Release notes for version 3.7.2

Important! Create a backup copy of your projects before updating to the new version. Projects saved in the new version can't be opened in versions earlier than 3.7.

Breaking changes

Evaluation of expressions without field references on empty tables

Transformations "Halt" and "Derive table" can have conditions defined as expressions. The expressions may or may not contain references to columns and parameters. The conditions also have a selector that prescribes what to do if the input dataset is empty (i.e. columns don't contain data) – either consider the condition failed, or not. However, this prescription was also used when the condition didn't include columns but included parameters, which wasn't correct.

Example
Transformation is: "Halt"
Expression is: `{Future date} - today()  < 10`
Selector "When table is empty" set to Fail.
The input dataset is empty.

Old behavior	New behavior
If dataset has no rows then expression result is always ignored, and condition result is defined by selector "When table is empty".	If dataset has no rows then expression result is ignored only when the expression contained a reference to column. If it didn't (for instance, the expression uses parameters, not columns), then the condition result is defined by expression evaluation. If the expression contains at least one column reference, on an empty table the condition result is defined by selector "When table is empty"
Result: Condition always fails.	Result: Condition fails when parameter {Future date} is less than 10 days ahead of today, or in the past.

Missing column leads to failure even on empty datasets

Previously, some transformations (e.g. filters) were automatically skipped if the input dataset had no rows. Starting from this release, if a transformation has a reference to a column and the column is missing in the input dataset it will always fail project execution with error "Missing column", even if the input dataset is empty and transformation result would've been the same.

Transformations affected by the change:

- Clean up
- Convert Data Type

- Deduplicate
- Fill Down
- Filter
- Filter by condition
- Filter by search
- Iterate
- Iterate program
- Iterate table
- Keep duplicates
- Pivot
- Replace
- Sort
- Split delimited text into rows
- Table-wide replace
- Trim by condition

Example
Transformation is: "Filter by expression"
Expression is: `[Amount] > 10000`
The input dataset has no rows and has no column "Amount".

Old behavior	New behavior
Result: Transformation successfully calculated. Data hasn't changed.	Result: Transformation failed with error "Column [Amount] is missing.

Bugfix: weekday in Calendar now starts from 1

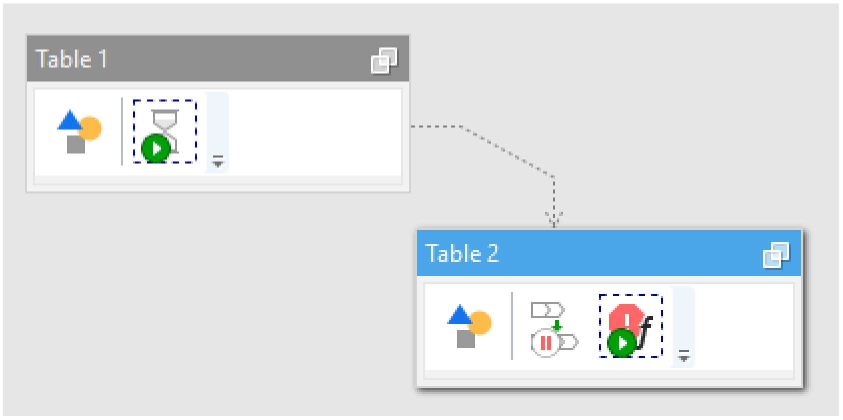
The weekday numbers generated by transformation "Calendar" were inconsistent with function **weekday()**. This has been fixed, but the bug fix is a breaking change. You may need to insert a "Modify column" transformation with expression `[Day of week (number)] - 1` right after the "Calendar" transformation to adapt existing projects to this breaking change.

Example
Transformation is: "Calendar".
Option "Day of week (number)" is checked.

Old behavior	New behavior
Result: Weekdays range from 0 (Sunday) to 6 (Saturday).	Result: Weekdays range from **1** (Sunday) to **7** (Saturday).

Bugfix: Disabled 2-input transformation now doesn't wait for the other table

Previously, when a 2-input transformation (e.g. Synchronize) was disabled it would still wait for the other table to become calculated. This behavior has changed: now disabled transformations are ignored completely during calculations, as if they don't exist. The behavior change is subtle, although it may affect the order in which transformations are executed.

Old behavior	New behavior
Example The “Wait” transformation is set for 10 seconds. Ctrl + click the “Halt” transformation to run it.  	
Result: Running the “Halt” transformation causes executing the “Wait” transformation first.	Result: The “Halt” transformation is executed **immediately**.

What’s new

New transformations

The **Filter by type** transformation keeps/removes rows with values of particular data type in one or more columns. Data types are: Text, Number, Boolean, Error, and Empty (null). The primary use case for the transformation is detection of wrongly typed values (e.g. Text instead of Number) before exporting to a database or Tableau extract. It can also be used together with “Halt” to assert data consistency.

The **Qlik Sense Command** transformation is intended for remote execution of different actions with Qlik Sense using public Qlik Sense APIs. Currently, the transformation has only one command: “Trigger QMC task” that starts specified QMC task. To simplify debugging, in case of task failure the transformation parses Qlik Sense logs and extracts error messages from them, if any.

Also, together with the “EasyMorph-Action” extension for Qlik Sense this transformation enables user-initiated, parameterized, selection-dependent Qlik Sense apps reloads right from Qlik Sense.

The **EasyMorph Server Command** transformation remotely executes one of these 4 actions with specified EasyMorph Server:

- Trigger task. Task parameters can be overridden in the transformation properties.
- Upload local file to EasyMorph Server
- Download remote file from EasyMorph Server into a local folder
- Delete remote file on EasyMorph Server

The transformation allows offsetting heavy CPU-intense and RAM-consuming computations from a locally executed project to a remote EasyMorph Server. For instance, a desktop EasyMorph project can copy a set of files to EasyMorph Server, trigger a Server task to process the files, and later download the results back to the local computer when they are ready. The transformation intelligently handles errors that can occur while executing a Server task – such errors are shown in the local project as well.

Another use case would be using the Server as a file transfer proxy, that fetches data from external systems in a centralized fashion, and then the data is transferred on demand to desktop users. It can work the other way around – the Server can serve as a distribution/sharing hub which is supplied with data from desktop EasyMorph users.

From a technical standpoint, file transfer is performed using HTTP over REST API. No FTP setup required.

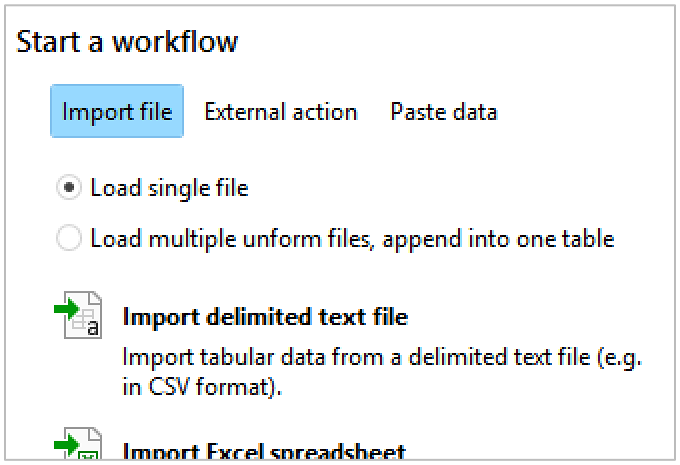
New connectors

The **Qlik Sense Connector** is used in the "Qlik Sense Command" transformation.

The **EasyMorph Server Connector** is used in the "EasyMorph Server Command" as well as for publishing projects to EasyMorph Server.

Multiple file loading from the Start screen

Now you can choose on the Start screen whether you want to load only one file, or load and automatically append multiple uniform files.


Start a workflow

(● Load single file) ○ Load multiple unform files, append into one table

Import delimited text file
Import tabular data from a delimited text file (e.g. in CSV format).

Import Excel spreadsheet

Screenshot 1: Selector single/multiple file load.

Miscellaneous

- The "Split delimited text into rows" now supports the line break separator.
- When selecting a connector in transformation properties, if only one connector of necessary type is available it is now selected automatically.
- Annotations for connectors are now shown in the transformation properties under the connector name.
- The parameterized text editor used in transformations such "Run Program" or "PowerShell" now has a button for easy insertion of path to an arbitrary file/folder.

Release notes for version 3.7.1

Important! Create a backup copy of your projects before updating to the new version. Projects saved in the new version can't be opened in versions earlier than 3.7.

What's new

New transformations

The **Send email** transformation sends an email with an attachment (optionally) up to 20MB via the SMTP protocol. The receiver, subject, message text, and attachment file can be specified using parameters. HTML formatting is supported as well (in Pro/Plus editions only). Use cases: rule-based notifications, data quality exceptions, and per-user data publishing.

The **Tableau Server Command** transformation triggers extract refreshes for workbooks and data sources on a remote Tableau Server. It can be used for updating dependent data sources and workbooks after an extract was published to Tableau Server, or a database data source was updated.

The **Metadata** transformation returns table metadata: column names, the total number of rows, and the total number of columns. The transformation can be convenient when using the "Rename by lookup" or "Select by lookup" transformations.

Changes to existing transformations

Changes in the **File Command** transformation:

- The new "Clone" mode that creates a copy of an existing file.
- The "Unzip" command now has action options for cases when a file already exists.

The **Convert Data Types** transformation now has a new mode: "Everything to Text". It helps quickly convert numbers and dates into their text representation.

The **Export to Delimited Text File** transformation now has an option for selecting a decimal separator (point or comma).

Connectors

The new **Email Server** connector specifies properties of a connection to an SMTP server. It's used in the "Send Email" transformation described above.

The **ODBC** connector has an updated look with more convenient schema management.

The Folder Path parameter type

A new parameter type, “Folder path”, is similar to “File path” but is intended for specifying folder locations.

User interface

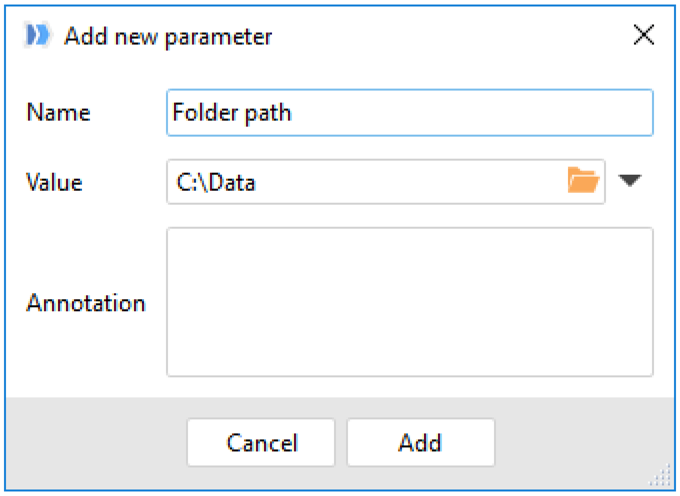
Changed Start Screen

The Start Screen has changed and now has 3 sections for creating workflows:

- File import transformations. This remains unchanged compared to previous versions.
- *Actions*, such as file downloading or running an external application.
- Pasting data. This can be done with specifying a decimal delimiter and enabling column headers.

Add Parameter dialog

Creating new parameters has become easier with the addition of the new Add Parameter dialog opened from a list of parameters in transformation properties. The dialog conveniently suggests a parameter name depending on the transformation property it will be used for.



Name: Folder path
Value: C:\Data
Annotation:

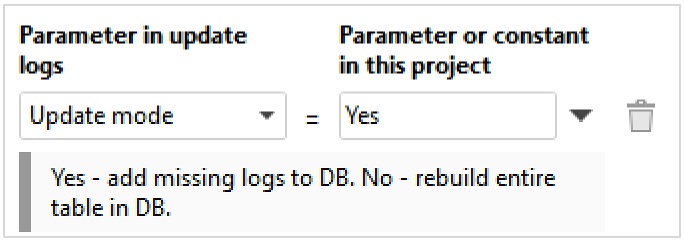
Screenshot 2: Parameter creation from a transformation property.

Advanced selectors

Starting with this release, a new two-column selector is displayed when selecting a project table, parameter, or connector. The selector allows searching and filtering that simplifies dealing with large projects.

Parameter annotation in transformations

Now transformations that assign parameters (i.e. Call / Iterate / Iterate Table) also display annotations of assigned parameters.



Parameter in update logs = **Parameter or constant in this project**

Update mode = Yes

Yes - add missing logs to DB. No - rebuild entire table in DB.

Screenshot 3: Parameter annotations in transformation settings.

Release notes for version 3.7

Important! Create a backup copy of your projects before updating to the new version. Projects saved in the new version can't be opened in earlier versions.

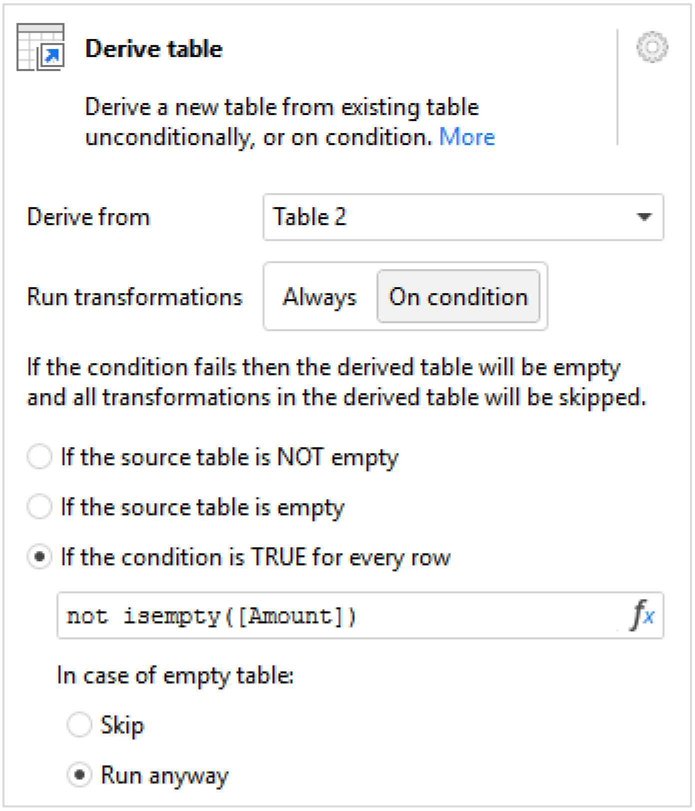
What's new

Conditionally derived tables

Previously, arranging IF…THEN…ELSE workflows required using various, frequently obscure, workarounds. Starting from this release, arranging conditional workflows has become simple and straight forward because derived tables in EasyMorph can now work in two modes:

Unconditional – this is the usual, regular mode known from the very first version of EasyMorph. In this mode a derived table obtains its dataset from the source table, and all transformations in the derived table are executed as usually.

Conditional – this is a new mode in which transformations in a derived table are executed **only** when a certain condition is satisfied. If the condition is not met, then all transformations are quietly skipped and the resulting dataset is always empty.


Derive table

Derive a new table from existing table unconditionally, or on condition. More

Derive from: Table 2

Run transformations: Always [✓ On condition]

If the condition fails then the derived table will be empty and all transformations in the derived table will be skipped.

- ○ If the source table is NOT empty
- ○ If the source table is empty
- ● If the condition is TRUE for every row

```
not isempty([Amount])
```

In case of empty table:

- ○ Skip
- ● Run anyway

Screenshot 4: Conditionally derived table.

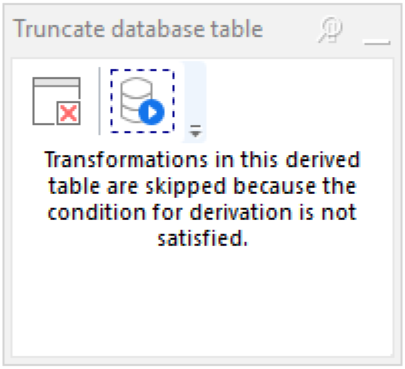
Possible conditions are similar to the ones used in the "Halt on condition" transformation, namely:

- When table is empty
- When table is NOT empty
- When an expression is TRUE for each row in table

To arrange an IF ... THEN workflow create a derived table with condition.

To arrange an IF ... THEN ... ELSE workflow create two derived tables with conditions that are inverse to each other.

The transformation icon indicates when derivation condition is not satisfied and all transformations are skipped.

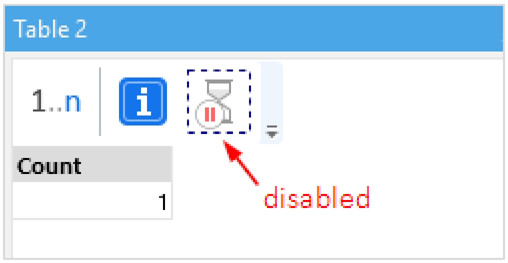

Truncate database table

Transformations in this derived table are skipped because the condition for derivation is not satisfied.

Screenshot 5: Skipped derived table.

Disable/enable transformations

Now it is possible to temporarily disable a transformation. A disabled transformation becomes "transparent" – it performs no action and simply passes through its input dataset to the next transformation. Disabled transformations are marked with a special overlay icon.



Count
1

Screenshot 6: Disabled transformation.

To disable (enable) a transformation right-click it and choose Disable (Enable).

New transformations

The **Replace** transformation allows replacing particular values in a column with new ones. It can be used for instance to replace mistyped values with correct ones.

The **File Command** transformation is a convenient way for simple file operations. Currently it has 5 commands:

- Copy/Move File
- Rename/Clone File
- Delete File
- Delete All Files in Folder
- Unzip File

The **PowerShell** transformation executes one or more PowerShell commands. It requires PowerShell version 3 or above installed. A column can be used to provide an input collection of values accessible through $input from within the PowerShell script. The output collection of values can be captured back into EasyMorph, as well as errors. This transformation is not available in the free edition.

The **XSLT** transformation modifies XML files using the Extensible Stylesheet Language Transformations (XSLT) language. It's a powerful language that allows extracting data from XML files, inserting new nodes, amending XML elements and attributes. For instance, file paths in Tableau workbooks and EasyMorph projects can be automatically modified using this transformation.

Changes to existing transformations

The **Export to Tableau** transformation now uses a connector to Tableau Server from a repository (or embedded). Also Tableau projects and data source names can be picked from a list, or defined using a parameter.

The **Iterate** transformation now has a new mode that allows continuing execution even if one or more iterations failed. In this mode, a new column "Iteration status" is created. The column contains errors of iterations, if any. This mode together with conditionally derived tables can be used for arranging failover logic.

The **Export to Excel** transformation now allows using a parameter to specify the target sheet name.

If no working directory defined in the **Run Program** transformation, it's now implicitly assumed that the working directory is the directory of the project.

It is now possible to change the order of rules in the **Rule** transformation.

New functions

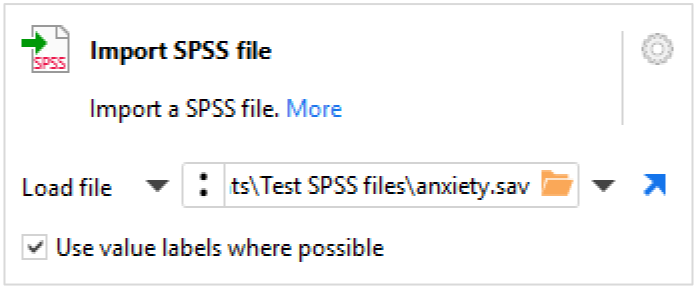
ispathrooted(path_text) – detects if a path to file or folder contains a root. Example `ispathrooted('C:\Documents\myproject.morph')` returns TRUE.

Non-database connectors

From now on, non-database connectors can be created and stored in a repository, or embedded in a project. In this release new connector types are Tableau Server, EasyMorph Server, and 1010data. Future releases will introduce new connector types for various applications and cloud services.

Support for SPSS/PSPP files

The “Import SPSS file” transformation imports one or multiple .sav files created in SPSS or PSPP. Column values can be automatically replaced with value labels, if necessary.


Import SPSS file

Import a SPSS file. More

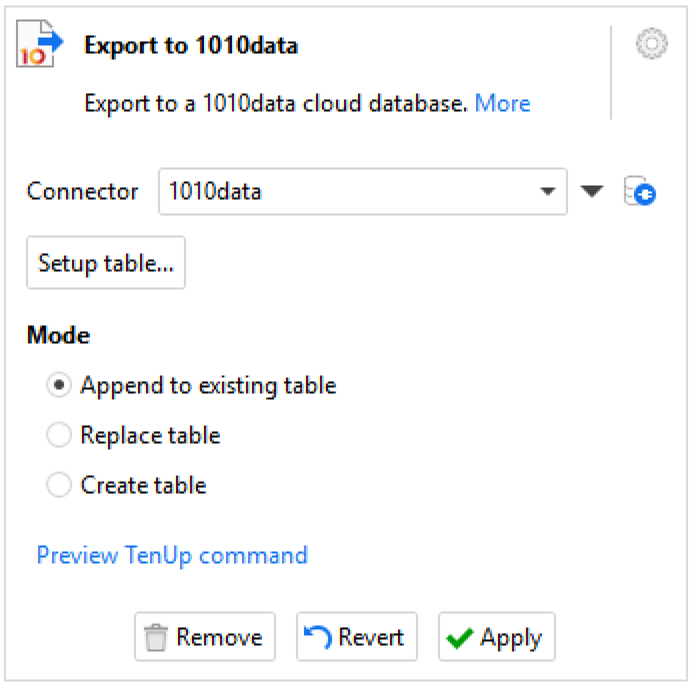
Load file: ts\Test SPSS files\anxiety.sav

✓ Use value labels where possible

Screenshot 7: Import SPSS file with value labels.

Export to 1010data (experimental)

We’re adding the ability to export from EasyMorph to 1010data cloud analytical database using the “Export to 1010data” transformation. The export is done using the TenUp utility which is included into the EasyMorph installation package. The generated command line for TenUp can be previewed as well as the contents of tree file.


Export to 1010data

Export to a 1010data cloud database. More

Connector: 1010data

Setup table...

Mode

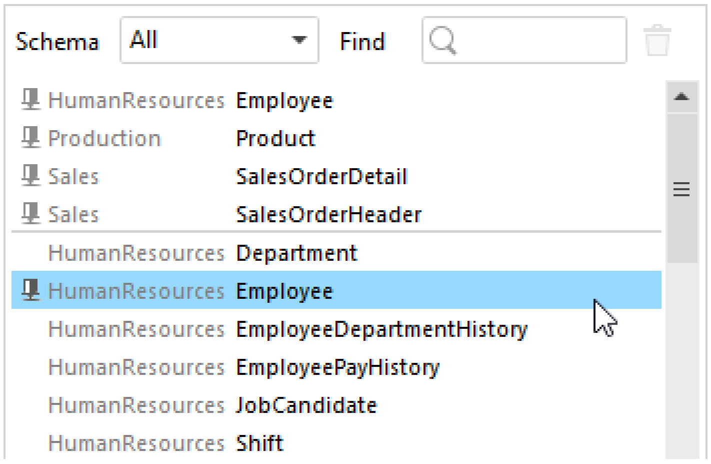
- [✓ Append to existing table]
- Replace table
- Create table

Preview TenUp command

Screenshot 8: Export to 1010data.

New database table picker

This release introduces a new database table picker. It's used in queries and transformations where a database table should be picked from list. The picker is capable of working smoothly with tens of thousands table names and hundreds of schemas. Frequently used tables can be "pinned" to the top of list for convenience.

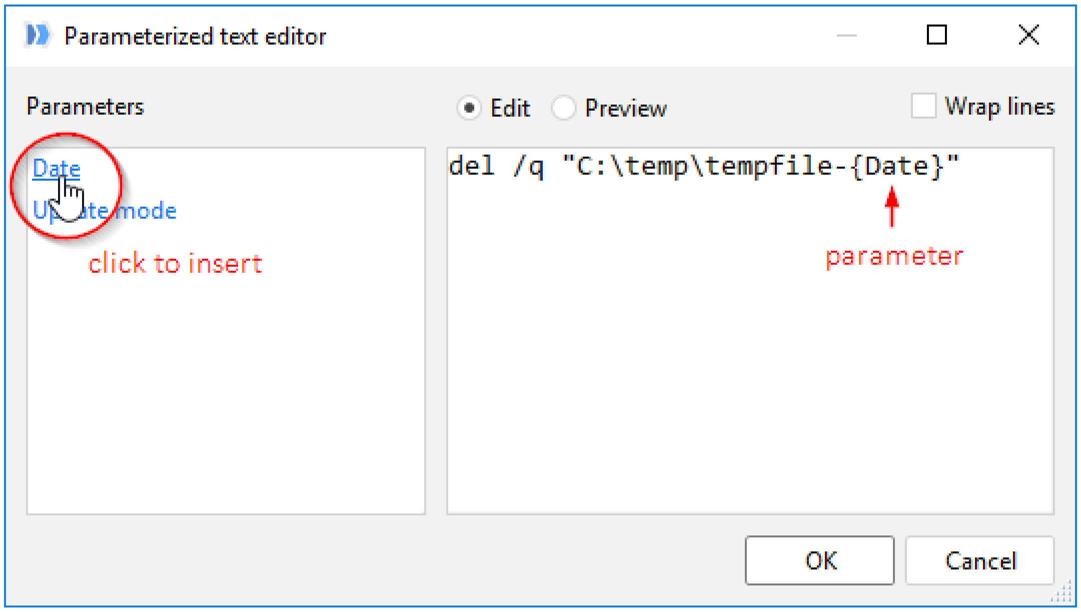

Schema: All   Find

HumanResources	Employee
Production	Product
Sales	SalesOrderDetail
Sales	SalesOrderHeader
HumanResources	Department
HumanResources	Employee
HumanResources	EmployeeDepartmentHistory
HumanResources	EmployeePayHistory
HumanResources	JobCandidate
HumanResources	Shift

Screenshot 9: New database table picker.

Multi-line editor for “Run Program” and “PowerShell”

The transformations got a “Pop-up” button that opens a multi-line editor for more convenient command editing. The editor has convenient means for inserting project parameters into text – just click a parameter name on the left and it will be inserted at the cursor position in the text. You can switch to the preview mode to see what the text would look like after parameter substitution.

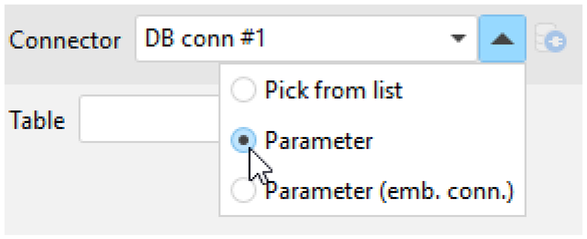


```
del /q "C:\temp\tempfile-{Date}"
```

Screenshot 10: Editor with parameter insertion.

Connectors specified by parameters

Starting from this release it is possible to specify connector using a parameter in all transformations that use a connector, as well as in the Query Editor. A parameter can be used to specify an embedded connector as well.



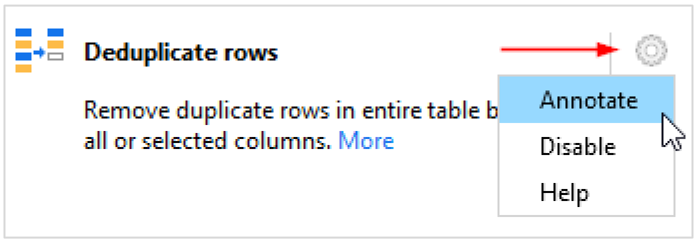
Screenshot 11: Connector specifies by a parameter.

Block indentation in text editor

EasyMorph now uses a new, improved text editor for editing expressions, custom SQL queries and commands, and other multi-line texts in the application. The editor allows performing block indentation (select multiple lines and press Tab or Shift+Tab).

Transformation settings menu

Creating annotations is moved into the transformation settings menu (the gear icon next to transformation name and description). The menu also contains commands for enabling/disabling transformations and a link to the web-help for the transformation.


Deduplicate rows
Remove duplicate rows in entire table b
all or selected columns. More
Annotate
Disable
Help

Screenshot 12: Transformation settings menu.

Miscellaneous

- Drag a folder with multiple files into EasyMorph to load all these files at once.
- If a called project contains only 1 table it's implicitly considered the default result table for Iterate/Call transformations.
- Documentation can now be generated even for projects with transformations with invalid properties. Such transformations will be marked red.