Easy Data Transform v1.5.0 for Windows

Getting started

1 Getting started

1.1 Introduction

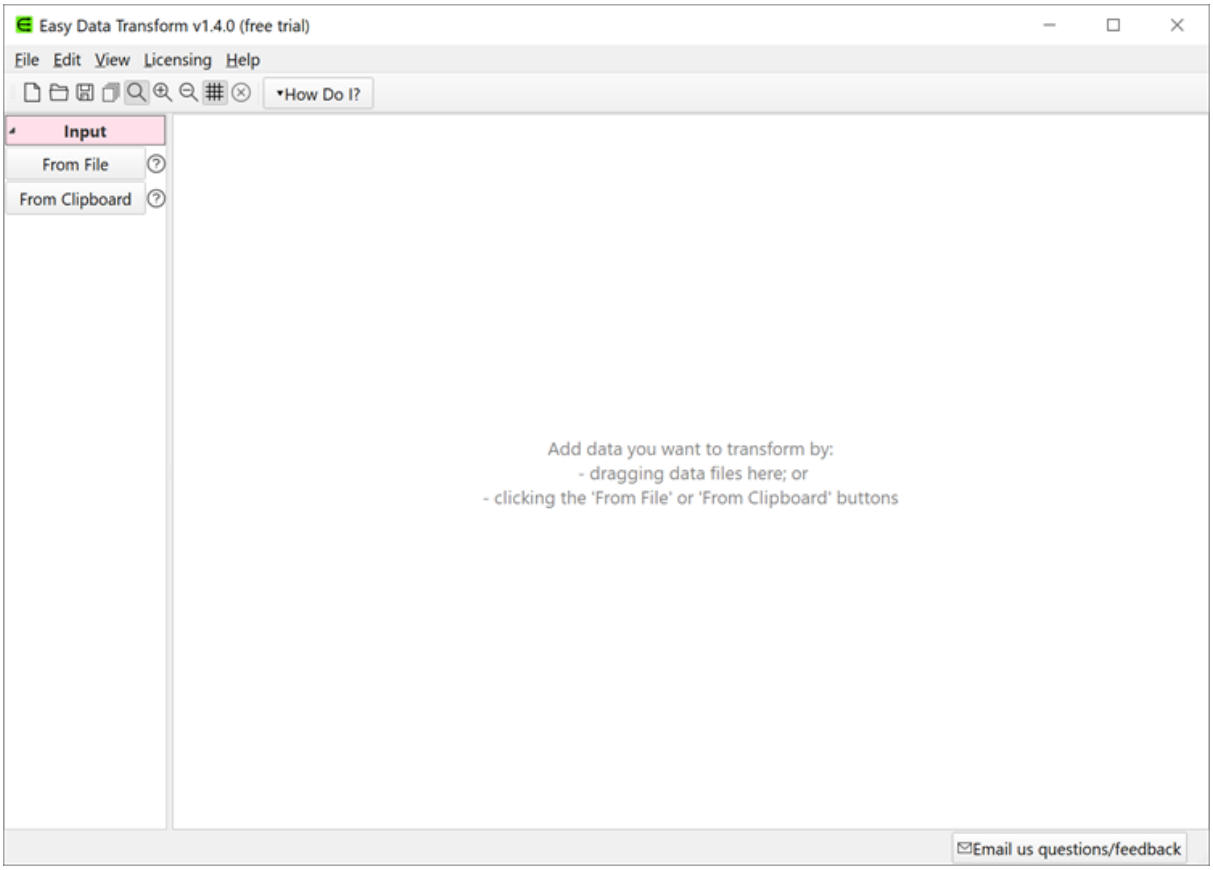
Easy Data Transform allows you to quickly transform table and list data into new and more useful forms, without programming. The step-by-step visual transformation is quicker, more interactive, more repeatable and less error prone than other approaches.

Please take a couple of minutes to read the Quick Start Guide.

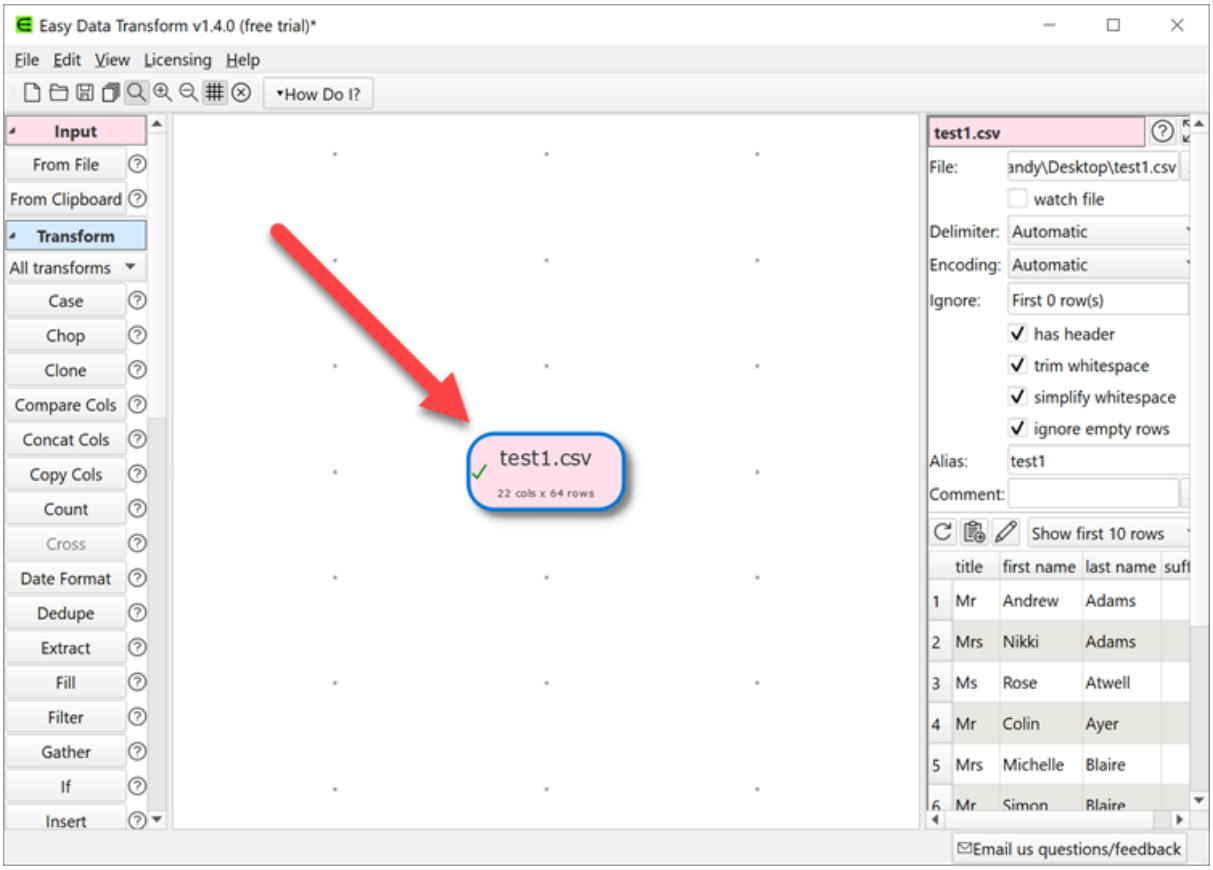
1.2 Quick start guide

This is a quick tour of some of Easy Data Transform's features. It should only take a couple of minutes to complete.

Start Easy Data Transform. If the **Free Trial** window appears, click **Continue free trial**. If the Getting Started window appears, click **I have used it before!** (or you will just end up back on this page). You should now see the main window.

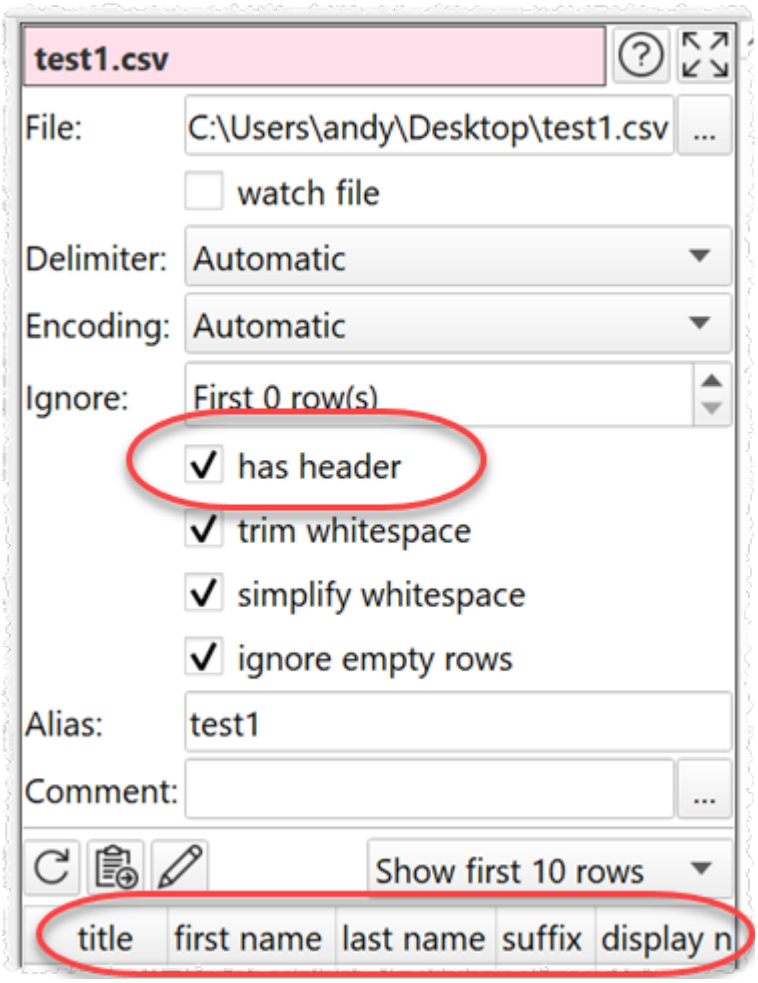


Drag a data file you want to transform onto Easy Data Transform. Any sort of table or list should be fine. For example a .csv file or an Excel .xlsx/.xls file.



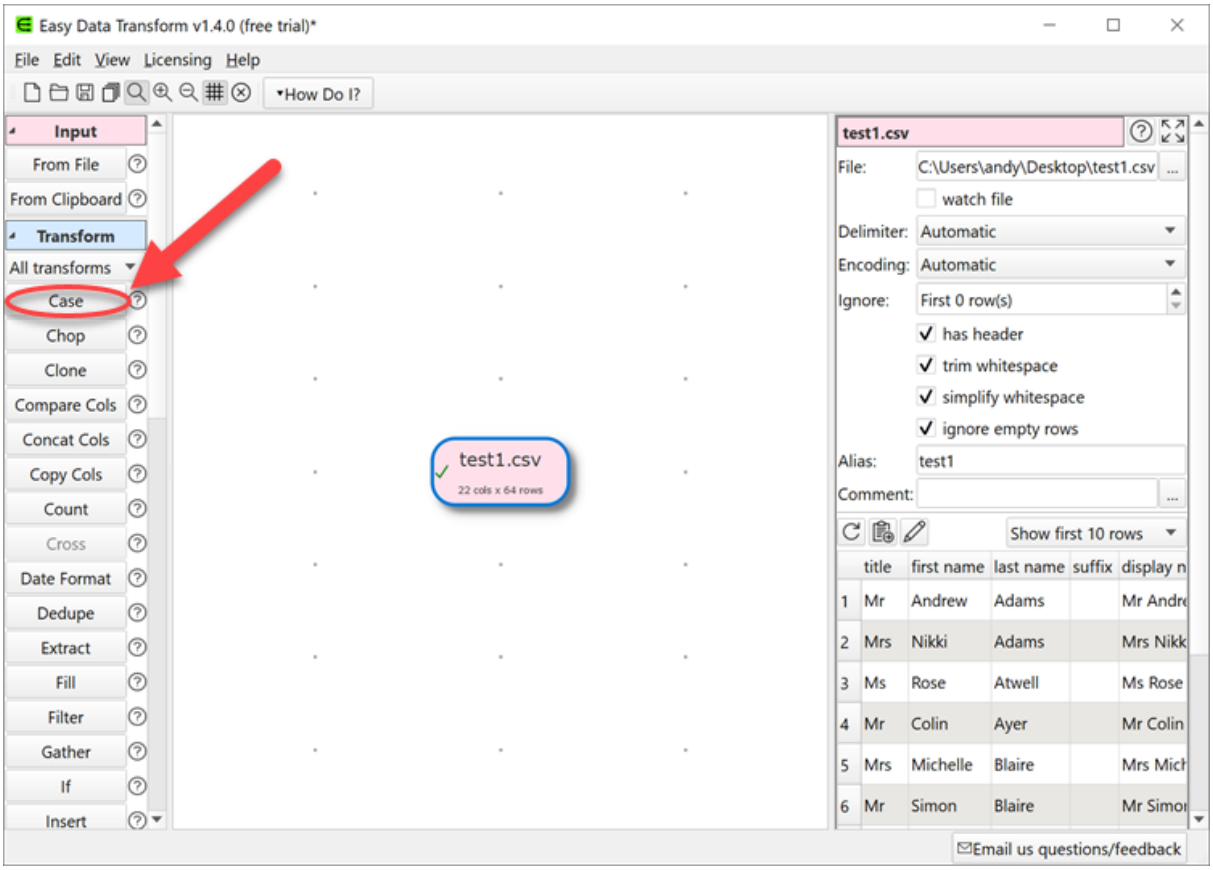
Notice that the available transforms are shown in the **Left** pane and the selected dataset is shown in the **Right** pane.

In the **Right** pane, you can check **has header**, depending on whether you want to treat the first row of the dataset as a header.

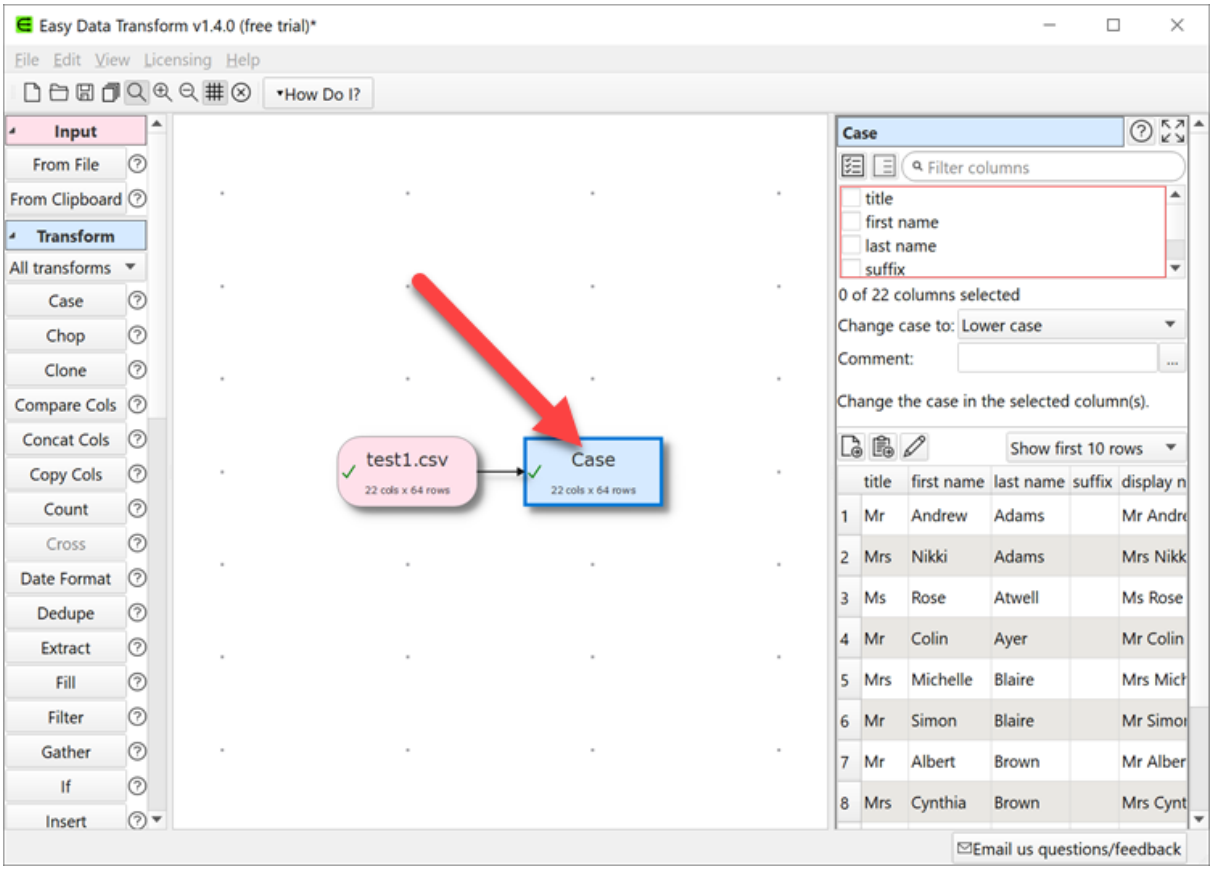


All the transforms available for a single dataset are now enabled in the **Left** pane. Hover over the transforms to see tooltips explaining what they do. Click on the **?** next to a transform button for more details.

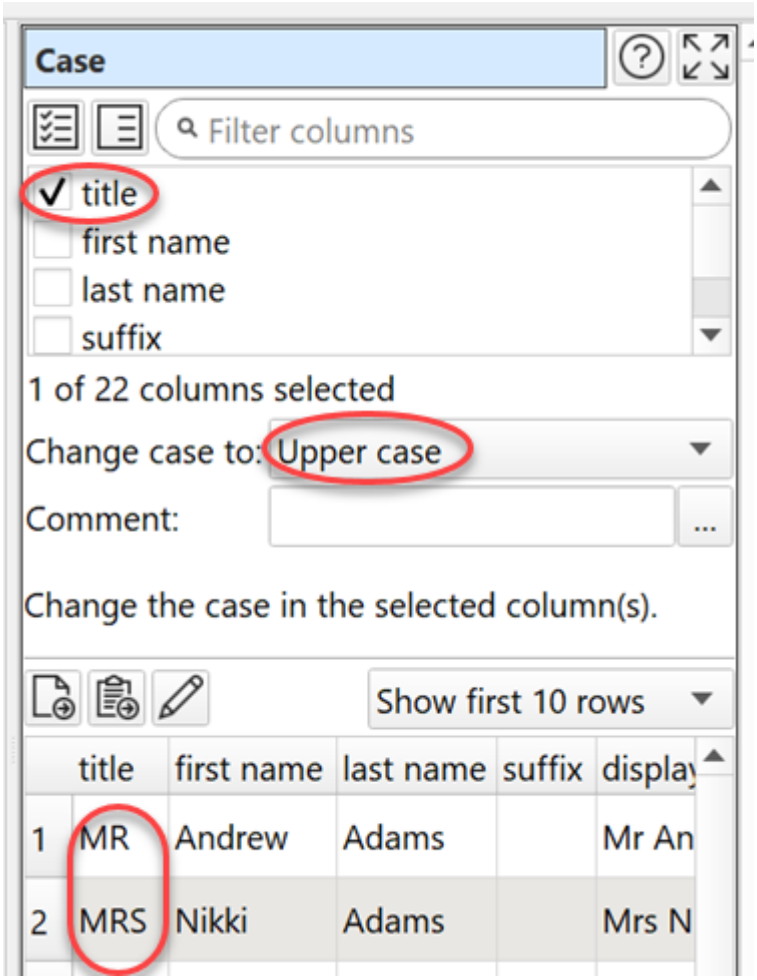
Ensure the input item is selected and click on the **Case** transform button to change the case of your data.



A **Case** transform item will now be added.



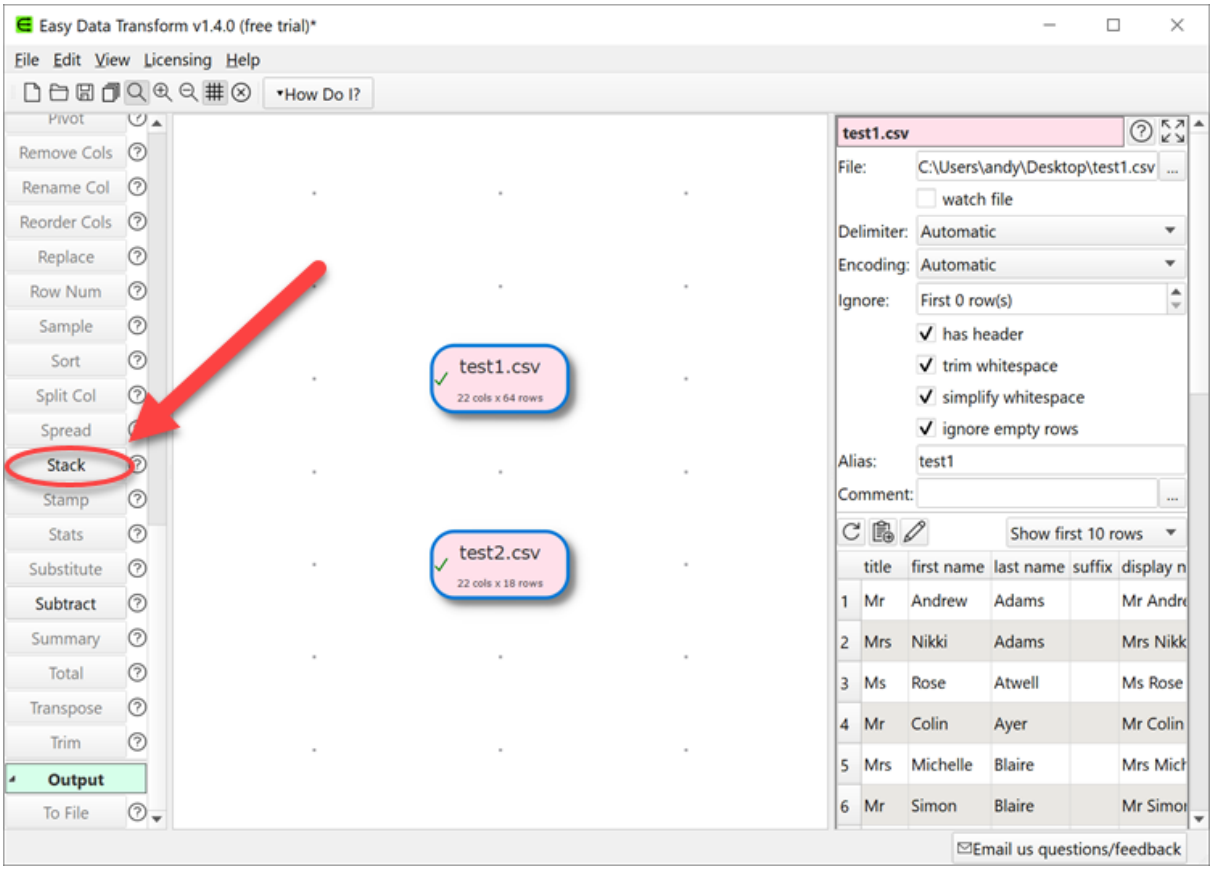
In the **Right** pane, check one of the columns and set **Change case to** to **Upper case**. All the text in that column will now be converted to upper case.



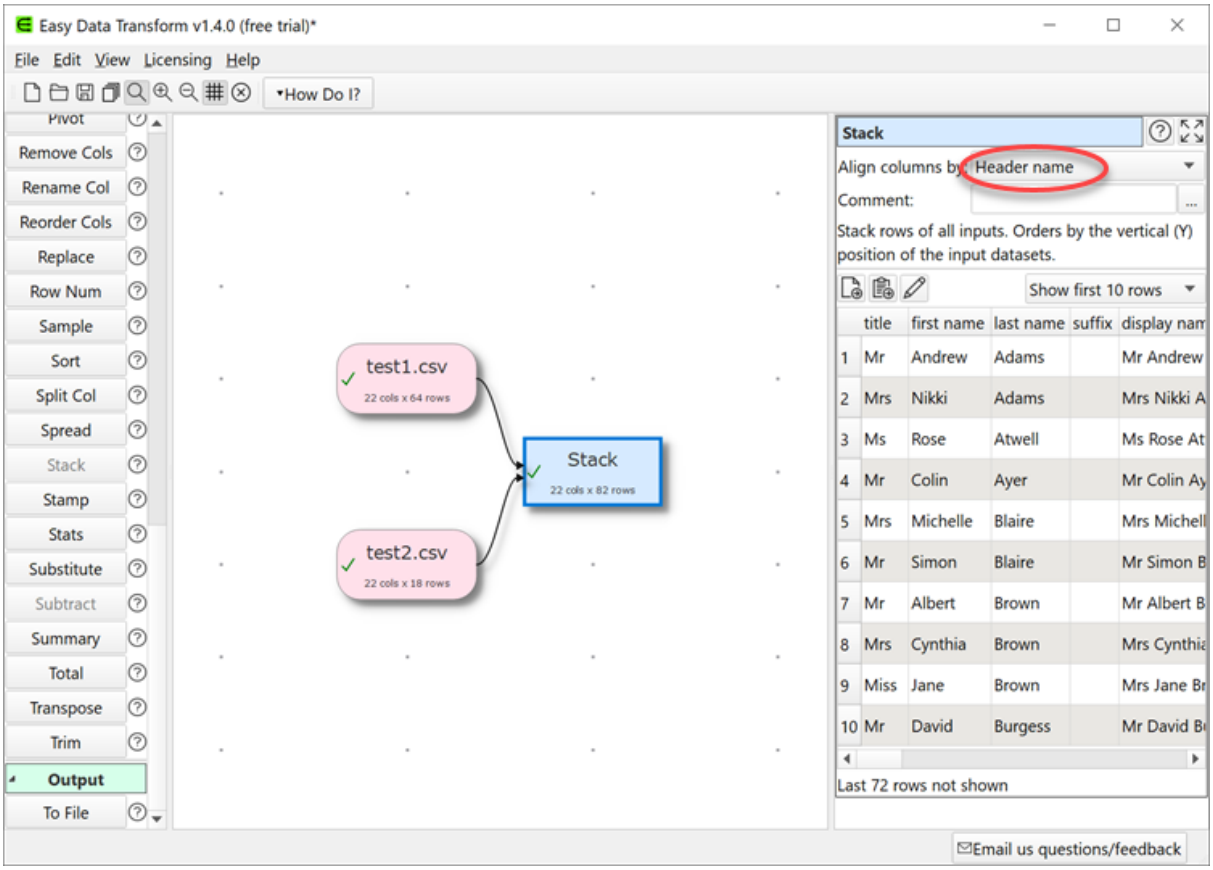
You can create a sequence of transforms to perform complex manipulations.

Some transforms require more than one input dataset. For example, to stack two tables, one on top of the other:

- Select **File>New** to start again. Don't save the changes.
- Drag two data files onto the **Center** pane.
- Select both input items (by dragging a box around them or using `Ctrl`+click).
- Click the **Stack** transform button (you may need to scroll the **Left** pane to see the button).



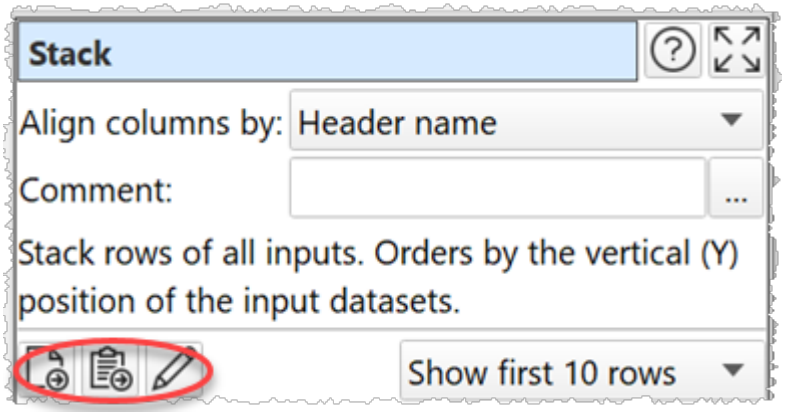
The tables are now stacked one on top of the other in a new dataset item. You can choose to match the columns by **Header name** or **Column number**.



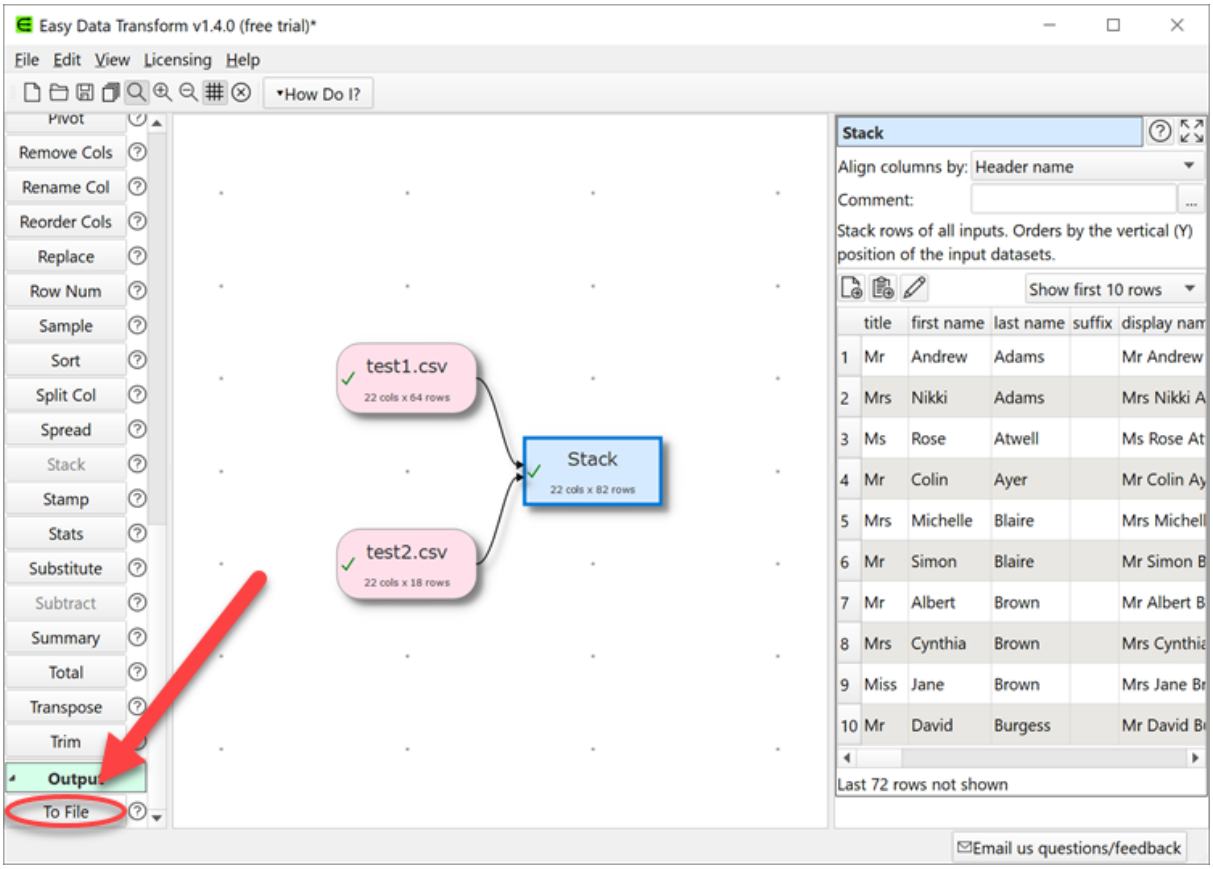
Note that the vertical (Y) position of the inputs affects the order the datasets are stacked. Try swapping the two inputs around and re-select **Stack** to see the affect.

Any changes to input files will be automatically read in. Any changes to input datasets or transform options will be automatically propagated 'downstream'.

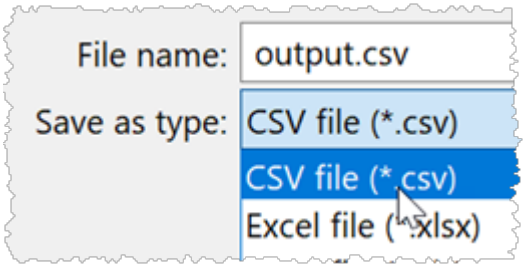
To export your transformed dataset to a file or the clipboard, or to view it in a local editor, select the dataset item and click on the appropriate button in the **Right** pane.



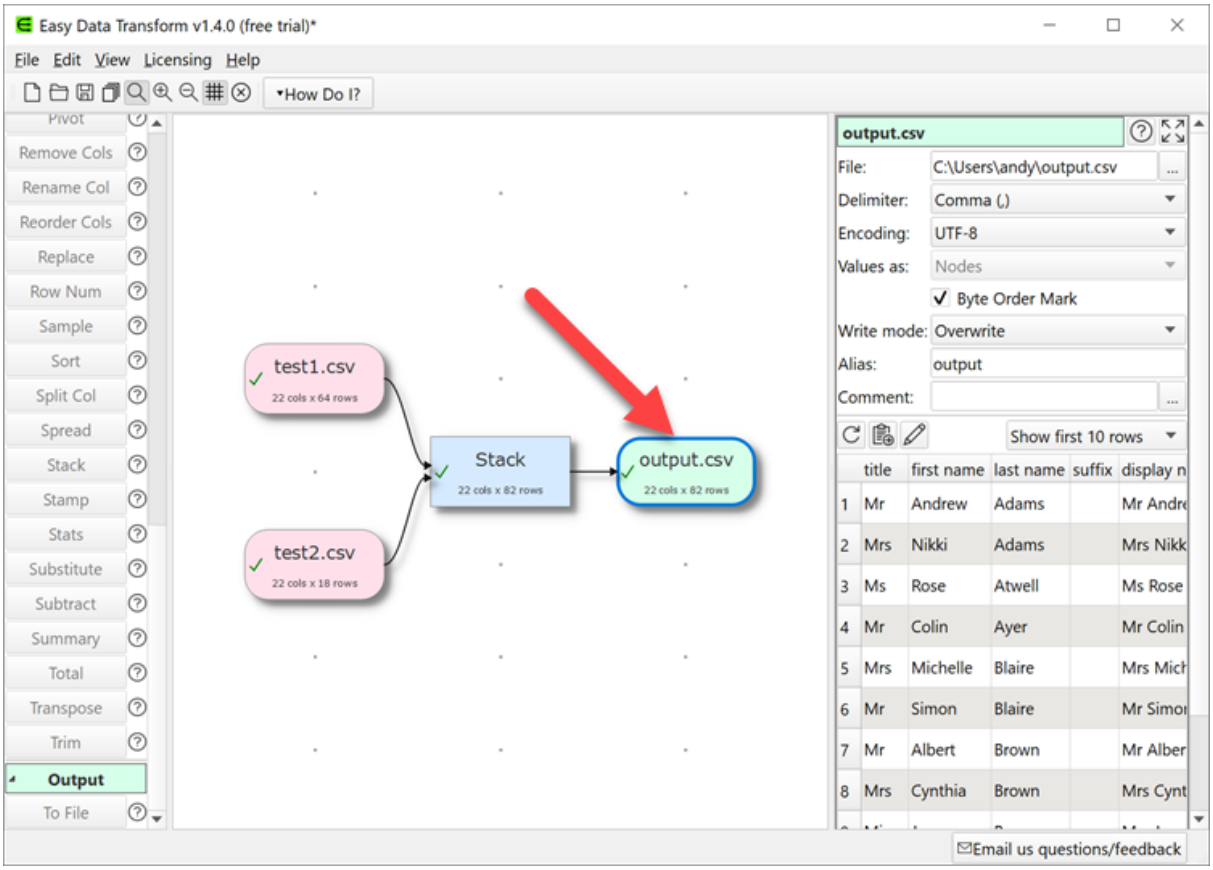
You can also add an output item to automatically write a dataset to file whenever it changes.



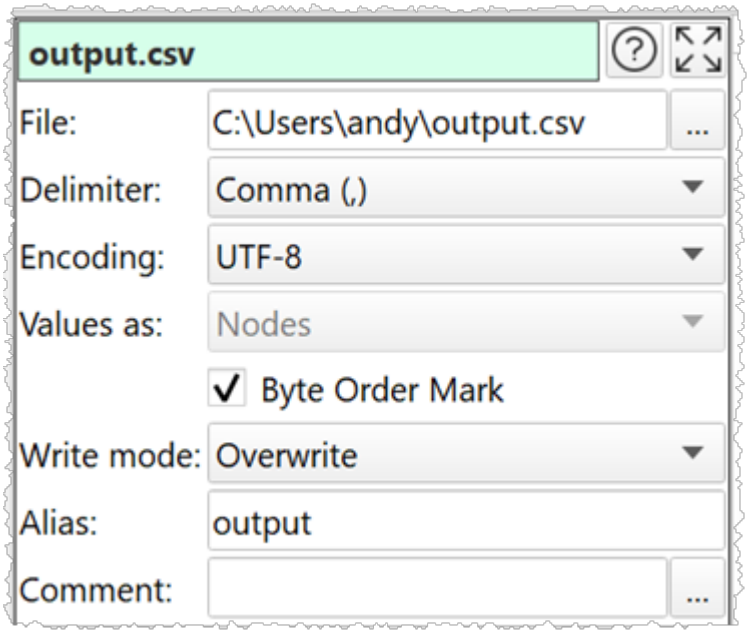
You will be asked for a file to write to. You can choose amongst multiple file formats. Select **CSV file**.



Your dataset will then be written to this file every time it changes.



You can also specify the delimiter and encoding for your CSV files in the **Right** pane.



You can save your transforms to a transform template document to use again with **File>Save**.

Have a play!

Tips:

- You can also paste in data from the clipboard (for example, a table from a web page or Word document).
- The **Compare cols**, **Filter**, **If** and **Sort** transforms take account of dates, numbers and text. You can define what date formats to recognize in the Preferences window.
- New columns are always added to the right of a table.
- Comparisons of text are always sensitive to case, unless stated otherwise. E.g. "CASE", "case" and "Case" are treated differently.
- Comparisons of text are always sensitive to whitespace (e.g. spaces and tabs), unless stated otherwise. You can use the **Trim** transform to remove leading and trailing whitespace.
- The contents of input and output data files are not saved in Easy Data Transform, only their locations.
- As well as stacking two datasets, you can also Join them, side-by-side, if they have a common ('key') column.
- You can insert a new transform between existing items by selecting the connection between the items and then adding the transform.
- You can perform the same set of transformation on multiple files using Batch processing or command line arguments.

We are interested in your feedback, so please contact us to ask a question, report a bug or request an enhancement.

Reference

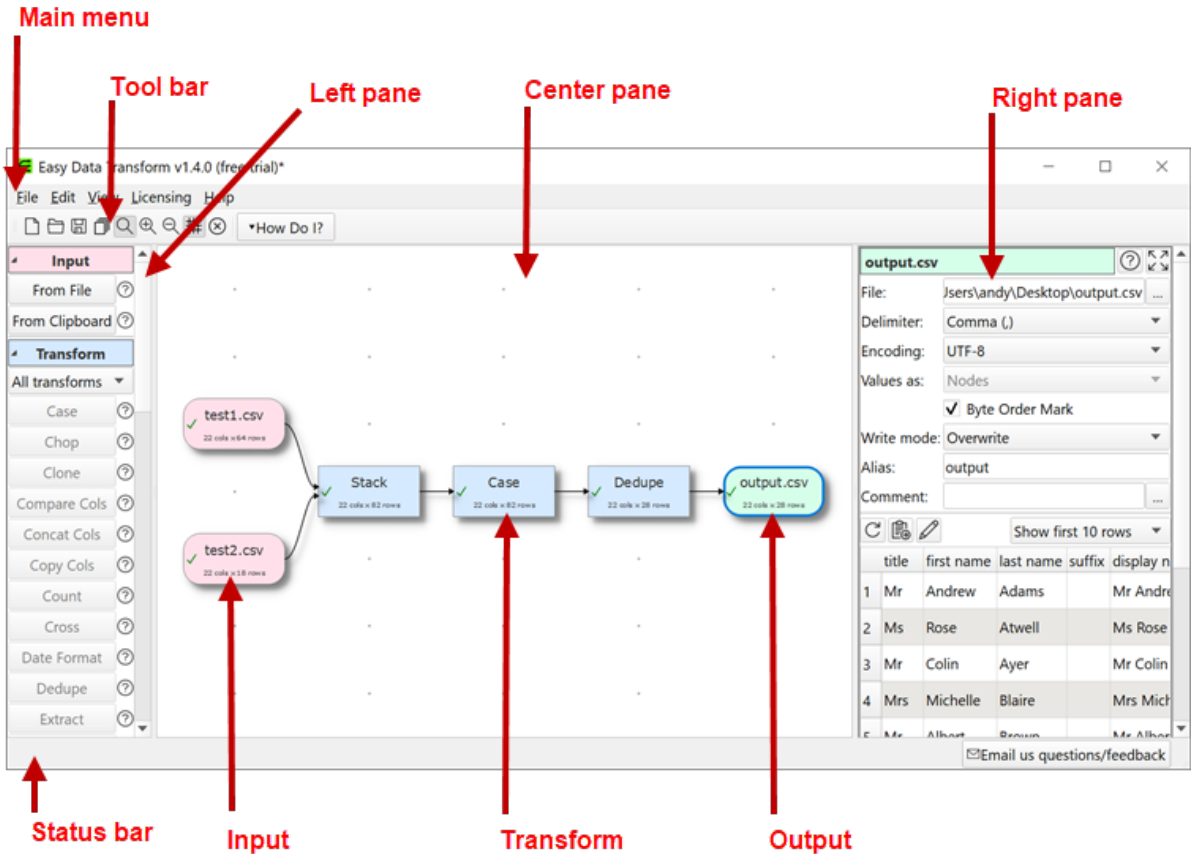
2 Reference

2.1 User Interface

2.1.1 Main window

The Main window comprises:

- **Main** menu
- **Tool** bar
- **Left** pane
- **Center** pane
- **Right** pane
- **Status** bar



2.1.2 Left pane

The **Left** pane shows all the available actions you can perform. Which actions are visible will depend on what is shown in the **Center** pane. Which actions are enabled depends on what is selected in the **Center** pane.

2.1.3 Center pane

The **Center** pane show the inputs, transforms and outputs you are using to transform your data.

2.1.4 Right pane

The **Right** pane shows details of any input, transform or output items you have selected in the **Center** pane.

2.1.5 Preferences window

Check **open previous file at start-up** if you want to start with the last file opened.

Check **give option to disable outputs when opening a file** if you want the option to disable any ouputs with write mode overwrite or append when you open a file, preventing accidentally writing over existing files. Note that this check is never made when using the -exit command line argument.

Check **make a sound when processing completed** if you want to make a system sound every time processing is completed.

Set **Tool bar icon size** to the size of the icons you wish to display in the tool bar.

Set **Right pane processing delay** depending on how long you want to wait after changes in the **Right** pane before starting processing. Setting the value to 0 is generally not recommended, as this means that every single click in the **Right** pane will cause processing.

Set **Zoom wheel behavior** according to how you want the mouse wheel to work in the **Center** pane. Hold down the `Ctrl` key while moving the mouse wheel to switch between zoom and scroll. Hold down the `Alt` key while moving the mouse wheel to switch between up/down and left/right scroll.

Data table font shows the font used in the data tables in the **Right** pane. Click **Choose...** to choose a new font. You might prefer a monospaced (fixed width) font such as Consolas, Lucida Console or Courier New. Click **Default** to set it back to the operating system default.

Set supported date formats to the date formats you wish to recognize.

2.2 Input

2.2.1 Input data

You need to input data before you can transform it. Data can be input by:

- dragging a file onto the **Center** pane; or

- clicking the **From File** or **From Clipboard** button in the **Left** pane

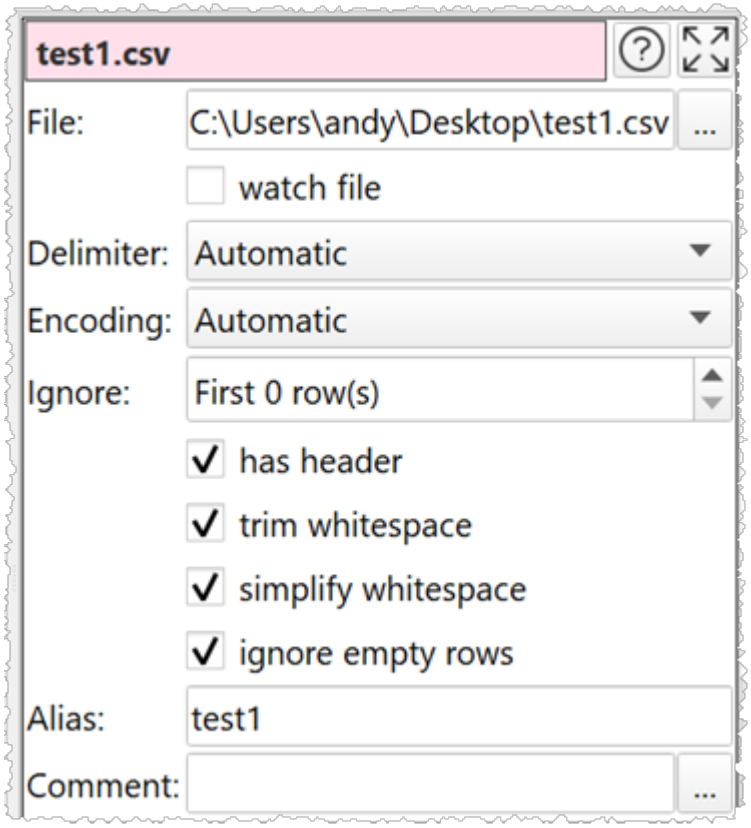
Enter the file location in **File** or click the browse button. For Excel spreadsheets you also need to add a sheet name, e.g. 'MySpreadsheet.xlsx[Sheet1]'.

Easy Data Transform can input data from files in the following formats:
- delimited text file (e.g. CSV or TSV) with various delimiters
- Excel .xlsx or .xls

Easy Data Transform will make an intelligent guess at the:
- column delimiter (e.g. comma) for CSV/TSV/text files
- text encoding (e.g. UTF-8) for CSV/TSV/text files
- presence of a header row in the data

But you can also do this manually by selecting the input item and changing the **Delimiter, Encoding** and **has header** fields in the **Right** pane.

You can select the input item in the **Center** pane and change any options related to the output in the **Right** pane.



Data will normally be read from the first non-blank line. Set **Ignore** if you want to ignore a number of rows before you start inputting. Note that empty rows are counted.

Check **watch file** if you want the file to be automatically reloaded every time that Easy Data Transform detects that it has been changed (which will then update everything 'downstream').

Check **trim white space** to trim any whitespace (e.g. tabs or spaces) off the start or end of data values.

Check **simplify whitespace** to replace any tabs or carriage returns within data values with spaces.

Check **Ignore empty rows** to remove any rows that have only empty values (whitespace is not considered empty).

Use **Comment** to record any notes that might be useful to a colleague or your future self.

2.3 Transforms

2.3.1 Transform data

Transforms operate on datasets from input data or other transforms. Some transforms only have a single input (e.g. Case), some transforms have two inputs (e.g. Join) and some transforms have two or more inputs (e.g. Stack).

To create a transform, select one or more input and/or transform items in the **Center** pane and then click the appropriate button in the **Left** pane.

Select from the drop-down list in the **Left** pane to choose which types of transform are displayed, e.g. select **Merge Transforms** to show only transforms related to blending data.

You can select the transform item in the **Center** pane and change any options related to the transform (e.g. which columns it acts on) in the **Right** pane.

The transform will be updated automatically if any input or transform 'upstream' of it changes.

Use **Comment** to record any notes that might be useful to a colleague or your future self.

2.3.2 Case

Description

Changes the case of text in one or more columns.

Inputs

One.

Options

- Check the column(s) you wish to transform.
- Set **Change case to** to **Lower case** (e.g. "text"), **Upper case** (e.g. "TEXT") or **Title case** (e.g. "Text").

See also

- Trim

2.3.3 Chop

Description

Remove characters from the start or end in one or more columns.

Inputs

One.

Options

- Check the column(s) you wish to transform.
- Set **Length** to the number of characters you want to remove.
- Set **From** to **Start** or **End** depending on whether you want to remove characters from the start or end.

Notes

- Whitespace is counted when calculating length. You can use Trim to remove whitespace before chopping.
- If you want to set a column to a fixed length use Pad and Chop together.

See also

- Extract

2.3.4 Clone

Description
Makes an exact copy of the input dataset.

Inputs
One.

Options
- None.

Notes
- Clone can be useful to simplify complicated layouts.

2.3.5 Compare Cols

Description
Creates a new column with a comparison of two other columns.

Inputs
One.

Options
- Select the two columns you wish to compare as **Column 1** and **Column 2**.

Notes
- Number, date and text values are treated differently. Any values that can be converted to a number will be treated as a number. Any values that match the supported date formats in Preferences will be treated as a date. All other values are treated as text.
- Comparisons of text are case and whitespace sensitive. You can use Case to change the case, Trim to remove whitespace before filtering and Replace to get of other unwanted characters (e.g. whitespace inside the text).
- The new column is added at the right end. You can change the column order with Reorder Cols.

See also
- Split Cols

2.3.6 Concat Cols

Description
Creates a new column by concatenating text from one or more existing columns.

Inputs
One.

Options
- Check the columns you wish to concatenate.
- Supply the **Delimiter** you wish to place between concatenated text (optional). For example ",".
- Check **keep empty** if you wish to keep the delimiter for empty columns.

Notes
- If there is a header, the header of the new column is formed from the header of the concatenated columns. You can Rename Cols it.
- Concatenating a single column makes a copy of the column.
- The new column is added at the right end. The values are in the order of the columns. You can change the column order before concatenation with Reorder Cols.

See also
- Split Cols
- Substitute

2.3.7 Copy Cols

Description
Creates one or more copies of the selected column(s).

Inputs
One.

Options
- Check the columns you wish to copy.
- Set **Copies** to the number of copies you want to make of each checked column.

Notes
- If there is a header, the header of each new column is the original column name. You can rename columns with Rename Cols.
- The new columns are added at the right end. You can change the column order with Reorder Cols.

See also
- New Col

2.3.8 Count

Description
Counts the number of occurence of each item of text in the selected column.

Inputs
One.

Options
- Select the **Column** whose values you wish to count.
- Set **Sort by** depending on whether you wish to sort alphabetically ascending by the **Text** or numerically ascending by the **Count**.

Notes
- Date and number values are treated as text.
- You can use the Sort transform to change the sort order from ascending to descending.

See also
- Pivot
- Stats
- Summary

2.3.9 Cross

Description
Creates an output from combining every possible row combination of each input. E.g. if the first input has N1 rows and the second input has N2 rows, then the result will have N1 X N2 rows. Also known as a 'Cartesian product' or 'cross join'.

Inputs
Two or more.

Options
- The output depends on the vertical (Y-axis) position of the inputs.

Notes
- It can create a very large number of rows!

See also
- Join
- Stack

2.3.10 Date Format

Description

Changes the date format in one or more columns.

Inputs

One.

Options

- Check the columns you wish to transform.
- Supply the existing date format in **Format from** (see below).
- Supply the new date format in **Format to** (see below).
- The following date formats are supported for input and output:

Format	Meaning
`d`	The day as number without a leading zero (1 to 31)
`dd`	The day as number with a leading zero (01 to 31)
`ddd`	The abbreviated localized day name (e.g. 'Mon' to 'Sun'). Uses the system locale to localize the name.
`dddd`	The long localized day name (e.g. 'Monday' to 'Sunday'). Uses the system locale to localize the name.
`M`	The month as number without a leading zero (1 to 12).
`MM`	The month as number with a leading zero (01 to 12)
`MMM`	The abbreviated localized month name (e.g. 'Jan' to 'Dec'). Uses the system locale to localize the name.
`MMMM`	The long localized month name (e.g. 'January' to 'December'). Uses the system locale to localize the name.
`yy`	The year as a two digit number (00 to 99).
`yyyy`	The year as a four digit number. If the year is negative, a minus sign is prepended in addition.

Notes

- You can also use Split Col to split a date into its component parts. For example to split "31/1/2019" into day, month and year components using the "/" delimiter.
- If the date to be converted has only two year digits, it is treated as a date between 1900 and 1999. E.g. "31/1/19" is interpreted in d-M-yy format as 31st January 1919.

Example

To change from "31/1/2019" to "01-31-19" set **Format from** to "`d/M/yyyy`" and **Format to** to "`MM-dd-yy`".

See also

- Num Format

2.3.11 Dedupe

Description

Remove duplicate rows.

Inputs

One.

Options

- Check the column(s) you wish to look for duplicate values in.

Notes

- Rows are considered duplicates if they have exactly the same value in all the columns selected.
- Comparisons are case and whitespace sensitive. You can use Case to change the case and Trim to remove whitespace before deduping.
- When several rows are duplicates, only the top one is retained.

Example

If you are cleaning up a mailing list, you might want to dedupe on the 'email' column, after converting all the emails to lower case.

See also

- Dedupe a dataset

2.3.12 Extract

Description

Extract a length of text in one or more columns.

Inputs

One.

Options

- Check the column(s) you wish to transform.
- Set **Length** to the length you want values in selected columns shortened to.
- Set **From** to **Start** or **End** depending on whether you want to take from the start or end.

- If **From** is **Start** then **Offset** is the offset of the first character from the start. If **From** is **End** then **Offset** is the offset of the last character from the end.

Notes

- Whitespace is counted when calculating length. You can use Trim to remove whitespace before extracting.
- If you want to set a column to a fixed length use Pad and Extract together.

See also

- Chop

2.3.13 Fill

Description

Fill empty cells in selected columns with the next non-empty cell value above/left (depending on direction of fill).

Inputs

One.

Options

- Check the column(s) you wish to fill.
- Select **Direction** depending on the direction you wish to fill from.

Example

This is useful for filling in gaps in hierarchical tables. For example filling down the first two columns:

Country 1	Area 1	City 1
		City 2
		City 3
	Area 2	City 1
		City 2

Country 1	Area 1	City 1
Country 1	Area 1	City 2
Country 1	Area 1	City 3
Country 1	Area 2	City 1
Country 1	Area 2	City 2

2.3.14 Filter

Description

Removes rows based on number, date and text values in selected columns.

Inputs

One.

Options

- Click the '**+**' button to add a new filter criteria.
- Click the '**x**' button to delete the selected filter criteria.
- Select **Keep** if you want to keep matching rows and **Remove** to remove matching rows.
- Select **Matching all** to match on all criteria (e.g. criteria 1 and criteria 2). Select **Matching any** to require a match on one or more criteria (e.g. criteria 1 or criteria 2).
- Set **Column** to the column you wish to match.
- Set **Op.** to the comparison operator.
- Set **Value** to the value you wish to compare.

Notes

- Number, date and text values are treated differently for **Equal to**, **Greater than**, **Less than**, **Greater than equal**, **Less than equal** and **Not equal to** operations. Any values that can be converted to a number will be treated as a number. Any values that match the supported date formats in Preferences will be treated as a date.
- All values are treated as text for **Contains**, **Starts with**, **Ends with**, **Matches regex**, **Is not**, **Doesn't start with, Doesn't end with** and **Doesn't match regex** operations.
- Comparisons of text are case and whitespace sensitive. You can use Case to change the case, Trim to remove whitespace before filtering and Replace to get of other unwanted characters (e.g. whitespace inside the text).
- See here for more details on Regular expressions (regex).

2.3.15 Gather

Description

Gather multiple columns into new key and value columns. Also called a long pivot or group by.

Inputs

One.

Options

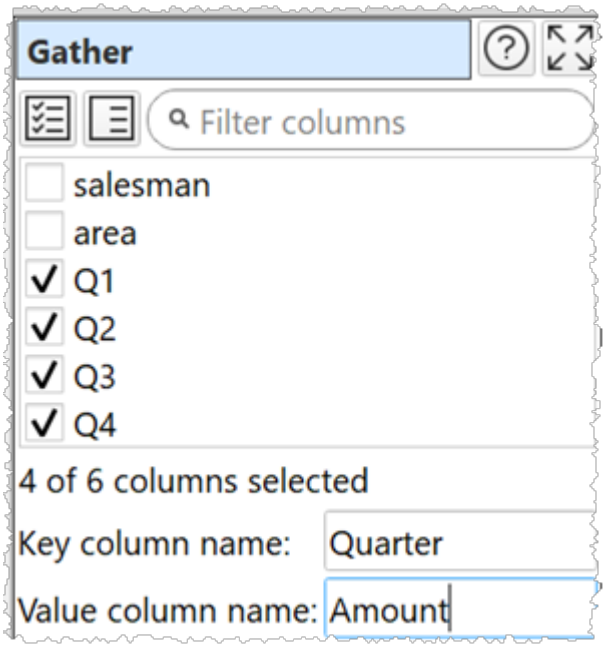
- Select the **Columns** you wish to gather.

- Set **Key column name** to the name of the new key column, which will have values based on the names of the columns selected.
- Set **Value column name** to the name of the new value column, which will have values based on the values in the columns selected.

Example

	salesman	area	Q1	Q2	Q3	Q4
1	Alice	North	11.3	89.3	44.3	18
2	Bob	East	4.5	7.9	8	3.3

With columns Q1, Q2, Q3 and Q4 gathered:



Gives:

	salesman	area	Quarter	Amount
1	Alice	North	Q1	11.3
2	Alice	North	Q2	89.3
3	Alice	North	Q3	44.3
4	Alice	North	Q4	18
5	Bob	East	Q1	4.5
6	Bob	East	Q2	7.9
7	Bob	East	Q3	8
8	Bob	East	Q4	3.3

Notes

- New columns are added at the right end. You can change the column order with Reorder Cols.
- You can merge the value and key columns into a single column with Concat Cols.
- The opposite of Gather is Spread.

2.3.16 If

Description

Sets the value of a new column based conditionally on values in one or more other columns.

Inputs

One.

Options

- Click the '**+IF**' button to add a new IF/ELSE IF..THEN condition.
- Click the '**+AND**' button to add an AND to the selected IF/ELSE IF..THEN.
- Click the '**x**' button to delete the selected IF/ELSE IF..THEN/AND.
- The **Logic** column shows the type of row.
- Set **Column** to the column you wish to match.
- Set **Op.** to the comparison operator.
- Set **Value** to the value you wish to compare.

Notes

- You can simulate OR with multiple IF statements. For example:

```
IF x = 1 OR y = 2
   THEN 3
```

Is equivalent to:

```
IF x = 1
   THEN 3
ELSE IF y = 2
   THEN 3
```

- Number, date and text values are treated differently for **Equal to**, **Greater than**, **Less than**, **Greater than equal, Less than equal** and **Not equal to** operations. Any values that can be converted to a number will be treated as a number. Any values that match the supported date formats in Preferences will be treated as a date.
- All values are treated as text for **Contains**, **Starts with**, **Ends with**, **Matches regex**, **Is not**, **Doesn't start with, Doesn't end with** and **Doesn't match regex** operations.
- Comparisons of text are case and whitespace sensitive. You can use Case to change the case, Trim to remove whitespace before filtering and Replace to get of other unwanted characters (e.g. whitespace inside the text).
- See here for more details on Regular expressions (regex).

See also

- Lookup

2.3.17 Insert

Description

Append/prepend text to one or more columns.

Inputs

One.

Options

- Check the column(s) you wish to transform.
- In **Insert** put the text you want to insert.
- In **At** put the position you want the text inserted.

Notes

- You can use Trim to remove whitespace before inserting.

See also

- Pad
- Extract

2.3.18 Intersect

Description

Keep only rows from the top dataset with key values that are present in the lower dataset.

Inputs

Two.

Options

- The output depends on the vertical (Y-axis) position of the inputs.
- Select **Top key column** for the column you want to match in the top input dataset.
- Select **Bottom key column** for the column you want to match in the bottom input dataset.

Notes

- Easy Data Transform will try to guess sensible default values for **Top key column** and **Bottom key column** based on column header names and contents.
- If the first input has a header, this will be used for the output.
- All values are treated as text and comparisons are case and whitespace sensitive. You can use Case to change the case and Trim to remove whitespace before the intersect.
- Does not remove duplicates. You can use Dedupe to do this.
- You can use Concat Cols to join several columns together (e.g. 'first name' and 'last name' columns) to form a key column.
- You can use Row Num to create a unique key column.

See also

- Subtract

2.3.19 Javascript

Description

Create a custom transform using Javascript (ECMAScript).

Easy Data Transform allows you to carry out a wide range of data transformations without programming. But sometimes you might need a specialist transformation that can't be done with the built-in transforms. For that you can use the **Javascript** transform. It allows you to write the body of a Javascript function, to calculate a value for each row in a new column. Existing column values can be used as variables.

Javascript is a fully-fledged programming language and can perform arbitrarily complex transforms. It can handle numbers, dates and text.

Inputs

One.

Examples

To multiply the value in column 'items' by the value in column 'item price':

```
return $(items) * $(item price);
```

To concatenate 'last' and 'first' columns with a comma and a space:

```
return $(last) + ', ' + first;
```

To calculate the biggest of columns 'v1' and 'v2':

```
return Math.max( $(v1), $(v2) );
```

To determine whether phone numbers in the 'phone_num' column are valid using a regular expression:

```
const validPhoneNum = /^[\+]?[(]?[0-9]{3}[)]?[-\s\.]?[0-9]{3}[-\s\.]?[0-9]{4,6}$/;
if ( validPhoneNum.test( $(phone_num) ) )
   return "valid";
else
   return "invalid";
```

To calculate the number of years difference between dates in column 1 and column 2:

```
return new Date( $(1) ).getFullYear() - new Date( $(2) ).getFullYear();
```

To calculate the number of whole days difference between a date in the 'created' column and today (negative for future dates):

```
return Math.floor( ( new Date() - new Date( $(created) ) ) / ( 1000*60*60*24 ) );
```

To reverse the text in the 'key' column:

```
var newString = $(key);
for (var i = a.length - 1; i >= 0; i--) {
   newString += a[i];
}
return newString;
```

Options

- Enter your script into the **Javascript** field. The script should be the body of a Javascript function.
- Select a column from **Insert variable** to add that variable into the **Javascript** field at the current cursor position.
- Click the **Evaluate** button to evaluate your Javascript expression over every row and show any errors.

Notes

- You can reference column values by their name (e.g. `$(item cost)` for the 'item cost' column) or index (e.g. `$(1)` for the first column). The column name is case sensitive. You will get a warning if more than one column has the same name.
- The **Javascript** transform is calculated every time:
  - The **Evaluate** button is pressed.
  - The **Javascript** transform item is unselected in the **Center** pane and script changes have been made without the **Evaluate** button being clicked.
  - The item upstream of it changes.
- Javascript may not recognize numeric values containing commas (e.g. '1,234') as numeric. But you can use the Replace transform to remove commas from numeric column before processing the Javascript transform.
- The new column is added at the right end. You can change the column order with Reorder Cols and the column name with Rename Cols.
- Any errors from the Javascript engine are shown in a message window when **Evaluate** is clicked.
- The Javascript Date() object evaluates to the number of milliseconds since 1 January 1970 UTC. Date() is the current date.
- Date values passed to Javascript Date() objects should be in ISO ('yyyy-mm-dd') format, e.g. '2020-01-31' (not '2020-1-31').
- If you want to carry out your transform across more than one dataset, you should Join them first.
- The Javascript transform is very versatile and quite fast. But is not as fast as built-in transforms. So we recommend you use built-in transforms where possible.
- Javascript running in Easy Data Transform is not 'sandboxed' and has the same privileges as the Easy Data Transform executable. However the Javascript does not have access to window(), XMLHttpRequest() or ActiveXObject(). So we aren't aware of any way that a bad actor could damage your system from Javascript sent in a .transform file.
- Javascript is far too big a topic to cover here. However there are many detailed resources online. If you are stuck contact support.
- If you only want to combine text from columns, use the simpler Substitute transform.

2.3.20 Join

Description

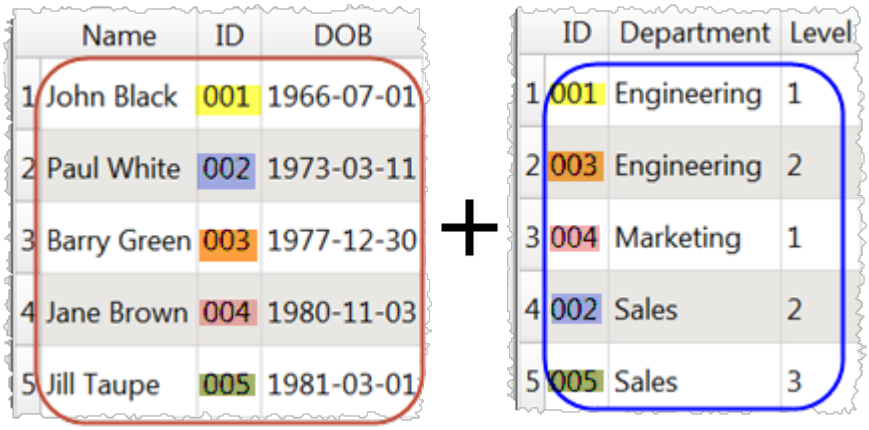
Join two inputs based on common (key) columns, e.g. on an email address or id column present in both inputs.

Inputs

Two.

Example

Joining these two datasets by the **ID** column in each:

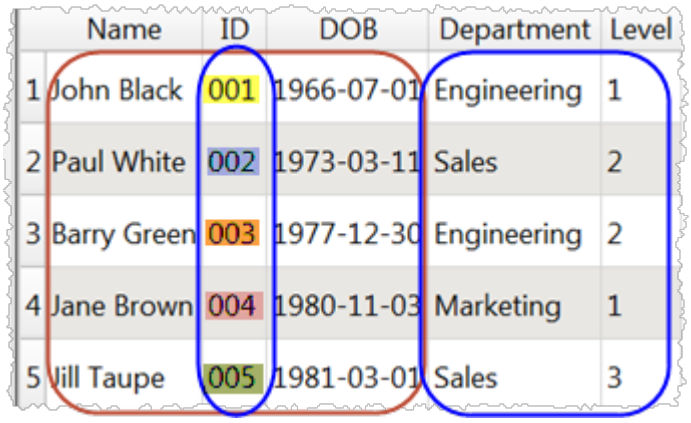


	Name	ID	DOB
1	John Black	001	1966-07-01
2	Paul White	002	1973-03-11
3	Barry Green	003	1977-12-30
4	Jane Brown	004	1980-11-03
5	Jill Taupe	005	1981-03-01

+

	ID	Department	Level
1	001	Engineering	1
2	003	Engineering	2
3	004	Marketing	1
4	002	Sales	2
5	005	Sales	3

Gives:



	Name	ID	DOB	Department	Level
1	John Black	001	1966-07-01	Engineering	1
2	Paul White	002	1973-03-11	Sales	2
3	Barry Green	003	1977-12-30	Engineering	2
4	Jane Brown	004	1980-11-03	Marketing	1
5	Jill Taupe	005	1981-03-01	Sales	3

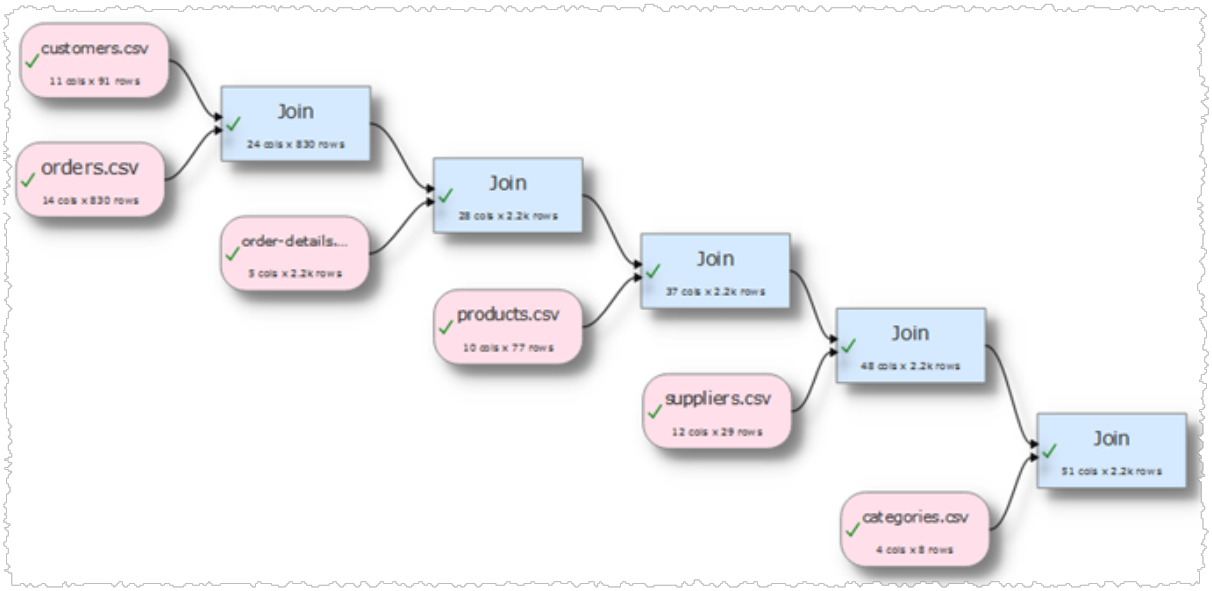
Options

- The output depends on the vertical (Y-axis) position of the inputs.
- Select **Top key column** for the column you want to match in the top input dataset.

- Select **Include top non-matching rows** if you want to include in the output any rows in the top input with no matching value in the bottom input (a 'left join').
- Select **Bottom key column** for the column you want to match in the bottom input dataset.
- Select **Include bottom non-matching rows** if you want to include in the output any rows in the bottom input with no matching value in the top input (a 'right join').

Notes

- Easy Data Transform will try to guess sensible default values for **Top key column** and **Bottom key column** based on column header names and contents.
- All values are treated as text and comparisons are case and whitespace sensitive. You can use Case to change the case and Trim to remove whitespace before the intersect.
- Use Concat Cols to join several columns together (e.g. 'first name' and 'last name' columns) to form a key column.
- Use Row Num to create a unique key column.
- Cascade multiple joins to join more than 2 datasets.



See also

- Cross
- Stack
- Lookup
- Merge datasets

2.3.21 Lookup

Description

Looks up the values of a column in the top input dataset in the bottom input dataset and puts the result in a new column.

Inputs

Two.

Example

If you have one dataset with category IDs and another dataset with category IDs and category names, you can create a new category name column in the first dataset by looking up the category ID in the second dataset.

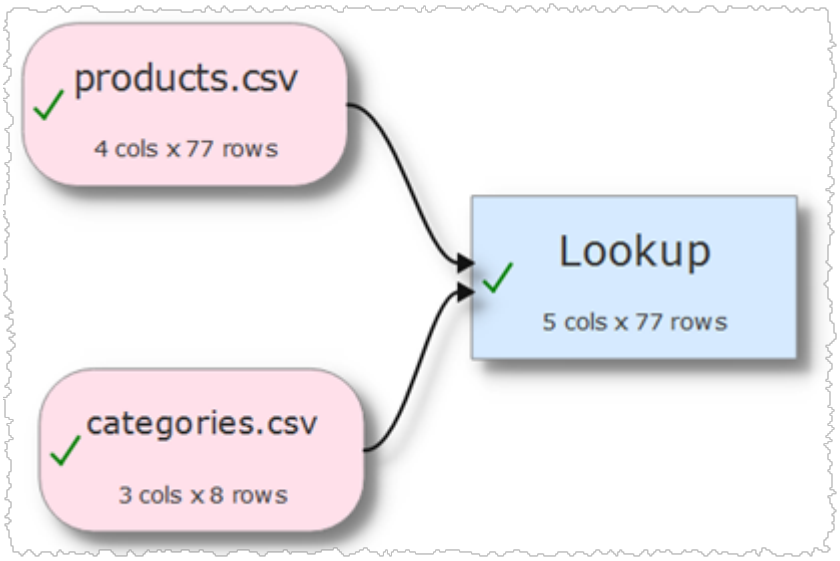
First dataset:

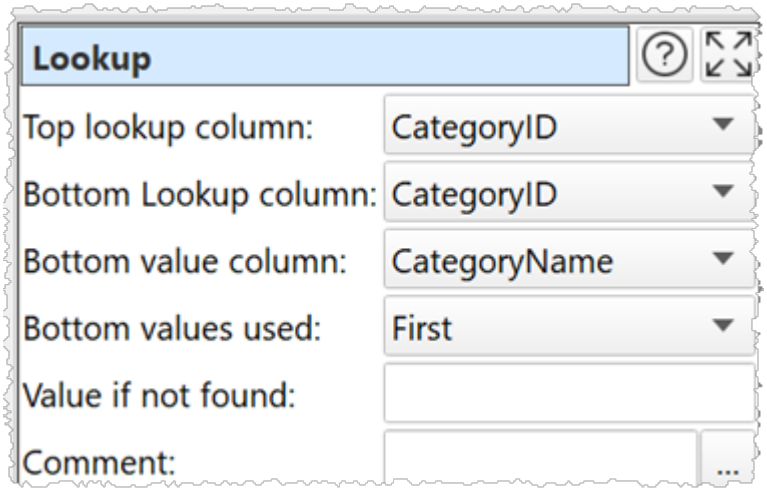
	ProductID	ProductName	SupplierID	CategoryID
1	1	Chai	1	1
2	2	Chang	1	1
3	3	Aniseed Syrup	1	2
4	4	Chef Anton's Cajun Seasoning	2	2
5	5	Chef Anton's Gumbo Mix	2	2

Second dataset:

	CategoryID	CategoryName	Description
1	1	Beverages	Soft drinks coffees teas beers and ales
2	2	Condiments	Sweet and savory sauces relishes spreads and seasonings
3	3	Confections	Desserts candies and sweet breads

Lookup transform:





Result:

	ProductID	ProductName	SupplierID	CategoryID	Lookup CategoryName
1	1	Chai	1	1	Beverages
2	2	Chang	1	1	Beverages
3	3	Aniseed Syrup	1	2	Condiments
4	4	Chef Anton's Cajun Seasoning	2	2	Condiments
5	5	Chef Anton's Gumbo Mix	2	2	Condiments

Options

- Place the dataset you want to modify as the top input and the dataset you want to lookup values from as the bottom input.

- Select **Top lookup column** for the column whose values you wish to lookup.
- Select **Bottom lookup column** for the column that matches the lookup in the bottom dataset.
- Select **Bottom value column** for the column that contains the values.
- Set **Bottom values used** to **First** if you want use the first match in **Bottom lookup column** and **All** if you want to use all matches.
- Set **Value if not found** to the value you want to set for values in **Top lookup column** that do not exist in **Bottom lookup column**.

Notes

- Easy Data Transform will try to guess sensible default values for **Top lookup column** and **Bottom lookup column** based on column header names and contents.
- **Bottom values used** is only important if there are duplicates in **Bottom lookup column**.
- If the first input has a header, this will be used for the output.
- All values are treated as text and comparisons are case and whitespace sensitive. You can use Case to change the case and Trim to remove whitespace before the intersect.
- If you want to lookup values in multiple columns, use Concat Cols to join several columns together to form new columns.

See also

- If
- Join

2.3.22 New Col

Description

Adds a new column, filled with a given value.

Inputs

One.

Options

- Set **New column value** to the value for every cell of the new column. You can leave it blank for an empty column.

Notes

- New columns are always added at the right end. You can change the column order with Reorder Cols and the column name with Rename Cols.

See also

- Copy Cols

- Remove Cols

2.3.23 Num Format

Description

Change the number format in one or more columns.

Inputs

One.

Options

- Check the column(s) you wish to transform.
- Set **Format** to the new number format (see below).
- For the **e**, **E**, and **f** formats, **Precision** represents the number of digits after the decimal point. For the **g** and **G** formats, **Precision** represents the maximum number of significant digits (trailing zeros are omitted).
- The following number formats are supported:

Format	Meaning
`e`	Format as [-]9.9e[+|-]999. E.g. `1234567.89` is shown as `1.235e+06`.
`E`	Format as [-]9.9E[+|-]999. E.g. `1234567.89` is shown as `1.235E+06`.
`f`	Format as [-]9.9. E.g. `1234567.89` is shown as `1234567.89`.
`g`	Use e or f format, whichever is the most concise
`G`	Use E or f format, whichever is the most concise

Notes

- Non-numerical values are ignored.
- You can also use Extract and Pad to change the number of characters.

See also

- Date Format

2.3.24 Pad

Description

Pad text to a minimum length in one or more columns.

Inputs

One.

Options

- Check the column(s) you wish to transform.
- Set **Minimum length** to the length you want values in selected columns padded to. Values this length or longer are unaffected.
- Set **Pad** to **Left** or **Right** depending on where you want any padding characters added.
- Set **Pad with** to the character you want to pad with.

Notes

- Whitespace is counted when calculating length. You can use Trim to remove whitespace before padding.

2.3.25 Pivot

Description

Creates a pivot table to summarise values for one or two columns.

Inputs

One.

Options

- Use **Pivot column** to select the column values you want to use as columns in your pivot table.
- Use **Pivot row** to select the column values you want to use as rows in your pivot table.
- Use **Pivot sum** to select which column you wish to sum.

Notes

- Pivot tables are very useful for analysing data.
- Any non-numerical values in the **Pivot sum** column are ignored.

See also

- Count
- Stats
- Summary

2.3.26 Remove Cols

Description

Removes columns.

Inputs
One.

Options
- Uncheck the column(s) you wish to remove.

Notes
- The column will be removed from any dataset 'downstream'.

See also
- New Col

2.3.27 Rename Col

This transform is deprecated. Use Rename Cols instead.

Description
Rename a column header.

Inputs
One.

Options
- Select the column header you wish to rename in **Column**.
- Set **Rename to** to the new column header name.

Notes
- The names of column headers do not have to be unique.

2.3.28 Rename Cols

Description
Rename column headers.

Inputs
One.

Options
- Change the column headers using the **New name** column.
- Click **Lower** to change all the names in the **New name** column to lower case.
- Click **Upper** to change all the names in the **New name** column to upper case.

- Click **Title** to change all the names in the **New name** column to title case.
- Click **Reset** to change all the names in the **New name** column back to their original name.

Notes

- The names of column headers do not have to be unique.

2.3.29 Reorder Cols

Description

Reorder columns.

Inputs

One.

Options

Drag the columns into the desired order (left-most at the top).

Notes

You can also rename columns with Rename Cols and remove unwanted columns with Remove Cols.

2.3.30 Replace

Description

Replace text in one or more columns.

Inputs

One.

Options

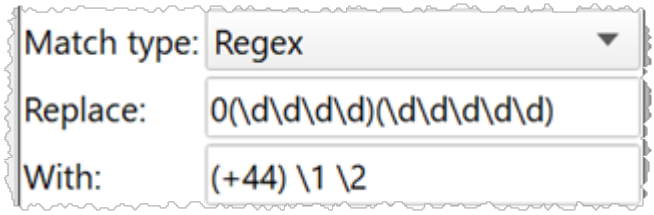
- Check the column(s) you wish to transform.
- Choose whether to use text or Regular expression matching.
- In **Replace** put the text you want to replace.
- In **With** put the text you want to replace it with.

Notes

- Comparisons are case and whitespace sensitive. You can use Case to change the case and Trim to remove whitespace before replacing.

Example

You can turn `0123456789` into `(+44) 1234 56789` using a Regular expression:



See also

- Insert
- Substitute

2.3.31 Row Num

Description

Add a new column that contains the row number.

Inputs

One.

Options

- Set **Start at** to the number you want to use for the first row.
- Set **Increment** to the amount you wish to increment by.
- set **Every** to how often to apply the increment (e.g. set to 5 to increment once every 5 rows).

Notes

- The new column is added at the right end. You can change the column order with Reorder Cols and the column name with Rename Cols.

2.3.32 Sample

Description

Selects a subset of rows from the input.

Inputs

One.

Options

- Set **Rows** to the number of rows you want to output. If it is the same or greater than the number of rows in the input, then the input will be unaffected.
- Set **Select** depending on how you want the rows sampled.

Notes

- If you are transforming a large dataset, then you can use Sample to test a small subset.

2.3.33 Sort

Description

Sorts rows by one or more columns.

Inputs

One.

Options

- Click the '**+**' button to add a new sort level.
- Click the '**x**' button to delete the selected sort level(s).
- Click the up arrow to move the selected sort level(s) up.
- Click the down arrow to move the selected sort level(s) down.
- Set **Column** to the column you want to sort by.
- Set **Order** depending on whether you want to sort this column **Ascending** or **Descending**.

Notes

- If you add multiple levels, it will sort by level 1 then level 1 values that are the same will be sorted by level 2 etc.
- Number, date and text values are treated differently for comparison purposes.
- Any values that can be converted to numbers will be treated as numbers.
- Any values that match the supported date formats in Preferences will be treated as dates.
- Comparisons of text are case and whitespace sensitive. You can use Case to change the case and Trim to remove whitespace before filtering.

2.3.34 Split Col

Description

Creates one or more new columns by splitting text at delimiters in a selected column.

Inputs

One.

Options

- Select the **Column** you wish to split.
- Supply the **Delimiter** you wish to use to split the column.
- Set **Ordering** depending on how you want to order values after splitting.
- Check **keep empty** if you wish to honor delimiters with nothing in between.

- set **Min. new cols** to the minimum number of new columns you wish to add.
- set **Max. new cols** to the maximum number of new columns you wish to add (ignored if less than minimum).

Notes

- If no **Delimiter** is supplied then no new columns are created.
- New columns are added at the right end. You can change the column order with Reorder Cols.
- If there is a header, the header of the new column is based on the original header. You can change the column name with Rename Cols.

See also

- Concat Cols

2.3.35 Spread

Description

Spread a column into multiple new columns. Also called a wide pivot or crosstab.

Inputs

One.

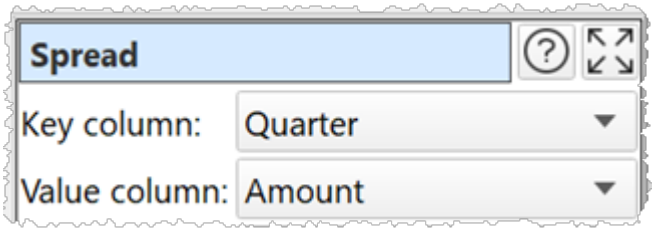
Options

- Select the **Key column** and **Value column** you wish to spread.
- **Missing values** is used for values missing from the input dataset.

Example

	salesman	area	Quarter	Amount
1	Alice	North	Q1	11.3
2	Alice	North	Q2	89.3
3	Alice	North	Q3	44.3
4	Alice	North	Q4	18
5	Bob	East	Q1	4.5
6	Bob	East	Q2	7.9
7	Bob	East	Q3	8
8	Bob	East	Q4	3.3

With Quarter and Amount columns spread:



Gives:

	salesman	area	Q1	Q2	Q3	Q4
1	Alice	North	11.3	89.3	44.3	18
2	Bob	East	4.5	7.9	8	3.3

Notes

- If there are rows that are duplicates, apart from the value column, this will cause errors.
- New columns are added at the right end. You can change the column order with Reorder Cols.

- You can merge the new columns into a single column with Concat Cols.
- The opposite of Spread is Gather.

2.3.36 Stack

Description

Stack the rows from inputs, one on top of the other.

Inputs

One or more.

Example

Stacking these two datasets by the **ID** column in each:

	Name	ID	DOB
1	John Black	001	01/07/1966
2	Paul White	002	11/03/1973
3	Barry Green	003	30/12/1977

+

	Name	ID	DOB
1	Jane Brown	004	03/11/1980
2	Jill Taupe	005	01/03/1981

Gives:

	Name	ID	DOB
1	John Black	001	1966-07-01
2	Paul White	002	1973-03-11
3	Barry Green	003	1977-12-30
4	Jane Brown	004	1980-11-03
5	Jill Taupe	005	1981-03-01

Options

- Select **Align columns by** to **Header name** if you want line up column values by header name (e.g. the 'id' column in input 1 with the 'id' column in input 2) and **Column number** to align by the column number (e.g. the first column of input 1 with the first column of input 2). The headers will be matched case insensitive (e.g. 'id' to 'ID'), if no case sensitive match is possible.
- The output depends on the vertical (Y-axis) position of the inputs.

Notes

- If you align by **Column number** the header of the first input is used.

See also

- Cross
- Join
- Merge datasets

2.3.37 Stamp

Description

Adds a time/date stamp as a new row or a new column.

Inputs

One.

Options

- Supply the processing date/time format in **Format** (see below).

Format	Meaning
`d`	The day as number without a leading zero (1 to 31)
`dd`	The day as number with a leading zero (01 to 31)
`ddd`	The abbreviated localized day name (e.g. 'Mon' to 'Sun'). Uses the system locale to localize the name.
`dddd`	The long localized day name (e.g. 'Monday' to 'Sunday'). Uses the system locale to localize the name.
`M`	The month as number without a leading zero (1 to 12).
`MM`	The month as number with a leading zero (01 to 12)
`MMM`	The abbreviated localized month name (e.g. 'Jan' to 'Dec'). Uses the system locale to localize the name.
`MMMM`	The long localized month name (e.g. 'January' to 'December'). Uses the system locale to localize the name.
`yy`	The year as two digit number (00 to 99).

Format	Meaning
`yyyy`	The year as four digit number. If the year is negative, a minus sign is prepended in addition.
`h`	The hour without a leading zero (0 to 23 or 1 to 12 if AM/PM display).
`hh`	The hour with a leading zero (00 to 23 or 01 to 12 if AM/PM display).
`H`	The hour without a leading zero (0 to 23, even with AM/PM display).
`HH`	The hour with a leading zero (00 to 23, even with AM/PM display).
`m`	The minute without a leading zero (0 to 59).
`mm`	The minute with a leading zero (00 to 59).
`s`	The whole second without a leading zero (0 to 59).
`ss`	The whole second with a leading zero where applicable (00 to 59).
`z`	The fractional part of the second, to go after a decimal point, without trailing zeroes (0 to 999). Thus "s.z" reports the seconds to full available (millisecond) precision without trailing zeroes.
`AP or A`	The fractional part of the second, to millisecond precision, including trailing.
`ap or a`	Use am/pm display. a/ap will be replaced by either "am" or "pm".
`t`	The timezone (for example "CEST").

- Select from **Position** whether you want the stamp row added to the start or end of the dataset or to every row in a new column.

Notes

- If you add the stamp to **Every Row** you can move the column using Reorder Cols.

2.3.38 Stats

Description

Add a row with the average, minimum or maximum of numeric values in one or more selected columns.

Inputs

One.

Options

- Check the column(s) you wish to add stats for.

Notes

- The average is the arithmetic mean.
- Non-numerical values are ignored.

See also

- Count
- Pivot
- Summary

2.3.39 Substitute

Description

Substitute column values into text.

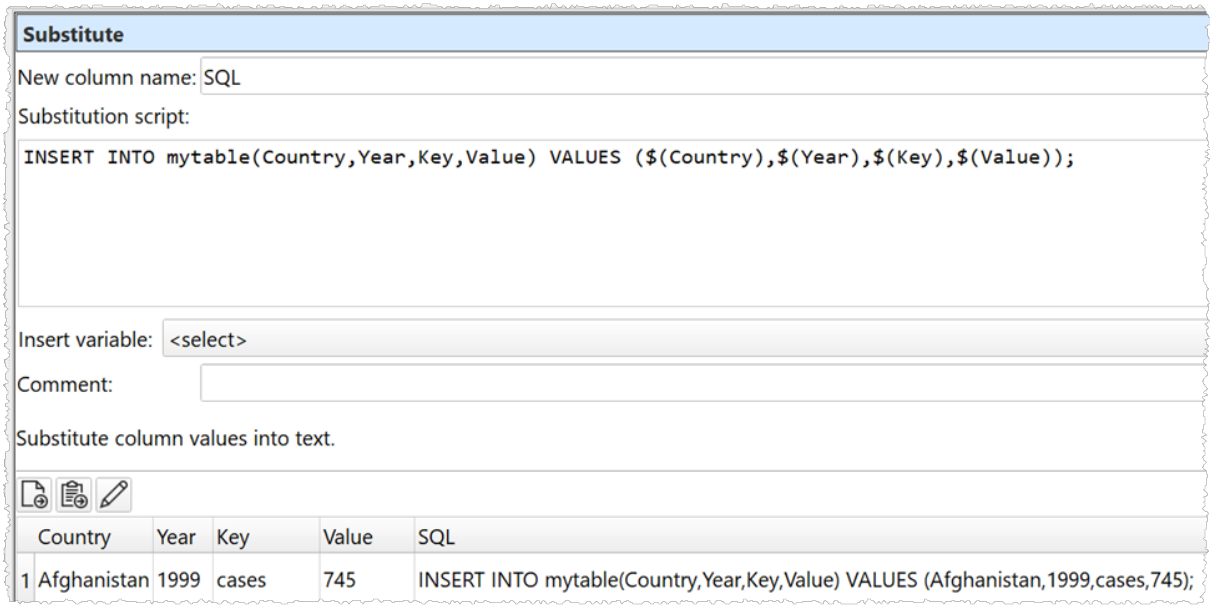
Inputs

One.

Example

To create SQL statements to insert 'Country', 'Year', 'Key' and 'Value' column values:

```
INSERT INTO mytable(Country,Year,Key,Value) VALUES ($(Country),$(Year),$(Key),$(Value
```



Options

- Enter your substitution script into the **Substitution script** field.
- Select a column from **Insert variable** to add that variable into the **Substitution script** field at the current cursor position.
- Click the **Evaluate** button to evaluate your script over every row.

Notes

- You can reference column values by their name (e.g. `$(item cost)` for the 'item cost' column) or index (e.g. `$(1)` for the first column). The column name is case sensitive. You will get a warning if more than one column has the same name.
- The transform is calculated every time:
  - The **Evaluate** button is pressed.
  - The **Substitute** transform item is unselected in the **Center** pane and script changes have been made without the **Evaluate** button being clicked.
  - The item upstream of it changes.
- The new column is added at the right end. You can change the column order with Reorder Cols and the column name with Rename Cols.
- If you want to carry out your transform across more than one dataset, you should Join them first.
- If you need to do something more complex than this transform allows, try the Javascript transform.

2.3.40 Subtract

Description

Remove rows from the top dataset with key values that are present in the lower dataset.

Inputs

Two.

Options

- The output depends on the vertical (Y-axis) position of the inputs.
- Select **Top key column** for the column you want to match in the top input dataset.
- Select **Bottom key column** for the column you want to match in the bottom input dataset.

Notes

- Easy Data Transform will try to guess sensible default values for **Top key column** and **Bottom key column** based on column header names and contents.
- If the first input has a header, this will be used for the output.
- All values are treated as text and comparisons are case and whitespace sensitive. You can use Case to change the case and Trim to remove whitespace before the subtract.
- Does not remove duplicates. You can use Dedupe to do this.
- You can use Concat Cols to join several columns together (e.g. 'first name' and 'last name' columns) to form a key column.
- You can use Row Num to create a unique key column.

See also

- Intersect

2.3.41 Summary

Description

Summarise the values in the selected columns.

Inputs

One.

Options

- Select the **Columns** you wish to summarise.
- Check **check for dates** if you wish to check for date values using supported date formats. This can be slow for large datasets.

Notes

- **Empty values** is the number of values in the column that are completely empty. Values with whitespace do not count as empty.
- **Numeric values** is the number of numeric of values in the column that can be interpreted as a number.
- **Date values** is the number of values in the column that can be interpreted as a date. Only shown if **check for dates** is checked.
- **Text values** is the number of values in the column that cannot be interpreted as empty, numeric or date.
- **Unique values** is the number of unique values in the column. Empty values are not counted. Date and numeric values are treated as text (e.g. '7' is treated as different to '7.0' and '1/1/2020' is treated as different to '01/01/2020'). Comparison between values is sensitive to case and whitespace.
- **Min length** is the minimum number of characters of a value in the column. Whitespace is counted. Date and numeric values are treated as text.
- **Max length** is the maximum number of characters of a value in the column. Whitespace is counted. Date and numeric values are treated as text.
- **Min numeric** is the minimum numeric value in the column.
- **Max numeric** is the maximum numeric value in the column.
- **Min date** is the minimum date value in the column. Only shown if **check for dates** is checked.
- **Max date** is the maximum date value in the column. Only shown if **check for dates** is checked.
- **Most frequent** lists the most common text in the column. Empty values are not counted. Date and numeric values are treated as text. Comparison between values is sensitive to case and whitespace.

- You can use Trim to remove any whitespace at the start or end of values before Summary.
- If you wish to have a row displayed per column you can Transpose the table.

See also

- Count
- Pivot
- Stats

2.3.42 Total

Description

Add a new column with a running (cumulative) total of the selected column.

Inputs

One.

Options

- Set **Column** to the column you want to total.

Notes

- Non-numerical values are ignored.
- The new column is added at the right end. You can change the column order with Reorder Cols and the column name with Rename Cols.

See also

- Count
- Pivot
- Stats

2.3.43 Transpose

Description

Swap (rotate) rows and columns, so that each row becomes a column and each column becomes a row.

Inputs

One.

Options

- Check **has header** to make the new first row into a header (requires >1 row).

Notes

- If the input dataset has a header, it will become the new first column. Use Remove Cols to remove it.
- Datasets with very large numbers of columns can be slow to display.

2.3.44 Trim

Description

Removes leading and trailing whitespace from one or more columns.

Inputs

One.

Options

- Check the column(s) you wish to transform.

See also

- Case

2.4 Output

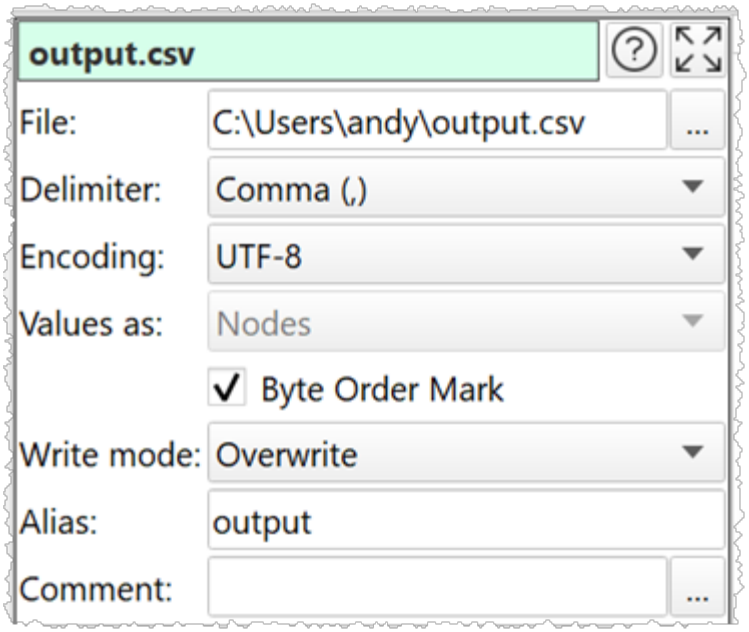
2.4.1 Output data

Once you have finished transforming your data you can output it in the following formats:

- CSV
- Excel
- JSON
- HTML
- Markdown
- TSV
- vCard
- XML
- YAML

To create an output, select 1 input and/or transform item in the **Center** pane and then click the **To File** button at the bottom of the **Left** pane. You can choose the file type in the **Save as type** drop-down list of the **Output** window.

You can select the output item in the **Center** pane and change any options related to the output in the **Right** pane.



Set **File** to the location of the file you want to output. If you are writing to a .xls or .xlsx file the output will be written to a sheet called 'Easy Data Transform'.

Set **Delimiter** to the delimiter you wish to use (only available for delimited text files, such as CSV and TSV).

Set **Encoding** to the text encoding you wish to use (only available for delimited text files).

Set **Values as** depending on how you want to structure the output (only available for XML files).

Set **Byte Order Mark** checked write a Unicode Byte Order Mark to the file (only available for UTF encodings).

Set **Write mode** to:
- 'Disabled' not to write to the file.
- 'Overwrite' to create a new file (if none exists) or overwrite (if the file exists).
- 'Append' to create a new file (if none exists) or append to it (if the file exists).
- 'New' to create a new file (if none exists) or do nothing (if the file exists).

Use **Comment** to record any notes that might be useful to a colleague or your future self.

2.5 File formats

Enter topic text here.

2.5.1 CSV format

Easy Data Transform can input from and output to CSV format files. File extension ".csv".

CSV (Comma Separated Value) format is commonly used for exchanging tabular data between programs.

CSV is a type of delimited text file format. Carriage return denotes the end of a row. The column delimiter is usually commas, but not always.

Easy Data Transform supports the following column delimiters:
- comma (,)
- semi-colon (;)
- colon (:)
- pipe (|)
- caret (^)

For all the above delimiters:
- If a value field contains a quote (") character, then the quote will be 'escaped' by an additional quote when output.
- If a value field contains a delimiter, quote or carriage return character, then the value be surrounded by quotes (") when output.

For example:

	CategoryID	CategoryName	Description	In stock
1	1	Beverages	Soft drinks, coffees & teas	true
2	2	Condiments	Sweet and savory sauces	false
3	3	Confections	Candies and sweet breads	true

Is output as:

```
CategoryID,CategoryName,Description,In stock
1,Beverages,"Soft drinks, coffees & teas",true
2,Condiments,Sweet and savory sauces,false
3,Confections,Candies and sweet breads,true
```

Many CSV file are not well formed. For example, they have unescaped quotes. As the CSV format is not well-defined, badly formed CSV files can be interpreted in more than one way. Easy Data Transform will do the best it can in these circumstances.

Tab delimited (TSV) files are treated a bit differently.

2.5.2 Excel format

Easy Data Transform can input from and output to Excel ".xlsx" and ".xls" format files, even if you don't have Excel installed.

Excel format is the native format of the Microsoft Excel spreadsheet application. It is commonly used for exchanging tabular data.

2.5.3 JSON format

Easy Data Transform can output to JSON format files. File extension ".json".

JSON (JavaScript Object Notation) format is commonly used for exchanging data between programs.

For example:

	CategoryID	CategoryName	Description	In stock
1	1	Beverages	Soft drinks, coffees & teas	true
2	2	Condiments	Sweet and savory sauces	false
3	3	Confections	Candies and sweet breads	true

Is output as:

```
[
  {
    "CategoryID": 1,
    "CategoryName": "Beverages",
    "Description": "Soft drinks, coffees & teas",
    "In stock": true
  },
  {
    "CategoryID": 2,
    "CategoryName": "Condiments",
    "Description": "Sweet and savory sauces",
    "In stock": false
  },
  {
    "CategoryID": 3,
```

```
    "CategoryName": "Confections",
    "Description": "Candies and sweet breads",
    "In stock": true
  }
]
```

2.5.4 HTML format

Easy Data Transform can output to tables in HTML format files. File extension ".html".

HTML (HyperText Markup Language) format is commonly used for creating web pages. If you don't need the data to take up a whole page, you can just copy the `<table>` to `</table>` part of the output.

For example:

	CategoryID	CategoryName	Description	In stock
1	1	Beverages	Soft drinks, coffees & teas	true
2	2	Condiments	Sweet and savory sauces	false
3	3	Confections	Candies and sweet breads	true

Is output as:

```
<!DOCTYPE html>
<html>
  <head>
    <meta charset="UTF-8"/>
    <title>C:\Users\andyb\Desktop\output.html</title>
    <style>table,td,th{border:1px solid black;text-align:left;vertical-align:top;borde
  </head>
  <body>
    <table>
      <tbody>
        <tr>
          <th>CategoryID</th>
          <th>CategoryName</th>
          <th>Description</th>
          <th>In stock</th>
        </tr>
        <tr>
          <td>1</td>
          <td>Beverages</td>
          <td>Soft drinks, coffees &amp;amp; teas</td>
          <td>true</td>
        </tr>
        <tr>
          <td>2</td>
          <td>Condiments</td>
          <td>Sweet and savory sauces</td>
```

```
          <td>false</td>
        </tr>
        <tr>
          <td>3</td>
          <td>Confections</td>
          <td>Candies and sweet breads</td>
          <td>true</td>
        </tr>
      </tbody>
    </table>
  </body>
</html>
```

2.5.5 Markdown format

Easy Data Transform can output to tables in Markdown format files. File extension ".md".

Markdown format is commonly used as a human-friendly markup language, which can be automatically translated to HTML.

For example:

	CategoryID	CategoryName	Description	In stock
1	1	Beverages	Soft drinks, coffees & teas	true
2	2	Condiments	Sweet and savory sauces	false
3	3	Confections	Candies and sweet breads	true

Is output as:

```
| CategoryID | CategoryName | Description                 | In stock |
|------------|--------------|-----------------------------|----------|
| 1          | Beverages    | Soft drinks, coffees & teas | true     |
| 2          | Condiments   | Sweet and savory sauces     | false    |
| 3          | Confections  | Candies and sweet breads    | true     |
```

You can also use Markdown when you need a plain text version of your data, for example in a code comment.

Note that not all Markdown implementations support tables. If your implementation does not support tables, you may need to output to HTML instead.

2.5.6 TSV format

Easy Data Transform can input from and output to TSV format files. File extension ".tsv".

TSV (Tab Separated Value) format is commonly used for exchanging tabular data between programs.

TSV is a type of delimited text file format. Values are separated by tab characters. Tabs are not allowed within data values, so there is no need for quoting or escaping delimiters, as with CSV files. This means that TSV files are generally a bit more compact and faster to read and write than CSV files.

If you have a tab character in a value, Easy Data Transform will convert it to a space on output.

2.5.7 vCard format

Easy Data Transform can output to vCard format files. File extension ".vcf".

VCard format is commonly used as way of exchanging contact details between programs.

Note that you need to change the column header names to the values expected by vCard (using the Rename Cols transform).

For example:

	N	FN	ORG	TEL;TYPE=WORK,VOICE	ADR;TYPE=WORK,PREF
1	Gump;Forrest;;Mr.;	Forrest Gump	Bubba Gump Shrimp Co.	(111) 555-1212	100 Waters Edge;Baytown;

Is output as:

```
BEGIN:VCARD
VERSION:3.0
N:Gump;Forrest;;Mr.;
FN:Forrest Gump
ORG:Bubba Gump Shrimp Co.
TEL;TYPE=WORK,VOICE:(111) 555-1212
ADR;TYPE=WORK,PREF:100 Waters Edge;Baytown;LA;30314;United States of America
END:VCARD
```

2.5.8 XML format

Easy Data Transform can output to XML format files. File extension ".xml".

XML (Extensible Markup Language) format is commonly used for exchanging data between programs.

For example:

	CategoryID	CategoryName	Description	In stock
1	1	Beverages	Soft drinks, coffees & teas	true
2	2	Condiments	Sweet and savory sauces	false
3	3	Confections	Candies and sweet breads	true

Is output with **Values as** set to **Nodes** as:

```
<?xml version="1.0" encoding="UTF-8"?>
<root>
  <record>
    <CategoryID>1</CategoryID>
    <CategoryName>Beverages</CategoryName>
    <Description>Soft drinks, coffees &amp;amp; teas</Description>
    <In-stock>true</In-stock>
  </record>
  <record>
    <CategoryID>2</CategoryID>
    <CategoryName>Condiments</CategoryName>
    <Description>Sweet and savory sauces</Description>
    <In-stock>false</In-stock>
  </record>
  <record>
    <CategoryID>3</CategoryID>
    <CategoryName>Confections</CategoryName>
    <Description>Candies and sweet breads</Description>
    <In-stock>true</In-stock>
  </record>
</root>
```

Is output with **Values as** set to **Attributes** as:

```
<?xml version="1.0" encoding="UTF-8"?>
<root>
  <record CategoryID="1" CategoryName="Beverages" Description="Soft drinks, coffees &a
  <record CategoryID="2" CategoryName="Condiments" Description="Sweet and savory sauce
  <record CategoryID="3" CategoryName="Confections" Description="Candies and sweet bre
</root>
```

2.5.9 YAML format

Easy Data Transform can output to YAML format files. File extension ".yaml".

YAML (YAML Ain't Markup Language) format is commonly used for exchanging data between programs and for configuration files.

For example:

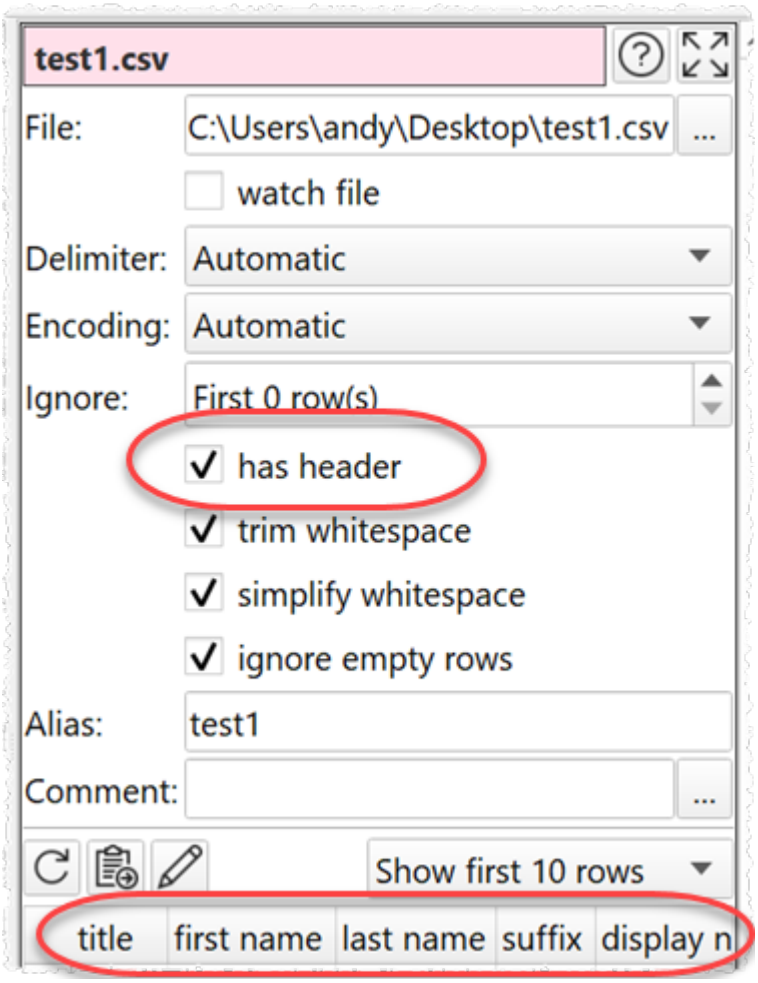
	CategoryID	CategoryName	Description	In stock
1	1	Beverages	Soft drinks, coffees & teas	true
2	2	Condiments	Sweet and savory sauces	false
3	3	Confections	Candies and sweet breads	true

Is output as:

```
---
-
  CategoryID: 1
  CategoryName: Beverages
  Description: Soft drinks, coffees & teas
  In stock: true
-
  CategoryID: 2
  CategoryName: Condiments
  Description: Sweet and savory sauces
  In stock: false
-
  CategoryID: 3
  CategoryName: Confections
  Description: Candies and sweet breads
  In stock: true
```

2.6 Headers

If the first row of an input is a header (i.e. one that describes the columns below) check **has header** for that input in the **Right** pane.



When you first read in a dataset Easy Data Transform will make a guess about whether the first row is a header (it will assume it is a header if it contains no dates or numbers).

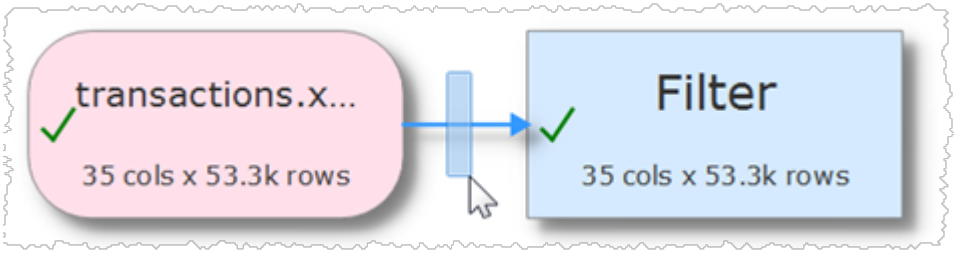
2.7 Connections

When you select an input or transform item and add a transform or output item, connections are added automatically.

To select a connection

To select a connection either:

- Click on the connection; or
- Click and drag a box over any part of the connection. This may be easier than clicking the connection when you are zoomed back.



To delete a connection

To delete a connection:

- Select the connection.
- Select **Edit>Delete** (or click the **Delete** tool bar button).

To add a transform to a connection

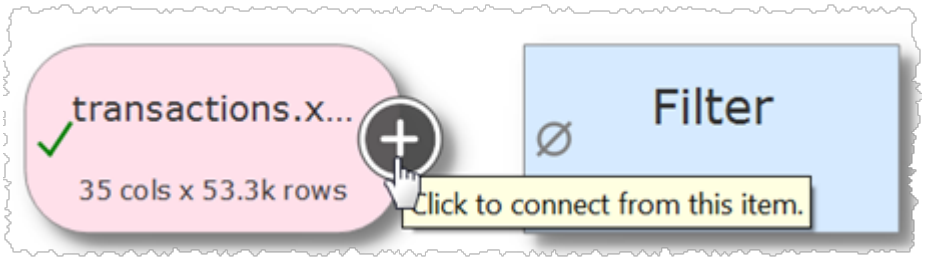
To add a transform between two already connected items:

- Select the connection.
- Choose the new transform from the **Left** pane or using the right click menu.

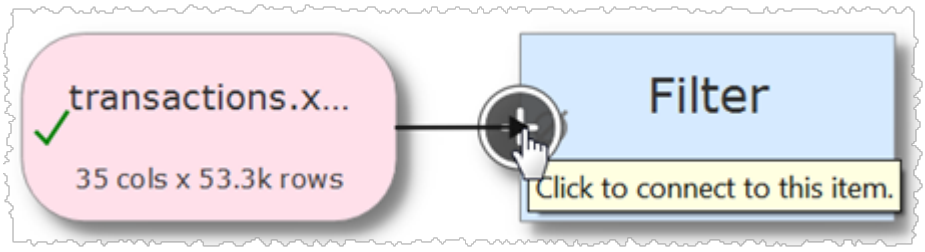
To add a connection

To add a new connection between two existing items:

- Hover over the start item.
- Click the '**+**' that appears.



- Hover over the end item
- Click the '**+**' that appears.



Press the 'Esc' key or click away from an item to cancel adding the connection.

Note that the '**+**' will only appear if an additional connection is allowed. For example you can't:

- Create a loop.
- Connect more than once from a transform.
- Connect more than once to an output.

2.8 Text

Whitespace (such as Space and Tab characters) and capitalization are always significant, unless stated otherwise.

You can remove leading and trailing white space by checking **trim whitespace** in the Input or using the Trim transform.

You can change the case using the Case transform.

2.9 Dates

Set the date formats you want to recognize in the **Preferences** window using the following format.

Format	Meaning
`d`	The day as number without a leading zero (1 to 31)
`dd`	The day as number with a leading zero (01 to 31)
`ddd`	The abbreviated localized day name (e.g. 'Mon' to 'Sun'). Uses the system locale to localize the name.
`dddd`	The long localized day name (e.g. 'Monday' to 'Sunday'). Uses the system locale to localize the name.
`M`	The month as number without a leading zero (1 to 12).
`MM`	The month as number with a leading zero (01 to 12)
`MMM`	The abbreviated localized month name (e.g. 'Jan' to 'Dec'). Uses the system locale to localize the name.
`MMMM`	The long localized month name (e.g. 'January' to 'December'). Uses the system locale to localize the name.
`yy`	The year as two digit number (00 to 99).
`yyyy`	The year as four digit number. If the year is negative, a minus sign is prepended in addition.

For example:

- To support a date such as 31/1/2019 add a supported date format: d/M/yyyy
- To support a date such as 1-31-19 add a supported date format: M-d-yy

Note that dates with only two year digits, are treated as a date between 1900 and 1999. E.g. "31/1/19" is interpreted in d-M-yy format as 31st January 1919.

Values that are in a recognized date format will be treated as dates in the Filter, If and Sort transforms. Supporting large numbers of date formats will slow down these transforms.

You can also change the format of dates using the Date Format transform.

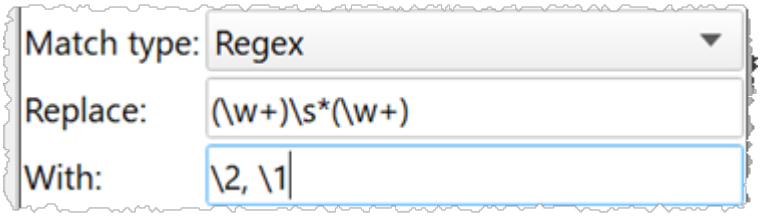
2.10 Numbers

Easy Data Transform uses the locale set on your computer to decide what is a number. For example, if your system locale is set to US or UK then "123.45" is a number and "123,45" isn't, and vice versa if your system locale is Germany or France.

2.11 Regular expressions

Easy Data Transform allows the use of regular expressions in the replace, if and filter transforms.

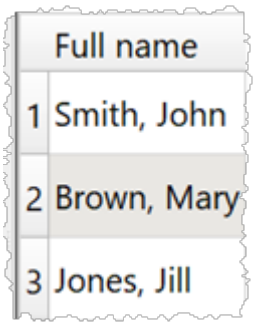
Regular expressions are a powerful way to match patterns in text (including text representation of dates and numbers). For example, you can use a regular expression in the Replace transform to swap first and last names:



Turns:

	Full name
1	John Smith
2	Mary Brown
3	Jill Jones

Into:

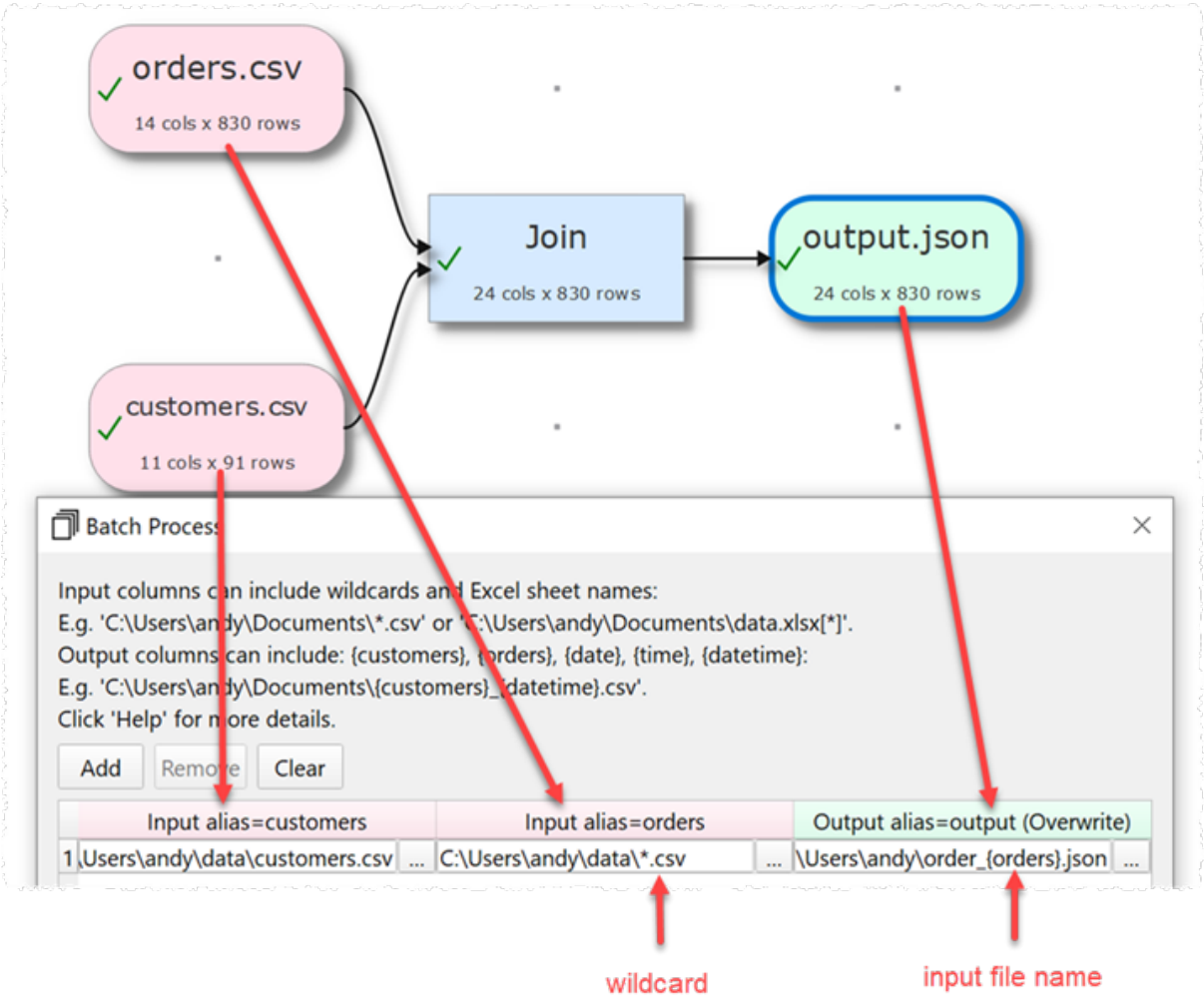


	Full name
1	Smith, John
2	Brown, Mary
3	Jones, Jill

Regular expressions are far too big a topic to cover here. However there are many detailed resources online, such as www.regular-expressions.info and regexr.com.

2.12 Batch processing

To apply the current transform template file to multiple input files select **File>Batch Process...** . The **Batch Process** window will appear with a column for each input item and a column for each output item. The **Alias** for each item is displayed in the column header.



Note:

- All input and output items must have an alias.
- An output item can't have the same alias as another output or input item.
- Output items with **Write mode**=Disabled are not shown.

Click **Add** to add a new processing row.

Click **Remove** to remove the selected processing row(s).

Click **Clear** to remove all processing rows.

In the (pink) input column you can use * and ? wildcards for file name stems, file extensions and Excel sheet names. E.g.:

Input	**Description**
`C:\Users\andy\Documents\*.c`	All files with extension .csv in the Documents folder

Input	Description
C:\Users\andy\Documents\d?.	All the files with name 'd' plus a single character in the Documents folder
C:\Users\andy\Documents\dat	All the sheets in data.xlsx in the Documents folder
C:\Users\andy\Documents*.x	All the sheets beginning with 'data' in all the .xslx files in the the Documents folder

Note:

- If there is more than 1 input column that specifies multiple files or sheets, then an output will be created for each possible permutation of input files/sheets in the row. E.g. 3 input files from column 1 x 4 sheets from column 2 = 12 outputs to process.
- Excel sheet names are not case sensitive.
- You cannot use wildcards for folder names.
- Batch processing will ignore files in sub-folders.

In the (green) output column you can use the following variables to create your output file name:

Output variable	Meaning	Example
{<input al	The name of the input file being processed in the column with the corresponding alias.	If input alias 'orders' is using file 'C:\Users\andy\Documents\orders_2020.csv' then '{orders}' is replaced with value 'orders_2020'. If input alias 'orders' is using file 'C:\Users\andy\Documents\orders_2020.xlsx' with sheet 'Sheet1' then '{orders}' is replaced with value 'orders_2020_Sheet1'.
{date}	Date processing was carried out in year_month_day format	2020_04_18
{time}	Time processing was carried out in hours_minutes_seconds_milliseconds format	15_21_56_599
{datetime}	Date/Time processing was carried out in year_month_day_hours_minutes_seconds_milliseconds format format	2020_04_18_15_21_56_599

Whether an ouput file is created, overwritten or appended to depends on the **Write mode** of the output item.

Click **Process** to start processing the rows.

Click **Stop** to stop processing the rows.

Click **Close** to close the window.

See also:
- Batch processing examples
- Command line arguments

2.13 Command line arguments

Easy Data Transform accepts the following command line arguments:

`<file nam`	The .transform file to open at start-up.
`-cli`	Close the application once any processing on the opened file is complete.
`-file <al`	Sets the input or output file with the given alias to the location (path) specified. Input Excel files should include the sheet name, e.g. `file.xlsx[sheet]`.
`-verbose`	Output additional information to the terminal.

This allows you to process .transform files in batch mode, e.g.:

```
"C:\Program Files (x86)\EasyDataTransform_v1\EasyDataTransform.exe" "C:\Users\andy\Do
"C:\Program Files (x86)\EasyDataTransform_v1\EasyDataTransform.exe" C:\Users\andy\Docu
```

Put quotes (") around any arguments with spaces (as shown in the examples above).

To do this on a schedule, call a .bat file from a scheduling program, such as Windows Task Scheduler.

See also:
- Batch processing

2.14 .transform files

.transforms file are stored in a simple XML format. So you can edit them with a standard text editor. However we recommend you make a copy first.

The results of transformations are not stored in the .transform file, and are recalculated whenever you **File>Open...** the file.

The contents of Input and Output files are not stored in the .transform file, only their locations. These locations are stored as 'absolute' locations, so you can move the .transform file without changing the locations of the Input and Output files.

If you open a .transform file in a different location from that in which it was saved and it can't find Input and Output files at the expected location it will look for them in the same location relative to the old .transform file. This allows you to easily move .transform files to different locations and computers if you keep the Input and Output files in the same relative location (e.g. in the same folder as the .transform file). This even works between Windows and Mac (and vice versa),

Example:

- `mytransform.transform` is in `C:\Users\andy\Documents\` on Windows and uses Input file `MyData.csv` in sub-folder `MyData` (`C:\Users\andy\Documents\Data\MyData.csv`).
- `mytransform.transform` is moved to `/Users/Bob/Documents/EDT` on a Mac.
- When `mytransform.transform` is opened it will look for `MyData.csv` in `/Users/andy/Documents/Data`.
- If it can't find that it will look for `MyData.csv` in sub-folder `MyData` (`/Users/Bob/Documents/EDT/Data/MyData.csv`).

If you paste in data **From Clipboard** this is stored in the .transform file. We don't recommend you do this for large datasets as XML is not very efficient for storing large amounts of data.

How do I?

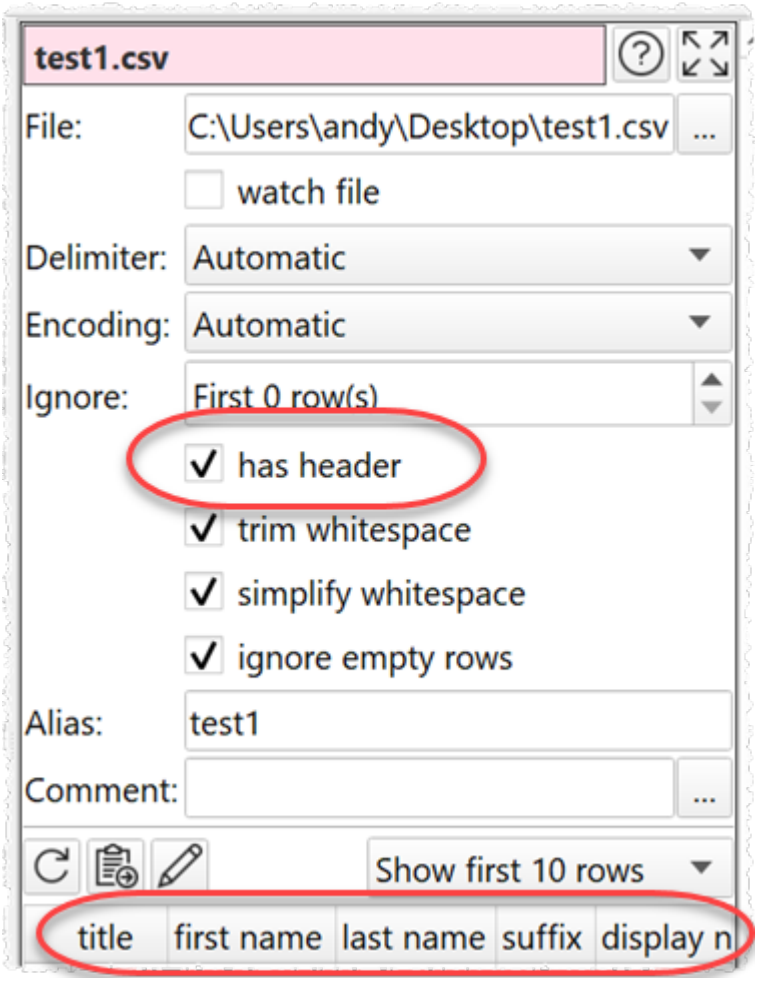
3 How do I?

3.1 Add a transform between existing items

To add a new transform between existing items (e.g. between 2 already connected transforms) see connections.

3.2 Add or remove a header

To add or remove a header just check or uncheck the **has header** checkbox for the appropriate input item.

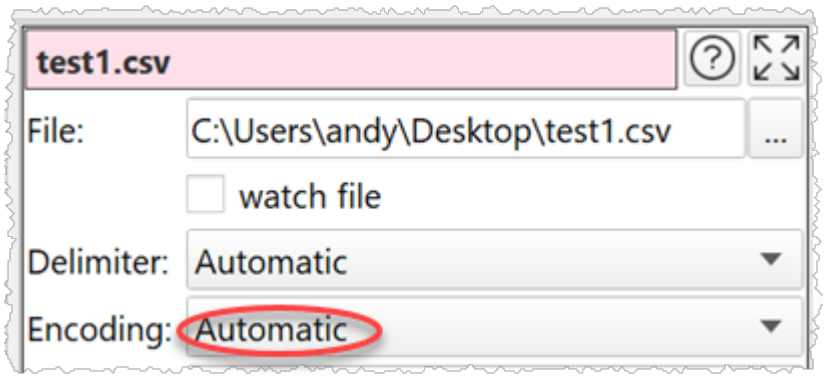


3.3 Change a connection

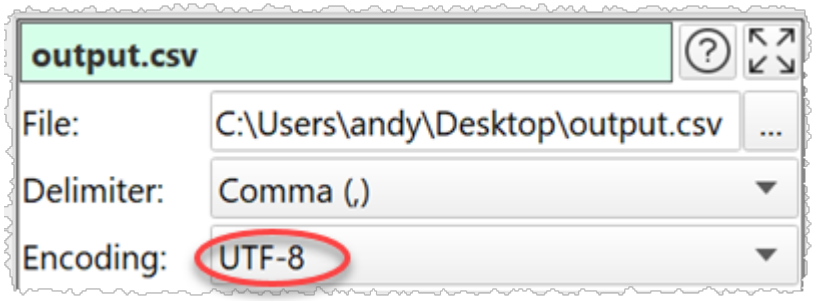
To change a connection see connections.

3.4 Change encoding

When Easy Data Transform inputs a text file (e.g. a CSV file) it will make a guess at the encoding. You can explicitly set the encoding by selecting an input item and changing **Encoding** from **Automatic** to one of the other encodings in the **Right** pane.



Similarly you can also set the encoding of a text file output by selecting the output item and changing **Encoding** in the **Right** pane.



3.5 Dedupe a dataset

If you want to remove duplicate entries from a dataset, use the Dedupe transform. For example, to remove the 2 rows that have the same email from this dataset:

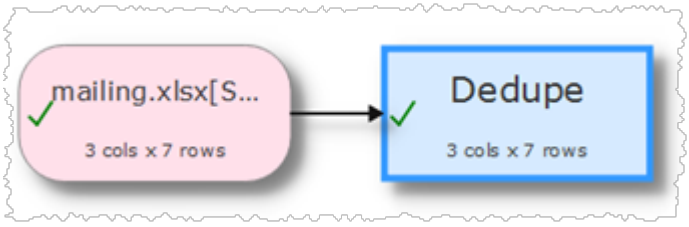
	First	Last	Email
1	J.A.	Black	jablack@gmail.com
2	Paul	White	p.white@hotmail.com
3	Barry	Green	bgreen@aol.com
4	Jane	Brown	Jb3423@gmail.com
5	J.	Taupe	taupe89759@gmail.com
6	John	Black	jablack@gmail.com
7	B.	Green	bgreen@aol.com

To get this dataset:

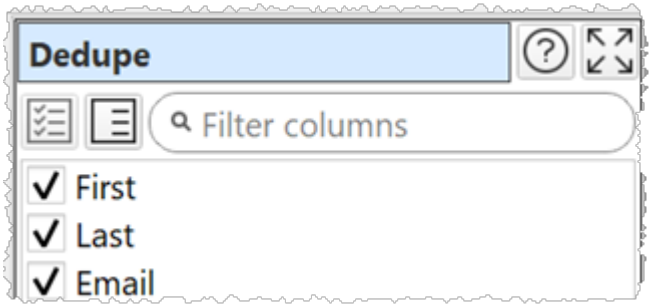
	First	Last	Email
1	J.A.	Black	jablack@gmail.com
2	Paul	White	p.white@hotmail.com
3	Barry	Green	bgreen@aol.com
4	Jane	Brown	Jb3423@gmail.com
5	J.	Taupe	taupe89759@gmail.com

Drag the dataset file onto the **Center** pane of Easy Data Transform.

Select the dataset then click the **Dedupe** transform in the **Left** pane.



Check **Email** in the **Right** pane to remove rows with duplicate emails.



Only the first row with a particular email is kept. Use Sort if you want to change the order before removing duplicates.

If you only want to remove rows with the same last name and same email, check both **Email** and **Last** checkboxes.

Note that de-duplicating columns takes account of whitespace and case. So you might need to do Trim and Case transforms before the dedupe.

3.6 Handle large datasets

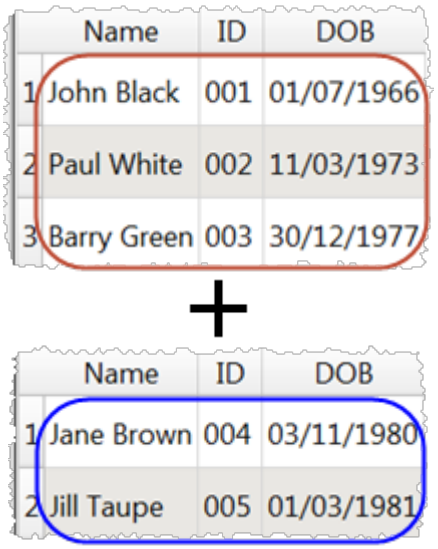
Large datasets (e.g. a million data points or more) can slow down updates. So we recommend you add a sample transform straight after the input and set **Rows** to pass through only the first 100 or so rows. Once you have completed all your transforms you can then change the sample transform to pass through all rows.

3.7 Merge datasets

Easy Data Transform has two options for merging two datasets. Stack and Join.

Stack datasets

If you want to merge the two datasets so they are one on top of another, use the Stack transform. For example, to Stack these two datasets:

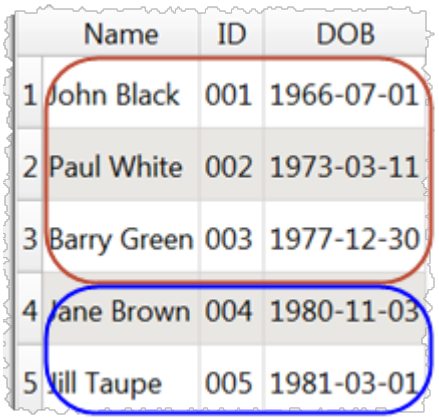


	Name	ID	DOB
1	John Black	001	01/07/1966
2	Paul White	002	11/03/1973
3	Barry Green	003	30/12/1977

+

	Name	ID	DOB
1	Jane Brown	004	03/11/1980
2	Jill Taupe	005	01/03/1981

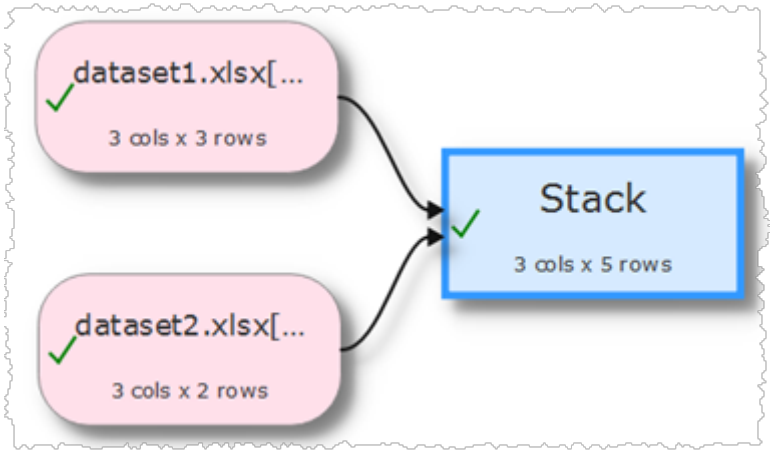
To get this dataset:



	Name	ID	DOB
1	John Black	001	1966-07-01
2	Paul White	002	1973-03-11
3	Barry Green	003	1977-12-30
4	Jane Brown	004	1980-11-03
5	Jill Taupe	005	1981-03-01

Drag the two dataset files onto the **Center** pane of Easy Data Transform.

Select the two datasets using `Ctrl`+click then click the **Stack** transform in the **Left** pane.



The datasets are now stacked in the vertical order that the datasets are shown on the screen. The top dataset is shown first. You can swap the the vertical positions of the datasets to change the order in which they are stacked.

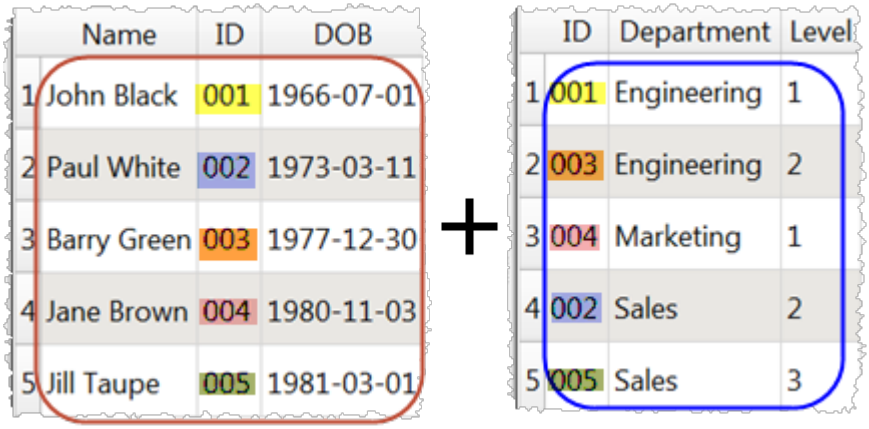
If you want to stack column n of the first dataset above column n of the second dataset, set **Align columns by** to **Column number**.

If you want to stack columns by common header names (even if they aren't in the same order), set **Align columns by** to **Header name**.

If you want to stack a large number of files you can do it by using batch processing to write to an output item with **Write Mode**=Append.

Join datasets

If you want to merge the two datasets side-by-side using a common ('key') column, use the Join transform. For example, to Join these two datasets:

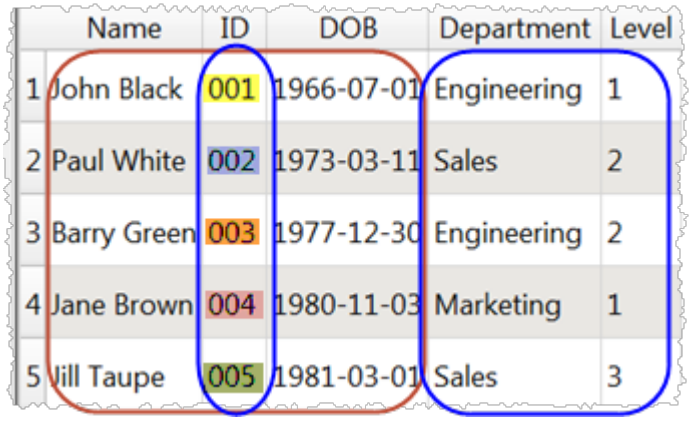


	Name	ID	DOB
1	John Black	001	1966-07-01
2	Paul White	002	1973-03-11
3	Barry Green	003	1977-12-30
4	Jane Brown	004	1980-11-03
5	Jill Taupe	005	1981-03-01

+

	ID	Department	Level
1	001	Engineering	1
2	003	Engineering	2
3	004	Marketing	1
4	002	Sales	2
5	005	Sales	3

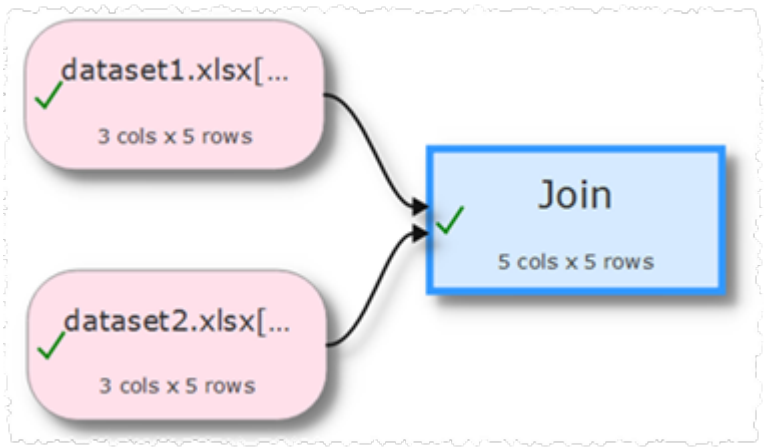
By common ID value to get this dataset:



	Name	ID	DOB	Department	Level
1	John Black	001	1966-07-01	Engineering	1
2	Paul White	002	1973-03-11	Sales	2
3	Barry Green	003	1977-12-30	Engineering	2
4	Jane Brown	004	1980-11-03	Marketing	1
5	Jill Taupe	005	1981-03-01	Sales	3

Drag the two dataset files onto the **Center** pane of Easy Data Transform.

Select the two datasets using `Ctrl`+click then click the **Join** transform in the **Left** pane.



Set both **Top key column** and **Bottom key column** to the common ('key') column.

The datasets are now joined side-by-side using the common column. The top dataset is shown on the left. You can swap the the vertical positions of the datasets to change the order in which they are joined.

If you just want to join row N of one dataset to row N of another dataset, you can use the Row Num transform to create a common column in each dataset.

Set **Include top non-matching rows** and **Include bottom non-matching rows** depending on what you want to do with top and bottom dataset rows for which there are no matches.

Note that matching columns takes account of whitespace and case. So you might need to do Trim and Case transforms before the join.

3.8 Move a .transform file

To move a .transform file to a different location on the same computer use **File>Save As...** or **Windows Explorer**. You either leave the Input files at the original location or move them to the same location relative to the .transform file (e.g. if they were in the same folder as the .transform file before, move them to the same folder as new .transform file).

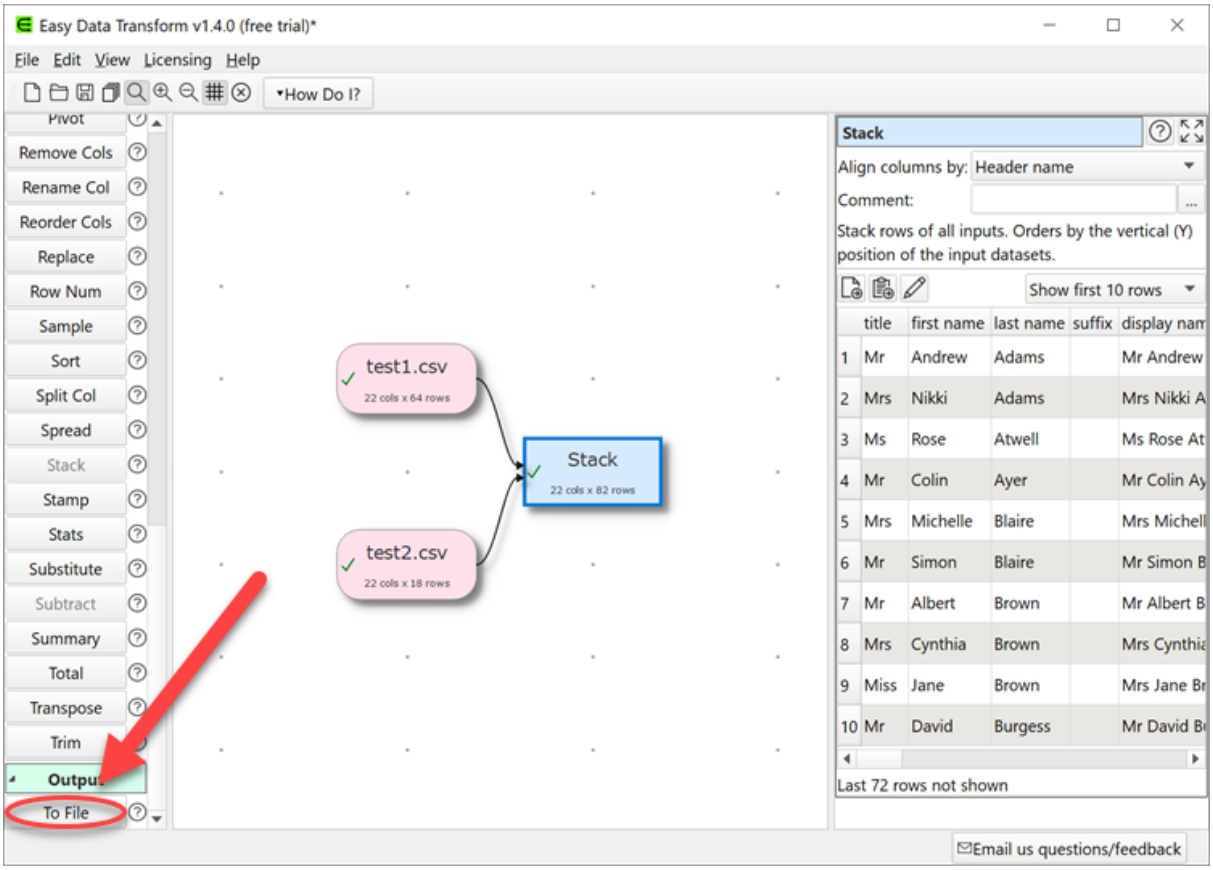
To move a .transform file to a different computer, move the Input files to the same location relative to the .transform file (e.g. if they were in the same folder as the .transform file before, move them to the same folder as new .transform file).

See also .transform files.

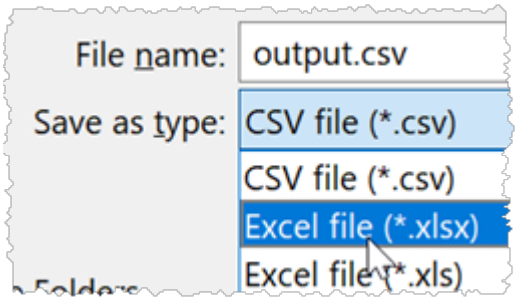
3.9 Output to Excel

To output results from a transform to an Excel .xlsx/.xls file:

- Select the transform item in the **Center** pane.
- Click **To File** at the bottom of the **Left** pane.



- Select ***.xlsx** or ***.xls** from the file type drop-down list that appears.



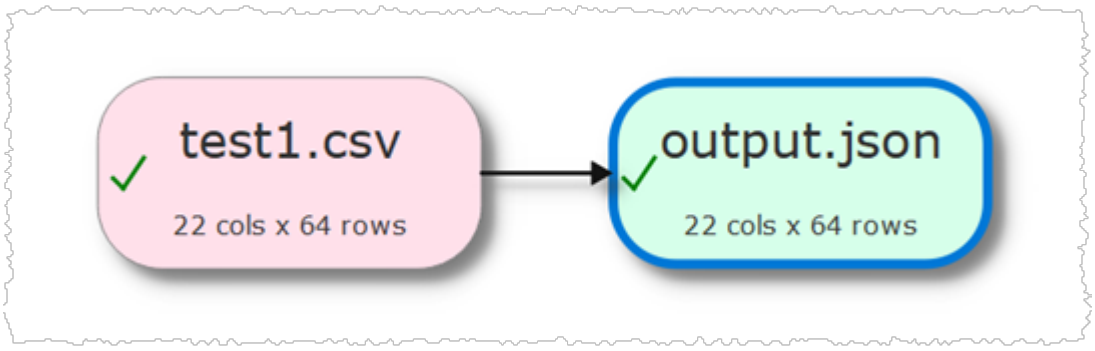
3.10 Perform the same transforms on many files

You can perform the same set of transforms on multiple inputs in one operation using batch Processing or command line arguments.

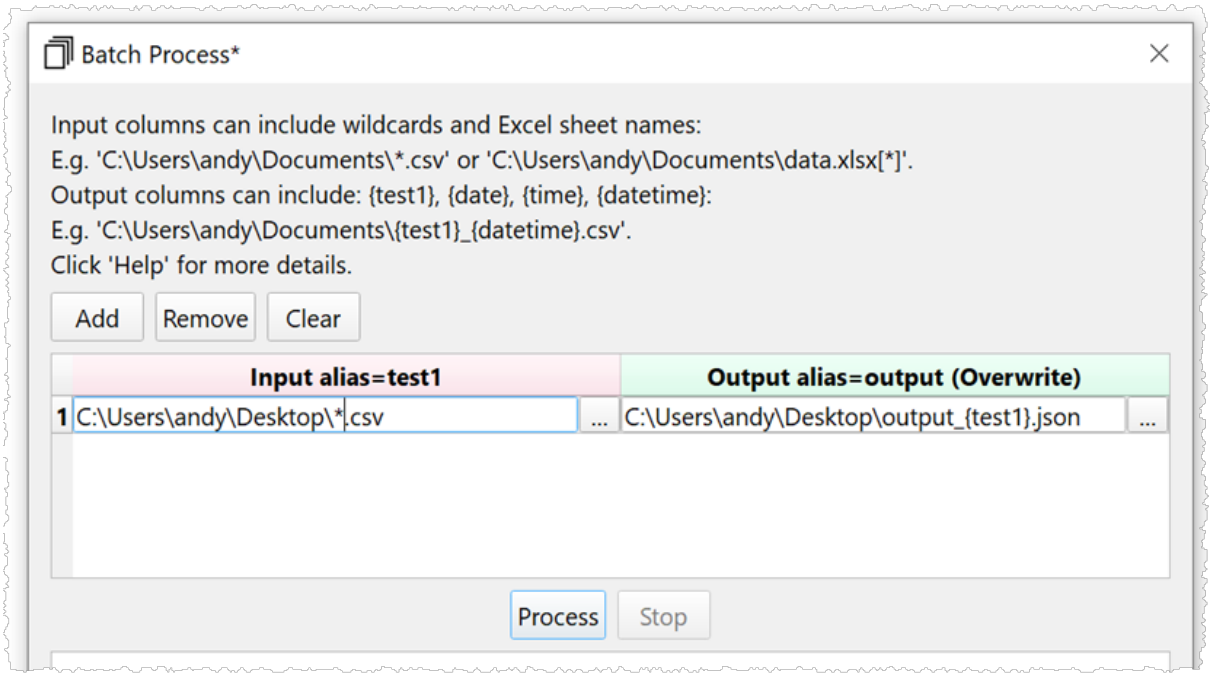
Example 1

To convert a folder full of .csv files to .json files:

1. Select **File>New** to create a new transform template file
2. Drag one of the .csv files onto the **Center pane**. Ensure the options (encoding etc) are correct in the **Right** pane.
3. Click on the **To File** button at the bottom of the **Left** pane and set the location of a .json file to create. Ensure the options (encoding etc) are correct in the **Right** pane.



4. Select **File>Batch Process**.
5. In the **Batch Process** window change the .csv file name to `*.csv` and `output.json` to `output_{test1}.json`.



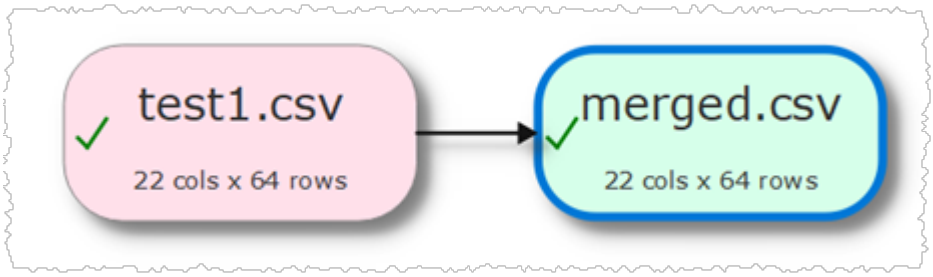
6. Press the **Process** button. A .json file will now be created for each .csv file in the folder.

If you want to process input files from another folder then click **Add** to add a new row and change the **test1** input folder.

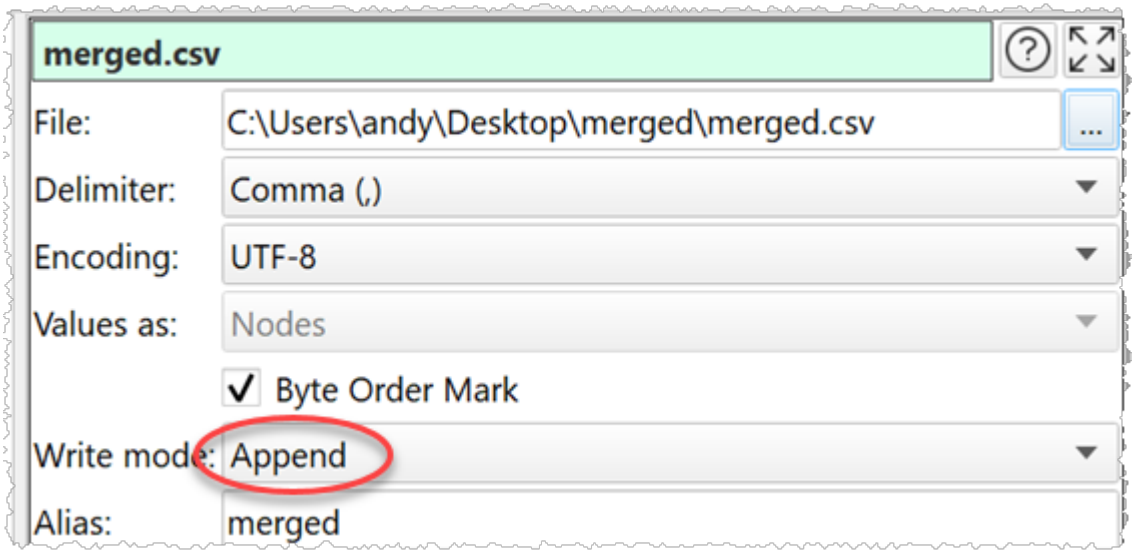
Example 2

Merge multiple .csv files into a single .csv file:

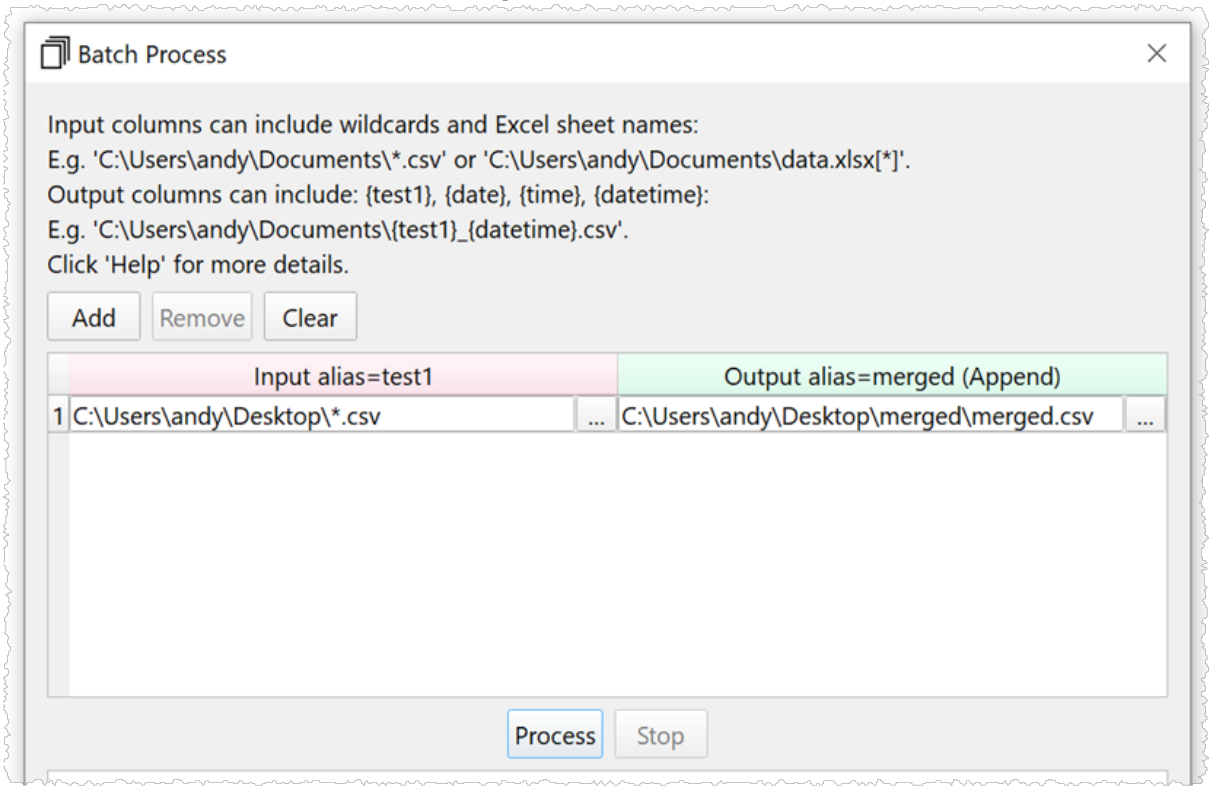
7. Select **File>New** to create a new transform template file
8. Drag one of the .csv files onto the **Center pane**. Ensure the options (encoding etc) are correct in the **Right** pane.
9. Click on the **To File** button at the bottom of the **Left** pane and set the location of a `merged.csv` file to create, in a different folder to the input .csv files. Ensure the options (encoding etc) are correct.



4. Set **Write Mode** to **Append** in the **Right** pane.

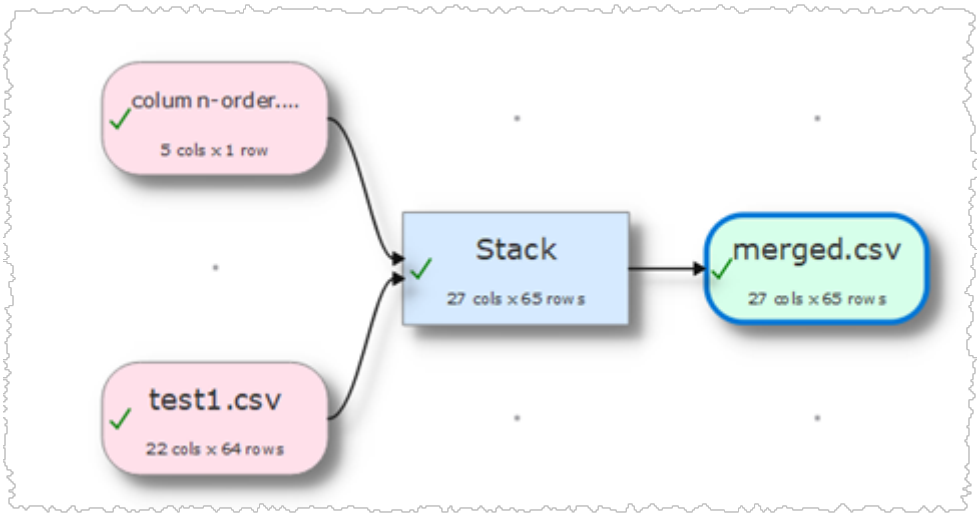


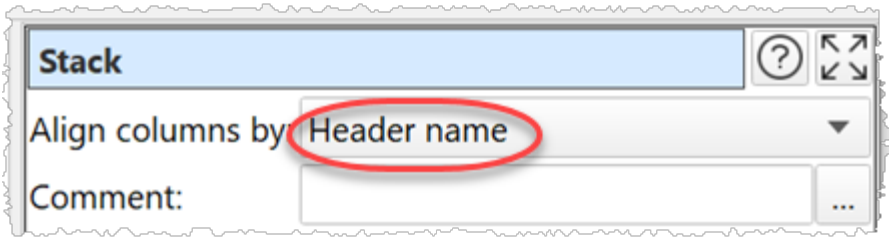
5. Select **File>Batch Process**.
6. In the **Batch Process** window change the input .csv file name to `*.csv`.



7. Press the **Process** button. A single `merged.csv` file will now be created that contains a concatenation of all the other .csv files. If `merged.csv` already exists, you may need to delete it first.

If the headers are different orders in different .csv files, then you can Stack by header name to get a consistent column order before outputting.





You can use a Filter transform on the output to remove any repeated headers.

Support

4 Support

4.1 Contact support

If you have any questions or suggestions, please contact us at support@easydatatransform.com .

4.2 Report a bug

Please report any bugs you find to support@easydatatransform.com and we will attempt to fix them. Please include:

- a description of the bug
- your operating system (e.g. Windows 10)
- the version of Easy Data Transform (from **Help>About**)
- a step-by-step description of how we can reproduce the problem
- a screen capture can often be helpful

The step-by-step description is particularly important - if we can't reproduce your problem, then we probably won't be able to fix it.

4.3 Request an enhancement

We are always very interested to hear your suggestions on how the software can be improved. Please email us at support@easydatatransform.com .

- . -

- B -

- C -

- D -

- E -

- F -

- G -

- H -

- I -

- J -

- L -

- M -

- N -

- O -

- P -

- Q -

- R -

- S -

- T -

- V -

- W -

- X -

- Y -