

EXAMPLE - Extractlist Transform

Source:

The following dataset contains counts of support emails processed by each member of the support team for individual customers over a six-month period. In this case, you are interested in the total number of emails processed for each customer.

Unfortunately, the data is ragged, as there are no entries for a support team member if he or she has not answered an email for a customer. So, the `unnest` transform is not usable.

custId	startDate	endDate	supportEmailCount
C001	7/15/2015	12/31/2015	["Max":2,"Ted":0,"Sally":12,"Jack":6,"Sue":4]
C002	7/15/2015	12/31/2015	["Sally":4,"Sue":3]
C003	7/15/2015	12/31/2015	["Ted":12,"Sally":2]
C004	7/15/2015	12/31/2015	["Jack":7,"Sue":4,"Ted":5]

If the data is imported from a CSV file, you might need to make some simple `replace` transforms to clean up the data to look like the above example.

Transform:

Use the following transform to extract just the numeric values from the `supportEmailCount` array:

```
extractlist col:supportEmailCount on:`{digit}+`
```

You should now have a column `extractlist_supportEmailCount` containing a ragged array. You can use the following transforms to convert this data to a comma-separated list of values:

```
replace col: extractlist_supportEmailCount on: `[` with: `,` global: true
```

```
replace col: extractlist_supportEmailCount on: `]` with: `,` global: true
```

```
replace col: extractlist_supportEmailCount on: `"` with: `,` global: true
```

Convert the column to String data type.

You can now split out the column into separate columns containing individual values in the modified source. The `limit` parameter specifies the number of splits to create, resulting in 5 new columns, which is the maximum number of entries in the source arrays.

```
split col:extractlist_supportEmailCount on:', ' limit:4
```

You might have to set the type for each generated column to Integer. If you try to use a `derive` transform to calculate the sum of all of the generated columns, it only returns values for the first row because the missing rows are null values.

In the columns containing null values, select the missing value bar in the data histogram. Select the Replace suggestion card, and modify the transform to write a 0 in place of the null value, as follows:

```
set col: extractlist_supportEmailCount3 value: '0' row: ISMISSING  
([extractlist_supportEmailCount3])
```

Repeat this step for any other column containing null values.

You can now use the following to sum the values in the generated columns:

```
derive type:single value:(extractlist_supportEmailCount1 + extractlist_supportEmailCount2  
+ extractlist_supportEmailCount3 + extractlist_supportEmailCount4 +  
extractlist_supportEmailCount5)
```

Results:

After renaming the generated column to `totalSupportEmails` and dropping the columns used to create it, your dataset should look like the following:

custId	startDate	endDate	supportEmailCount	totalSupportEmails
C001	7/15/2015	12/31/2015	["Max":2,"Ted":0,"Sally":12,"Jack":6,"Sue":4]	24
C002	7/15/2015	12/31/2015	["Sally":4,"Sue":3]	7
C003	7/15/2015	12/31/2015	["Ted":12,"Sally":2]	14
C004	7/15/2015	12/31/2015	["Jack":7,"Sue":4,"Ted":5]	16