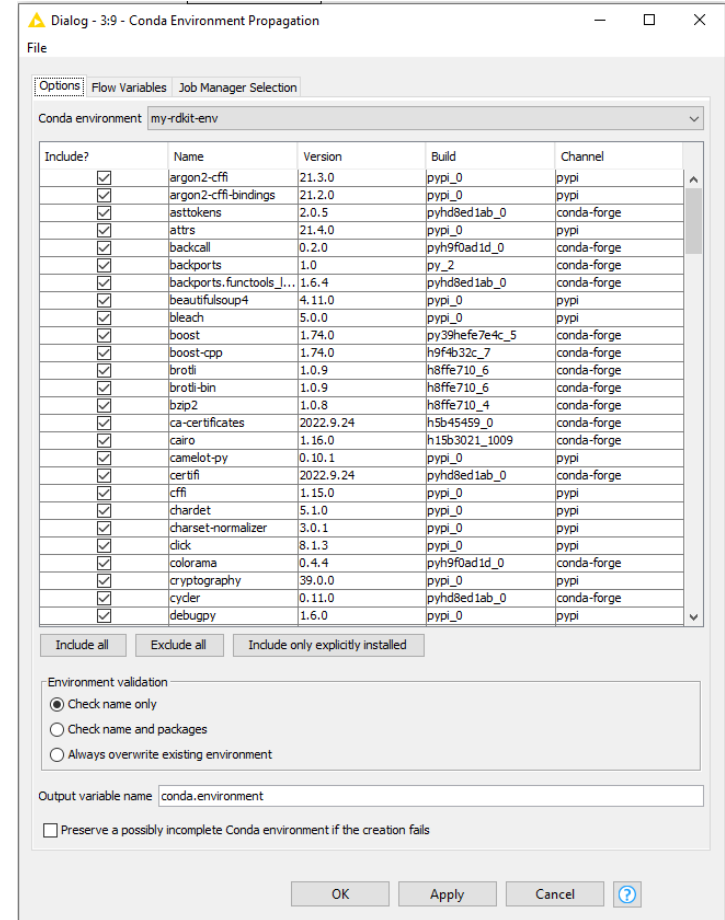
- *KNIME > File > Preferences*



Conda Environment Propagation Node

- Configure custom Python environments to be used
- Bundle environments together with workflows for easier re-execution and workflow sharing
- Define Conda and custom Python environments under *Preferences > KNIME > Python* first

Conda Environment Propagation




Conda Environment Propagation Node

- *Tip:* choose only explicitly installed packages to avoid conflicts when using the workflow on different operating systems

<input checked="" type="checkbox"/>	numpy-base	1.20.2	py37hc2deb75_0	pkgs/main
<input checked="" type="checkbox"/>	openssl	1.1.1k	h2bbff1b_0	pkgs/main
<input checked="" type="checkbox"/>	pandas	1.2.5	py37hd77b12b_0	pkgs/main
<input checked="" type="checkbox"/>	pip	21.1.3	py37haa95532_0	pkgs/main
<input checked="" type="checkbox"/>	python	3.7.10	h6244533_0	pkgs/main
<input checked="" type="checkbox"/>	python-dateutil	2.8.1	pyhd3eb1b0_0	pkgs/main
<input checked="" type="checkbox"/>	pytz	2021.1	pyhd3eb1b0_0	pkgs/main
<input checked="" type="checkbox"/>	setuptools	52.0.0	py37haa95532_0	pkgs/main
<input checked="" type="checkbox"/>	six	1.16.0	pyhd3eb1b0_0	pkgs/main
<input checked="" type="checkbox"/>	sqlite	3.36.0	h2bbff1b_0	pkgs/main
<input checked="" type="checkbox"/>	vc	14.2	h21ff451_1	pkgs/main
<input checked="" type="checkbox"/>	vs2015_runtime	14.27.29016	h5e58377_2	pkgs/main
<input checked="" type="checkbox"/>	wheel	0.36.2	pyhd3eb1b0_0	pkgs/main
<input checked="" type="checkbox"/>	wincertstore	0.2	py37_0	pkgs/main

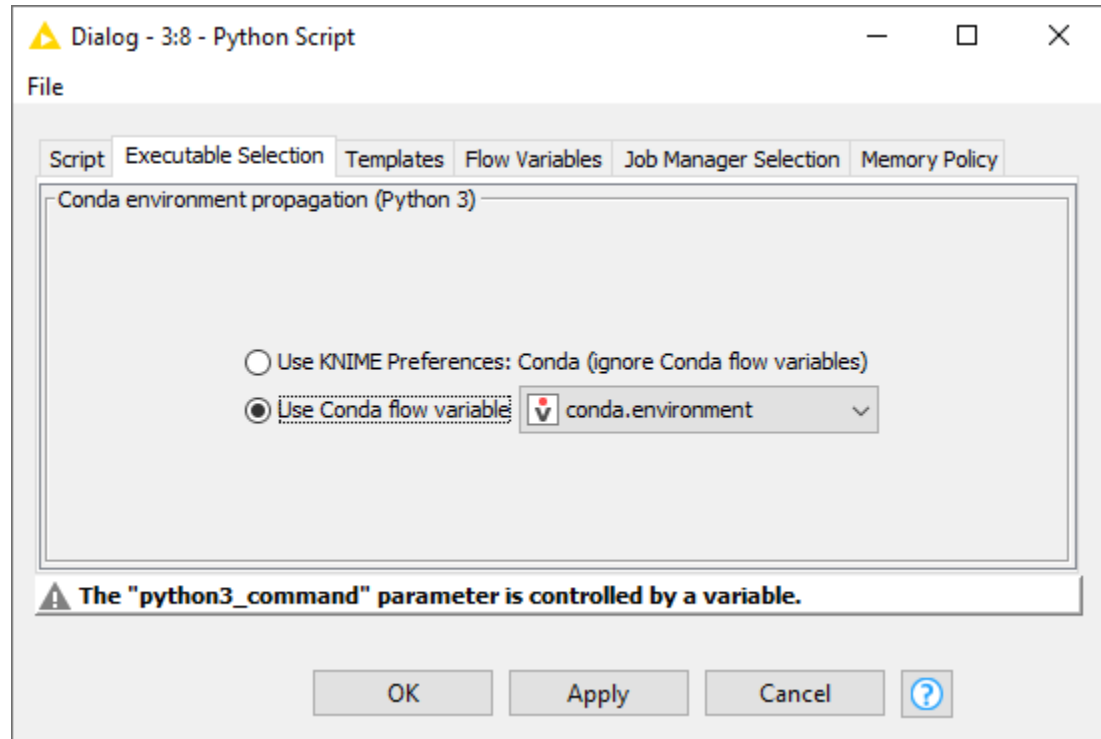
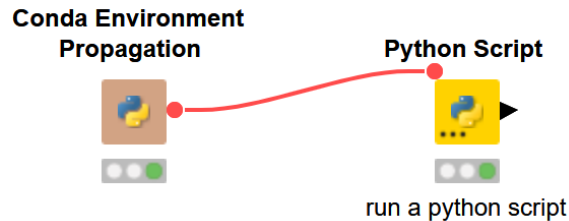
Environment validation
☐ Check name only
☒ Check name and packages
☐ Always overwrite existing environment

Output variable name



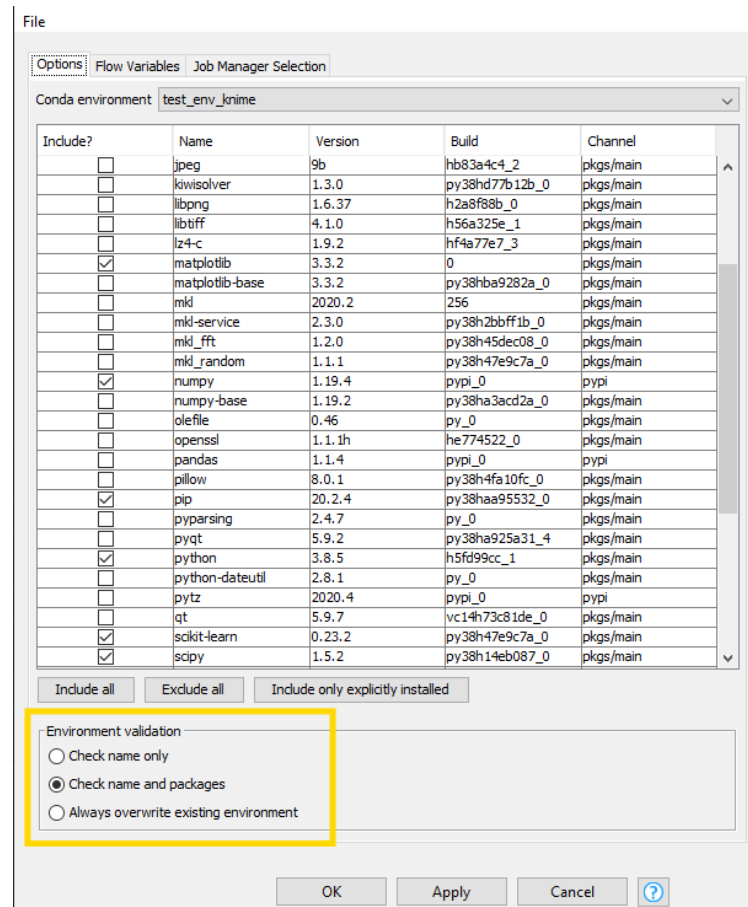
Define the Environment to be used

- The Conda Environment Propagation node allows using different envs in different nodes
- Set the flow variable in the scripting nodes



Establish the defined env on the target machine

- Conda needs to be set up and configured in the preferences
- the node will check whether a local Conda environment exists that matches its configured environment
- 3 different types of validation:
 - *Check name only*: env with the same name
 - *Check name and packages*: check both name and requested packages
 - *Always overwrite existing environment*: disregard the existence of an equal environment on the target machine and complete recreation



Chemistry in Python scripting

<https://forum.knime.com/t/announcing-the-interoperability-between-rdkit-and-python-in-knime-4-7/60416>

- Converting between molecule types in Python
- Import the library *knime.types.chemistry*
 - `import knime.types.chemistry as ktchem`
 - `ktchem.SdfValue`
 - `ktchem.RxnValue`
 - `ktchem.SmilesValue`
 - `ktchem.SmartsValue`

Chemistry in Python

AdapterValue - class

AdapterValueFactory - class

CDXMLValue - class

CDXMLValueFactory - class

CMLAdapterValue - class

CMLAdapterValueFactory - class

CMLValue - class

CMLValueFactory - class

CtabValue - class

CtabValueFactory - class

HELMAdapterValue - class

HELMAdapterValue - class

HELMAdapterValueFactory - class

HELMValue - class

HELMValueFactory - class

InchiAdapterValue - class

InchiAdapterValueFactory - class

InchiValue - class

InchiValueFactory - class

kt - module

Mol2AdapterValue - class

Mol2AdapterValueFactory - class

Mol2AdapterValue - class

Mol2AdapterValueFactory - class

Mol2Value - class

Mol2ValueFactory - class

MolAdapterValue - class

MolAdapterValueFactory - class

MolValue - class

MolValueFactory - class

RxnAdapterValue - class

RxnAdapterValueFactory - class

RxnValue - class

RxnAdapterValue - class

RxnAdapterValueFactory - class

RxnValue - class

RxnValueFactory - class

SdfAdapterValue - class

SdfAdapterValueFactory - class

SdfValue - class

SdfValueFactory - class

SInValue - class

SInValue - class

SInValueFactory - class

SmartsAdapterValue - class

SmartsAdapterValueFactory - class

SmartsValue - class

SmartsValueFactory - class

SmilesAdapterValue - class

SmilesAdapterValueFactory - class

SmilesValue - class

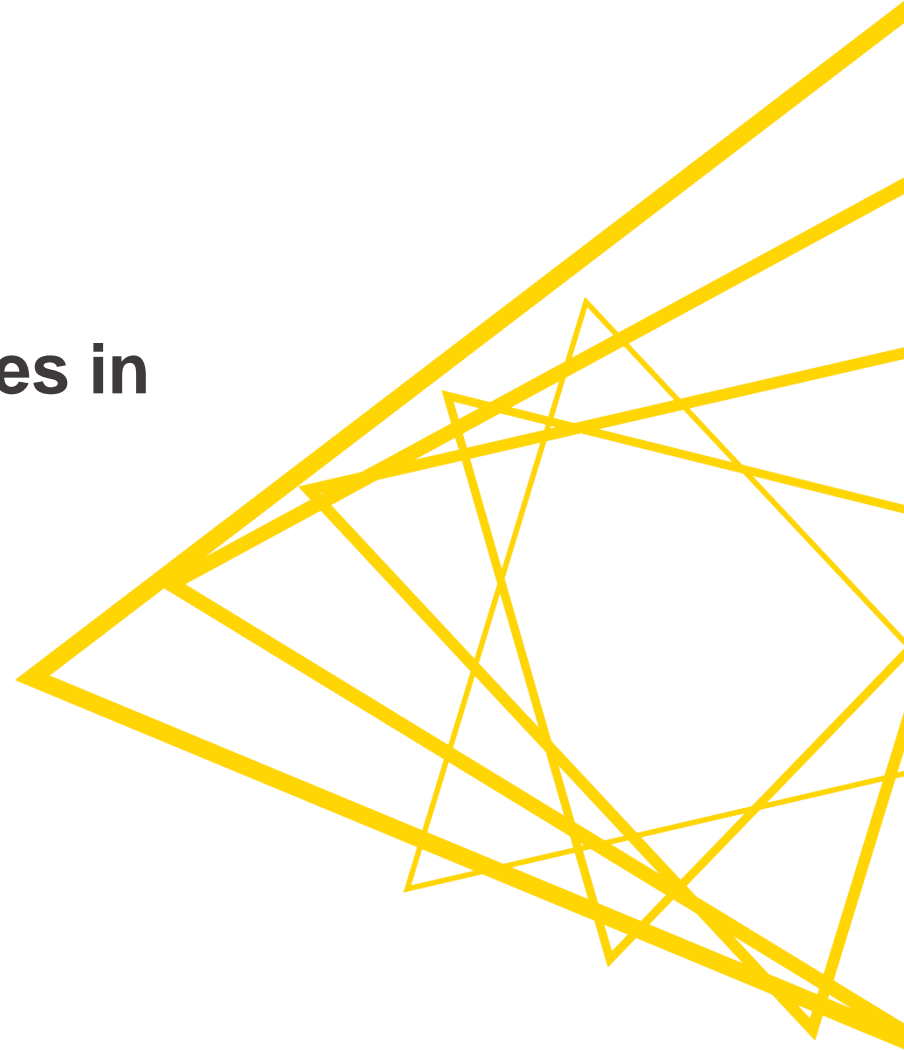
SmilesValueFactory - class

StringBasedValueFactory - class

Useful links

- KNIME Python Integration Guide:
 - https://docs.knime.com/latest/python_installation_guide/index.html
- Blog post “KNIME and Jupyter”:
 - <https://www.knime.com/blog/knime-and-jupyter>
- Blog post “Manage Your Python Environments with Conda and KNIME”:
 - <https://www.knime.com/blog/how-to-manage-python-environments-conda-and-knime>
- KNIME Python Node Extension guide:
 - https://docs.knime.com/2022-06/pure_python_node_extensions_guide/index.html#introduction
- Find the documentation for the new API:
 - <https://knime-python.readthedocs.io/en/stable/>
- Get in touch with the developers at KNIME and provide feedback:
 - <https://forum.knime.com/c/knime-development/>
 - please add the tag “Python”
- Blog post “4 Steps for your Python Team to Develop KNIME Nodes”:
 - <https://www.knime.com/blog/4-steps-for-your-python-team-to-develop-knime-nodes>

**Create your own KNIME nodes in
Python**



Needed extension

- Install the Python Extension Development (Labs)

Extension

KNIME Python Extension Development (Labs)

v 5.1.0

This extension allows the development and integration of nodes written in Python.

KNIME packages for environments

- *knime-python-base*
 - contains the basic packages which are always needed
- *knime-extension*
 - contains the node development API
- Requires -c knime -c conda-forge

Configurations

- **config.yml:** information that binds your extension and the corresponding Python environment with KNIME Analytics Platform

```
org.knime.rdkit_extension: # {group_id}.{name} from the knime.yml
src: C:\knime-python\basic-default\tutorial_extension # Path to folder containing the extension files
conda_env_path: C:\Users\alice.krebs\anaconda3\envs\rdkit47 # Path to the Python environment to use
debug_mode: true # Optional line, if set to true, it will always use the latest changes of execute/configure, when that method is used within
```

- **my_conda_env.yml:**

```
name: rdkit_node_env
channels:
  - knime
  - conda-forge
dependencies:
  - packaging
  - python=3.9
  - knime-extension
  - knime-python-base
  - rdkit
```

```
.
├── tutorial_extension
│   ├── icon.png
│   ├── knime.yml
│   ├── LICENSE.TXT
│   └── my_extension.py
├── config.yml
├── my_conda_env.yml
├── Example_with_Python_node.knwf
└── README.md
```

- **knime.yml:** important metadata about your extension

```
name: rdkit_extension # Will be concatenated with the group_id to an ID
author: Alice
env_yaml_path: C:\knime-python\basic-default\my_conda_env.yml # This is necessary for bundling, but not needed during development
extension_module: new_rdkit_nodes # The .py Python module containing the nodes of your extension
description: RDKit Extension # Human readable bundle name / description
long_description: This extension has new RDKit nodes written in Python.
group_id: org.knime
version: 0.1.0 # Version of this Python node extension
vendor: KNIME AG, Zurich, Switzerland
license_file: LICENSE.TXT # Best practice: put your LICENSE.TXT next to the knime.yml; otherwise you would need to change to path/to/LICENSE.txt
```

Configurations

- Point KNIME AP to the *config.yml* in the knime.ini file, in order to allow it to use our extension and its Python environment

```
--add-opens=java.security.jgss/sun.security.jgss.krb5=ALL-UNNAMED
--add-exports=java.security.jgss/sun.security.jgss=ALL-UNNAMED
--add-exports=java.security.jgss/sun.security.jgss.spi=ALL-UNNAMED
--add-exports=java.security.jgss/sun.security.krb5.internal=ALL-UNNAMED
--add-exports=java.security.jgss/sun.security.krb5=ALL-UNNAMED
-DXXchromium.remote_debugging_port=8888
-Dknime.python.extension.config=C:/knime-python/basic-default/config.yml
--add-opens=java.base/sun.security.ssl=ALL-UNNAMED
--add-opens=java.base/sun.security.util=ALL-UNNAMED
-server
-Dsun.java2d.d3d=false
-Dosgi.classloader.lock=classname
```

- Add the following line to the ini file:
 - *-Dknime.python.extension.config=<path/to/your/config.yml>*

Simple Example – Python code for KNIME node

C: > knime-python > basic-default > basic > tutorial_extension > MolFormula.py > ...

```
1  import logging
2  import knime_extension as knext
3  from rdkit import Chem
4  import rdkit.Chem.rdMolDescriptors
5  import pandas as pd
6  LOGGER = logging.getLogger(__name__)
7
8  @knext.node(name="Get Molecular Formula", node_type=knext.NodeType.LEARNER, icon_path="icon.png", category="/")
9  @knext.input_table(name="Input Data", description="Input table containing SMILES")
10 @knext.output_table(name="Output Data", description="Input table appended with a column containing the molecular formula")
11 class MolFormula:
12     """
13     This node returns the molecular formula from SMILES in an appended column.
14     """
15
16     molecule_column = knext.ColumnParameter(label="Select the column containing SMILES", description="Choose the column containing SMILES")
17
18     def configure(self, configure_context, input_schema_1):
19         return input_schema_1.append(knext.Column(knext.string(), "MolFormula"))
20
21
22     def execute(self, exec_context, input_1):
23         input_1_pandas = input_1.to_pandas()
24
25         mols = []
26         for smi in input_1_pandas[self.molecule_column]:
27             mols.append(Chem.MolFromSmiles(smi))
28
29         MolFormula = []
30         for x in mols:
31             MolFormula.append(Chem.rdMolDescriptors.CalcMolFormula(x))
32
33         input_1_pandas['MolFormula'] = MolFormula
34         return knext.Table.from_pandas(input_1_pandas)
35
```

Import libraries

Manipulate data

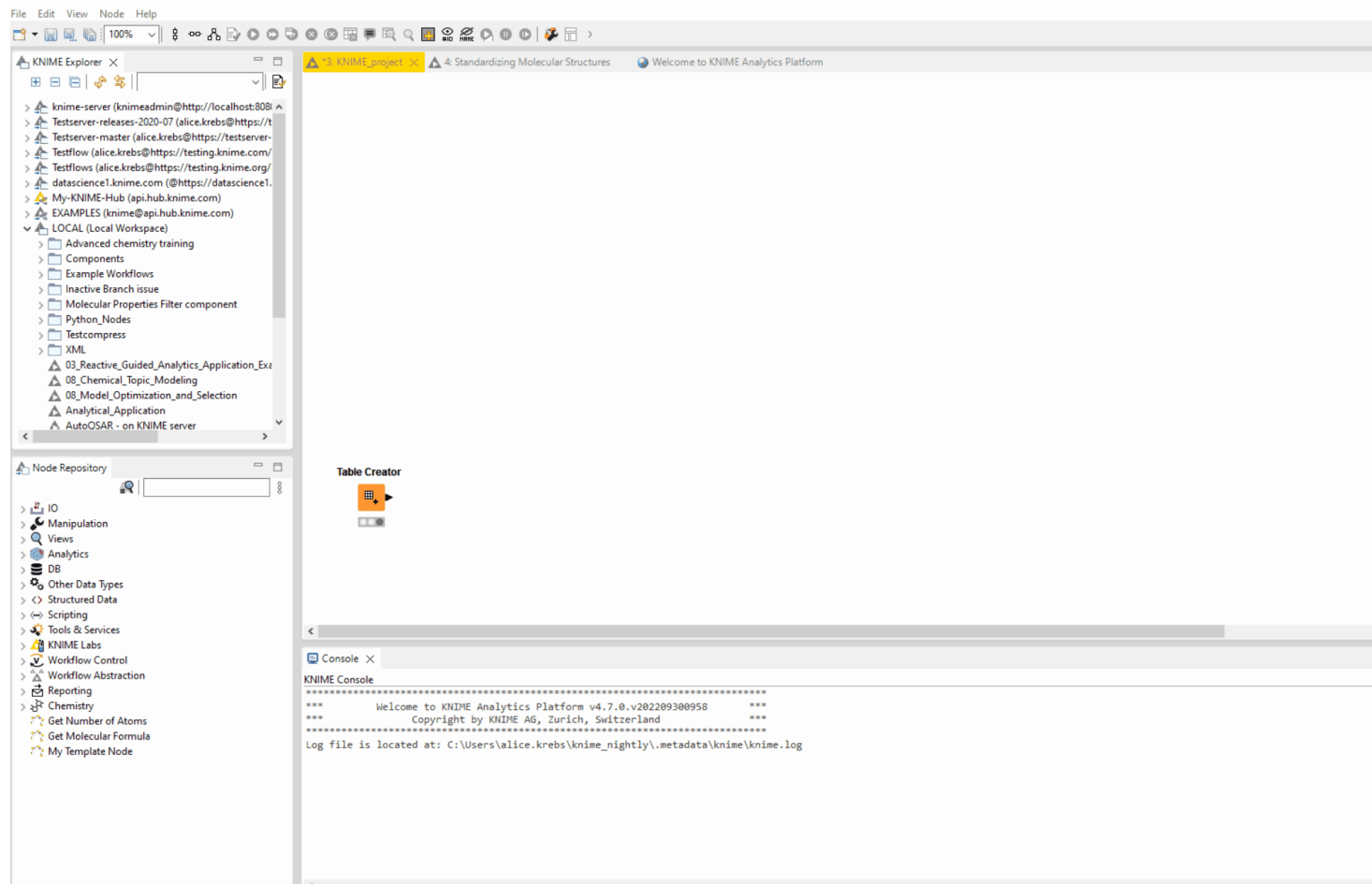
Simple Example – Python code for KNIME node

```
C: > knime-python > basic-default > basic > tutorial_extension > MolFormula.py > ...
1  import logging
2  import knime_extension as knext
3  from rdkit import Chem
4  import rdkit.Chem.rdMolDescriptors
5  import pandas as pd
6  LOGGER = logging.getLogger(__name__)
7
8  @knext.node(name="Get Molecular Formula", node_type=knext.NodeType.LEARNER, icon_path="icon.png", category="/")
9  @knext.input_table(name="Input Data", description="Input table containing SMILES")
10 @knext.output_table(name="Output Data", description="Input table appended with a column containing the molecular formula")
11 class MolFormula:
12     """
13     This node returns the molecular formula from SMILES in an appended column.
14     """
15
16     molecule_column = knext.ColumnParameter(label="Select the column containing SMILES", description="Choose the column containing SMILES")
17
18     def configure(self, configure_context, input_schema_1):
19         return input_schema_1.append(knext.Column(knext.string(), "MolFormula"))
20
21
22     def execute(self, exec_context, input_1):
23         input_1_pandas = input_1.to_pandas()
24
25         mols = []
26         for smi in input_1_pandas[self.molecule_column]:
27             mols.append(Chem.MolFromSmiles(smi))
28
29         MolFormula = []
30         for x in mols:
31             MolFormula.append(Chem.rdMolDescriptors.CalcMolFormula(x))
32
33         input_1_pandas['MolFormula'] = MolFormula
34         return knext.Table.from_pandas(input_1_pandas)
35
```

Define input and output ports

Define node dialogue, input and output table spec

Simple Example – Resulting node



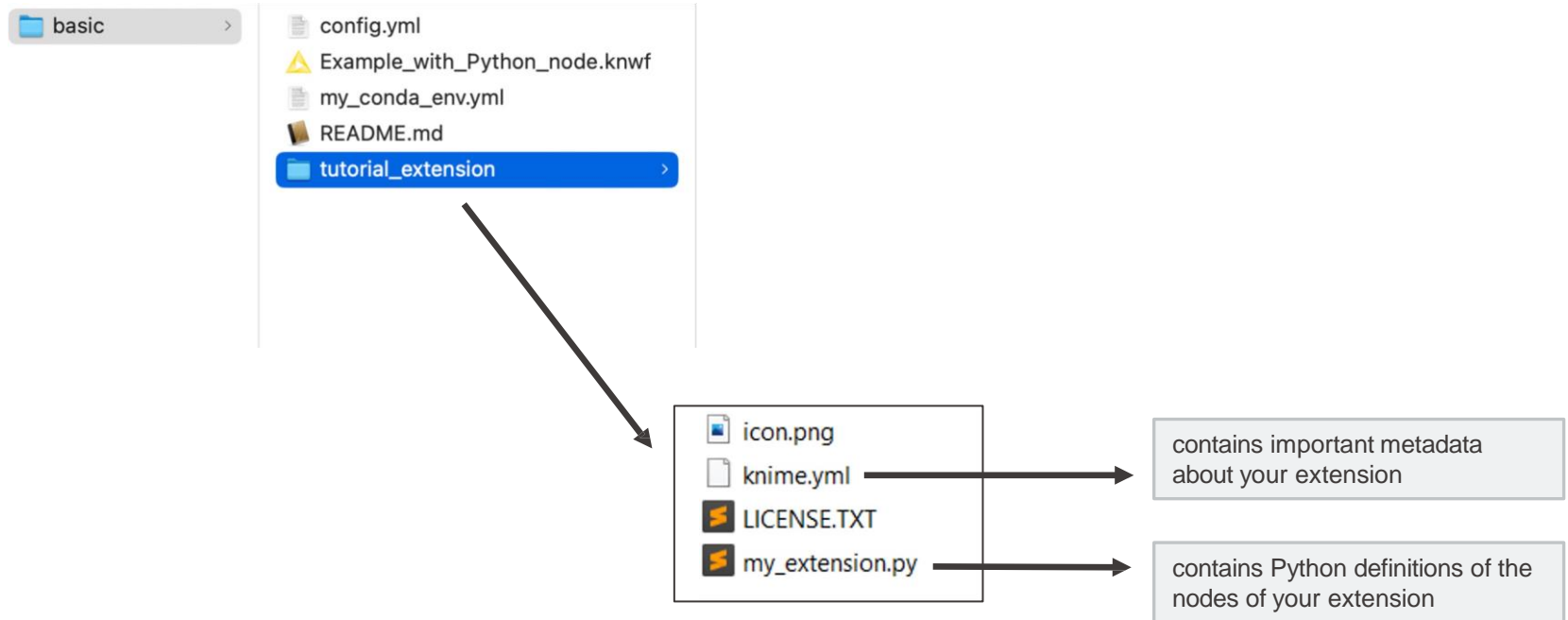
3 steps to create a custom Python-based node

- Step 1: **Install** necessities
- Step 2: **Establish** connection between KAP and external Python script(s)
- Step 3: **Write** Python-KNIME specific code

Step 1: Install Necessities

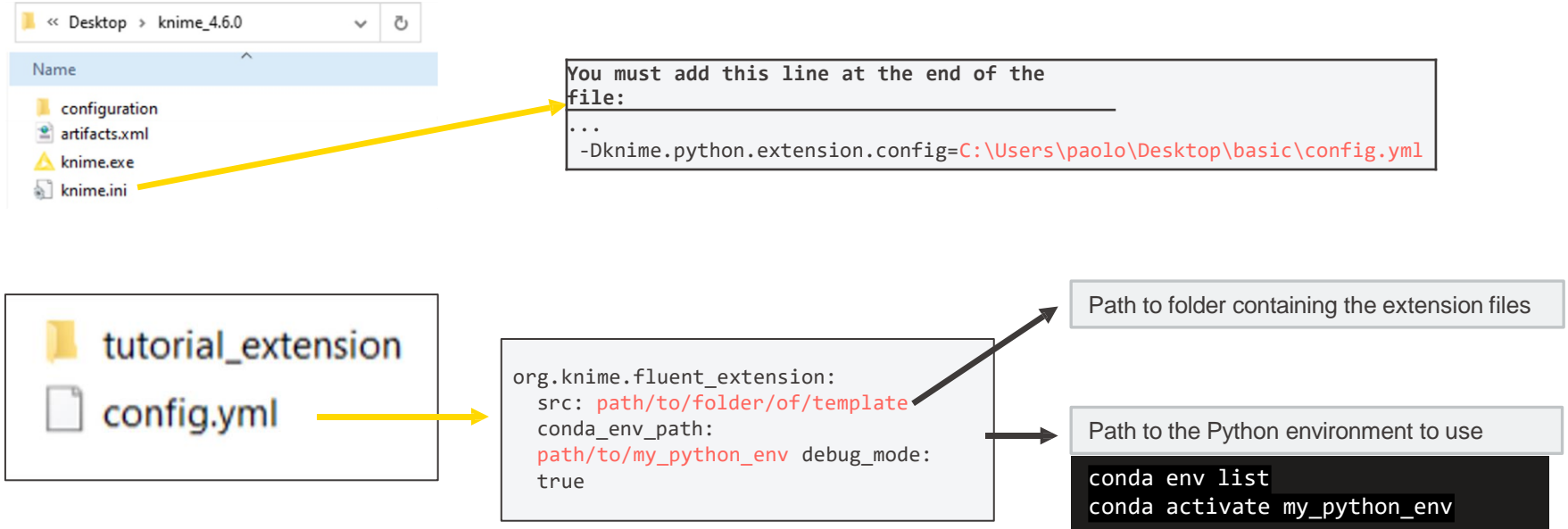
- Install KNIME
 - Install KNIME Analytics Platform version **4.6.0** or higher
- Install Extension
 - File > Install KNIME Extensions > “*KNIME Python Node Development Extension (Labs)*”
- Install (mini)conda
 - docs.conda.io/en/latest/miniconda.html
 - ```
conda create -n my_python_env python=3.9 packaging knime-python-base knime-extension -c knime -c conda-forge
```
- Download the “basic” folder from [KNIME Docs](#)

# “Basic” Folder Structure



# Step 2: Establish Connection

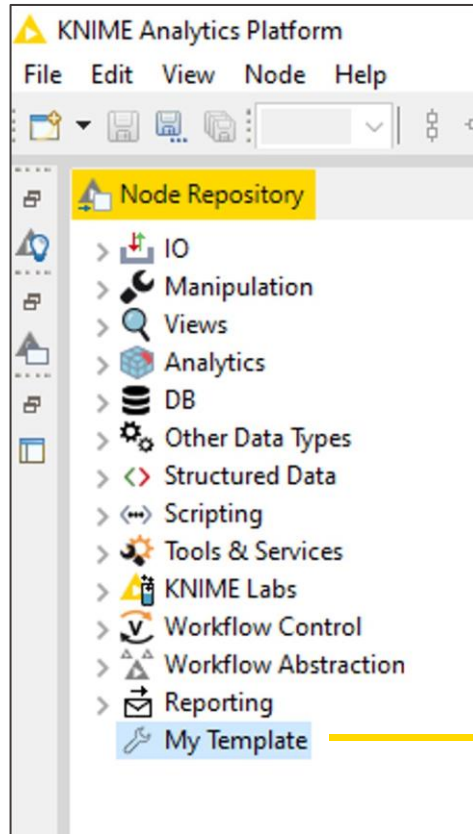
You will change 2 files: knime.ini and config.yml



## EXAMPLE:

```
org.knime.fluent_extension:
 src: C:\Users\paolo\Desktop\basic\tutorial_extension
 conda_env_path:
 C:\Users\paolo\miniconda3\envs\my_python_env debug_mode:
 true
```

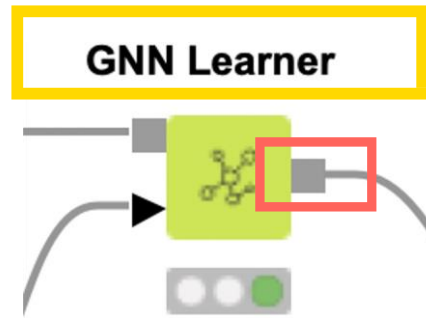
# Did it work so far? Start KNIME and See!



A new node should appear in your node repository!

## Step 3: Write Code (node aesthetics)

```
@knext.node(name="GNN Learner", node_type=knext.NodeType.LEARNER, icon_path="icon.png", category="/")
@knext.input_binary("Full Graph", "Pickled Graph", "org.knime.torch.graphcreator")
@knext.input_table(name="Train Set",
 description="A table that contains nodes need to be masked and passed downstream.")
```



```
@knext.output_binary(
 name="Model",
 description="Pytorch Geometric Model",
 id="org.knime.torch.learner",
)
class GNNLearner:
```



## Step 3: Write Code (dialog)

```
class GNNLearner:
 hidden_channels = knext.IntParameter("Hidden Channels", "The number of hidden channels desired.", 1)
 number_of_hidden_layers = knext.IntParameter("Hidden Layers", "The number of hidden layers desired."
 learning_rate = knext.DoubleParameter("Learning Rate", "The learning rate to use in the optimizer",
 epochs = knext.IntParameter("Epochs", "The number of epochs to train for.", 1)
```

Dialog - 3:2333 - GNN Learner

Hidden Channels ⓘ

16 ^  
v

Hidden Layers ⓘ

1 ^  
v

Learning Rate ⓘ

0.1 ^  
v

Epochs ⓘ

20 ^  
v

Cancel Ok

## Step 3: Write Code (functionality)

---

Within the class, two functions are required: configure and execute

**Configure** defines specifications and rules:

```
def configure(self, configuration_context, input_table_1):

 if knext.string() not in [x.ktype for x in list(input_table_1)]:
 raise knext.InvalidParametersError(
 "The input table does not have any string columns. You need to have a string column for this node."
)
```

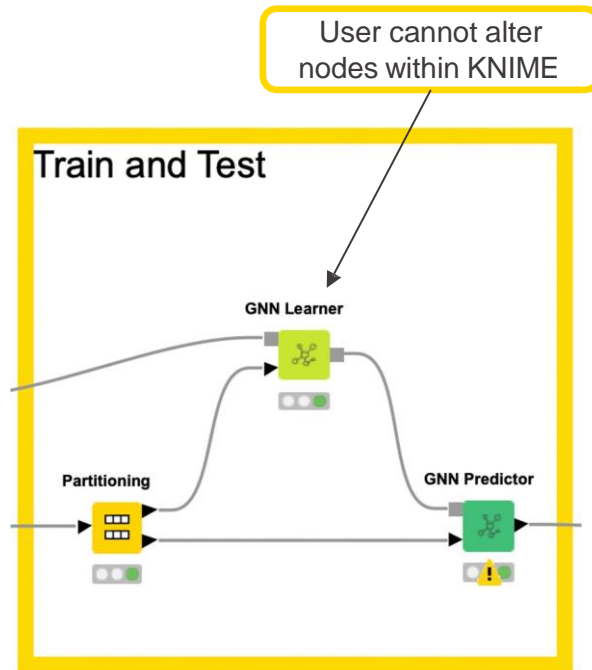
**Execute** performs the operations:

```
def execute(self, exec_context, graph_dict, train_set):
 train_set = train_set.to_pandas()
 graph_dict = pickle.loads(graph_dict)

 graph = graph_dict['graph']
 msk = AddMask(graph_dict['key'])
 masked_graph = msk(graph, train_set)
```

# Python Script node vs Custom Python-based node

- Script node: easier to use for beginners and can help prototype
- Custom node: control, modularity, versioning, and collaboration



Can write scripts in Visual Studio Code and version/collaborate with GitHub

```
class GNNLearner:
 """
 Train GNN
 """

 hidden_channels = knext.IntParameter("Hidden Channels", "The number of hidden channels desired.", 1)
 number_of_hidden_layers = knext.IntParameter("Hidden Layers", "The number of hidden layers desired.", 1)
 learning_rate = knext.DoubleParameter("Learning Rate", "The learning rate to use in the optimizer", 0.01)
 epochs = knext.IntParameter("Epochs", "The number of epochs to train for.", 1)
 criterion = nn.CrossEntropyLoss()

 def configure(self, configure_context, input_binary_1, input_schema_1):
 return knext.BinaryPortObjectSpec("org.knime.torch.learner")

 def execute(self, exec_context, graph_dict, train_set):
 train_set = train_set.to_pandas()
 graph_dict = pickle.loads(graph_dict)

 graph = graph_dict['graph']
 msk = AddMask(graph_dict['key'])
 masked_graph = msk(graph, train_set)
```

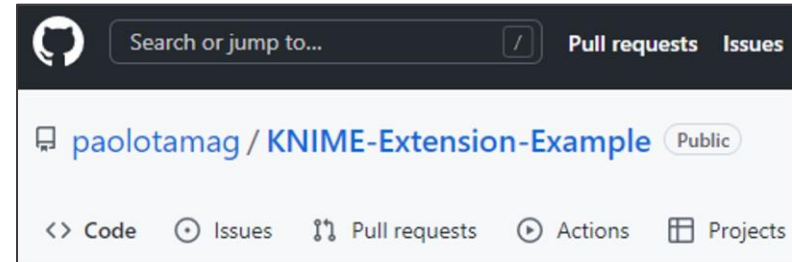
# Resources for custom Python-based nodes

- Reminder: Must be using KNIME Analytics Platform 4.6+

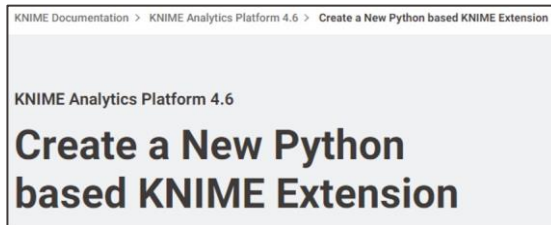
## [Blog](#)



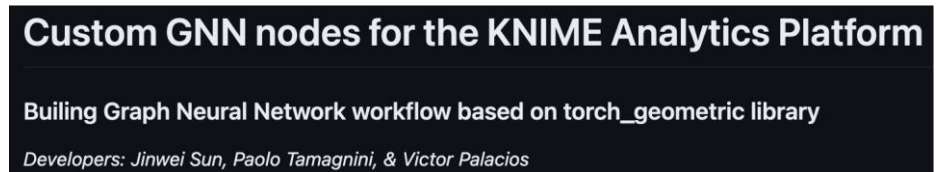
## [Geo Distances](#)



## [Documentation](#)



## [Graph Neural Networks](#)



# Chemistry in Pure Python Extensions

---

<https://forum.knime.com/t/announcing-the-interoperability-between-rdkit-and-python-in-knime-4-7/60416>

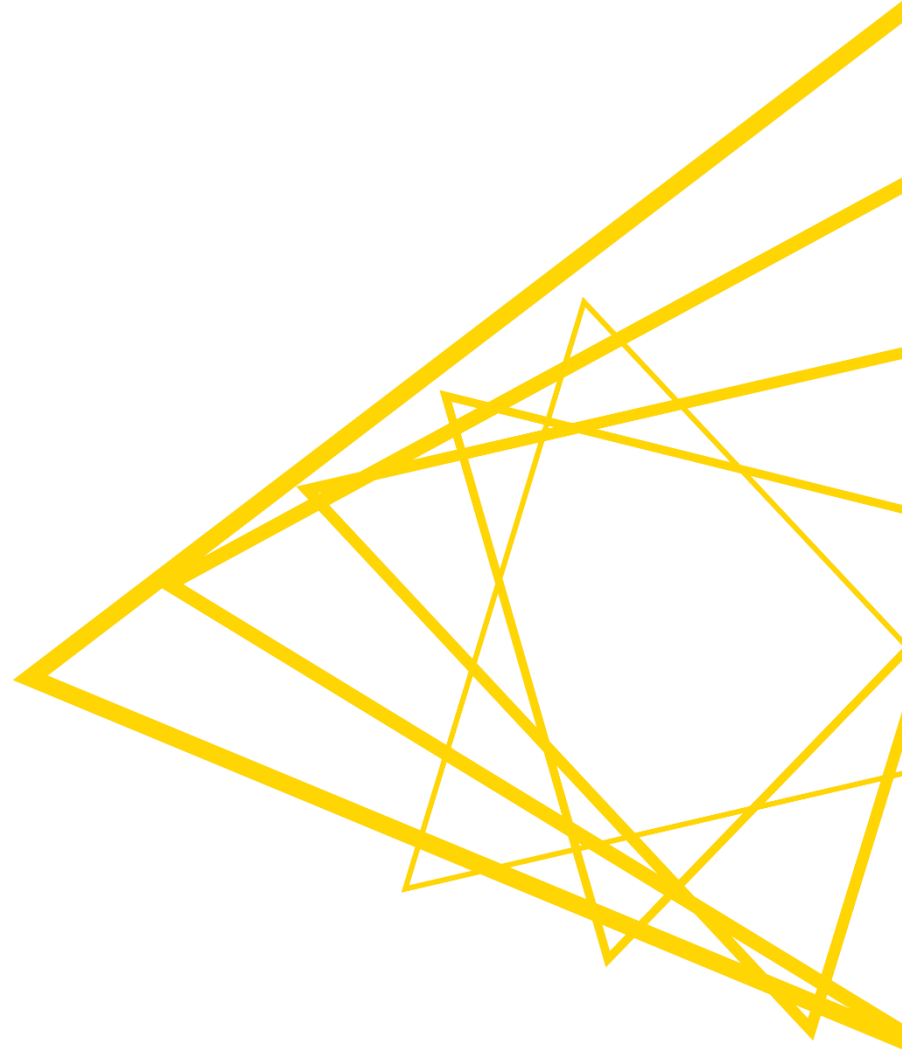
- Converting between molecule types in Python
- Using RDKit in Pure-Python nodes
  - Sending fingerprints from Python to KNIME
  - Working with KNIME-provided fingerprints in Python
  - Working with RXN reactions
-

# Useful links

---

- KNIME Python Integration Guide:
  - [https://docs.knime.com/latest/python\\_installation\\_guide/index.html](https://docs.knime.com/latest/python_installation_guide/index.html)
- Blog post “KNIME and Jupyter”:
  - <https://www.knime.com/blog/knime-and-jupyter>
- Blog post “Manage Your Python Environments with Conda and KNIME”:
  - <https://www.knime.com/blog/how-to-manage-python-environments-conda-and-knime>
- **KNIME Python Node Extension guide:**
  - [https://docs.knime.com/latest/pure\\_python\\_node\\_extensions\\_guide/index.html#introduction](https://docs.knime.com/latest/pure_python_node_extensions_guide/index.html#introduction)
- Find the documentation for the new API:
  - <https://knime-python.readthedocs.io/en/stable/>
- Get in touch with the developers at KNIME and provide feedback:
  - <https://forum.knime.com/c/knime-development/>
  - please add the tag “Python”
- Blog post “4 Steps for your Python Team to Develop KNIME Nodes”:
  - <https://www.knime.com/blog/4-steps-for-your-python-team-to-develop-knime-nodes>

# Chemistry Use Cases

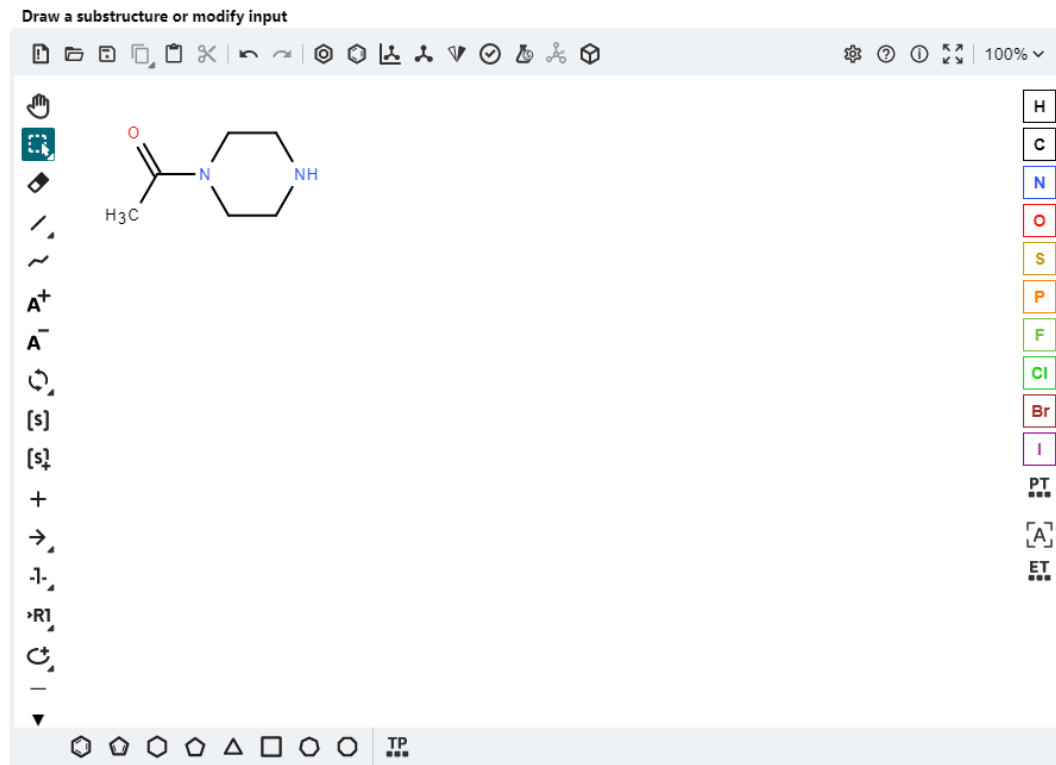


# Molecule Sketcher Widget

## ■ Possible output formats

- MOL
- MOL V3000
- RXN
- RXN V3000
- SMILES
- Extended SMILES
- SMARTS
- CML
- InChI
- InChI with AuxInfo
- KET

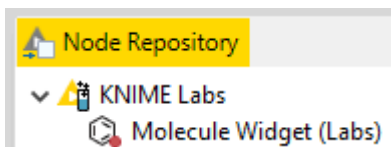
**Molecule  
Widget (Labs)**





# Molecule Sketcher Widget

- Part of the Quick Forms extension
- Under *KNIME Labs* in the Node Repository
- Based on the [EPAM Ketcher](#) (version 2.7.2)



## Extension

The Molecule Widget (Labs) node is part of this extension:

**KNIME Quick Forms** ✓

Contains nodes that contribute quick form elements. These elements can be used to abstrac...

KNIME AG, Zurich, Switzerland



# Today's goal

---

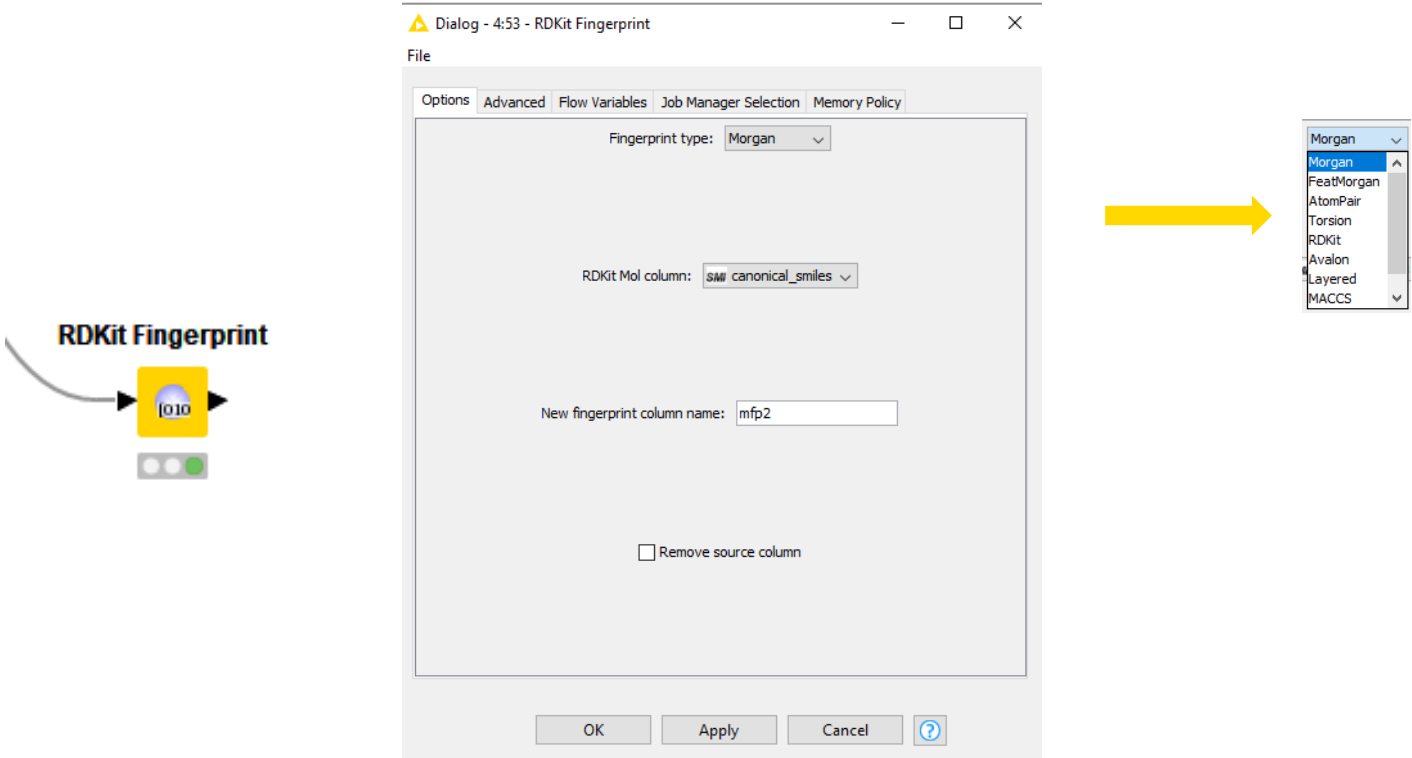
- Perform hierarchical clustering based on molecular fingerprints and create an interactive view to pick interesting clusters
- Fingerprints
- Bit Vector Distances
- Hierarchical Clustering

# Molecular Fingerprints Encode Substructures

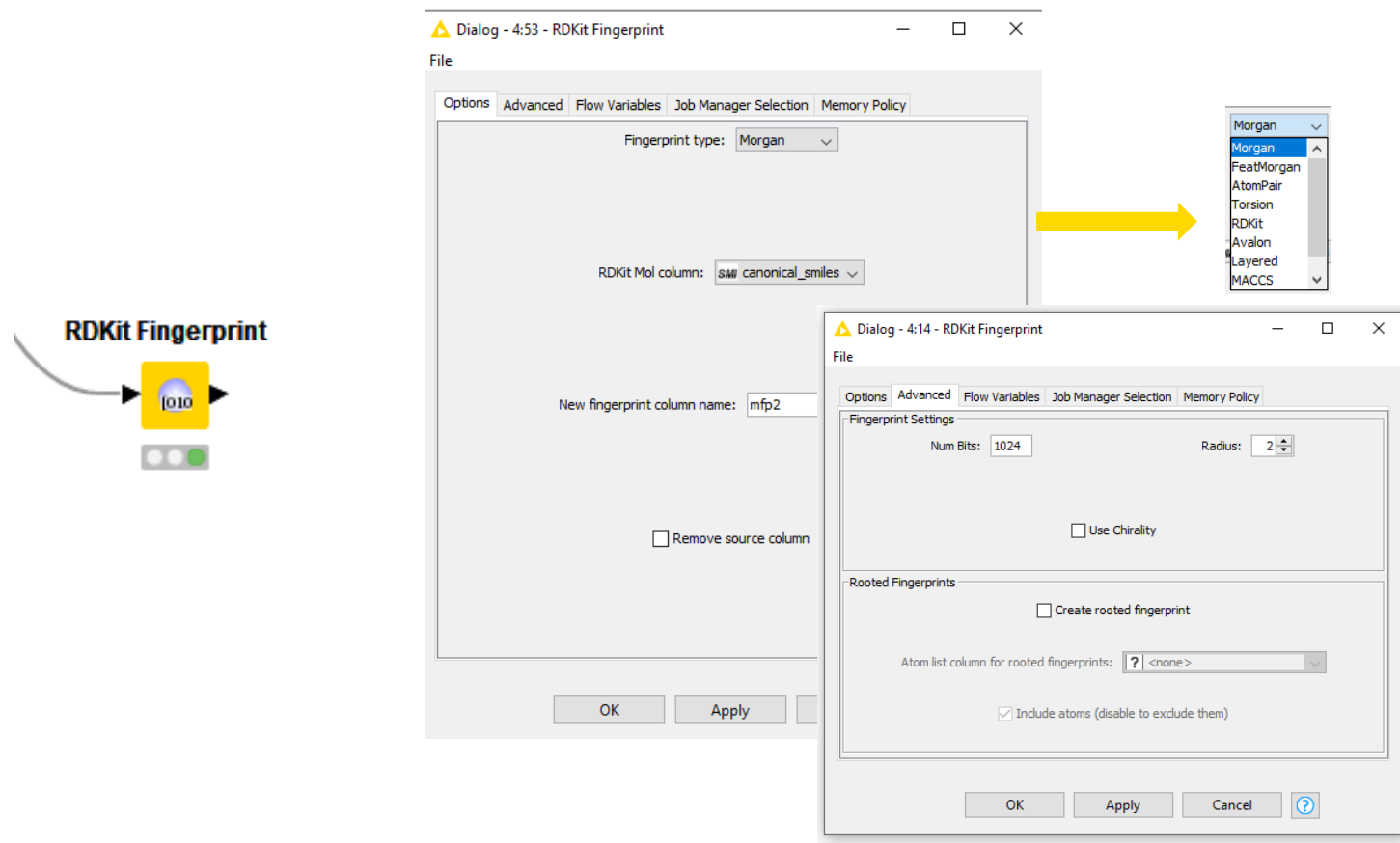


- Bit vectors, where each bit corresponds to a presence (1) or absence (0) of a certain atom and its neighbors

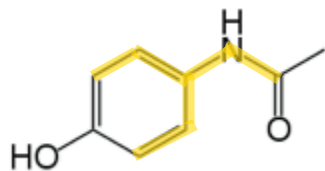
# RDKit Fingerprint node



# RDKit Fingerprint node



# Tanimoto Similarity



[0000010000000000000000001000...]

$$\text{Tanimoto}(A,B) = \frac{|A \cap B|}{|A \cup B|}$$

[0,1]

0 – no similarity

1 – “identical”

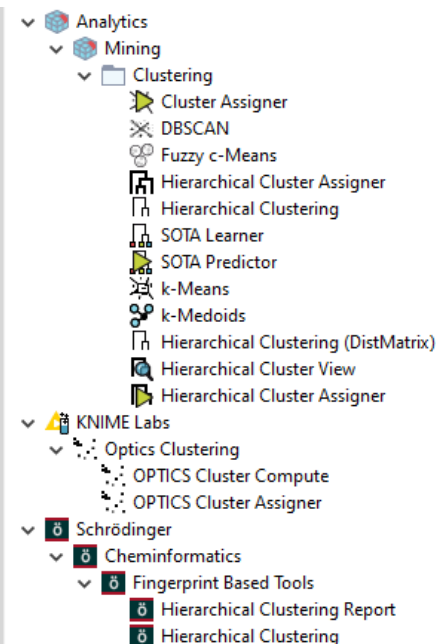
# Goal of Cluster Analysis

Discover hidden structures in **unlabeled** data (unsupervised)

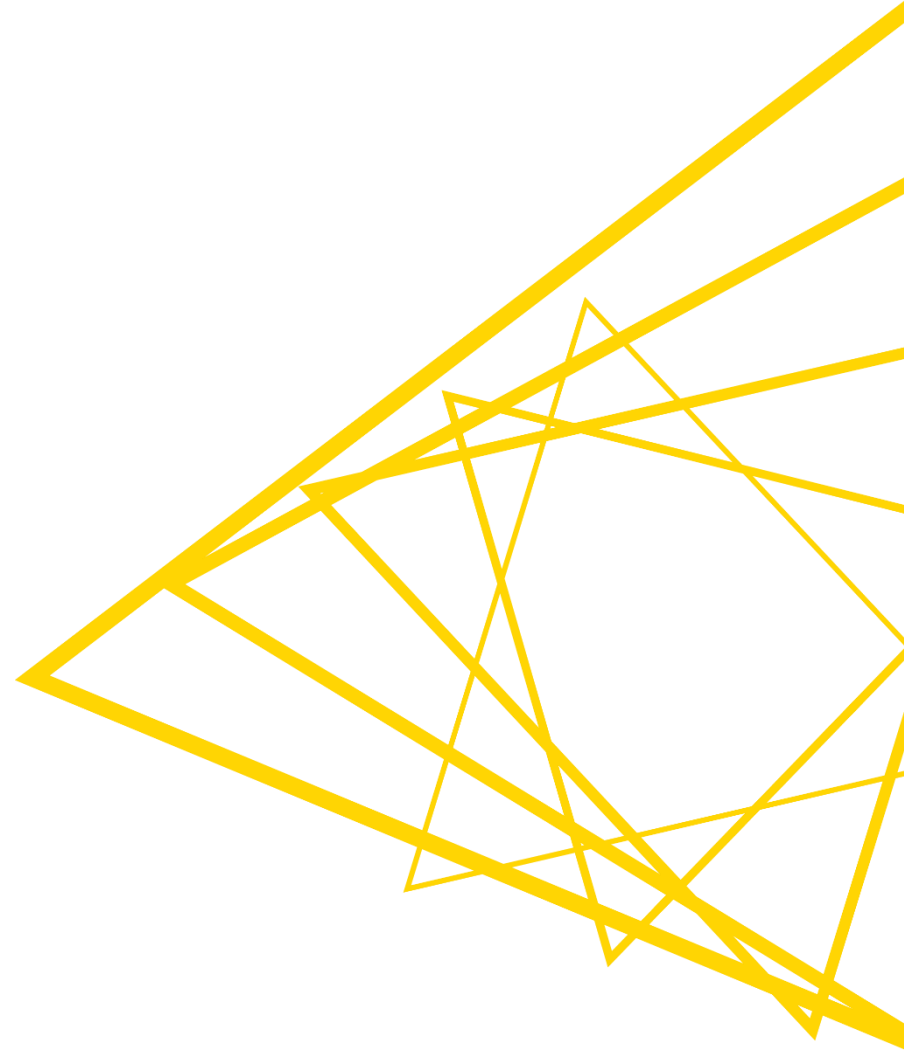
**Clustering** identifies a set of groups (*clusters*)

$C_1, C_2 \dots, C_k$  in the dataset such that:

- Objects within the *same* cluster  $C_i$  shall be as similar as possible
- Objects of *different* clusters  $C_i, C_j$  ( $i \neq j$ ) shall be as dissimilar as possible



# Hierarchical Clustering





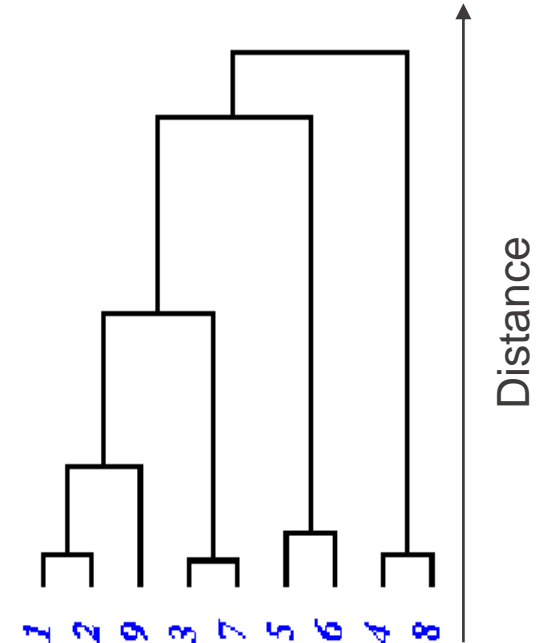
# Linkage Hierarchies: Basics

## Goal

- Construction of a hierarchy of clusters (*dendrogram*) by merging/separating clusters with minimum/maximum distance

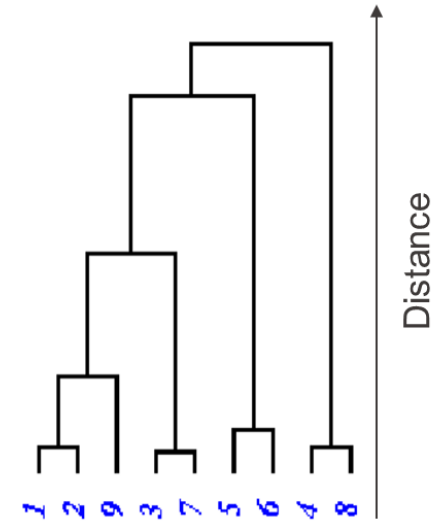
## Dendrogram:

- A tree representing hierarchy of clusters, with the following properties:
  - Root: single cluster with the whole data set.
  - Leaves: clusters containing a single object.
  - Branches: merges / separations between larger clusters and smaller clusters / objects



# Base Algorithm

1. Form initial clusters consisting of a single object and compute the distance between each pair of clusters.
2. Merge the two clusters having minimum distance.
3. Calculate the distance between the new cluster and all other clusters.
4. If there is only one cluster containing all objects:  
Stop, otherwise go to step 2.



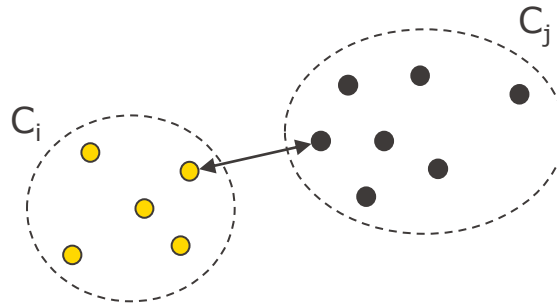
# Single Linkage

- Distance between clusters (nodes):

$$Dist(C_1, C_2) = \min_{p \in C_1, q \in C_2} \{dist(p, q)\}$$

Distance of the closest two points, one from each cluster

- Merge Step: Union of two subsets of data points



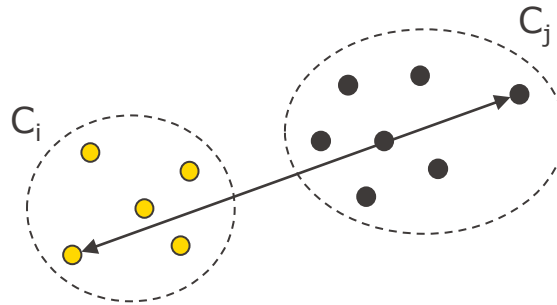
# Complete Linkage

- Distance between clusters (nodes):

$$Dist(C_1, C_2) = \max_{p \in C_1, q \in C_2} \{dist(p, q)\}$$

Distance of the farthest two points, one from each cluster

- Merge Step: Union of two subsets of data points



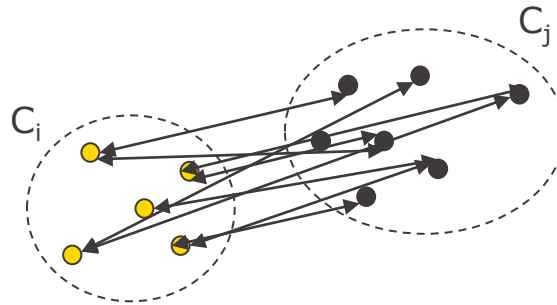
# Average Linkage

- Distance between clusters (nodes):

$$Dist_{avg}(C_1, C_2) = \frac{1}{|C_1| \cdot |C_2|} \sum_{p \in C_1} \sum_{q \in C_2} dist(p, q)$$

Average distance of all possible pairs of points between  $C_1$  and  $C_2$

- Merge Step: Union of two subsets of data points

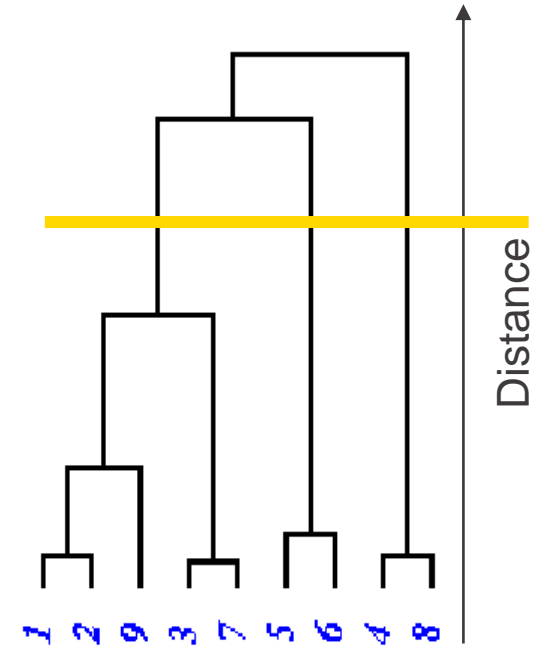


# Comments on Single Linkage and Variants

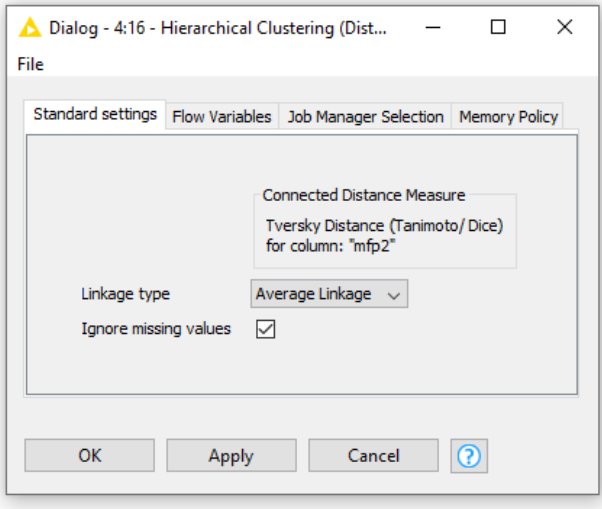
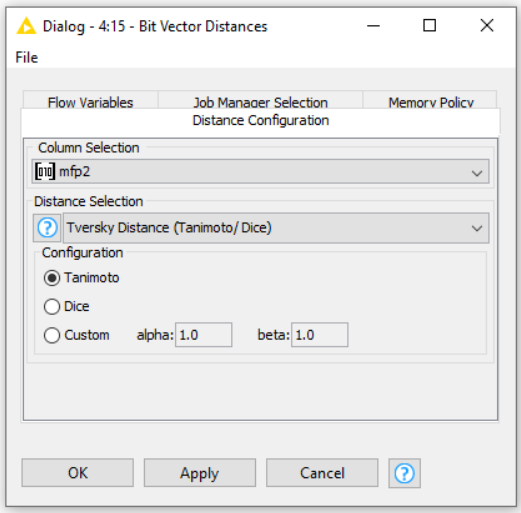
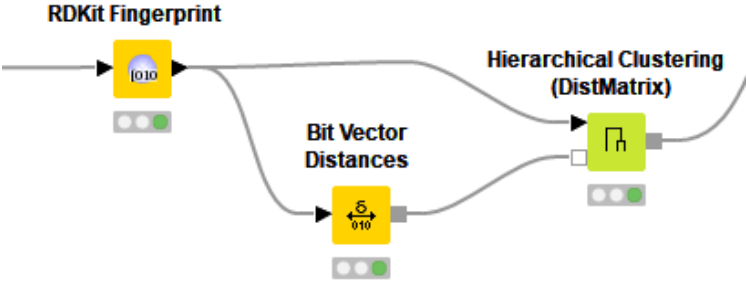
+ Finds not only a „flat“ clustering, but a hierarchy of clusters (dendrogram)

+ A single clustering can be obtained from the dendrogram (e.g., by performing a horizontal cut)

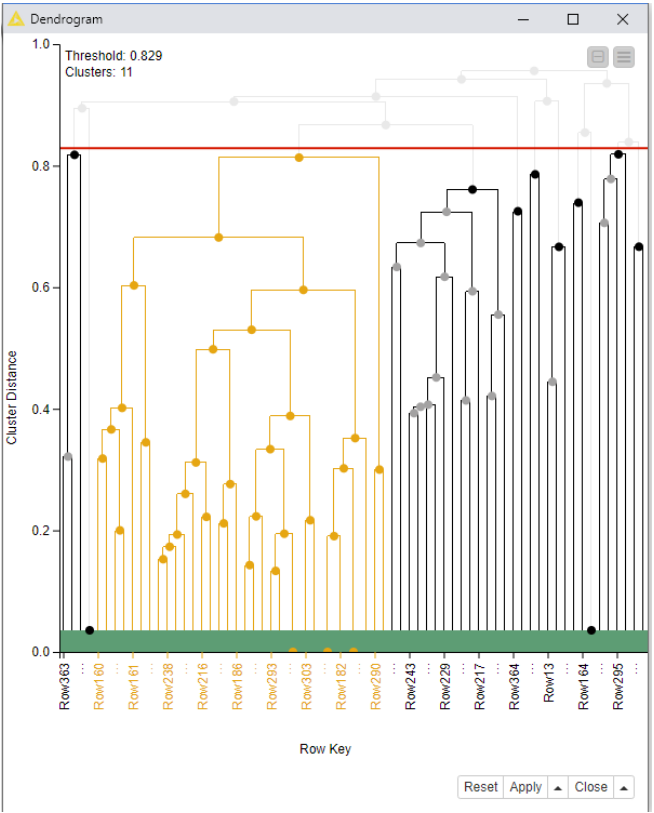
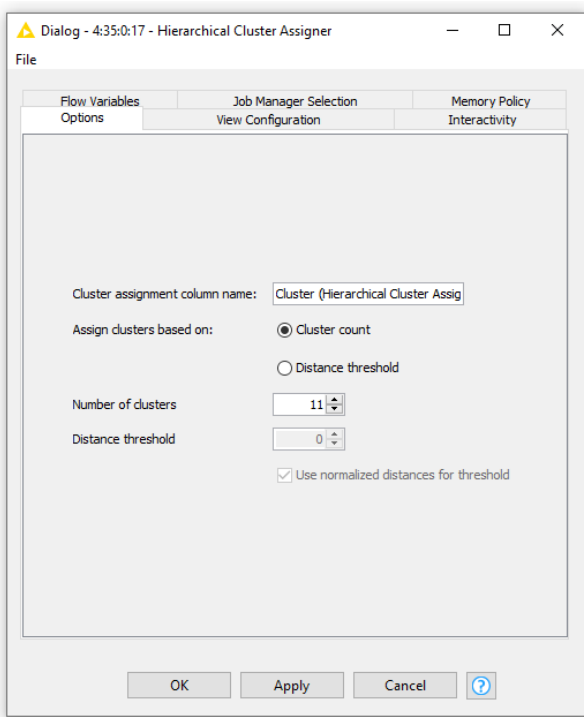
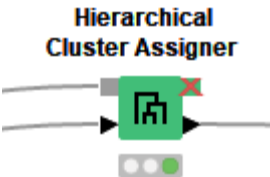
- Decisions (merges/splits) cannot be undone
- Sensitive to noise (Single-Link) (a „line“ of objects can connect two clusters)
- Inefficient → Runtime complexity at least  $O(n^2)$  for  $n$  objects



# Bit Vector Distances and Hierarchical Clustering



# Hierarchical Cluster Assigner








# The Data

← → ↺ [pubs.acs.org/doi/10.1021/acs.jmedchem.9b01658](https://pubs.acs.org/doi/10.1021/acs.jmedchem.9b01658) ☆ 15 G Paused +

ACS ACS Publications C&EN CAS Find my institution Log In


 Search text, DOI, authors, etc.  My Activity Publications 

COVID-19 Remote Access Support: [Learn More](#) about expanded access to ACS Publications research.

RETURN TO ISSUE | < PREV ARTICLE NEXT >

## Optimization of Tetrahydroindazoles as Inhibitors of Human Dihydroorotate Dehydrogenase and Evaluation of Their Activity and In Vitro Metabolic Stability




Gergana Popova\*, Marcus J. G. W. Ladds, Lars Johansson, Aljona Saleh, Johanna Larsson, Lars Sandberg, Sara Häggblad Sahlberg, Weixing Qian, Hjalmar Gullberg, Neeraj Garg, Anna-Lena Gustavsson, Martin Haraldsson, David Lane, Ulrika Yngve, and Sonia Lain


✓ Cite this: *J. Med. Chem.* 2020, 63, 8, 3915–3934  
Publication Date: March 26, 2020  
<https://doi.org/10.1021/acs.jmedchem.9b01658>  
Copyright © 2020 American Chemical Society  
[RIGHTS & PERMISSIONS](#)  ACS AuthorChoice

| Article Views | Altmetric | Citations |
|---------------|-----------|-----------|
| 954           | 10        | –         |

[LEARN ABOUT THESE METRICS](#)

Share Add to Export

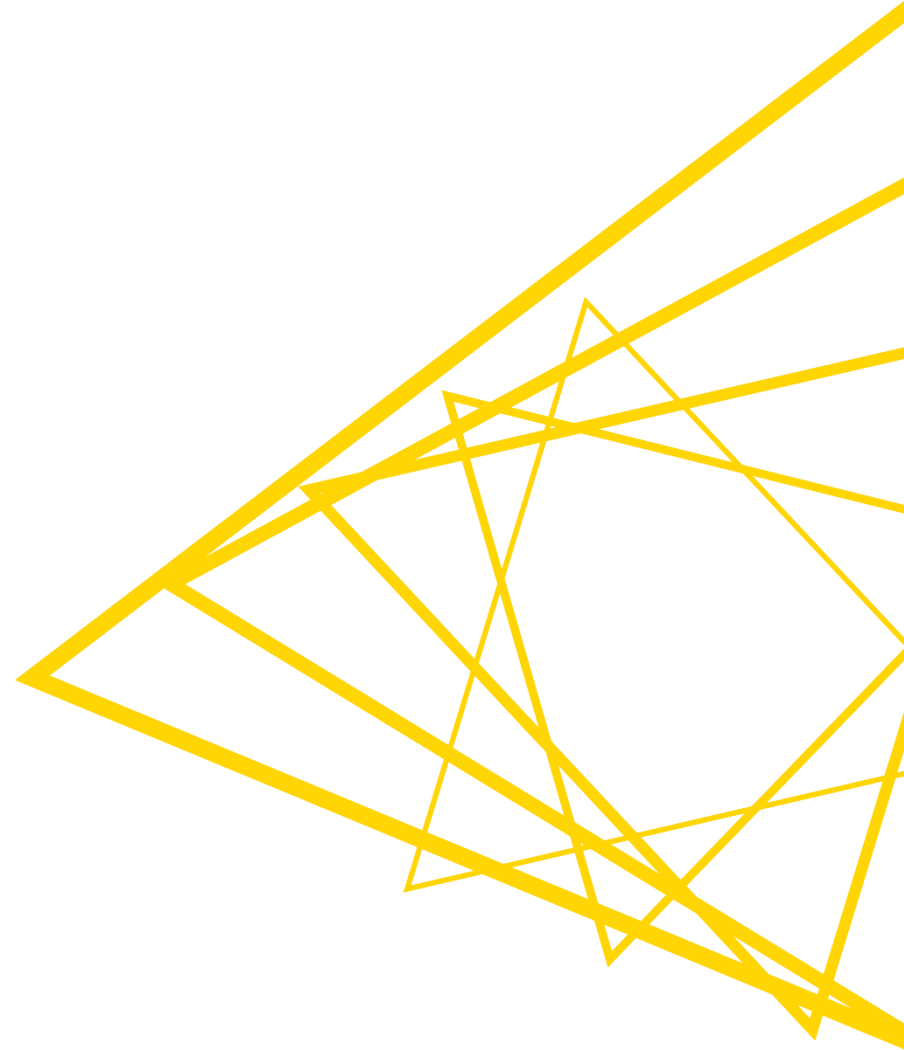


Journal of Medicinal Chemistry

Note: the paper is open access

<https://doi.org/10.1021/acs.jmedchem.9b01658>

# Similarity search



# Today's goal

---

- Expand our results by finding building blocks similar to those we selected in the previous exercise (03\_Clustering)
- The catalog of building blocks is taken from:  
<https://zinc15.docking.org/catalogs/enaminebbe/substances/>
- First calculate the fingerprints from the compounds and the building blocks. To ensure that the settings for this calculation are the same we will create and use a **shared component**. Then we perform a similarity search and end with a final selection and annotation step

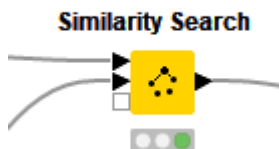
# Agenda for Today

---

- Similarity search
- Shared components
- Configuration nodes
- Search for functional groups
- Annotations in interactive view



# Similarity Search



Dialog - 5:45 - Similarity Search

File

Options | Flow Variables | Job Manager Selection | Memory Policy

Distance Selection: Tanimoto

☒ Manual Selection ☐ Wildcard/Regex Selection

**Exclude**

Filter

No columns in this list

☒ Enforce exclusion

**Include**

Filter

mfp2

☐ Enforce inclusion

> >> < <<

Search Options

Coefficient Type

☐ distance

☒ similarity (1 - distance) - only for tanimoto

Neighbors selection

☒ Nearest (most similar)

☐ Farthest (most dissimilar)

Neighbor Count: 5

☒ Use range filter (min/max similarity)

Minimum: 0.35

Maximum: 1.00

Output Options

Output column prefix: nearest neighbor

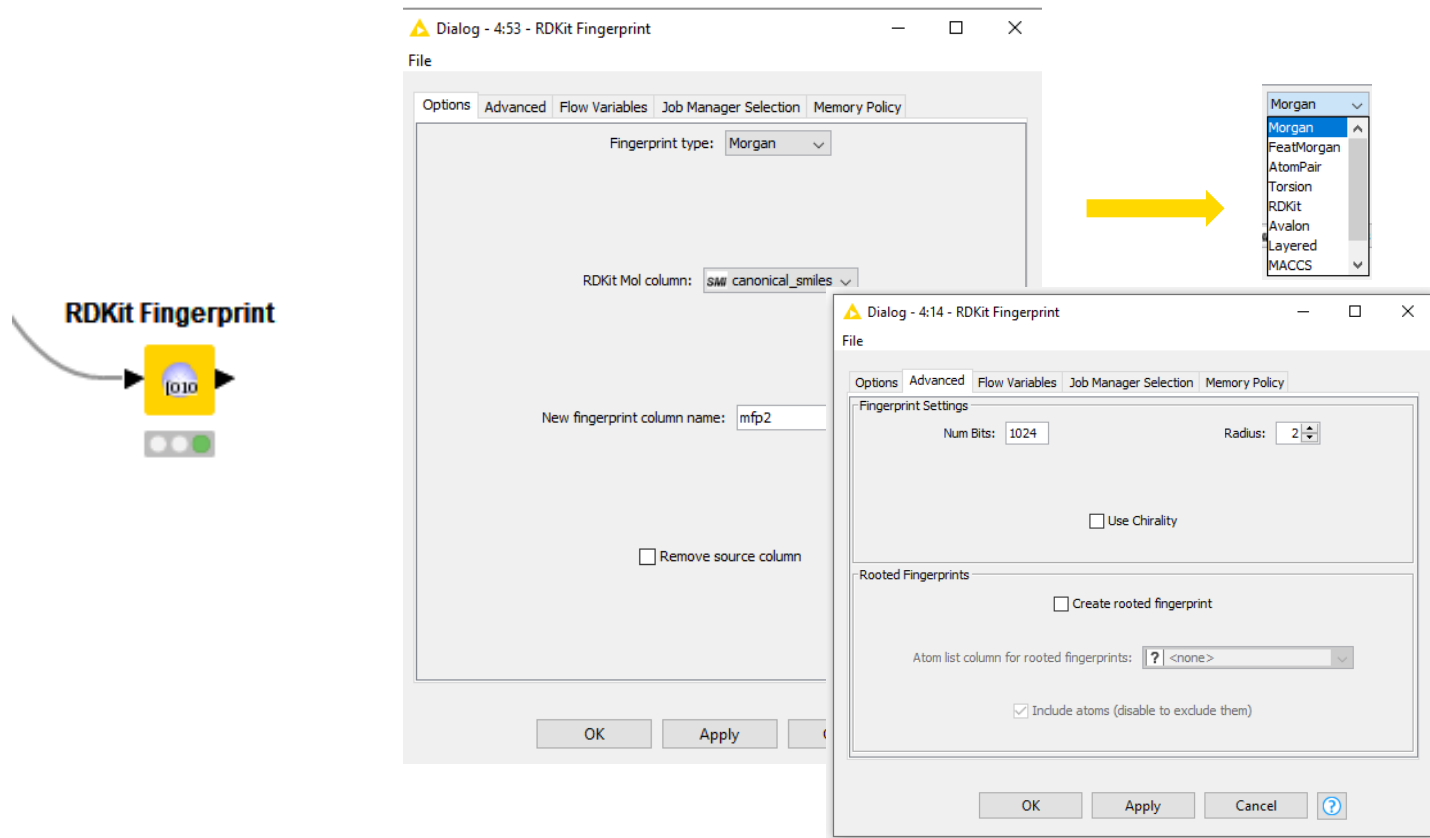
Representative Column (2nd input): S | zinc\_id

RowID Suffix Separator: \_

OK Apply Cancel ?

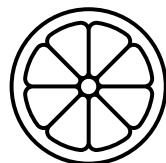


# RDKit Fingerprints

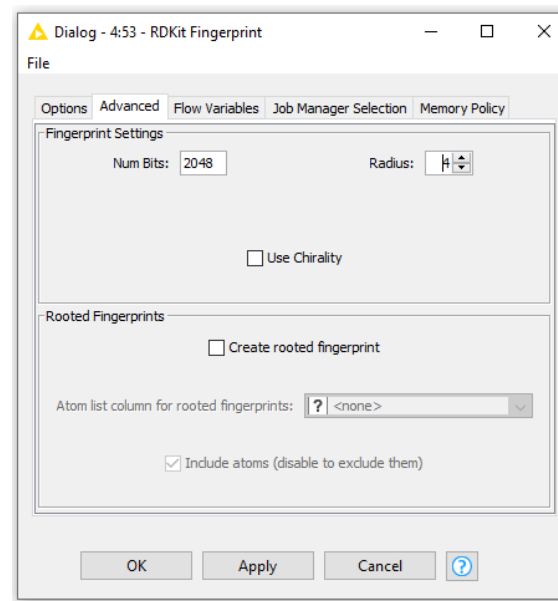
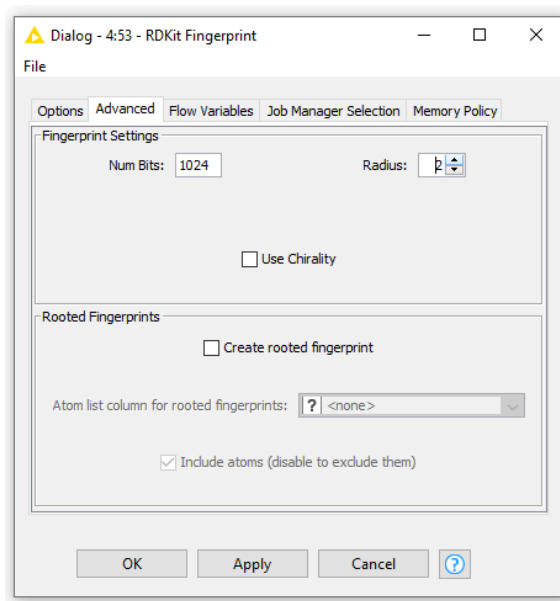




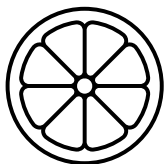
# Similarity Search with Fingerprints



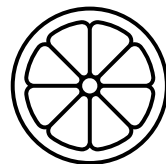
=



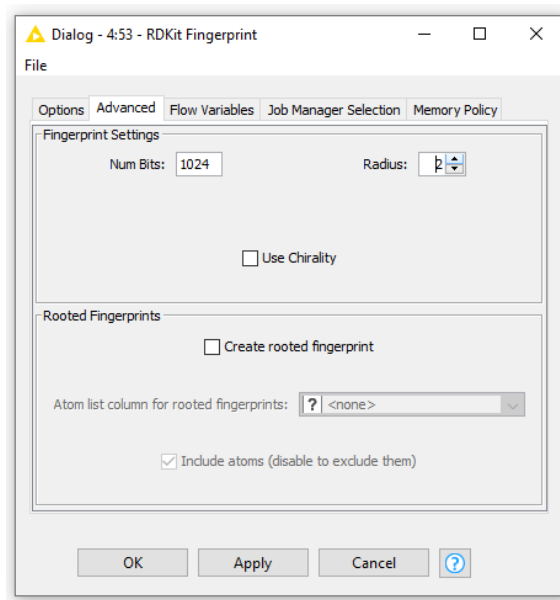
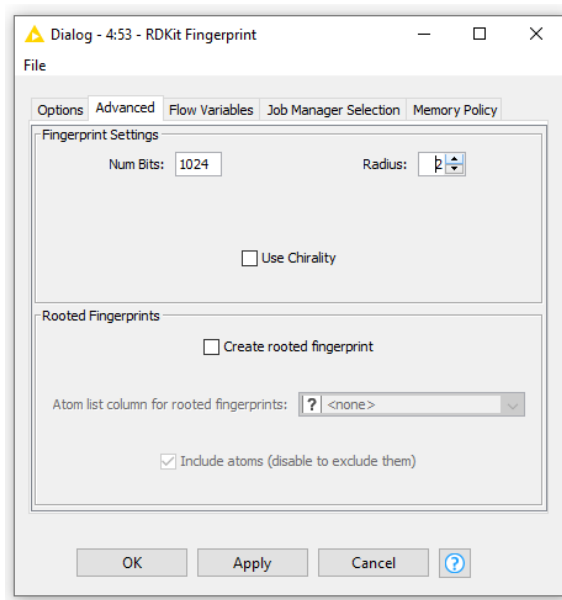
# Similarity Search with Fingerprints



=

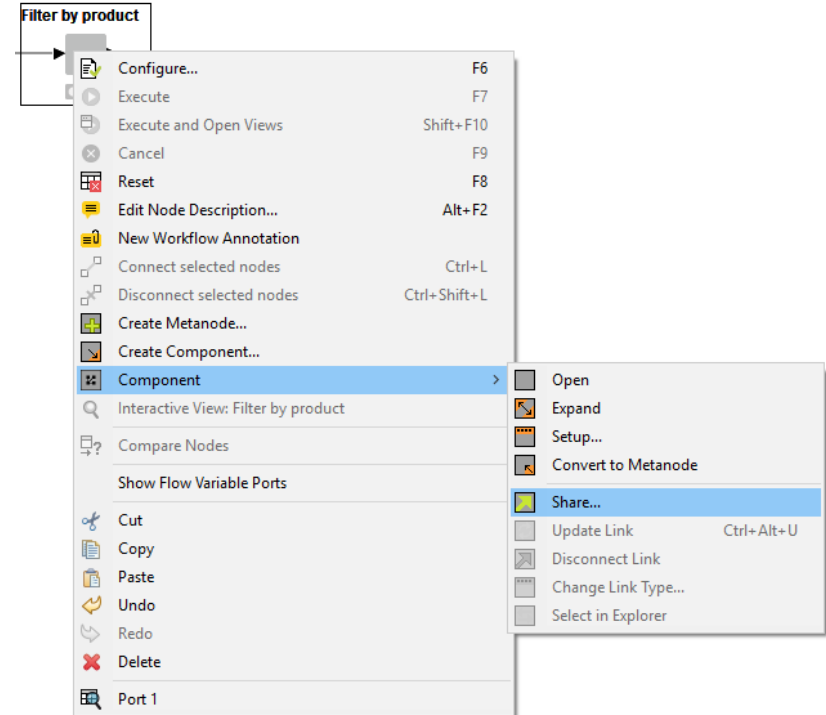


Reusable  
Shared  
Components!



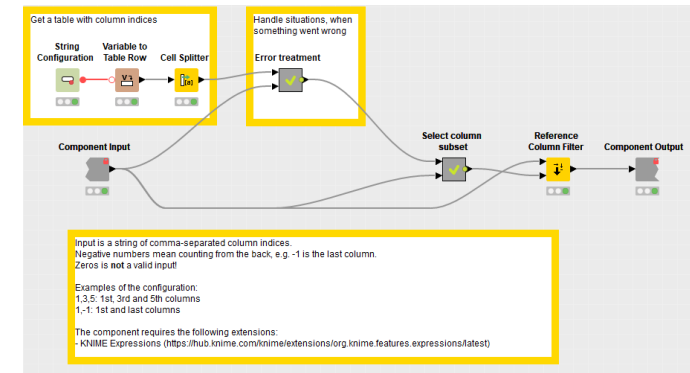
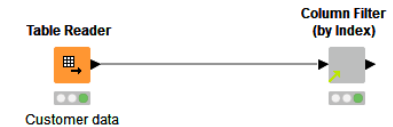
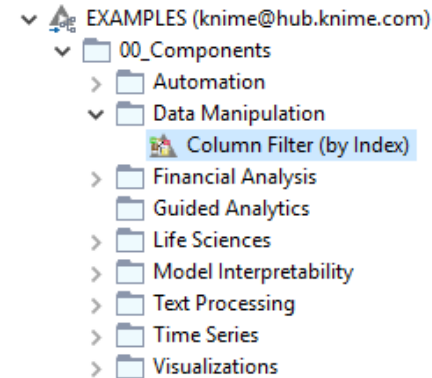
# Shared Components

- Shared Components are read-only instances of a Component
- Components can be saved in your KNIME workspace, KNIME Server, or the KNIME Hub for later reuse
- To do this, simply right-click any Component and select “Share...”



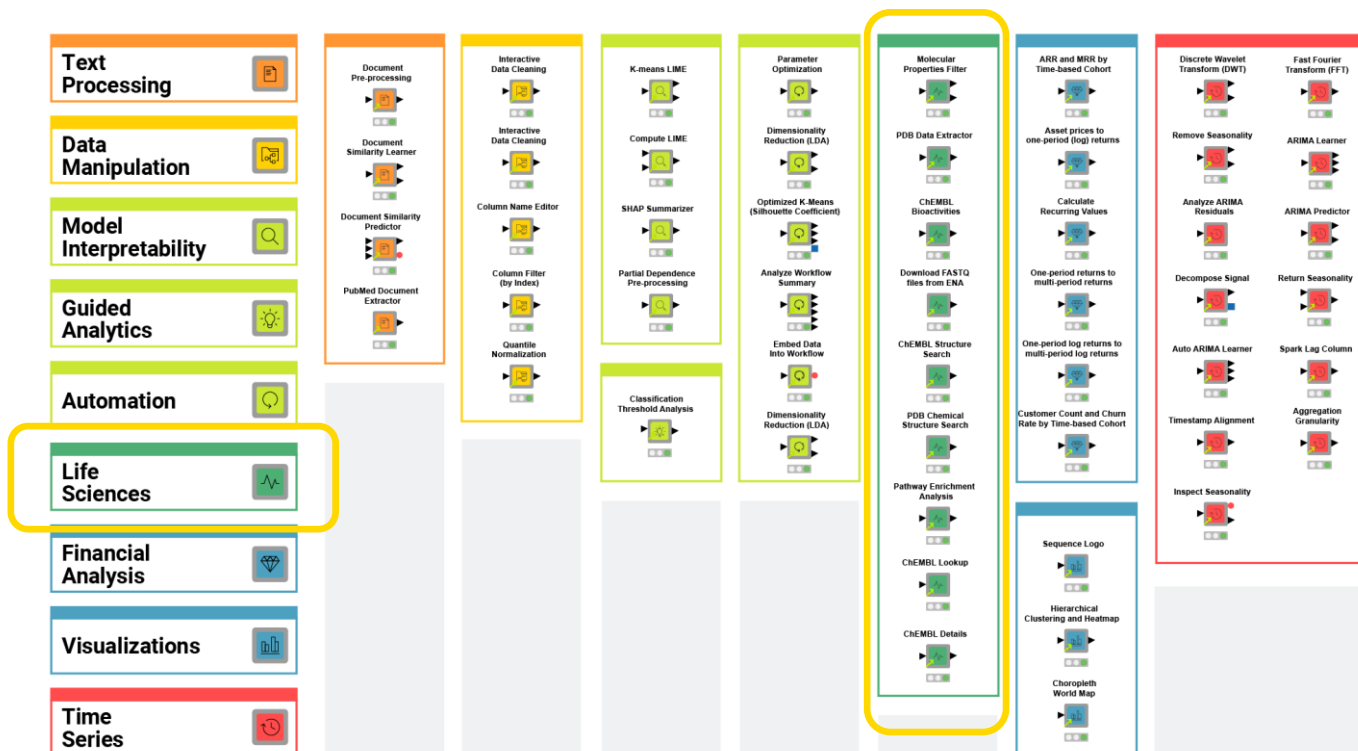
# Shared Components

- To use a Shared Component, drag and drop it to the workflow editor
- Instances of Shared Components can be updated either manually or when workflow is opened
- Shared Component can also be unlinked from its original location, which makes it editable in the workflow directly
- Update Shared Components by overwriting them

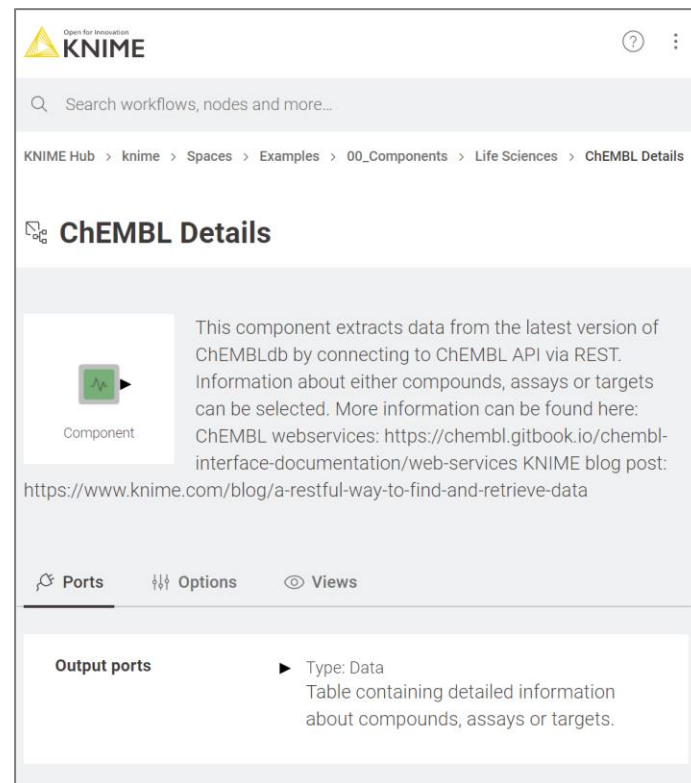
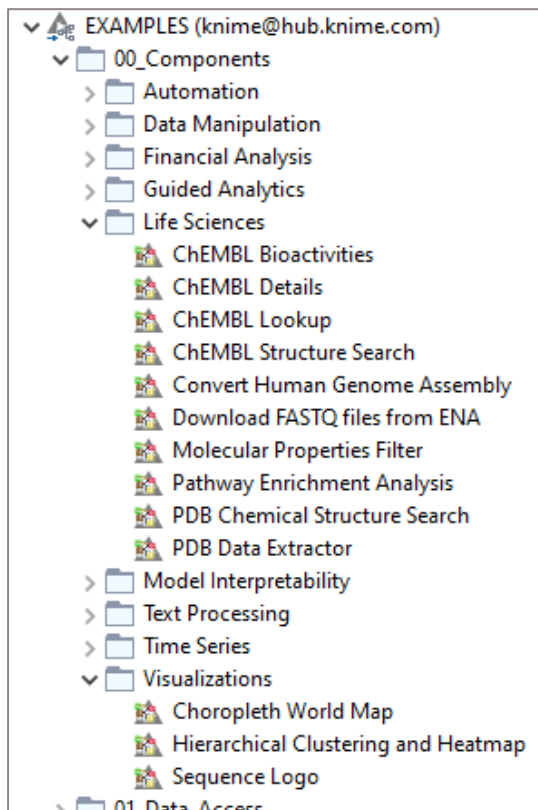


# KNIME Verified Components

- [Verified Components](#) reuse bundled functionalities, verified by KNIME experts
- Released and updated on the [KNIME Hub](#)

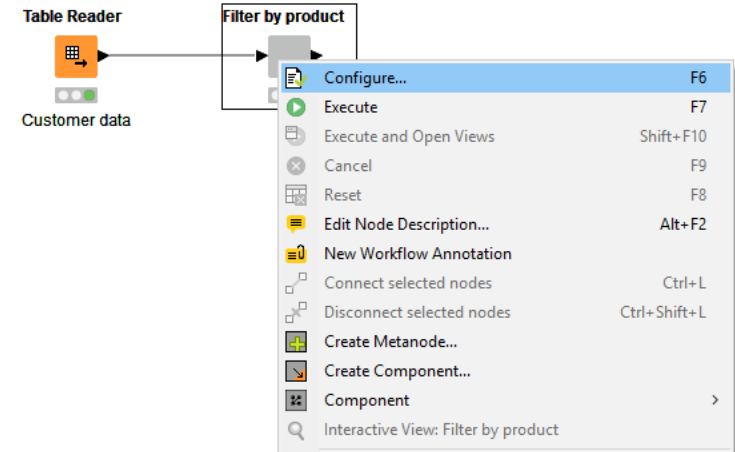
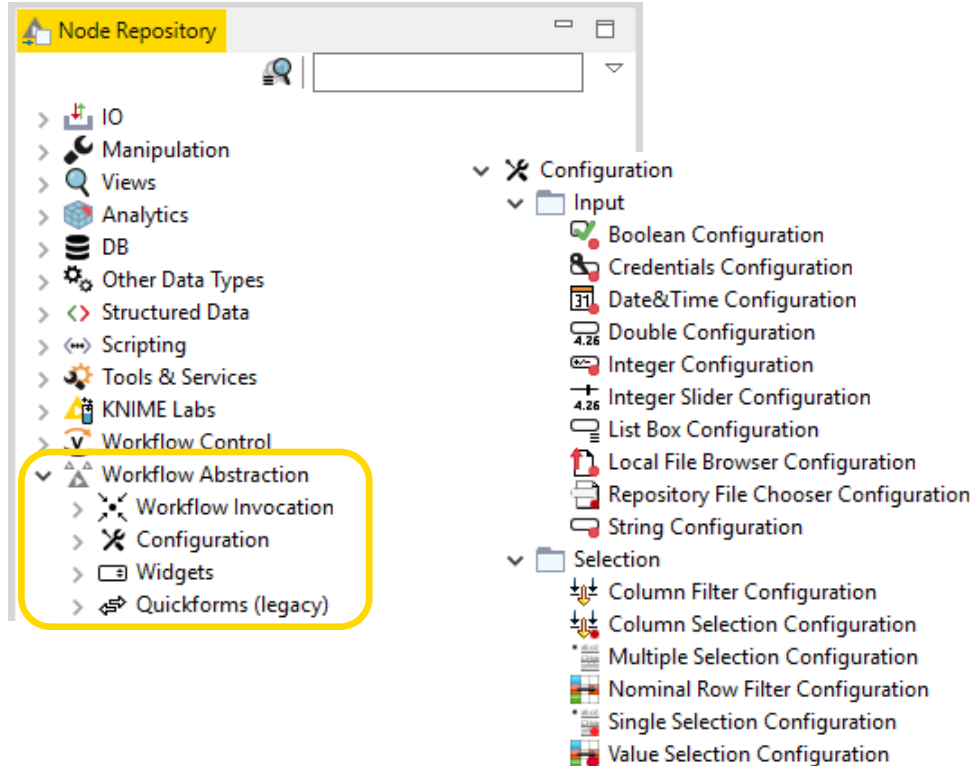


# Publicly Shared Components



<https://www.knime.com/blog/knime-analytics-platform-40-components-are-for-sharing>

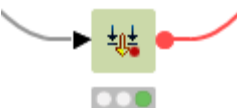
# Configuration Nodes



- Double click a component to configure

# Column Selection Configuration

Column Selection Configuration



Dialog - 5:0:54 - Column Selection Configuration

File

Control | Flow Variables | Job Manager Selection | Memory Policy

Label: Smiles column

Description: Enter Description

Parameter/Variable Name: smiles\_selection

Selection Type: Dropdown

Type Filter:

Allowed types: ☐ Allow all types ☐ Hide columns without domain

☒ **SMI** Smiles ☐ **S** String ☐ **B** Boolean value

☐ **I** Number (Integer) ☐ **D** Number (double) ☐ **L** Number (long)

☐ **Local Date Time**

Default Value: **SMI** smiles

Limit number of visible options: ☐

Number of visible options: 10

OK Apply Cancel ?

Generate fingerprints



Dialog - 5:0 - Generate fingerprints

File

Options | Flow Variables | Memory Policy | Job Manager Selection

Smiles column

smiles

OK Apply Cancel ?



# Component Description

Make your component look like a KNIME node

**Add description of the component**

**Add description of the input and output ports**

**Add background color or icon**

**Filter by Product**

Description

This component filters the data by the selected product name

Component Icon

Drag and Drop a square image file

PNG image of size 16x16 or larger

Ports

In Port

Out Port

Filter by Product

Table Reader

Customer data

selected product category via Configuration

Dialog Options

Select product:

The product name to use for filtering

Ports

Input Ports

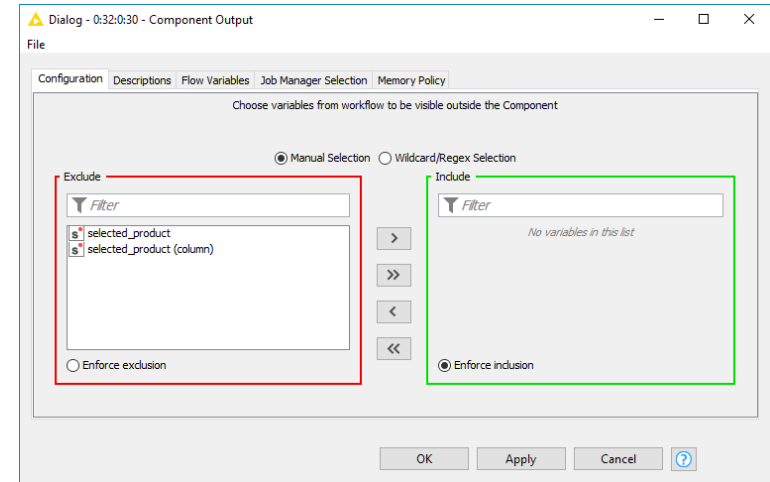
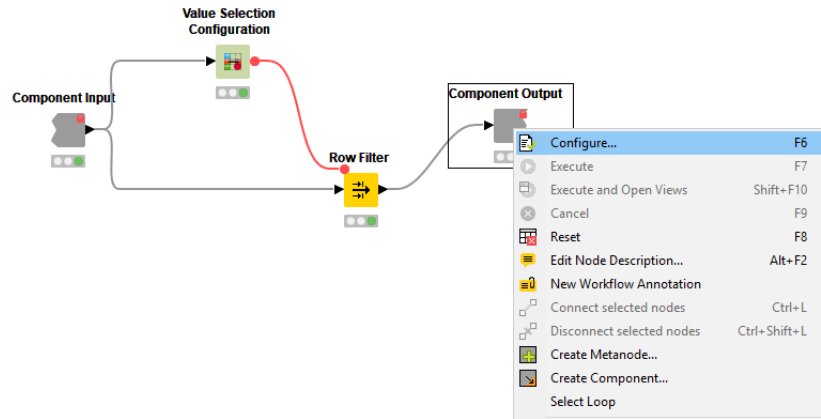
0 Data containing records for all products

Output Ports

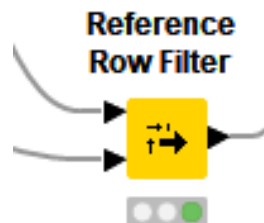
0 Data containing records for the selected product

# Passing Variables into/from Components

- Flow Variables by default only available locally inside Component
- Configure Component Input/Output to pass Flow Variables from/to outside Component



# Reference Row Filter



Dialog - 3:52 - Reference Row Filter

File

Options | Flow Variables | Job Manager Selection | Memory Policy

Reference columns

Data table column:

Reference table column:

Include rows from reference table

☐ Include rows from reference table

☒ Exclude rows from reference table

OK Apply Cancel ?

## Ports

### Input Ports

- 0 Table from which rows are to be included or excluded
- 1 Table rows used as reference filter

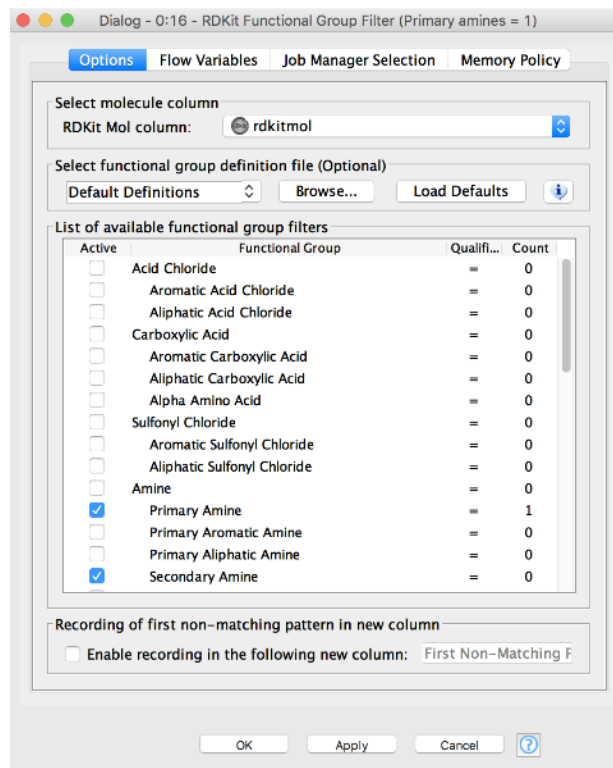
### Output Ports

- 0 Table with filtered rows

# Search for Functional Groups

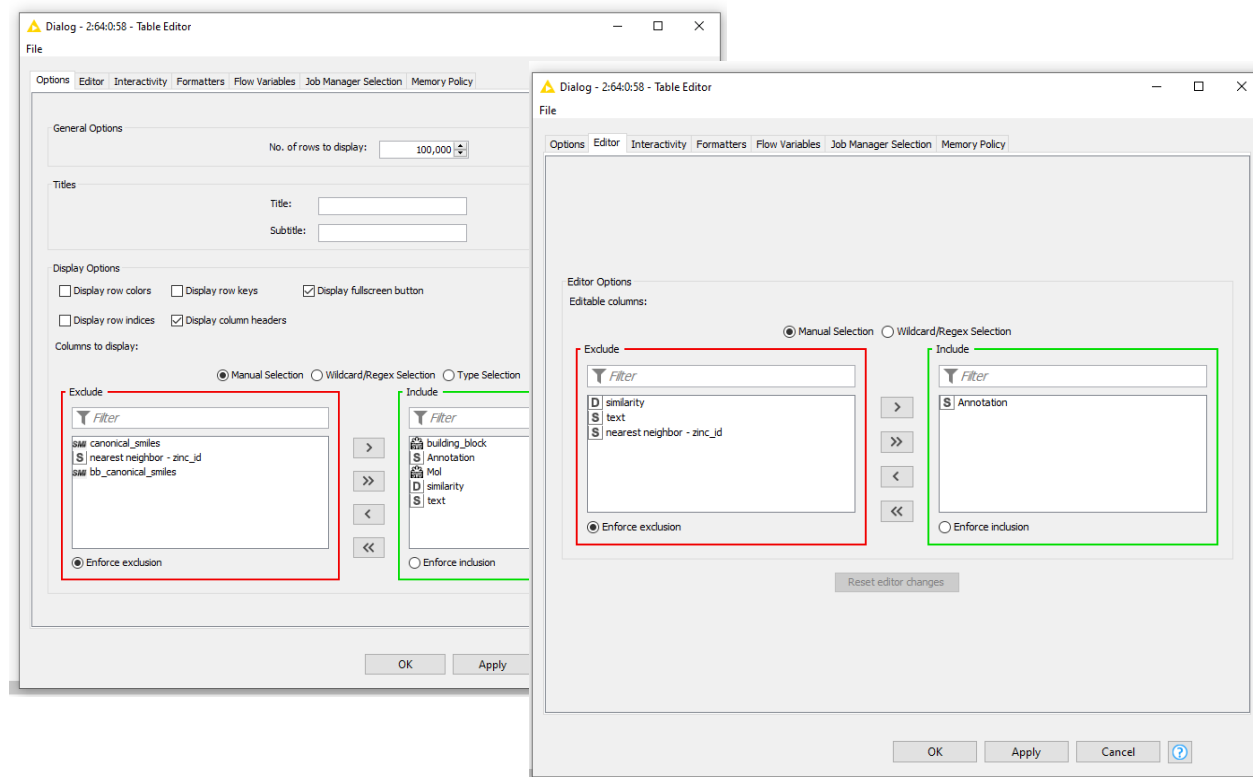
- Splits molecules matching and failing the filters

**RDKit Functional  
Group Filter**



# Table Editor

- Interactive table view and possibility to edit



## 04\_SimilaritySearch

---

- Create shared component with configuration to calculate fingerprints
- Perform similarity search for compounds and building blocks
- Create an interactive view for final selection and annotation

# Confirmation of Attendance and Survey

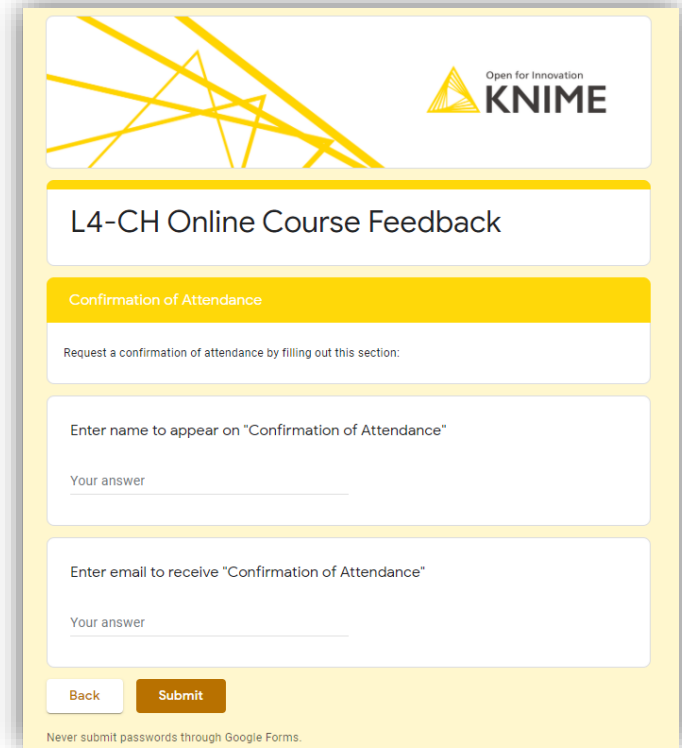
- If you would like to get a “Confirmation of Attendance” please click on the link below\*

LINK

- The link also takes you to our course feedback survey. Filling it in is optional but highly appreciated!

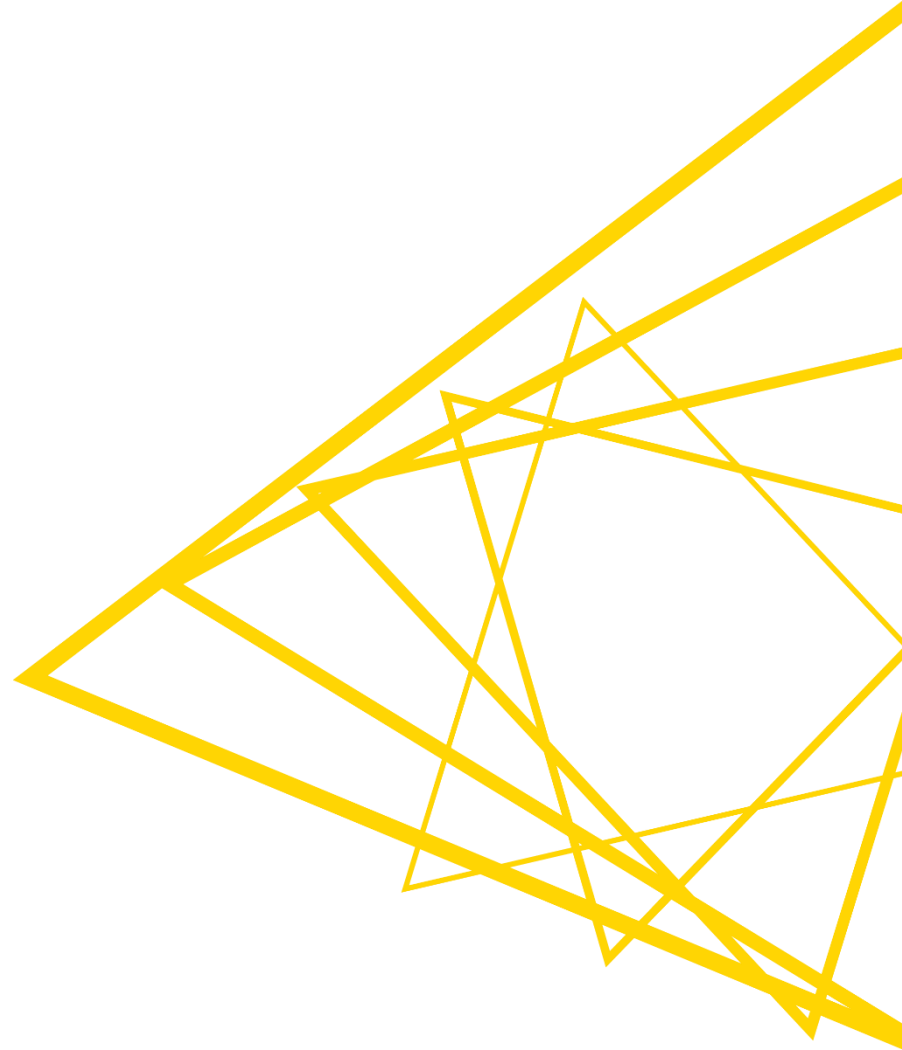
Thank you! 😊

\*Please send your request within 3 days



The screenshot shows a Google Form titled "L4-CH Online Course Feedback". At the top right is the KNIME logo with the tagline "Open for Innovation". Below the title is a yellow section header "Confirmation of Attendance". The form contains two text input fields: "Enter name to appear on 'Confirmation of Attendance'" and "Enter email to receive 'Confirmation of Attendance'", both with "Your answer" placeholder text. At the bottom are "Back" and "Submit" buttons. A footer note states "Never submit passwords through Google Forms."

**End and Backup**

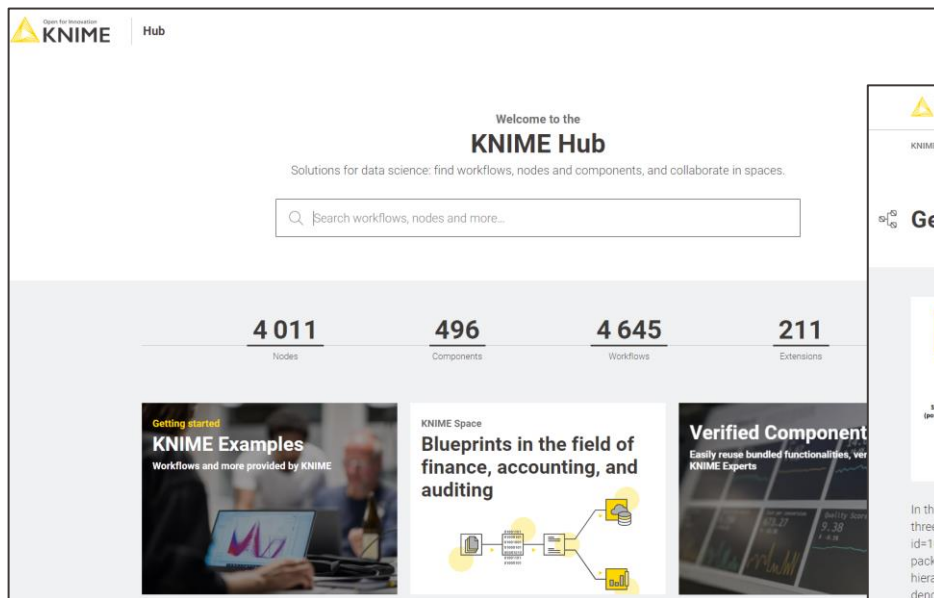




# Hot Keys (for Future Reference)

| Task                       | Hot key                    | Description                                                                              |
|----------------------------|----------------------------|------------------------------------------------------------------------------------------|
| Node Configuration         | F6                         | opens the configuration window of the selected node                                      |
| Node Execution             | F7                         | executes selected configured nodes                                                       |
|                            | Shift + F7                 | executes all configured nodes                                                            |
|                            | Shift + F10                | executes all configured nodes and opens all views                                        |
|                            | F9                         | cancels selected running nodes                                                           |
|                            | Shift + F9                 | cancels all running nodes                                                                |
| Node Connections           | Ctrl + L                   | connects selected nodes                                                                  |
|                            | Ctrl + Shift + L           | disconnects selected nodes                                                               |
| Move Nodes and Annotations | Ctrl + Shift + Arrow       | moves the selected node in the arrow direction                                           |
|                            | Ctrl + Shift + PgUp/PgDown | moves the selected annotation in the front or in the back of all overlapping annotations |
| Workflow Operations        | F8                         | resets selected nodes                                                                    |
|                            | Ctrl + S                   | saves the workflow                                                                       |
|                            | Ctrl + Shift + S           | saves all open workflows                                                                 |
|                            | Ctrl + Shift + W           | closes all open workflows                                                                |
|                            | Ctrl + F                   | search workflow for nodes                                                                |
| Metanode                   | Shift + F12                | opens metanode wizard                                                                    |

# KNIME Community Hub



KNIME Hub

Welcome to the  
**KNIME Hub**

Solutions for data science: find workflows, nodes and components, and collaborate in spaces.

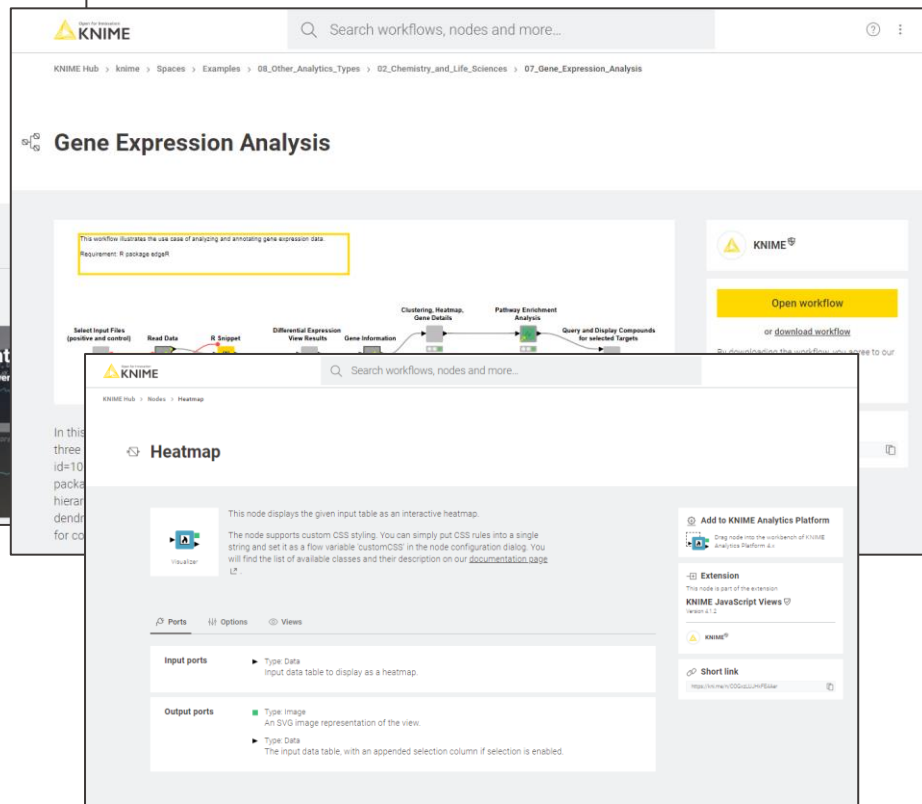
Search workflows, nodes and more...

4 011 Nodes    496 Components    4 645 Workflows    211 Extensions

Getting started  
**KNIME Examples**  
Workflows and more provided by KNIME

KNIME Space  
**Blueprints in the field of finance, accounting, and auditing**

Verified Component  
Easily reuse bundled functionalities, verified by KNIME Experts



KNIME Hub

Search workflows, nodes and more...

KNIME Hub > knime > Spaces > Examples > 08\_Other\_Analytics\_Types > 02\_Chemistry\_and\_Life\_Sciences > 07\_Gene\_Expression\_Analysis

## Gene Expression Analysis

This workflow illustrates the use case of analyzing and annotating gene expression data.  
Requirement: R package edgeR

Open workflow  
or download workflow

KNIME Hub > Nodes > Heatmap

### Heatmap

This node displays the given input table as an interactive heatmap.

The node supports custom CSS styling. You can simply put CSS rules into a single string and set it as a flow variable customCSS in the node configuration dialog. You will find the list of available classes and their description on our [documentation page](#).

Ports Options Views

**Input ports**

- Type Data  
Input data table to display as a heatmap.

**Output ports**

- Type Image  
An SVG image representation of the view.
- Type Data  
The input data table, with an appended selection column if selection is enabled.


Add to KNIME Analytics Platform  
Drag node into the workspace of KNIME Analytics Platform 4.0

Extension  
This node is part of the extension  
**KNIME JavaScript Views**  
Version 1.2

Short link  
[https://knime.com/CommunityFlow](#)

A place to share knowledge  
about Workflows and Nodes  
<https://hub.knime.com>

# The KNIME Community Hub

Open for Innovation

?

⋮

Welcome to the

**KNIME Hub**

The place to find and collaborate on KNIME workflows and nodes. Here you can find solutions for your data science questions.

🔍 heatmap

×

3 680

Nodes

196

Components

2 300


Workflows

196

Extensions

Extended KNIME Summit

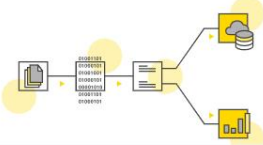
Online courses, workshops, and webinars during April and May



How to


**Getting started**

From downloading through to building your first workflow



Forum

Get help from our community and help others




# Searching Nodes and Workflows

The screenshot shows the KNIME Hub search interface. At the top, the KNIME logo is on the left, a search bar with the text 'heatmap' is in the center, and a close button (X) and a help icon (?) are on the right. Below the search bar, the text 'KNIME Hub > Search' is displayed. The main section shows '42 results'. Below this, there are tabs for 'All', 'Nodes', 'Components', 'Workflows', and 'Extensions'. The 'Workflows' tab is selected. The search results are listed in a table-like format. The second result, 'Gene Expression Analysis', is highlighted with a dashed yellow border. Each result includes a workflow icon, a title, tags, a description, a path, and a user profile picture.

| Workflow Icon | Title                                                 | Tags                                                               | Description                                                                                                                                                                                                                                                                                    | Path                                                                                                       | User Profile |
|---------------|-------------------------------------------------------|--------------------------------------------------------------------|------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------|------------------------------------------------------------------------------------------------------------|--------------|
|               | Visualization of screening data with HCS-Tools        | High-content screening                                             | The workflow shows how to use the 'Plate Heatmap Viewer' node from the HCS-Tools plugin for visualization                                                                                                                                                                                      | knime > Examples > 99_Community > 02_HCS_Tools > 01_Visualization_of_screening_data                        |              |
|               | Gene Expression Analysis                              | Life Sciences Bioinformatics R Shared components Interactive views | In this workflow, we analyze RNA-Seq data from tumors and matched normal tissue from three patients with oral squamous cell carcinomas ( <a href="https://journals.plos.org/plosone/article?id=10.1371/journal.po...">https://journals.plos.org/plosone/article?id=10.1371/journal.po...</a> ) | knime > Examples > 08_Other_Analytics_Types > 02_Chemistry_and_Life_Sciences > 07_Gene_Expression_Analysis |              |
|               | Create Heatmaps in KNIME and use R to add Dendrograms | knime heatmap dendrogram                                           | It is possible to create heatmaps with dendrograms with the help to R. You would have to check the code if the distance and cluster settings fit your needs. You can also export the numbers behind the...                                                                                     | m1auber71 > Public > kn_example_r_heatmap_cars                                                             |              |
|               | heatmap_test                                          |                                                                    |                                                                                                                                                                                                                                                                                                | jtyler > Public > heatmap_test                                                                             |              |

# Opening a Workflow from the Hub



Hub

Search workflows, nodes and more...

KNIME Hub > knime > Spaces > Examples > 08\_Other\_Analytics\_Types > 02\_Chemistry\_and\_Life\_Sciences > 07\_Gene\_Expression\_Analysis

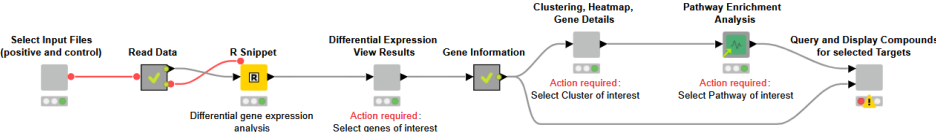
Workflow

## Gene Expression Analysis

Life Sciences | Bioinformatics | R | Shared components | Interactive views

Last edited: 24 Apr 2020

This workflow illustrates the use case of analyzing and annotating gene expression data.  
Requirement: R package edgeR



In this workflow, we analyze RNA-Seq data from tumors and matched normal tissue from three patients with oral squamous cell carcinomas (<https://journals.plos.org/plosone/article?id=10.1371/journal.pone.0009317>). Differentially expressed genes are discovered using the R package edgeR and are then displayed in an interactive view. Subsequently, genes are hierarchically clustered based on their expression pattern, and the results are shown via a dendrogram alongside a heatmap. We then perform a pathway enrichment analysis and look for compounds targeting the gene product of interest.

Short link

<https://kni.me/w/vUgr-iyGudXur-1B>

Used extensions & nodes

Legal

Discussion

# Edit the Workflow

The image displays the KNIME Analytics Platform interface with a workflow titled `*0: 02_StringManipulation_MathFormula_RuleEngine`. The workflow consists of four nodes: File Reader, Rule Engine, Math Formula, and String Manipulation, connected sequentially. A yellow box highlights the Rule Engine node, and a yellow callout bubble labeled "Drag & Drop" points to it.

On the left, a web browser window shows the KNIME Hub page for the "Row Filter" node. The page includes a search bar, the KNIME logo, and a description of the node's functionality. A yellow box highlights the "Manipulator" icon, and a yellow callout bubble labeled "Drag & Drop" points to it.

**String Manipulation, Math Formula and Rule Engine Example**

This workflow shows three different data manipulation operations, namely:

- creating three categories of people based on their weekly work hours with the Rule Engine node
- rounding up people's age to the nearest 10 with the Math Formula node
- replacing hyphens with " " characters in the native country column

**File Reader**  
Read adult data

**Rule Engine**  
New column for work status:  
- unemployed  
- employed part-time  
- employed full-time

**Math Formula**  
Round up age to the nearest 10

**String Manipulation**  
Replace "-" with " " in native country values

**Row Filter**

The node allows for row filtering a criteria. It can include or exclude: row number), rows with a certain a certain value in a selectable col are the steps on how to configure configuration dialog. Note: the domain of the data table. I. e. the upper and lower possible values in the table spec are not adapted, ever or one value is fully filtered out.

# Sharing the Workflow on the Hub

**1. Save your Edits**

**2. Connect to KNIME Hub**

**KNIME Explorer**

- My-KNIME-Hub (hub.knime.com)  
Double click to connect to KNIME Hub
- EXAMPLES (knime@hub.knime.com)
- LOCAL (Local Workspace)

**Node Repository**

- IO
- Manipulation
- Views
- Analytics
- DB
- Other Data Types
- Structured Data
- Scripting
- Tools & Services
- KNIME Labs
- Workflow Control
- Workflow Abstraction
- Social Media
- Reporting
- Chemistry

**String Manipulation, Math Formula and Rule Engine Example**

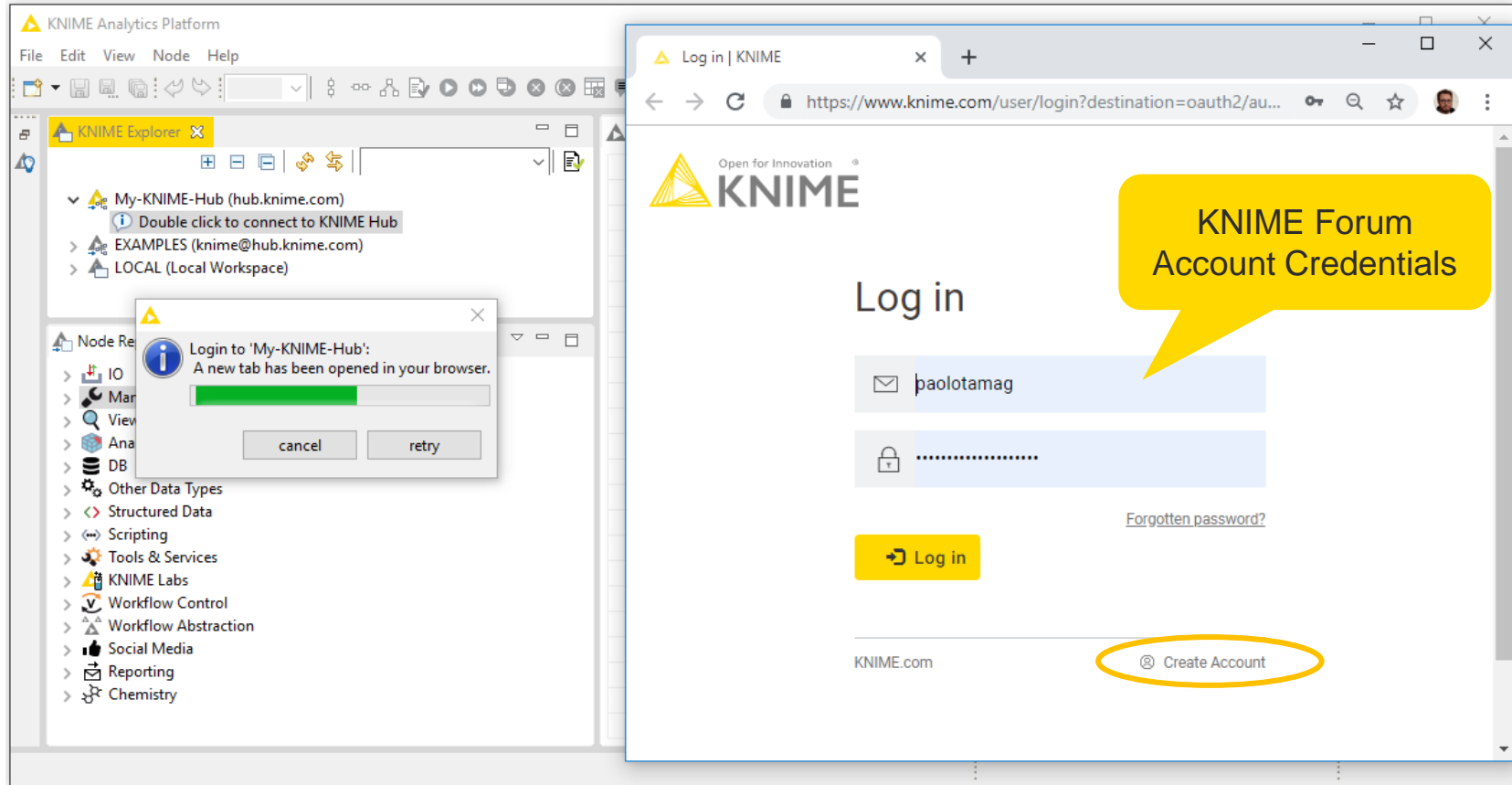
This workflow shows three different data manipulation operations, namely:

- creating three categories of people based on their weekly work hours with the Rule Engine node
- rounding up people's age to the nearest 10 with the Math Formula node
- replacing hyphens with " " characters in the native country column

**Workflow Steps:**

- File Reader**  
Read adult data
- Row Filter**  
filter data
- Rule Engine**  
New column for work status:
  - unemployed
  - employed part-time
  - employed full-time
- Math Formula**  
Round up age to the nearest 10
- String Manipulation**  
Replace "-" with " " in native country values

# Log in the Hub





# Stay connected with KNIME

---



**Blog:** [knime.com/blog](https://knime.com/blog)



**Forum:** [forum.knime.com](https://forum.knime.com)



**KNIME Hub:** [hub.knime.com](https://hub.knime.com)



**KNIME E-Learning Course:**  
[www.knime.com/e-learning-course](https://www.knime.com/e-learning-course)

Follow us on social media:



# Resources

---

- Self Paced E-Learning Data Science Courses
  - <https://www.knime.com/knime-self-paced-courses>
- KNIME certification
  - <https://www.knime.com/certification-program>
- KNIME Documentation
  - <https://www.docs.knime.com>
- YouTube KNIME TV
  - <https://www.youtube.com/user/KNIMETV>
- KNIME Events and instructor-lead courses
  - <https://www.knime.com/learning/events>
- KNIME Cheat Sheets
  - <https://www.knime.com/learning/cheatsheets>
- KNIME Books
  - <https://www.knime.com/knimepress>

**Thank You!**

**Questions? Please reach out**

[alice.krebs@knime.com](mailto:alice.krebs@knime.com)

[johannes.judt@knime.com](mailto:johannes.judt@knime.com)

