

# KNIME Anleitung zum Ausdrucken

KNIME AG, Zürich, Schweiz

Version 5.7 (letzte Aktualisierung auf )



## Inhaltsverzeichnis

[<a href="#page2" style="color: #000000; text-decoration: underline;">Einleitung . . . . .](#page2) [<a href="#page2" style="color: #000000; text-decoration: underline;">Ausdrückungsknoten .](#page2) [<a href="#page2" style="color: #000000; text-decoration: underline;">Allgemeines Verhalten](#page2) [<a href="#page3" style="color: #000000; text-decoration: underline;">Expression node . .](#page3) [<a href="#page4" style="color: #000000; text-decoration: underline;">Expression Row Filter](#page4) [<a href="#page5" style="color: #000000; text-decoration: underline;">Variable Expression](#page5) [<a href="#page6" style="color: #000000; text-decoration: underline;">Ausdruckssprache. . .](#page6) [<a href="#page7" style="color: #000000; text-decoration: underline;">Wertetypen und Literate](#page7) [<a href="#page12" style="color: #000000; text-decoration: underline;">Eingabedatenzugriff](#page12) [<a href="#page14" style="color: #000000; text-decoration: underline;">Betreiber . . . . .](#page14) [<a href="#page23" style="color: #000000; text-decoration: underline;">Funktionen . . . .](#page23) [<a href="#page25" style="color: #000000; text-decoration: underline;">Konstanten . . . .](#page25)

# Einleitung

In diesem Leitfaden finden Sie Dokumentation zu:

- Die [Expressionsknoten](#page2) : Erfahren Sie, wie man vielseitige Datenmanipulation in KNIME durchführt  
Workflows mit der KNIME Expression Language.
- Die [KNIME Sprache](#page6) : Anleitung für die Syntax, Semantik und Nutzung finden  
der KNIME Expression Language.

## Expressionsknoten

Derzeit gibt es drei Knoten in der KNIME Analytics Platform, die es Ihnen ermöglichen,  
die KNIME Expression Language, um Ihre Daten innerhalb von KNIME Workflows zu manipulieren:

- [Ausdruck](#) : Ermöglicht die Datenmanipulation von Zeilendaten, um Spalten hinzuzufügen oder zu ersetzen.
- [Filter für die Kategorie](#) : Filtert Zeilen basierend auf einer Bedingung.
- [Variabler Ausdruck](#) : Ermöglicht es Ihnen, Flussvariablen zu erstellen oder zu modifizieren.

Ziehen Sie einfach einen der Knoten aus dem Knoten-Repository und verbinden Sie ihn.

### Allgemeines Verhalten

Sie können die KNIME Expression-Knoten zur Manipulation Ihrer Daten verwenden. Die Knoten  
die KNIME Expression Language verwenden, die Sie in der

[KNIME Sprache](#page6)

Sie können Ihren Ausdruck im Expression Editor schreiben, indem Sie die von  
das Eingabefeld, das auf der linken Seite gefunden werden kann. Auf der rechten Seite finden Sie den Katalog  
alle verfügbaren Funktionen mit Dokumentation über ihre Nutzung. Sie können sie filtern und  
auch die verfügbaren Kategorien erweitern oder zusammenbrechen.

Die meisten Expression-Knoten unterstützen mehrere Ausdrücke, die in Folge ausgewertet werden. Du  
kann einen neuen Ausdruck hinzufügen, indem Sie auf [Ausdruck hinzufügen](#) Knopf. Jeder einzelne Ausdruck  
Editor hat eine [Steuerleiste](#) in der oberen rechten Ecke, die Sie den Ausdruck nach oben oder  
nach unten, duplizieren oder löschen.

Jeder Expressionseditor hat einen darunter angebrachten Ausgabeabschnitt, der die Konfiguration der  
Einstellungen der Ausgabe. Editoren werden von oben nach unten ausgewertet, so dass Sie  
das Ergebnis eines Editors im nächsten.

Es gibt eine Schaltfläche, um den Ausdruck auszuwerten und eine Vorschau des Ergebnisses zu erzeugen. Das ist

nützlich, um zu überprüfen, ob der Ausdruck korrekt ist und das Ergebnis der Manipulation zu sehen.

Zusätzlich integriert sich der Knoten mit der KNIME AI Assistant Erweiterung, die AI-Unterstützung der Expressionserzeugung und -änderung, die den Prozess weiter vereinfacht. Von der Frage K-AI zur Unterstützung erhalten Sie Vorschläge für Ausdrücke basierend auf der Spalte Namen und Spaltentypen in Ihrer Tabelle.

Selbst mit K-AI aktiviert, werden keine Daten aus der Tabelle an den AI-Service gesendet.

Expressionsknoten

Sie können den Expression-Knoten verwenden, um Zeilen-für-Reihen-Manipulation Ihrer Daten durchzuführen und hinzufügen oder Spaltendaten ersetzen.

Öffnen Sie den Konfigurationsdialog des Knotens und Sie werden etwas wie folgt sehen:

**Inputs & Outputs**

Available input data (via table or flow variable) usable in the expression. Double click or drag&drop to insert.

**Expression editors**

Multiple code editors to craft expressions. Provides auto completion and displays errors. Copy, reorder or delete editors with the control bar in the top right.

**Output target**

Choose where the result of the expression is stored. Replace an existing column or create a new one.

**Function catalog**

Built-in functions to manipulate data. Click on a function to see an extensive documentation about the usage of every function. Double click or drag&drop to insert.

**Result preview**

Evaluate the expression for a limited number of rows to see whether your expression works as expected.

Abbildung 1. KNIME Übersicht über den Index

In der      Ausgangsspalte      Abschnitt an der Unterseite jedes Ausdrucks-Editors können Sie wählen, ob Sie das Ergebnis der Expression in einer neuen Spalte ausgeben und der Spalte eine gewünschte Name oder die bestehende Spalte ersetzen.

Finden Sie ein Beispiel für die Verwendung des Expressionsknotens auf [KNIME Gemeinschaft](#)  
[Hubraum](#).

Sie können auf die      Erst 10 Zeilen auswerten      Taste oder wählen Sie die Anzahl der Zeilen, die Sie möchten auswerten, indem Sie auf      Icon. Sie können zwischen 10 (Standard), 100, 1000 wählen.

Berücksichtigung, dass dies mehr Zeit auf der Grundlage der Anzahl der Zeilen zu ausgewertet werden.

Expression Row Filter Node

Sie können den Expression Row Filter-Knoten verwenden, um Zeilen basierend auf einer in die KNIME Expression Language.

Öffnen Sie den Konfigurationsdialog des Knotens und Sie werden etwas wie folgt sehen:

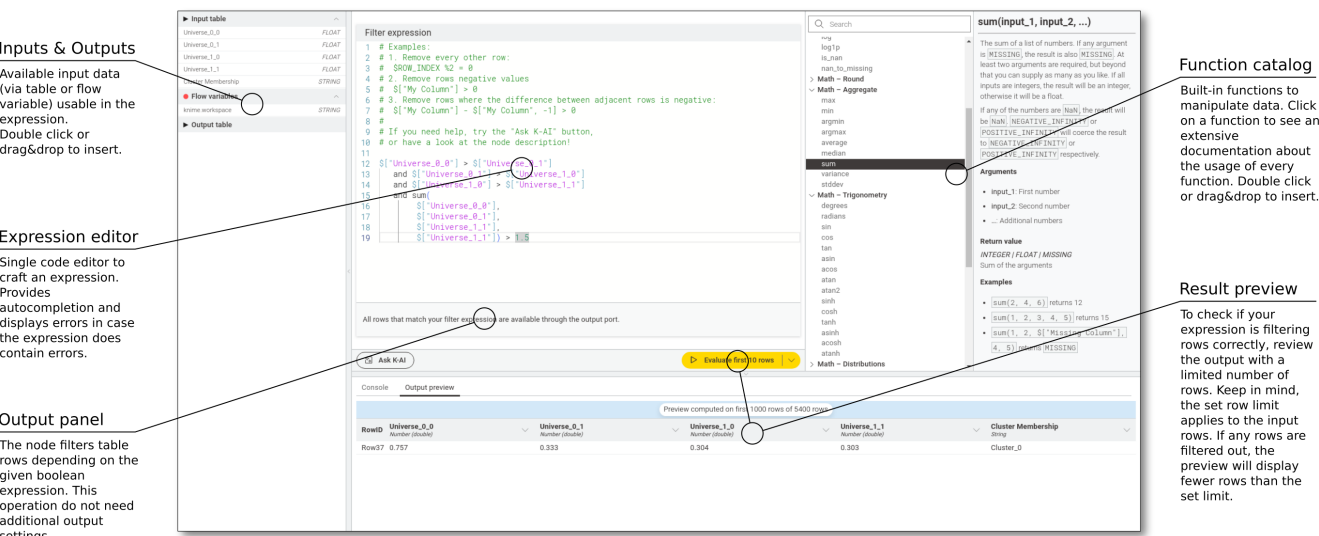


Abbildung 2. KNIME Expression Row Filter Knotenübersicht

Der Expression Row Filter-Knoten hat einen einzigen Expressions-Editor, in dem Sie die Bedingung definieren zur Filterung. Wenn der Ausdruck ausgewertet wird, werden die Zeilen, die die Bedingung nicht erfüllen, entfernt. Wenn es wertet zu TRUE, die Zeile bleibt im Ausgang. Die Ausgabe des Ausdrucks muss daher ein BOOLEAN Wert (für weitere Details zu den Typen siehe

Wenn Sie Zeilen basierend auf mehreren Bedingungen entfernen müssen, können Sie logische Operatoren wie AND, OR und NOT verwenden, um diese Bedingungen zu kombinieren (siehe Abschnitt für mehr Beispiele).

Sie können auf die Evaluate button icon klicken, um die Anzahl der Zeilen, die Sie möchten auswerten, indem Sie auf das Icon klicken. Sie können zwischen 10 (Standard), 100, 1000 wählen.

Die Grenze wird an die Eingangszeilen angelegt, nicht an die Ausgangszeilen. Wenn Zeilen sind gefiltert, die Vorschau wird weniger Zeilen als die eingestellte Grenze anzeigen. Nehmen Sie an, dass Sie 10 Zeilen auswerten, aber nur 5 Zeilen durch den Filter kommen, dann werden nur 5 Zeilen in der Vorschau angezeigt.

Variabler Expressionsknoten

Mit dem Variable Expression-Knoten können Sie Flussvariablen mithilfe des KNIME erstellen oder modifizieren Ausdruckssprache. Dieser Knoten ist nützlich, wenn Sie eine neue Flow-Variable erstellen möchten oder eine bestehende, basierend auf den Werten anderer Flussgrößen ändern. Es gibt keine Tabelleneingabe für diesen Knoten.

Öffnen Sie den Konfigurationsdialog des Knotens und Sie werden etwas wie folgt sehen:

**Inputs**

Available input flow variables usable in the expression. Double click or drag&drop to insert.

**Expression editors**

Multiple code editors to craft expressions. Provides autocompletion and displays errors. Copy, reorder or delete editors with the control bar in the top right.

**Output target**

Choose where the result of the expression is stored. Replace an existing flow variable or create a new one.

**Function catalog**

Built-in functions to manipulate data. Click on a function to see an extensive documentation about the usage of every function. Double click or drag&drop to insert.

**Output type**

The expression language uses only one integer type, i.e., LongType. If needed the output can be converted to an IntType.

**Result preview**

Evaluate the expressions to see whether your expression works and you have created or changed flow variables as expected.

Abbildung 3. KNIME Variable Expression-Knotenübersicht

In der Ausgangsstromvariable Abschnitt an der Unterseite jedes Expression Editors können Sie wählen, ob Sie wollen das Ergebnis des Ausdrucks als neue Flussgröße ausgeben und den Fluss geben einen gewünschten Namen verändern oder eine bestehende Durchflussgröße ersetzen.

Sie können auf die Bewertung Schaltfläche, um den Ausdruck zu bewerten und eine Vorschau der Ergebnis. Dies ist nützlich, um zu überprüfen, ob der Ausdruck korrekt ist und alle Flussgrößen verfügbar sind wie erwartet.

[Wiederholen Sie diesen Leitfaden](#)

in diesem Leitfaden unterstützt die KNIME Expression Language nur eine Art von Integralzahlen ( `INTEGER` ) während der KNIME Analytics Plattform unterstützt zwei Arten von Integralzahlen für Flussgrößen: `IntType` und `LongType` . Standardmäßig wird die Ausgabe eines numerischen Ausdrucks vom Fluss `LongType` . Wenn Sie eine `IntType` , gibt es einen Dropdown Menü im Ausgabebereich eines jeden Expressionseditors, in dem Sie die gewünschter Typ.

# Sprache

Die KNIME Expression Language ist eine spezialisierte Sprache für die Datenmanipulation und Analyse innerhalb von KNIME-Workflows. Sein Ziel ist es, eine intuitive und effiziente Weise zu bieten für die Benutzer Berechnungen, String-Manipulationen und Zeilen- oder Spaltenoperationen durchführen ohne umfangreiche Programmierkenntnisse. Dieses Dokument dient als Leitfaden für Syntax, Semantik und Nutzung der KNIME Expression Language.

Werttypen und Literatur

Die KNIME Expression Language unterstützt mehrere Basiswerttypen, die jeweils einen bestimmten

Art der Daten. Einige Operationen sind nur für eine Teilmenge von Werttypen gültig. Dies ist beschrieben  
gegebenenfalls. Jeder Typ kann optional sein, siehe Abschnitt über

Informationen.

[für mehr](#page11)

1 KNIME Expression Language verwendet leicht verschiedene Typen als KNIME  
Nutzung der Analytics Plattform Spaltentypen und Durchflussgrößen . Letzteres zwei  
nur in den Bezeichnungen unterscheiden. Im Folgenden werden die Arten des Ausdrucks  
Sprache werden beschrieben und wie sie zu den kombinierten Typen, die als  
(Spalte , Durchflussgrößentyp ) in der KNIME Analytics Plattform.

BOOLEAN

Der Werttyp **BOOLEAN** wird für logische Werte verwendet, die entweder wahr oder falsch sind. Bei der Handhabung

optionale Werte, d.h. Typ **BOOLEAN | MISSEN** , Kleines dreiwertige Logik wird angewendet werden. Für  
[<a href="#page18" style="color: #000000; text-decoration: none;">Details siehe Abschnitt](#page18)

**BOOLEAN** entweder **TR** oder **FALDEN** .

**BOOLEAN** in den Ausdruckssprachenkarten zu

Analyseplattform.

[Boolean, BooleanType]

[<a href="#page7" style="color: #000000; text-decoration: none;">Details siehe Abschnitt](#page7)

in KNIME

Anzahl Typen - **INTEGER** und **FLOSE**

Die KNIME Expression Language unterstützt nur eine Art von Integralzahlen ( **INTEGER** ) und  
eine Art schwimmender Punktzahlen ( **FLOSE** ) Für die Einfachheit sind unterschiedliche Präzisionen nicht  
unterstützt.

Für Vorhaben, die angewandt werden **INTEGER** und **FLOSE** , wie 3.14 , die **INTEGER**  
Wert wird in **FLOSE** automatisch. Dies kann zu einem Verlust an Präzision führen  
für sehr große Zahlen.

INTEGER

Der Werttyp **INTEGER** wird für ganze Zahlen verwendet. **INTEGER** Literatur sind in dezimal geschrieben  
als Ziffern zwischen 0) und <sup>ANSI</sup> mit optionalem <sup>ANSI</sup> zur visuellen Trennung. Die erste Ziffer kann nicht



<sup>0)</sup>es sei denn, die Nummer ist <sup>0)</sup>selbst.

[

Die Werte werden als 64-Bit unterschriebene zweier-komplement ganze Zahlen dargestellt  
einen Wertebereich von -9\_223\_372\_036\_854\_775\_808 bis  
223\_372\_036\_854\_775\_807 (inklusive).

KNIME Analytics Plattformtypen [Anzahl (Integer), IntType] und [Anzahl:  
Integer), LongTyp [1](#page7)  
werden abgebildet INTEGER in der Ausdruckssprache ohne Verlust  
Präzision. Die Ausgabe eines Ausdrucks, der die INTEGER Der Ausdruckstyp wird  
Der Spaltentyp Anzahl (Long Integer) . Für Flussgrößen gibt es ein Dropdown-Menü in  
den Ausgabeabschnitt jedes Expressionseditors, in dem Sie den gewünschten Typ auswählen können, d.h.  
IntTyp oder lang Typ .

FLOSE

Der Werttyp FLOSE wird für Zahlen mit fraktionierten Teilen verwendet.

A FLOSE die Zahl wird mit einem dezimalen Punkt geschrieben. Der Dezimalpunkt kann an jeder Position in  
die Zahl, auch am Anfang oder am Ende, wie 0,123 oder .123 oder 123. Sie können underscores verwenden  
Ja die Ziffern zu trennen, um große Zahlen leichter zu lesen, wie 1\_000.567\_890 gleich  
1000

Sie können schreiben FLOSE Zahlen mit wissenschaftlichen Angaben, die für sehr große oder sehr große  
kleine Zahlen. In der wissenschaftlichen Notation, <sup>e)</sup>oder E wird verwendet, um "Zeiten zehn zu der Macht von". Du  
kann auch ein Plus verwenden<sup>+</sup>oder minus <sup>Ja</sup>. nach dem <sup>e)</sup>oder E positive oder negative Exponenten angeben,  
und 1.23 bis 4 oder 1.23E+4 Mittel (+ 4) oder 12300

[

Die Syntax für FLOSE Literatur ist ähnlich wie die Syntax in [im Python](#)  
[Programmiersprache](#) . Die Werte sind als Doppelpräzision dargestellt.  
[Schwimmpunktzahlen](#) ( [64bit IEEE 754](#) ) Dies führt zu einem Wertebereich von 4.9E-  
324 bis 1.8E+ 308 (inklusive) und eine Präzision von etwa 15 Dezimalstellen.

FLOAT in den Sprachkarten der Ausdruckssprache [Anzahl (Float), DoubleTyp]  
Typ in der KNIME Analytics Plattform und hat die gleiche Präzision.

[1](#page7)  
Spalte

STRUKTUR

Der Werttyp STRUKTUR wird für Sequenzen von Unicode Zeichen (Text) verwendet. Die Werte sind

als Folge von in Doppelzitatn eingeschlossenen Zeichen dargestellt "Text" oder einzelne Zitate  
„Text“ .


Neue Zeilen in Strings sind erlaubt, so dass der String mehrere Zeilen ohne  
mit einem besonderen Charakter.

```
"multi-line  
String  
- >  
Multiline  
Zeichen
```

Fluchtsequenzen

Der Kampf ( kann für Fluchtsequenzen verwendet werden. Ein Gegenschlag, der nicht mit einem von  
Die folgenden Escape-Sequenzen sind eine ungültige Syntax.

Tabelle 1. Escape Sequenzen

Escape-Sequenz	Warenbezeichnung	Beispiel
\> neue Werte	Backslash und neue Linie in Eingabetext ignoriert	"xyz \ abc)" → xyz ab
	Ausweichen des Gegenspiels selbst	"\something\\" → \something\
('	Escaping einzelne Zitate	"\'quoted text\'" → 'angebener Text'
\"	Doppelte Zitate entkommen	"\"notiert text\"" → "angebener Text"
)	ASCII-Rückraum verursacht die Cursor rückwärts bewegen in der Vergangenheit Charakter	"Hallo, W\bWorld!" → Hallo, Welt!
Waren	ASCII Kutschenrückführung Ursachen der Cursor zu dem Anfang der Linie	"Hallo,\rWorld!" → Welt!

Escape-Sequenz	Warenbezeichnung	Beispiel
Waren	ASCII-Linefeed verursacht die Cursor zum nächsten Linie. Beachten Sie, dass auf unix-like Systeme, dies ist die einzige Charakter verwendet für Neuheiten und unter Windows-Systemen, es wird in Kombination mit	"Hallo,\nWorld!" → Hallo. Welt!
Warenbezeichnung	ASCII horizontale Tab Ursachen der Cursor zu dem Nächster Tab Stop	"Hallo,\tWorld!" → Hallo. Welt!
PERSONAL	Unicode-Zeichen können als Fluchtsequenzen verwendet. Die    xxxx   Teil ist eine 16-Bit-Hex Wert	"\u0041" → A "\u00E4" → ä "\u2328" → ☞

Escape-Sequenzen werden von links nach rechts und dem resultierenden Charakter einer Flucht ersetzt.  
Eine Sequenz kann nicht Teil einer anderen Fluchtsequenz sein.

STRUKTUR	in den Ausdruckssprachenkarten	(String, StringType)	<a href="#page7" style="color: #ff69b4;">in KNIME Analytics</a>
Plattform.			

Datumstypen

Die KNIME Expression Language unterstützt mehrere verschiedene Datums- und Zeittypen, gemeinsam „Datumstypen“ genannt. Die unterstützten Typen sind:

- LOCAL\_DAT , für Termine ohne Zeitangaben,
- LOCAL\_TIME , für Zeiten ohne Datumsinformationen und ohne Zeitzoneinformationen,
- ENTWICKLUNG , für kombinierte Daten und Zeiten ohne Zeitzone Informationen,
- ZONED\_DATE\_TIME , für kombinierte Daten und Zeiten mit Zeitzoneinformationen.

Diese Typen können nicht als Literal geschrieben werden, aber Spalten der entsprechenden Typen können in der Ausdruckssprache verwendet werden, und verschiedene Funktionen und Operationen sind verfügbar, die erstellen und manipulieren.

Dauertypen

Die KNIME Expression Language unterstützt Dauertypen. Diese stellen Zeitbeträge dar, wie der Unterschied zwischen zwei Daten oder zwei Mal. Die unterstützten Typen sind:

- `DATE_DURATION` , für Intervalle zwischen den Daten, welche Karte zur KNIME Analytics Laufzeit der Plattform,
- `TIME_DURATION` , für Intervalle zwischen den Zeiten, welche Karte zur KNIME Analytics Die Zeitdauer der Plattform.

Ähnlich wie die Datums- und Zeittypen können diese nicht als Literal geschrieben werden, sondern als Spalten der entsprechende Typen können in der Ausdruckssprache verwendet werden, und verschiedene Funktionen und Operationen sind verfügbar, die sie erstellen und manipulieren können.

MISSEN

Der Werttyp `MISSEN` wird für fehlende Werte verwendet. Es wird verwendet, um das Fehlen von Wert in einer Zelle oder Zeile, entweder weil der Wert in den Eingabedaten fehlte oder weil Wert konnte nicht berechnet werden.

Alle vorstehenden Arten außer `MISSEN` kann erweitert werden, um fehlende Werte während Bewertung. Dies wird durch `| MISSEN` , so eine Spalte vom Typ `INTEGER` `MISSEN` kann beide `INTEGER` Werte und `MISSEN` Werte.

Liter `MISSEN`

Der Literal für einen fehlenden Wert ist `MISSEN` . Es ist fallempfindlich und muss in Oberkörper. Die buchstäbliche, d.h. explizite Verwendung von `MISSEN` in einem Ausdruck nicht austauschbar mit dem optionalen Typ. Also, während `Einige_Funktion($"Spalte mit nur MISSING-Werten")` ist gültig, `Einige Funktionen (MISSING)` ist und wird zu einem Syntaxfehler führen. Für einen Ausdruck, Nur Rückgaben `MISSEN` ohne weitere Operation wird auch ein Syntax-Fehler als Eine Art des Ausdrucks wäre nicht definiert.

## Eingabedatenzugriff

### Zugang zum Netz

Um den Wert aus einer Spalte in der aktuellen Zeile abzurufen, stehen zwei Syntaxoptionen zur Verfügung:

Verwendung `$["column name"]` für alle Spaltennamen, einschließlich der Leerzeichen oder Sonderzeichen Zeichen. Die Spaltenbezeichnungsreferenz zwischen den quadratischen Klammern folgt den Regeln der STRUKTUR Literatur.

Bei Spaltennamen, die ausschließlich aus Buchstaben, Zahlen und Unterstrichen bestehen (ohne Start mit einer Nummer), eine kurze Syntax `$column_name` ist erlaubt.

Kolumnnamen sind case-sensitive.

- `$["Customer ID"]` Wert der Spalte „Kustomer ID“
- `$["Column mit einem \"double\" Zitat"]` Wert der Spalte „Spalte mit einem „Doppel“ Zitat ‘
- `$customer_id` Wert der Spalte „customer\_id“

Es gibt auch spezielle Kennungen, um auf die

- `$(ROW_NUMBER)` die aktuelle Zeilennummer ab 1.
- `$(ROW_INDEX)` den aktuellen Zeilenindex ab 0 erhalten.
- `$(ROW_ID)` die RowID zu bekommen, wie "Row99".



Die Zeilennummer, Zeilenindex und Zeilen-ID sind keine Spaltennamen und sind daher nicht in Zitaten eingeschlossen. Kurzhand-Syntax ist nicht erlaubt für diese spezielle Kennungen.

### Row Offsets

Manchmal muss man auf Werte aus anderen Zeilen in der Tabelle zugreifen, um

Berechnungen. Die KNIME Expression Language ermöglicht die Nutzung `$["column_name", Offset]` auf die Referenz früherer oder nächster Zeilen relativ zum aktuellen.

Der Offset ist eine statische Zahl und **nicht** sei ein Ausdruck selbst.

Negative Offsets weisen auf frühere Zeilen hin, positive Offsets auf Zeilen neben der aktuellen Zeile.

Das Wiederholen einer Spalte wird erst nach der Auswertung des Ausdrucks für die gesamte Tabelle wirksam.

Dies bedeutet, dass der Ausdruck nur die ursprünglichen Daten aus dieser Spalte verwendet.

• `$["column_name", -1]`                      Wert der Spalte „column\_name“ aus der vorherigen Zeile

□

Die Verwendung eines Offsets wird zwangsläufig auf Werte aus Zeilen zugreifen, die nicht existieren. In  
in diesem Fall wird das Ergebnis                      MISSEN                      .

## Flow Variabler Zugriff

Flow-Variablen werden mit Syntax ähnlich dem Zeilenzugriff, aber mit zwei Dollar-Zeichen aufgerufen:

Verwendung `$$["flow var name"]`                      für jeden flussvariablen Namen.

Für flussvariable Namen, die nur aus Buchstaben, Zahlen und Unterstrichen bestehen (ohne

beginnend mit einer Nummer, eine kurze Syntax                      `$$flow_var_name`                      gemäß Spalte  
Namen.

Flow-Variable-Namen sind case-sensitive.

Betreiber

Die KNIME Expression Language unterstützt eine Vielzahl von Operatoren für arithmetic, vergleichen, und logische Operationen. Die folgenden Abschnitte beschreiben die in den Sprache und ihre jeweiligen Regeln und Verhaltensweisen.

Bemerkungen

Text nach einem `#` Symbol gilt als Kommentar und wird vom Dolmetscher ignoriert. Kommentare können verwendet werden, um Code für Klarheit zu nennen. Kommentare können auf eine separate Zeile oder am Ende einer Codezeile.

```
# Dies ist ein Kommentar
1 + 2 # Dies ist ein weiterer Kommentar, aber "1 + 2" ist der Ausdruck
```

Arithmetik

In der folgenden Tabelle sind die im KNIME-Ausdruck verfügbaren arithmetischen Operatoren aufgeführt. Sprache, zusammen mit ihren Beschreibungen und Schreibhinweisen. Arithmetische Operationen gelten auch für optionale Typen. Wenn einer oder beide der Operanden fehlt, fehlt das Ergebnis. Für Klarheit, wir den optionalen Typ in der folgenden Tabelle ausgeben.

Tabelle 2. Arithmetische Operatoren

Name	Betreiber	Warenbezeichnung	Schreibhinweise
Addition	+	Rendite der Summe zwei Zahlen.	Anwendbar auf INTEGER und FLOSE .
Subtraktion	- Ja.	Ergibt den Unterschied von zwei Zahlen. Kann auch als Unary Operator, Den Operanden negieren.	Anwendbar auf INTEGER und FLOSE .
Multiplikation	*	Das Produkt von zwei Zahlen.	Anwendbar auf INTEGER und FLOSE .

Name	Betreiber	Warenbezeichnung	Schreibhinweise
Abteilung	/	Rendite des Quotienten von zwei Zahlen.	Gilt für <code>INTEGER</code> und <code>FLOSE</code> . Das Ergebnis ist immer von Typ <code>FLOSE</code> .
Abteilung für Boden	//	Rendite des Quotienten von zwei Zahlen zum nächsten <code>INTEGER</code> Nummer.	Nur anwendbar <code>INTEGER</code> Werte.
Exposition	**	Ergibt die Macht der zwei Zahlen.	Anwendbar auf <code>INTEGER</code> und <code>FLOSE</code> .
Restbestand	%	Der Rest aus der Teilung das erste Argument von die zweite.	Anwendbar auf <code>INTEGER</code> und <code>FLOSE</code> .

Sind beide Operanden gleichartig, so ist das Ergebnis gleich, wenn nicht spezifiziert ansonsten für den jeweiligen Betreiber. Ist eine oder beide der Operandentypen optional, so ergibt sich das Ergebnis ist optional. Wenn die Operanden vom Typ sind `INTEGER` und `FLOSE` (Bestellung irrelevant) `INTEGER` Wert wird in den nächsten Wert des Typs umgewandelt `FLOSE`, und das Ergebnis des Typs `FLOSE`.

Division durch Null

Die Aufteilung einer Zahl um Null mit dem Divisions-, Bodenteil- oder Restoperator ergibt eine Laufzeitwarnung. Der Ausgang der Operation wird über die folgenden Regeln definiert.

Tabelle 3. Division nach Null

Name	Betreiber	Zustand	Ausgangsleistung
Abteilung	/	Der erste Operand ist 0)	0. / 0 → NaN
		Beide Operanden haben das gleiche Zeichen	1. / 0 → INFINITÄT
		Die Operanden haben verschiedene Zeichen	-1. / 0 → - INFINITÄT



Name	Betreiber	Zustand	Ausgangsleistung
Abteilung für Boden	//	Ja	0) Bodenaufteilung kehrt zurück immer INTEGER
Restbestand	%	Der erste Operand ist Art FLOSE	NaN
		Beide Operanden Typ INTEGER	0)

Die arithmetischen Operatoren gelten auch für Datums- und Intervalltypen. Folgender Tabelle listet die für Datums- und Intervalltypen verfügbaren arithmetischen Operatoren auf.

Tabelle 4. Arithmetische Operatoren für Datums- und Intervalltypen

Name	Warenbezeichnung
Zusatz ( + )	<p>Kann auf:</p> <ul style="list-style-type: none"><li>• Datumsangaben + DATE_DURATION , um ein neues Datum zu geben Zeit des gleichen Typs mit Datum hinzugefügt (was lassen in der Regel die Zeitinformatio- nen unverändert, außer in Fällen, in denen dies ist aufgrund von z.B. Tageslicht nicht möglich Sparzeiten). Anmerkung: Datum muss der erste Operand sein.</li><li>• Datums-Zeit mit Zeitinformatio- nen + TIME_DURATION , um ein neues Datum zu geben Zeit des gleichen Typs mit Zeitdauer hinzugefügt (was Änderung der Datumsinformatio- nen). Anmerkung: die Datumszeit muss die erste sein Opernd.</li><li>• A DATE_DURATION mit einem DATE_DURATION oder TIME_DURATION mit einem TIME_DURATION ein Intervall als solange die beiden Operanden zusammenkamen.</li></ul>

Name	Warenbezeichnung
Subtraktion ( <input type="checkbox"/> Ja. )	<p>Kann auf:</p> <ul style="list-style-type: none"><li>• Datumsangaben - DATE_DURATION , um ein neues Datum zu geben Zeit des gleichen Typs mit Laufzeit ab dem Datum (die normalerweise die Zeit verlassen wird Angaben unverändert, ausgenommen in Fälle, in denen dies nicht möglich ist z.B. Tageslichtsparzeiten). Anmerkung dass die Datumsdauer die erste sein muss Opernd.</li><li>• Ein Datum mit Zeitinformationen - TIME_DURATION , um ein neues Datum zu geben Zeit des gleichen Typs mit von der Zeit abgezogene Dauer (die das Datum ändern kann) Informationen. Anmerkung: Datum muss der erste Operand sein.</li><li>• A DATE_DURATION mit einem DATE_DURATION oder TIME_DURATION mit einem TIME_DURATION um ein Intervall zu erhalten gleich der Differenz der beiden Operanden.</li><li>• Zwei LOCAL_DAT Werte, um eine DATE_DURATION gleich der Differenz der beiden Termine.</li><li>• Alle zwei LOCAL_TIME , alle zwei ENTWICKLUNG , oder zwei ZONED_DATE_TIME Werte, um eine TIME_DURATION gleich der Differenz der beiden Datums-Zeitwerte.</li></ul>

Name	Warenbezeichnung
Multiplikation ( * )	<div>Kann auf:</div> <div><div><div>• a</div><div>TIME_DURATION</div><div>mit einem</div><div>INTEGER</div><div>bis</div></div><div>erhalten</div><div>TIME_DURATION</div><div>Wert, der</div><div>die gleiche Länge wie das Original</div><div>Dauer multipliziert mit der Zahl.</div></div> <div><div>• a</div><div>DATE_DURATION</div><div>mit einem</div><div>INTEGER</div><div>bis</div></div> <div>erhalten</div> <div>DATE_DURATION</div> <div>Wert, der</div> <div>die gleiche Länge wie das Original</div> <div>Dauer multipliziert mit der Zahl.</div>

Vergleich

Vergleichsoperatoren werden verwendet, um zwei Zahlenwerte zu vergleichen ( FLOSE und INTEGER ) und Ausbeute aBOOLEAN Ergebnis. Die Vergleichsoperatoren gelten auch für Datum und Intervall Arten, mit Ausnahme der DATE\_DURATION Art, die keine Art von Bestellung. Es gibt zwei Arten von Vergleichen: Ordnung und Gleichheit.




Vergleiche geben nie optionale Ergebnisse zurück. Dies stellt sicher, dass optionale nicht durch Vergleich propagiert. Daher ist es weniger wahrscheinlich, dass das Ergebnis eines Ausdruck ist optional.

Allgemeine Vorschriften für Vergleichsbetreiber



Vergleich Operatoren vergleichen numerische, Intervall- und Datumstypen, während sie mit MISSEN immer wieder FALDEN. Gleiche Operatoren arbeiten an allen Arten, solange beide Operanden gleichartig sind oder MISSEN , mit Ausnahme von INTEGER und FLOSE kann überprüft werden für Gleichheit miteinander und ZONED\_DATE\_TIMEN kann überhaupt nicht auf Gleichheit überprüft werden.

Tabelle 5. Vergleicher

Name	Betreiber	Art	Anmerkungen
Weniger als	<	Bestellung	

Name	Betreiber	Art	Anmerkungen
Weniger als oder gleich Zu		Bestellung	Anmerkung: MISSEN < = MISSEN ist TR
Mehr zum Thema	>	Bestellung	
Großartig Than oder Gleichheit		Bestellung	Anmerkung: MISSEN > = MISSEN ist TR
Gleich	=oder =	Qualität	
Nicht gleich	!= oder 	Qualität	Gleich wie (a == ) b)

Bei der Prüfung von Datums- und Dauertypen arbeiten die Vergleichsoperatoren als folgendes:

- < und > die Datums-Zeitwerte in chronologischer Reihenfolge vergleichen. Eine spätere Datumszeit ist als größer als eine frühere Datumsfrist betrachtet. Eine längere Dauer wird berücksichtigt größer als eine kürzere Dauer. Anmerkung: DATE\_DURATION wird nicht bestellt.
-  und  mit > und <, aber auch Gleichheit.
- = und != die Datums-Zeit-Werte für Gleichheit vergleichen. Zwei Datums- oder zwei Dauer Werte gelten als gleich, wenn sie den gleichen Typ haben und den gleichen Zeitpunkt oder die gleiche Dauer. Beachten Sie, dass diese Betreiber nicht anwendbar auf ZONED\_DATE\_TIME Werte.

Logische Operatoren

Die logischen Operatoren und , oder, und nichtAnwendung BOOLEAN Typen sowie optional BOOLEAN | MISSEN Typen. und und oderbinäre Operatoren sind, nichtist ein unary Operator.

Wenn beide Operanden vom Typ sind BOOLEAN , das Ergebnis des Typs BOOLEAN . Wenn die Art eines oder beide Operanden sind optional, d.h. BOOLEAN | MISSEN , fehlende Werte werden als unbekannt nach Kleenes dreiwertige Logik .

Tabelle 6. Logistische Operatoren

Name	Betreiber	Warenbezeichnung	Beispiele
Logischer und	und	Erträge TR wenn beide Operanden sind TR . Erträge FALDEN wenn mindestens ein Operand ist FALDEN und MISSEN ansonsten.	TRUE und FALSE → FALDEN TRUE und MISSING → MISSEN FALSE und MISSING → FALDEN
Logischer ODER	oder	Erträge FALDEN wenn beide Operanden sind FALDEN . Erträge TR mindestens ein Operand ist TR , andere MISSEN .	TRUE oder FALSE → TR TRUE oder MISSING → TR MISSEN oder FALSE → MISSEN
Logischer NICHT	nicht	Erträge TR wenn Opernd ist FALDEN . Erträge FALDEN wenn Opernd ist TR und MISSEN wenn Opernd ist MISSEN .	nicht TRUE → FALDEN nicht FALSE → TR nicht MISSING → MISSEN

String-Konzentration

Der Betreiber `+` kann auch zur Saitenverkettung verwendet werden, wenn mindestens einer der Operanden von Typ `STRUKTUR` oder `STRUKTUR` ist.

Ein Literal `MISSEN` ist kein unterstützter Typ und führt zu einem Syntaxfehler. Die Ausgangstyp einer Stringverkettung ist immer ein `STRUKTUR`. Fehlende Werte in der Eingabedaten werden dem String zugeordnet „MISSEN“.

```
"Hallo" + " " + "Welt" -> "Hallo Welt"
"Hello" + 42 -> "Hello42"
"Hello" + $["Säule mit fehlendem Wert"] -> "HelloMISSING"
"Hallo" + MISSING -> Syntax-Fehler
```

Versäumter Steinkohlenbergbau

Der fehlende Koaleszenzbetreiber `??` ist ein binärer Operator, der den linken Operanden zurückgibt, wenn er nicht `MISSEN`, sonst gibt es den richtigen Operanden zurück. Beide Operanden müssen den gleichen Typ haben und das Ergebnis gleichartig ist. Wenn beide Operanden sind `MISSEN` Werte, das Ergebnis ist `MISSEN`. Obwohl es selten nützlich sein wird, können Sie `MISSEN` als eines der Argumente `??`. Allerdings `MISSING? MISSEN` wird als Syntaxfehler behandelt.

```
1? 2 —> 1
MISSING? 2 —> 2.
MISSING? MISSING -> Syntax-Fehler
```

Vorfahren des Betreibers

Das Vorzeichen des Betreibers definiert die Reihenfolge, in der Operationen in einem Ausdruck ausgewertet werden, wenn es mehr als einen Bediener in Serie enthält. Sie können immer Klammern verwenden `( , )` bis eine bestimmte Bewertungsordnung durchsetzen. Die folgende Tabelle listet die Betreiber in Reihenfolge Vorrang, von höchster bis niedrigster.

- ANHANG Missing Coalescing ( `??` )
- 2. Exposition ( `**` )
- 3. Verhandlungen (ungültig<sub>Ja.</sub>)
- 4. Multiplikation ( `*` ), Abteilung ( `/` ), Remainder ( `%` ), Integer Division ( `//` )
- 5. Zusatz ( `+` ), Subtraktion ( `-`<sub>Ja.</sub> )
- 6. Vergleicher ( `<`, `>`, `==`, `!=`, `<=`, `>=` )
- 7. Logisch NICHT ( `!` nicht )
- 8. Logisch und ( `&` und )
- 9. Logischer OR ( `|` oder )

Operationen mit höherem Vorrang werden vor denen mit niedrigerem Vorrang bewertet. Operationen mit der gleichen Vorleistungsstufe werden von links nach rechts mit Ausnahme von `**` die von rechts nach links ausgewertet wird. Im Folgenden geben wir einige Beispiele, um die Vorrang der Betreiber.

Tabelle 7. Betreiberbeispiele

Ausdruck	Mit Parenthesis	Ergebnis	Erläuterung
1 + 2 * 3	1 + 2 * 3)	7	Multiplikation ist bewertet.
1 + 2 ** 3 * 4	1 + (( 2 ** 3) * 4)	ANHANG	Exposition ist vor Multiplikation und Multiplikation ist vor Ergänzung
2 * 2 ** 3 ** 2	2 * (2 ** (3 ** 2))	1024	Exposition ist zuerst bewertet und von rechts nach links
TRUE oder FALSE und FALDEN	TRUE oder (FALSE) FALSE)	TR	und wird zuerst ausgewertet

Funktionen

Es gibt zwei Arten von Funktionen, die in einem Ausdruck verwendbar sind, der sich in [dem folgenden Abschnitt](#page23) (Zelle) oder [dem folgenden Abschnitt](#page23) (Zeile) findet. Die Funktionen werden in zwei Kategorien unterteilt: Zeilen- und Spaltenfunktionen. Zeilenfunktionen werden reihenweise angewendet, um einen neuen Wert zu erzeugen, der dann in der nächsten Zeile verwendet werden kann. Spaltenfunktionen werden auf eine ganze Spalte angewendet, um einen einzigen Wert zu erzeugen, der für jede ausgewertete Zeile verwendet werden kann.

Es gibt einen Funktionskatalog im Editor, der bei der Auswahl der Funktionen und ihre Argumente durch detaillierte Beschreibungen und Beispiele. Sie werden eingebaut finden [Konstanten](#page25) auch.

Zeilenmäßige Funktionen

Funktionen, die reihenweise ausgewertet werden, sind immer kleiner und werden über die Funktion aufgerufen Name, gefolgt von Klammern, die Argumente enthalten:

Funktion\_name(arg1, arg2, ... )

Jede Funktion hat eine bestimmte Anzahl von Argumenten und Typen, die sie erwartet. Wenn die Argumente nicht mit den erwarteten Typen übereinstimmen, wird ein Typfehler erhöht. Die Rückgabeart einer Funktion ist durch die Funktion selbst bestimmt und nicht notwendigerweise gleich den Eingangstypen ist.



Jede Funktion gibt einen Wert zurück und es gibt keine Leer Funktionen. Funktionen kann geschachtelt werden, d.h. ein Funktionsaufruf kann ein Argument zu einer anderen Funktion sein.

Wenn es mehrere Argumente gibt, müssen sie durch Kommas getrennt werden. Jedes Argument kann jeder gültige Ausdruck. Nach dem letzten Argument können Sie optional eine Nachfolgekoma einschließen.

Beispiele:

```
sqrt(4) -> 2.
Pow(abs(-sqrt(3.14*2)),2) -> 3.14
if(TRUE, "true branch","false branch") -> "true branch"
```

Aggregationsfunktionen

Aggregationsfunktionen sind eine spezielle Reihe von Funktionen, die mit COLUMN\_ Berechnung Aggregationen über ganze Spalten, wie deren Mindest-, Höchst- oder Mittelwerte, für Beispiel: COLUMN\_MIN("Column Name") .



□

Die Aggregationsfunktionen nehmen einen String literal "Column name" anstatt eines Wertes aus einer Zeile ( `$["column name"]` oder `$column_name` ) als Eingabe.

In Aggregationsfunktionen bieten wir an, die Argumente positionell und nach Namen der Argument. Positionsargumente sind immer zuerst, gefolgt von benannten Argumenten. Name Argumente werden immer als `arg_name = value` .

Lassen Sie uns zeigen, dass für die Aggregationsfunktion `COLUMN_AVERAGE(Spalte, ignorieren_nan)`

- Nur Positionsvorschläge: `COLUMN_AVERAGE("Column Name", TRUE)`
- Nur benannte Argumente: `COLUMN_AVERAGE(column = "Column Name", ignore_nan = TRUE)`
- Gemischte Argumente: `COLUMN_AVERAGE("Column Name", ignorieren_nan = TRUE,)`

Konstanten

Die KNIME Expression Language bietet eine Reihe von vordefinierten Konstanten, die in Ausdrücken. Diese Konstanten dienen zur Darstellung gemeinsamer mathematischer Werte und Sonderwerte. Folgende Konstanten sind vorgegeben und können in Ausdrücken verwendet werden:

Tabelle 8. Konstanten

Name	Symbol	Typ	Warenbezeichnung
Wahrheitswert	TR	BOOLEAN	Der Boolean wahr .
Falscher Wert	FALDEN	BOOLEAN	Der Boolean falsch .
Eulers Nummer <sup>e)</sup>	E	FLOSE	Eulers Nummer, ~2.71828, gebraucht als die Basis der natürlichen Logarithmen und in exponentiell Funktionen.
Pi oder $\pi$	PI	FLOSE	Die Konstante Pi, ~3.14159, das Verhältnis von einem Kreis Umfang zu seiner Durchmesser.
Positive Unendlichkeit	INFINITÄT	FLOSE	Eine besondere Konstante positiv Infinity.
Nicht eine Nummer	NaN	FLOSE	Eine besondere Konstante von "Nicht ein Nummer".
Kleiner positiver Flossen	ZEITSCHRIFT	FLOSE	Die kleinste positiver Schwimmwert darstellbar durch Computer.
Großer positiver Schwimmer	MAX_FLOAT	FLOSE	Der größte positive Wert, der als FLOAT.

Name	Symbol	Typ	Warenbezeichnung
Kleinstes Negativ Flossen	MIN_FLOAT	FLOSE	Die kleinste negativer Wert darstellbar als FLOAT.
Große positive ganze	MAX_INTEGER	INTEGER	Der größte positive Wert, der als ein INTEGER.
Kleinstes Negativ ganze	MIN_INTEGER	INTEGER	Die kleinste negativer Wert darstellbar als INTEGER.
Fehlender Wert	MISSEN	MISSEN	Eine besondere Konstante für fehlender Wert.

KNIME AG  
Talacker 50  
8001 Zürich, Schweiz  
[www.knime.com](http://www.knime.com)  
[Info@knime.com](mailto:Info@knime.com)