EXAMPLE - Replacement Transforms

This example illustrates the different uses of the following transformations to replace or extract cell data:

- `set` - defines the values to use in a predefined column. See *Set Transform*.

  Tip: Use the `derive` transform to generate a new column containing a defined set of values. See *Derive Transform*.

- `replace` - replaces a string literal or pattern appearing in the values of a column with a specific string. See *Replace Transform*.
- `extract` - extracts a pattern-based value from a column and stores it in a new column. See *Extract Transform*.

Source:

The following dataset contains contact information that has been gathered by your marketing platform from actions taken by visitors on your website. You must clean up this data and prepare it for use in an analytics platform.

LeadId	LastName	FirstName	Title	Phone	Request
LE160301001	Jones	Charles	Chief Technical Officer	415-555-1212	reg
LE160301002	Lyons	Edward		415-012-3456	download whitepaper
LE160301003	Martin	Mary	CEO	510-555-5555	delete account
LE160301004	Smith	Talia	Engineer	510-123-4567	free trial

Transformation:

Title column: For example, you first notice that some data is missing. Your analytics platform recognizes the string value, "`#MISSING#`" as an indicator of a missing value. So, you click the missing values bar in the Title column. Then, you select the Replace suggestion card. Note that the default replacement is a null value, so you click **Edit** and update it:

Transformation Name	`Edit column with formula`
Parameter: Columns	`Title`
Parameter: Formula	`if(ismissing([Title]),'#MISSING#',Title)`

Request column: In the Request column, you notice that the `reg` entry should be cleaned up. Add the following transformation, which replaces that value:

Transformation Name	`Replace text or pattern`
Parameter: Column	`Request`
Parameter: Find	``` `{start}reg{end}` ```
Parameter: Replace with	`Registration`

The above transformation uses a Pattern as the expression of the `on:` parameter. This expression indicates to match from the start of the cell value, the string literal `reg`, and then the end of the cell value, which matches on complete cell values of `reg` only.

This transformation works great on the sample, but what happens if the value is `Reg` with a capital `R`? That value might not be replaced. To improve the transformation, you can modify the transformation with the following Pattern in the on parameter, which captures differences in capitalization:

Transformation Name	`Replace text or pattern`
Parameter: Column	`Request`
Parameter: Find	``` `{start}{[R|r]}eg{end}` ```
Parameter: Replace with	`'Registration'`

Add the above transformation to your recipe. Then, it occurs to you that all of the values in the `Request` column should be capitalized in title or proper case:

Transformation Name	`Edit column with formula`
Parameter: Columns	`Request`
Parameter: Formula	`proper(Request)`

Now, all values are capitalized as titles.

Phone column: You might have noticed some issues with the values in the `Phone` column. In the United States, the prefix `555` is only used for gathering information; these are invalid phone numbers.

In the data grid, you select the first instance of `555` in the column. However, it selects all instances of that pattern, including ones that you don't want to modify. In this case, continue your selection by selecting the similar instance of `555` in the other row. In the suggestion cards, you click the Replace Text or Pattern transformation.

Notice, however, that the default Replace Text or Pattern transformation has also highlighted the second `555` pattern in one instance, which could be a problem in other phone numbers not displayed in the sample. You must modify the selection pattern for this transformation. In the `on:` parameter below, the Pattern has been modified to match only the instances of `555` that appear in the second segment in the phone number format:

Transformation Name	`Replace text or pattern`
Parameter: Column	`Phone`
Parameter: Find	``` `{start}%{3}-555-%*{end}` ```
Parameter: Replace with	`'#INVALID#'`
Parameter: Match all occurrences	`true`

Note the wildcard construct has been added (`%*`). While it might be possible to add a pattern that matches on the last four characters exactly (`%{4}`), that matching pattern would not capture the possibility of a phone number having an extension at the end of it. The above expression does.

NOTE: The above transformation creates values that are mismatched with the Phone Number data type. In this example, however, these mismatches are understood to be for the benefit of the system consuming your Trifacta output.

LeadId column: You might have noticed that the lead identifier column (`LeadId`) contains some embedded information: a date value and an identifier for the instance within the day. The following steps can be used to break out this information. The first one creates a separate working column with this information, which allows us to preserve the original, unmodified column:

Transformation Name	`New formula`
Parameter: Formula type	`Single row formula`
Parameter: Formula	`LeadId`
Parameter: New column name	`'LeadIdworking'`

You can now work off of this column to create your new ones. First, you can use the following replace transformation to remove the leading two characters, which are not required for the new columns:

Transformation Name	`Replace text or pattern`
Parameter: Column	`LeadIdworking`
Parameter: Find	`'LE'`
Parameter: Replace with	`''`

Notice that the date information is now neatly contained in the first characters of the working column. Use the following to extract these values to a new column:

Transformation Name	`Extract text or pattern`
Parameter: Column to extract from	`LeadIdworking`
Parameter: Option	`Custom text or pattern`
Parameter: Text to extract	``` `{start}%{6}` ```

The new `LeadIdworking2` column now contains only the date information. Cleaning up this column requires reformatting the data, retyping it as a Datetime type, and then applying the `dateformat` function to format it to your satisfaction. These steps are left as a separate exercise.

For now, let's just rename the column:

Transformation Name	`Rename columns`
Parameter: Option	`Manual rename`
Parameter: Column	`LeadIdworking1`
Parameter: New column name	`'LeadIdDate'`

In the first working column, you can now remove the date information using the following:

Transformation Name	`Replace text or pattern`
Parameter: Column	`LeadIdworking`

Parameter: Find	`` `{start}%{6}` ``
Parameter: Replace with	`''`

You can rename this column to indicate it is a daily identifier:

Transformation Name	`Rename columns`
Parameter: Option	`Manual rename`
Parameter: Column	`LeadIdworking`
Parameter: New column name	`'LeadIdDaily'`

Results:

LeadId	LeadIdDaily	LeadIdDate	LastName	FirstName	Title	Phone	Request
LE160301001	001	160301	Jones	Charles	Chief Technical Officer	#INVALID#	Registration
LE160301002	002	160301	Lyons	Edward	#MISSING#	415-012-3456	Download Whitepaper
LE160301003	003	160301	Martin	Mary	CEO	#INVALID#	Delete Account
LE160301004	004	160301	Smith	Talia	Engineer	510-123-4567	Free Trial