

Nest Transform

Contents:

- *Basic Usage*
- *Syntax and Parameters*
 - *col*
 - *into*
 - *as*
- *Examples*

i **NOTE:** Transforms are a part of the underlying language, which is not directly accessible to users. This content is maintained for reference purposes only. For more information on the user-accessible equivalent to transforms, see *Transformation Reference*.

Creates an Object or Array of values using column names and their values as key-value pairs for one or more columns. Generated column type is determined by the `into` parameter.

The `nest` transform is the opposite of `unnest`, which unpacks Object data into separate columns and rows. See *Unnest Transform*.

Basic Usage

ItemA	ItemB
22	33
44	55

Object example:

```
nest col:ItemA,ItemB into:'obj' as:'myObj'
```

Output: See below.

ItemA	ItemB	myObj
22	33	{"ItemA":"22","ItemB","33"}
44	55	{"ItemA":"44","ItemB","55"}

Array example:

```
nest col:ItemA,ItemB into:'array' as:'myArray'
```

Output: Output arrays do not include the column name.

ItemA	ItemB	myArray
22	33	["22","33"]
44	55	["44","55"]

Syntax and Parameters

```
nest col:column_ref [into: object|array] [as:'new_column_name']
```

Token	Required?	Data Type	Description
nest	Y	transform	Name of the transform
col	Y	string	Source column name
into	N	string	Data type of output column: <code>object</code> (default) or <code>array</code>
as	N	string	Name of newly generated column

For more information on syntax standards, see *Language Documentation Syntax Notes*.

col

Identifies the column or columns to which to apply the transform. You can specify one column or more columns.

To specify multiple columns:

- Discrete column names are comma-separated.
- Values for column names are case-sensitive.

For each listed column, a new pair of key and value columns is generated.

```
nest col: Qty, Amount
```

Output: Builds an Object of the data from the columns `Qty` and `Amount` .

You can also specify ranges of columns using the tilde (~) operator:

```
nest col:Column1~Column20 as:'bigNest'
```

Output: Nests the data from columns `Column1` and `Column20` and all columns displayed in between them in the data grid into the new column `bigNest`.

Usage Notes:

Required?	Data Type
Yes	String (column name)

into

Defines the output column type. Accepted values:

- `object`
- `array`

If this parameter is not specified, the output type is `Object`.

Usage Notes:

Required?	Data Type
No (Object is default)	String (data type name)

as

Name of the new column that is being generated. If the `as` parameter is not specified, a default name is used.


```
nest col: CustId,ProdId as:'masterNest'
```

Output: Nests the data from the columns `CustId` and `ProdId` into a new column called, `masterNest`.

Usage Notes:

Required?	Data Type
No	String (column name)

Examples

 **Tip:** For additional examples, see *Common Tasks*.

Source:

In the following example, furniture product dimensions are stored in separate columns in cm.

Category	Name	Length	Width	Height
bench	Hooska	118.11	74.93	46.34
lamp	Tansk	30.48	30.48	165.1
bookshelf	Brock	27.94	160.02	201.93
couch	Loafy	95	227	83

Transform:

Use the `nest` transform to bundle the data into a single column.

```
nest col:Length~Height as:'Dimensions'
```

Results:

Category	Name	Length	Width	Height	Dimensions
bench	Hooska	118.11	74.93	46.34	{"Length_cm":"118.11","Width_cm":"74.93","Height_cm":"46.34"}
lamp	Tansk	30.48	30.48	165.1	{"Length_cm":"30.48","Width_cm":"30.48","Height_cm":"165.1"}
bookshelf	Brock	27.94	160.02	201.93	{"Length_cm":"27.94","Width_cm":"160.02","Height_cm":"201.93"}
couch	Loafy	95	227	83	{"Length_cm":"95","Width_cm":"227","Height_cm":"83"}