**tMap Joins & Filtering**

## tMap Component Joins & Filtering

In our article on the [tMap (Transformation)](https://www.talendbyexample.com/talend-tmap-component-reference.html) component, we introduced the component and it's core functionality of transforming input data to output data. The [tMap](https://www.talendbyexample.com/talend-tmap-component-reference.html) component provides much more functionality beyond this, including [Joins](http://en.wikipedia.org/wiki/Relational_algebra" \l ".CE.B8-join_and_equijoin)and **Filtering**.

## Join Basics

In any join performed by Talend, one input acts as the *primary* input, while the other acts as the *look-up*. Prior to reading the *primary* input, the *look-up* input will be read (by default) in to memory, in its entirety. As each *primary* input row is processed, a *look-up* will be performed. Talend allows this processing to be configured, depending on your requirements and data volume.

Let's look at a simple example, where we have a following text file of *People* and we want to join them to a *Postal Address* file.

### Person File ( create fileby the name of PersonFile.txt)

Id;Title;FirstName;LastName;AddressId

1;Mr;Austin;Patel;4

2;Ms;Sophia;Watson;2

3;Mr;Ewan;Parker;3

4;Ms;Evie;Cunningham;2

5;Mr;Dexter;Booth;

### Address File (create file by the name of AddressFile.txt)

Id;Street;Town;County;Postcode

1;19 West Close;Shefford;West Sussex;WE24 8ST

2;140 Great Square;Slough;Cornwall;CO43 3RN

3;178 North Close;Biggleswade;Hereford and Worcester;HE91 7RE

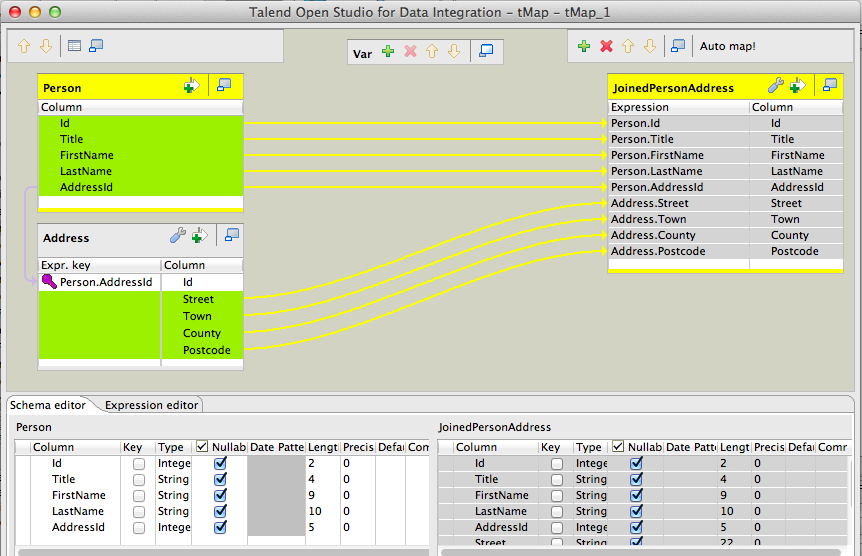
4;89 Windsor Street;Warrington;Gloucestershire;GL38 5OU

5;153 Dee Avenue;Shefford;County Down;CO63 5UN

### Example Job #1

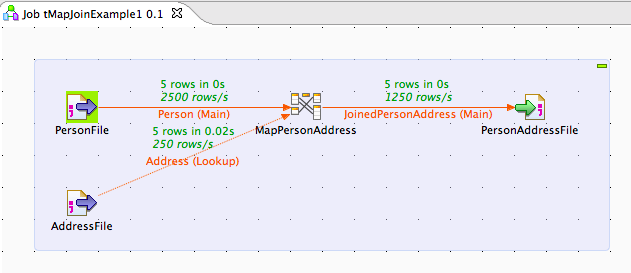
As can be seen from the following screenshot, these two files have been added to a Job, and have been joined on Address.Id. You can make this join by either dragging the column from the *primary* input to the *look-up* input, or manually inputting the column name. All columns have been added to our output, except for Address.Id; which becomes redundant.

Note: In tMap one source act as a primary input and while other act as a look-up



#### Running Example Job #1

The following screenshot shows the successful execution of this Job.



#### Example Job #1 - Review

There are some key points to note, when reviewing our output file.

* Addresses Id=1, Id=5 did not appear appear in the person file.
* Dexter Booth Id=5 does not have an Address Id and, therefore, does not match a row in the address file.
* Two people have AddressId=2.

Id;Title;FirstName;LastName;AddressId;Street;Town;County;Postcode

1;Mr;Austin;Patel;4;89 Windsor Street;Warrington;Gloucestershire;GL38 5OU

2;Ms;Sophia;Watson;2;140 Great Square;Slough;Cornwall;CO43 3RN

3;Mr;Ewan;Parker;3;178 North Close;Biggleswade;Hereford and Worcester;HE91 7RE

4;Ms;Evie;Cunningham;2;140 Great Square;Slough;Cornwall;CO43 3RN

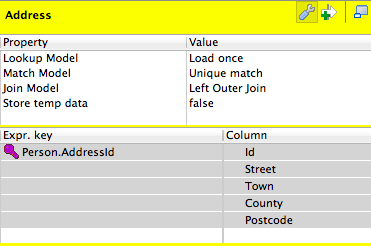
5;Mr;Dexter;Booth;;;;;

## Configuring Inputs

**Example Job #1** demonstrated a basic join, using the default input options. Against each input, you will notice some option Input Option Buttons buttons.

Selecting the first button Join Options Button (this is only available on *look-up* inputs), reveals the **tMap Settings (Join Options)**. Notice that, when you change one of the settings from *default*, the icon will indicate the number of changes that you have made, for example, Join Options Button Modified.

### Join Options



#### Lookup Model

Normally, you would choose to load your *look-up* input only once. This is the default **Lookup Model** of **Load Once**. If you have a use-case where you need to reload the *look-up* input for each of your *primary* rows, then you have the option of **Reload at each row** or **Reload at each row (cache)**. Depending on the size of your inputs, either of these latter two options are likely to severely impact the throughput of your Job.

#### Match Model

The default **Match Model** is the curiously named **Unique match**. If your *primary* row matches multiple rows in your *look-up*input, then only the **last** matching row will be output. The remaining options are **First match**, where only the first matching row will be output, and **All matches** where all matching rows will be output.

#### Join Model

The default **Join Model** is **Left Outer Join**, that is, if no matching row appears in the *look-up* input, rows from the *primary*input will still be output. If you need to perform an [Antijoin](http://en.wikipedia.org/wiki/Relational_algebra" \l "Antijoin_.28.E2.96.B7.29), then select this option and exclude rows later, by outputting a *key value* from your *look-up* input and subsequently excluding rows where this value is *null*. The second option available is **Inner Join**. In this case, only rows where a successful match has been made against the *look-up*, will be output.

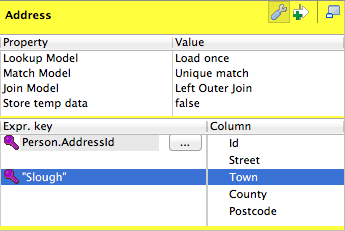
#### Store temp Data

If you are processing large datasets, you may find it helpful to set this option to *true*, to conserve memory. When set to *true*, you will also need to set **Temp data directory path**; which may be found on the **Advanced settings** component tab of [tMap](https://www.talendbyexample.com/talend-tmap-component-reference.html). Note that this is likely to have a negative impact on the overall throughput of your Job.

## Basic Look-up Filtering

You can perform some basic filtering against your *look-up* inputs from within [tMap](https://www.talendbyexample.com/talend-tmap-component-reference.html). Remember that all input data is read by [tMap](https://www.talendbyexample.com/talend-tmap-component-reference.html), so consider pre-filtering large datasets when reading sources such as databases (see [t[*DB*]Input](https://www.talendbyexample.com/talend-tdbinput-reference.html)).

As well as entering a *join-key* in to a *look-up* columns's **Expr. key** field, you may also enter an *expression*, as shown below.



Using this filter in our original example, all of our *Person* rows will be output; however, only *Addresses* where Address.Town == "Slough" would be included.

Id;Title;FirstName;LastName;AddressId;Street;Town;County;Postcode

1;Mr;Austin;Patel;4;;;;

2;Ms;Sophia;Watson;2;140 Great Square;Slough;Cornwall;CO43 3RN

3;Mr;Ewan;Parker;3;;;;

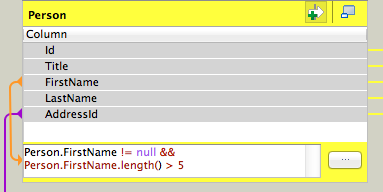
4;Ms;Evie;Cunningham;2;140 Great Square;Slough;Cornwall;CO43 3RN

5;Mr;Dexter;Booth;;;;;

## Input Expression Filters

Each input allows you to specify an **Expression Filter**. Selecting the Activate/unactivate expression filter button, reveals the **tMap Expression Editor**. Here, you can enter complex **Expression Filters** and enter filtering for both *primary* and *look-up* inputs (note that outputs also support **Expression Filters**). It is unfortunate that, when you close the **Expression Filter**, Talend provides no indication as to whether or not an expression has been entered. For this reason, I avoid using *hidden* expression filters.

In the following example, an **Expression Filter** has been entered for the *primary* input, performing some filtering against the column *Person.FirstName*.



As can be seen from the following output, we have now reduced our data set to People who's **FirstName** has more than 5 characters.

Id;Title;FirstName;LastName;AddressId;Street;Town;County;Postcode

1;Mr;Austin;Patel;4;;;;

2;Ms;Sophia;Watson;2;140 Great Square;Slough;Cornwall;CO43 3RN

5;Mr;Dexter;Booth;;;;;