EXAMPLE - IF Data Type Validation Functions

This example illustrates how to use the IF* functions for data type validation.

Functions:

Item	Description
IFNULL Function	The `IFNULL` function writes out a specified value if the source value is a null. Otherwise, it writes the source value. Input can be a literal, a column reference, or a function.
IFMISSING Function	The `IFMISSING` function writes out a specified value if the source value is a null or missing value. Otherwise, it writes the source value. Input can be a literal, a column reference, or a function.
IFMISMATCHED Function	The `IFMISMATCHED` function writes out a specified value if the input expression does not match the specified data type or typing array. Otherwise, it writes the source value. Input can be a literal, a column reference, or a function.
IFVALID Function	The `IFVALID` function writes out a specified value if the input expression matches the specified data type. Otherwise, it writes the source value. Input can be a literal, a column reference, or a function.
MERGE Function	Merges two or more columns of String type to generate output of String type. Optionally, you can insert a delimiter between the merged values.

Source:

The following simple table lists zip codes by customer identifier:

custId	custZip
C001	98123
C002	94105
C003	12415
C004	12451-2234
C005	12441-298
C006	
C007	
C008	1242
C009	1104

Transformation:

When the above is imported into the Transformer page, you notice the following:

- The `custZip` column is typed as Integer.
- There are two missing and two mismatched values in the `custZip` column.

First, you test for valid values in the `custZip` column. Using the `IFVALID` function, you can validate against any data type:

Transformation Name	`New formula`
Parameter: Formula type	`Single row formula`
Parameter: Formula	`IFVALID(custZip, 'Zipcode', 'ok')`

Parameter: New column name	`'status'`

Fix four-digit zips: In the `status` column are instances of `ok` for the top four rows. You notice that the bottom two rows contain four-digit codes.

Since the `custZip` values were originally imported as Integer, any leading `0` values are deleted. In this case, you can add back the leading zero. Before the previous step, change the data type of `zip` to String and insert the following:

Transformation Name	`New formula`
Parameter: Formula type	`Single row formula`
Parameter: Formula	`IF(LEN(custZip)==4,'0','')`
Parameter: New column name	`'FourDigitZip'`

Transformation Name	`New formula`
Parameter: Formula type	`Single row formula`
Parameter: Formula	`MERGE([FourDigitZip,custZip])`
Parameter: New column name	`'custZip2'`

Transformation Name	`Edit column with formula`
Parameter: Columns	`zip`
Parameter: Formula	`custZip2`

Transformation Name	`Delete columns`
Parameter: Columns	`FourDigitZip,custZip2`
Parameter: Action	`Delete selected columns`

Now, when you click the last recipe step, you should see that two more rows in `status` are listed as `Ok`.

For the zip code with the three-digit extension, you can simply remove that extension to make it valid. Click the step above the last one. In the data grid, highlight the value. Click the Replace suggestion card. Select the option that uses the following for the matching pattern:

```
'-{digit}{3}{end}'
```

The above means that all three-digit extensions are deleted from the zip. You can do the same for any two- and one-digit extensions, although there are none in this sample.

Missing and null values: Now, you need to address how to handle missing and null values. The `IFMISSING` tests for both missing and null values, while the `IFNULL` tests just for null values. In this example, you want to delete null values, which could mean that the data for that row is malformed and to write a status of `missing` for missing values.

Click above the last line in the recipe to insert the following:

Transformation Name	Edit column with formula
Parameter: Columns	custZip
Parameter: Formula	IFNULL(custZip, 'xxxxx')

Transformation Name	Edit column with formula
Parameter: Columns	custZip
Parameter: Formula	IFMISSING(custZip, '00000')

Now, when you click the last line of the recipe, only the null value is listed as having a status other than `ok`. You can use the following to remove this row and all like it:

Transformation Name	Filter rows
Parameter: Condition	Custom formula
Parameter: Type of formula	Custom single
Parameter: Condition	(status == 'xxxxx')
Parameter: Action	Delete matching rows

Results:

custId	custZip	status
C001	98123	ok
C002	94105	ok
C003	12415	ok
C004	12451-2234	ok
C005	12441-298	ok
C006	00000	ok
C008	1242	ok
C009	1104	ok

As an exercise, you might repeat the above steps starting with the `IFMISMATCHED` function determining the value in the `status` column:

Transformation Name	New formula
Parameter: Formula type	Single row formula
Parameter: Formula	IFMISMATCHED(custZip, 'Zipcode', 'mismatched')
Parameter: New column name	'status'