Pentaho Data Integration version 2.4.0

Integrating more, easier & faster!

Key differences with Kettle 2.3.0

List of changes on 20-01-'07

Compiled by Matt Casters, mcasters (at) pentaho.org

Send additional changes you found to this address.

Index

1. Changes summary.... 4
1.1. Preface.... 4
1.2. Overview.... 5
2. General changes.... 6
2.1. Spoon enhanced, RIP Chef.... 6
2.2. A new welcome screen.... 7
2.3. Variable support with visual feedback.... 7
2.4. A new execution configuration dialog.... 8
2.5. Slave servers & clustering.... 8
2.6. Databases.... 9
2.7. Shared objects.... 11
2.8. Repository improvements.... 12
2.9. Samples.... 13
2.9.1. Jobs.... 13
2.9.2. Transformations.... 13
3. Spoon.... 15
3.1. Extra options.... 15
3.2. Search meta-data.... 16
3.3. Unique connections / transactional fitness.... 16
3.4. Feedback configurations.... 16
3.5. Thread priority management.... 17
4. Steps.... 18
4.1. Changed steps.... 18
4.1.1. Text file input.... 18
4.1.2. Get System Info.... 18
4.1.3. Generate rows.... 19
4.1.4. Text file output.... 19
4.1.5. Table Output, Insert/Update, Update, Delete, Database Lookup, Dimension Lookup/Update, Combination lookup/update.... 19
4.1.6. De-serialize from / Serialize to file.... 19
4.1.7. Stream Value Lookup.... 20
4.1.8. Sort rows.... 20
4.1.9. Add sequence.... 21
4.1.10. Group by.... 22
4.1.11. Calculator.... 22
4.1.12. Denormaliser.... 23
4.2. New steps.... 24
4.2.1. Modified Javascript Value.... 24
4.2.2. Streaming XML Input.... 25
4.2.3. Excel Output.... 26
4.2.4. Access Output.... 26
4.2.5. HTTP Web service.... 27
4.2.6. Sorted Merge.... 27
4.2.7. Merge join.... 28
4.2.8. Socket reader & writer.... 28
5. Job entries.... 29

5.1. Changed job entries..... 29
5.1.1. Transformation..... 29
5.1.2. Mail..... 29
5.1.3. SQL..... 30
5.1.4. FTP..... 30
6. Source code improvements..... 31
6.1. A few extra lines of code..... 31
6.2. Trackers..... 31
6.3. Committers..... 31
6.4. Bug reporters..... 32
6.5. Feature requesters..... 32
6.6. Other contributors..... 33

1. Changes summary

1.1. Preface

It's been almost 7 months since the last major revision (version 2.3.0) of the Pentaho Data Integration software (a.k.a. Kettle). There has been an update (2.3.1) in between, but this change log will cover all changes that occurred between versions 2.3.0 and 2.4.0.

As you will see below in this document, all is well with the evolution of Pentaho Data Integration:

- The amount of support from the community seems to be growing and growing
- The amount of bug fixes is going up and up which is great for stability
- The amount of implemented features is staggering
- We had massive amounts of downloads (>200k) of the latest versions
- We have had a lot of feedback and in general it's fair to say that the community is thriving on the new forum to which we migrated.
- Our software is now faster and easier to use than ever before.

This document was written as a special “thank you” note to all people involved in the community and to keep everyone informed about the incredible progress we are making.

1.2. Overview

These are the most notable changes that have been made:

- Improved Performance and Scalability
  - Remote monitoring and execution of Transformations
  - Clustering support (MPP)
  - Database partitioning support (not to be confused with table partitioning)
  - Numerous memory and performance improvements

- Enhanced database support
  - Improved quoting algorithms
  - separation of schema/owner and table names throughout the steps
  - Databases connection pooling support
  - Clustering support
  - 25 database types (+generic) are now supported. The new database types that are supported now are: *Teradata*, *Oracle RDB*, *H2*, *Netezza*, *IBM UniVerse* and *SQLite*.

- Ease of Use
  - Spoon is now an integrated designer for Jobs and Transformations
  - Design and edit multiple transformations simultaneously
  - Repository improvements including the ability to sort contents by name, user, object type and modified date
  - Variable enhancements including added support in many step types and visual indicators on all fields that support variables

- Powerful New Steps
  - High performance, expression-based Javascript step
  - Add XML step for building sophisticated XML constructs from stream fields
  - Merge Join step for advanced joins like INNER, LEFT OUTER, RIGHT OUTER, AND FULL OUTER
  - Fast Sorted Merge join for merging multiple streams sorted on the same key
  - HTTP Client lookup to dynamically retrieve parameters using a web service

- Miscellaneous
  - Remote monitoring of transformation and job execution
  - Ability to share database connections using XML without having to use the repository
  - Ability to roll-back entire transformations on error (unique connections)

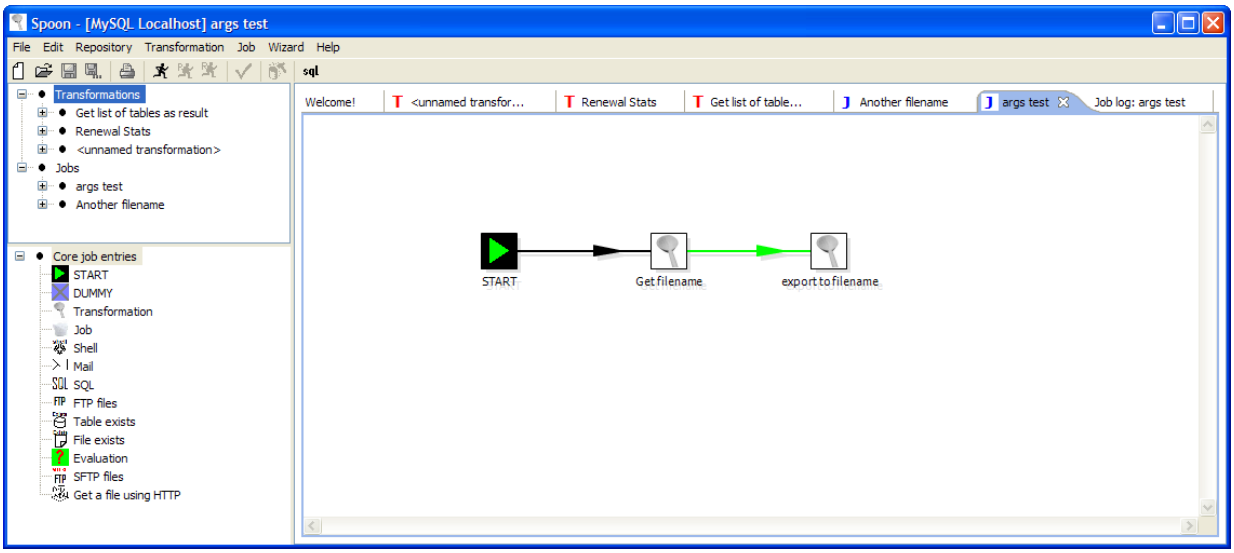
- By the numbers
  - >200.000 downloads of 2.3.0 and 2.3.1 (mostly 2.3.1)
  - 4000 SVN commits
  - >1000 forum threads
  - 262 closed bugs with an average closing time of 29 days.
  - 51 feature requests implemented with an average implementation time of 100 days.
  - 10 developers committing to SVN

2. General changes

2.1. Spoon enhanced, RIP Chef

As you can see from the image below you can now load multiple transformations and jobs into a single Spoon editor. This removed the need for the Chef job editor and makes it easier to edit larger jobs.

Not only can you load jobs and transformations, but you can also run multiple jobs & transformations, monitor remote slave servers and view execution history in the same environment.

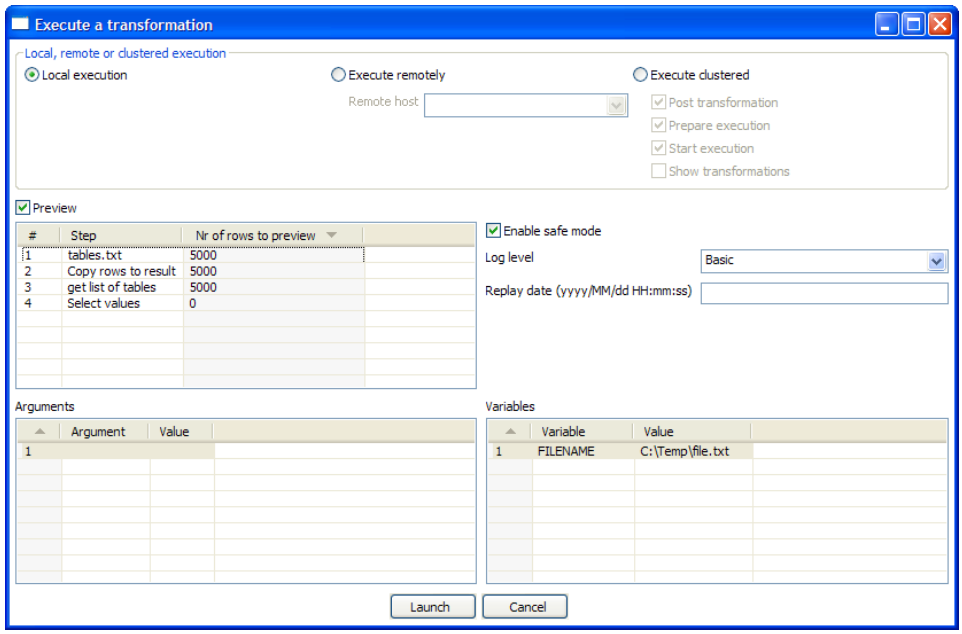


2.2. A new welcome screen

2.3. Variable support with visual feedback

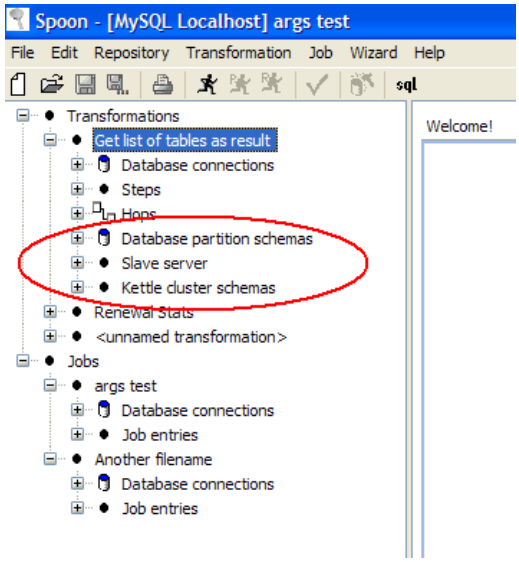
2.4. A new execution configuration dialog

This dialog combines local, remote and clustered execution as well as preview, log settings, safe mode, arguments and variable support in one dialog.



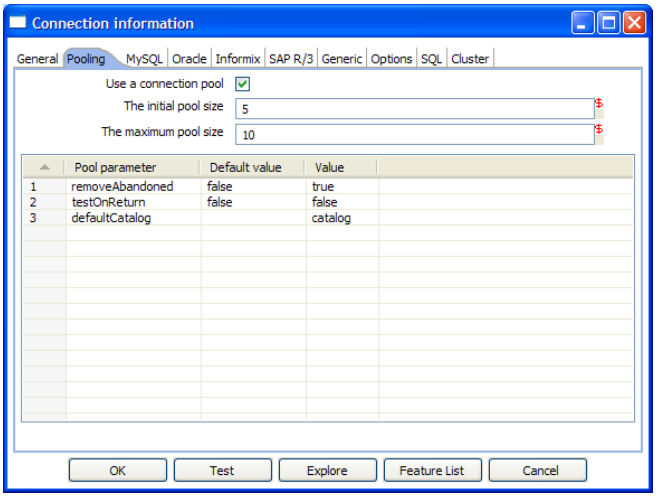
2.5. Slave servers & clustering

With this new release we move into the fascinating realm of massive parallel computing. It is now not only possible to execute a transformation on a remote server, but also to execute steps in a clustered way. By clustered we mean in parallel on more than one server. To allow this to work we created a small web-server called “Carte” that will accept input from either Spoon (remote & clustered execution) or from the Transformation job entry. (clustered execution)

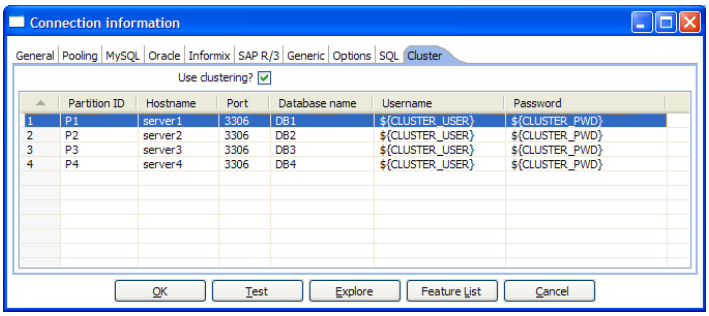


2.6. Databases

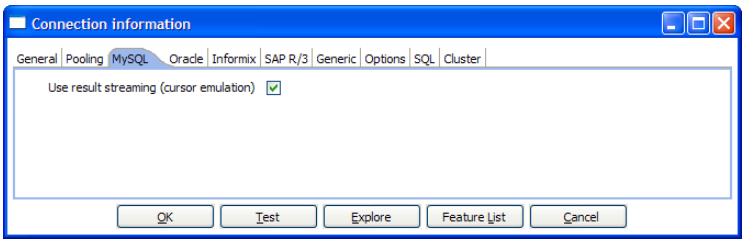
- Teradata support (http://www.teradata.com/)
- Oracle RDB support (http://www.oracle.com)
- H2 support (http://www.h2database.com), requested first by Kent Andrews
- Netezza support (http://www.netezza.com), dialect and driver by Biswapesh Chattopadhyay for Google
- IBM UniVerse support (http://www-306.ibm.com/software/data/u2/universe/), testing & driver by Frans van Dortmont)
- SQLite support (http://www.sqlite.org/), implemented with help from Tom Gleeson http://gobansaor.wordpress.com/2006/12/18/sqlite-jdbc-and-kettle-pentaho-data-integration-etl/
- Connection pooling support added, written by 陈萍 (a.k.a. Apple : <chenping (at) gxlu.com.cn>), feature request 3838 by Jeremy Haile <javaforge (at) jhaile.fastmail.fm> was also implemented.



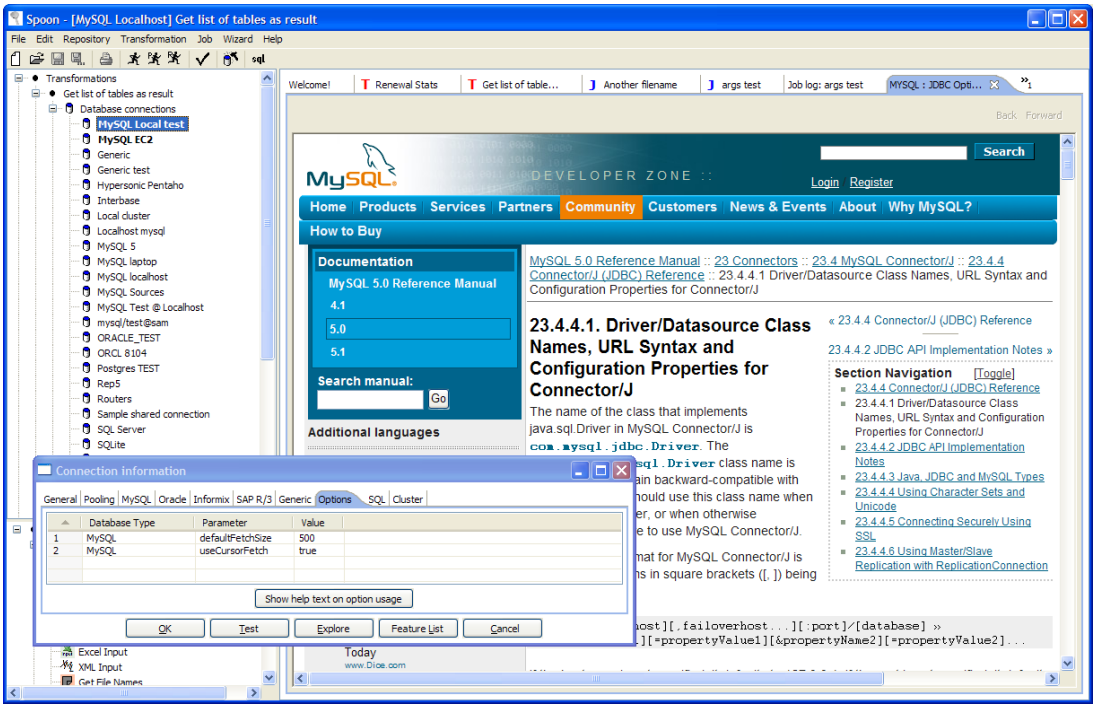
- Clustering/partitioning support to allow data to be split up among databases



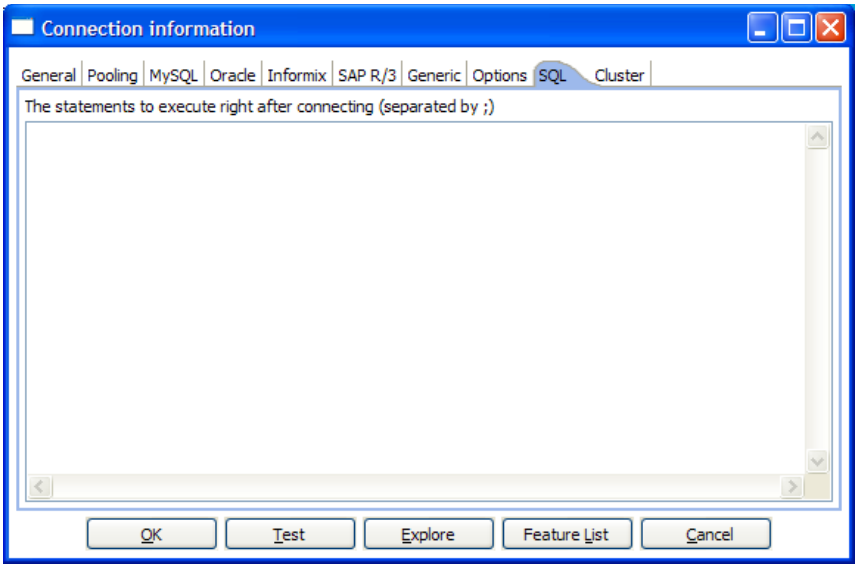
- MySQL: Allow cursor emulation (a.k.a. result streaming) to be turned off



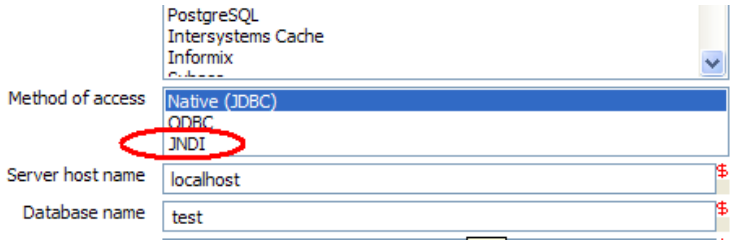
- Clicking the options help button now opens a new browser tab in Spoon:



- Right after connecting, it is now possible to run a number of SQL commands. This is sometimes needed for various reasons such as licensing, configuration, logging, tracing, etc.



- JDNI connections are now also supported (code by James Dixon, Pentaho)

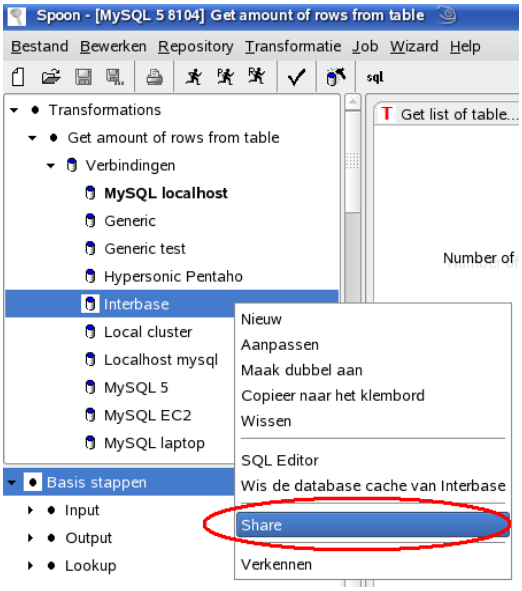


2.7. Shared objects

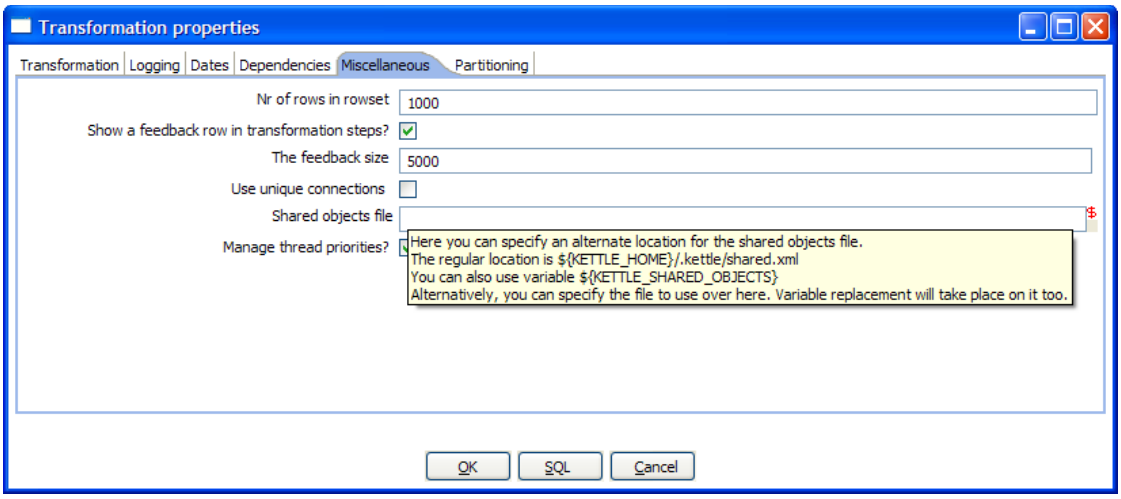
Various objects can now be placed in a shared objects file. This file (default: $HOME/.kettle/shared.xml) can be used to define:

- Database connections
- Steps
- Slave servers
- Partition schemas
- Cluster schemas

On these objects you can simply click right and select “Share”.

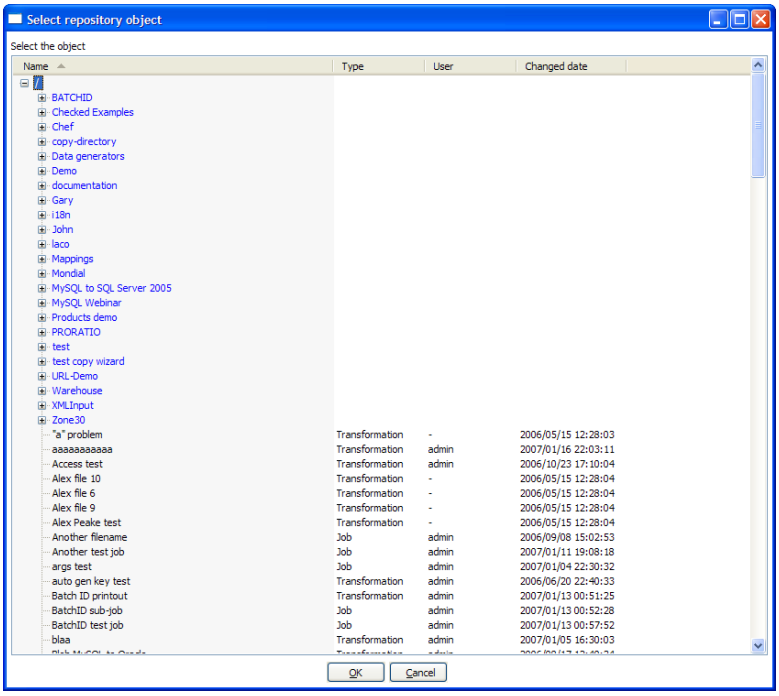


Doing so will place the object in the shared objects file. The location of the file is configurable:

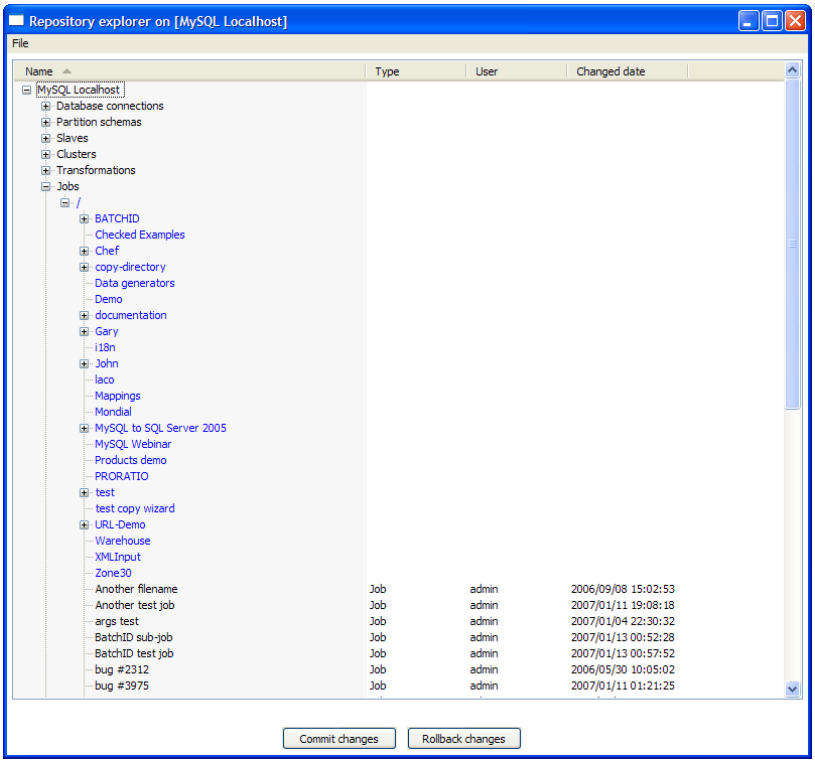


2.8. Repository improvements

We included extra information on the jobs and transformations stored in the repository in as well the repository explorer as the File/Open operations. This makes it easier for you to search for a particular transformation or job:



The repository also contains support for this as well as for the new objects like slave servers:



2.9. Samples

Because you sometimes need to get a starting point when learning a new tool, we added a simple list of samples for both jobs and transformations.

2.9.1. Jobs

Folder name	*Description*
Arguments	This job runs a transformation with a fixed set of arguments
Arguments2	This job passes the command line arguments down to the transformation and writes to a file to see that it worked
Changelog	Checks if a “changelog” table exist. If so it gets dropped and re-created. The changelog.txt file is processed. After that and if all records are read correctly in the previous job entry, the rows from the file are split up by “New”, “Deleted”, “Changed” and processed accordingly. The last job entry is executed for each input row.
Changelog-groups	This is a more optimal example of the previous sample as it groups similar data together in groups making the loop more efficient.
Process all tables	Process data from a list of tables, in this sample a “count(*)” is done on all tables in a MySQL user schema and the results get written to files called ${TABLENAME}.txt
Process flow with adding streams	Sample for - Get data from a lot of different input steps - Do some complex transformation for each input - Sort all the results from all the inputs together - Write all to 1 file
Run_all	This job looks in a directory and finds all transformations with a certain name and executes them all.

2.9.2. Transformations

Transformation	*Description*
Add sequence - specify a common counter.ktr	Example on how to use the new "Counter" feature in the "Add Sequence" step
Aggregate - basics.ktr	Basic aggregation example
Calculator - Substract constant value one from a number.ktr	Use the calculator to do a simple "minus one" calculation
Combination Lookup - Create new ids.ktr	Create new Ids using the "Combination Lookup/Update" step
Database - generic driver usage.ktr	Shows the usage of the "Generic" database driver
Denormaliser - 2 series of key-value pairs.ktr	Advanced "Denormaliser" example using 2 series of key-value pairs
Denormaliser - Simple example.ktr	Simple "Denormaliser" example
Excel and Row Normaliser - basics.ktr	Basic example of using "Excel Inpu" and the "Normaliser" steps.
Excel Output - Write different data types.ktr	Write different data types to an Excel worksheet using "Excel Output"
General - Copy Data.ktr	Simple example on how to copy data to the next steps (in stead of distribute)
General - Load images and store into database table.ktr	Loads images using javascript and stores these images into binary

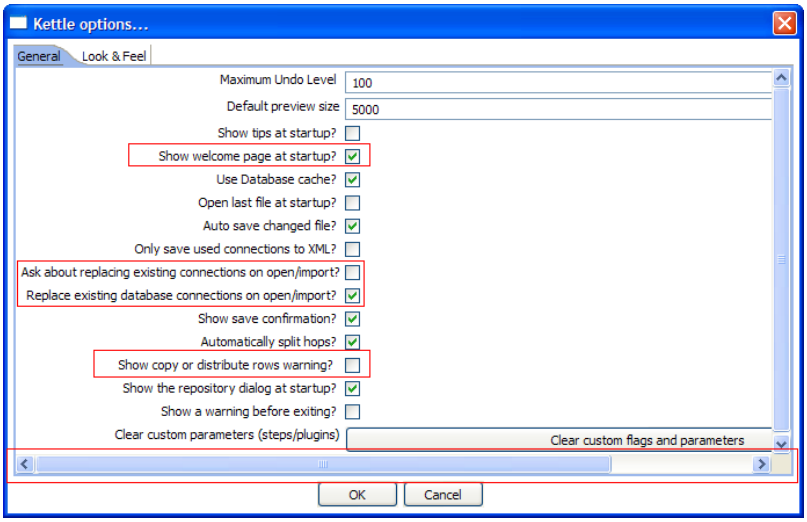
	(BLOB) fields in a database table.
General - Populate date dimension.ktr	Simple reference implementation of a Date dimension, created from scratch.
General - Repeat fields - Group by - Denormalize.ktr	Sample using the "Repeat fields" option in "Text file input", a group by and a row denormaliser
Generate Row - basics.ktr	Basic "Generate rows" example
Get System Info - Command Line Argument and filter.ktr	Simple example on how to filter data based on command line settings.
Group by - include all rows and calculations .ktr	Shows the "Include all rows" functionality in the "Group By" step.
Group By - include all rows without a grouping.ktr	Shows the "Include all rows" functionality in the "Group By" step but without using a grouping.
JavaScript - Access database connection metadata.ktr	Access the metadata used by the transformation itself. (reflection example)
JavaScript - Access environment variables.ktr	How to access environment variables
JavaScript - Add 2 and a half seconds to a Date.ktr	How to add two and a half seconds to a date.
JavaScript - Base64 Decoder and Encoder.ktr	Basic Base64 decoding and encoding example
JavaScript - create new rows.ktr	Create new rows from within Javascript
JavaScript - date to string conversion.ktr	Example on how to convert a date into a string
JavaScript - dialog.ktr	Show two dialogs asking for date values
JavaScript - extract date from filename.ktr	Extract the date from a filename
JavaScript - get subdir.ktr	Get a sub-directory
JavaScript - Hex to Integer conversion.ktr	Convert a hexadecimal string value into an decimal integer
JavaScript - parse Debet-Credit prefix in numbers.ktr	Parses debit-credit prefixes in numbers and signs values accordingly
JavaScript - process all fields per row.ktr	Process all the fields in all rows.
JavaScript - replace values in a string with other values.ktr	Replace values in a string with other values
JavaScript - Split String.ktr	Split a string into multiple parts using a delimiter
JavaScript - String to Number to String conversion.ktr	Convert a string to a number and back to a string
JavaScript - Strip line feeds from text.ktr	Strip line feeds from text fields
Join Rows - adding fields to a stream.ktr	Add fields to a stream using the "Join Rows" step
Merge Join - Join 2 sorted data sets.ktr	Join 2 sorted data sets using "Merge Join"
Merge rows - mergs 2 streams of data and add a flag.ktr	Merge two streams of data using "Merge rows" and add a flag showing the changes
Row Normaliser - turn a single row into 3 rows.ktr	Turn a single row into 3 rows using the "Row Normaliser" step.
Select Values - copy field values to new fields.ktr	Copy field values to new fields using the "Select Values" step.
Select values - some variants.ktr	Shows the possible uses for the "Select Values" step
Stream lookup - basics.ktr	Basic example of a "Stream Lookup" in action
Table Output - Tablename in field.ktr	This shows you how to handle a situation where you have a tablename to write to in a separate field in the rows.
Text File Output - Number formatting.ktr	How to do advanced number formatting in the "Text File Output" step
Textfile input - fixed length.ktr	How to read a fixed length text file
Unique - Case insensitive unique.ktr	This is a simple example on how to apply a case insensitive unique function to your data set
Value Mapper - Same and new field.ktr	Showing the difference between mapping values to the same or a new field using the Value Mapper
XBase - dBase.ktr	Read a DBF file using the "Xbase input" step
XML Add - encode fields in XML and add the field.ktr	Shows you how to encode fields in XML into a new field
XML Input - basic reading flat XML.ktr	Simple example of reading an XML flat file

3. Spoon

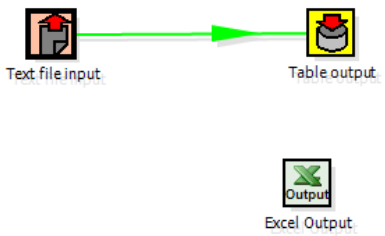
For a list of the changes that were done in the steps & job entries, please see the corresponding chapters below.

3.1. Extra options

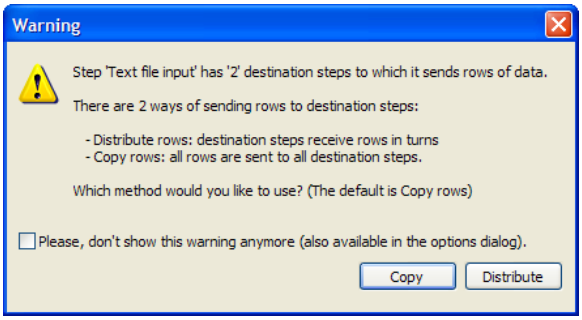
These are the new Spoon interface options that were added:



Most of the new options explain themselves. However, here are the details on the “copy/distribute rows warning” option. If you have a transformation like this:

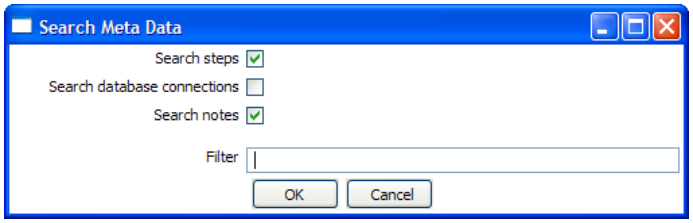


At the time you link step “Text file input” to “Excel Output”, you will be shown this dialog:



Apart from this, we made the dialog content scroll if it is sized too small. We do this to prevent options from being hidden involuntarily.

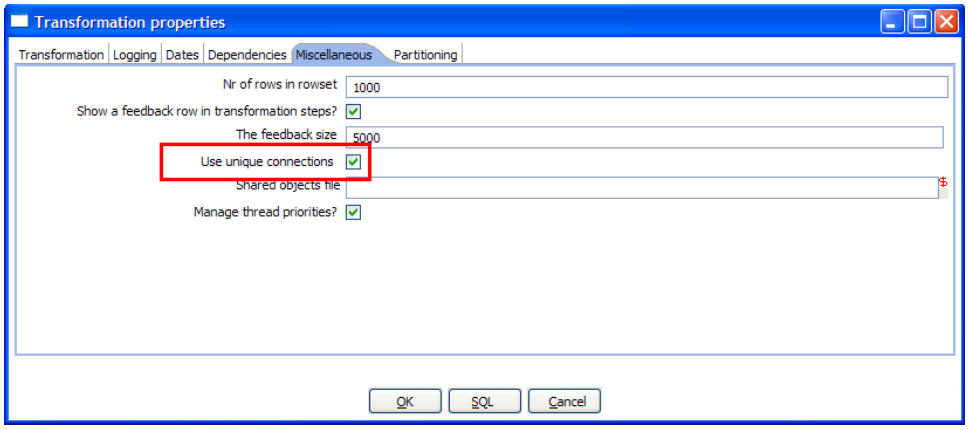
3.2. Search meta-data



This option was introduced in 2.3.0 but will now search through all transformations and jobs that are loaded in Spoon.

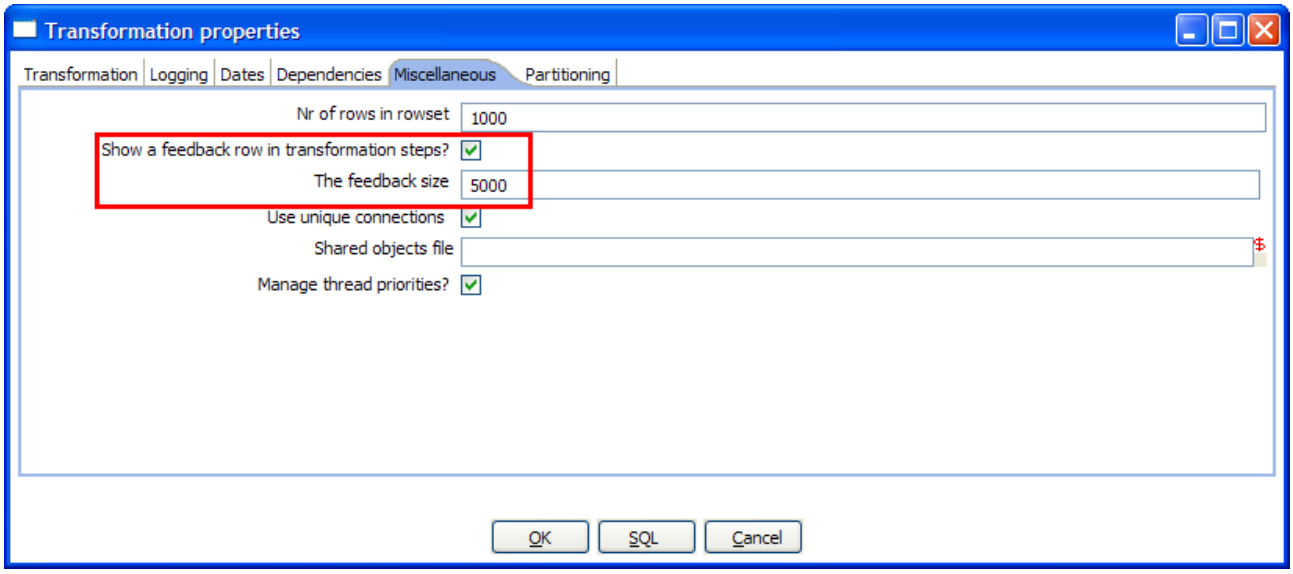
3.3. Unique connections / transactional fitness

We have made a serious attempt to make sure that failing transformations can be rolled back completely if this is desired. Typically you might use this for smaller data volumes and/or data migration/integration exercises. This feature is accomplished by opening one unique connection per defined and used database connection in the transformation.



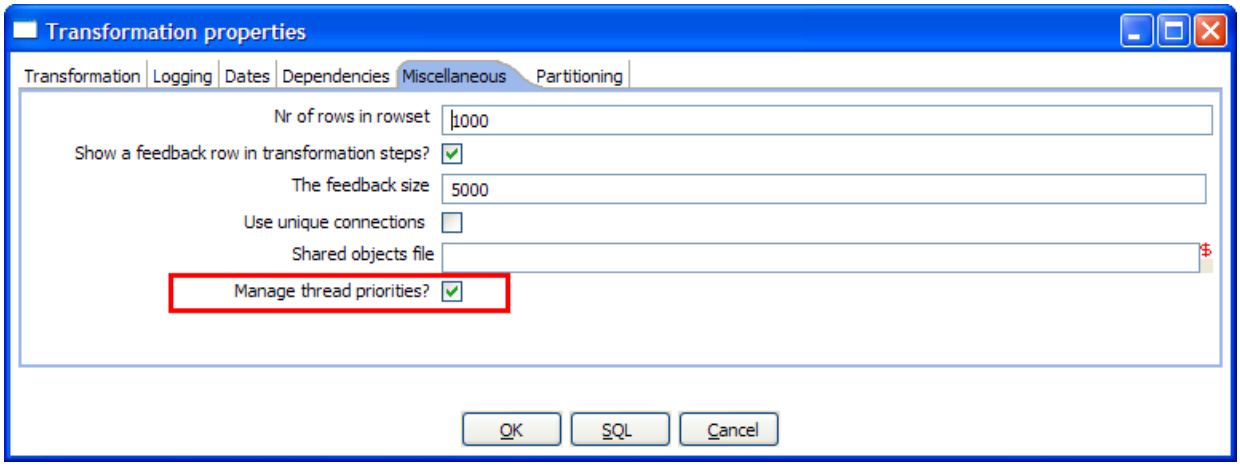
3.4. Feedback configurations

While processing millions of rows it logging can grow to large volume with a feedback every 5000 rows. This is now configurable in the transformations settings. You can turn it of or change the interval or feedback size.



3.5. Thread priority management

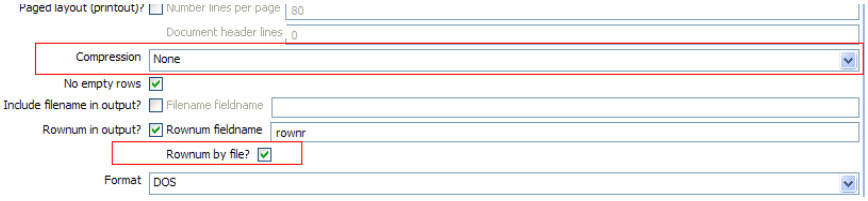
Since a few years back, Kettle transformations change Java thread priorities based on the number of input and output rows in the respective “rowset” buffers. However, in certain simplistic situations, this is costing more time than it's worth and therefor you can now turn it off it you want.



4. Steps

4.1. Changed steps

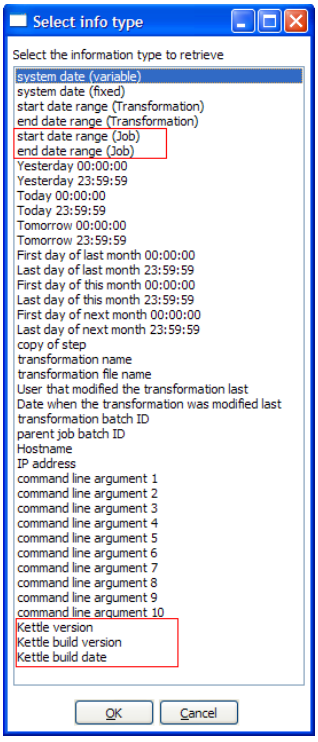
4.1.1. Text file input



- Allow the method of compression to be specified as either Gzip or Zip.
- Allow the row number to be reset per file.

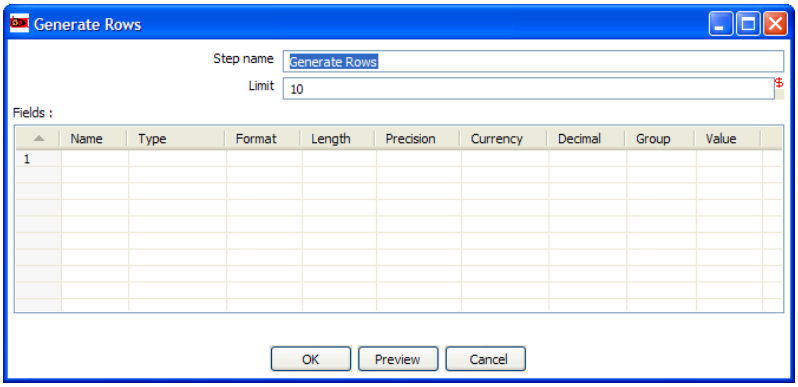
4.1.2. Get System Info

We gave access to the job start and end of the date range as well as the Kettle specifics like version, build date and build version.



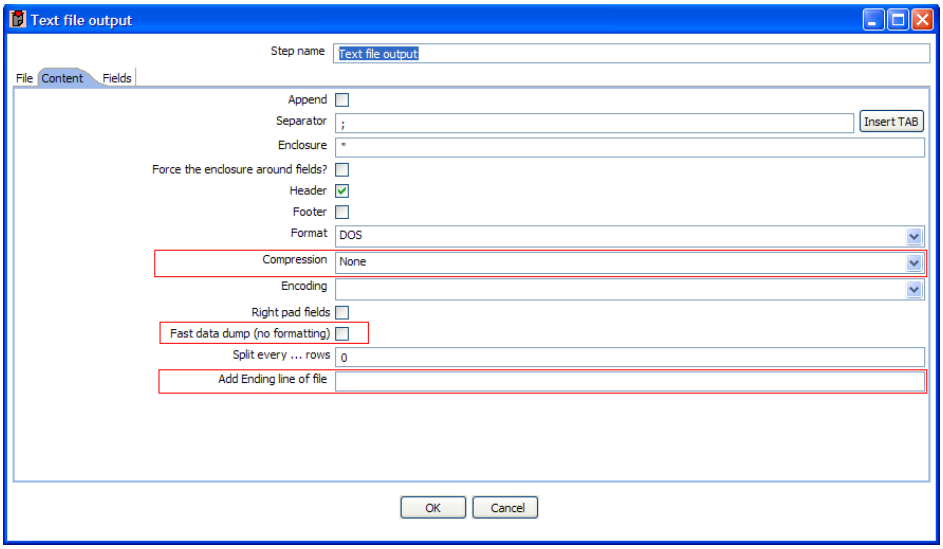
4.1.3. Generate rows

Now accepts for the limit to the number or generated rows a variable:



4.1.4. Text file output

We added support for Gzip and Zip compression as well as a “Fast data dump” option to speed up dumping large amounts of data to a text file. We also added the option to come up with an alternative ending for your text.



4.1.5. Table Output, Insert/Update, Update, Delete, Database Lookup, Dimension Lookup/Update, Combination lookup/update

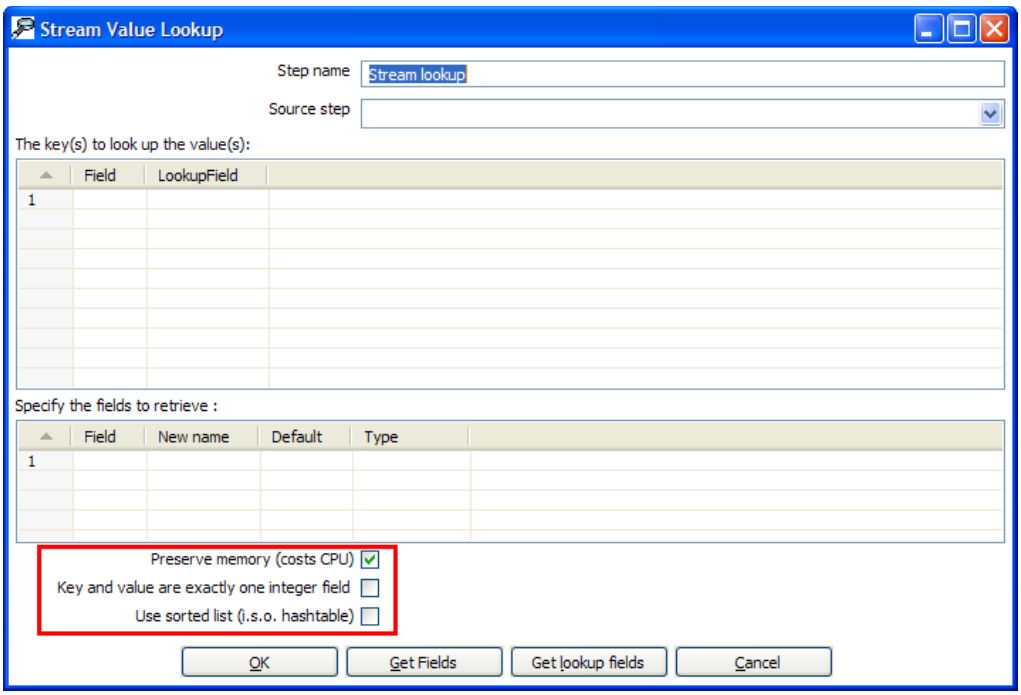
Added the schema name to improve precision in the quoting and allow for table-names with dots '.' in it. (like on certain CRM systems like Axapta)

4.1.6. De-serialize from / Serialize to file

Re-branded Cube-input/output steps. (same as before but optimized for speed)

4.1.7. Stream Value Lookup

We have included the option to encode rows of data to preserve memory. These options, especially 'Preserve memory' and 'Key and Value are exactly one integer field' lead to enormous gains in memory efficiency. While these encodings can lead to a slight decrease in performance, it is in general still worth to do it. We have seen tens of millions of rows being loaded in less than a GB of RAM this way. This opens up possibilities that were not available previously.



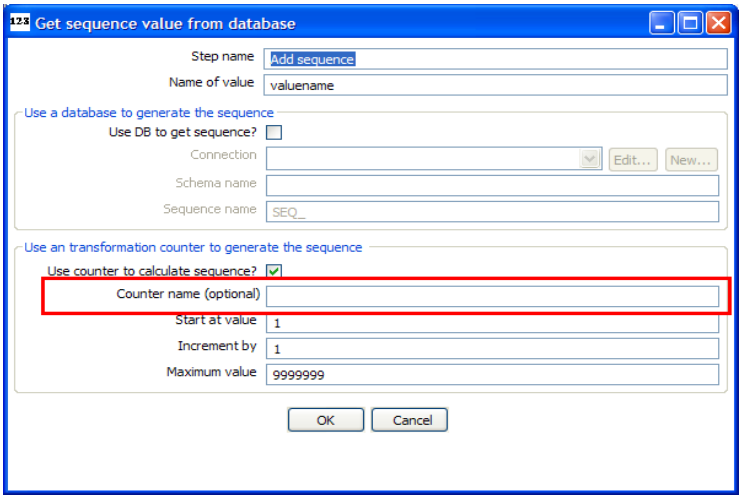
The sorted list option is less performing than the others but will lead to the possibility to allow for other-than -equal key comparisons in the next version of Kettle.

4.1.8. Sort rows

The sorting algorithm has been optimized and will now perform a lot better than before, expect 100% or more speed gains.

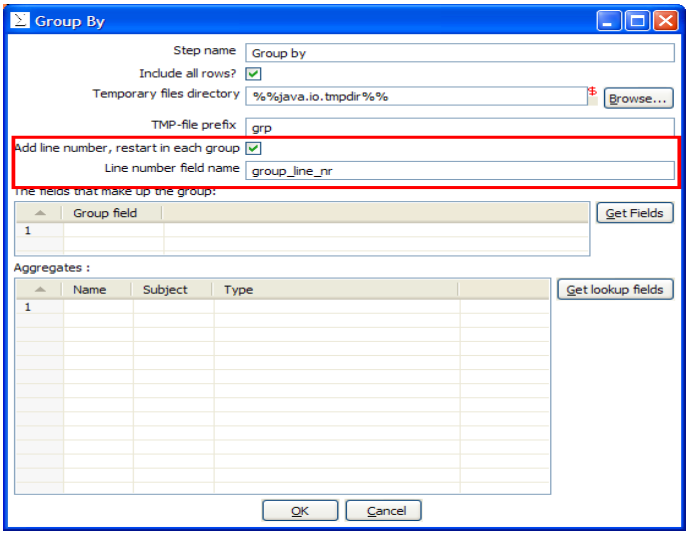
4.1.9. Add sequence

The old behavior of the Add Sequence step was that if you had multiple steps in a single transformation that generated the same value name, the sequence would be giving unique numbers to these steps. That is because it uses the same counter internally. Now you can set the name of the counter in cases where you do not want that behavior.



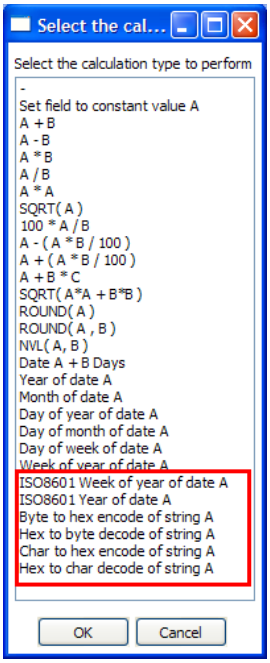
4.1.10. Group by

In case you allow all rows to pass and the aggregated information is added to all rows in the defined groups, you can now also add a line-number that restarts at 1 in each new group.



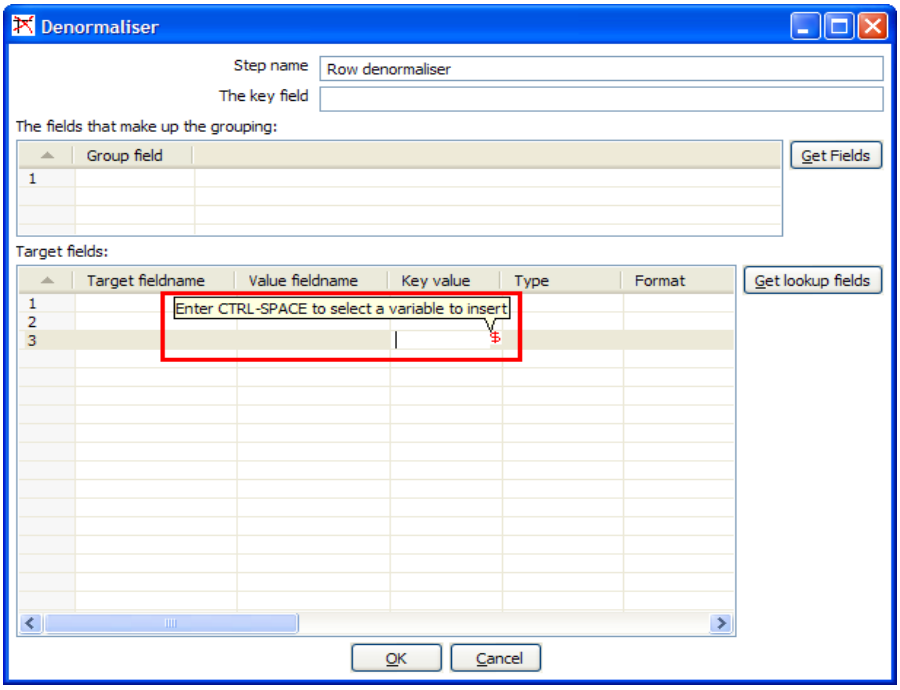
4.1.11. Calculator

The calculator can now perform a few extra calculations.



4.1.12. Denormaliser

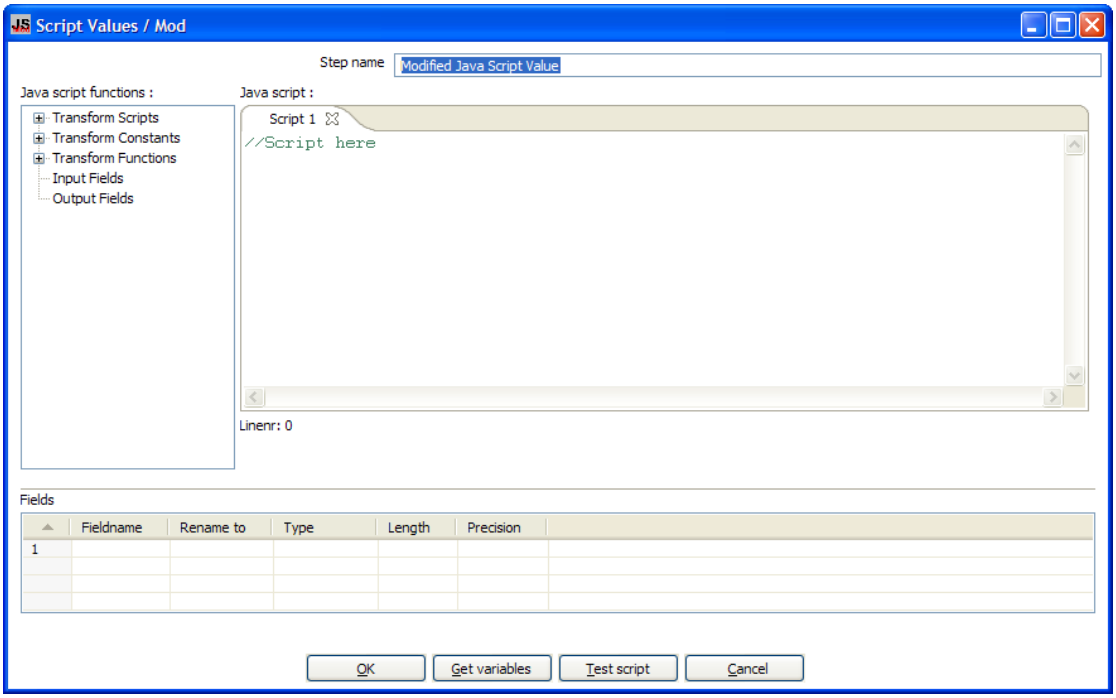
It is now also possible to use variables in stead of hard-coded entries for the key-values.



4.2. New steps

4.2.1. Modified Javascript Value

Martin Lange from company Proconis (http://www.proconis.de/) rewrote the existing “Javascript values” looking for better ease of use in the GUI and better performance. This has resulted in the “Modified Javascript Value” step.

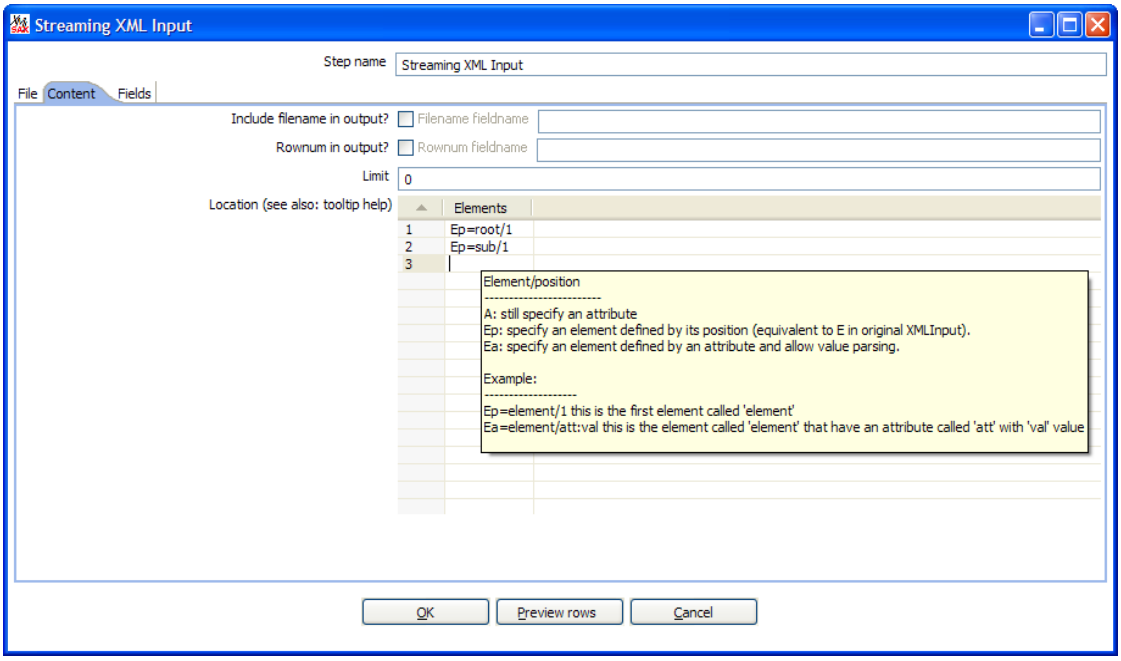


As you can see from the screenshot above it is now possible to have multiple separated script tables, but also “Start” and “End“ scripts (before and after).

Color coding is also available for known keywords, functions, constants, comments, etc.

Because the executable code is not 100% compatible (for performance reasons) we have opted to keep the regular and the modified Javascript steps alongside each other.

4.2.2. Streaming XML Input



The purpose of this step is to provide value parsing.
This step is based on SAX parser to provide better performances with larger files.
It is very similar to XML Input, there are only differences in content and field tabs.

Content

Location specifies the path by way of elements to the repeating part of the XML file. The new thing, is that in element column you specify the position of the element as described below.

Field

There is a new grid that allows specifying an association between an element name an it defining attribute name. This makes the step able to retrieve the good fields when pressing "get fields" button.
The position column is also different it works like described above

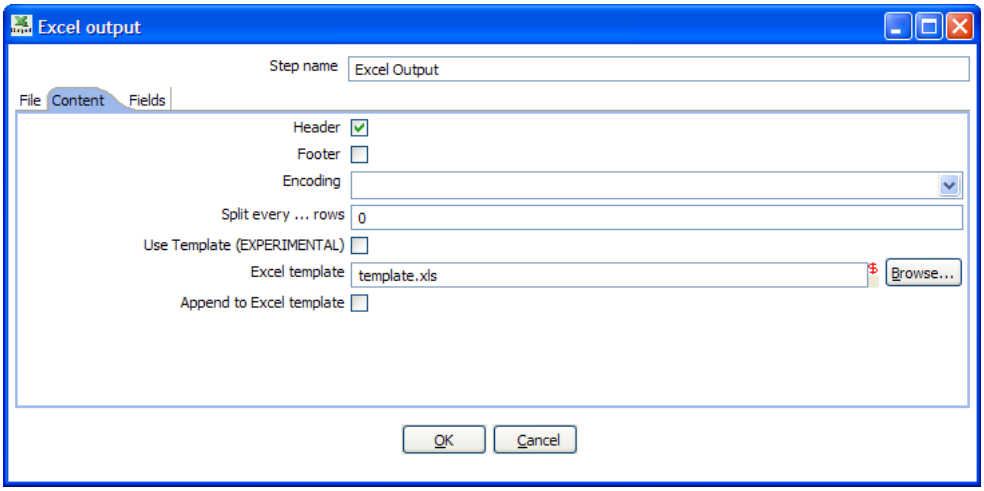
Element/position

A: still specify an attribute
Ep: specify an element defined by its position (equivalent to E in original XML Input).
Ea: specify an element defined by an attribute and allow *value parsing*.

4.2.3. Excel Output

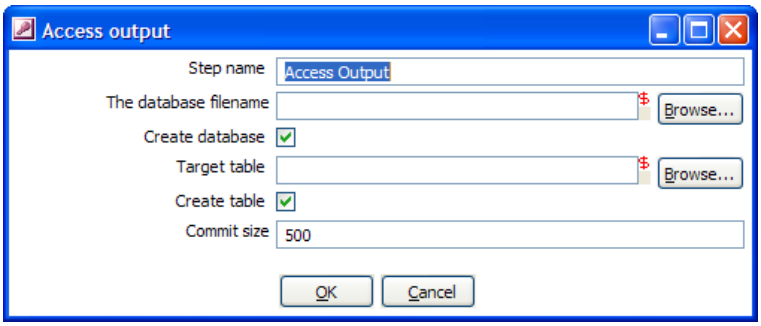
Kettle has no intention whatsoever to go and replace functionality found in complex reporting engines.

That being said, once in a while it is useful to be able to output to simple Excel sheets natively. Because of that the Excel Output step was written. It only allows for very basic formatting options, but keep in mind that that is not the intent of this step.



4.2.4. Access Output

This falls in the same category as the Excel output step: it can sometimes be handy to create new Access database files, but don't expect us to add complex functionality to this.

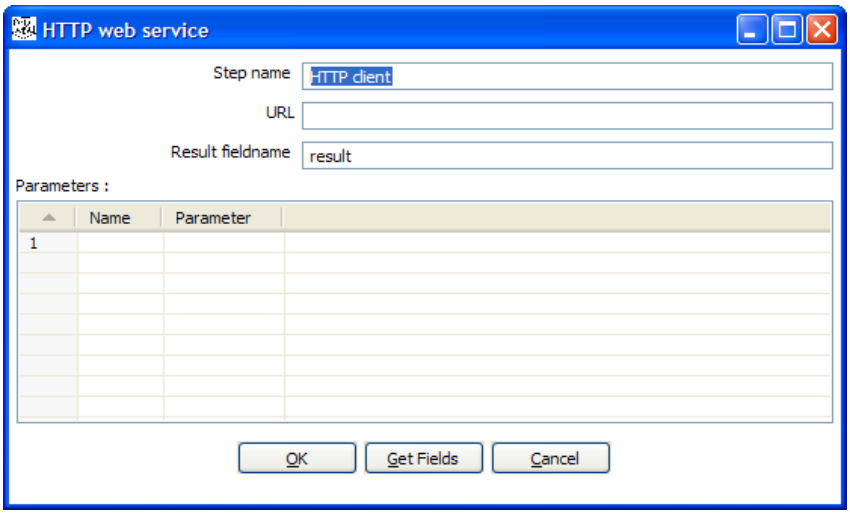


4.2.5. HTTP Web service

The HTTP Web service performs a very simple call to a base URL with options appended to it like this:

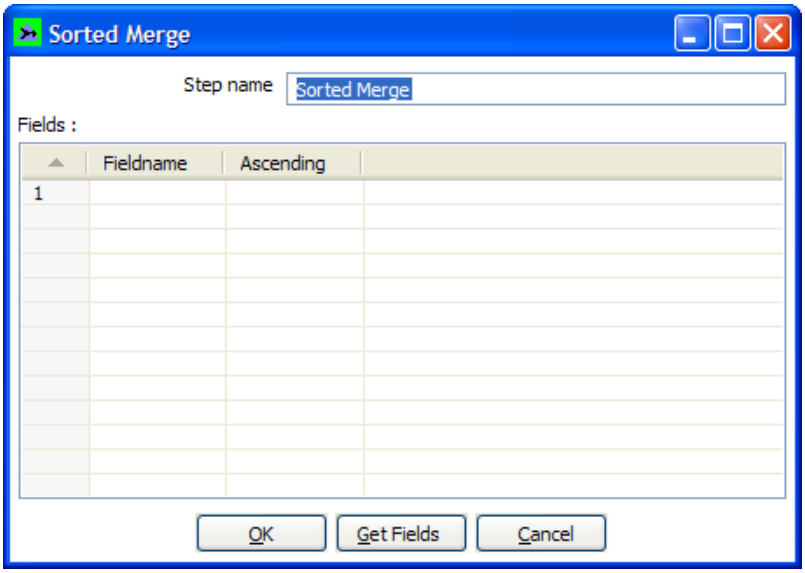
`http://<URL>?param1=value1&param2=value2&...`

The result is stored in a String field with the specified name.



4.2.6. Sorted Merge

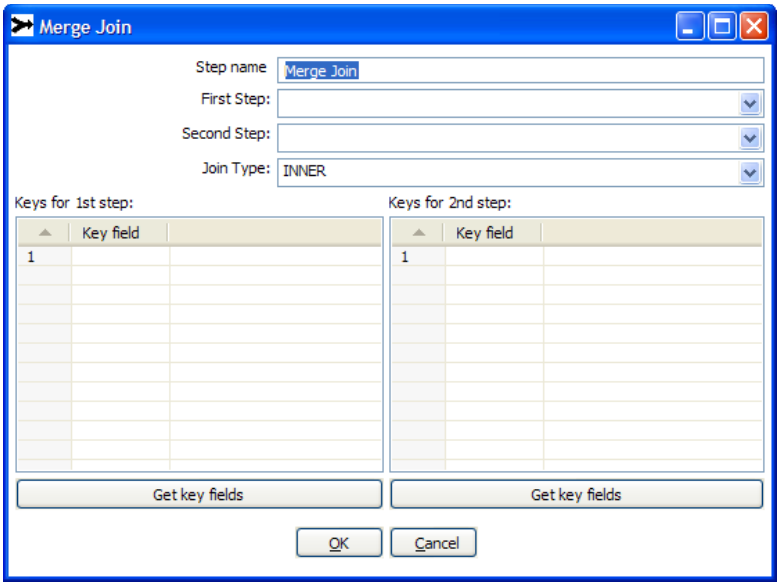
The sorted merge step merges rows coming from multiple input steps providing these rows are sorted themselves on the given key fields.



4.2.7. Merge join

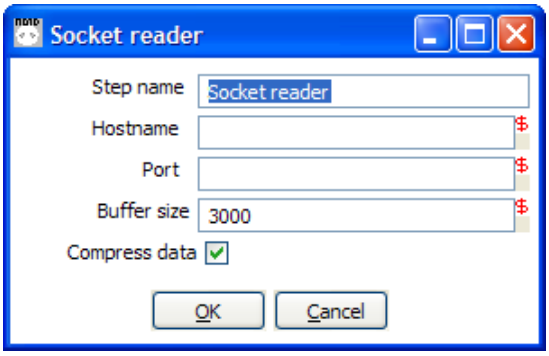
This step performs a classic merge join between data sets of data coming from 2 input steps.

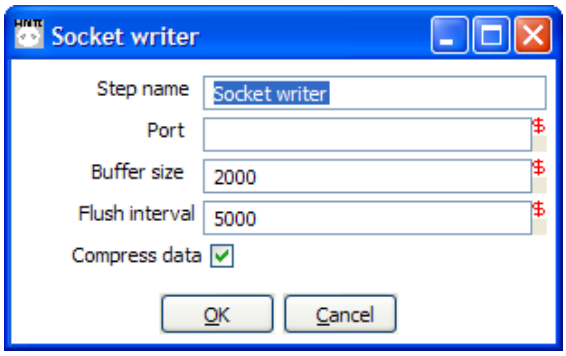
The join options are: INNER, LEFT OUTER, RIGHT OUTER, FULL OUTER. This step too expects the rows to be sorted on the specified key fields.



4.2.8. Socket reader & writer

Socket readers and writers are used to transfer data from one server to another over TCP/IP. The primary use for these steps is in-line in a clustering environment. If you want to use these yourself, make sure to synchronize the preparation and start cycles of the transformations between the hosts. (like the clustered transformation does)



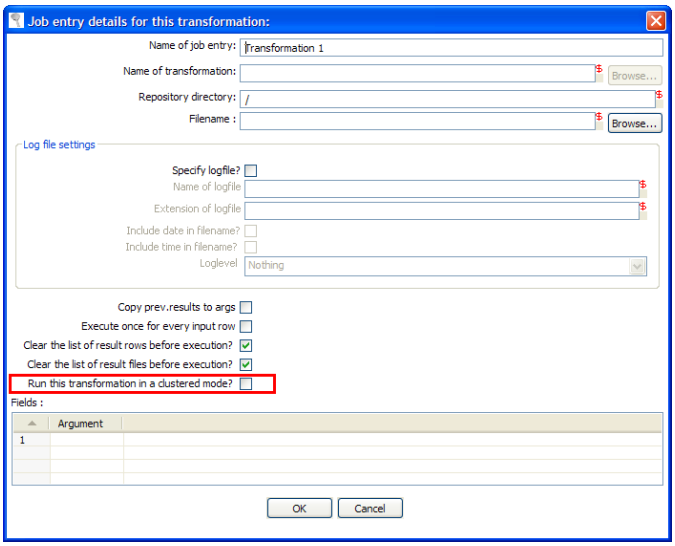


5. Job entries

5.1. Changed job entries

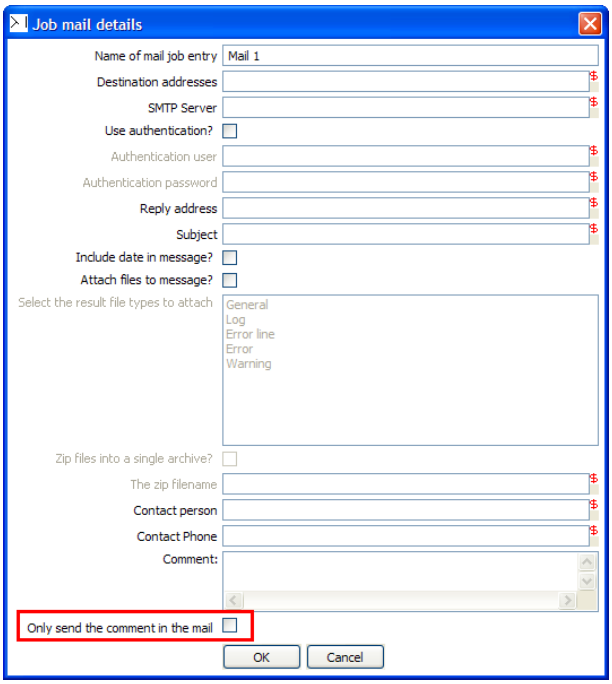
5.1.1. Transformation

We added the ability to execute a transformation in a clustered fashion. The job entry waits in this case until the master and all involved slave servers have finished processing. This option also allows you to execute clustered transformations in batch using the Kitchen command line utility. (no other option is of yet available in Pan to do this).



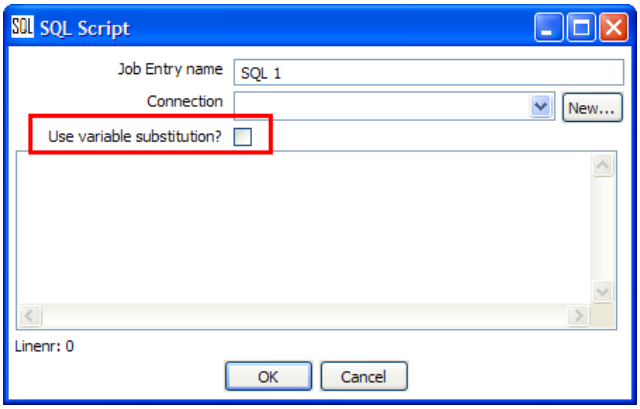
5.1.2. Mail

To make feedback mails easier to read we added the option to only send comments in the mail:



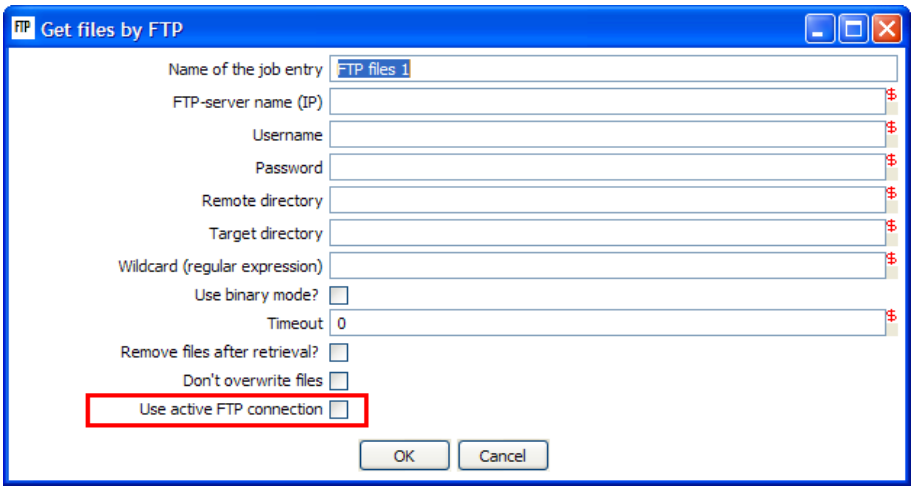
5.1.3. SQL

The SQL script can now also do variable substitution:



5.1.4. FTP

It is now possible to specify the use of an active FTP connection:



6. Source code improvements

6.1. A few extra lines of code

Version 2.1.4 contains 160,000 lines of code.

Version 2.2.2 contains 177,450 lines of code, an increase of 17,450 lines.

Version 2.3.0 contains 213,489 lines of code, an increase of 36,039 lines.

Version 2.4.0 contains 256,030 lines of code, an increase of 42,541 lines.

6.2. Trackers

In the period between the release of version 2.3.0 on June 27th 2006 until January 25th, 2007 we fixed 262 bugs and 51 feature requests where implemented.

6.3. Committers

ID / Commits	e-Mail	Name	Country	Work
sboden 316	Svenboden (at) hotmail.com	Sven Boden	B	First rate bug fixing, translations, unit tests, implemented many a change request and was a big help out on the forum. Sven was all over the place :-)
berarma 167	Bernardo (at) tsolucio.com	Arlandis Bernardo	ES	He and his co-workers translated the software AND manuals in Spanish, bug reporting & many code changes.
jbleuel 118	Jens.bleuel (at) proratio.de	Jens Bleuel	D	Various bug reports, fixes, German translations and i18n fixes, big help out on the forum.
qinhui99 18	Qinhui99 (at) hotmail.com	Tom Qin	CH	Translation into Simplified Chinese Various bug fixes
Biswapesh 20	Biswapesh (at) gmail.com	Biswapesh Chattopadhyay	UK	Profiling, sort improvements, stream lookup stress testing, memory consumption analysis, merge join step & many suggestions, bug reports, etc.
Mjansen 3	Mjansen (a) betterbe.com	Michel Jansen	NL	Wrote & maintains paper: “Building data warehouses using open source technologies”
Jdixon 27	Jdixon (a) pentaho.org	James 'Chief Geek' Dixon	US	Added support for JNDI database connections, added checks for nulls in quoting algorithms.
Sven.thiergen 12	s.thiergen (a) itcampus.de	Sven Thiergen	D	Various bug fixes, 2 new sample transformations, German i18n fixes and Props sorting “fun bucket” item.
Wconroy 6	wconroy (at) gmail.com	Bo Conroy	US	FTP/enterprisedt upgrade, console logging improvements
MattCasters	Mcasters (at) pentaho.org	Matt Casters	B	The rest
TOTAL 4,000	***10 developers***		***7 countries***	

6.4. Bug reporters

ACE., ackdesha, asioli, audinchan, begunrom, berarma, bicycler, bmmadsen, btalbot, ChrisPerrin, clavigne, cpitis, dawe, dhartford, dmoran, eddietam, equake, etdube, fjsfjsfjs, -furet-, gbacskai, gbottazzi, giyoram, guido.leenders (at) invantive.com, guzaldon, HawkNewton, hazmatt, hoezx6r, huangfan575, illum_user, itzik, JasonToyne, jbleuel, jdixon, jgargus, Jgunderson, jjkarppi, joebordes, JonathonC, Jonhoni, joson, JuliaNiuNiu, kandrews, lmeader, marc_swingler, MattCasters, ndefontenay, ngoodman (at) bayontechnologies.com, nricheton, phancox, plissak, pstnotpd, radek, rchristy, RPBouman, rsheldon, RutgerDOW, saratchev, sayuso, sboden, scesbron, seckendorff, shassan2, stemey, sven.thiergen, tardifma, ThePerchik, Tihben, TimPigden, tom.gleeson (at) ireland.com, vishalsaha, wceuppens, wconroy, zverina and of-course all the others that reported bugs as *anonymous.*

Thank you all, 262 times!

6.5. Feature requesters

Avitous, btalbot, Bumbo, gbacskai, guido.leenders (at) invantive.com, hazmatt, illum_user, jbleuel, jhaile, kandrews, lmeader, m_sharp, marc_swingler, MattCasters, ngoodman (at) bayontechnologies.com, RoelVerbeeck, RPBouman, saratchev, sboden, shassan2, surenm, wconroy and again *anonymous.*

Thank you all 51 times for the many good suggestions!

6.6. Other contributors

Disclaimer: *The list below is by no means complete. Many people files bug fixes and feature requests anonymously. Others were probably shamelessly bypassed. Please remember that this is not done on purpose but rather a result of the limited time we can spend on these type of documents. If you were one of the victims of this horrible policy, simply contact the maintainer of this document and you will be added immediately. The same is true for everyone that DOESN'T want to be on this list: just let us know.*

- The entire crew at Pentaho for the many suggestions for improvements, bug reports and continued support of the entire Kettle project. Special thanks goes to Gretchen Moran for her successful Kettle Forum migration & the fantastic new forum software.
- Special thanks to Martin Lange from the company Proconis (http://www.proconis.de) for writing and donating the the new Javascript (Mod) step.
- One of the big advances we made in terms of massive parallel processing and partitioning logic couldn't have been made if it wasn't without the help of Google's Biswapesh Chattopadhyay. Biswa had lots of time constraints but he still managed to an awful lot of testing and code writing for which it is difficult to thank him enough.
- Special thanks to Youssef Mrabet for writing and donating the Streaming XML input step.
- Special thanks to Michel Jansen from the company Better.be (http://www.betterbe.com) for writing & maintaining the paper: “Building data warehouses using open source technologies”
- Special thanks goes to the entire team behind Bernardo Arlandis at Tsolucio (http://www.tsolucio.com)
- Very special thanks go to Samatar Hassan (shassan2) for his enthusiasm, the long list of bugs and change requests filed, continued support and obviously his French i18n efforts.
- Special thanks to the JBoss development team, especially Peter Van Weert (Peter.VanWeert (at) cs.kuleuven.be) and Mark Proctor (mproctor (at) codehaus.org) for donating their implementation of a HashIndex that uses a lot less memory and works faster than the standard Java Hashtable or Hashmap classes. It has dramatically reduced memory consumption for the “Stream Lookup” step.
- Herbert Laroca (herbert.laroca (at) gmail.com) for translating into Brazilian Portuguese (pt_BR)
- BreadBoard BI (http://www.breadboardbi.com) wrote a nice paper on ETL with Pentaho: http://www.breadboardbi.com/white_papers/pentaho_etl_whitepaper.pdf
- Shibu Mohapatra (shibu_x (at) yahoo.com) & colleagues for the many suggestions.
- Roel VanEck (Roel.VanEck (at) iex.com) for sending patches to improve job entry plugin support, “Group By step” patch and more.
- ... everyone that provided feedback, sent transformations and samples.