



Spoon version 2.1

© 2001 - 2005 *i-Bridge*

www.kettle.be

Table of contents

1 Spoon.....	5
1.1 What is spoon?.....	5
1.2 Installation.....	5
1.3 Launching spoon.....	5
1.4 Screenshot.....	6
1.5 Command line options.....	7
1.6 Repository.....	8
1.7 License.....	8
1.8 Notes.....	8
1.9 Definitions.....	9
1.10 Toolbar.....	10
1.11 Options.....	11
1.11.1 General tab.....	11
1.11.2 Look & Feel tab.....	12
2 Database Connections.....	14
2.1 Description.....	14
2.2 Screenshot.....	14
2.3 Options.....	14
2.4 Database Usage Grid.....	15
2.5 Usage.....	16
2.5.1 Create a new connection.....	16
2.5.2 Edit a connection.....	16
2.5.3 Duplicate a connection.....	16
2.5.4 Copy to clipboard.....	16
2.5.5 Delete a connection.....	16
2.5.6 Test a connection.....	16
2.5.7 Execute SQL commands on a connection.....	17
2.5.8 Clear DB Cache option.....	17
2.5.9 Explore.....	17
2.6 Unsupported databases.....	17
3 SQL editor.....	18
3.1 Description.....	18
3.2 Screenshot.....	18
4 Database Explorer.....	19
4.1 Screenshot.....	19
4.2 Description.....	19
5 Hops.....	20
5.1 Description.....	20
5.2 Screenshot.....	20
5.3 Creating A Hop.....	20
5.4 Splitting A Hop.....	20
5.5 Loops.....	20
6 Transformation settings.....	21
6.1 Description.....	21
6.2 Screenshots.....	21
6.3 Options.....	23
6.4 Extra.....	24
7 Steps.....	25
7.1 Description.....	25
7.2 Launching several copies of a step.....	25

7.3 Distribute or copy?	26
7.4 Step Types	27
7.4.1 Text File Input	27
7.4.2 Text File Output	32
7.4.3 Table input	35
7.4.4 Table output	38
7.4.5 Select values	39
7.4.6 Filter rows	41
7.4.7 Database lookup	43
7.4.8 Sort rows	44
7.4.9 Stream lookup	45
7.4.10 Add sequence	47
7.4.11 Dimension update/lookup	49
7.4.12 Combination update/lookup	53
7.4.13 Dummy (do nothing)	55
7.4.14 Join Rows (Cartesian product)	57
7.4.15 Aggregate Rows	59
7.4.16 Get System Info	60
7.4.17 Generate Rows	62
7.4.18 Java Script Value	63
7.4.19 Call DB Procedure	73
7.4.20 Insert / Update	74
7.4.21 Update	75
7.4.22 Row Normaliser	76
7.4.23 Split Fields	79
7.4.24 Unique rows	81
7.4.25 Group By	82
7.4.26 Copy rows to result	83
7.4.27 Get rows from result	83
7.4.28 XBase input	84
7.4.29 Excel input	85
7.4.30 Database Join	87
7.4.31 Cube output	88
7.4.32 Cube input	88
7.4.33 Calculator	89
7.4.34 Execute SQL script	91
8 Graphical View	93
8.1 Description	93
8.2 Adding steps	93
8.2.1 Dragging	93
8.2.2 Creating a new step	94
8.2.3 Hiding a step	94
8.2.4 Step options (popup menu)	94
8.3 Adding hops	96
9 Log View	97
9.1 Description	97
9.2 Screenshot	97
9.3 Log Grid	97
9.4 Buttons	98
9.4.1 Start transformation	98
9.4.2 Preview	99
9.4.3 Show error lines	99
9.4.4 Clear log	99
9.4.5 Log Settings	100
10 Grids	101

User manual Last updated: 11/09/2005	Kettle ETL Environment	
	Spoon 2.1 by i-Bridge	

10.1 Description.....	101
10.2 Functions.....	101
10.3 Navigating.....	101
11 Repository Explorer.....	102
11.1 Description.....	102
11.2 Screenshot.....	102
11.3 Functions.....	102
11.4 Backup / Recovery.....	103

1 SPOON

1.1 What is spoon?

Kettle is an acronym for “Kettle E.T.T.L. Environment”. This means it has been designed to help you with your ETL needs: the Extraction, Transformation, Transportation and Loading of data.

Spoon is a graphical user interface for that allows you to design transformations that can be run with the Kettle tool Pan. Pan is a data transformation engine that is capable of performing a multitude of functions such as: reading, manipulating and writing data to and from various data sources.

NOTE: *For a complete description of Pan, please check out the Pan documentation.*

Transformations can describe themselves using an XML file or can be put in a Kettle database repository. This information can then be read by Pan to execute the described steps in the transformation.

In short: ***Kettle makes data warehouses easier to build, update and maintain!***

1.2 Installation

The first step is the installation of Sun Microsystems Java Runtime Environment version 1.4 or higher. You can download a JRE for free at <http://www.javasoft.com/>.

After this, you can simply unzip the zip-file: Kettle-2.1.zip in a directory of your choice. In the Kettle directory where you unzipped the file, you will find a number of files. Under Unix-like environments (Solaris, Linux, MacOS, ...) you will need to make the shell scripts executable. Execute these commands to make all shell scripts in the Kettle directory executable:

```
cd Kettle
chmod +x *.sh
```

1.3 Launching spoon

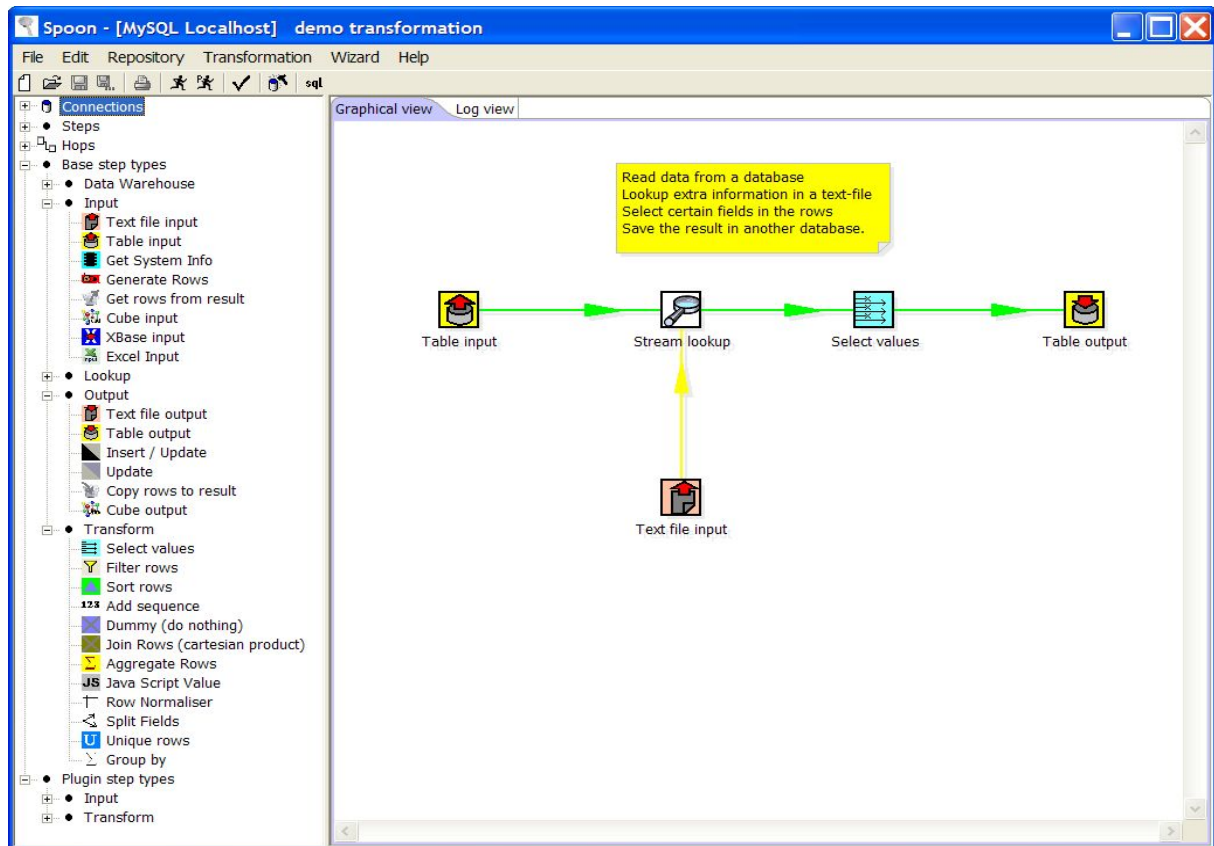
To launch Spoon on the different platforms these are the scripts that are provided:

- ✓ Spoon.bat: launch Spoon on the Windows platform.
- ✓ spoon.sh: launch Spoon on a Unix-like platform: Linux, Solaris, AIX, HP-UX, MacOS X

If you want to make a shortcut under the Windows platform an Icon is provided: “spoon.ico” to set the correct icon. Simply point the shortcut to the Spoon.bat file.



1.4 Screenshot



The following items are visible in Spoon: Connections, Steps, Hops, Base Step Types, Graphical View and the Log View.
These items are described in detailed in the chapters below: Connections in §2, Hops in §5, Steps in §7, the Graphical View in §8

1.5 Command line options

These are the command line options that you can use.
 Fields in *italic* represent the values that the options use.

```
-file=filename
```

This option runs the transformation defined in the XML file. (.ktr : Kettle Transformation)

```
-log=Logging Filename
```

Specifies the log file. The default is the standard output.

```
-level=Logging Level
```

The level option sets the log level for the transformation that's being run.
 These are the possible values:

- ✓ Error: Only show errors
- ✓ Nothing: Don't show any output
- ✓ Minimal: Only use minimal logging
- ✓ Basic: This is the default basic logging level
- ✓ Detailed: Give detailed logging output
- ✓ Debug: For debugging purposes, very detailed output.
- ✓ Rowlevel: Logging at a row level, this can generate a lot of data.

```
-rep=Repository name
```

Connect to the repository with name "*Repository name*".
 You also need to specify the options `-user`, `-pass` and `-trans`.
 The repository details are loaded from the file `repositories.xml` in the local directory or in the Kettle directory: `<homedirectory>/kettle/`

```
-user=Username
```

This is the username with which you want to connect to the repository.

```
-pass=Password
```

The password to use to connect to the repository

```
-trans=Transformation Name
```

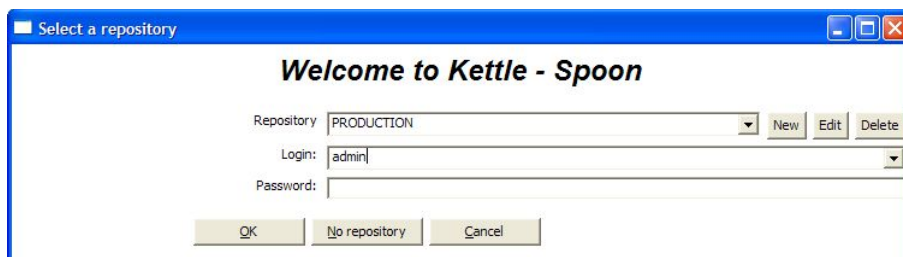
Use this option to select the transformation to run from the repository

NOTE: *It's important that if spaces are present in the option values, you use quotes or double quotes to keep them together.*

1.6 Repository

A Kettle repository can contain among other things transformations. This means that in order to load a transformation from a database repository, you need to connect to this repository.

To do this, you need to define a database connection to this repository. You can do this using the repositories dialog you are presented with when you start up Spoon.



The information concerning repositories is stored in a file called "repositories.xml". This file resides in the hidden directory ".kettle" in your default home directory.

HINT: The complete path and filename of this file is displayed on the Spoon console.

IMPORTANT: The default password for the admin user is also admin. You should change this default password right after the creation using the Repository Explorer.

1.7 License

When Spoon starts up it checks whether or not a valid license is present for the computer system it's running on. The license is checked against the hardware address of the first network card it finds in the system.

If you don't have a license, simply send a mail to info@kettle.be specifying this hardware address (MAC-Address) as well as your name, company name and the products for which you want to obtain a license from i-Bridge. If you did not purchase the software before sending the mail, make sure to include invoicing information as well. (VAT number and address information)

If no license is present, Spoon will run in demo mode to allow users to test its functionality.

NOTE: Spoon is free for non-profit organisations and for educational purposes.

1.8 Notes

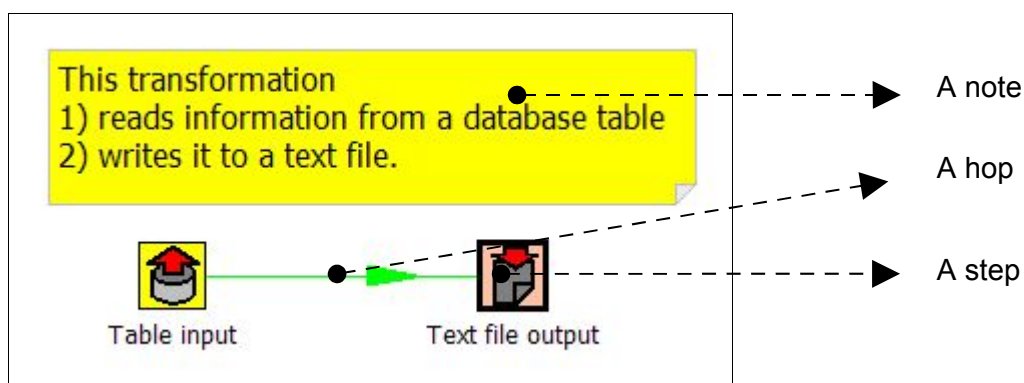
You can put notes on the graphical view everywhere simply by clicking right on the canvas and selecting "Add note".

Later these notes can be edited by double clicking on them and dragged around the screen by dragging on them with the mouse using the left button.

Removing a note can be done by a right click on the note and by selecting "Delete note".











1.9 Definitions

- ✓ Value: a value is part of a row and can contain a String, a floating point Number, an Integer, a Date or a Boolean value.
- ✓ Row: a row exists of 0 or more values
- ✓ Output stream: an output stream is a stack of rows that leaves a step.
- ✓ Input streams: an input stream is a stack of rows that enters a step.
- ✓ Hop: a hop is a graphical representation of one or more data streams between 2 steps. A hop always represents the output stream of one step and the input stream of another. The number of streams is equal to the copies of the destination step. (1 or more)
- ✓ Note: a note is a piece of information that can be added to a transformation



1.10 Toolbar

The icons on the toolbar of the main screen are from left to right:

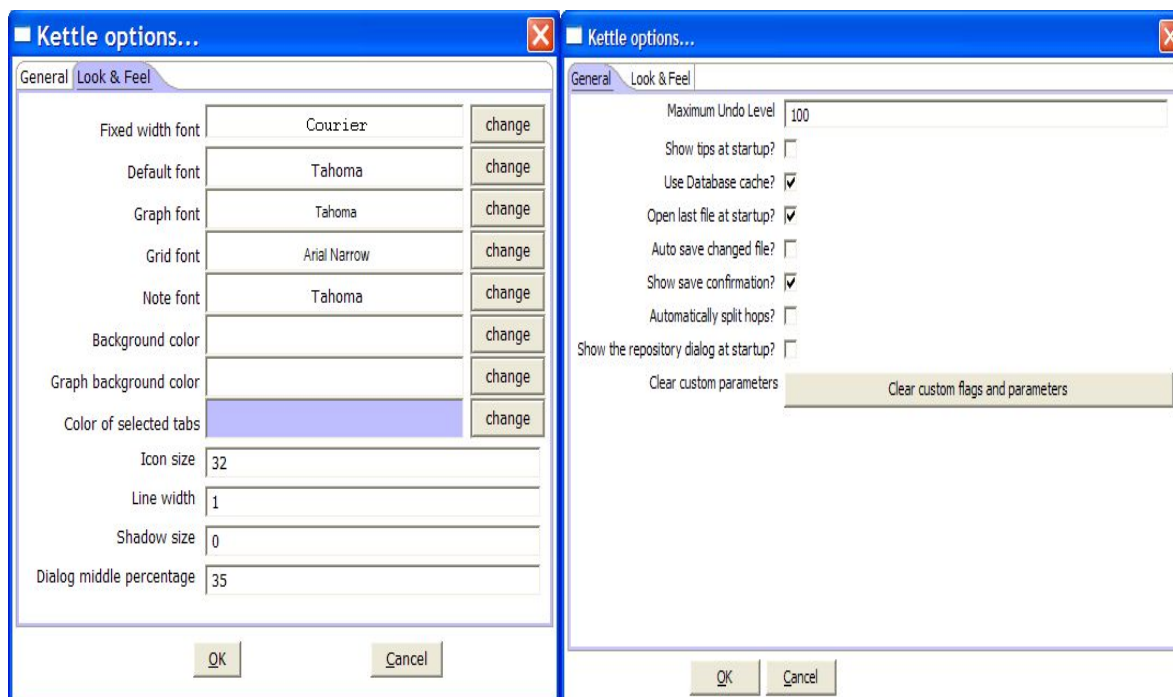
<i>Icon</i>	<i>Meaning</i>
	New transformation
	Open transformation from file if you're not connected to a repository or from the repository if you are connected to one.
	Save the transformation to a file or to the repository.
	Save the transformation under a different name or filename.
	Print: you will be presented with a print-dialog asking you to specify the number of pages, margins etc.
	Run transformation: runs the current transformation from XML file or repository.
	Preview transformation: runs the current transformation from memory. You can preview the rows that are produced by selected steps.
	Verify transformation: Spoon runs a number of checks for every step to see if everything is going to run as it should.
	Run an impact analyses: what impact does the transformation have on the used databases.
	Generate the SQL that's needed to run the loaded transformation.

1.11 Options

There are a number of options that you can change to enhance the graphical user interface, such as the fonts and the colors of the screens.

To change these options, please check out the Edit menu option “Options”.

These are screenshots of the options:



1.11.1 General tab

1.11.1.1 Maximum Undo Level

This parameter sets the maximum number of steps that can be undone (or redone) by Spoon.

1.11.1.2 Show tips at startup

This option sets the display of tips at startup.

1.11.1.3 Use database cache

Spoon caches information that is stored on source and target databases. In some cases this can lead to incorrect results when you're in the process of changing those very databases. In those cases it is possible to disable the cache altogether in stead of clearing the caches every time.

NOTE: Spoon automatically clears the database cache when you launch DDL (Data Definition Language) statements towards a database connection. However, when using 3rd party tools, clearing the database cache manually may be necessary.

1.11.1.4 Open last file at startup

Enable this option to automatically (try to) load the last transformation you used (opened or saved) from XML or repository.

1.11.1.5 auto save changed file

This option automatically saves a changed transformation before running.

1.11.1.6 show save confirmation

This flag allows you to turn off the confirmation dialogs you receive when a transformation has been changed.

1.11.1.7 automatically split hops

This option turns off the confirmation dialogs you get when you want to split a hop. (see also §5.4)

1.11.1.8 show the repositories dialog at startup

This option controls whether or not the repositories dialog shows up at startup.

1.11.1.9 Clear custom flags and parameters

This option clears all parameters and flags that were set in the plugin or step dialogs.

1.11.2 Look & Feel tab

1.11.2.1 Default font

This is the font that's used in the dialog boxes, trees, input fields, etc.

1.11.2.2 Graph font

This is the font that's used on the graphical view.

1.11.2.3 Grid font

This font is used in all the grids that are used in Spoon.

1.11.2.4 Note font

This font is used in the notes that are displayed in the Graphical View.

1.11.2.5 Background color

Sets the background color in Spoon. It affects all dialogs too.

1.11.2.6 Graph background color

Sets the background color in the Graphical View of Spoon.

1.11.2.7 Color of selected tabs

This is the color that's being used to indicate tabs that are active/selected.

1.11.2.8 Icon size

This affects the size of the icons in the graph window. The original size of an icon is 32x32 pixels. The best results (graphically) are probably at sizes 16,24,32,48,64 and other multiples of 32.

1.11.2.9 Line width

This affects the line width of the hops on the Graphical View and the border around the steps.

1.11.2.10 Shadow size

If this size is larger than 0, a shadow of the steps, hops and notes is drawn on the canvas, making it look like the transformation floats above the canvas.

1.11.2.11 Dialog middle percentage

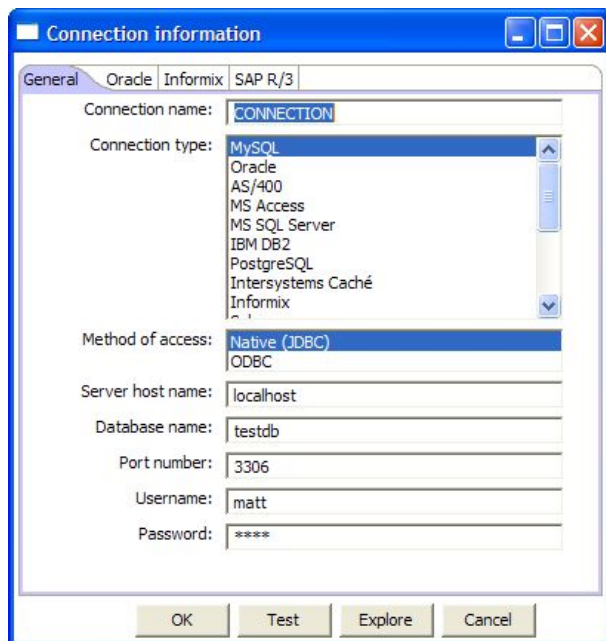
By default, a parameter is drawn at 35% of the width of the dialog, counted from the left. You can change this with this parameter. Perhaps this can be useful in cases where you use unusually large fonts.

2 DATABASE CONNECTIONS

2.1 Description

A connection describes the method by which Kettle can connect to a database. The top entries in the tree on the left describe the available connections.

2.2 Screenshot



2.3 Options

- ✓ **Connection name:** a connection name uniquely name defines a connection across transformations.
- ✓ **Connection type:** the type of database you're connecting to.
- ✓ **Method of access:** This can be either Native (JDBC), ODBC, or OCI.
- ✓ **Server host name:** specify the host name of the server on which the database resides. You can also specify it's IP-address.
- ✓ **Database name:** identifies the database name you want to connect to. In case of ODBC specify the DSN name here. (see also §2.4)
- ✓ **Port number:** sets the TCP/IP port number on which the database listens.
- ✓ **Username/password:** optionally specifies the username and password to connect to the database.

EXTRA:

- For Oracle you can specify the default tablespaces in which Kettle will places objects generating SQL for tables and indexes.
- For Informix, you need to specify the Informix Server name in the Informix tab in order for a connection to be useable.
- For SAP R/3 connections, extra parameters Language, System Number and SAP Client can be specified in the SAP R/3 tab.

User manual Last updated: 11/09/2005	Kettle ETL Environment	<i>i-Bridge BVBA</i>
	Spoon Graphical Transformation Designer	

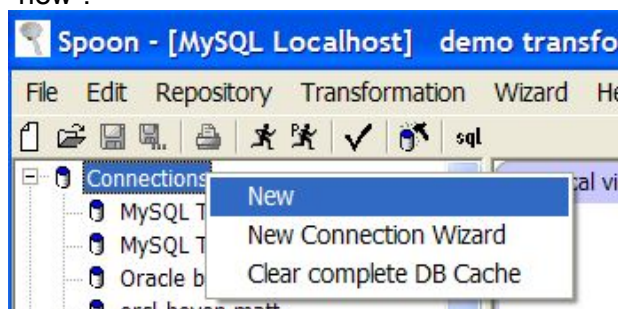
2.4 Database Usage Grid

Database	Access Method	Server name or address	Database name	Port nr (default)	Username & password
Oracle	Native	Required	Oracle database SID	Required (1521)	Required
	ODBC		ODBC DSN name		Required
	OCI		Database TNS name		Required
MySQL	Native	Required	MySQL database name	Required (3306)	Required
	ODBC		ODBC DSN name		Required
AS/400	Native	Required	AS/400 Library name	Optional	Required
	ODBC		ODBC DSN name		Required
MS Access	ODBC		ODBC DSN name		Optional
MS SQL Server	Native	Required	Database name	Required (1433)	Required
	ODBC		ODBC DSN name		Required
IBM DB2	Native	Required	Database name	Required (50000)	Required
	ODBC		ODBC DSN name		Required
PostgreSQL	Native	Required	Database name	Required (5432)	Required
	ODBC		ODBC DSN name		Required
Intersystems Caché	Native	Required	Database name	Required (1972)	Required
	ODBC		ODBC DSN name		Required
Informix	Native	Required	Database name	Required (1526)	Required
	ODBC		ODBC DSN name		Required
Sybase	Native	Required	Database name	Required (5001)	Required
	ODBC		ODBC DSN name		Required
Gupta SQL Base	Native	Required	Database name	Required (2155)	Required
	ODBC		ODBC DSN name		Required
Dbase III, IV or 5.0	ODBC		ODBC DSN name		Optional
Firebird SQL	Native	Required	Database name	Required (3050)	Required
	ODBC		ODBC DSN name		Required
Hypersonic	Native	Required	Database name	Required (9001)	Required
SAP DB	Native	Required	Database name		Required
	ODBC		ODBC DSN name		Required
Generic	ODBC		ODBC DSN name		Optional

2.5 Usage

2.5.1 Create a new connection

You can create a new connection by clicking right on the "Connections" tree entry and selecting "new".

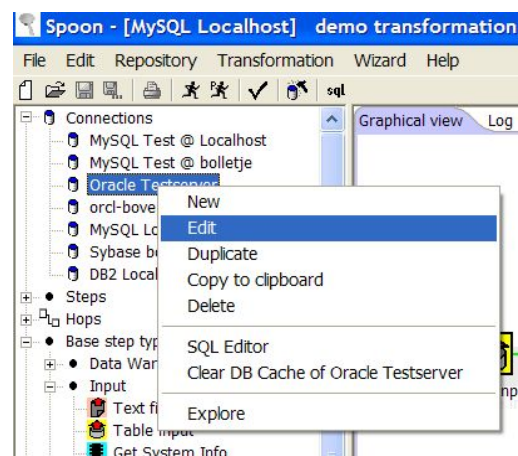


Or : simply double click on the Connections tree entry.

Or : press F3 to start the "new connection" wizard.

2.5.2 Edit a connection

Double click on the connection name in the left tree. Or right click on the name and select "Edit connection".



2.5.3 Duplicate a connection

Right click on the connection name and select "Duplicate".

2.5.4 Copy to clipboard

Copies the XML describing the connection to the clipboard.

2.5.5 Delete a connection

Right click on the connection name and select "Delete".

2.5.6 Test a connection

In the edit window (see above), select the "Test" button. If Spoon can connect, an OK message is displayed after a short delay.

User manual Last updated: 11/09/2005	Kettle ETL Environment	<i>i-Bridge BVBA</i>
	Spoon Graphical Transformation Designer	

2.5.7 Execute SQL commands on a connection

Right click on the connection name and select "SQL Editor". See also §3 for more information.

2.5.8 Clear DB Cache option

To speed up connections Spoon uses a database cache. Use this option when the information in the cache no longer represents the layout of the database. This is the case when databases tables have been changed, created or deleted.

2.5.9 Explore

This option will start the database explorer for the selected database connection.
Please see §4 for more information.

2.6 Unsupported databases

If you want to access a database type that is not yet supported, let us know and we will try to find a solution. A few database types are not supported in this release because of the lack of sample database and/or software.

Please note that it is usually still possible to read from these databases by using the Generic database driver through an ODBC connection.

3 SQL EDITOR

3.1 Description

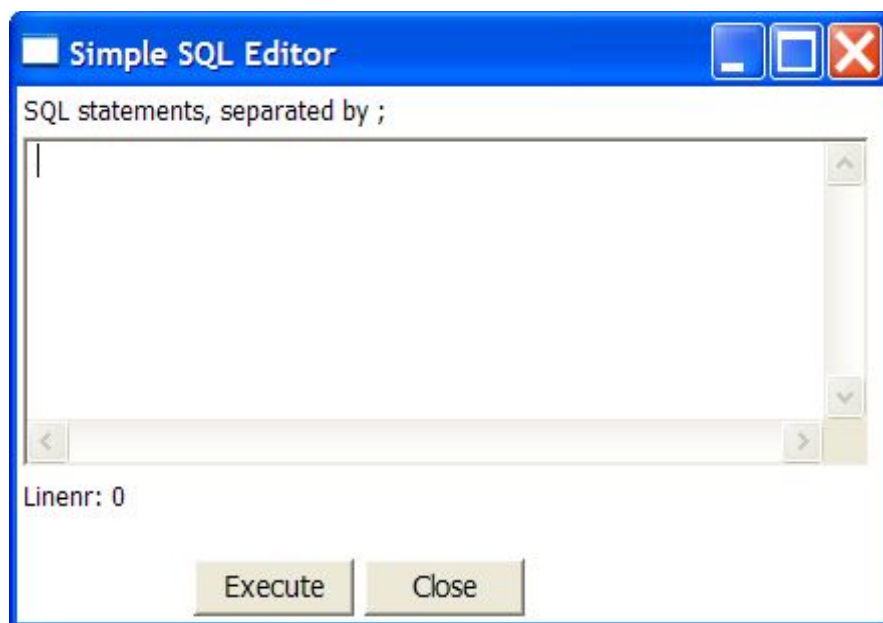
Sometimes a simple SQL Editor can be nice to have. Especially when you're creating tables, dropping indexes and modifying fields. The simple SQL editor supplied in Spoon, allows you to do this. In fact, most of the DDL (Data Definition Language) such as "create/alter table", "create index" and "create sequence" SQL commands are created automatically for you via the SQL Editor window.

NOTE: Multiple SQL Statements have to be separated by semi-colons (;).

NOTE: Before these SQL Statements are sent to the database to be executed, Spoon removes returns, line-feeds and the separating semi-colons.

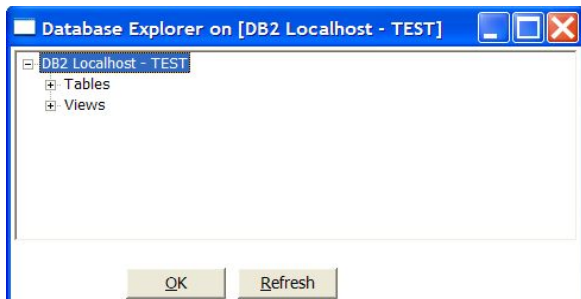
NOTE: Kettle clears the database cache for the database connection on which you launch DDL statements.

3.2 Screenshot



4 DATABASE EXPLORER

4.1 Screenshot



4.2 Description

The database explorer allows you to explore the database to which the database connection points. At the moment, it only shows available tables and the catalog and/or schema to which the table belongs.

It is possible to click right on a shown table or view (lowest level in the tree) and select one of the following options:

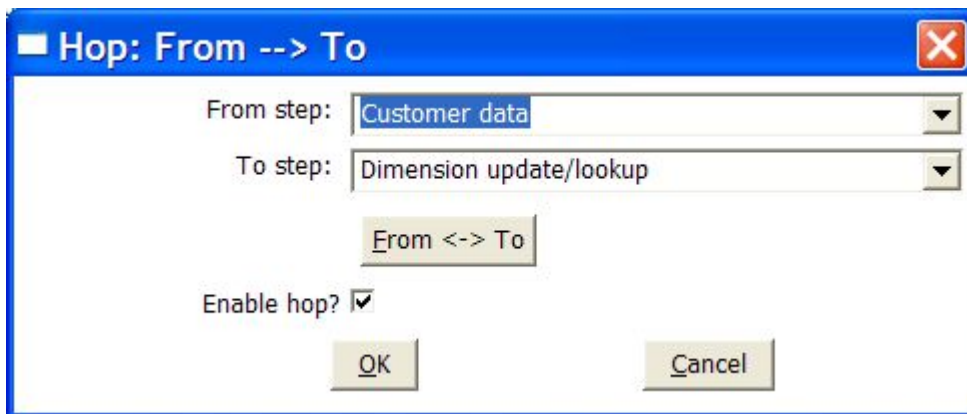
- ✓ Display the first 100 rows of the table (also available through double-click on table name)
- ✓ Display the first ... lines of the table
- ✓ Show the size (in rows) of the table.
- ✓ Show layout of the table
- ✓ Generate the DDL statement (create table ...) for this table.
- ✓ Generate the DDL statement (create table ...) for this table on another database connection
- ✓ Show the SQL statement to read from this table. (in SQL Editor)

5 HOPS

5.1 Description

A hop connects one step with another. The direction of the data flow is indicated with an arrow on the graphical view pane. A hop can be enabled or disabled. (For testing purposes for example).

5.2 Screenshot



5.3 Creating A Hop

You can easily create a new hop between 2 steps by one of the following options:

- ✓ Dragging on the Graphical View between 2 steps while using the middle mouse button.
- ✓ Dragging on the Graphical View between 2 steps while pressing the SHIFT key and using the left mouse button.
- ✓ Selecting 2 steps in the tree, clicking right and selecting "new hop"
- ✓ Selecting 2 steps in the graphical view (CTRL + left mouse click), clicking right on a step and selecting "new hop"

5.4 Splitting A Hop

You can easily insert a new step into a new hop between 2 steps by dragging the step (in the Graphical View) over a hop until the hop becomes drawn in bold. Release the left button and you will be asked if you want to split the hop. This works only with steps that have not yet been connected to another step.

5.5 Loops

Loops are not allowed in transformations because Spoon depends heavily on the previous steps to determine the field values that are passed from one step to another. If we would allow loops in transformations we often would get endless loops and undetermined results.

6 TRANSFORMATION SETTINGS

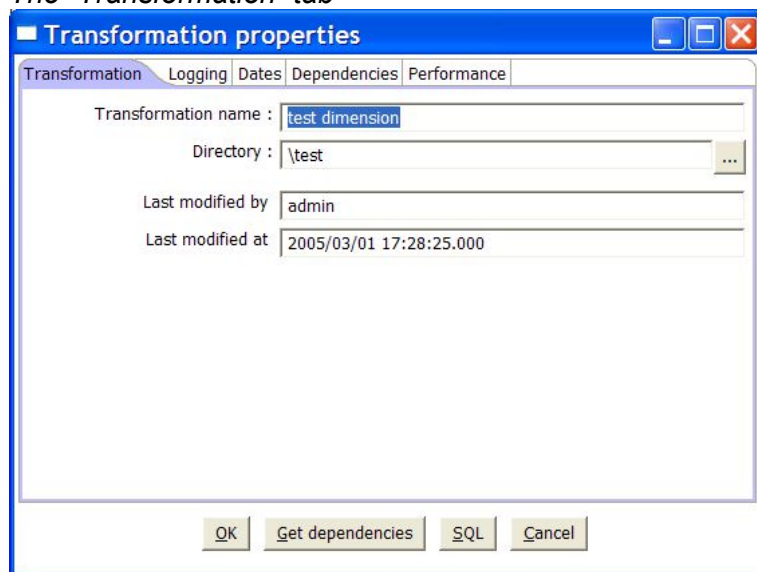
6.1 Description

There are a few options that control how a transformation is behaving and how it is logging what it is doing.

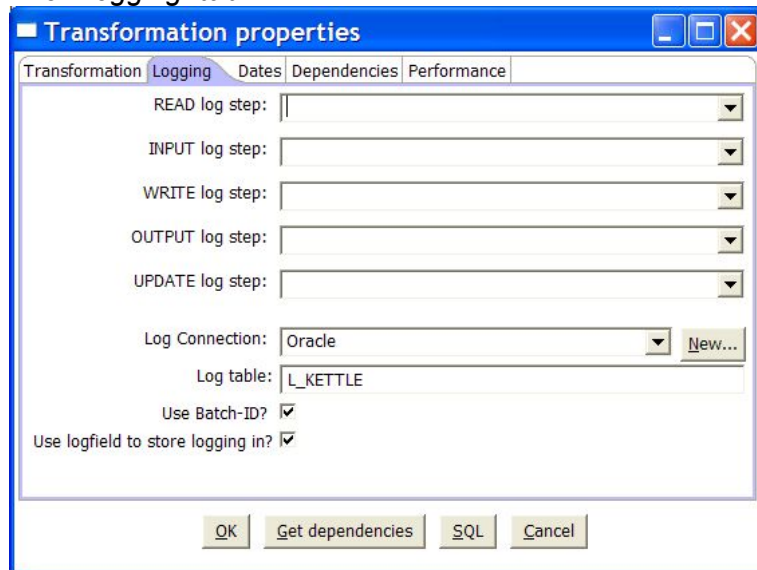
A number of settings also define how a transformation acquires data from source tables. To be more precise, the settings in this dialog define the date range for the source table.

6.2 Screenshots

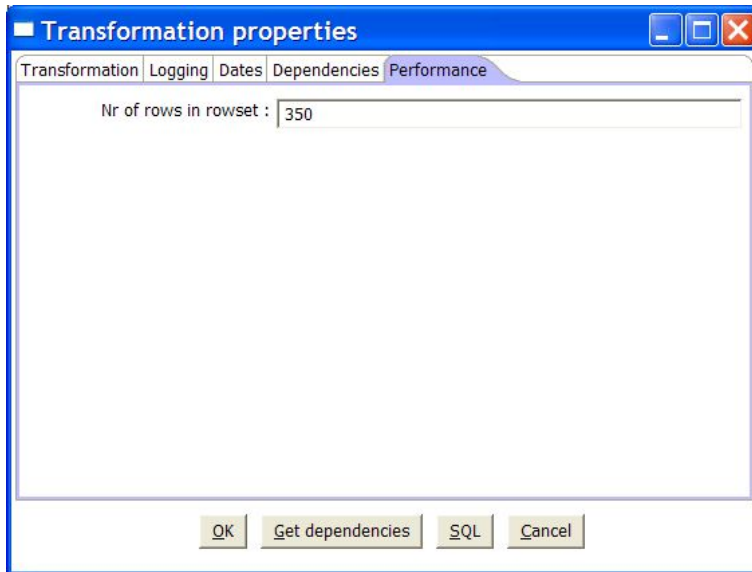
The “Transformation” tab



The “Logging” tab



The “Performance” tab.



6.3 Options

<i>Option</i>	<i>Description</i>
Transformation name	the name of the transformation. Required information if you want to save to a repository.
Directory	the directory in the repository in which you want to save the transformation.
READ log step	use the number of read lines from this step to write to the log table. Read means: read from source steps.
INPUT log step	use the number of input lines from this step to write to the log table. Input means: input from file or database.
WRITE log step	use the number of written lines from this step to write to the log table. Written means: written to target steps.
OUTPUT log step	use the number of output lines from this step to write to the log table. Output means: output to file or database.
UPDATE log step	use the number of updated lines from this step to write to the log table. Update means: updated in a database.
Log connection	Use this connection to write to a log table
Log table	specifies the name of the log table (for example L_ETL)
Use Batch-ID?	enable this if you want to have a batch ID in the L_ETL file. Disable for backward compatibility with Spoon/Pan version < 2.0.
Use logfield to store logging in	this option stores the logging text in a CLOB field in the logging table. This allows you to have the logging text together with the run results in the same table. Disable for backward compatibility with Spoon/Pan version < 2.1
Maxdate connection	Get the upper limit for a date range on this connection.
Maxdate table	get the upper limit for a date range in this table.
Maxdate field	get the upper limit for a date range in this field.

User manual Last updated: 11/09/2005	Kettle ETTL Environment	<i>i-Bridge BVBA</i>
	Spoon Graphical Transformation Designer	

<i>Option</i>	<i>Description</i>
Maxdate offset	increases the upper date limit with this amount. Use this for example, If you find that the field DATE_LAST_UPD has a maximum value of 2004-05-29 23:00:00, but you know that the values for the last minute are not complete. In this case, simply set the offset to -60.
Maximum date difference	Sets the maximum date difference in the obtained date range. This will allow you to limit job sizes.
Dependencies	This table will allow you to enter all the dependencies. For example if a dimension is depending on 3 lookup tables, we have to make sure that these lookup tables have not changed. If the values in these lookup tables have changed, we need to extend the date range to force a full refresh of the dimension. The dependencies allow you to look up whether a table has changed in case you have a “data last changed” column in the table.
Number of rows in rowsets	this option allows you to change the size of the buffers between the connected steps in a transformation. You will rarely/never need to change this parameter. Only when you run low on memory it might be an option to lower this parameter.

6.4 Extra

- ✓ Get dependencies button: Tries to automatically detect dependencies.
- ✓ SQL button: generates the SQL needed to create the logging table and allows you to execute this SQL statement.

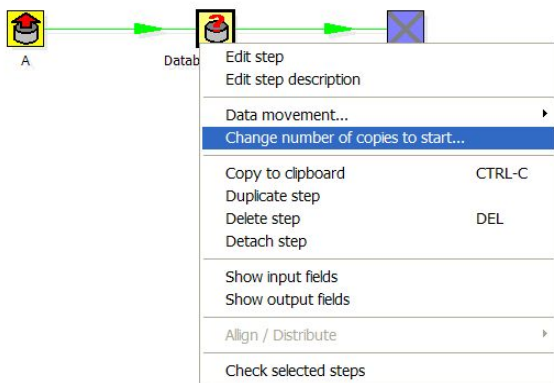
7 STEPS

7.1 Description

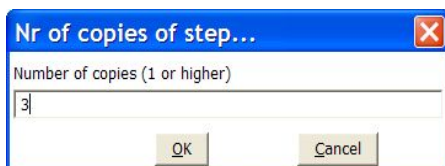
A step is one part of a transformation. Steps can provide you with a wide range of functionality ranging from reading text-files to implementing slowly changing dimensions. Please see below for a complete list of all available step-types.

7.2 Launching several copies of a step

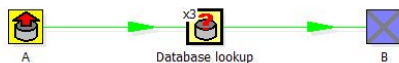
Sometimes it can be useful to launch the same step several times. For example, for performance reasons it can be useful to launch a database lookup step 3 times or more. That is because database connections usually have a certain latency. Launching the same step several times keeps the database busy on different connections, effectively lowering the latency. You can launch several copies of step in a transformation simply by clicking right on a step in the graphical view and then by selecting "change number of copies to start..."



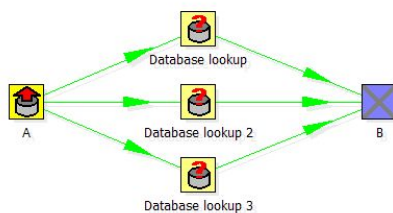
You will get this dialog:



If you enter 3 this will be shown:



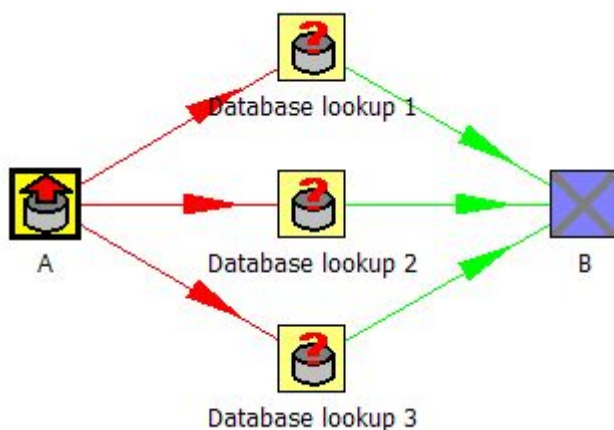
It is technically the equivalent of this:



7.3 Distribute or copy?

In the example above green lines are shown between the steps. This indicates that rows are distributed among the target steps. In this case it means that the first row coming from step “A” goes to step “database lookup 1”, the second to “database lookup 2”, the third to “Database lookup 3”, the fourth back to “database lookup 1”, etc.

However, if we click right on step “A”, and select “Copy data”, you will get the hops drawn in red:



“Copy data” means that all rows from step “A” are copied to all 3 the target steps. In this case it means that step “B” gets 3 copies of all the rows that “A” has sent out.

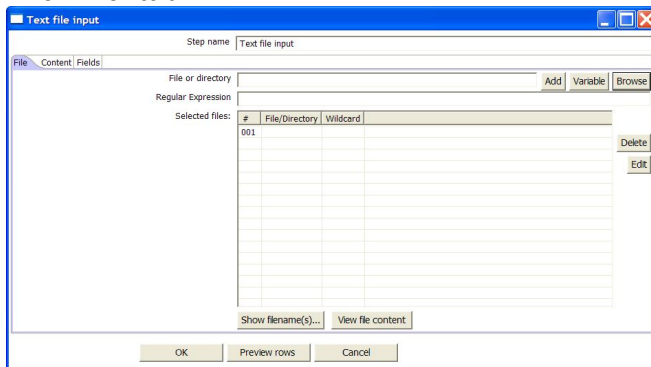
NOTE: *Because of the fact that all these steps are run as different threads, the order in which the single rows arrive at step “B” is probably not going to be the same as they left step “A”.*

7.4 Step Types

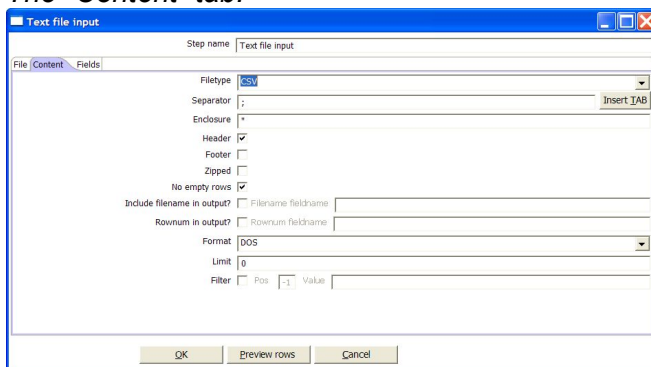
7.4.1 Text File Input

7.4.1.1 Screenshots

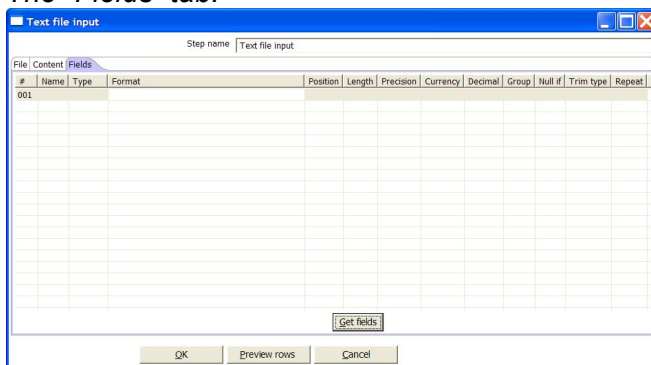
The “File” tab:



The “Content” tab:



The “Fields” tab:



7.4.1.2 Icon



User manual Last updated: 11/09/2005	Kettle ETL Environment	<i>i-Bridge BVBA</i>
	Spoon Graphical Transformation Designer	

7.4.1.3 General description

You can use the *Text File Input* step to read a large number of different text-files.

Most commonly these are Comma Separated Values (CSV files) generated by spreadsheets and the like.

From version 2.1 on you can specify a list of files to read, or a list of directories with wildcards in the form of regular expressions.

7.4.1.4 Options

Option	Description
Step name	Name of the step. This name has to be unique in a single transformation.
File or directory	This field specifies the location and/or name of the input text file. NOTE: press the “add” button to add the file/directory/wildcard combination to the list of selected files (grid) below.
Regular expression	Specify the regular expression you want to use to select the files in the directory specified in the previous option.
File type	This can be either CSV or Fixed length. Based on this selection, Spoon will launch a different helper GUI when you press the “get fields” button in the last “fields” tab.
Separator	One or more characters that separate the fields in a single line of text. Typically this is ; or a tab
Enclosure	Some fields can be enclosed by a pair of strings to allow separator characters in fields. The enclosure string is optional.
Header	enable this option if your text file has a header row. (First line in the file)
Footer	enable this option if your text file has a footer row. (Last line in the file)
Zipped	enable this option if your text file is placed in a Zip archive. NOTE: <i>At the moment, only the first file in the archive is read.</i>
No empty rows	Don't send empty rows to the next steps
Include filename in output	Enable this if you want the filename to be part of the output
Filename fieldname	The name of the field that will contain the filename
Row number in output	Enable this if you want the row number to be part of the output
Row number fieldname	The name of the field that will contain the row number
Format	This can be either DOS or UNIX. UNIX files have lines are separated by linefeeds. DOS files have lines separated by carriage returns <i>and</i> line feeds.
Limit	Sets the number of lines that is read from the file. 0 means: read all lines.
Filter	Enable this option if you want to filter rows based on a certain string being present on a certain location. (0 is the first position)

User manual Last updated: 11/09/2005	Kettle ETTL Environment	<i>i-Bridge BVBA</i>
	Spoon Graphical Transformation Designer	

<i>Option</i>	<i>Description</i>
Fields grid	<ul style="list-style-type: none"> ✓ Name: name of the field ✓ Type: Type of the field can be either String, Date or Number ✓ Format: See 7.4.1.5 for a complete description of format symbols. ✓ Length: <ul style="list-style-type: none"> ○ Number: Total number of significant figures in a number ○ String: total length of string ○ Date: length of <i>printed</i> output of the string (e.g. 4 only gives back the year) ✓ Precision: <ul style="list-style-type: none"> ○ Number: Number of floating point digits. ○ String: unused. ○ Date: unused. ✓ Currency: used to interpret numbers like \$10,000.00 or €5.000,00 ✓ Decimal: A decimal point can be a "." (10;000.00) or "," (5.000,00) ✓ Grouping: A grouping can be a dot "," (10;000.00) or "." (5.000,00) ✓ Null if: treat this value as NULL. ✓ Trim type: trim this field (left, right, both) before processing. ✓ Repeat: Y/N: If the corresponding value in this row is empty: repeat the one from the last time it was not empty.

User manual Last updated: 11/09/2005	Kettle ETL Environment	<i>i-Bridge BVBA</i>
	Spoon Graphical Transformation Designer	

7.4.1.5 Formats

7.4.1.5.1 Number formats

Symbol	Location	Localized	Meaning
0	Number	Yes	Digit
#	Number	Yes	Digit, zero shows as absent
.	Number	Yes	Decimal separator or monetary decimal separator
-	Number	Yes	Minus sign
,	Number	Yes	Grouping separator
E	Number	Yes	Separates mantissa and exponent in scientific notation. Need not be quoted in prefix or suffix.
;	Sub pattern boundary	Yes	Separates positive and negative sub patterns
%	Prefix or suffix	Yes	Multiply by 100 and show as percentage
\u2030	Prefix or suffix	Yes	Multiply by 1000 and show as per mille
¤ (\u00A4)	Prefix or suffix	No	Currency sign, replaced by currency symbol. If doubled, replaced by international currency symbol. If present in a pattern, the monetary decimal separator is used instead of the decimal separator.
'	Prefix or suffix	No	Used to quote special characters in a prefix or suffix, for example, "'#'" formats 123 to "#123". To create a single quote itself, use two in a row: "' o'clock".

Scientific Notation

In a pattern, the exponent character immediately followed by one or more digit characters indicates scientific notation. Example: "0.###E0" formats the number 1234 as "1.234E3".

Date formats

Letter	Date or Time Component	Presentation	Examples
G	Era designator	Text	AD
y	Year	Year	1996; 96
M	Month in year	Month	July; Jul; 07
w	Week in year	Number	27
W	Week in month	Number	2
D	Day in year	Number	189
d	Day in month	Number	10
F	Day of week in month	Number	2
E	Day in week	Text	Tuesday; Tue
a	Am/pm marker	Text	PM
H	Hour in day (0-23)	Number	0
k	Hour in day (1-24)	Number	24
K	Hour in am/pm (0-11)	Number	0
h	Hour in am/pm (1-12)	Number	12
m	Minute in hour	Number	30
s	Second in minute	Number	55
S	Millisecond	Number	978
z	Time zone	General time zone	Pacific Standard Time; PST; GMT-08:00
Z	Time zone	RFC 822 time zone	-0800

User manual Last updated: 11/09/2005	Kettle ETL Environment	<i>i-Bridge BVBA</i>
	Spoon Graphical Transformation Designer	

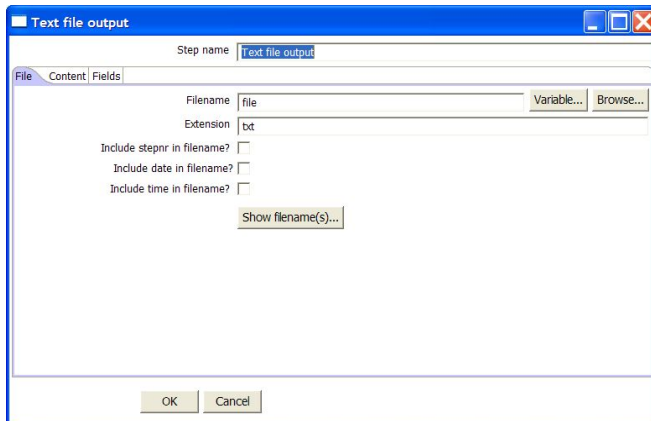
7.4.1.6 Extras

<i>Function/Button</i>	<i>Description</i>
Show filenames	This option shows a list of all the files selected. Please note that if the transformation is to be run on a separate server, the result might be incorrect.
View file content	The “View” button shows the first lines of the text-file. Make sure that the file-format is correct. When in doubt, try both DOS and UNIX formats.
Get fields	This button allows you to guess the layout of the file. In case of a CSV file, this is done pretty much automatically. When you selected a file with fixed length fields, you need to specify the field boundaries using a wizard.
Preview rows	Press this button to preview the rows generated by this step.

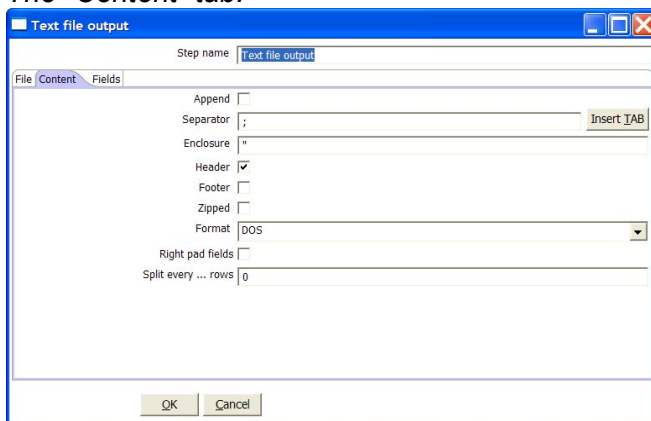
7.4.2 Text File Output

7.4.2.1 Screenshots

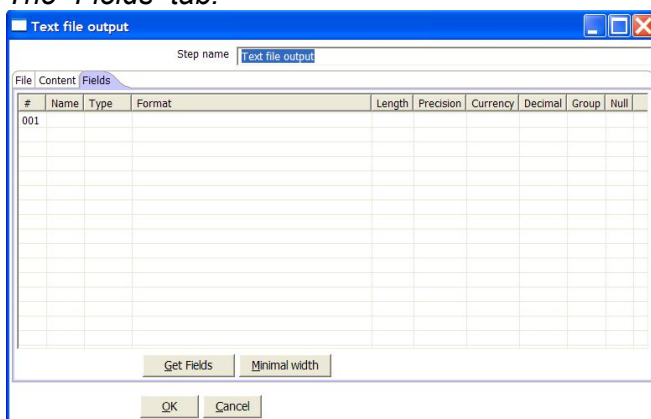
The “File” tab:



The “Content” tab:



The “Fields” tab:



User manual Last updated: 11/09/2005	Kettle ETL Environment	<i>i-Bridge BVBA</i>
	Spoon Graphical Transformation Designer	

7.4.2.2 Icon



7.4.2.3 General Description

You can use the *Text File Output* step to export data to text file format.

Most commonly these are Comma Separated Values (CSV files) that can be read by spreadsheets and the like.

7.4.2.4 Options

Option	Description
Step name	Name of the step. This name has to be unique in a single transformation.
Filename	This field specifies the filename and location of the output text file
Extension	adds a point and the extension to the end of the filename. (.txt)
Include stepnr in filename	If you run the step in multiple copies (see §7.2), the copy number is included in the filename, before the extension. (_0)
Include date in filename	includes the system date in the filename. (_20041231)
Include time in filename	includes the system date in the filename. (_235959)
Append	check this if you want to append lines to the end of the specified file.
Separator	Here you can specify the character that separates the fields in a single line of text. Typically this is ; or a tab
Enclosure	A pair of strings can enclose some fields. This allows separator characters in fields. The enclosure string is optional.
Header	enable this option if you want the text file to have a header row. (First line in the file)
Footer	enable this option if you want the text file to have a footer row. (Last line in the file)
Zipped	Check this box if you want the text file(s) to be zipped. NOTE: <i>At the moment, only one file is placed in a single archive.</i>
Format	This can be either DOS or UNIX. UNIX files have lines separated by linefeeds. DOS files have lines separated by carriage returns <i>and</i> line feeds.
Right pad fields	add spaces to the end of the fields (or remove characters at the end) until they have the specified length.
Split every ... rows	If this number N is larger than zero, split the resulting text-file into multiple parts of N rows.

User manual Last updated: 11/09/2005	Kettle ETL Environment	<i>i-Bridge BVBA</i>
	Spoon Graphical Transformation Designer	

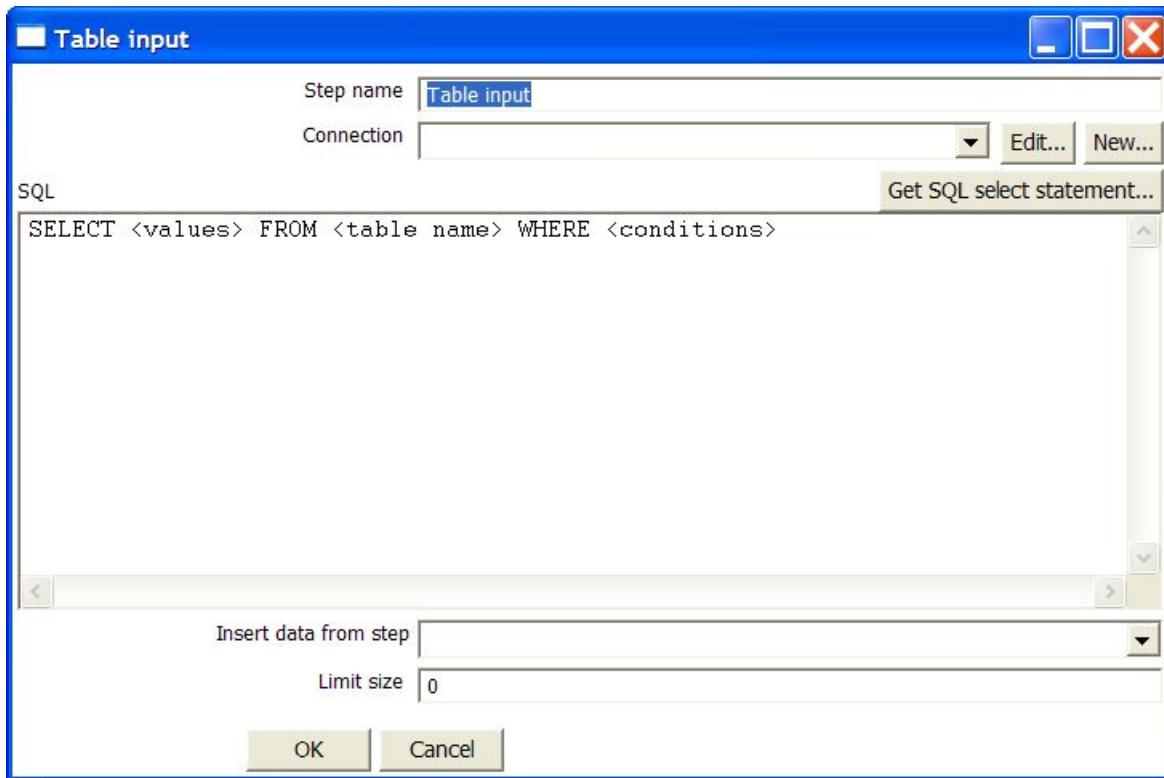
<i>Option</i>	<i>Description</i>
Fields	<ul style="list-style-type: none"> ✓ Name: name of the field ✓ Type: Type of the field can be either String, Date or Number ✓ Format: See 7.4.1.5 for a complete description of format specifiers. ✓ Length: <ul style="list-style-type: none"> ○ Number: Total number of significant figures in a number ○ String: total length of string ○ Date: length of <i>printed</i> output of the string (e.g. 4 only gives back the year) ✓ Precision: <ul style="list-style-type: none"> ○ Number: Number of floating point digits. ○ String: unused. ○ Date: unused. ✓ Currency: Symbol used to represent currencies like \$10,000.00 or €5.000,00 ✓ Decimal: A decimal point can be a "." (10,000.00) or "," (5.000,00) ✓ Grouping: A grouping can be a "," (10,000.00) or "." (5.000,00) ✓ Null: If the value of the field is null, insert this string into the text-file.

7.4.2.5 Extras

<i>Function/Button</i>	<i>Description</i>
Show filenames	This option shows a list of the files the will be generated. NOTE: <i>This is a simulation and sometimes depends on the number of rows in each file, etc.</i>
Minimal width	Alter the options in the fields tab in such a way that the resulting width of lines in the text file is minimal. So instead of save 0000001, we write 1, etc.

7.4.3 Table input

7.4.3.1 Screenshot



7.4.3.2 Icon



7.4.3.3 General description

This step is used to read information from a database, using a connection and SQL. Basic SQL statements are generated automatically.

7.4.3.4 Options

<i>Option</i>	<i>Description</i>
Step name	Name of the step. This name has to be unique in a single transformation.
Connection	The database connection used to read data from.
SQL	The SQL statement used to read information from the database connection.
Insert data from step	Specify the input step name where we can expect information to come from. This information can then be inserted into the SQL statement. The locators where we insert information is indicated by ? (question marks).
Limit	Sets the number of lines that is read from the database. 0 means: read all lines.

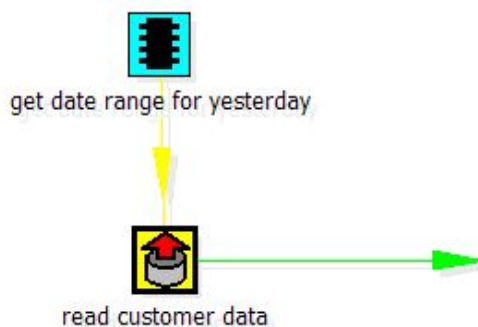
7.4.3.5 Example

Consider for example the following SQL statement:

```
SELECT * FROM customers WHERE changed_date BETWEEN ? AND ?
```

This statement needs 2 dates that are read on the "Insert data from" step.

NOTE: The dates can be provided using the "Get System Info" step type. For example if you want to read all customers that have had their data changed yesterday, you might do it like this:



The "read customer data" step looks like this:

Table input

Step name:

Connection:

SQL:

Insert data from step:

Limit size:

And the “get date range for yesterday” looks like this:

Get System Data

Step name :

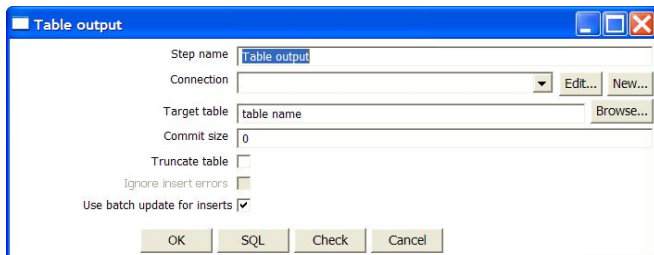
Fields:

#	Name	Type
001	start_range	Yesterday 00:00:00
002	end_range	Yesterday 23:59:59

See also: *Get System Data step §7.4.16.*

7.4.4 Table output

7.4.4.1 Screen dump



7.4.4.2 Icon



7.4.4.3 General description

This step type allows you to store information in a database table.

7.4.4.4 Options

Option	Description
Step name	Name of the step. This name has to be unique in a single transformation.
Connection	The database connection used to write data to
Target table	the name of the table to write data to
Commit size	use transactions to insert rows in the database table. Commit the connection every N rows if N is larger than 0. Otherwise, don't use transactions. (Slower) NOTE: <i>Transactions are not supported on all database platforms.</i>
Truncate table	Select this if you want the table to be truncated <i>before</i> the first row is inserted into the table
Ignore insert errors	Makes Kettle ignore all insert errors such as violated primary keys. A maximum of 20 warnings will be logged however. This option is not available for batch inserts.
Use batch update for inserts	Enable this option if you want to use batch inserts. This feature groups inserts statements to limit round trips to the database. This is the fastest option and is enabled by default.

7.4.4.5 Extras

Function/Button	Description
SQL	Generate the SQL to create the output table automatically.
Check	Verifies that all fields are available in the target table and vice-versa

7.4.5 Select values

7.4.5.1 Screen dump

The “Select & Alter” tab:

The screenshot shows the 'Select / Rename values' dialog box with the 'Select & Alter' tab selected. The 'Step name' is 'Select values'. The 'Fields' table has columns: #, Fieldname, Rename to, Length, Precision. A 'Get fields to select' button is on the right. 'OK' and 'Cancel' buttons are at the bottom.

#	Fieldname	Rename to	Length	Precision
001				

The “Remove” tab:

The screenshot shows the 'Select / Rename values' dialog box with the 'Remove' tab selected. The 'Fields to remove' table has columns: #, Fieldname. A 'Get fields to remove' button is on the right. 'OK' and 'Cancel' buttons are at the bottom.

#	Fieldname
001	

The “Meta-data” tab:

The screenshot shows the 'Select / Rename values' dialog box with the 'Meta-data' tab selected. The 'Fields to alter the meta-data for' table has columns: #, Fieldname, Rename to, Type, Length, Precision. A 'Get fields to change' button is on the right. 'OK' and 'Cancel' buttons are at the bottom.

#	Fieldname	Rename to	Type	Length	Precision
001					

7.4.5.2 Icon



User manual Last updated: 11/09/2005	Kettle ETL Environment	<i>i-Bridge BVBA</i>
	Spoon Graphical Transformation Designer	

7.4.5.3 General description

This step type is useful to

- ✓ Select fields
- ✓ Rename fields
- ✓ Specify the length and/or precision of the fields

These are the functions of the 3 different tabs:

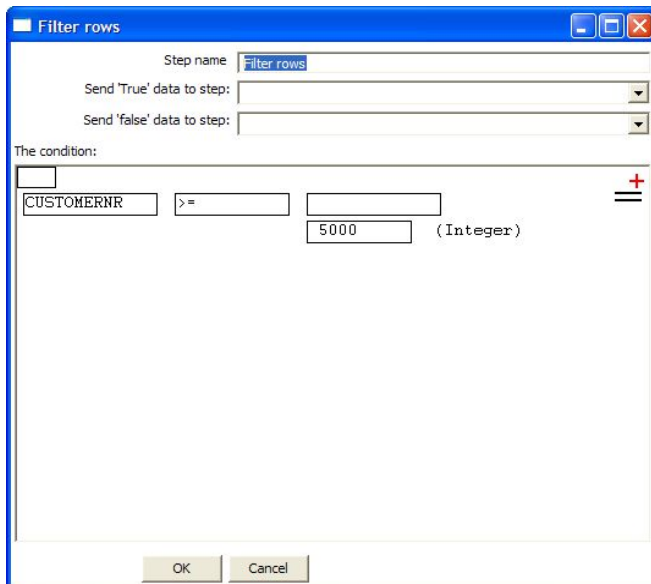
- Select & Alter: Specify the exact order and name in which the fields have to be placed in the output rows.
- Remove: Specify the fields that have to be removed from the output rows
- Meta-data: Change the name, type, length and precision (the meta-data) of one or more fields.

7.4.5.4 Options

- ✓ Step name: Name of the step. This name has to be unique in a single transformation.
- ✓ Fields of the table:
 - Fieldname: The fieldname to select or change
 - Rename to: leave blank if you don't want to rename
 - Length: enter number to specify the length. (-1: no length specified)
 - Precision: enter number to specify the precision. (-1: no precision specified)

7.4.6 Filter rows

7.4.6.1 Screenshot



7.4.6.2 Icon



7.4.6.3 General description

This step type allows you to filter rows based upon conditions and comparisons.

Once this step is connected to a previous step (one or more and receiving input), you can simply click on the “<field>”, “=” and “<value>” areas to construct a condition.

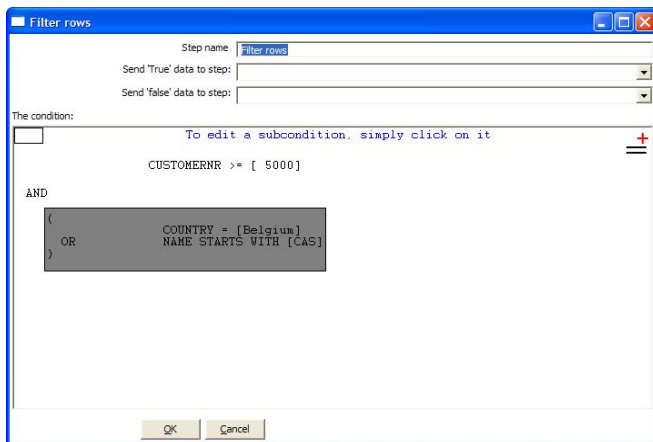
In the example in the screenshot, it is possible to click on the following icon to add more conditions:



It will convert the original condition to a subcondition and add one more.

A subcondition can be edited simply by clicking on it. (going down one level into the condition tree)

For example, this is a more complex example:



7.4.6.4 Options

- ✓ Step name: Name of the step. This name has to be unique in a single transformation.
- ✓ Send “true” data to step: The rows for which the condition specified evaluates to true are send to this step.
- ✓ Send “false” data to step: The rows for which the condition specified evaluates to false are send to this step.

7.4.7 Database lookup

7.4.7.1 Screen dump

7.4.7.2 Icon



7.4.7.3 General description

This step type allows you to look up values in a database table.

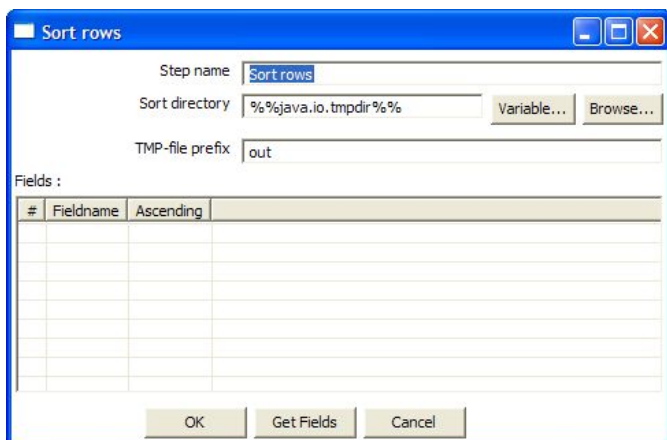
7.4.7.4 Options

- ✓ Step name: Name of the step. This name has to be unique in a single transformation.
- ✓ Connection: The database connection used to write data to.
- ✓ Lookup table: the name of the table where we do the lookup.
- ✓ Enable cache: This option caches database lookups. This means that we expect the database to return the same value all the time for a certain lookup value.

IMPORTANT: *if other processes are changing values in the table where you do the lookup, it might be unwise to cache values. However, in all other cases, enabling this option can seriously increase the performance because database lookups are generally speaking very slow. If you find that you can't use the cache, consider launching several copies of this step at the same time. This will keep the database busy via different connections. To see how to do this, please see §7.2*

7.4.8 Sort rows

7.4.8.1 Screenshot



7.4.8.2 Icon



7.4.8.3 General description

This step type sorts rows based upon the fields you specify and whether or not they should be sorted ascending or descending.

NOTE: Kettle has to sort rows using temporary files when the number of rows exceeds 5000.

7.4.8.4 Options

- ✓ Step name: Name of the step. This name has to be unique in a single transformation.
- ✓ Sort directory: This is the directory in which the temporary files are stored in case it is needed. The default is the standard temporary directory for the system.
- ✓ The TMP-file prefix: Choose a recognizable prefix in order to recognize the files when they show up in the temp directory.
- ✓ A list of fields and whether they should be sorted ascending or not.

7.4.8.5 Extras

- ✓ The “Get fields” button inserts the fields into the “Fields” grid if the step is connected to the previous ones.

7.4.9 Stream lookup

7.4.9.1 Screenshot

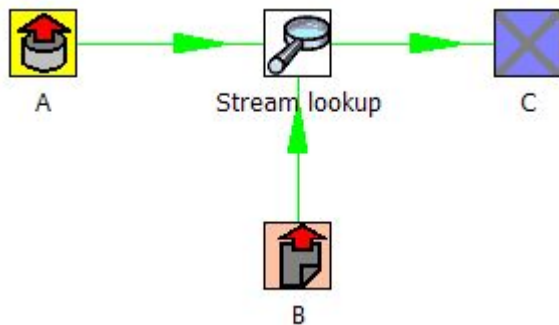
7.4.9.2 Icon



7.4.9.3 General description

This step type allows you to look up data using information coming from other steps in the transformation. The data coming from the “Source step” is first read into memory and is then used to look up data from the main stream.

For example, this transformation add information coming from a text-file to data coming from a database table:



The fact that we use information from B to do the lookups is indicated by the option: “Source step” (see below):

Source step

7.4.9.4 Options

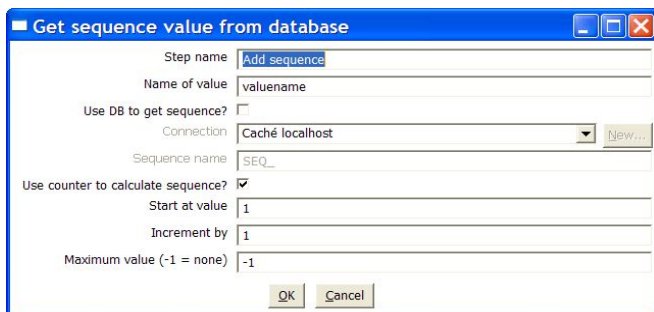
- ✓ Step name: Name of the step. This name has to be unique in a single transformation.
- ✓ Source step: This is the step name where the lookup data is coming from
- ✓ The key(s) to lookup value(s): Allows you to specify the names of the fields that are used to lookup values. Values are always searched using the “equal” comparison.
- ✓ Fields to retrieve: You can specify the names of the fields to retrieve here, as well as the default value in case the value was not found or a new fieldname in case you didn’t like the old one.

7.4.9.5 Extra

- ✓ “Get Fields”: This will automatically fill in the names of all the available fields on the source side (A). You can then delete all the fields you don’t want to use for lookup.
- ✓ “Get lookup fields”: This will automatically insert the names of all the available fields on the lookup side (B). You can then delete the fields you don’t want to retrieve.

7.4.10 Add sequence

7.4.10.1 Screen dump



7.4.10.2 Icon

123

7.4.10.3 General description

Steps based on this step type will add a sequence to the stream. A sequence is an ever-changing integer value with a certain start and increment value.

You can either use a database (Oracle) sequence to determine the value of the sequence, or let Kettle decide.

NOTE: *Kettle sequences are only unique when used in the same transformation. Also, they are not stored, so the values start back at the same value every time the transformation is launched.*

7.4.10.4 Options

- ✓ Step name: Name of the step. This name has to be unique in a single transformation.
- ✓ Name of value: name of the new sequence value that is added to the stream.
- ✓ Use DB to get sequence: Enable this option if you want the sequence to be driven by a database sequence.
 - Connection name: choose the name of the connection on which the database sequence resides.
 - Sequence name: allows you to enter the name of the database sequence.
- ✓ Use counter to calculate sequence: Enable this option if you want the sequence to be generated by Kettle.
 - Start at: give the start value of the sequence.
 - Increment by : give the increment of the sequence.
 - Maximum value: this is the maximum value after which the sequence will start back at the start value. (Start At)

User manual Last updated: 11/09/2005	Kettle ETL Environment	<i>i-Bridge BVBA</i>
	Spoon Graphical Transformation Designer	

Examples:

```

Start at = 1, increment by = 1, maximum value = 3
    → This will produce: 1, 2, 3, 1, 2, 3, 1, 2, 3, ...

Start at = 0, increment by = -1, maximum value = -5
    → This will produce: 0, -1, -2, -3, -4, -5, 0, -1, -2, ...

```


The fields tab:

Step name: update CUSTOMER dimension

Table | Keys | Fields

Lookup/Update fields

#	Dimension field	Stream field to compare with	Type of dimension update
001	NAME	NAME	Insert
002	FIRSTNAME	FIRSTNAME	Insert
003	LANGUAGE	LANGUAGE	Insert
004	GENDER	GENDER	Insert
005	STREET	STREET	Insert
006	HOUSENR	HOUSENR	Insert
007	BUSNR	BUSNR	Insert
008	ZIPCODE	ZIPCODE	Insert
009	CITY	CITY	Insert
010	COUNTRY	COUNTRY	Insert
011	DATE_OF_BIRTH	DATE_OF_BIRTH	Insert

Get Fields

OK SQL Check fields Cancel

7.4.11.2 Icon



7.4.11.3 General description

This step type allows you to implement Ralph Kimball's slowly changing dimension for both types: Type I (insert) and Type II (update). Not only can you use this dimension for updating a dimension table, it can also be used for looking up values in a dimension.

In our dimension implementation each entry in the dimension table has the following fields:

1. Technical key: this is the primary key of the dimension
2. Version field: shows the version of the dimension entry. (a revision number)
3. Start of date range: this is the fieldname containing the validity starting date.
4. End of date range: this is the fieldname containing the validity ending date.
5. Keys: these are the keys used in your source systems. For example: customer numbers, product id, etc.
6. Fields: these fields contain the actual information of a dimension.

As a result of the lookup or update operation of this step type, a field is added to the stream containing the technical key of the dimension. In case the field is not found, the value of the dimension entry for not found (0 or 1) is returned.

NOTE: This dimension entry is added automatically to the dimension table when the update is first run.

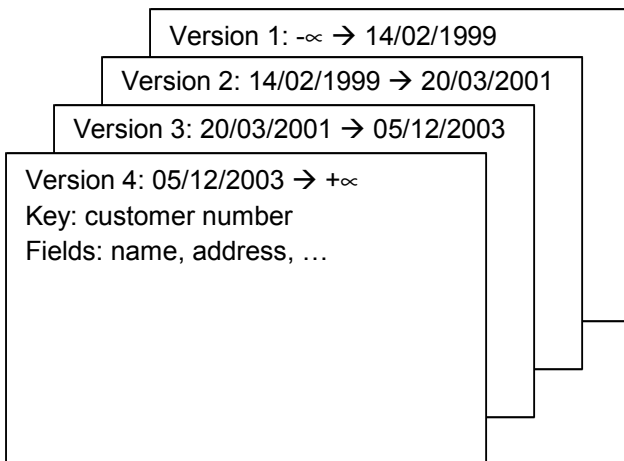
User manual Last updated: 11/09/2005	Kettle ETL Environment	<i>i-Bridge BVBA</i>
	Spoon Graphical Transformation Designer	

7.4.11.4 Options

- ✓ Step name: Name of the step. This name has to be unique in a single transformation.
- ✓ Connection: name of the database connection on which the dimension table resides.
- ✓ Target table: name of the dimension table
- ✓ Commit size: setting this to 10 will generate a commit every 10 inserts or updates.
- ✓ Technical Key field: this indicates the primary key of the dimension. It is also referred to as Surrogate Key.
- ✓ Technical key, new name: rename the technical key after a lookup, for example, if you need to lookup different types of products like ORIGINAL_PRODUCT_TK, REPLACEMENT_PRODUCT_TK, ...
- ✓ Use auto-increment field: Some databases support auto-increment, auto-number or identity fields that determine automatically the next value for a primary key. Enable this option if the field in the table you use is auto-incremented.
NOTE: *This field is not enabled if the database doesn't support auto increment fields.*
- ✓ Optional sequence: if the database you use supports sequences, you can fill in the name of the sequence to use:
NOTE: *This field is not enabled if the database doesn't support sequences.*
- ✓ Version field: specifies the name of the field to store the version (revision number) in.
- ✓ Stream date field: If you have the date at which the dimension entry was last changed, you can specify the name of that field here. It allows the dimension entry to be accurately described for what the date range concerns. If you don't have such a date, the system date will be taken.
NOTE: *Consider adding an extra date field from System Info if you don't want the date ranges to be different all the time. For example if you have extracts from a source system being done every night at midnight, consider adding date "Yesterday 23:59:59" as a field to the stream by using a Join step.*
- ✓ Date range start and end fields: specify the names of the dimension entries validity range.
- ✓ Keys: Specify the names of the keys in the stream and in the dimension table. This will enable the step to do the lookup.
- ✓ Update dimension: Check this option if you want to update the dimension based on the information in the input stream. If this option is not enabled, the dimension only does lookups and only adds the technical key field to the streams.
- ✓ Fields: For each of the fields you need to have in the dimension, you can specify whether you want the values to be updated (for all versions, this is a Type II operation) or you want to have the values inserted into the dimension as a new version. In the example we used in the screenshot the birth date is something that's not variable in time, so if the birth date changes, it means that it was wrong in previous versions. It's only logical then, that the previous values are corrected in all versions of the dimension entry.

7.4.11.5 Extra

- ✓ Get fields: fills in all the available fields on the input stream, except for the keys you specified.
- ✓ Check fields: Checks whether everything is filled in correctly
- ✓ SQL: generates the SQL to build the dimension and allows you to execute this SQL.



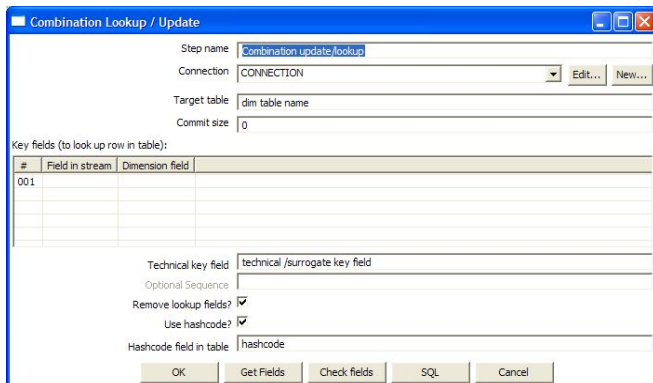
Each dimension entry can be represented by a stack of papers containing the information valid during a certain period of time.

Insert then means that we add a new piece of paper containing the new information. (Type II)

Punch through means that we overwrite certain data on all pieces of paper for that certain customer number. (Type I)

7.4.12 Combination update/lookup

7.4.12.1 Screenshot



7.4.12.2 Icon



7.4.12.3 General description

The generally accepted theory of data warehousing and multi-dimensional modeling suggests that only keys and facts can be present in a fact table. This can present you with a problem if you have information that doesn't have a primary key. This step type then comes to the rescue by allowing you to store information in a junk-dimension table.

For example if you have an address of a customer, but you don't have a customer number, you can use this step type.

Kettle will store the information in a table where the primary key is the combination of all fields in the table. Because this process can be very slow in case you have a large number of fields, Kettle also supports a "hash code" field representing all fields in the dimension.

This can speed up lookup performance dramatically while limiting the fields to index to 1.

7.4.12.4 Options

- ✓ Step name: Name of the step. This name has to be unique in a single transformation.
- ✓ Connection: name of the database connection on which the dimension table resides.
- ✓ Target table: name of the dimension table
- ✓ Commit size: setting this to 10 will generate a commit every 10 inserts or updates.
- ✓ Key fields: Specify all the fields you want to store in the dimension.
- ✓ Technical Key field: this indicates the primary key of the dimension. It is also referred to as Surrogate Key.
- ✓ Optional sequence: Specify the sequence name if you want to use a database sequence on the table connection to generate the technical key.
- ✓ Remove lookup fields: Enable this option if you want to remove all the lookup fields from the input stream in the output. The only extra fields added is then the technical key.
- ✓ Use hash code: This option allows you to generate a hash code, representing all values in the key fields in a numerical form (a signed 64 bit integer). This hash code has to be stored in the table.

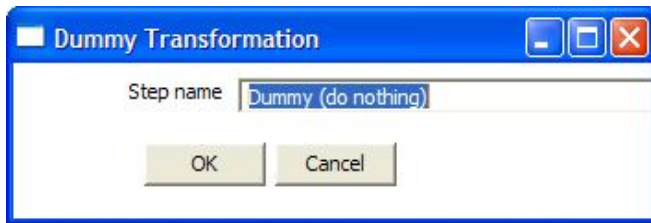
6.4.11.5 Extra

User manual Last updated: 11/09/2005	Kettle ETL Environment	<i>i-Bridge BVBA</i>
	Spoon Graphical Transformation Designer	

- ✓ Get fields: fills in all the available fields on the input stream, except for the keys you specified.
- ✓ Check fields: Checks whether everything is filled in correctly
- ✓ SQL: generates the SQL to build the dimension and allows you to execute this SQL.

7.4.13 Dummy (do nothing)

7.4.13.1 Screenshot



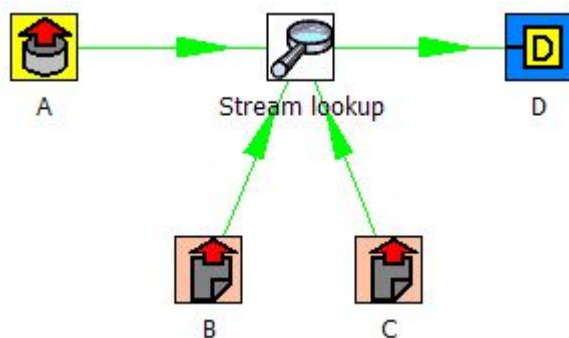
7.4.13.2 Icon



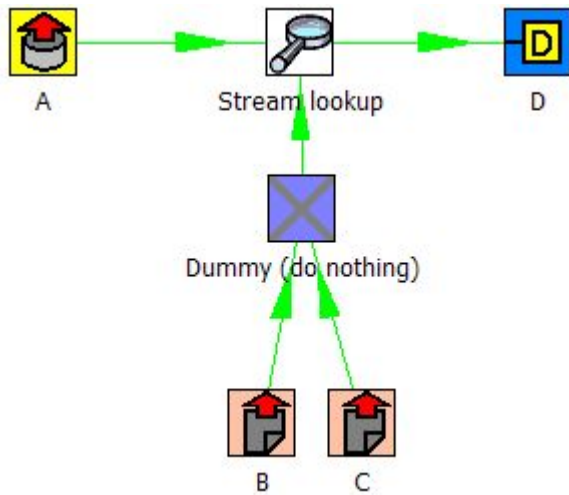
7.4.13.3 General description

This step doesn't do anything. It's main function is perform as a placeholder in case you want to test something. For example, to have a transformation, you need at least 2 steps connected to each other. If you want to test for example a test file input step, you can connect it to a dummy step.

The placeholder function also comes into play in the following case:



Unfortunately, the “Stream Lookup” step can only read lookup information from one stream, so we need to redo the transformation like this:

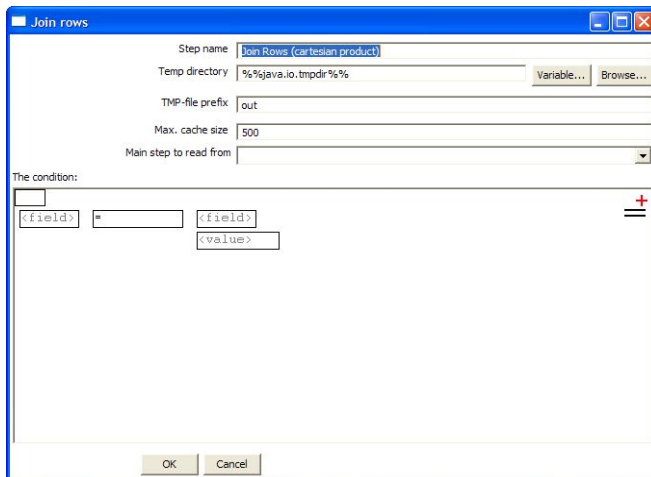


7.4.13.4 Options

- ✓ Step name: Name of the step. This name has to be unique in a single transformation.

7.4.14 Join Rows (Cartesian product)

7.4.14.1 Screenshot

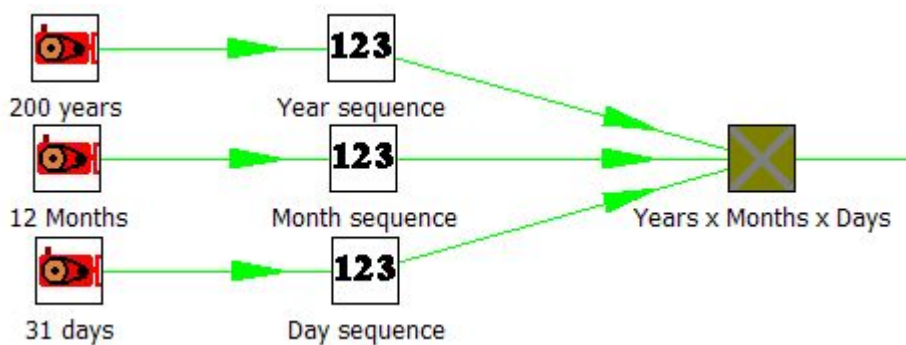


7.4.14.2 Icon



7.4.14.3 General description

This step allows you to produce combinations (Cartesian product) of all rows on the input streams. This is an example:



The “Years x Months x Days” step outputs all combinations of Year, Month and Day. (1900, 1, 1 → 2100, 12, 31) and can be used to create a date dimension.

7.4.14.4 Options

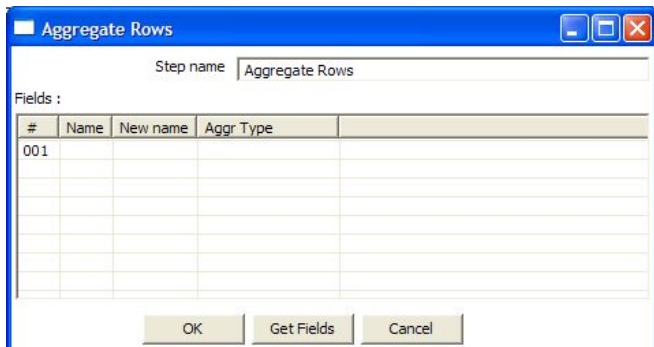
- ✓ Step name: Name of the step. This name has to be unique in a single transformation.
- ✓ Temp directory: specify the name of the directory where the system stores temporary files in case you want to combine more then the cached number of rows.
- ✓ Temporary file prefix: this is the prefix of the temporary files that will be generated.

User manual Last updated: 11/09/2005	Kettle ETL Environment	<i>i-Bridge BVBA</i>
	Spoon Graphical Transformation Designer	

- ✓ Max. cache size: the number of rows to cache before the systems reads data from temporary files. This is needed in case you want to combine large row sets that don't fit into memory.
- ✓ Main step to read from: Specifies the step to read the most data from. This step is not cached or spooled to disk, the others are.
- ✓ The condition: you can enter a complex condition to limit the number of output rows.
See also: §7.4.6

7.4.15 Aggregate Rows

7.4.15.1 Screenshot



7.4.15.2 Icon



7.4.15.3 General description

This step type allows you to quickly aggregate rows based upon all the rows. These are the available aggregation types:

SUM:	the sum of a field
AVERAGE:	the average of a field
COUNT:	the number of (not null) values
MIN:	the minimum value of a field
MAX:	the maximum value of a field
FIRST:	the first value of a field
LAST:	the last value of a field

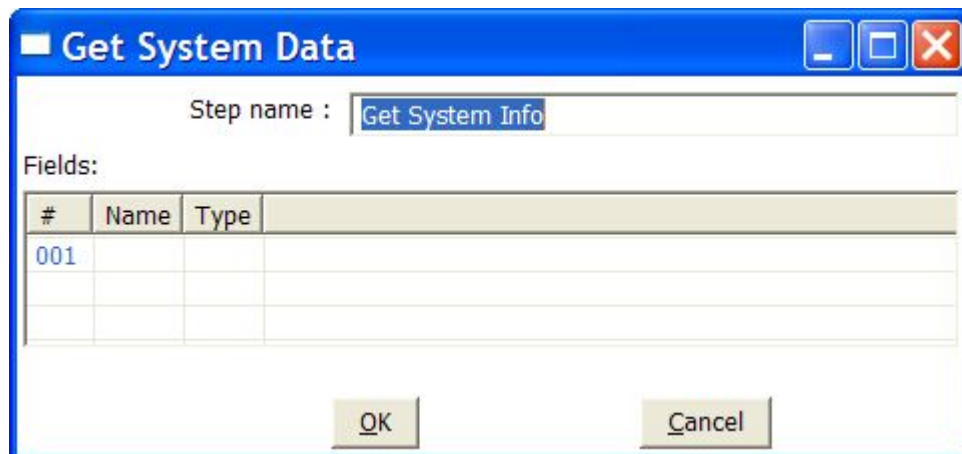
NOTE: This step type is deprecated. See the “Group By” step in §7.4.25 for a more powerfull way of aggregating rows of data.

7.4.15.4 Options

- ✓ Step name: Name of the step. This name has to be unique in a single transformation.
- ✓ Fields: the fields and the aggregation type

7.4.16 Get System Info

7.4.16.1 Screen dump



7.4.16.2 Icon



7.4.16.3 General description

This step type retrieves information from the Kettle environment.
These are the possible pieces of information you can retrieve:

<i>Option</i>	<i>Description</i>
system date (variable)	System time, changes every time you ask a date.
system date (fixed)	System time, determined at the start of the transformation
start date range	Start of date range, based upon information in ETL log table
end date range	End of date range, based upon information in ETL log table
Yesterday 00:00:00	start of yesterday
Yesterday 23:59:59	end of yesterday
Today 00:00:00	start of today
Today 23:59:59	end of today
Tomorrow 00:00:00	start of tomorrow
Tomorrow 23:59:59	end of tomorrow
First day of last month 00:00:00	start of last month
Last day of last month 23:59:59	end of last month
First day of this month 00:00:00	start of this month
Last day of this month 23:59:59	end of this month
First day of next month 00:00:00	start of next month
Last day of next month 23:59:59	end of next month
copy of step	Copy nr of the step (see also: §7.2)
transformation name	Name of the transformation

User manual Last updated: 11/09/2005	Kettle ETTL Environment	<i>i-Bridge BVBA</i>
	Spoon Graphical Transformation Designer	

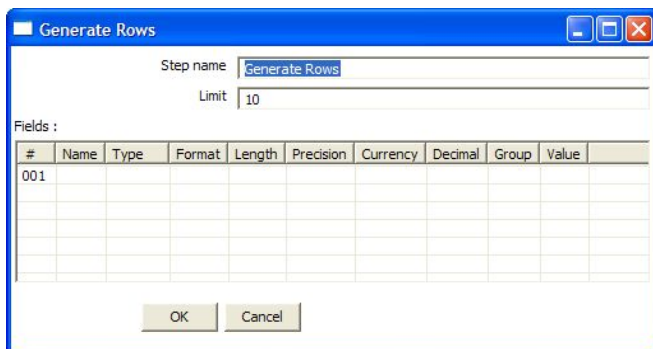
<i>Option</i>	<i>Description</i>
transformation file name	File name of the transformation (XML only)
User that modified the transformation last	
Date when the transformation was modified last	
transformation batch ID	ID_BATCH value in the logging table, see §6
Hostname	returns the hostname of the server
IP address	returns the IP address of the server
command line argument 1	Argument 1 on the command line
command line argument 2	Argument 2 on the command line
command line argument 3	Argument 3 on the command line
command line argument 4	Argument 4 on the command line
command line argument 5	Argument 5 on the command line
command line argument 6	Argument 6 on the command line
command line argument 7	Argument 7 on the command line
command line argument 8	Argument 8 on the command line
command line argument 9	Argument 9 on the command line
command line argument 10	Argument 10 on the command line

7.4.16.4 Options

- ✓ Step name: Name of the step. This name has to be unique in a single transformation.
- ✓ Fields: the fields to output.

7.4.17 Generate Rows

7.4.17.1 Screen dump



7.4.17.2 Icon



7.4.17.3 General description

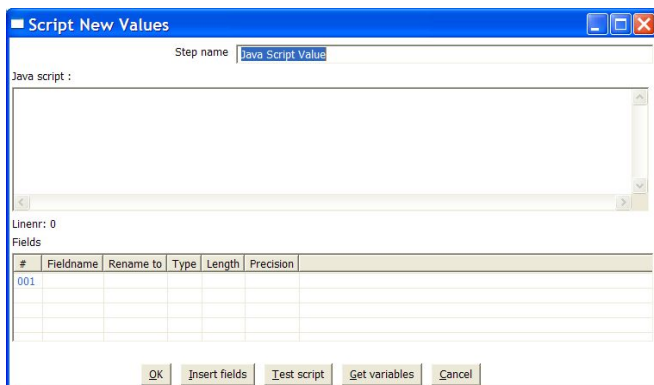
This step type outputs a number of rows, empty but optionally a number of static fields.

7.4.17.4 Options

- ✓ Step name: Name of the step. This name has to be unique in a single transformation.
- ✓ Limit: The number of rows you want to output.
- ✓ Fields: static fields you might want to include in the output row.

7.4.18 Java Script Value

7.4.18.1 Screenshot



7.4.18.2 Icon



7.4.18.3 General description

This step type allows you to do complex calculations using the JavaScript language.

7.4.18.4 Options

- ✓ Step name: Name of the step. This name has to be unique in a single transformation.
- ✓ Java Script: the script in question.
- ✓ Fields: these are the fields that we want to add to the output stream.

7.4.18.5 Extra

- ✓ Insert fields: inserts the fields and the standard method to grab the value of the field.
- ✓ Test script: tests whether or not the script compiles.
- ✓ Get variables: gets the newly created variables and inserts them into the “Fields” grid.

7.4.18.6 Value functions

This is a list of functions that you can use to manipulate values...

- ✓ Value Clone()
Builds a copy of a value and returns a Value.
- ✓ void setName(String name)
Sets the name of a value
- ✓ String getName()
Get the name of a value
- ✓ void setValue(double num)
Set the value to a floating point value
- ✓ void setValue(String str)
Set the value to a floating point value

User manual Last updated: 11/09/2005	Kettle ETL Environment	<i>i-Bridge BVBA</i>
	Spoon Graphical Transformation Designer	

- ✓ void setValue(Date dat)
Set the value to a Date value
- ✓ void setValue(boolean bool)
Set the value to a Boolean value
- ✓ void setValue(long l)
Set the value to an integer value
- ✓ void setValue(Value v)
Set the value to the value contained in another field
- ✓ double getNumber()
Gets the value of a field as a floating point value
- ✓ String getString()
Gets the value of a field as a textual string representation
- ✓ int getStringLength()
Gets the length of the string representation
- ✓ Date getDate()
Gets the value of the field as a date value

User manual Last updated: 11/09/2005	Kettle ETL Environment	<i>i-Bridge BVBA</i>
	Spoon Graphical Transformation Designer	

- ✓ **boolean getBoolean()**
Gets the value of a field as a Boolean.
NOTE: *String “Y” or “true” is converted to true.*
NOTE: *Numeric value 0 is converted to false, everything else is true.*
- ✓ **long getInteger()**
Gets the value of a field as an integer.
NOTE: *Date fields are converted to the number of milliseconds since January 1st 1970 00:00:00 GMT.*
- ✓ **boolean isEmpty()**
If the value has no type, this function returns true.
- ✓ **boolean isString()**
If the value is of type String, this function returns true.
- ✓ **boolean isDate()**
If the value is of type String, this function returns true.
- ✓ **boolean isNumber()**
If the value is of type Number, this function returns true.
- ✓ **boolean isBoolean()**
If the value is of type Boolean, this function returns true.
- ✓ **boolean isInteger()**
If the value is of type Integer, this function returns true.
- ✓ **boolean isNumeric()**
If the value is of type Number or Integer, this function returns true.
- ✓ **String toString()**
Returns the textual representation of the value.
- ✓ **String toString(boolean pad)**
Returns the textual representation of the value, padded to the length of the string if pad = true.
- ✓ **String toStringMeta()**
Returns the meta-data information of the value as a string.
- ✓ **void setLength(int l)**
Sets the length of the value
- ✓ **void setLength(int l, int p)**
Sets the length and precision of the value
- ✓ **int getLength()**
Returns the length of the value
- ✓ **int getPrecision()**
Returns the precision of the value
- ✓ **void setPrecision(int p)**
Sets the precision of a value

User manual Last updated: 11/09/2005	Kettle ETL Environment	<i>i-Bridge BVBA</i>
	Spoon Graphical Transformation Designer	

- ✓ **String getTypeDesc()**
Returns the description of a value as a string. (e.g. "String", "Number", "Date", "Integer", "Boolean")
- ✓ **void setNull()**
Sets the value to Null.
- ✓ **void clearNull()**
Removes the null setting.
- ✓ **void setNull(boolean n)**
- ✓ **boolean isNull()**
Returns true if the value is null.
- ✓ **int compare(Value v)**
Compares two values and returns 1 if the first value is larger than the second, -1 if it is smaller and 0 if they are equal.
- ✓ **boolean equals(Object v)**
Compares two values and returns true if the two values have the same value.
- ✓ **int hashCode()**
Returns a signed 64 values representing the value in the form of a hash code.
- ✓ **Value negate()**
If the value is numeric, multiplies the value by -1, in all other cases it doesn't do anything.
- ✓ **Value and(Value v)**
Calculates the bitwise AND of two integer values.
- ✓ **Value xor(Value v)**
Calculates the bitwise XOR of two integer values.
- ✓ **Value or(Value v)**
Calculates the bitwise OR of two integer values.
- ✓ **Value bool_and(Value v)**
Calculates the boolean AND of two boolean values.
- ✓ **Value bool_or(Value v)**
Calculates the boolean OR of two boolean values.
- ✓ **Value bool_xor(Value v)**
Calculates the boolean XOR of two boolean values.
- ✓ **Value bool_not()**
Calculates the boolean NOT of a boolean value.
- ✓ **Value greater_equal(Value v)**
Compares two values and sets the first to true if the second is greater or equal to the first.
- ✓ **Value smaller_equal(Value v)**
Compares two values and sets the first to true if the second is smaller or equal to the first.

User manual Last updated: 11/09/2005	Kettle ETL Environment	<i>i-Bridge BVBA</i>
	Spoon Graphical Transformation Designer	

- ✓ Value different(Value v)
Compares two values and sets the first to true if the second is different from the first.
- ✓ Value equal(Value v)
Compares two values and sets the first to true if the second is equal to the first.
- ✓ Value like(Value v)
Sets the first value to true if the second string is part of the first.
- ✓ Value greater(Value v)
Compares two values and sets the first to true if the second is greater than the first.
- ✓ Value smaller(Value v)
Compares two values and sets the first to true if the second is smaller than the first.
- ✓ Value minus(double v)
- ✓ Value minus(long v)
- ✓ Value minus(int v)
- ✓ Value minus(byte v)
- ✓ Value minus(Value v)
Subtracts v from the field value.
- ✓ Value plus(double v)
- ✓ Value plus(long v)
- ✓ Value plus(int v)
- ✓ Value plus(byte v)
- ✓ Value plus(Value v)
Adds v to the field value.
- ✓ Value divide(double v)
- ✓ Value divide(long v)
- ✓ Value divide(int v)
- ✓ Value divide(byte v)
- ✓ Value divide(Value v)
Divides the field value by v.
- ✓ Value multiply(double v)
- ✓ Value multiply(long v)
- ✓ Value multiply(int v)
- ✓ Value multiply(byte v)
- ✓ Value multiply(Value v)
Multiplies the field value by v.
NOTE: *Strings can be multiplied as well: the result is v times the string concatenated.*
- ✓ Value abs()
Sets the field value to the –field value if the value was negative.
- ✓ Value acos()
Sets the field value to the cosine of the number value.
- ✓ Value asin()
Sets the field value to the arc sine of the number value.

User manual Last updated: 11/09/2005	Kettle ETL Environment	<i>i-Bridge BVBA</i>
	Spoon Graphical Transformation Designer	

- ✓ Value atan()
Sets the field value to the arc tangents of the number value.
- ✓ Value atan2(Value arg0)
- ✓ Value atan2(double arg0)
Sets the field value to the second arc tangents of the number value.
- ✓ Value ceil()
Sets the field value to the ceiling of a number value.
- ✓ Value cos()
Sets the field value to the cosine of a number value.
- ✓ Value cosh()
Sets the field value to the hyperbolic cosine of a number value.
- ✓ Value exp()
Sets the field value to the exp of a number value.
- ✓ Value floor()
Sets the field value to the floor of a number value.
- ✓ Value initcap()
Sets the all first characters of words in a string to uppercase.
"matt casters" → "Matt Casters"
- ✓ Value length()
Sets the value of the field to the length of the String value.
- ✓ Value log()
Sets the field value to the log of a number value.
- ✓ Value lower()
Sets the field value to the string value in lowercase.
- ✓ Value lpad(Value len)
- ✓ Value lpad(Value len, Value padstr)
- ✓ Value lpad(int len)
- ✓ Value lpad(int len, String padstr)
Sets the field value to the string value, left padded to a certain length.
Default the padding string is a single space.
Optionally, you can specify your own padding string.
- ✓ Value ltrim()
Sets the field value to the string, without spaces to the left of the string.
- ✓ Value mod(Value arg)
- ✓ Value mod(double arg0)
Sets the value to the modulus of the first and the second number.
- ✓ Value nvl(Value alt)
If the field value is Null, set the value to alt
- ✓ Value power(Value arg)
- ✓ Value power(double arg0)

User manual Last updated: 11/09/2005	Kettle ETL Environment	<i>i-Bridge BVBA</i>
	Spoon Graphical Transformation Designer	

Raises the field value to the power arg

- ✓ Value replace(Value repl, Value with)
- ✓ Value replace(String repl, String with)
Replaces a string in the field value with another.
- ✓ Value round()
Rounds the field value to the nearest integer.
- ✓ Value rpad(Value len)
- ✓ Value rpad(Value len, Value padstr)
- ✓ Value rpad(int len)
- ✓ Value rpad(int len, String padstr)
Sets the field value to the string value, right padded to a certain length.
Default the padding string is a single space.
Optionally, you can specify your own padding string.
- ✓ Value rtrim()
Remove the spaces to the right of the field value.
- ✓ Value sign()
Sets the value of the string to -1, 0 or 1 in case the field value is negative, zero or positive.
- ✓ Value sin()
Sets the value of the field to the sine of the number value.
- ✓ Value sqrt()
Sets the value of the field to the square root of the number value.
- ✓ Value substr(Value from, Value to)
- ✓ Value substr(Value from)
- ✓ Value substr(int from)
- ✓ Value substr(int from, int to)
Sets the value of the field to the substring of the string value.
- ✓ Value sysdate()
Sets the field value to the system date.
- ✓ Value tan(Value args[])
Sets the field value to the tangents of the number value

- ✓ Value num2str()
- ✓ Value num2str(String arg0)
- ✓ Value num2str(String arg0, String arg1)
- ✓ Value num2str(String arg0, String arg1, String arg2)
- ✓ Value num2str(String arg0, String arg1, String arg2, String arg3)

Converts a number to a string.

Arg0: format pattern, see also §7.4.1.5.1

Arg1: Decimal separator (either . or ,)

Arg2: Grouping separator (either . or ,)

Arg3: Currency symbol

For example converting value:

1234.56	using num2str("###,##0.00", "\", "\", ".")	gives	1.234,56
.23	using num2str("###,##0.00", "\", "\", ".")	gives	0,23
1234.56	using num2str("000,000.00", "\", "\", ".")	gives	001.234,56

- ✓ Value dat2str()
 - ✓ Value dat2str(String arg0)
 - ✓ Value dat2str(String arg0, String arg1)
- Converts a date into a string
- Arg0: format pattern, see also §7.4.1.5.1
- Arg1: localized date-time pattern characters (u, t)
- ✓ Value num2dat()
- Converts a number to a date based upon the number of milliseconds since January 1st, 1970 00:00:00 GMT.
- ✓ Value str2dat(String arg0)
 - ✓ Value str2dat(String arg0, String arg1)
- Converts a string to a date
- Arg0: format pattern, see also §7.4.1.5.1
- Arg1: localized date-time pattern characters (u, t)
- ✓ Value str2num()
 - ✓ Value str2num(String arg0)
 - ✓ Value str2num(String arg0, String arg1)
 - ✓ Value str2num(String arg0, String arg1, String arg2)
 - ✓ Value str2num(String arg0, String arg1, String arg2, String arg3)
- Converts a string into a number
- Arg0: format pattern, see also §7.4.1.5.1
- Arg1: Decimal separator (either . or ,)
- Arg2: Grouping separator (either . or ,)
- Arg3: Currency symbol
- ✓ Value dat2num()
- Converts a date into a number being the number of milliseconds since January 1st, 1970 00:00:00 GMT.
- ✓ Value trim()
- Remove spaces left and right of the string value.
- ✓ Value upper()
- Sets the field value to the uppercase string value.

User manual Last updated: 11/09/2005	Kettle ETL Environment	<i>i-Bridge BVBA</i>
	Spoon Graphical Transformation Designer	

- ✓ Value e()
Sets the value to e
- ✓ Value pi()
Sets the value to π
- ✓ Value add_months(int months)
Adds a number of months to the date value.
- ✓ Value last_day()
Sets the field value to the last day of the month of the date value.
- ✓ Value first_day()
Sets the field value to the first day of the month of the date value.
- ✓ Value trunc()
- ✓ Value trunc(double level)
- ✓ Value trunc(int level)
Set the field value to the truncated number or date value.
Level means the number of positions behind the comma or in the case of a date, 5=months, 4=days, 3=hours, 2=minutes, 1=seconds

7.4.18.7 JavaScript Examples

7.4.18.7.1 Remember the previous row:

Sometimes it can be useful to know the value of the previous row.

This can be accomplished by this piece of code:

```
var prev_row;
if (prev_row == null) prev_row = row;

...
String previousName = prev_row.getString("Name", "-");
...

prev_row = row;
```

row is a special field that contains all values in the current row.

7.4.18.7.2 Sets the location name of an address to uppercase

```
location.upper();
```

7.4.18.7.3 Extract information from a date field

```
// Year/Month/Day representation:
ymd = date_field.Clone().dat2str("yyyy/MM/dd").getString();

// Day/Month/Year representation:
dmy = date_field.Clone().dat2str("dd/MM/yyyy").getString();

// Year/Month :
ym = date_field.Clone().dat2str("yyyy/MM").getString();

// Long description of the month in the local language:
month_long_desc= date_field.Clone().dat2str("MMMM").initcap().getString();

// Week of the year (1-53)
week_of_year = date_field.Clone().dat2str("w").getInteger();

// day of week, short description (MON-SUN)
day_of_week_short_desc =date_field.Clone().dat2str("EEE").upper().getString();

// Day of the week (Monday-Sunday)
day_of_week_desc = date_field.Clone().dat2str("EEEE").initcap().getString();

// Day of week (1-7)
day_of_week = date_field.Clone().dat2str("F").getInteger();
```

See also: §7.4.1.5.1

NOTE: If you don't use Clone(), the original value will be overwritten by the methods that work on the Kettle Values.

7.4.19 Call DB Procedure

7.4.19.1 Screen dump

7.4.19.2 Icon



7.4.19.3 General description

This step type allows you to execute a database procedure (or function) and get the result(s) back.

7.4.19.4 Options

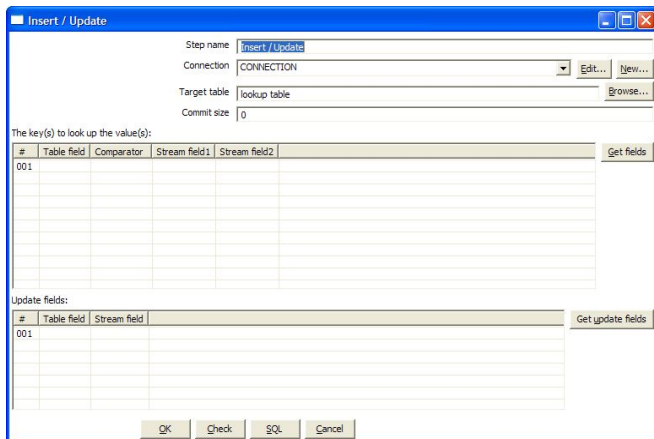
- ✓ Step name: Name of the step. This name has to be unique in a single transformation.
- ✓ Connection: name of the database connection on which the procedure resides.
- ✓ Proc-name: name of the procedure or function to call.
- ✓ Result name: name of the result of the function call, leave this empty in case of procedure.
- ✓ Result type: type of the result of the function call. Not used in case of a procedure.
- ✓ Parameters: List of parameters that the procedure or function needs
 - Field name
 - Direction: can be either IN (input only), OUT (output only), INOUT (value is changed on the database)
 - Type: used for output parameters so that Kettle knows what comes back.

7.4.19.5 Extra

- ✓ “Find it...” button: searches on the specified database connection for available procedures and functions. (at the moment only on Oracle and SQLServer)
- ✓ “Get fields” button: this function fills in all the fields in the input streams to make your life easier. Simply delete the lines you don’t need and re-order the ones you do need.

7.4.20 Insert / Update

7.4.20.1 Screen dump



7.4.20.2 Icon



7.4.20.3 General description

This step type first looks up a row in a table using one or more lookup keys. If the row can't be found, it inserts the row. If it can be found and the fields to update are the same, nothing is done. If they are not all the same, the row in the table is updated.

7.4.20.4 Options

- ✓ Step name: Name of the step. This name has to be unique in a single transformation.
- ✓ Connection: name of the database connection on which the target table resides.
- ✓ Target table: name of the table in which you want to do the insert or update
- ✓ Keys: here you can specify a list of field values and comparators.

You can use the following comparators:

- ☐ =
- ☐ <>
- ☐ <
- ☐ <=
- ☐ >
- ☐ >=
- ☐ LIKE
- ☐ BETWEEN
- ☐ IS NULL
- ☐ IS NOT NULL

7.4.20.5 Extra

- ✓ “Get fields” button: get the fields from the input stream(s) and fills them in into the keys grid.
- ✓ “Get update fields” button: gets the update fields from the input streams and fills them in into the update grid.
- ✓ “Check” button: checks whether or not all fields are available in the target table.
- ✓ “SQL” button: generates the SQL to create the table and indexes for correct operation.

7.4.21 Update

7.4.21.1 Screen dump

7.4.21.2 Icon

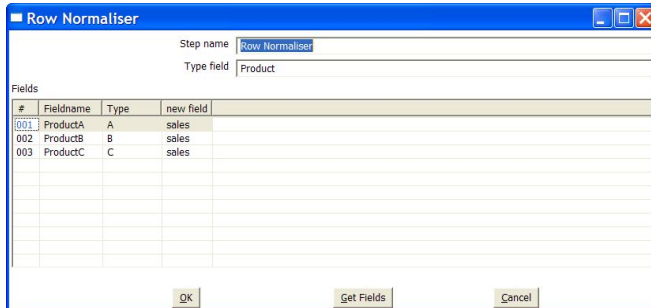


7.4.21.3 General description

This step is the same as the Insert/Update step, except that no insert is ever done in the database table. ONLY updates are performed ever.

7.4.22 Row Normaliser

7.4.22.1 Screen dump



7.4.22.2 Icon



7.4.22.3 General description

This step normalizes data back from pivoted tables for example. Take this example of product sales data:

Month	Product A	Product B	Product C
2003/01	10	5	17
2003/02	12	7	19
...

If you want to convert this data into:

Month	Product	sales
2003/01	A	10
2003/01	B	5
2003/01	C	17
2003/02	A	12
2003/02	B	7
2003/02	C	19
...

For example if you want to update a fact table, this form of data is a lot easier to handle. Well, that's what this step type does.

7.4.22.4 Options

- ✓ Step name: Name of the step. This name has to be unique in a single transformation.
- ✓ Type field: the name of the type field. (Product in our example)
- ✓ Fields: a list of fields to normalize:
 - Fieldname: name of the fields to normalize (Product A → C in our example)
 - Type: give a string to classify the field (A, B or C in our example)
 - New field: you can give one or more fields where the new value should be transferred to. (sales in our example).

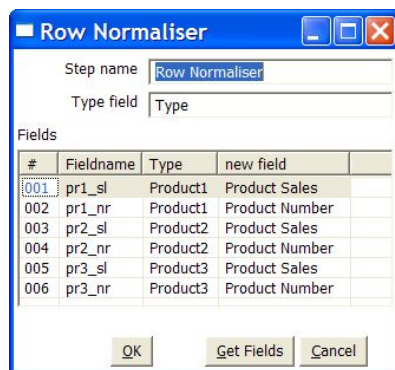
NOTE: *It is possible to normalize more then on field at a time.
Consider this example:*

DATE	PR1_Nr	PR1_SL	PR2_Nr	PR2_SL	PR3_Nr	PR3_SL
20030101	5	100	10	250	4	150
...

You can convert in into:

DATE	Type	Product Sales	Product Number
20030101	Product1	100	5
20030101	Product2	250	10
20030101	Product3	150	4
...

This would be the setup to do it with:

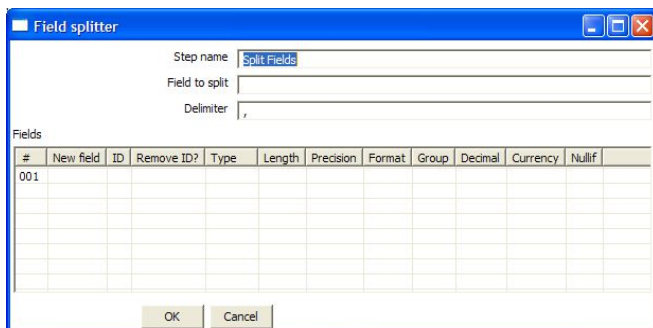


7.4.22.5 Extra

- ✓ “Get fields” button: this fills in all fields that are available on the input stream(s).

7.4.23 Split Fields

7.4.23.1 Screen dump



7.4.23.2 Icon



7.4.23.3 General description

This step allows you to split fields based upon delimiter information.

7.4.23.4 Options

- ✓ Step name: Name of the step. This name has to be unique in a single transformation.
- ✓ Field to split: the name of the field you want to split
- ✓ Delimiter: Delimiter that determines the end of a field.
- ✓ Fields: list of fields to split into.

Example: SALES_VALUES field containing: “500,300,200,100”

Use these settings to split the field into 4 new fields:

- Delimiter: ,
- Field: SALES1, SALES2, SALES3, SALES4
- Id:
- remove ID no, no, no, no
- type: Number, Number, Number, Number
- format: ###.##, ###.##, ###.##, ###.##
- group:
- decimal: .
- currency:
- length: 3, 3, 3, 3
- precision: 0, 0, 0, 0

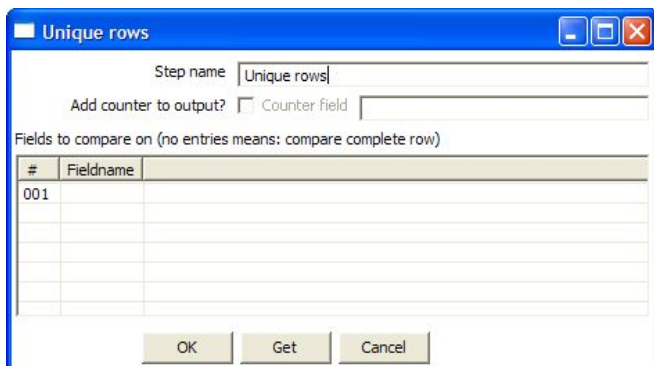
Example: SALES_VALUES field containing “Sales2=310.50, Sales4=150.23”

Use these settings to split the field into 4 new fields:

- Delimiter: ,
- Field: SALES1, SALES2, SALES3, SALES4
- Id: Sales1=, Sales2=, Sales3=, Sales4=
- remove ID yes, yes, yes, yes
- type: Number, Number, Number, Number
- format: ###.##, ###.##, ###.##, ###.##
- group:
- decimal: .
- currency:
- length: 7, 7, 7, 7
- precision: 2, 2, 2, 2

7.4.24 Unique rows

7.4.24.1 Screenshot



7.4.24.2 Icon



7.4.24.3 General description

This step removes double rows from the input stream(s).

IMPORTANT: Make sure that the input stream is sorted! Otherwise only consecutive double rows are evaluated correctly. (sometimes this can be the whole idea but just be careful :-))

7.4.24.4 Options

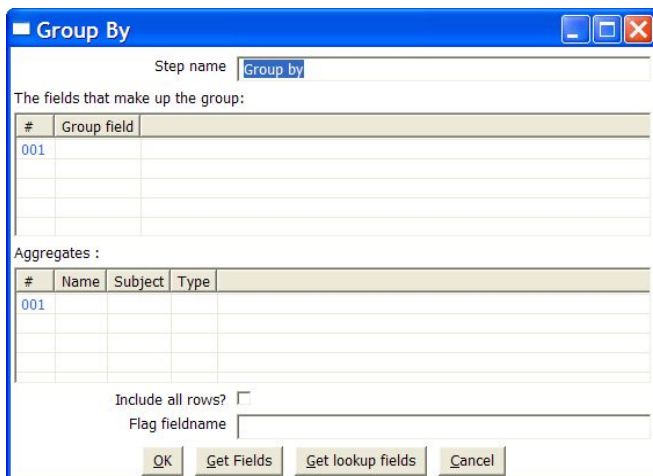
- ✓ Step name: Name of the step. This name has to be unique in a single transformation.
- ✓ Add counter to output?: Enable this if you want to know how many rows were doubles for each row in the output.
- ✓ Counter fields: name of the numeric field containing the number of rows
- ✓ Fieldnames: a list of fieldnames on which the uniqueness is compared. Data in the other fields of the row is ignored.

7.4.24.5 Extra

- ✓ “Get” button to automatically fill in the fields that are available to the step.

7.4.25 Group By

7.4.25.1 Screenshot



7.4.25.2 Icon



7.4.25.3 General description

This step allows you to calculate values over a defined group of fields.

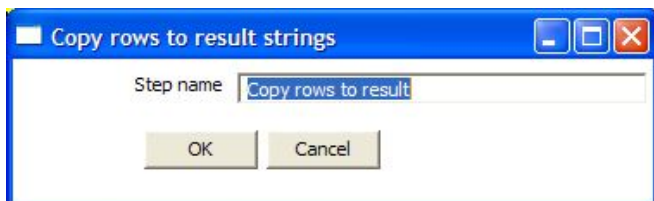
Examples that are common are: calculate the average sales per product, get the number of yellow shirts that we have in stock, etc.

7.4.25.4 Options

- ✓ Step name: Name of the step. This name has to be unique in a single transformation.
- ✓ Group fields: specify the fields over which you want to group.
- ✓ Aggregates: specify the fields that need to be aggregated, the method and the name of the resulting new field.
- ✓ Include all rows: check this if you want all rows in the output, not just the aggregation. To differentiate between the 2 types of rows in the output, we need a flag in the output. You need to specify the name of the flag field in that case. (the type is boolean)

7.4.26 Copy rows to result

7.4.26.1 Screenshot



7.4.26.2 Icon

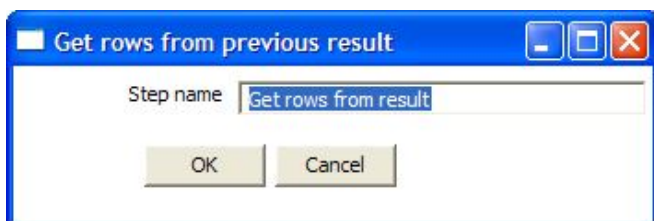


7.4.26.3 General description

This step allows you to transfer rows of data (in memory) to the next transformation (job entry) in a job. (see also: Chef user documentation)

7.4.27 Get rows from result

7.4.27.1 Screenshot



7.4.27.2 Icon

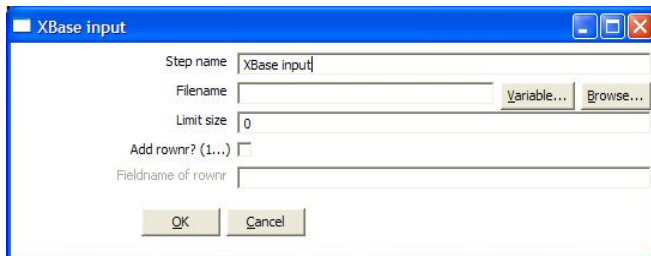


7.4.27.3 General description

This step returns rows that were previously generated by **another** transformation in a job. The rows were passed on to this step using the “Copy rows to result” step.

7.4.28 XBase input

7.4.28.1 Screenshot



7.4.28.2 Icon



7.4.28.3 General description

With this step it is possible to read data from most types of DBF file derivatives called the XBase family. (dBase III/IV, Foxpro, Clipper, ...)

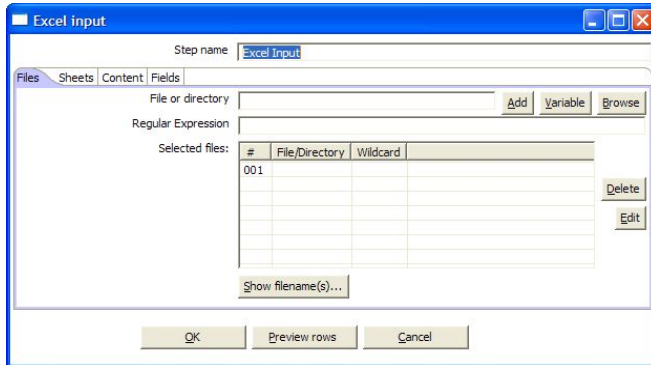
7.4.28.4 Options

- ✓ Step name: Name of the step. This name has to be unique in a single transformation.
- ✓ The filename, with variable support
- ✓ Limit size: only read this number of rows, 0 means: unlimited
- ✓ Add rownr?: This option adds a field to the output with the specified name that contains the row number.

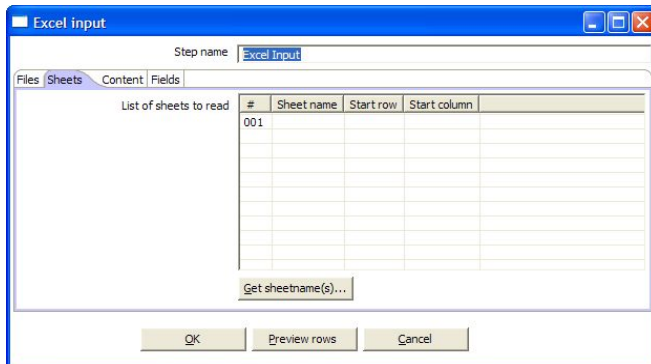
7.4.29 Excel input

7.4.29.1 Screenshots

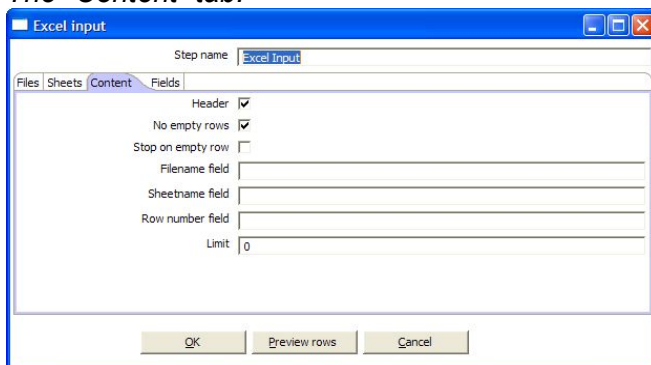
The “Files” tab:



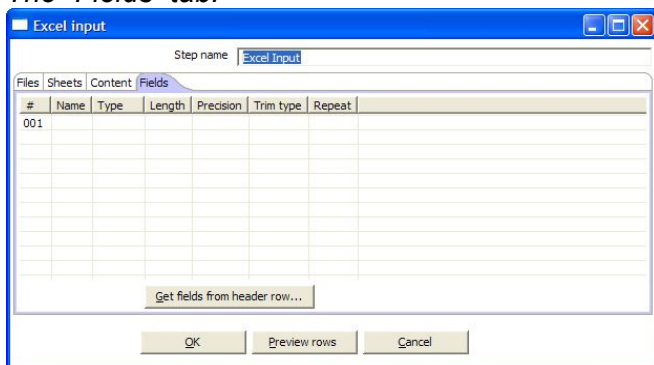
The “Sheets” tab:



The “Content” tab:



The “Fields” tab:



7.4.29.2 Icon



7.4.29.3 General description

With this step you can read data from one or more Excel files.

Note that it is not possible to convert calculated cells in the sheets as that would require Excel to be rewritten.

7.4.29.4 Options

- ✓ Step name: Name of the step. This name has to be unique in a single transformation.

File tab:

- ✓ The filenames, with variable support

Sheets tab:

- ✓ The sheets to import. *Use the “Get sheetname(s)” button to fill in the available sheets automatically.* Please note that you need to specify the start row and column, the coordinate where the step should start reading.

Content tab:

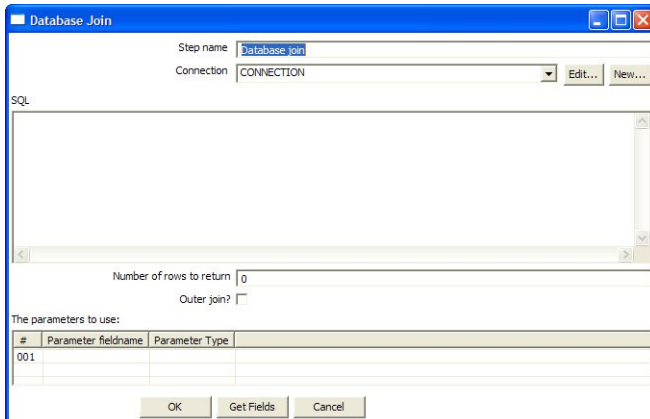
- ✓ Header: check if the sheets specified have a header row that we need to skip.
- ✓ No empty rows: check this if you don't want empty rows in the output of this step.
- ✓ Stop on empty rows: this will make the step stop reading the current sheet of a file when an empty line is encountered.
- ✓ Filename field: specify a field name to include the filename in the output of this step.
- ✓ Sheetname field: specify a field name to include the sheetname in the output of this step.
- ✓ Row number field: specify a field name to include the row number in the output of the step.
- ✓ Limit: limit the number of rows to this number, 0 means: all rows.

Fields tab:

- ✓ Here you can specify the fields that need to be read from the Excel files. A button “Get fields from header row” is provided to automatically fill in the available fields **if** the sheets have a header row.

7.4.30 Database Join

7.4.30.1 Screenshot



7.4.30.2 Icon



7.4.30.3 General description

Using data from previous steps, this step allows you to run a query against a database. The parameters for this query can be specified:

- as question marks (?) in the SQL query
- as fields in the data grid.

The two need to be in the same order.

For example, this step allows you to run queries looking up the oldest person that bought a certain product like this:

```
SELECT  customernr
FROM    product_orders, customer
WHERE   orders.customernr = customer.customernr
AND     orders.productnr = ?
ORDER BY customer.date_of_birth
```

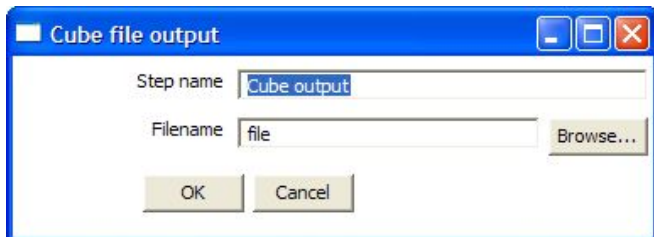
All you then need to specify as a parameter is the productnr and you'll get the customernr included in the result.

7.4.30.4 Options

- ✓ Step name: Name of the step. This name has to be unique in a single transformation.
- ✓ The database connection to use
- ✓ The SQL query to launch towards the database. (use question marks (?) as parameter placeholders.
- ✓ Number of rows to return: 0 means all, any other number limits the number of rows.
- ✓ Outer join?: check this to always return a result, even if the query didn't return a result.
- ✓ The parameters to use in the query.

7.4.31 Cube output

7.4.31.1 Screenshot



7.4.31.2 Icon

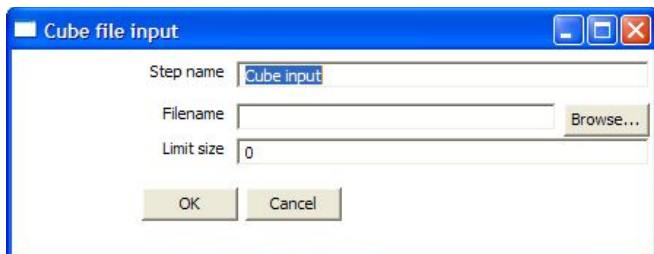


7.4.31.3 General description

This step stores rows of data in a binary form in a file. It has the advantage over a text (flat) file that the content doesn't have to be parsed when read back. This is because the meta-data is stored in the cube file as well.

7.4.32 Cube input

7.4.32.1 Screenshot



7.4.32.2 Icon

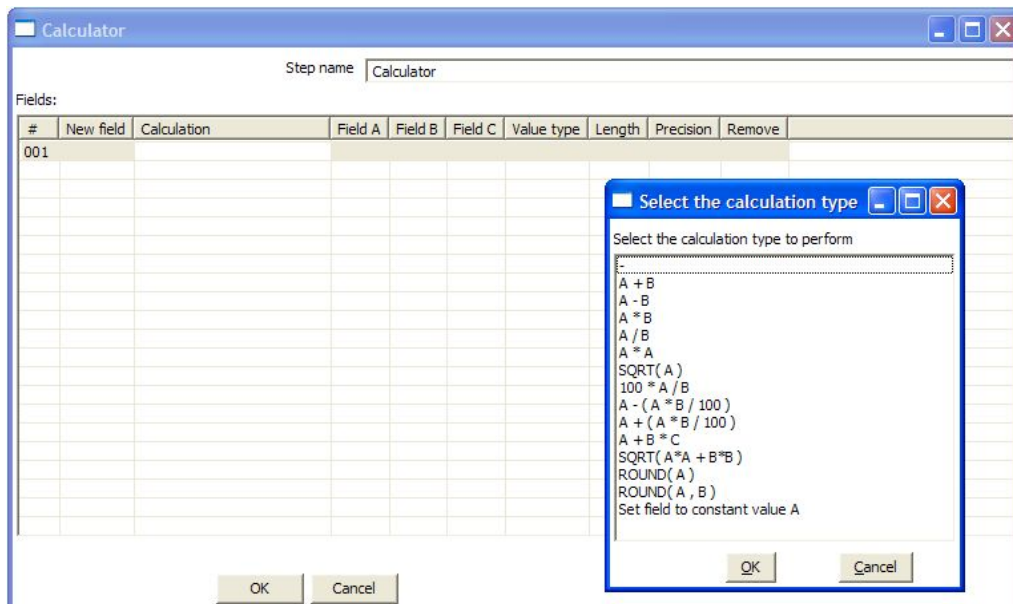


7.4.32.3 General description

Read rows of data from a binary Kettle cube file. (see also "Cube output")

7.4.33 Calculator

7.4.33.1 Screenshot



7.4.33.2 Icon



7.4.33.3 General description

This calculator step is making your life easier by providing a list of functions that can be executed on field values. Version 2.1 of Spoon is the first version to include this (short) list of functions. If you have a need for other (generic, often used) functions, simply write an e-mail to requests@kettle.be specifying the needed functionality.

An important advantage Calculator has over custom JavaScript scripts is that the execution speed of Calculator is many times that of a script.

Besides the arguments (Field A, Field B and Field C) you also need to specify the return type of the function. You can also opt to remove the field from the result (output) after all values where calculated. This is useful for removing temporary values.

This is the current list of functions:

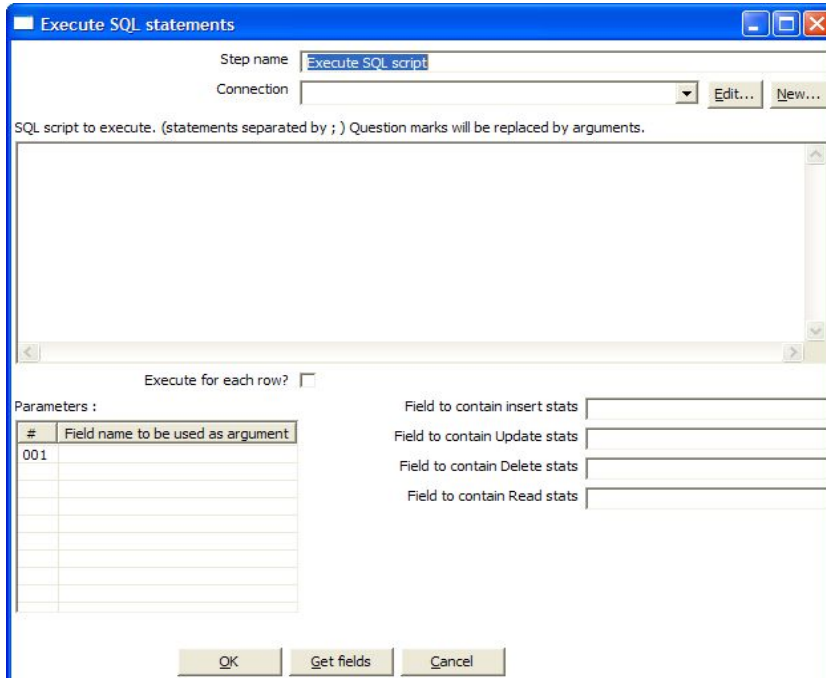
Function	Description	Required fields
A + B	A plus B	A and B
A - B	A minus B	A and B
A * B	A multiplied by B	A and B
A / B	A divided by B	A and B
A * A	The square of A	A
SQRT(A)	The square root of A	A

User manual Last updated: 11/09/2005	Kettle ETL Environment	<i>i-Bridge BVBA</i>
	Spoon Graphical Transformation Designer	

<i>Function</i>	<i>Description</i>	<i>Required fields</i>
$100 * A / B$	Percentage of A in B	A and B
$A - (A * B / 100)$	Subtract B% of A	A and B
$A + (A * B / 100)$	Add B% to A	A and B
$A + B * C$	Add A and B times C	A, B and C
$SQRT(A*A + B*B)$	Calculate $\sqrt{(A^2+B^2)}$	A and B
ROUND(A)	Round A to the nearest integer	A
ROUND(A, B)	Round A to B decimal positions	A and B
Set field to constant A	Create a field with a constant value	A

7.4.34 Execute SQL script

7.4.34.1 Screenshot



7.4.34.2 Icon

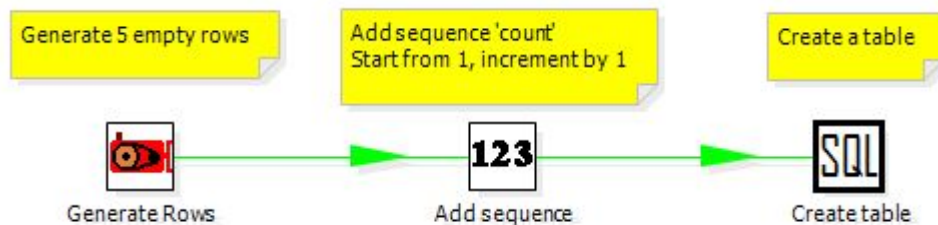


7.4.34.3 General description

You can execute SQL script with this step, either once, during the initialisation phase of the transformation, or once for every input-row that the step is given.

The second option can be used to use parameters in SQL scripts.

For example if you want to create 5 tables (tab1, tab2, tab3, tab4 and tab5) you could make such a transformation:



The SQL script to execute might look like this:

```
CREATE TABLE tab?
(
  a INTEGER
)
;
```

The field name to specify as parameter is then the “count” sequence we defined in the second step.

NOTE: *The execution of the transformation will halt when a statement in the script fails.*

As extra option, you can return the total number of inserts (INSERT INTO statements), updates (UPDATE table), deletes (DELETE FROM table) and reads (SELECT statements) by specifying the field names in the lower right of the dialog.

8 GRAPHICAL VIEW

8.1 Description

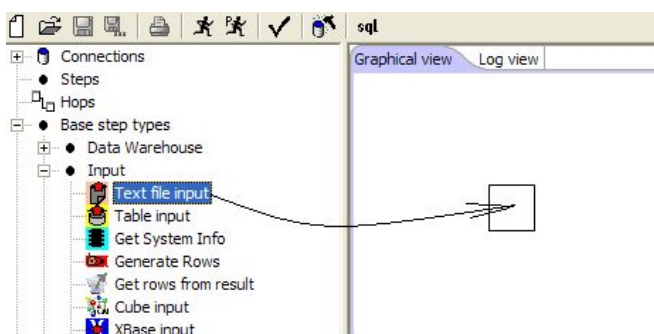
The Graphical View is the canvas on which transformations are drawn.

It shows an easy to understand representation of the work that needs to be done and the flow of the data.

8.2 Adding steps

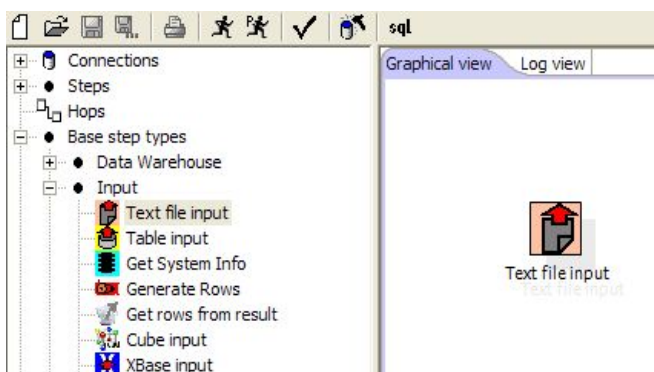
8.2.1 Dragging

Adding steps to a transformation on the canvas is easy: simply select a step type from the tree on the left and drag in onto the canvas:



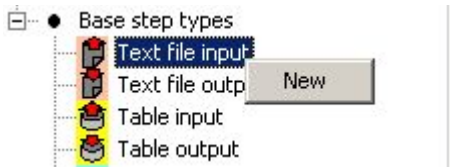
At the location of the mouse you will see a square that represents the location of the steps when you let go of the button.

When you let go of the mouse button the selected step (Text file input) will become part of the transformation.



8.2.2 Creating a new step

Another way to create an extra step for your transformation is by selecting “New” step when you click right on the step type in the tree on the left. You can also just double-click on the step type.



After this, you will be shown a dialog where you can enter the settings. For the complete description of the dialogs and settings, please check out §7.4.

When you press OK in the dialog a new step is created in the transformation.

You will see in this case however that the step is not drawn. That is of-course because we haven't told Spoon where to put it. We say that the step is not drawn but hidden.



If you drag this step onto the canvas, it will become drawn.

8.2.3 Hiding a step

If you click right on a step that is drawn on the graphical view, you will get a popup-menu that allows you to select the option: “Hide step”. This will remove the step from the graphical view, but not delete it.

8.2.4 Step options (popup menu)

8.2.4.1 Edit step

This opens the step dialog so that you can change its settings.

8.2.4.2 Edit description

This opens a dialog that allows you to enter a textual description of the step.

8.2.4.3 Copy data / distribute data

See §7.3 (Distribute or copy)

8.2.4.4 Copies

See §7.2 (Launching several copies of a step)

8.2.4.5 Duplicate step

This option will create a copy, positioned a bit lower to the right of the original step.

8.2.4.6 Delete step

This will permanently remove the step from the transformation.

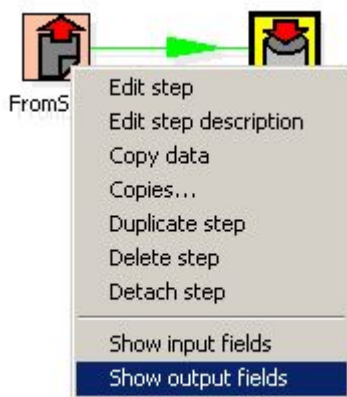
8.2.4.7 Show input fields

This option tries to determine all the fields and their origin by tracing the input-streams back to their source.

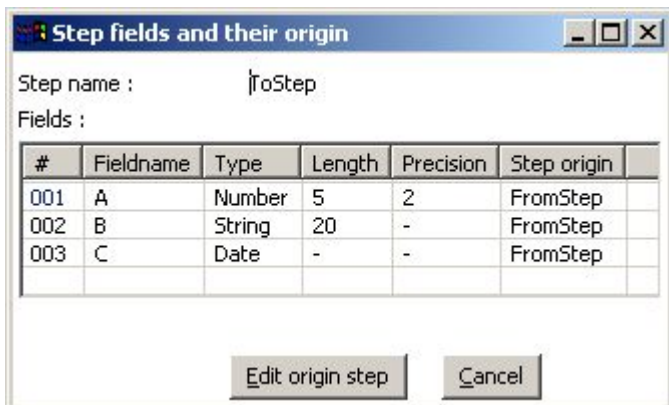
8.2.4.8 Show output fields

This option adds the fields of the current step to the ones of the input fields.

For example this might be an example:



The result could then possibly be:



Step name : ToStep

Fields :

#	Fieldname	Type	Length	Precision	Step origin
001	A	Number	5	2	FromStep
002	B	String	20	-	FromStep
003	C	Date	-	-	FromStep

Edit origin step Cancel

8.3 Adding hops

On the graphical view the quickest way to create a new hop is by dragging with the mouse from one step to another using the middle button.

You can also drag the left button and press the control key at the same time.

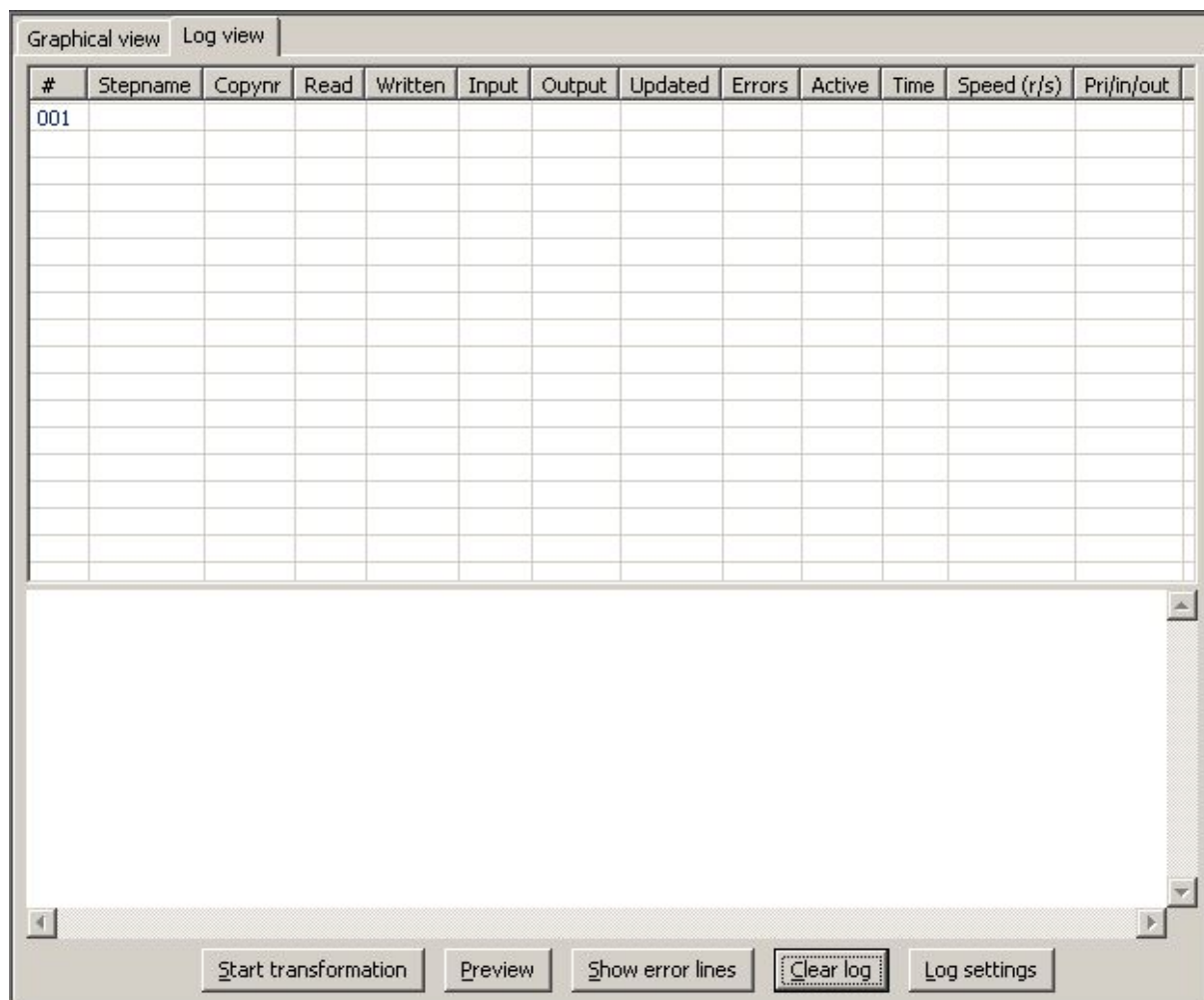
For a more complete explanation regarding hops, please check out §5 (Hops)

9 LOG VIEW

9.1 Description

The Log View shows what's happening when a transformation is running. First of all it shows all the (copies of) the steps that are launched as a part of the transformation. Secondly, it shows the log as it would be shown if the transformation would be launched by Pan.

9.2 Screenshot



Log Grid

Buttons

Log Text

9.3 Log Grid

The log grid shows the following columns:

- ✓ The name of the step
- ✓ Copy number of the step
- ✓ Number of lines read from input-streams
- ✓ Number of lines written to output-streams

User manual Last updated: 11/09/2005	Kettle ETL Environment	<i>i-Bridge BVBA</i>
	Spoon Graphical Transformation Designer	

- ✓ Number of lines read from file or database
- ✓ Number of lines written to file or database
- ✓ Number of lines updated in the database
- ✓ Number of errors that occurred
- ✓ The status of the step: running, finished or stopped
- ✓ The number of seconds that the step has been running.
- ✓ The speed in rows per second at which the step processes rows.
- ✓ Priority of the step (10=highest, 1=lowest), nr of rows in the input-stream(s), nr of rows in the output-stream(s).

NOTE: *The system is tuning the steps priority in such a way that the slowest steps get the highest priority.*

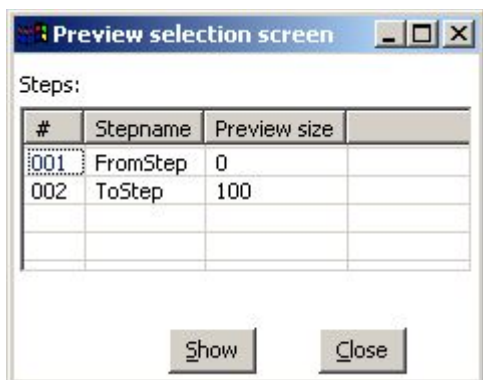
9.4 Buttons

9.4.1 Start transformation

This button starts the transformation you're designing. Please note that Spoon tries to launch this as Pan would: from XML-file or repository. It is therefore necessary that the transformation is saved. The output of the execution is displayed in the Log Text part of the Log View.

9.4.2 Preview

This buttons starts the transformation as it is defined in memory at the time you press the Preview button. The output of the execution is displayed in the Log Text part of the Log View. After you press the button you will be presented with a dialog like this:



Here you can enter the rows per step you want to preview. The preview size is automatically set to 100 rows for the steps that are selected. If no step is selected, the previous setting is taken. After the requested rows are obtained from the different steps, the transformation is ended. After the transformation has ended, the result is shown.

9.4.3 Show error lines

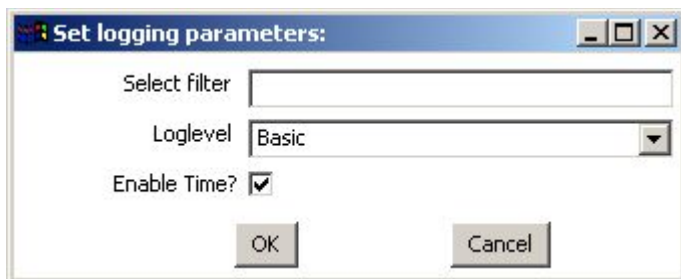
This option will show you all lines of the Log Text that contain the word ERROR (lower- or uppercase). You can then choose to edit the source step of the error.

9.4.4 Clear log

This clears the text in the Log Text Window.

9.4.5 Log Settings

This is the “Log Settings” dialog:



If you put a text in the filter field, only the lines that contain this text will be shown in the Log Text window.

The “Log level” setting allows you to select the logging level. You can choose one of these:

- ✓ Error: Only show errors
- ✓ Nothing: Don't show any output
- ✓ Minimal: Only use minimal logging
- ✓ Basic: This is the default basic logging level
- ✓ Detailed: Give detailed logging output
- ✓ Debug: For debugging purposes, very detailed output.
- ✓ Row level: Logging at a row level, this can generate a lot of data.

If the “Enable time” option is enabled, all lines in the logging will be preceded by the time of day.

User manual Last updated: 11/09/2005	Kettle ETL Environment	<i>i-Bridge BVBA</i>
	Spoon Graphical Transformation Designer	

10 GRIDS

10.1 Description

Grids are used everywhere in Spoon & Kettle. They are used to enter or display information.

10.2 Functions

Most of the functions available in a grid (that is not read-only) are available by right clicking with the mouse on a grid:

- ✓ Insert before this row: inserts an empty row before the row you clicked on
- ✓ Insert after this row: inserts an empty row after the row you clicked on
- ✓ Move the row up: move the row you clicked on up. The keyboard shortcut for this is CTRL-UP
- ✓ Move the row down: move the row you clicked on down. The keyboard shortcut for this is CTRL-DOWN
- ✓ Optimal column size including header: resize all columns so that it displays all values completely, including the header. The keyboard shortcut for this function is function key F3.
- ✓ Optimal column size excluding header: resize all columns so that it displays all values completely. The keyboard shortcut for this function is function key F4.
- ✓ Clear all: clears all information in the grid. You will be asked to confirm this operation.
- ✓ Select all rows: selects all rows in the grid. The keyboard shortcut for this function is CTRL-A
- ✓ Clear selection: clears the selection of rows in the grid. The keyboard shortcut for this function is ESC
- ✓ Copy selected lines to clipboard: Copies the selected lines to the clipboard in a textual representation. These lines can then be exchanged with other programs such as spreadsheets or even other Spoon & Kettle dialogs. The keyboard shortcut for this function is CTRL-C
- ✓ Paste clipboard to table: Insert the lines that are on the clipboard to the grid, right after the line on which you clicked. The keyboard shortcut for this function is CTRL-V
- ✓ Cut selected lines: Copies the selected lines to the clipboard in a textual representation. After that, the lines are deleted from the grid. The keyboard shortcut for this function is CTRL-X
- ✓ Delete selected lines: deletes all selected lines from the grid. The keyboard shortcut for this function is DEL
- ✓ Keep only selected lines: if there are more lines to delete then there are to keep, simply select the lines you want to keep and use this function. The keyboard shortcut for this function is CTRL-K
- ✓ Copy field values to all rows: if all rows in the grid need to have the same value for a certain column, you can use this function to do this.
- ✓ Undo: undo the previous grid operation. The keyboard shortcut for this function is CTRL-Z
- ✓ Redo: redo the next grid operation. The keyboard shortcut for this function is CTRL-Y

10.3 Navigating

If you click on a cell in a grid, you can edit this field.

After pressing enter, you can navigate the grid by using the cursor keys.

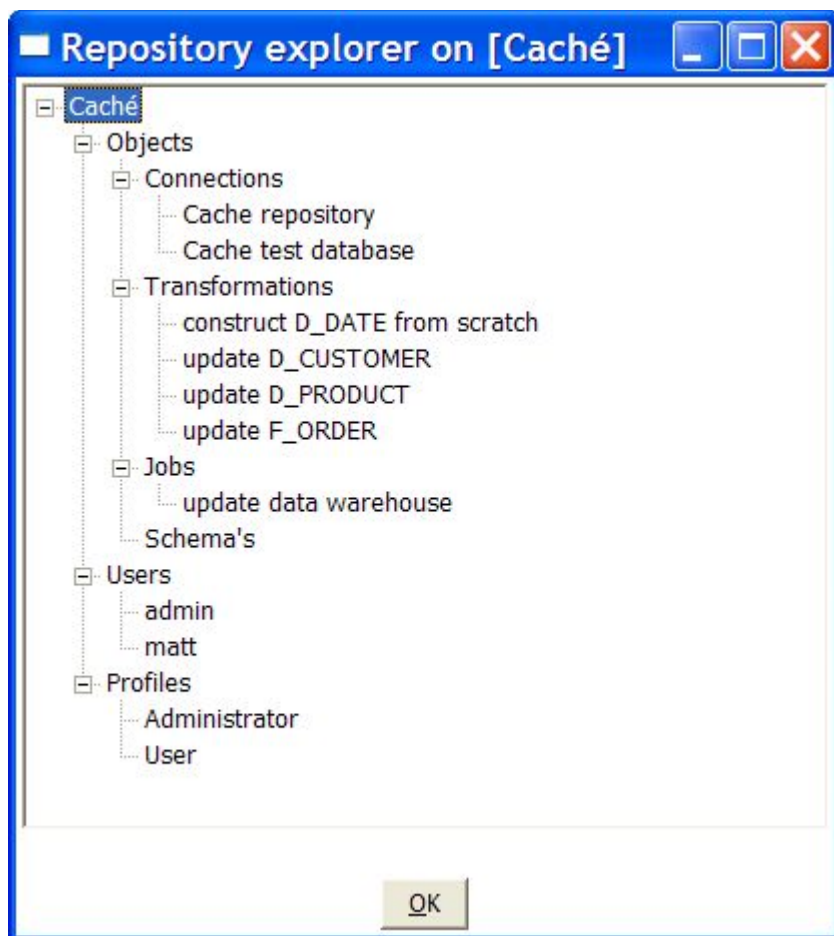
Pressing enter again allows you to edit a field.

11 REPOSITORY EXPLORER

11.1 Description

The repository Explorer shows you a tree view on the database repository to which you are connected. It allows you to examine and modify the content.

11.2 Screenshot

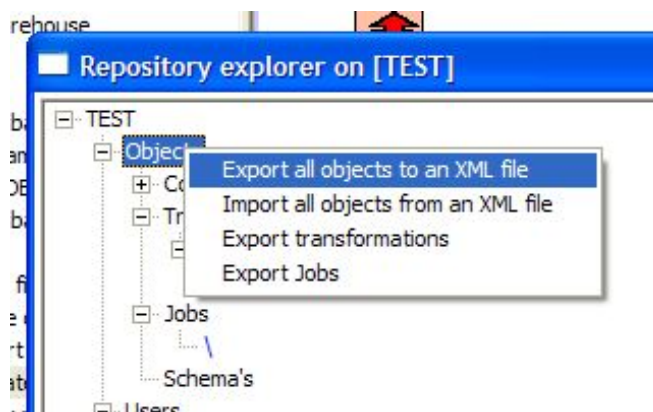


11.3 Functions

Check out the different functions when right clicking in the tree. It's fairly straightforward.

11.4 Backup / Recovery

It is possible to export the complete repository in XML: click right on “Objects” in the repository and select one of these options.



NOTE: you can restore the objects from a backed up repository anywhere in the target repository directory tree.