Open for Innovation
KNIME

KNIME Workbench Guide

KNIME AG, Zurich, Switzerland
Version 4.4 (last updated on 2022-12-21)

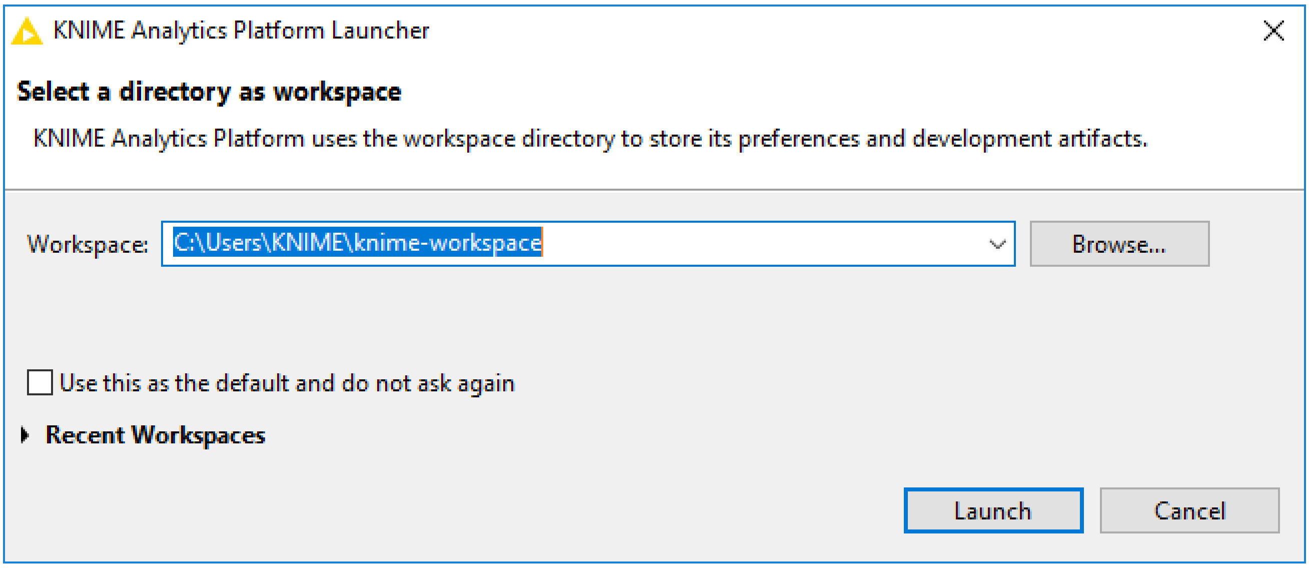
Table of Contents

This guide describes the first steps to take after starting KNIME Analytics Platform and points you to the resources available in the KNIME Workbench for building workflows. It also explains how to customize the workbench and configure KNIME Analytics Platform to best suit specific needs. In the last part of this guide we introduce data tables.

Workspaces

When you start KNIME Analytics Platform, the KNIME Analytics Platform launcher window appears and you are asked to define the KNIME workspace, as shown in Figure 1.

The KNIME workspace is a folder on the local computer to store KNIME workflows, node settings, and data produced by the workflow.



KNIME Analytics Platform Launcher

Select a directory as workspace

KNIME Analytics Platform uses the workspace directory to store its preferences and development artifacts.

Workspace: C:\Users\KNIME\knime-workspace

☐ Use this as the default and do not ask again

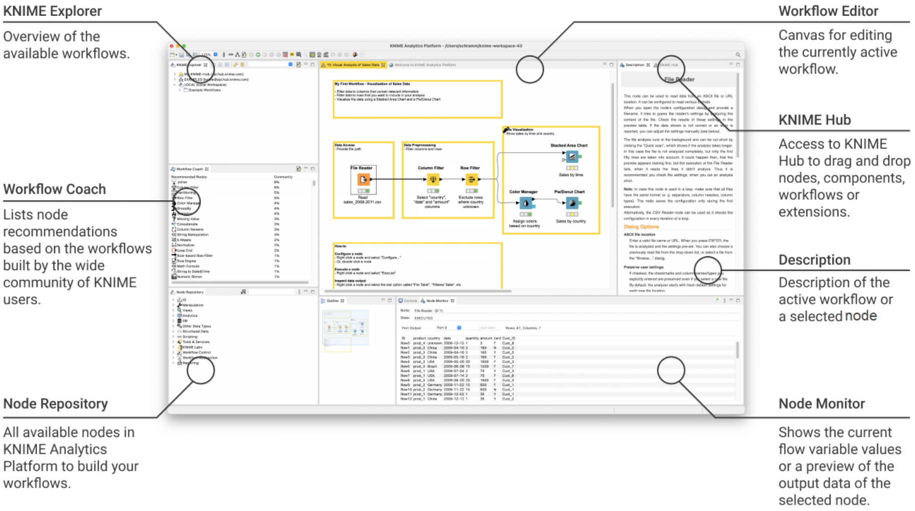
Figure 1. KNIME Analytics Platform launcher

The workflows and data stored in the workspace are available through the KNIME Explorer in the upper left corner of the KNIME Workbench.

KNIME Workbench

After selecting a workspace for the current project, click *Launch*. The KNIME Analytics Platform user interface - the KNIME Workbench - opens.

It is typically organized as shown in Figure 2.



KNIME Explorer

Overview of the available workflows.

Workflow Coach

Lists node recommendations based on the workflows built by the wide community of KNIME users.

Node Repository

All available nodes in KNIME Analytics Platform to build your workflows.

Workflow Editor

Canvas for editing the currently active workflow.

KNIME Hub

Access to KNIME Hub to drag and drop nodes, components, workflows or extensions.

Description

Description of the active workflow or a selected node

Node Monitor

Shows the current flow variable values or a preview of the output data of the selected node.

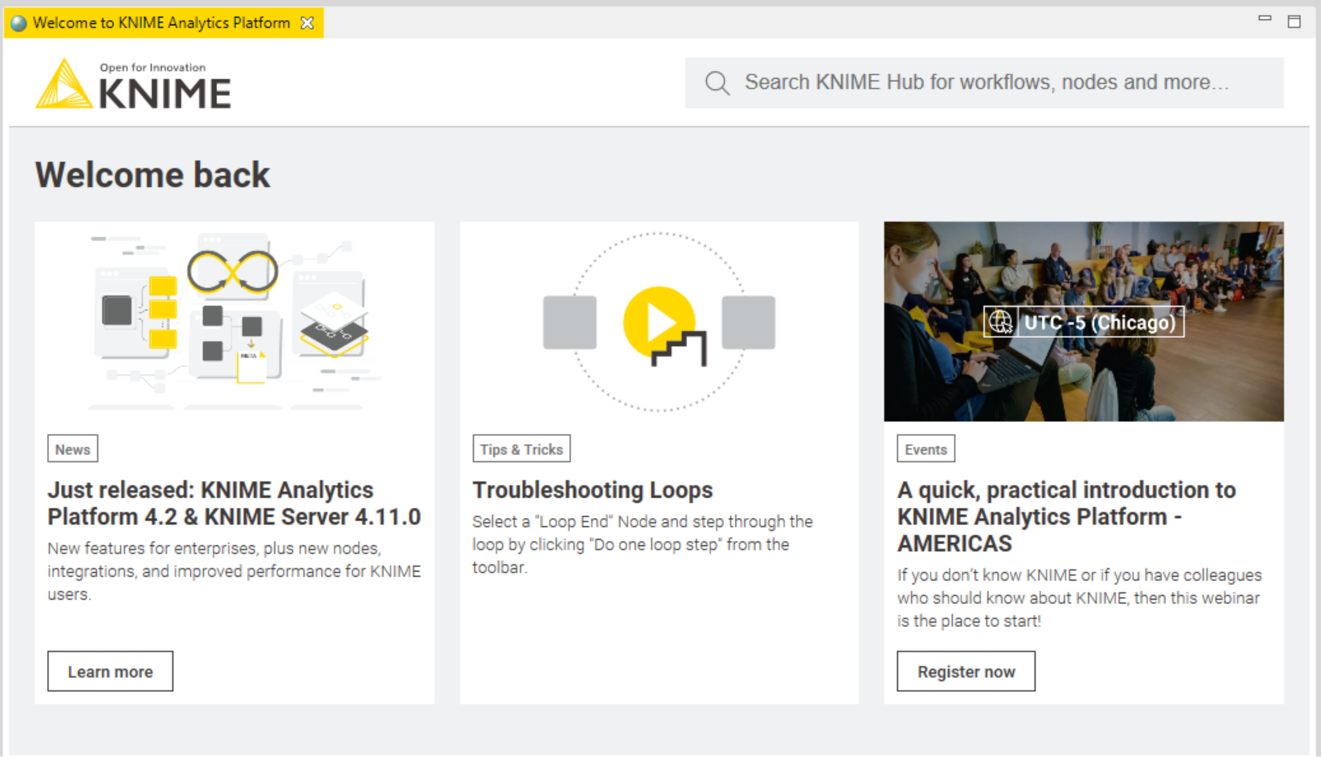
Figure 2. KNIME Workbench

In the next few sections we explain the functionality of these components of the workbench:

- Welcome Page
- Workflow Editor & nodes
- KNIME Explorer
- Workflow Coach
- Node Repository
- KNIME Hub view
- Description

Welcome page

The welcome page shown in Figure 3 is located in the middle of the KNIME Workbench.



Welcome to KNIME Analytics Platform
Open for Innovation
KNIME
Search KNIME Hub for workflows, nodes and more...
UTC -5 (Chicago)
Learn more
Register now

Welcome back

News

Just released: KNIME Analytics Platform 4.2 & KNIME Server 4.11.0

New features for enterprises, plus new nodes, integrations, and improved performance for KNIME users.

Tips & Tricks

Troubleshooting Loops

Select a "Loop End" Node and step through the loop by clicking "Do one loop step" from the toolbar.

Events

A quick, practical introduction to KNIME Analytics Platform - AMERICAS

If you don't know KNIME or if you have colleagues who should know about KNIME, then this webinar is the place to start!

Figure 3. Welcome page

This page links to information, for example available updates and the latest KNIME news, upcoming events, and tips and tricks.

After closing the welcome page, if no previously created workflows are available, you need to create an empty workflow editor, as explained in the next section.

Workflow editor & nodes

The workflow editor is where workflows are assembled. Workflows are made up of individual tasks, represented by nodes.

Create a new workflow editor going to *File* → *New…* and selecting the *New KNIME Workflow* option in the window that opens. Then click *Next* and give the new workflow a name in the field next to *Name of the workflow to create:* and click *Finish*. Other options are available as explained in the Building workflows section.

In the new empty workflow editor, create a workflow by dragging nodes from the node repository to the workflow editor, then connecting, configuring, and executing them.

Nodes

In KNIME Analytics Platform, individual tasks are represented by nodes. Nodes can perform all sorts of tasks, including reading/writing files, transforming data, training models, creating visualizations, and so on.

Facts about nodes

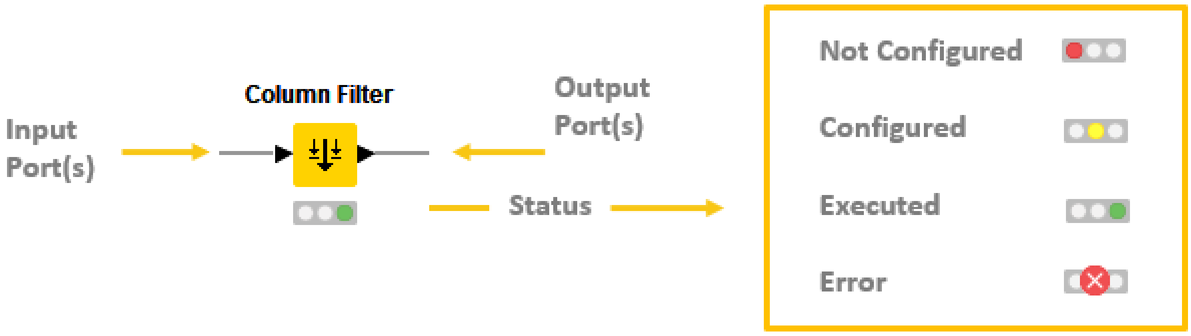


Figure 4. Node ports and node status

- Each node is displayed as a colored box with input and output ports, as well as a status, as shown in Figure 4
- The input port(s) hold the data that the node processes, and the output port(s) hold the resulting datasets of the operation
- The data is transferred over a connection from the output port of one to the input port of another node.

For simplicity we refer to data when we refer to node input and output ports, but nodes can also have input and output ports that hold a model, a database query, or another type explained in Node Ports.

Changing the status of a node

The status of a node can be changed, either configuring, executing, or resetting it. All these options can be found in the context menu of a node shown in Figure 5.

Open the context menu by right clicking a node. From the context menu it is also possible to open output tables and views, as well as copy nodes, along with some more advanced node options.

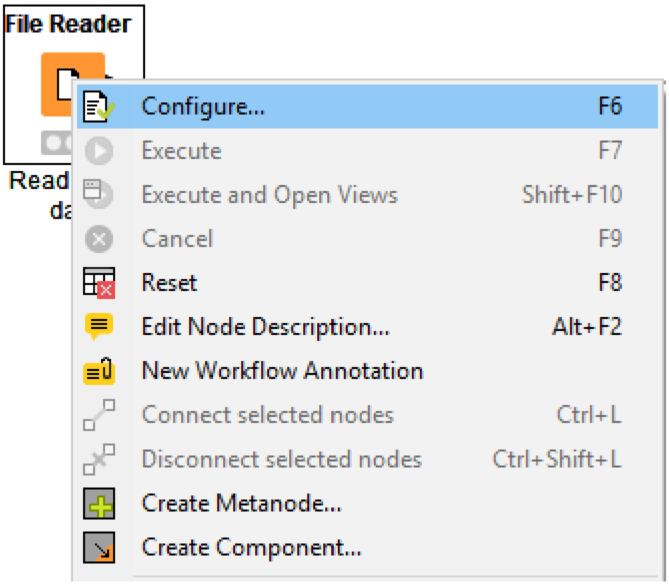

File Reader
Read
da
Configure... F6
Execute F7
Execute and Open Views Shift+F10
Cancel F9
Reset F8
Edit Node Description... Alt+F2
New Workflow Annotation
Connect selected nodes Ctrl+L
Disconnect selected nodes Ctrl+Shift+L
Create Metanode...
Create Component...

Figure 5. Context menu of a node

Identifying the node status

The traffic light below each node shows the status of the node. When a node is configured, the traffic light changes from red to yellow, i.e. from "not configured" to "configured".

When a new node is first added to the workflow editor, its status is "not configured" - shown by the red traffic light below the node.

Configuring the node

The node can be configured by adjusting the settings in its configuration dialog.

Open the configuration dialog of a node by either:

- Double clicking the node
- Right clicking a node and selecting *Configure…* in the context menu
- Or, selecting the node and pressing F6

In addition to the task specific settings, each node configuration dialog has a:

- "Memory Policy"-tab: here it is possible to define whether tables are attempted to be kept in memory, or if all tables are written to disk (see the section on In-Memory Caching for details).
- "Flow Variables"-tab: flow variables are explained in the Flow Control Guide

Executing the node

Some nodes have the status "configured" already when they are created. These nodes are executable without adjusting any of the default settings.

Execute a node by either:

- Right clicking the node and selecting *Execute*
- Or, selecting the node and pressing F7

If execution is successful, the node status becomes "executed", which corresponds to a green traffic light. If the execution fails, an error sign will be shown on the traffic light, and the node settings and inputs will have to be adjusted as necessary.

Right click the node and select one of the last options in the menu to inspect the outputs, such as data tables, and views of an executed node. If the node produces an (interactive) view in its output, like all JavaScript based nodes, select *(Interactive) View: …* from the context menu to open it.

Canceling execution of the node

To cancel the execution of a node, right click it and select *Cancel* or select it and press F9.

Resetting the node

To reset a node, right click it and select *Reset* or select it and press F8.

> ❗ Resetting a node also resets all of its subsequent nodes in the workflow. Now, the status of the node(s) turns from "executed" into "configured", the nodes' outputs are cleared.

Node ports

A node may have multiple input ports and multiple output ports. A collection of interconnected nodes, using the input ports on the left and output ports on the right, constitutes a workflow. The input ports consume the data from the output ports of the predecessor nodes, and the output ports provide data to the successor nodes in the workflow.

Besides data tables, input and output ports can provide other types of inputs and outputs. For each type the pair of input and output port looks different, as shown in Figure 6.

An output port can only be connected to an input port of the same type - data to data, model to model, and so on.

Some input ports can be empty, like the data input port of the Decision Tree View node in Figure 6. This means that the input is optional, and the node can be executed without the input. The mandatory inputs, shown by filled input ports, have to be provided to execute the node.

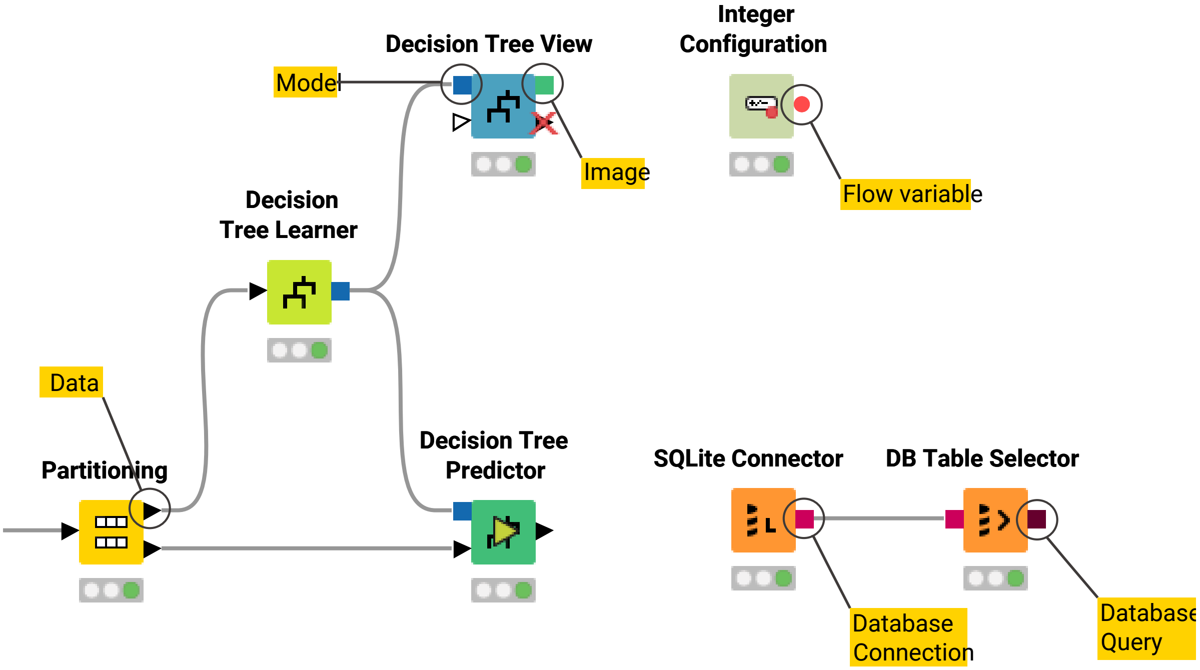


Figure 6. Common port types

A tooltip gives a short explanation of the input and output ports. If the node is executed, the dimensions of the outcoming data are shown in its data output port. A more detailed explanation of the input and output ports is in the node description.

How to select, move, copy, and replace nodes in a workflow

Nodes can be moved into the workflow editor by dragging and dropping them. To copy nodes between workflows, select the chosen nodes, right click the selection, and select *Copy* in the menu. In the destination workflow, right click the workflow editor, and select *Paste* in the menu.

To select a node in the workflow editor, click it once, and it will be surrounded by a border. To select multiple nodes, either press "Ctrl" and select nodes by mouse click, or draw a rectangle over the nodes with the mouse.

Replace a node by dragging a new node onto an existing node. Now the existing node will be covered with a colored box with an arrow and boxes inside as shown in Figure 7. Releasing the mouse replaces the node.

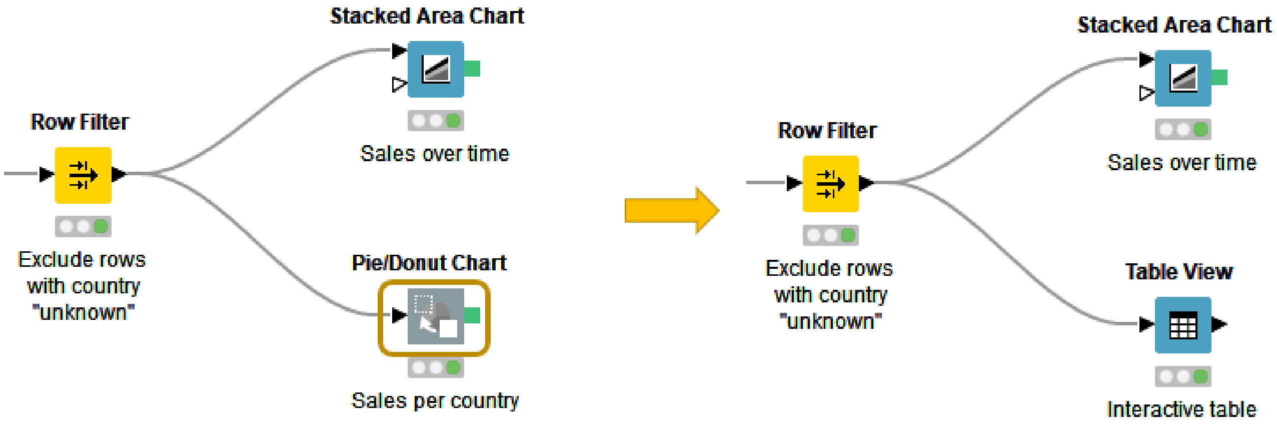


Figure 7. Replacing a node in a workflow

Comments and annotations

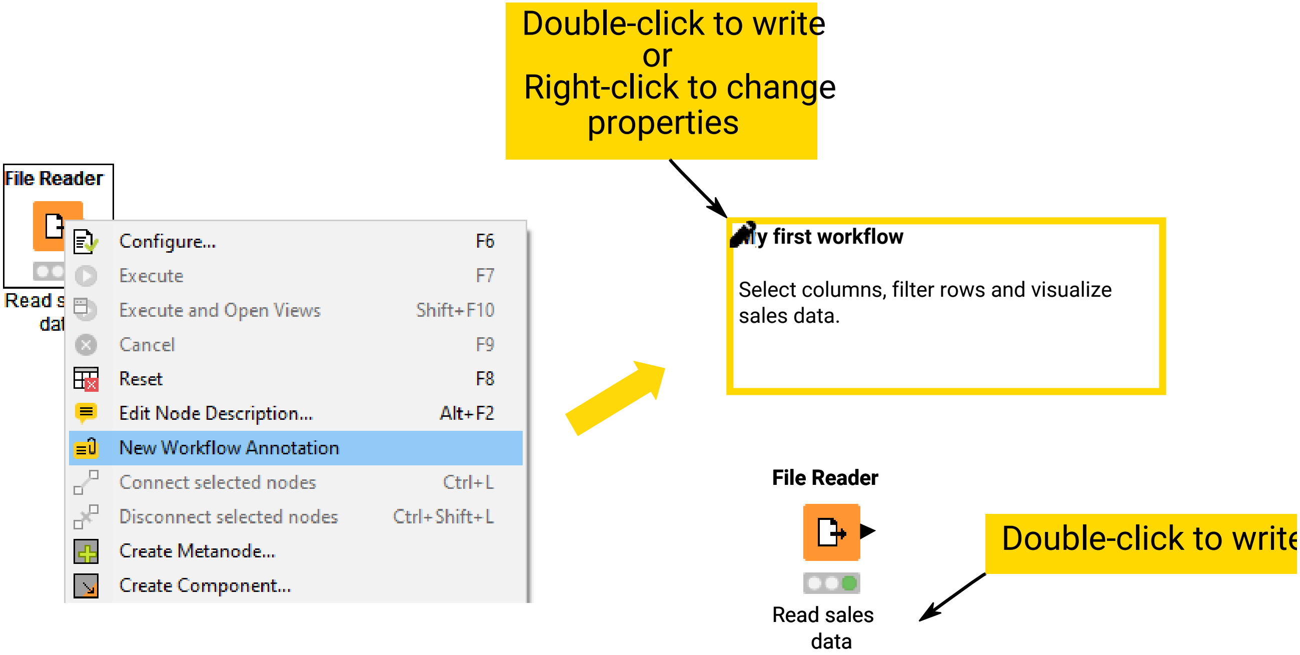
You have two options in the workflow editor to document a workflow (as shown in Figure 8):

- Add a comment to an individual node by double clicking the text field below the node and editing the text
- Add a general comment to the workflow, right click the workflow editor and select *New Workflow Annotation* in the menu. Now a yellow box will appear in the workflow editor.

You can do the following actions on the workflow annotation box.

- To move the workflow annotation box inside the workflow editor, first activate it from the top left corner, and then drag the box.
- To resize the box drag any of its edges.
- To edit the text inside double click the top left corner of the annotation box and type new text in the text field.

- To change the properties of the text and the border double click the top left corner and then right click inside the box. A menu opens showing the available editing options.



Double-click to write
or
Right-click to change properties
File Reader
Configure... F6
Execute F7
Execute and Open Views Shift+F10
Cancel F9
Reset F8
Edit Node Description... Alt+F2
New Workflow Annotation
Connect selected nodes Ctrl+L
Disconnect selected nodes Ctrl+Shift+L
Create Metanode...
Create Component...
My first workflow
Select columns, filter rows and visualize sales data.
File Reader
Read sales data
Double-click to write

Figure 8. Writing a node comment and creating a workflow annotation

> The video *Annotations & Comments* gives you a few hints about how to document a workflow.

Workflow editor settings

Change the visual properties of the workflow editor by clicking the "Workflow Editor Settings" button in the toolbar shown in Figure 9.

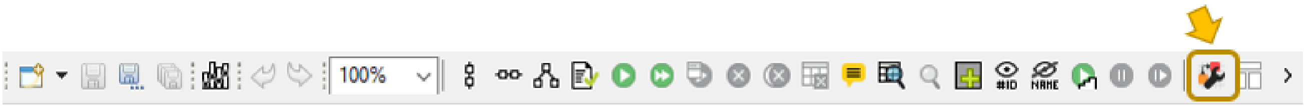


Figure 9. Changing visual properties of the workflow

In the dialog that opens you can change the size of the grid or remove the grid lines completely. You can also change the connection style from angular to curved, and make the connections thicker or narrower.

The changes will only apply to the currently active workflow editor. To change the default workflow editor settings, go to *File* → *Preferences* → *KNIME* → *KNIME GUI* → *Workflow Editor*.

Keyboard shortcuts

To view a full list of keyboard shortcuts, choose *Help* → *Show Active Keybindings* from the toolbar. Here, it is also possible to modify the bindings, and create personalized shortcuts.

KNIME Explorer

The KNIME Explorer is where you can manage workflows, workflow groups, and server connections. By default only the local workspace, the EXAMPLES server and the link to connect to your personal KNIME Hub space are visible in the KNIME Explorer.

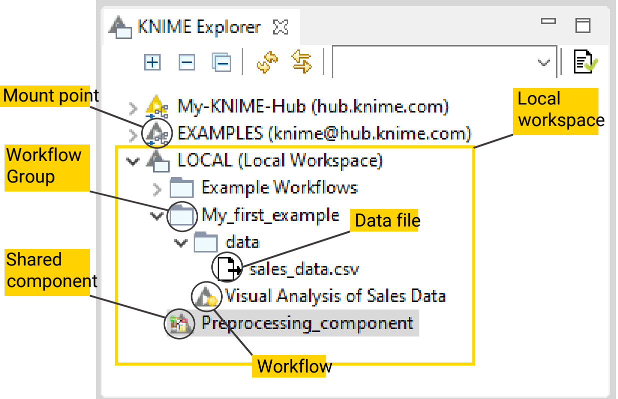


Figure 10. KNIME Explorer

Mount points

Mount points are workflow repositories that are accessible from KNIME Analytics Platform. They can be displayed as root directories in the KNIME Explorer view.

Each mount point consists of the location of the workflow repository, and a mount ID. For a local workflow repository, the location is the path to the folder, and for a server it is the address of the server. The mount ID is used to reference files and workflows under the mount point.

KNIME Explorer toolbar

At the top of the KNIME Explorer are several icons arranged in a toolbar shown in Figure 11.

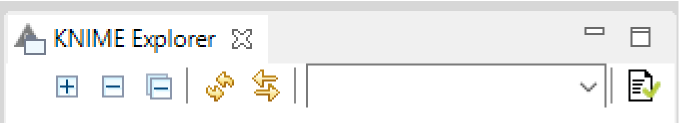


Figure 11. KNIME Explorer toolbar

The functions of the icons are explained in Table 1 below:

Table 1. Functions of the KNIME Explorer toolbar icons

⊞ ⊟ ⊟	• (+) expands the selected workflow group showing its content • (-) collapses the element • ⊟ collapses all elements in the KNIME Explorer showing only the mount points
⟳	Refreshes the view, in case it is out of sync with the underlying file system
⇆	Selects the workflow that is open in the workflow editor
	Add text to the field and press "Enter". The KNIME Explorer will only show items that contain the text in their name or are in a workflow group containing the text in its name.
📝	Opens the explorer preference page, allowing to add/remove/edit mount points

KNIME Explorer content

The types of content that you can see in the KNIME Explorer are described in Table 2.

Table 2. Types of Items in KNIME Explorer

▲	Workflow	A collection of nodes used to analyze data in KNIME
🗀	Workflow Group	A folder within the KNIME Explorer, which can be used to store workflows, data files, components, and metanodes.
🗋	Data File	Dragging a data file from the KNIME Explorer to the workflow editor automatically creates the correct node to read the file type. Storing data files in the currently active workspace allows for defining file paths relatively to their location in the KNIME Explorer.
▦	Component/metanode	Components and metanodes contain a pre-configured sub-workflow, which can be integrated in any part of a workflow. Components are nodes that encapsulate and abstract functionality. Metanodes, instead, are used to organize the workflow, collapsing part of it to hide that part of the workflow’s functionality.

ℹ A more comprehensive overview on components and metanodes is available in the KNIME Components Guide.

Explorer operations

Dragging and dropping elements

In the KNIME Explorer, elements can be moved between the repositories in the same way as in any other file explorer. Besides that, operations can be applied that affect the workflows stored in the KNIME Explorer: create nodes to read different file types and use a component or a metanode within a workflow. These operations are summarized in Table 3.

Table 3. Drag and Drop Operations in the KNIME Explorer

Move	To move an item, simply drag it and drop it to the desired location
Copy	Copying an item is the same process as moving it. Keep the "Ctrl"-key pressed during the drag and drop step. A small plus-sign next to the mouse cursor indicates the copy operation. Additionally, "Ctrl" + "c"/"v" shortcuts can be used to copy and paste elements from one repository to another.
Node creation	Drop a data file into the workflow editor. KNIME will create the appropriate file reading node automatically and preconfigures the node.
Component/metanode usage	A component or a metanode can be saved in the KNIME Explorer for later reuse. To do this, right click any component or metanode and select *Component* (or *Metanode*) → *Share…* . The resulting dialog gives you the possibility to choose a destination and the link type. To use a component or metanode stored in the KNIME Explorer, drag and drop it to the workflow editor.

Context menu

Other useful operations you can do in the KNIME Explorer are available in the context menu. Right click the KNIME Explorer or an item of the view to open the menu shown in Figure 12.

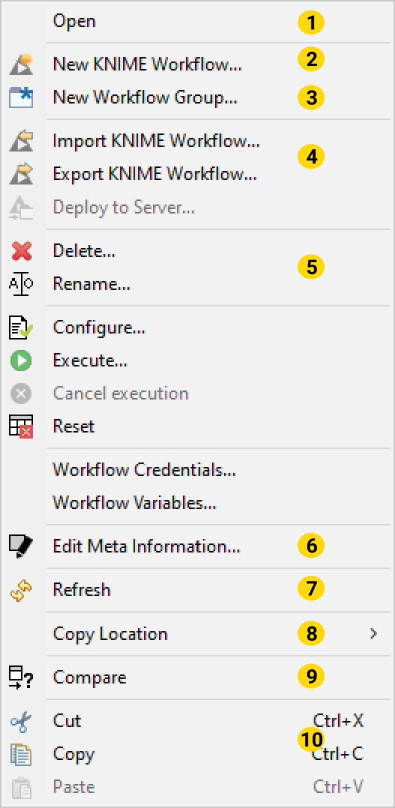


Figure 12. KNIME Explorer Context Menu

The menu items shown in Figure 12 are the operations available for workflows that are stored in your local workspace, without any remote workspace available. The possible operations are listed below:

1 Opens the workflow
2 Creates a new, empty workflow, places it in the selected workflow group and opens it in the workflow editor
3 Creates a new, empty workflow group and places it in the selected workflow group or directly under the "LOCAL" mount point
4 Opens the workflow import or export wizard
5 Deletes or renames the selected item. If a workflow is currently opened in the workflow editor, or a workflow group contains an open workflow, it is locked and cannot be renamed nor deleted.
6 Opens the meta information editor, where it is possible to write a description associated with the selected workflow or workflow group
7 Refreshes the workflow (group), in case it is out of sync with the underlying file system
8 Expands a menu for different file path types to copy the path to the item
9 Compares two selected items

10 Cuts/copies the selected item

Creating a new workflow

To create an empty workflow, right click anywhere in the local workspace, and select *New KNIME Workflow…* in the menu, or use one of the options explained in Building Workflows.
Give the workflow a name, and define the destination of the new workflow.
Click *Finish*, and the new workflow will appear in the selected workflow group in the KNIME Explorer.
To learn how to build a workflow, take a look at the next section Building Workflows, follow the steps in the Quickstart Guide, or check the video Workflows and Workflow Groups.

Building workflows

To create a workflow, you need an empty workflow editor. To create a new empty workflow editor take any of these actions:

- Navigate to *File* → *New…*, and select *New KNIME Workflow*
- Click the leftmost icon in the toolbar
- Right click in the local workspace and select *New KNIME Workflow…*

A workflow is built by dragging nodes from the node repository to the workflow editor and connecting them. To add a node from the node repository or from the workflow coach to the workflow editor, you have two options as shown in Figure 13:

- Drag and drop the node into the workflow editor
- Double click the node

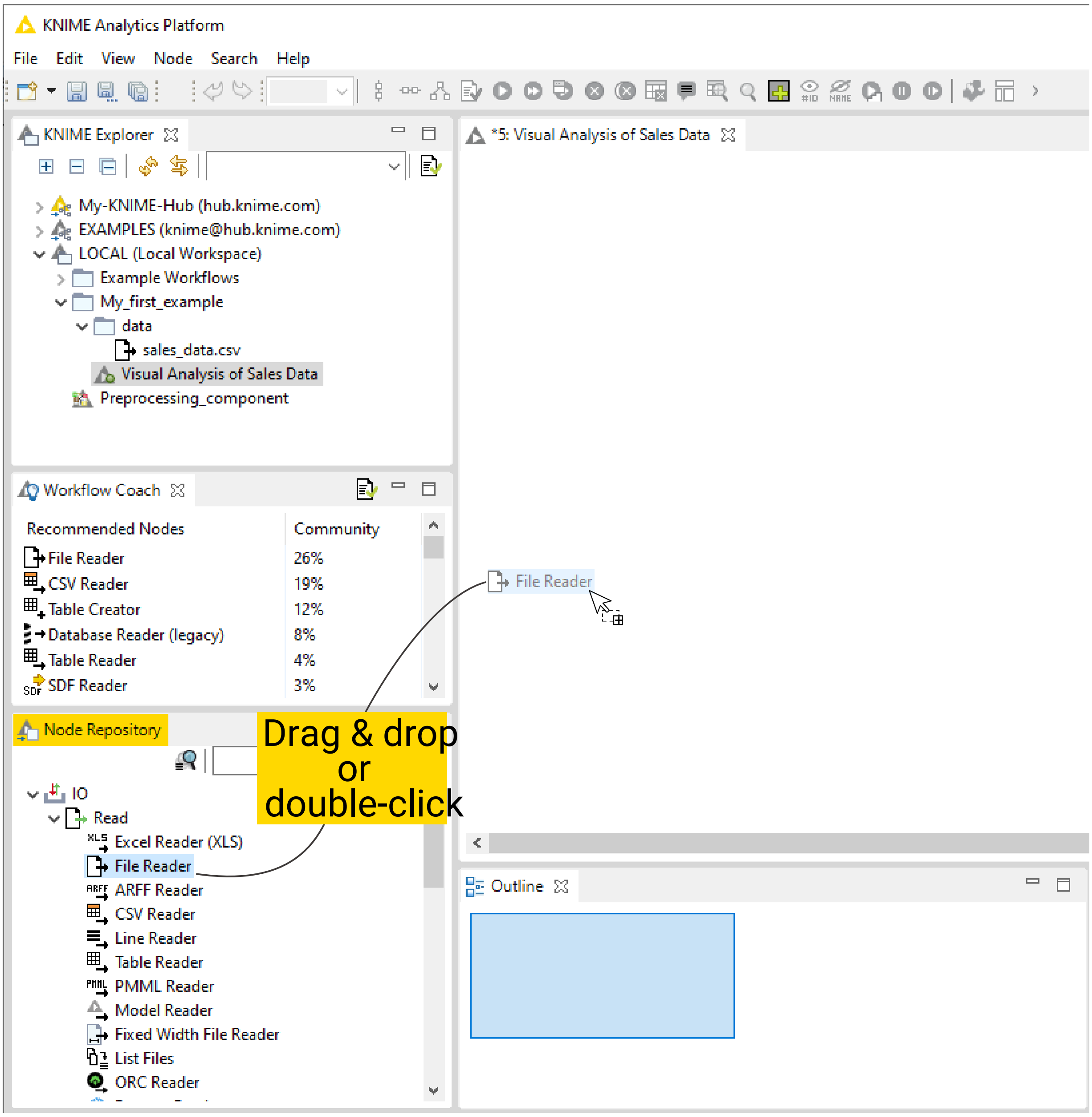


Figure 13. Adding a node into the workflow editor

Once two nodes are added to the workflow editor, they can be connected in any of these three ways:

- Click the output port of the first node and release the mouse at the input port of the second node. Now, the nodes are connected.
- Select a node in the workflow editor, and then double click the next node in the node repository. This double click creates a new node, and connects it to the selected node in the workflow editor.
- Select the nodes to connect in the workflow editor and press "Ctrl + L"

To add a node between two nodes in a workflow, drag the node from the node repository, and release it at its place in the workflow when the connector has turned red, as shown in Figure 14. The red connection means that it is ready to accept the new node. Release the mouse and the node is put in place.

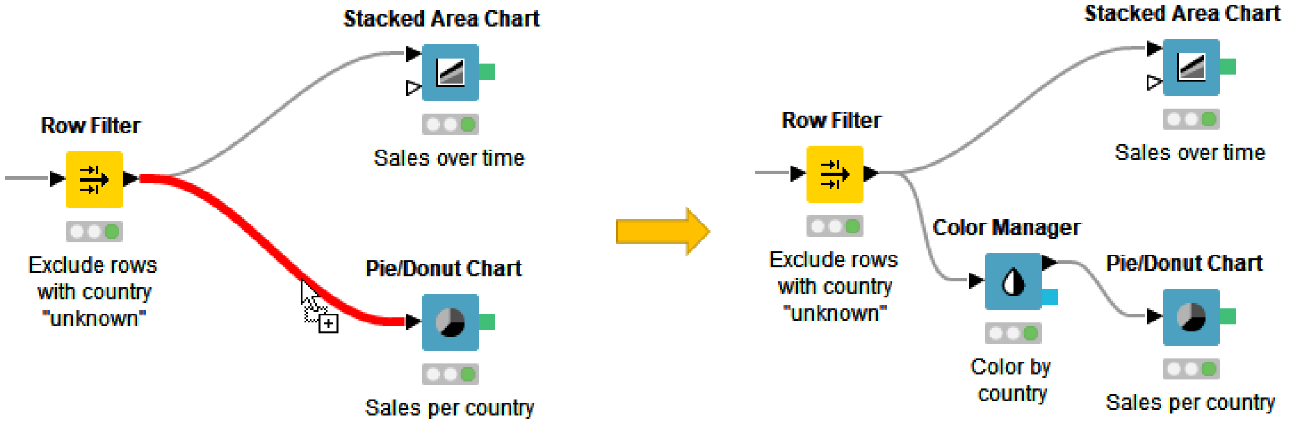


Figure 14. Add a node in the middle of a workflow

Workflow groups

Multiple workflows can be organized into workflow groups. Workflow groups are folders in the KNIME workspace that can include multiple workflows, as well as associated datafiles, shared components and metanodes, and even other workflow groups.

The workflow groups are in the currently active local workspace under the `LOCAL` mount point in the KNIME Explorer.

You have three ways to create a new, empty workflow group:

- Right click in the local workspace in the KNIME Explorer, and select *New Workflow Group…* in the menu
- Click the arrow next to the leftmost icon in the toolbar and select *New KNIME Workflow Group*
- Navigate to *File* → *New…*, select *New KNIME Workflow Group* in the list, and click *Next*.

In the dialog that opens, give the folder a name, and define where to save the folder in the local workspace. Click *Finish*. Now the new folder will appear in the selected destination in the KNIME Explorer.

Import/export workflows and workflow groups

You have three options to export a workflow or a workflow group:

- Export it as a file
- Save it into your personal KNIME Hub space, either in your public or in your private space
- Or, deploy it to a server (requires a license)

> To save workflows into your personal KNIME Hub space you need to first sign in. In the KNIME Explorer right click *My-KNIME-Hub (hub.knime.com)* and click *Connect to KNIME Hub*. Please, be aware that when saving a workflow group to your public folder on the KNIME Hub including data they become publicly available.

In the same way, you can import a workflow to your local workspace in the following ways:

- Import a file containing a workflow to your local workspace
- Save a workflow that is on a server to your local workspace. For example, you can access the EXAMPLES server (no credentials required) and save any workflow located there to your local workspace.

How to import and export a workflow (or workflow group)

You can import or export a workflow or a workflow group in the following ways:

- Right click anywhere in the local KNIME workspace, and select *Import(Export) KNIME Workflow...*, as shown in Figure 15.

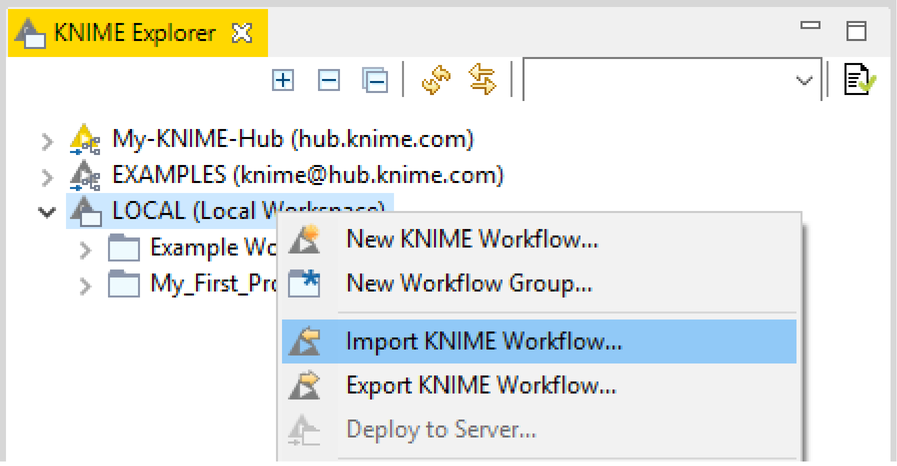


Figure 15. Importing and exporting workflow (groups)

- Go to *File* menu and select *Import (Export) KNIME Workflow…*

The dialog shown in Figure 16 opens.

Importing a workflow

In the upper part of the "Import" dialog, select the items to import, i.e define the file or folder path to import. In the "Destination" field underneath, define the destination folder in the KNIME workspace to import to.

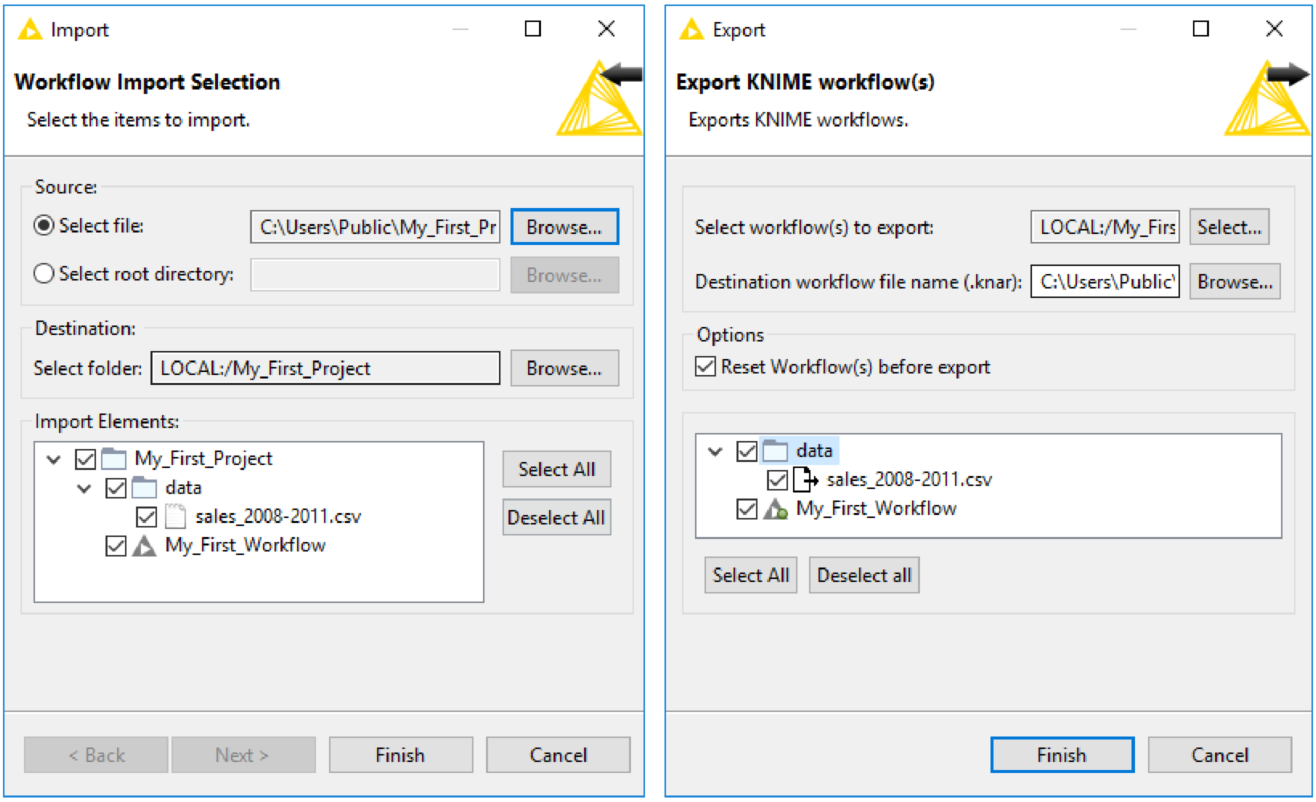
Importing a workflow group will show a list of elements inside the workflow group in the lower part of the dialog. Here you can select single elements to import them.

Exporting a workflow

In the upper part of the "Export" dialog, select the workflow (or group) to export. In the "Destination" field underneath, define the path to the destination folder on the local system, and the name of the file.

In "Options" you can choose to reset the workflow(s) before exporting. After resetting a node, the node status changes from "executed" to "configured", and the output of the node is no longer available. When exporting a workflow in an executed state, the data used in the workflow are exported as well. See the section on Reset and Logging for more information.

When exporting a workflow group, you can select the elements that you want the exported file to contain.



Import

Workflow Import Selection

Select the items to import.

Source:

● Select file: C:\Users\Public\My_First_Pr [Browse...]

○ Select root directory: [Browse...]

Destination:

Select folder: LOCAL:/My_First_Project [Browse...]

Import Elements:

- ☑ My_First_Project
  - ☑ data
    - ☑ sales_2008-2011.csv
  - ☑ My_First_Workflow

[Select All] [Deselect All]

[< Back] [Next >] [Finish] [Cancel]

Export

Export KNIME workflow(s)

Exports KNIME workflows.

Select workflow(s) to export: LOCAL:/My_Firs [Select...]

Destination workflow file name (.knar): C:\Users\Public\ [Browse...]

Options

☑ Reset Workflow(s) before export

- ☑ data
  - ☑ sales_2008-2011.csv
- ☑ My_First_Workflow

[Select All] [Deselect all]

[Finish] [Cancel]

Figure 16. Defining settings for importing and exporting workflow (groups)

> Importing and exporting workflows are also introduced in this video: *Import/Export Workflows*.

- The file extension for a KNIME workflow, is `.knwf` (KNIME workflow file)
- The file extension for a workflow group, is `.knar` (KNIME archive file)

KNIME Workflow Comparison

The Workflow Comparison feature provides tools and views that compare workflow structures and node settings. Workflow Difference allows you to view changes in different versions of a workflow. The feature allows users to spot insertions, deletions, substitutions or similar/combined changes of nodes. The node settings comparison makes it possible to track changes in the configuration of a node.

A Workflow Comparison can be triggered from every view that shows multiple workflows, e.g. KNIME Explorer and Server History.

In order to compare two workflows, select them in the KNIME Explorer, with "Ctrl"+click, and select *Compare* from the context menu, as shown in Figure 17.

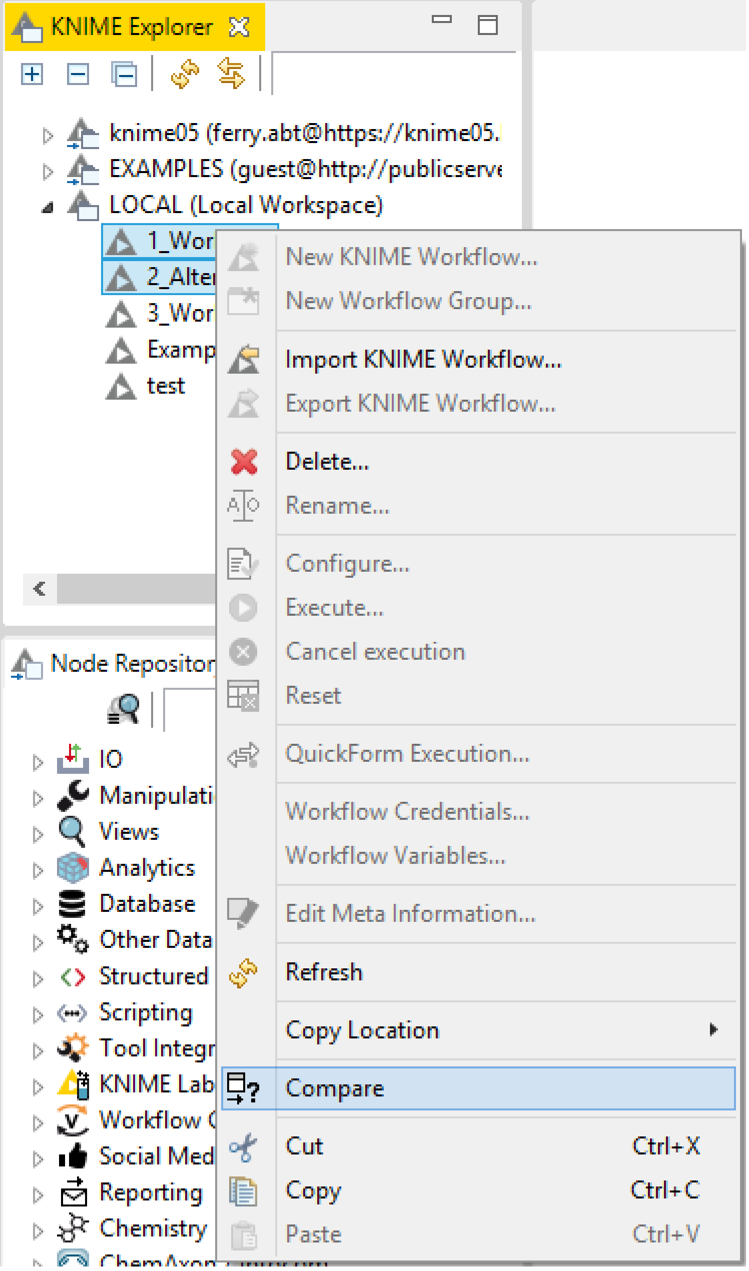


Figure 17. Compare two workflows in KNIME Explorer

It is also possible to compare a workflow with itself. With this option, you are given a list of nodes from which you can choose the two nodes you want to compare (see Node comparison section below).

The comparison of two workflows or of a workflow with itself, opens a tab in the KNIME Workbench, as shown in Figure 18.

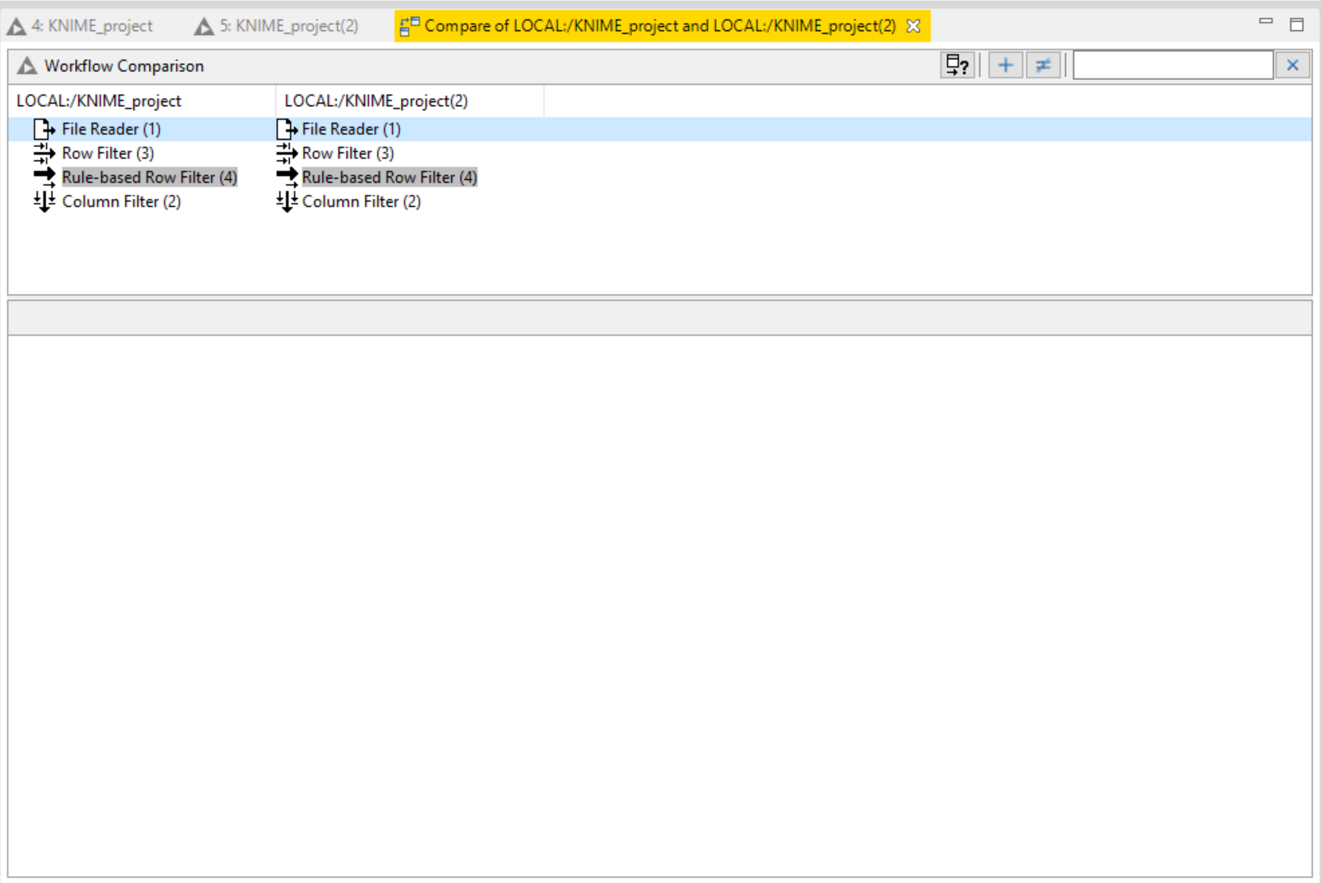


Figure 18. The Workflow Comparison tab

To make it easier to see which changes have been identified there are three buttons in the upper right corner of the Workflow Comparison view.

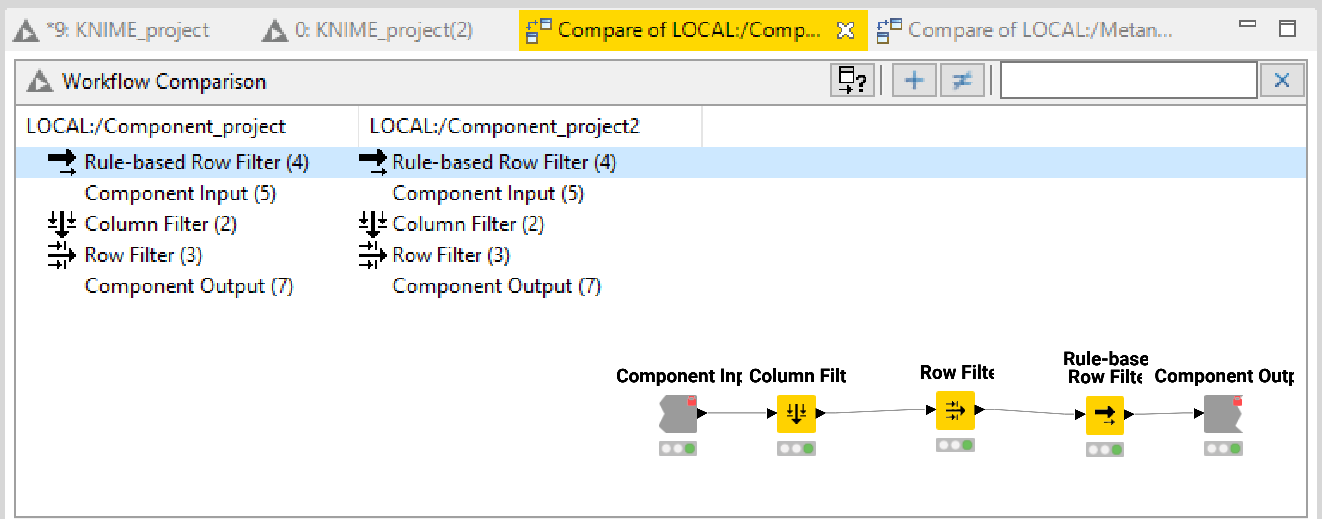
These buttons (from left to right) filter the list to:

- Perform a node comparison of the selected nodes (see Node comparison section below)
- Show the added or removed nodes only
- Hide the nodes with equal settings

Additionally you can use the search field, to check a special node or node type for changes. The last icon clears the search field and displays all nodes.

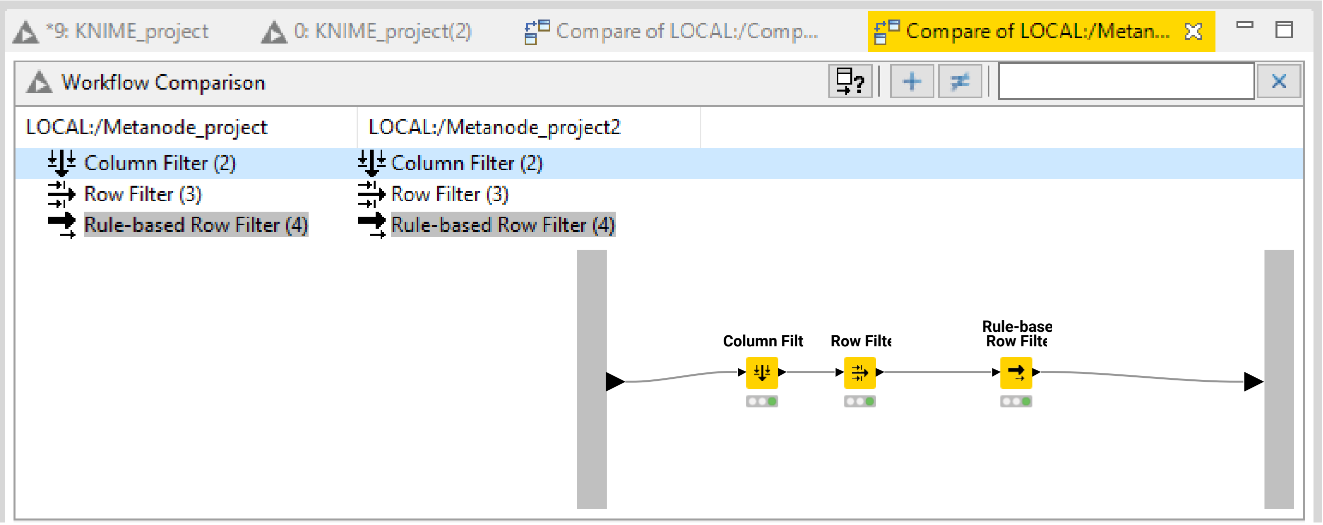
Workflow Comparison is structure based. This means, that not only workflows, but also components, snapshots, metanodes, i.e. all items that act as a workflow can be compared with each other. This basically includes everything that can be seen in KNIME Explorer and Server History (except data).

Workflow comparison focuses on the functional structure of a workflow. When comparing shared components (Figure 19) or metanodes (Figure 20) they are expanded.



LOCAL:/Component_project	LOCAL:/Component_project2
Rule-based Row Filter (4)	Rule-based Row Filter (4)
Component Input (5)	Component Input (5)
Column Filter (2)	Column Filter (2)
Row Filter (3)	Row Filter (3)
Component Output (7)	Component Output (7)

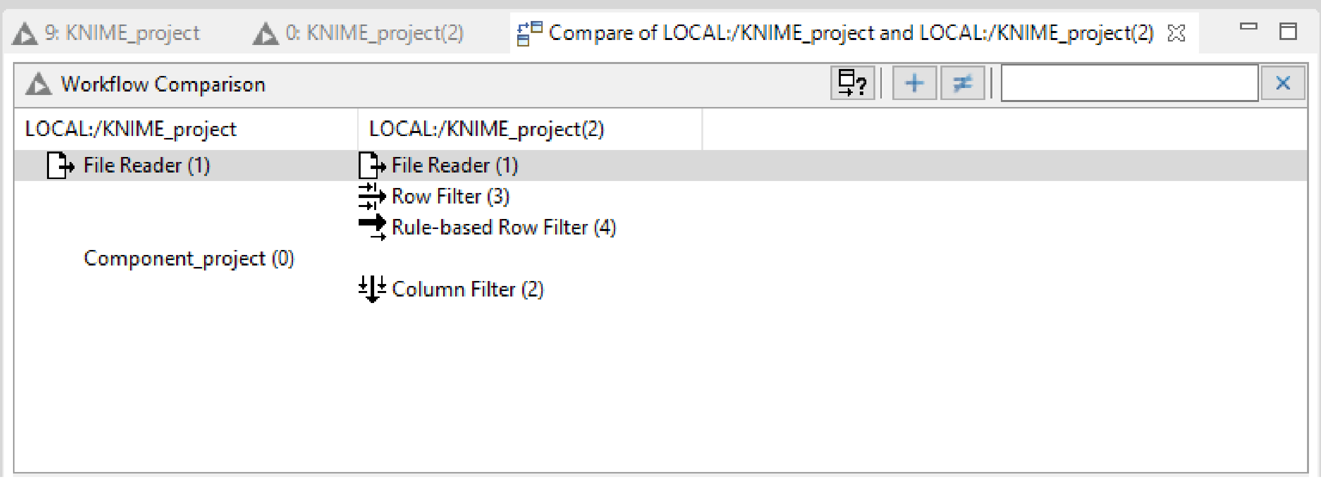
Figure 19. Components comparison with an expanded component for reference



LOCAL:/Metanode_project	LOCAL:/Metanode_project2
Column Filter (2)	Column Filter (2)
Row Filter (3)	Row Filter (3)
Rule-based Row Filter (4)	Rule-based Row Filter (4)

Figure 20. Metanodes comparison with an expanded metanode for reference

Components inside a workflow, when comparing workflows, on the other hand, are treated as normal nodes, and their content does not appear in the view. This is not true instead for metanodes inside a workflow that are expanded when comparing the workflow. In Figure 21, the comparison between two workflows containing a component and a metanode, respectively, is shown.



LOCAL:/KNIME_project	LOCAL:/KNIME_project(2)
File Reader (1)	File Reader (1)
	Row Filter (3)
	Rule-based Row Filter (4)
Component_project (0)	
	Column Filter (2)

Figure 21. Workflow comparison of two workflows containing a component and a metanode

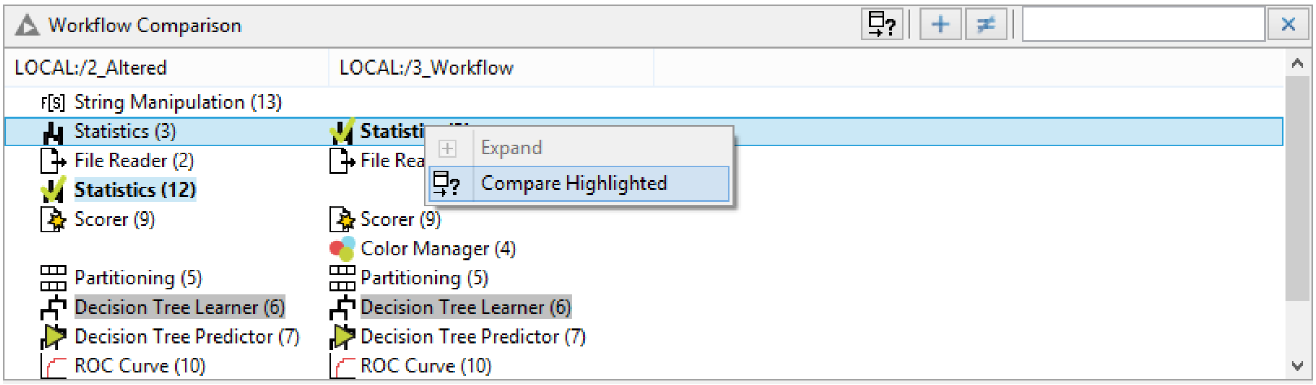
On the left column the component contained in the first workflow is shown as a node while the metanode contained in the second workflow, on the right column, is expanded.

If a component has been changed, it is highlighted in the Workflow Compare view, and the node comparison will show two additional entries: "Component Content Hash" and "Component Internal Settings Hash". These two numbers change whenever the content of the Component (e.g. node insertion/substitution) or the settings of an internal node change, respectively.

> The structural comparison is based on a sequential alignment with respect to the attributes (neighborhood, settings, etc.) of a node. It is designed to identify changes in a workflow. It might still be used to find similarities/common parts in any two workflows, however the usefulness of the results of these comparisons is often limited.

Node Comparison

Node Comparison is an additional view to Workflow Comparison. To compare two nodes select the nodes in the Workflow Comparison view and then click either the first button in the view or right-click and select *Compare Highlighted* in the context menu.



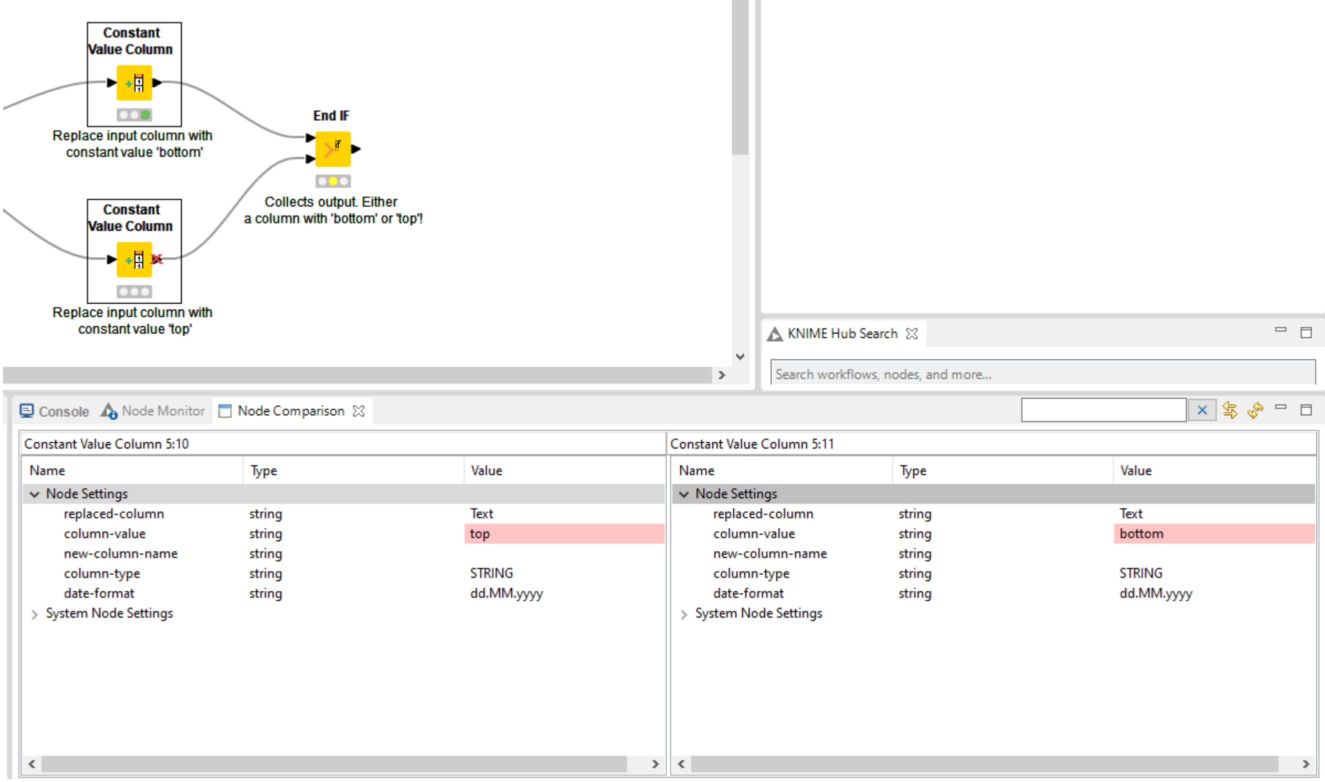
LOCAL:/2_Altered	LOCAL:/3_Workflow
String Manipulation (13)	
Statistics (3)	Statisti
File Reader (2)	File Rea
Statistics (12)	
Scorer (9)	Scorer (9)
	Color Manager (4)
Partitioning (5)	Partitioning (5)
Decision Tree Learner (6)	Decision Tree Learner (6)
Decision Tree Predictor (7)	Decision Tree Predictor (7)
ROC Curve (10)	ROC Curve (10)

The selected nodes are written in bold and include a green checkmark on the icon.

Node Comparison shows the settings of the nodes in two trees. Differing values and settings, which are not present in both nodes, are highlighted in red. If a changed setting is nested, the parent setting is highlighted in gray to indicate the hidden change. Click the arrow to expand the setting and show all nested settings.

To look for a specific setting the user can type the name into the search field in the upper right corner. This filters the list to show only those settings that contain the search query.

To compare the settings of two nodes in the same workflow (for example to compare two similar branches) the user can either compare the workflow with itself to retrieve the list of nodes, or open the workflow in the editor, select the two nodes of interest, right-click and select *Compare Nodes* in the context menu. This opens the Node Comparison view as a tab close to Console and Node Monitor views as shown in Figure 22. This view is identical to the one in Workflow Comparison, but is an independent view.


Constant Value Column
Replace input column with constant value 'bottom'
End IF
Collects output. Either a column with 'bottom' or 'top'!
Constant Value Column
Replace input column with constant value 'top'
KNIME Hub Search
Search workflows, nodes, and more...
Console
Node Monitor
Node Comparison

Constant Value Column 5:10

Name	Type	Value
∨ Node Settings		
replaced-column	string	Text
column-value	string	top
new-column-name	string	
column-type	string	STRING
date-format	string	dd.MM.yyyy
› System Node Settings		

Constant Value Column 5:11

Name	Type	Value
∨ Node Settings		
replaced-column	string	Text
column-value	string	bottom
new-column-name	string	
column-type	string	STRING
date-format	string	dd.MM.yyyy
› System Node Settings		

Figure 22. The Node Comparison view

A subtle but powerful difference between Node Comparison in Workflow Comparison and the independent view is the toolbar, which for the independent view has two additional buttons. The right button refreshes the view, retrieving the settings from the workflow again. This is useful for example if you find a difference between two nodes which should actually be identical. After identifying and changing the setting, click the refresh button to show the new settings and confirm the new equality. The second button enables you to find the compared nodes in the workflow. If the workflow is still open in an editor, the nodes are selected and scrolled into the viewport.

knime:// protocol

❗ Please note that starting from KNIME Analytics Platform version 4.3 most of the nodes have been updated to work with a new File Handling framework and the below described `knime::/` protocol has been substituted by a new way of addressing standard file systems. For an overview on how to work with the KNIME URLs please refer to the KNIME File Handling Guide.

`knime://` protocol is a protocol specific to KNIME that allows to specify file paths relatively to the KNIME workspace or even the location of the currently executing workflow.

The first element in the file path after `knime://` is the base for the path. It is either the workflow itself, the current mount point or a specific mount point like `LOCAL` in the following example:

```
knime://LOCAL/My_First_Project/data/sales_2008-2011.csv
```

ℹ The portable file path options are explained in the subsections below and in this video: The knime:// Protocol.

Absolute URLs

Absolute URLs are defined relative to a specific mount point. The following file path is defined using the absolute path to the file based on the mount point `LOCAL`:

```
knime://LOCAL/My_First_Project/data/sales_2008-2011.csv
```

The file path would now work on any system where the workflow is saved in the local workspace, and the file path inside the local workspace folder is the same.

Mountpoint-relative URLs

Because of the `LOCAL` term in the absolute path, accessing the file with the absolute URL is not possible, if the workflow is deployed to a server.

To enable access to a data file both locally and on a server, select the path to the file relative to the currently active mount point.

To do this, change the `LOCAL` term in the file path to `knime.mountpoint` as in this file path:

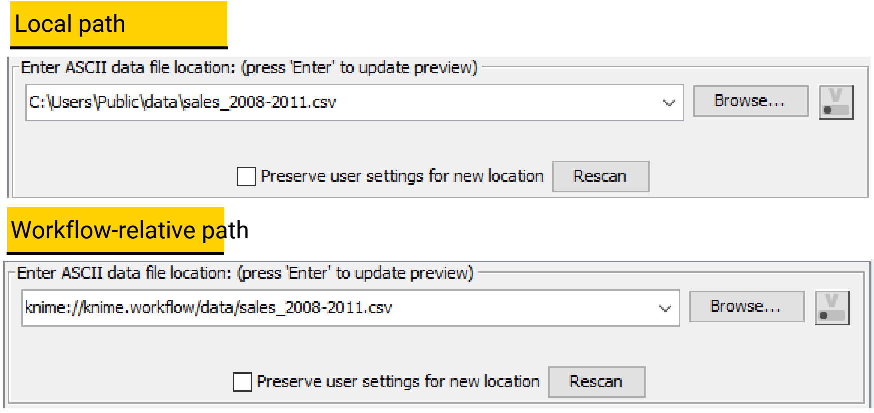
```
knime://knime.mountpoint/My_First_Project/data/sales_2008-2011.csv
```

In the mountpoint-relative file path, the `knime.mountpoint` refers to the uppermost folder level, which can be `LOCAL` or the mount ID of a server.

Workflow-relative path

The most flexible portable file path is the workflow-relative path. A workflow-relative path defines the file path relative to the currently executing workflow. Using this file path you can access data files in workflows in local workspaces on different systems, or on a server, as long as the folder structure between the workflow and the data file is the same.

Compared to the absolute path and mountpoint-relative path, the name of the folder containing the workflow does not have to be the same in the different locations. That's because an upper folder level is denoted by `/../` instead of the name of the folder.



Local path

Enter ASCII data file location: (press 'Enter' to update preview)

C:\Users\Public\data\sales_2008-2011.csv

□ Preserve user settings for new location

Workflow-relative path

Enter ASCII data file location: (press 'Enter' to update preview)

knime://knime.workflow/data/sales_2008-2011.csv

□ Preserve user settings for new location

Figure 23. Workflow-relative file path

Save workflows with data

> ❗ Please note that starting from KNIME Analytics Platform version 4.3 most of the nodes have been updated to work with a new File Handling framework and the below described process has been substituted by a new way of addressing standard file systems. For example, with the new File Handling framework nodes xou can easily save your workflows with data by using a Writer node and choosing the write to *Relative to* > *Current workflow data area* output location when configuring the node.

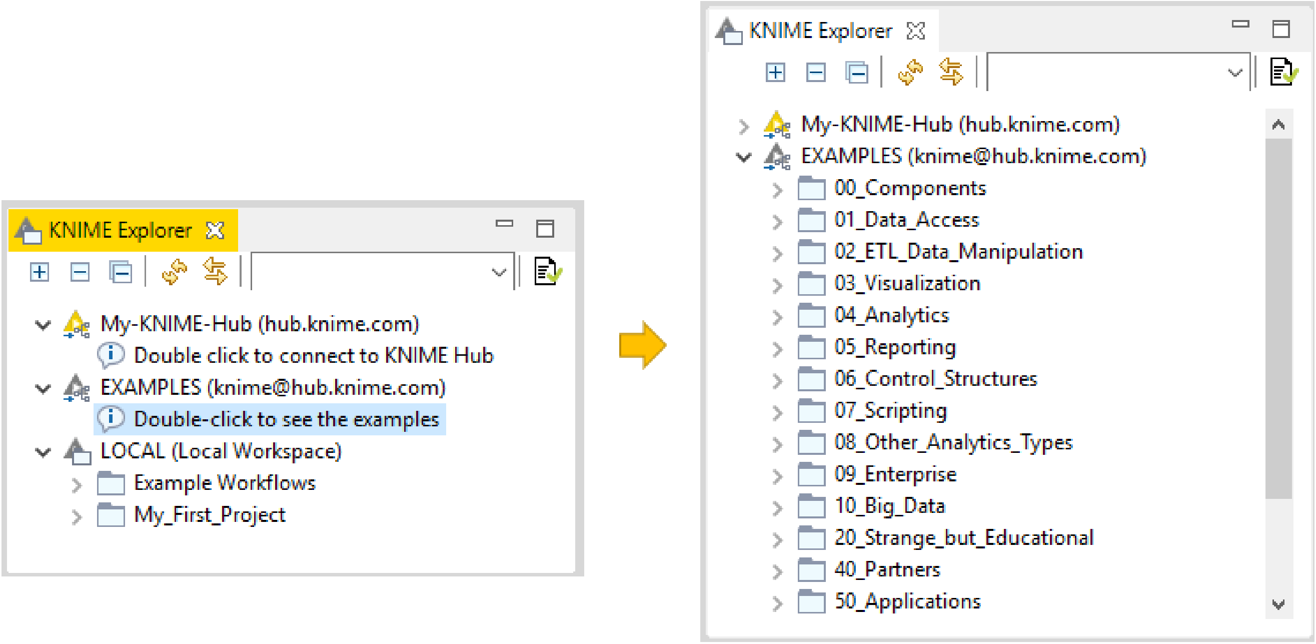
You can easily include data into your workflow by using the workflow-relative paths as described above. First, access the workflow in your KNIME workspace from your operating system, then manually create a folder called `data`, and place your data inside this folder. In this way you can easily reference your data within nodes using the workflow-relative path, which makes sure that your data will remain with your workflow whenever you archive it, export it, or upload it to a KNIME Server or the KNIME Hub.

EXAMPLES server

You can explore the example workflows, which includes also some real-world use cases, on the public EXAMPLES server.

Inspect the workflow groups for different categories by expanding the EXAMPLES mount point in the KNIME Explorer, and then double clicking the text below as shown in Figure 24.

You can download an example workflow by drag and drop, or copy and paste of the workflow into the local workspace. Double click the downloaded copy of the example workflow to open and edit it like any other workflow.
Alternatively, double click the example workflow directly on the EXAMPLES server to open it in the workflow editor. Save it to the local workspace via "*File*" and then "*Save As…*".



KNIME Explorer

My-KNIME-Hub (hub.knime.com)
Double click to connect to KNIME Hub
EXAMPLES (knime@hub.knime.com)
Double-click to see the examples
LOCAL (Local Workspace)
Example Workflows
My_First_Project

KNIME Explorer

My-KNIME-Hub (hub.knime.com)
EXAMPLES (knime@hub.knime.com)
00_Components
01_Data_Access
02_ETL_Data_Manipulation
03_Visualization
04_Analytics
05_Reporting
06_Control_Structures
07_Scripting
08_Other_Analytics_Types
09_Enterprise
10_Big_Data
20_Strange_but_Educational
40_Partners
50_Applications

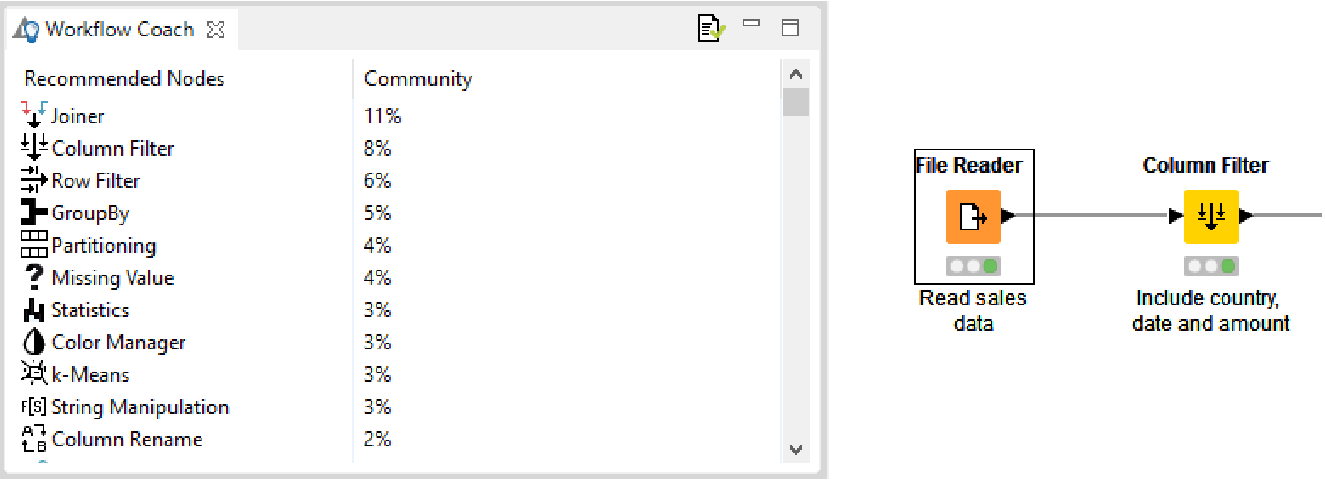
Figure 24. Example workflow available on the EXAMPLES server

> The video The EXAMPLES server provides a more detailed introduction to the EXAMPLES server.

Workflow Coach

The workflow coach shown in Figure 25 provides node recommendations. If a node is selected in the workflow editor, the workflow coach shows the most popular nodes to follow the selected node. Otherwise, the recommendations represent the most popular nodes to start a workflow.

The recommendations are based on KNIME community usage statistics about workflows built in KNIME Analytics Platform. Nodes can be added from the workflow coach to the workflow editor in the same way as from the node repository, by drag and drop, or by a double click.



Recommended Nodes	Community
Joiner	11%
Column Filter	8%
Row Filter	6%
GroupBy	5%
Partitioning	4%
Missing Value	4%
Statistics	3%
Color Manager	3%
k-Means	3%
String Manipulation	3%
Column Rename	2%

Figure 25. Workflow Coach

Note: start or stop sending anonymous usage data any time by checking the option *Yes, help improve KNIME.* in the "KNIME" dialog in Preferences.

Customizing node recommendations

Customize the node recommendations in the "Workflow Coach" dialog, under *File* → *Preferences* → *KNIME* → *Workflow Coach*. You have the following three options:

- Add node recommendations based on workflows in the currently active local workspace by enabling the *Workspace Node Recommendations* option in the "Workspace Recommendations" dialog
- Add node recommendations based on the workflows on a server by selecting the KNIME Server in the "Server Recommendations" dialog
- Disable the node recommendations by the community by unchecking the *Node Recommendations by the Community* option in the "Workflow Coach" dialog

The video Workflow Coach: The Wisdom of the KNIME Crowd provides a more detailed introduction to node recommendations.

Favorite Nodes

The Favorite Nodes view displays your favorite, most frequently used and last used nodes. To add this go to `View > Others` and under `KNIME Views` select `Favorite Nodes`. A node is added to your favorites by dragging it from the node repository into the personal favorite nodes category. Whenever a node is dragged onto the workflow editor, the last used and most frequently used categories are updated. Refer to Figure 26.

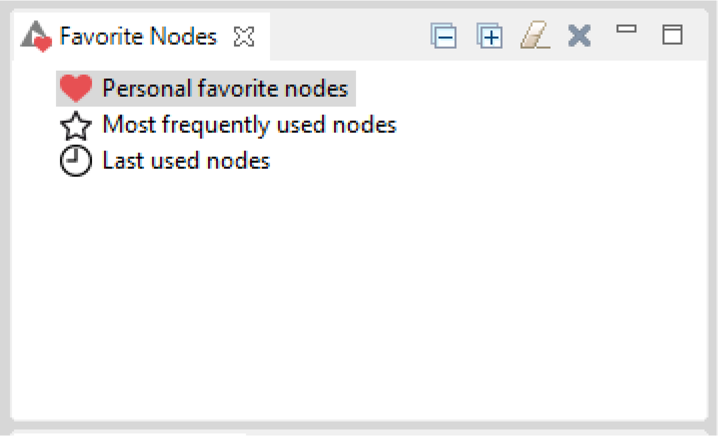


Figure 26. Favorite Nodes view

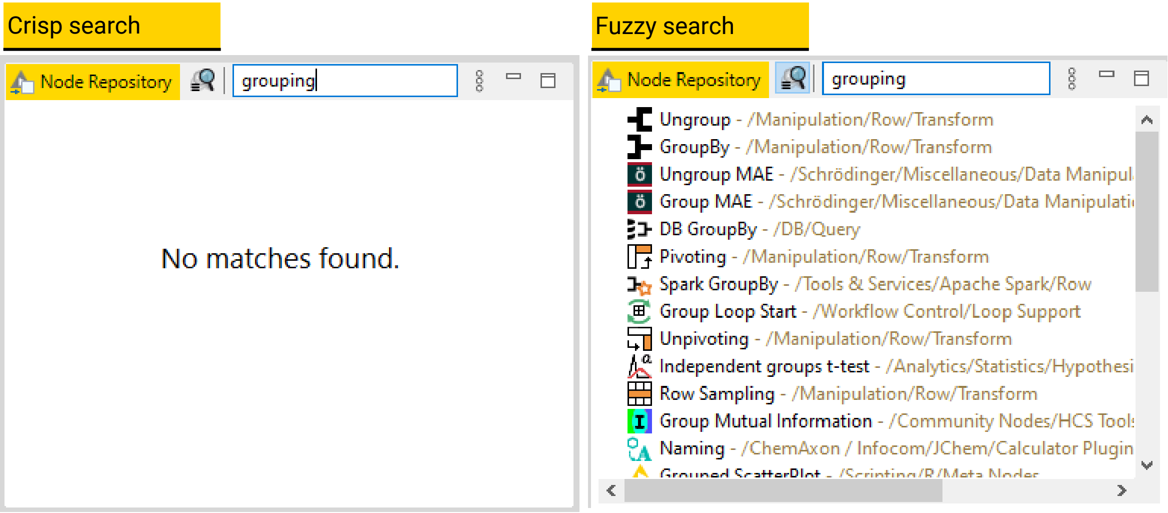
The number of nodes in the most frequently and last used categories is per default restricted to ten nodes. This number can be adjusted in the preferences. Select `File > Preferences…` and under `/KNIME/KNIME GUI` set different values for the maximum size of frequently used nodes and maximum number of last used nodes.

Node repository

Currently installed nodes are available in the node repository where they are organized under different categories. You can add a node from the node repository into the workflow editor by drag and drop, or by a double click, as explained in the section Building Workflows.

Search for a node by expanding the categories or by typing a search term in the search field on top of the node repository, as shown in Figure 27. The default search mode is *crisp search*. Using this search mode, the interface returns all the nodes that either have the search term in their names, or are in a subcategory whose name includes the search term.

Switch the search mode to *fuzzy search* by clicking the icon next to the search field. In this search mode the interface returns all the nodes that are related to the search term.



No matches found.

Ungroup - /Manipulation/Row/Transform
GroupBy - /Manipulation/Row/Transform
Ungroup MAE - /Schrödinger/Miscellaneous/Data Manipul
Group MAE - /Schrödinger/Miscellaneous/Data Manipulati
DB GroupBy - /DB/Query
Pivoting - /Manipulation/Row/Transform
Spark GroupBy - /Tools & Services/Apache Spark/Row
Group Loop Start - /Workflow Control/Loop Support
Unpivoting - /Manipulation/Row/Transform
Independent groups t-test - /Analytics/Statistics/Hypothesi
Row Sampling - /Manipulation/Row/Transform
Group Mutual Information - /Community Nodes/HCS Tool:
Naming - /ChemAxon / Infocom/JChem/Calculator Plugin
Grouped ScatterPlot - /Scripting/R/Meta Nodes

Figure 27. Node repository with two search modes

An introduction to the node repository is also available in the video Node Repository.

KNIME Hub view

The KNIME Hub view on the right of the KNIME Workbench, is shown in Figure 28 provides a convenient way to directly use all the features of the KNIME Hub from within KNIME Analytics Platform.

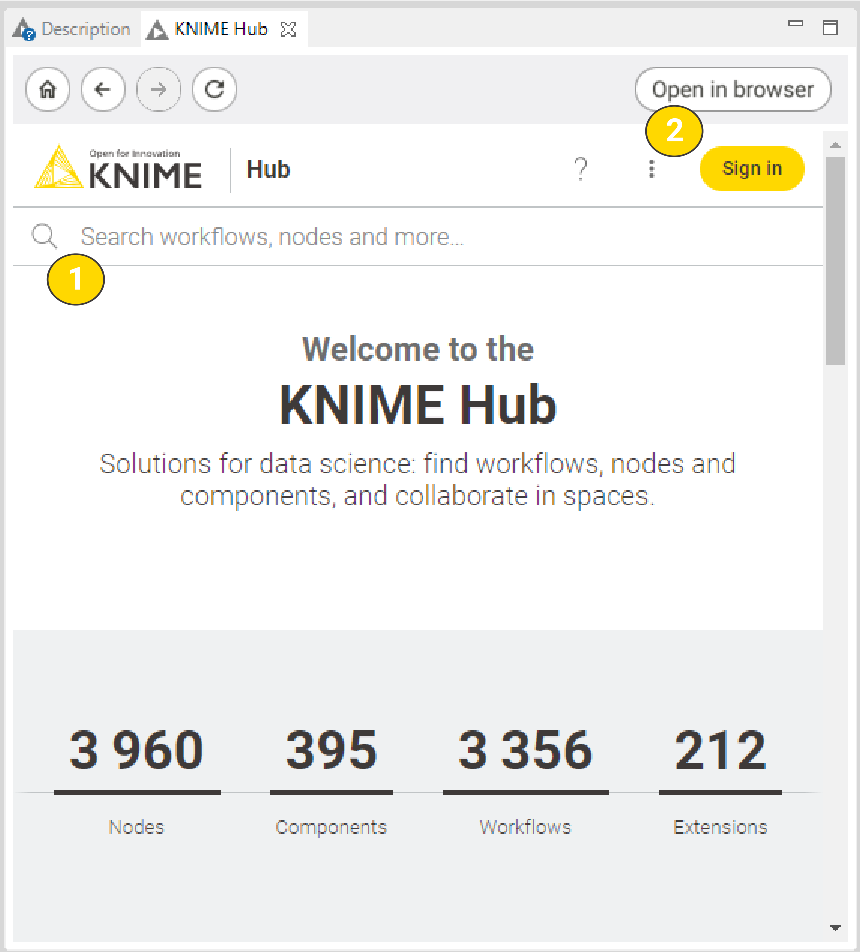

Description
KNIME Hub
Open in browser
2
Open for Innovation KNIME
Hub
Sign in
Search workflows, nodes and more...
1
Welcome to the
KNIME Hub
Solutions for data science: find workflows, nodes and components, and collaborate in spaces.
3 960
Nodes
395
Components
3 356
Workflows
212
Extensions

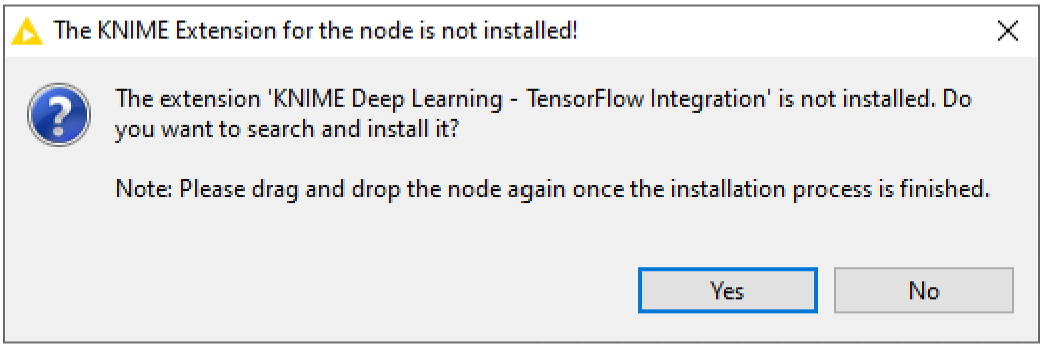
Figure 28. The KNIME Hub view

Within the KNIME Hub view you have access to the following features:

- 1 **Search**: Enter a search term or sentence in the search field on the top of the view, press "Enter" and navigate the KNIME Hub. The search on the KNIME Hub looks for nodes, extensions, components, and workflows, among KNIME example workflows and components as well as workflows and components built and uploaded by the community. The search results display detailed information, e.g. where to find a specific node, links to documentation or external links to useful blog posts.

  You can filter your search results based on the following categories:

  - Nodes or components: you can add a node or a component to a currently open workflow, configure it, and run it. If the node is part of an extension that is still not installed in the KNIME Analytics Platform in use, a message will prompt as shown in Figure 29, and you can automatically proceed installing the missing extension. The same will happen if the component you drag and drop to the workflow editor contains a node that is part of a not installed extension.



The KNIME Extension for the node is not installed!

The extension 'KNIME Deep Learning - TensorFlow Integration' is not installed. Do you want to search and install it?

Note: Please drag and drop the node again once the installation process is finished.

Figure 29. Message prompted in case a node dragged from the KNIME Hub is missing an extension

- Workflows: you can download workflows or drag and drop them (use the ⊞ icon) to your local workspace to open them directly in the workflow editor
- Extensions: you can drag and drop the extension you want to install (use the ⊞ icon). In case the update site required for the installation of the extension is not enabled you will be asked to enable it.

- 2 **Sign in**: Click *Sign in* button on the top right and you can enter your *Username or email address* and *Password*, or *Create account* in case you do not have one, yet. Once you sign in you have access from the view to your own profile and spaces. Click your icon on the top right of the KNIME Hub view and choose *Profile* or *Spaces* from the drop-down menu.

> If you sign in to KNIME Hub from the view is independent from the sign in to KNIME Hub from the browser as well as the sign in to you KNIME Hub mount point in KNIME Explorer.

For more information on how to use KNIME Hub please refer to the KNIME Hub About page.

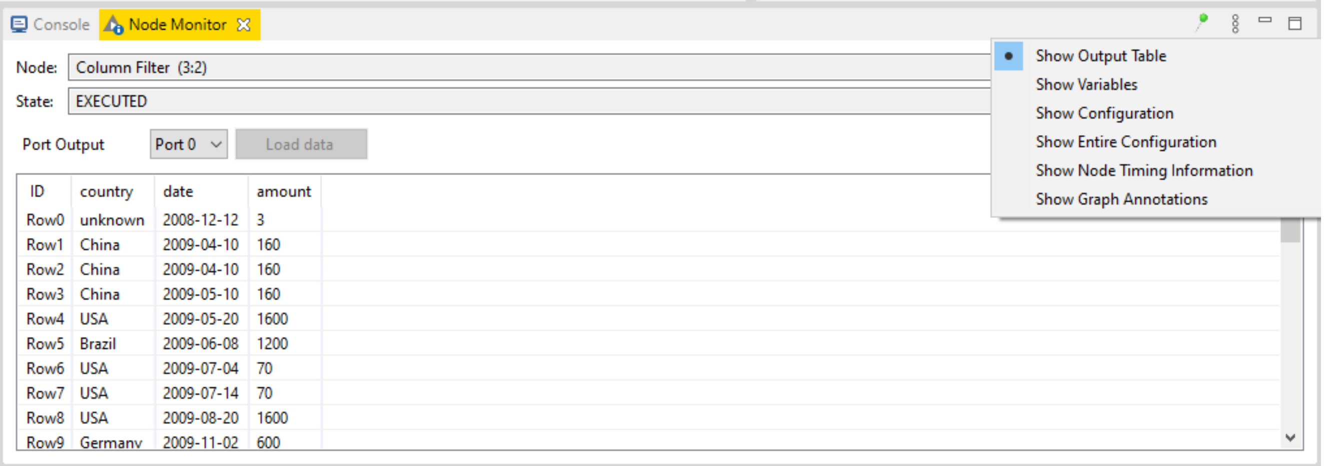
Description

The description panel on the right of the KNIME Workbench shown in Figure 2 provides a description of the currently active workflow, or a node selected in the node repository or workflow editor. For a workflow, the first part is a general description, followed by tags and links to other resources related to the workflow. For a node, the first part is a general description, followed by the available setting options, and finally a list of input and output ports.

Node Monitor

The Node Monitor tab is located on the same panel of the console tab on the bottom part of the KNIME Workbench shown in Figure 30. It is especially useful to inspect intermediate output tables in the workflow.

The Node Monitor tab is shown by default since KNIME Analytics Platform version 4.2. For KNIME Analytics Platform version <4.1, or in case you closed the Node Monitor tab and you want to restore it, go to *View* in the toolbar and select *Node Monitor* from the menu.



Node: Column Filter (3:2)

State: EXECUTED

Port Output: Port 0

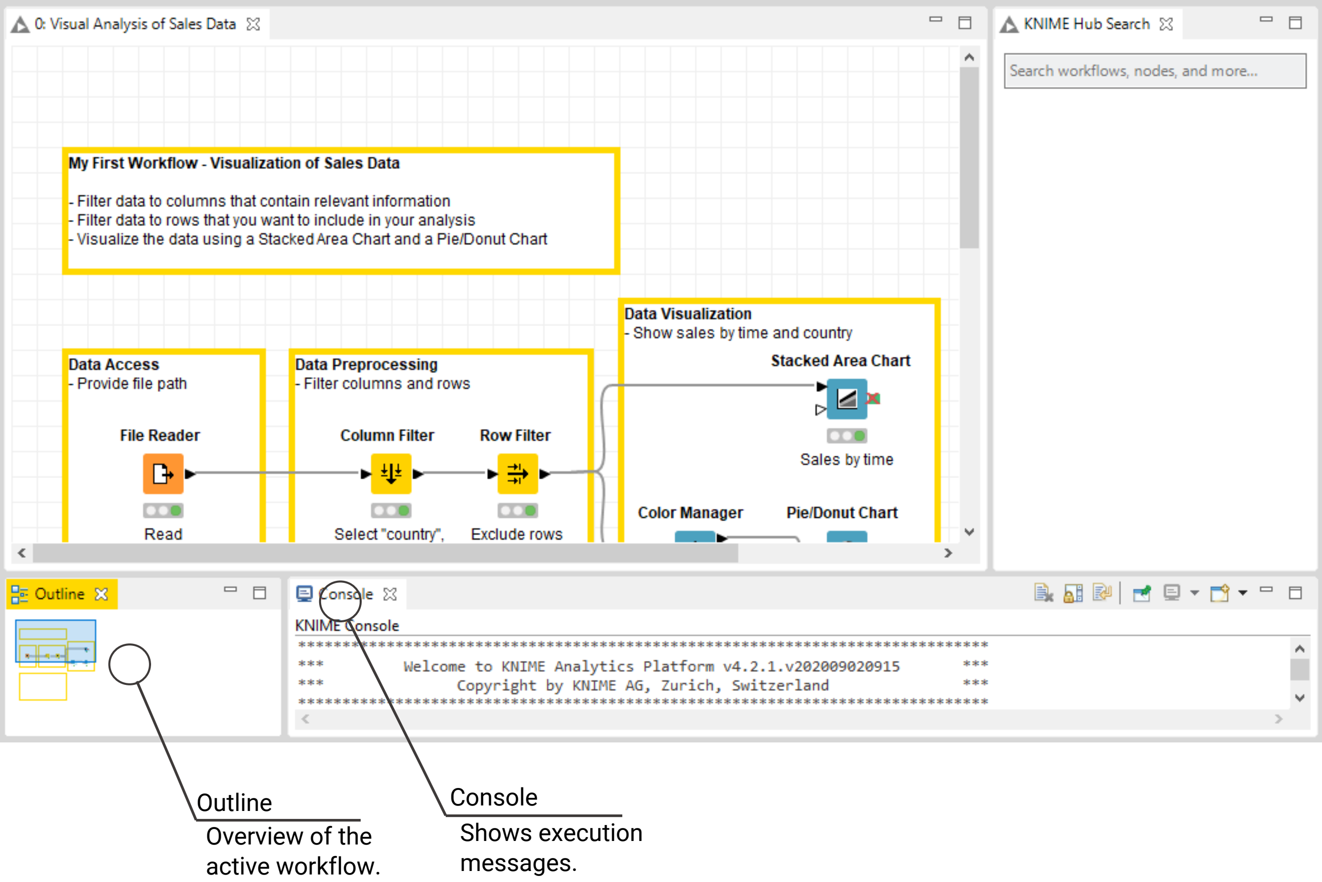
ID	country	date	amount
Row0	unknown	2008-12-12	3
Row1	China	2009-04-10	160
Row2	China	2009-04-10	160
Row3	China	2009-05-10	160
Row4	USA	2009-05-20	1600
Row5	Brazil	2009-06-08	1200
Row6	USA	2009-07-04	70
Row7	USA	2009-07-14	70
Row8	USA	2009-08-20	1600
Row9	Germany	2009-11-02	600

Figure 30. Node Monitor

Here you can choose to show the flow variables or a preview of the output data at any port of a selected node in the active workflow. To choose what kind of output you want to visualize, click the three vertical dots on the right-upper side of the node monitor tab, and select the You can also pin the monitor view to a specific node independently of the selection in the workflow editor, in order, for example, to follow the evolution of data flow and help with debugging. To do this select the node you want to pin, and click the green pin symbol on the right-upper side of the node monitor tab.

Outline

In the outline, on the bottom part of the KNIME Workbench shown in Figure 31, you find an overview of the currently active workflow. If the whole workflow does not fit in the workflow editor, you can change the active area by scrolling the blue, transparent rectangle.



My First Workflow - Visualization of Sales Data

- Filter data to columns that contain relevant information
- Filter data to rows that you want to include in your analysis
- Visualize the data using a Stacked Area Chart and a Pie/Donut Chart

Data Access
- Provide file path

Data Preprocessing
- Filter columns and rows

Data Visualization
- Show sales by time and country

KNIME Console

```
******************************************************************
***         Welcome to KNIME Analytics Platform v4.2.1.v202009020915      ***
***                Copyright by KNIME AG, Zurich, Switzerland             ***
******************************************************************
```

Outline
Overview of the active workflow.

Console
Shows execution messages.

Figure 31. Outline and console

Console

The console tab on the bottom part of the KNIME Workbench shown in Figure 31 shows all warning and error messages related to the workflow execution. To debug and log information messages to be reported in the console, change the console log level in *File* → *Preferences* → *KNIME* → *KNIME GUI*.

Customizing the KNIME Workbench

Reset and logging

When a node is reset, the node status changes from "executed" to "configured" and the output of the node is not available any more. When saving a workflow in an executed state, the data used in the workflow are saved as well. That is, the larger the dataset, the larger the file size. Therefore, resetting workflows before saving them is recommended in case the dataset can be accessed without any restrictions.

A reset workflow only saves the node configurations, and not any results. However, resetting a node does not undo the operation executed before. All operations done during creation, configuration, and execution of a workflow are reported in the `knime.log` file.

To inspect the `knime.log` file you go to *View* → *Open KNIME log*. The log file opens in the workflow editor. The `knime.log` file has a limited size, and after reaching it the rows will be overwritten from the top.

The `knime.log` file is also located in the `knime`-folder inside the `.metadata`-folder, in the KNIME workspace folder defined when launching KNIME Analytics Platform.

Show heap status

The heap status panel shows the memory usage during the execution of a workflow, and helps to monitor memory usage for the project. To add the heap status panel to the workbench, go to *File* → *Preferences*. In the dialog that opens, click *General*, select *Show heap status*, and click *Apply and Close*.

A heap status bar showing the memory usage appears in the bottom right part of the status bar, directly below the console panel. Next to the heap status bar is the "Run Garbage Collector" button. Click it to manually allocate and free up memory.

Configuring KNIME Analytics Platform

Preferences

In *Preferences* you can adjust the default settings of KNIME Analytics Platform. Go to *File → Preferences*, and a list of subcategories is displayed in the dialog that opens. Each category contains a separate dialog for specific settings like database drivers, available update sites, and appearance.

KNIME

Selecting *KNIME* in the list of subcategories, allows you to define the log file log level. By default it is set to *DEBUG*. This log level helps developers to find reasons for any unexpected behavior.

Directly below, you can define the maximum number of threads for all nodes. Separate branches of the workflow are distributed to several threads to optimize the overall execution time. By default the number of threads is set to twice the number of CPUs on the running machine.

In the same dialog, you can also define the folder for temporary files.

Check the last option *Yes, help improve KNIME.* to agree to sending us anonymous usage data. This agreement activates the node recommendations by community in the Workflow Coach.

KNIME GUI

The *KNIME* category, contains a subcategory *KNIME GUI*. In this dialog, you can define the console view log level. By default it is set to "WARN", because more detailed information is only useful for diagnosis purposes.

Further below, you can select which confirmation dialogs are shown when using KNIME Analytics Platform. Choose from the following:

- Confirmation after resetting a node
- Deleting a node or connection
- Replacing a connection
- Saving and executing workflow

- Loading workflows created with a nightly build

In the same dialog, you can define what happens if an operation requires executing the previous nodes in the workflow. You have these three options:

- Execute the nodes automatically
- Always reject the node execution
- Show a dialog to execute or not

The following options allow you to define whether workflows should be saved automatically and after what time interval, also whether linked components and metanodes should be automatically updated. You can also define visual properties such as the border width of workflow annotations.

Table backend

In order to optimize the use of main memory in KNIME Analytics Platform as cell elements in a table are represented by Java objects, reviewing the underlying data representation. Starting with KNIME Analytics Platform version 4.3 a new Columnar Table Backend is introduced. This extension addresses these issues by using a different underlying data layer (backed by Apache Arrow), which is based on a columnar representation.

To work with the Columnar Table Backend you need to first install the extension. From the KNIME Analytics Platform go to *File* → *Install KNIME Extensions…* and select *KNIME Columnar Table Backend* extension, under KNIME Labs Extensions category.

The type of table backend used is defined at the workflow level. Right click any workflow in the KNIME Explorer and select *Configure…* from the context menu, as shown in Figure 32.

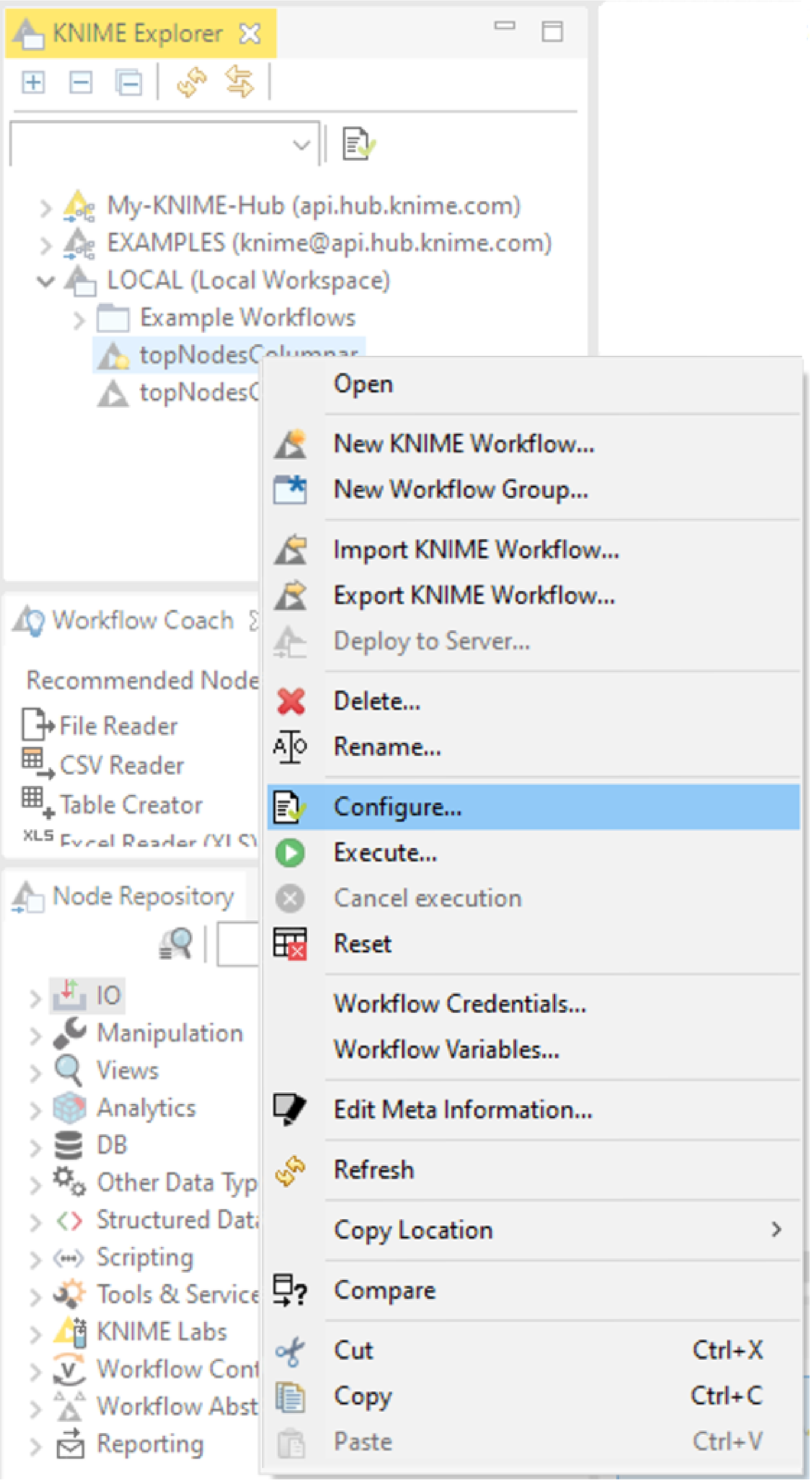
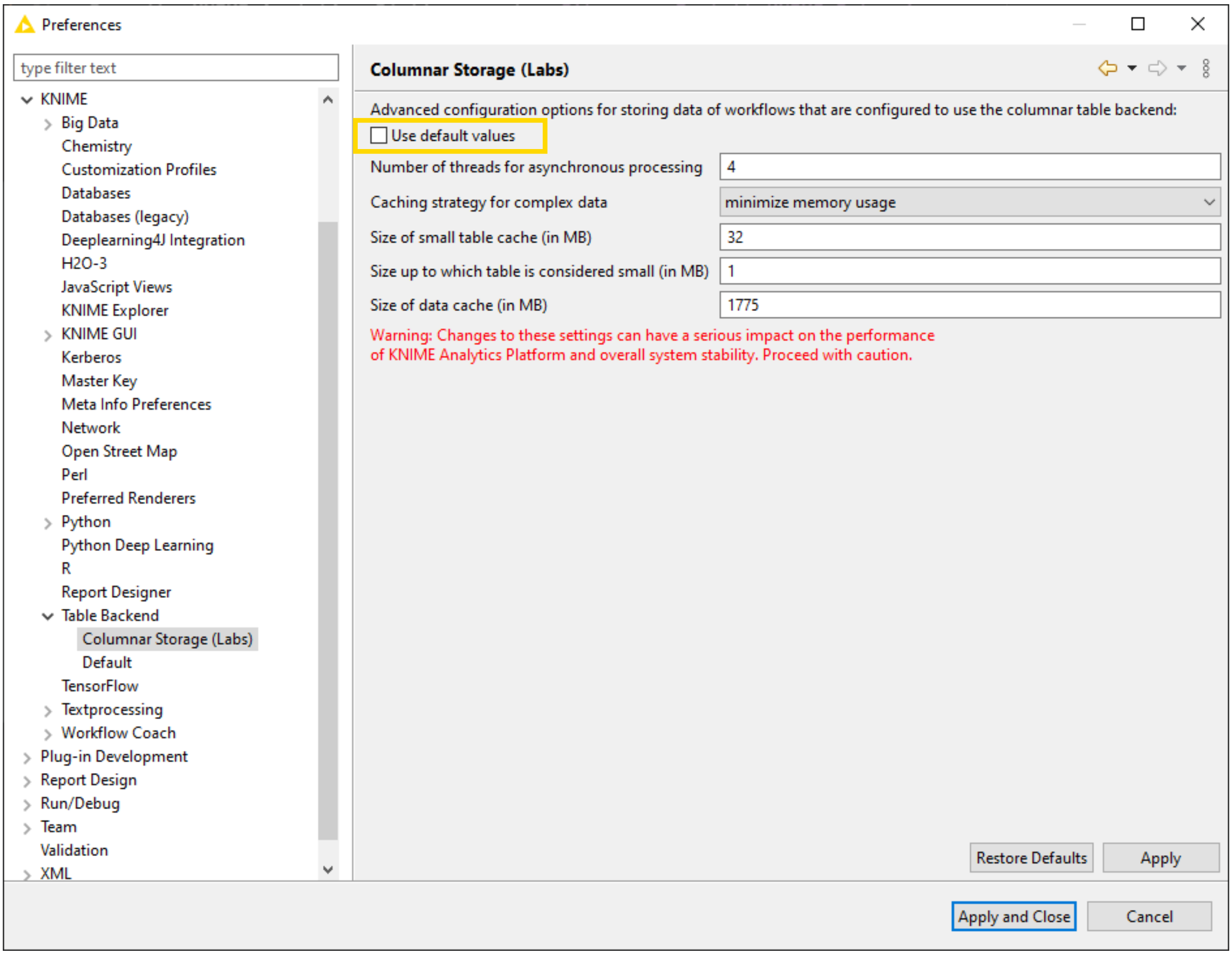


Figure 32. Selecting Configure to define the type of Table Backend used for the selected workflow

The parameters relative to memory usage of the Columnar Table Backend can also be configured. Go to *File* → *Preferences* and select *Table Backend* → *Columnar Storage (Labs)* under KNIME in the left pane of the preferences window, as shown in Figure 33.



Columnar Storage (Labs)

Advanced configuration options for storing data of workflows that are configured to use the columnar table backend:

□ Use default values

Number of threads for asynchronous processing: 4

Caching strategy for complex data: minimize memory usage

Size of small table cache (in MB): 32

Size up to which table is considered small (in MB): 1

Size of data cache (in MB): 1775

Warning: Changes to these settings can have a serious impact on the performance of KNIME Analytics Platform and overall system stability. Proceed with caution.

Figure 33. The Columnar Table Backend preference page.

Some default values are already set automatically, based on the system specifications where the current KNIME Analytics Platform is installed. However, unchecking the *Use default values* option activates the fields below, where the advanced configuration options can be set. Be aware that changes to these settings can seriously impact the performance of KNIME Analytics Platform and overall system stability.

In the Columnar Table Backend, there are currently three caches, the size and behavior of which can be configured via the Columnar Table Backend preference page as well as, eventually, through the knime.ini.

- *Caching strategy for complex data*: The Complex Data Cache holds data in the Java Virtual Machine’s heap region of memory and can be configured to minimize memory usage or maximize performance.

- *Size of small table cache (in MB)* and *Size up to which table is considered small (in MB)*: The Small Table Cache holds recently used small tables in the off-heap memory region. The threshold up to which a table is considered small and the size of the cache (in megabytes) can be configured through the Preference page
- *Size of data cache (in MB)*: The General Data Cache holds recently used chunks of arbitrarily-sized tables up to a configurable total size (in megabytes) in the off-heap memory region.

Note that the caches that reside in the off-heap memory region require an amount of memory in addition to whatever memory you have allotted to the heap space of your KNIME's Java Virtual Machine via the `-Xmx` parameter in the `knime.ini`. When altering the sizes of these cache via the preference page, make sure not to exceed your system's physical memory size as otherwise you might encounter system instability or even crashes.

> ℹ For a more detailed explanation of the Columnar Table Backend technical background please refer to this post on KNIME Blog.

Setting up knime.ini

When installing KNIME Analytics Platform, configuration options are set to their defaults. The configuration options, i.e. options used by KNIME Analytics Platform, range from memory settings to system properties required by some extensions.

You can change the default settings in the `knime.ini` file. The `knime.ini` file is located in the installation folder of KNIME Analytics Platform.

> ℹ To locate the `knime.ini` file on MacOS, open Finder and navigate to the installed Applications.
> Next, right click the KNIME application, select *Show Package Contents* in the menu, and navigate to *Contents*, and open *Eclipse*.

Edit the `knime.ini` file with any plaintext editor, such as Notepad (Windows), TextEdit (MacOS) or gedit (Linux).

The entry `-Xmx1024m` in the `knime.ini` file specifies how much memory KNIME Analytics Platform is allowed to use. The setting for this value will depend on how much memory is available in the running machine. We recommend setting it to approximately one half of the available memory, but this value can be modified and personalized. For example, if the computer has 16GB of memory, the entry might be set to `-Xmx8G`.

Besides the memory available, you can define many other settings in the `knime.ini` file. Find an overview of some of the most common settings in Table 4 or in this complete list of the configuration options.

Table 4. Common configuration settings in knime.ini file

Setting	Explanation
`-Xmx` • default value: `1024m` • example: `-Xmx16G`	Sets the maximum amount of memory available for KNIME Analytics Platform.
`-Dknime.compress.io` • default value: `SNAPPY` • possible values: `[SNAPPY|GZIP|NONE]` • example: `-Dknime.compress.io=SNAPPY`	Determines which compression algorithm (if any) to use when writing temporary tables to disk.
`-Dorg.knime.container.cellsinmemory` • default value: 5,000 • possible values: any value between 0 and 2,147,483,647 • example: `-Dorg.knime.container.cellsinmemory=100,000`	This setting defines the size of a "small table". Small tables are attempted to be kept in memory, independent of the Table Caching strategy. By increasing the size of a small table, the number of swaps to the disk can be limited, which comes at the cost of reducing memory space available for other operations.
`-Dknime.table.cache` • default value: `LRU` • possible values: `[LRU|SMALL]` • example: `-Dknime.table.cache=SMALL`	Determines whether to attempt to cache large tables (i.e., tables that are not considered to be "small"; see setting `-Dorg.knime.container.cellsinmemory`) in memory. If set to `LRU`, large tables are evicted from memory in least-recently used (LRU) order or when memory becomes scarce. If set to `SMALL`, large tables are always flushed to disk.

Setting	Explanation
`-Dknime.url.timeout` • default value: 1,000 ms • example: `-Dknime.url.timeout=100`	When trying to connect or read data from an URL, this value defines a timeout for the request. Increase the value if a reader node fails. A too high timeout value may lead to slow websites blocking dialogs in KNIME Analytics Platform.

KNIME runtime options

KNIME's runtime behavior can be configured in various ways by passing options on the command line during startup. Since KNIME is based on Eclipse, all Eclipse runtime options also apply to KNIME.

KNIME also adds additional options, which are described below.

Command line arguments

Listed below are the command line arguments processed by KNIME. They can either be specified permanently in the `knime.ini` in the root of the KNIME installation, or be passed to the KNIME executable. Please note that command line arguments must be specified *before* the system properties (see below) i.e. before the `-vmargs` parameter.
Note that headless KNIME applications, such as the batch executor, offer quite a few command line arguments. They are not described here but are printed if you call the application without any arguments.

`-checkForUpdates`

If this arguments is used, KNIME automatically checks for updates during startup. If new versions of installed features are found, the user will be prompted to install them. A restart is required after updates have been installed.

Java system properties

Listed below are the Java system properties with which KNIME's behavior can be changed. They can either be specified permanently in the `knime.ini` in the root of the KNIME installation, or be passed to the KNIME executable. Please note that system properties must be specified *after* the `-vmargs` parameter. The required format is `-DpropName=propValue`.

General properties

`org.knime.core.maxThreads=<number>`

Sets the maximum number of threads that KNIME is using for executing nodes. By default this number is 1.5 times the number of cores. This property overrides the value from the KNIME preference page.

`knime.tmpdir=<directory>`

Sets the default directory for temporary files KNIME files (such as data files). This property overrides the value from the preference pages and is by default the same as the `java.io.tmpdir` .

`knime.synchronous.io=(true|false)`

Can be used to enforce the sequential processing of rows for KNIME tables. By default, each table container processes its rows asynchronously in a number of (potentially re-used) threads. The default value is `false`. Setting this field to `true` will instruct KNIME to always handle rows sequentially and synchronously, which in some cases may be slower.

`knime.async.io.cachesize=<number>`

Sets the batch size for non-sequential and asynchronous handling of rows (see knime.synchronous.io). It specifies the amount of data rows that are handled by a single container thread. The larger the buffer, the smaller the synchronization overhead but the larger the memory requirements. This property has no effect if rows are handled sequentially. The default value is 10.

`knime.domain.valuecount=<number>`

The number of nominal values kept in the domain when adding rows to a table. This is only the default and may be overruled by individual node implementations.If no value is specified a default of 60 will be used.

`org.knime.container.threads.total=<number>`

Sets the maximum number of threads that can be used to write KNIME native output tables. By default this number equals the number of processors available to the JVM. *Note:* This value has to be greater than 0.

`org.knime.container.threads.instance=<number>`

Sets the maximum number of threads that can be used to write a *single* KNIME native output table. By default this number equals the number of processors available to the JVM. *Note:* This value has to be greater than 0 and cannot be larger than `org.knime.container.threads.total`.

`knime.discourage.gc=(true|false)`

If set to true, discourages KNIME from triggering a full stop-the-world garbage collection. Note that (a) individual nodes are allowed to disregard this setting and (b) the garbage collector may independently decide that a full stop-the-world garbage collection is warranted. Set to true by default.

`org.knime.container.minspace.temp=<number>`

Java property to specify the minimum free disc space in MB that needs to be available.
If less is available, no further table files & blobs will be created (resulting in an exception).

`knime.columnar.chunksize=<number>`

The columnar table backend horizontally divides tables into batches and vertically divides these batches into column chunks. This property controls the initial size of these chunks and thereby the number of rows per batch. A chunk is the smallest unit that must be materialized to access a single value. Changing this value can therefore impact memory footprint and overall performance. Do not change this value unless you have good reasons. The default value is 28,000.

`knime.columnar.reservedmemorymb=<number>`

The columnar table backend caches table data off-heap. To this end, it requires memory in addition to the JVM’s heap memory, whose size is controlled via the -Xmx parameter. If no explicit cache sizes are set in the preferences, the default memory available for caching is computed as follows: Total physical memory minus reserved memory minus 1.25 times heap memory. The reserved memory size in this equation (in MB) can be configured via this property. The default is 4,096.

`knime.columnar.verbose=(true|false)`

Setting this property to true activates verbose debug logging in the columnar table backend.

-

`knime.disable.rowid.duplicatecheck=(true|false)`

Enables/disables row ID duplicate checks on tables. Tables in KNIME are supposed to have unique IDs, whereby the uniqueness is asserted using a duplicate checker. This property will disable this check.
Warning: This property should not be changed by the user.

`knime.disable.vmfilelock=(true|false)`

Enables/disables workflow locks. As of KNIME 2.4 workflows will be locked when opened; this property will disable the locking (allowing multiple instances to have the same workflow open).
Warning: This property should not be changed by the user.

`knime.database.timeout=<number>`

Sets the timeout in seconds trying to establish a connection to a database.
The default value is 15 seconds.

`knime.database.fetchsize=<number>`

Sets the fetch size for retrieving data from a database.
The default value depends on the used JDBC driver.

`knime.database.batch_write_size=<number>`

Sets the batch write size for writing data rows into a database.
The default value is 1, that is one row at a time.

`knime.database.enable.concurrency=(true|false)`

Used to switch on/off the database connection access (applies only for the same database connection).
Default is true, that is all database accesses are synchronized based on single connection; false means off, that is, the access is not synchronized and may lead to database errors.

`knime.logfile.maxsize=<number>[mk]`

Allows one to change the maximum log file size (default is 10 MB).
Values must be integer, possibly succeeded by "m" or "k" to denote that the given value is in mega or kilo byte.

`knime.settings.passwords.forbidden=(true|false)`

If *true*, nodes using passwords as part of their configuration (e.g. DB connection or SendEmail) will not store the password as part of the workflow on disc. Instead a null value is stored, which will cause the node’s configuration to be incorrect (but valid) after the workflow is restored from disc. Default is *false*.

`knime.repository.non-instant-search=(true|false)`

Allows to disable the live update in the node repository search.
-

`knime.macosx.dialogworkaround=(true|false)`

Allows to disable the workaround for freezes when opening node dialogs under MacOSX.
-

`knime.data.bitvector.maxDisplayBits=<number>`

Sets the maximum number of bits that are display in string representations of bit vectors.
-

`knime.xml.disable_external_entities=(true|false)`

If set to true, all nodes that parse XML files will not read external entities defined via a DTD.
This is usually only useful when running as an executor on the server and you want prevent XXE attacks.

Plug-in dependent properties

These properties only affect some plug-ins and are only applicable if they are installed.

`org.knime.cmlminblobsize=<number>[mMkK]`

Allows to change the minimum size in bytes (or kilobyte or megabytes) a CML molecule must have before it is stored in a blob cell. Otherwise it is stored inline. The latter is a bit faster but needs more memory. The default is 8kB.

`org.knime.ctabminblobsize=<number>[mMkK]`

Allows to change the minimum size in bytes (or kilobyte or megabytes) a Ctab molecule must have before it is stored in a blob cell. Otherwise it is stored inline. The latter is a bit faster but needs more memory. The default is 8kB.

`org.knime.mol2minblobsize=<number>[mMkK]`

Allows to change the minimum size in bytes (or kilobyte or megabytes) a Mol2 molecule must have before it is stored in a blob cell. Otherwise it is stored inline. The latter is a bit faster but needs more memory. The default is 8kB.

`org.knime.molminblobsize=<number>[mMkK]`

Allows to change the minimum size in bytes (or kilobyte or megabytes) a Mol molecule must have before it is stored in a blob cell. Otherwise it is stored inline. The latter is a bit faster but needs more memory. The default is 8kB.

`org.knime.rxnminblobsize=<number>[mMkK]`

Allows to change the minimum size in bytes (or kilobyte or megabytes) a Rxn molecule must have before it is stored in a blob cell. Otherwise it is stored inline. The latter is a bit faster but needs more memory. The default is 8kB.

`org.knime.sdfminblobsize=<number>[mMkK]`

Allows to change the minimum size in bytes (or kilobyte or megabytes) a SDF molecule must have before it is stored in a blob cell. Otherwise it is stored inline. The latter is a bit faster but needs more memory. The default is 8kB.

KNIME tables

Data table

Very common input and output ports of nodes are data input ports and data output ports, which correspond to the black triangles in Figure 34.

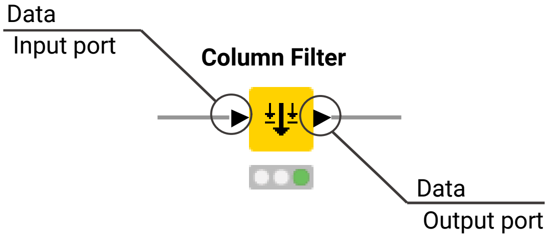


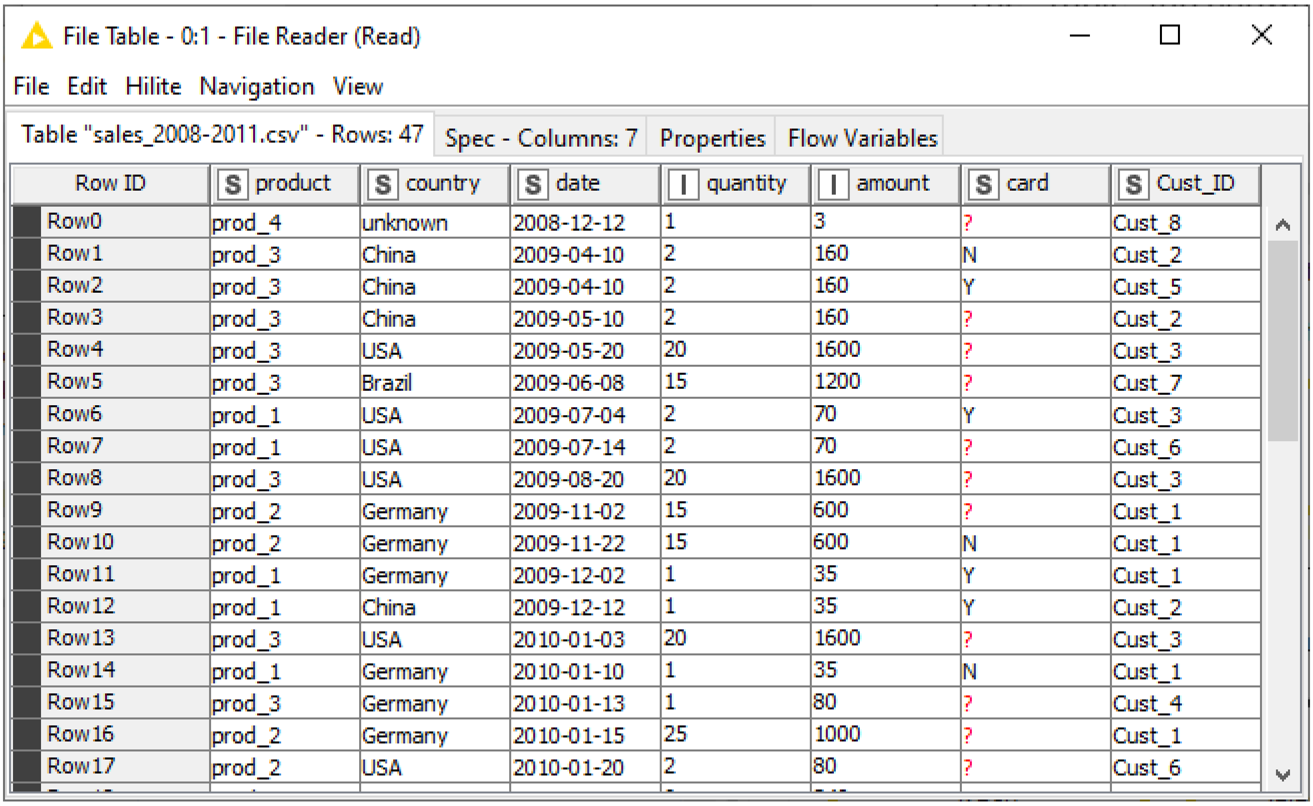
Figure 34. Data input and output port

A data table is organized by columns and rows, and it contains a number of equal-length rows. Elements in each column must have the same data type.

The data table shown in Figure 35 is produced by the File Reader node, which is one of the many nodes with a black triangle output port for data output. To open the table, right click the node and select the last item *File Table* in the menu. The output table has unique row IDs and column headers. The row IDs are automatically created by the reader node, but they can also be defined manually. The row IDs and the column headers can therefore be used to identify each data cell in the table. Missing values in the data are shown by a question mark.

Besides the data table, the node output contains the following tabs:

1. The "Table" tab shows the contents of the table
2. The "Spec" tab shows the meta information of the table, including the column name, column type, and optional properties like the domain of the values in the column
3. The "Properties" tab, shows metadata related to some columns, for example the width of the histogram in the "Histogram" column produced by the Statistics node
4. The "Flow Variables" tab shows the available flow variables in the node output and their current values.


File Table - 0:1 - File Reader (Read)
File Edit Hilite Navigation View
Table "sales_2008-2011.csv" - Rows: 47
Spec - Columns: 7
Properties
Flow Variables

Row ID	S product	S country	S date	I quantity	I amount	S card	S Cust_ID
Row0	prod_4	unknown	2008-12-12	1	3	?	Cust_8
Row1	prod_3	China	2009-04-10	2	160	N	Cust_2
Row2	prod_3	China	2009-04-10	2	160	Y	Cust_5
Row3	prod_3	China	2009-05-10	2	160	?	Cust_2
Row4	prod_3	USA	2009-05-20	20	1600	?	Cust_3
Row5	prod_3	Brazil	2009-06-08	15	1200	?	Cust_7
Row6	prod_1	USA	2009-07-04	2	70	Y	Cust_3
Row7	prod_1	USA	2009-07-14	2	70	?	Cust_6
Row8	prod_3	USA	2009-08-20	20	1600	?	Cust_3
Row9	prod_2	Germany	2009-11-02	15	600	?	Cust_1
Row10	prod_2	Germany	2009-11-22	15	600	N	Cust_1
Row11	prod_1	Germany	2009-12-02	1	35	Y	Cust_1
Row12	prod_1	China	2009-12-12	1	35	Y	Cust_2
Row13	prod_3	USA	2010-01-03	20	1600	?	Cust_3
Row14	prod_1	Germany	2010-01-10	1	35	N	Cust_1
Row15	prod_3	Germany	2010-01-13	1	80	?	Cust_4
Row16	prod_2	Germany	2010-01-15	25	1000	?	Cust_1
Row17	prod_2	USA	2010-01-20	2	80	?	Cust_6

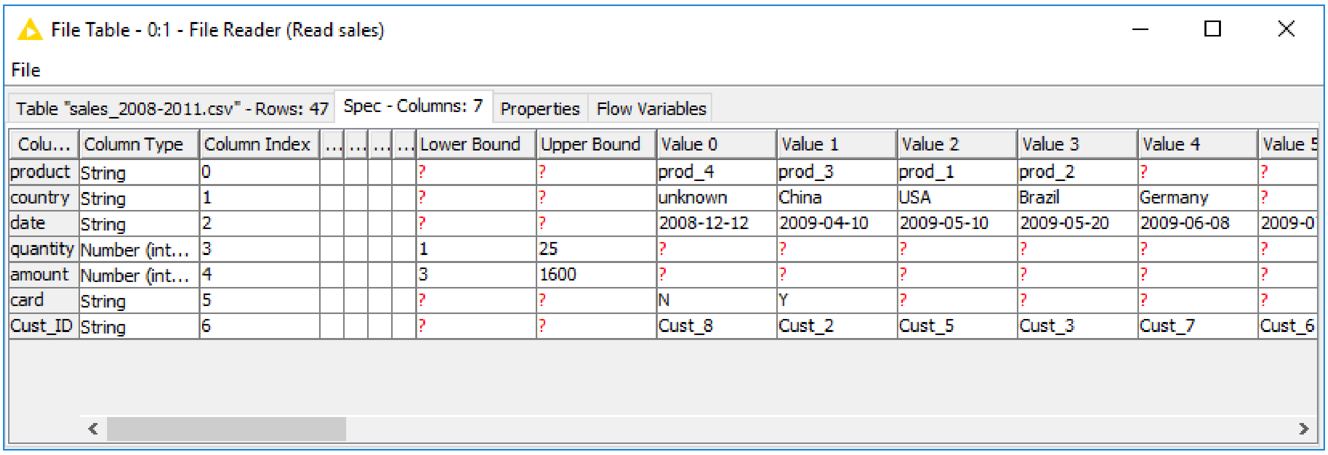
Figure 35. Data output in KNIME Analytics Platform

In the video Data Table Structure we introduce the data organization and data representation in KNIME Analytics Platform in more detail.

Column types

The basic data types in KNIME Analytics Platform are `Integer`, `Double`, and `String`, along with other supported data types such as `Long`, `Boolean` value, `JSON`, `URI`, `Document`, `Date&Time`, `Bit vector`, `Image`, and `Blob`. KNIME Analytics Platform also supports customized data types, for example, a representation of a molecule.

Click the "Spec" tab in an output table, to see the data types of the columns in the data table, as well as the domain of the values in the columns, as shown in Figure 36. For numerical values, only the range of the values in the data is shown. For string values, the different values appearing in the data are shown.



File Table - 0:1 - File Reader (Read sales)

File

Table "sales_2008-2011.csv" - Rows: 47 | Spec - Columns: 7 | Properties | Flow Variables

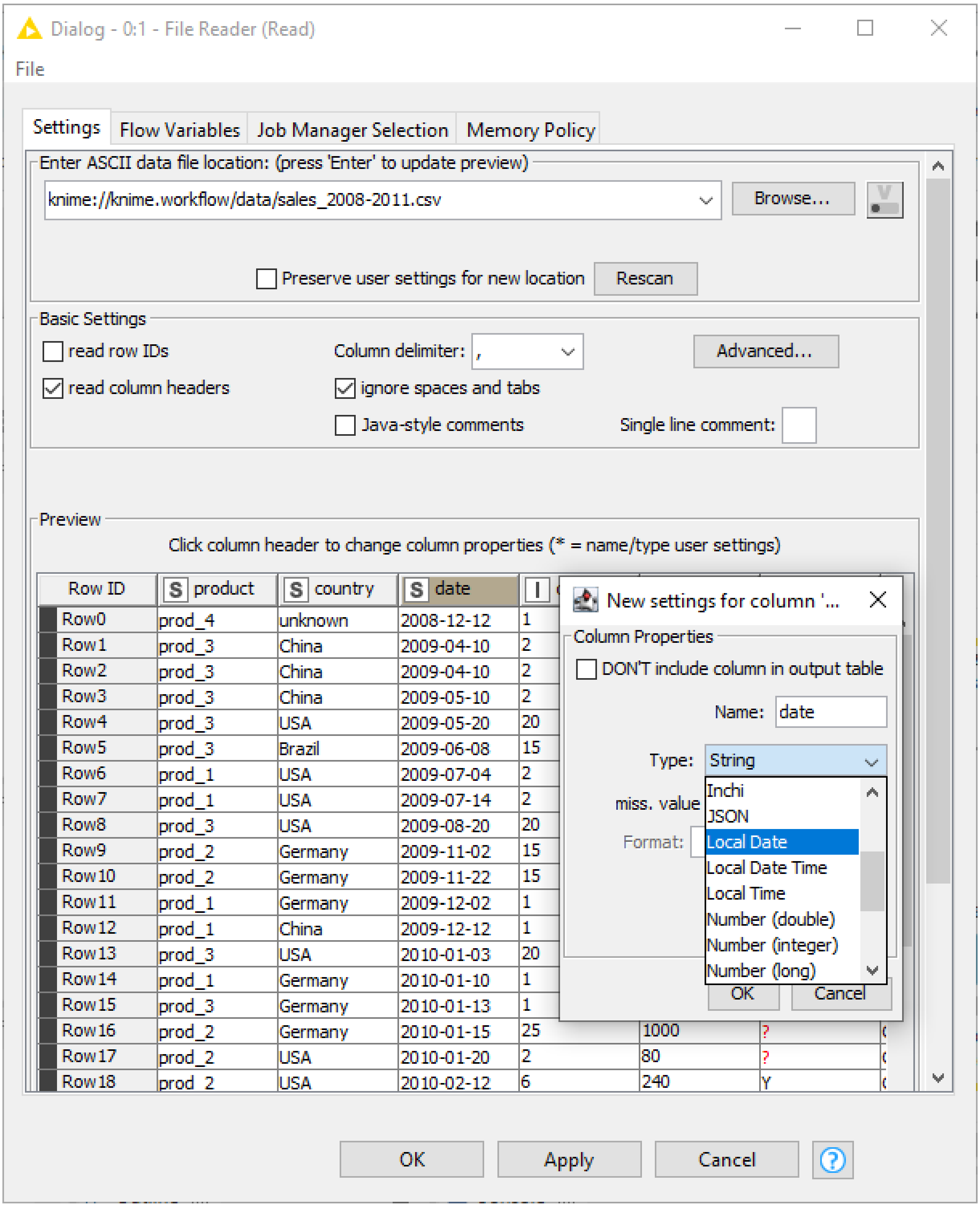
Colu…	Column Type	Column Index	…	…	…	…	Lower Bound	Upper Bound	Value 0	Value 1	Value 2	Value 3	Value 4	Value 5
product	String	0					?	?	prod_4	prod_3	prod_1	prod_2	?	?
country	String	1					?	?	unknown	China	USA	Brazil	Germany	?
date	String	2					?	?	2008-12-12	2009-04-10	2009-05-10	2009-05-20	2009-06-08	2009-0
quantity	Number (int…	3					1	25	?	?	?	?	?	?
amount	Number (int…	4					3	1600	?	?	?	?	?	?
card	String	5					?	?	N	Y	?	?	?	?
Cust_ID	String	6					?	?	Cust_8	Cust_2	Cust_5	Cust_3	Cust_7	Cust_6

Figure 36. Data types and data domain in "Spec" tab

The reader nodes in KNIME Analytics Platform assign a data type to each column based on their interpretation of the content. If the correct data type of a column is not recognized by the reader node, the data type can be corrected afterwards. There are nodes available to convert data types. For example: String to Number, Number to String, Double to Int, String to Date&Time, String to JSON, and String to URI.

Many of the special data types are recognized as `String` by the reader nodes. To convert these `String` columns to their correct data types, use the Column Type Auto Cast node.

When you use the File Reader node to read a file you can convert the column types directly via the node configuration dialog. To do this double click a column header in the preview and change the column type in the dialog that opens, as shown in Figure 37.



Enter ASCII data file location: (press 'Enter' to update preview)

knime://knime.workflow/data/sales_2008-2011.csv

□ Preserve user settings for new location

Basic Settings

□ read row IDs

Column delimiter: ,

☑ read column headers

☑ ignore spaces and tabs

□ Java-style comments

Single line comment:

Preview

Click column header to change column properties (* = name/type user settings)

Row ID	S product	S country	S date	I			
Row0	prod_4	unknown	2008-12-12	1			
Row1	prod_3	China	2009-04-10	2			
Row2	prod_3	China	2009-04-10	2			
Row3	prod_3	China	2009-05-10	2			
Row4	prod_3	USA	2009-05-20	20			
Row5	prod_3	Brazil	2009-06-08	15			
Row6	prod_1	USA	2009-07-04	2			
Row7	prod_1	USA	2009-07-14	2			
Row8	prod_3	USA	2009-08-20	20			
Row9	prod_2	Germany	2009-11-02	15			
Row10	prod_2	Germany	2009-11-22	15			
Row11	prod_1	Germany	2009-12-02	1			
Row12	prod_1	China	2009-12-12	1			
Row13	prod_3	USA	2010-01-03	20			
Row14	prod_1	Germany	2010-01-10	1			
Row15	prod_3	Germany	2010-01-13	1			
Row16	prod_2	Germany	2010-01-15	25	1000	?	(
Row17	prod_2	USA	2010-01-20	2	80	?	(
Row18	prod_2	USA	2010-02-12	6	240	Y	(

New settings for column '...

Column Properties

□ DON'T include column in output table

Name: date

Type: String

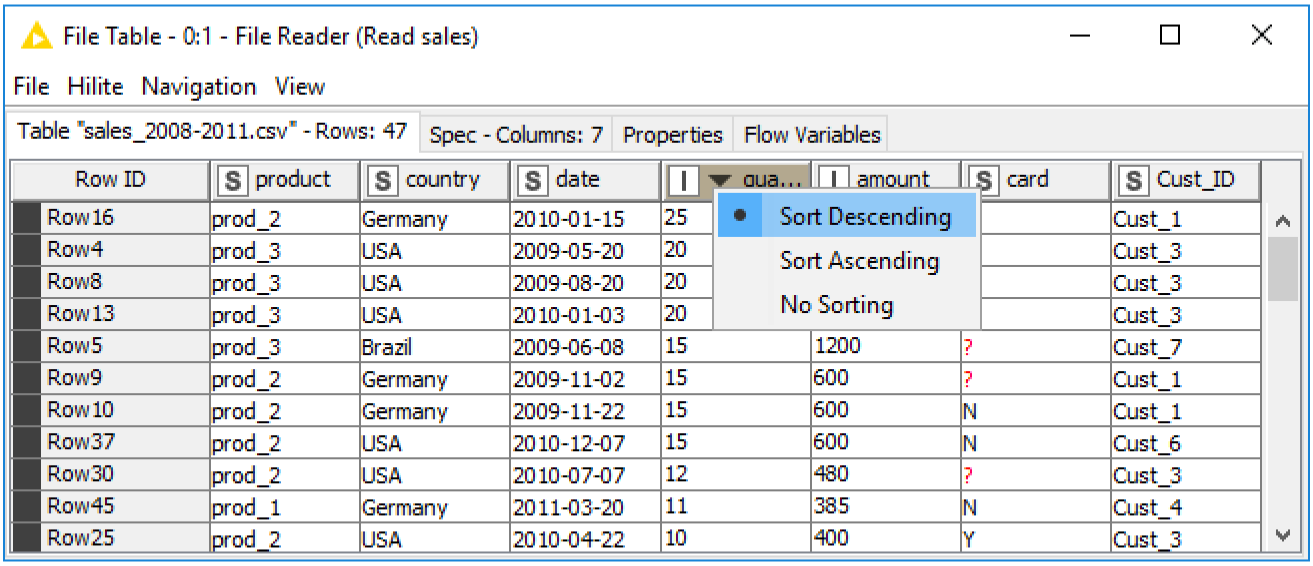
miss. value

Format:

Figure 37. Change column type in File Reader node

Sorting

Rows in the table view output can be sorted by values in one column by clicking the column header and selecting *Sort Descending* or *Sort Ascending* as shown in Figure 38. Note that this sorting only affects the current output view and has no effect on the node output.



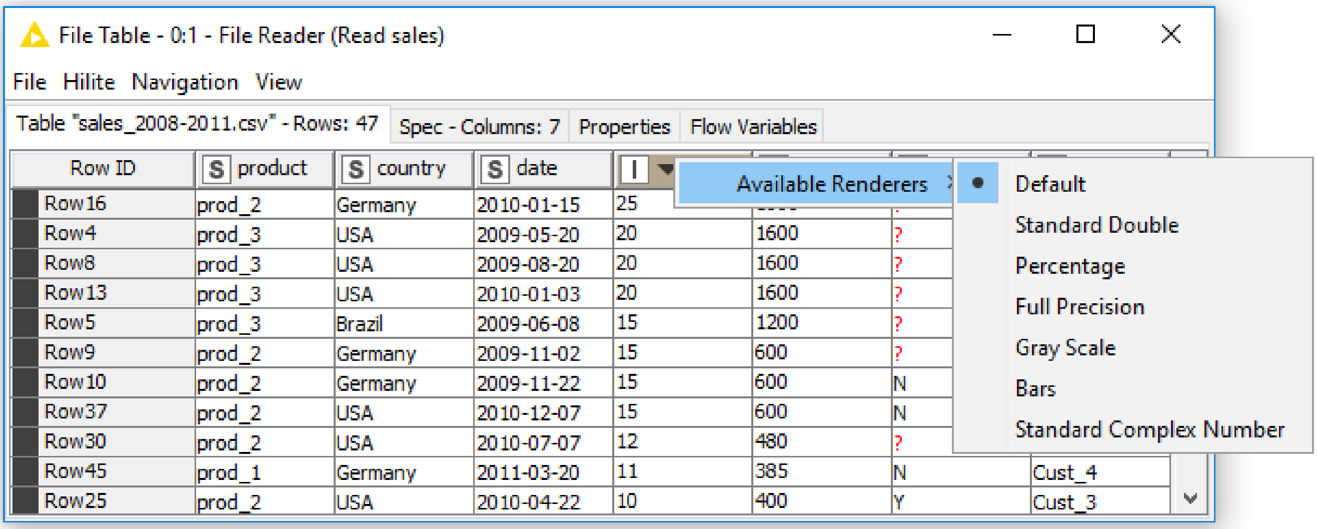
Row ID	S product	S country	S date	I qua...	I amount	S card	S Cust_ID
Row16	prod_2	Germany	2010-01-15	25			Cust_1
Row4	prod_3	USA	2009-05-20	20			Cust_3
Row8	prod_3	USA	2009-08-20	20			Cust_3
Row13	prod_3	USA	2010-01-03	20			Cust_3
Row5	prod_3	Brazil	2009-06-08	15	1200	?	Cust_7
Row9	prod_2	Germany	2009-11-02	15	600	?	Cust_1
Row10	prod_2	Germany	2009-11-22	15	600	N	Cust_1
Row37	prod_2	USA	2010-12-07	15	600	N	Cust_6
Row30	prod_2	USA	2010-07-07	12	480	?	Cust_3
Row45	prod_1	Germany	2011-03-20	11	385	N	Cust_4
Row25	prod_2	USA	2010-04-22	10	400	Y	Cust_3

Figure 38. Sorting data in table view

To sort rows in an output table permanently, use the Sorter node. Use the Column Resorter node to reorder columns.

Column rendering

In a table view output, you can also change the way in which numeric values are displayed in a data table. For example, it is possible to display numeric values as percentages, with full precision, or replace digits by a color scale or bars. To see these and other rendering options for a column, right click the column header, and select *Available Renderers* as shown in Figure 39. Note that these changes are temporary and have no effect on the node output.



Row ID	S product	S country	S date	I			
Row16	prod_2	Germany	2010-01-15	25			
Row4	prod_3	USA	2009-05-20	20	1600	?	
Row8	prod_3	USA	2009-08-20	20	1600	?	
Row13	prod_3	USA	2010-01-03	20	1600	?	
Row5	prod_3	Brazil	2009-06-08	15	1200	?	
Row9	prod_2	Germany	2009-11-02	15	600	?	
Row10	prod_2	Germany	2009-11-22	15	600	N	
Row37	prod_2	USA	2010-12-07	15	600	N	
Row30	prod_2	USA	2010-07-07	12	480	?	
Row45	prod_1	Germany	2011-03-20	11	385	N	Cust_4
Row25	prod_2	USA	2010-04-22	10	400	Y	Cust_3

Figure 39. Rendering data in table view

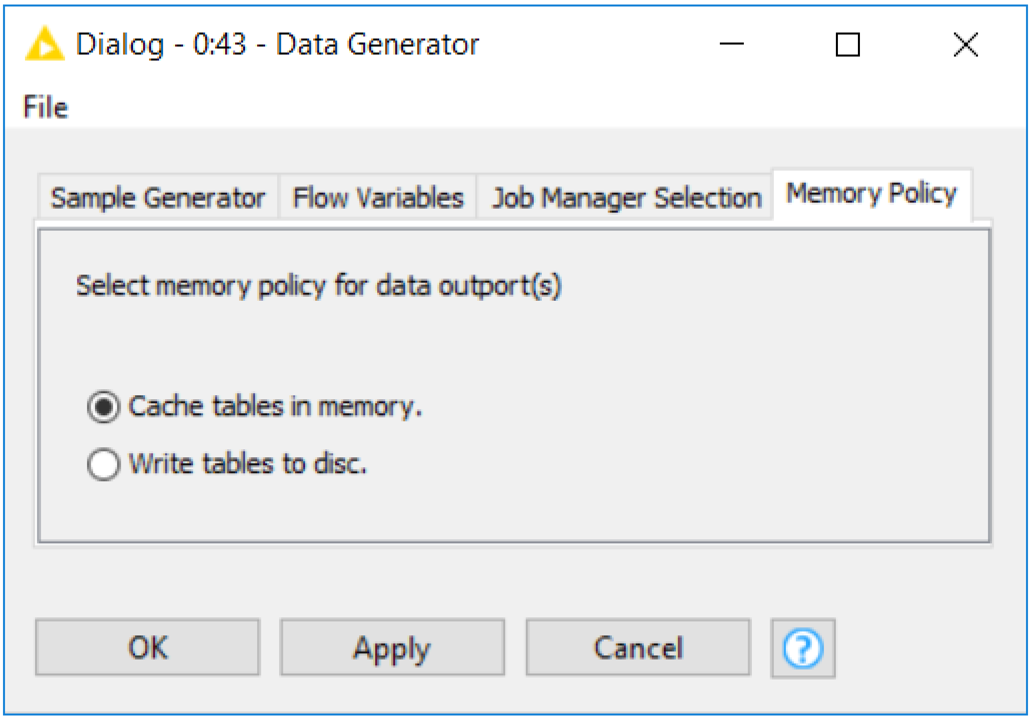
Table storage

When executed, many KNIME nodes generate and provide access to tabular data at their output ports. These tables might be small or large and, therefore, might fit into the main memory of the executing machine or not. Several options are available for configuring which tables to hold in memory as well as when and how to write tables to disk. These options are outlined in this section.

In-memory caching

KNIME Analytics Platform differentiates between small and large tables. Tables are considered to be small (large) when they are composed of up to (more than) 5000 cells. This threshold of 5000 cells can be adjusted via the `-Dorg.knime.container.cellsinmemory` parameter in the `knime.ini` file. KNIME Analytics Platform always attempts to hold small tables in memory, flushing them to disk only when memory becomes scarce.

In addition, KNIME Analytics Platform attempts to keep recently used large tables in memory while sufficient memory is available. However, it writes these tables asynchronously to disk in the background, such that they can be dropped from memory when they have not been accessed for some time or when memory becomes scarce. You can configure the memory consumption of a specific node to never attempt to hold its tables in memory and, instead, write them to disk on execution. This is helpful if you know that a node will generate a table that cannot be held in memory or if you want to reduce the memory footprint of a node.



Select memory policy for data outport(s)

● Cache tables in memory.

○ Write tables to disc.

Figure 40. Configuring a node's memory policy

Alternatively, by putting the line `-Dknime.table.cache=SMALL` into the `knime.ini` file, KNIME Analytics Platform can be globally configured to use a less memory-consuming, albeit much slower caching strategy. This strategy only ever keeps small tables in memory.

Disk storage

KNIME Analytics Platform compresses tables written to disk to reduce the amount of occupied disk space. By default, KNIME Analytics Platform uses the Snappy compression algorithm to compress its tables. However, you can configure KNIME Analytics Platform to use GZIP compression or no compression scheme at all via the `-Dknime.compress.io` parameter in the `knime.ini` file.

Columnar Table backend

Starting with KNIME Analytics Platform version 4.3 a new Columnar Table Backend is introduced. This extension addresses these issues by using a different underlying data layer (backed by Apache Arrow), which is based on a columnar representation.

For information on how to set up this type of backend please refer to the Table backend section.

Open for Innovation
KNIME

KNIME AG
Talacker 50
8001 Zurich, Switzerland
www.knime.com
info@knime.com