

Работа в ETL-системе Talend

Reading movies information from a CSV file

If you want to replicate the example described in this document and use the exact input data, you can download `tos_di_gettingstarted_source_files.zip` from the **Downloads** tab in the left panel of this page, and then save the source files in your local directory `C:\getting_started\input_data\`.

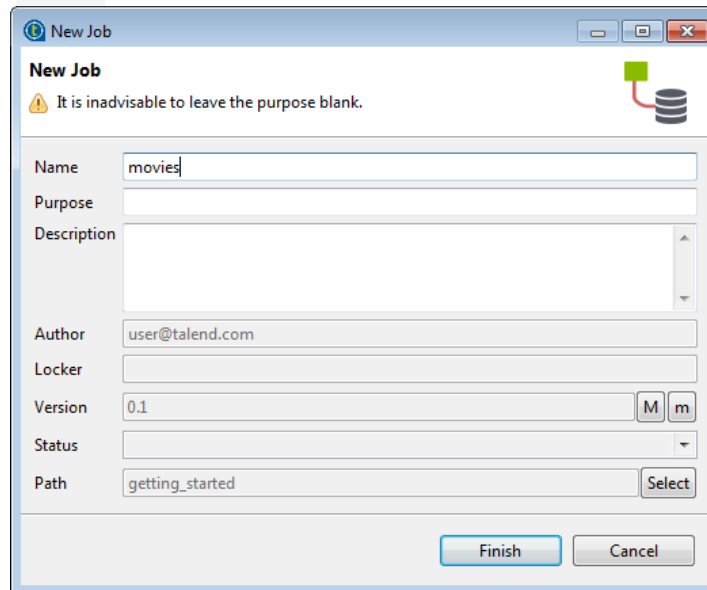
Creating your first Job

This procedure describes how to create a Job folder named `getting_started` and a Job named `movies` in the folder.

Procedure

1. In the **Repository** tree view, right click the **Job Designs** node, and select **Create folder** from the contextual menu.
2. In the **New Folder** wizard, name your Job folder `getting_started` and click **Finish** to create your folder.
3. Right-click the `getting_started` folder and select **Create Job** from the contextual menu.
4. In the **New Job** wizard, give a name to the Job you are going to create and provide other useful information if needed.

In this example, enter `movies` in the the **Name** field.



In this step of the wizard, **Name** is the only mandatory field. The information you provide in the **Description** field will appear as hover text when you move your mouse pointer over the Job in the **Repository** tree view.

5. Click **Finish** to create your Job.

An empty Job is opened in the Studio.

Adding and linking components

This example describes how to add and link components in the newly created Job, so that it will read a CSV file and display the data on the console.

Procedure

1. Drop a **tFileInputDelimited** and a **tLogRow** component from the **Palette** onto the design workspace.
You can find the **tFileInputDelimited** component in the **Input** group of the **File** family and the **tLogRow** component in the **Logs & Errors** family in the **Palette**.
2. Click the **tFileInputDelimited** component, and drag and drop the arrow icon displayed onto the **tLogRow** component.

The two components are now connected via a **Row > Main** connection.



Results

Now you have added the required components to the Job. In the next steps you will need to prepare the required metadata and configure the Job.

Preparing the movies metadata

This example describes how to set up the metadata of the source file `movies.csv` in the **Repository**.

Repository metadata can be used across Jobs, allowing you to configure your Jobs quickly without having to define each parameter and schema manually.

Before you begin

- You have the source file `movies.csv` ready in the directory `C:\getting_started\input_data\`.

Procedure

1. In the **Repository** tree view, expand the **Metadata** node, right-click **File delimited**, and select **Create file delimited** from the contextual menu to open the **New Delimited File** wizard.
2. In the **New Delimited File** wizard, enter a name for the file metadata, `movies` in this example, and other useful information to better describe your file metadata, and then click **Next** to go to the next step and define the general properties of the file.

In this step of the wizard, **Name** is the only mandatory field. The information you provide in the **Description** field will appear as a tooltip when you move your mouse pointer over the file connection.

3. In the **File** field specify the path of the source file, or click **Browse** to browse to the file.

The file is loaded, and the **File Viewer** area displays an abstract of the file, allowing you to check the file consistency, the presence of header and more generally the file structure.

4. From the **Format** list, select your operating system, and click **Next** to parse the file.
5. On the **Preview** tab, select the **Set heading row as column names** check box to retrieve the file column names from the first row, and then click **Refresh Preview**.

New Delimited File
File - Step 3 of 4
Add a Metadata File on repository
Define the setting of the parse job

File Settings
Encoding: US-ASCII
Field Separator: Semicolon Corresponding Character: ";"
Row Separator: Standard EOL Corresponding Character: "\n"

Escape Char Settings
CSV: ☒ Delimited
Escape Char: Empty
Text Enclosure: Empty
☐ Split row before field

Rows To Skip
If any rows must be ignored, specify the following parameters
Header: ☒ 1
Footer: ☐
☐ Skip empty row

Limit Of Rows
If the number of lines must be limited, specify this number
Limit:

Preview Output
☒ Set heading row as column names Refresh Preview

movieID	title	releaseYear	url	directorID
315	Apt Pupil	1998	http://us.imdb.com/Title?Apt+Pupil+(1998)	26
1294	Ayn Rand: A Sense of Life	1998	http://us.imdb.com/Title?Ayn+Rand%3A+A+Sense+of+Life+(1997)	123
1679	B. Monkey	1998	http://us.imdb.com/M/title-exact?B%2E+Monkey+(1998)	124
1649	Big One, The	1998	http://us.imdb.com/Title?Big+One,+The+(1997)	122

Export as context Revert Context

< Back Next > Finish Cancel

The file preview is refreshed, and the **Header** check box in the **Rows To Skip** area is automatically selected, with the number of header rows to be skipped incremented by 1.

- If the file contains more than one heading row, which need to be skipped in file parsing, specify the number in this field and click **Refresh Preview** again.
- Click **Next** to retrieve the file schema.

The **Description of the Schema** table displays the generated file schema.

- Name the schema `movies_schema` and check the file schema and edit it according to your actual needs.

In this example, increase the length of the **title** and **url** columns.

New Delimited File
File - Step 4 of 4
Add a Schema on repository
Define the Schema

Name: `movies_schema`
Comment:

Schema
Click to update schema preview

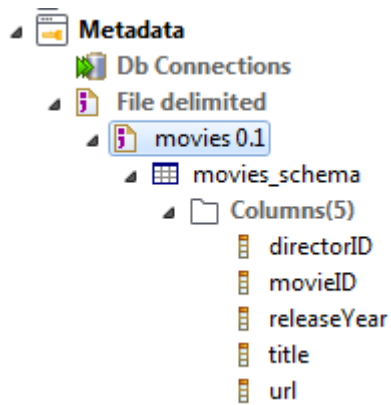
Description of the Schema

Column	Key	Type	<input checked="" type="checkbox"/>	N..	Date Pattern (Ctrl+Sp...	Length	Precision	Default	Comment
movieID	<input type="checkbox"/>	Integer	<input checked="" type="checkbox"/>			4	0		
title	<input type="checkbox"/>	String	<input checked="" type="checkbox"/>			72	0		
releaseYear	<input type="checkbox"/>	Integer	<input checked="" type="checkbox"/>			4	0		
url	<input type="checkbox"/>	String	<input checked="" type="checkbox"/>			120	0		
directorID	<input type="checkbox"/>	String	<input checked="" type="checkbox"/>			3	0		

< Back Next > Finish Cancel

- Click **Finish** to validate the schema close the wizard.

The created file metadata is shown in the **Repository** tree view.



Results

You now have the movies file metadata ready for use. Next, you need to apply the created metadata to the component that reads the source file.

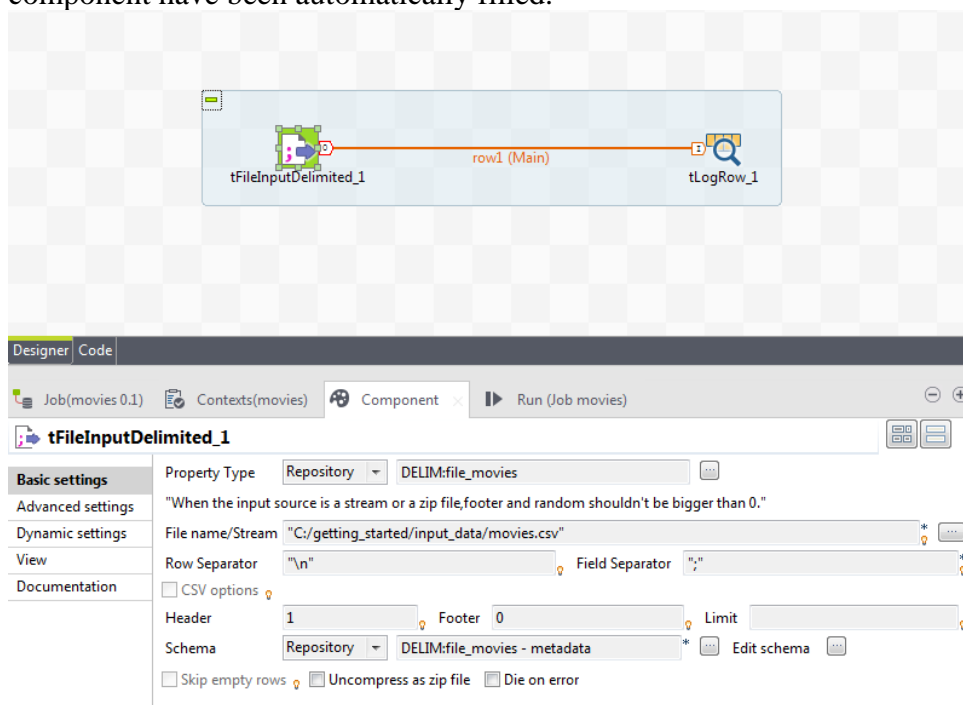
Configuring and executing your Job

This example describes how to configure the components using the metadata created in the previous procedure and run your Job.

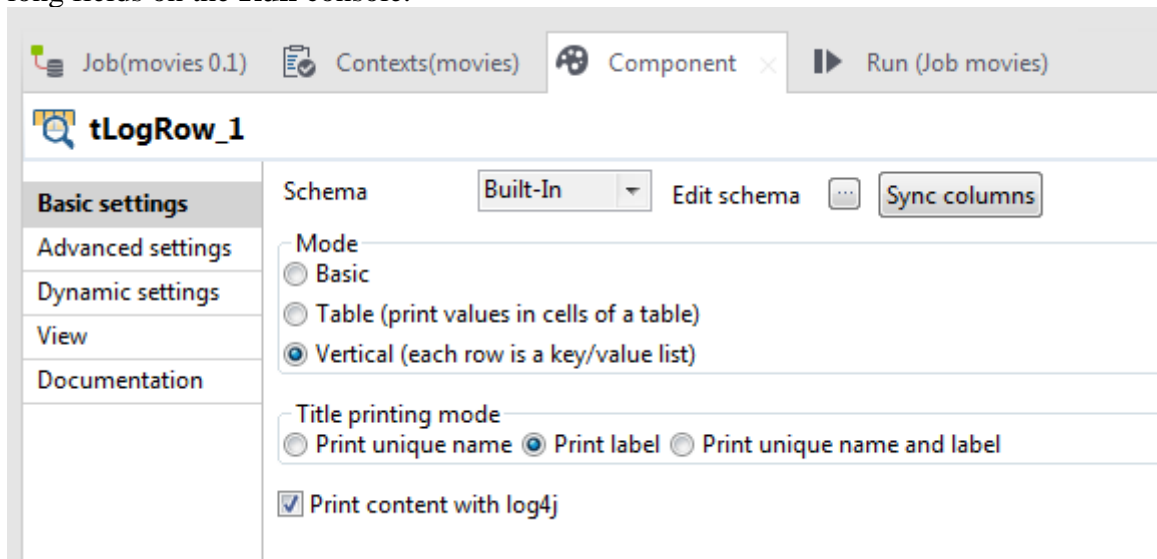
Procedure

1. In the **Repository** tree view, double-click the Job **movies** to open it in the design workspace. You can skip this step if the Job is already open and active in the design workspace.
2. In the **Repository** tree view, expand **Metadata > File delimited**, and drag and drop the file connection **movies** or its schema **movies_schema** onto the **tFileInputDelimited** component in the design workspace. When asked whether to propagate the changes to the output component, click **Yes**.

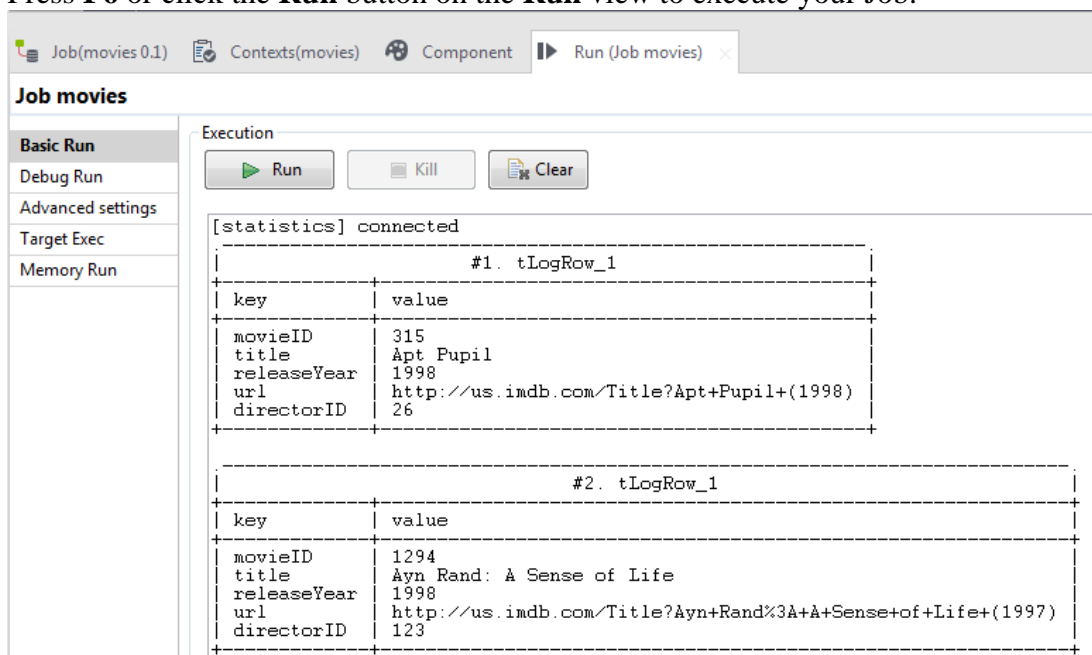
In the **Basic settings** tab of the **Component** view, you'll find that all the parameters of the component have been automatically filled.



3. Double-click the **tLogRow** component to open its **Basic settings** tab view.
4. In the **Mode** area, select the **Vertical (each row is a key/value list)** option for better readability of long fields on the **Run** console.



5. Press **F6** or click the **Run** button on the **Run** view to execute your Job.



Results

The **Run** console displays the movies information read from the source file.

Filtering the movies information

This scenario will extend the Job described in [Reading movies information from a CSV file](#) to filter the data flow to get only those movies with valid director information.

This scenario demonstrates:

- How to duplicate a Job. See [Duplicating the existing Job](#) for details.
- How to add a component by typing its name on a connection or on the design workspace. See [Adding a mapping component](#) for details.

- How to drop a metadata item or its schema as a component on the design workspace. See [Adding a lookup component](#) for details.
- How to perform basic processing to data flows using **tMap**. See [Configuring mappings and executing the Job](#) for details.

Preparing directors file metadata

This procedure shows how to set up the metadata of the reference file *directors.txt* in the **Repository**. This metadata item will be used to add and set up the lookup input in this scenario.

Before you begin

- You have the file `directors.txt` ready in the directory `C:\getting_started\input_data\`.

Procedure

1. In the **Repository** tree view, expand the **Metadata** node, right-click **File delimited**, and select **Create file delimited** from the contextual menu to open the **New Delimited File** wizard.
2. Enter a name for the file connection, `directors` in this example, and other useful information to better describe your file metadata, and then click **Next** to go to the next step and define the general properties of the file.

New Delimited File

File - Step 1 of 4

Add a Metadata File on repository
Define the properties

Name:

Purpose:

Description:

Author:

Locker:

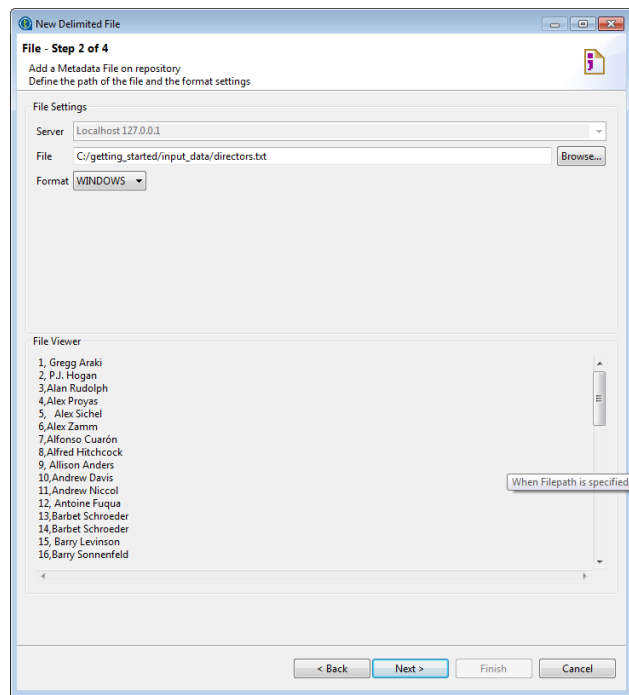
Version:

Status:

Path:

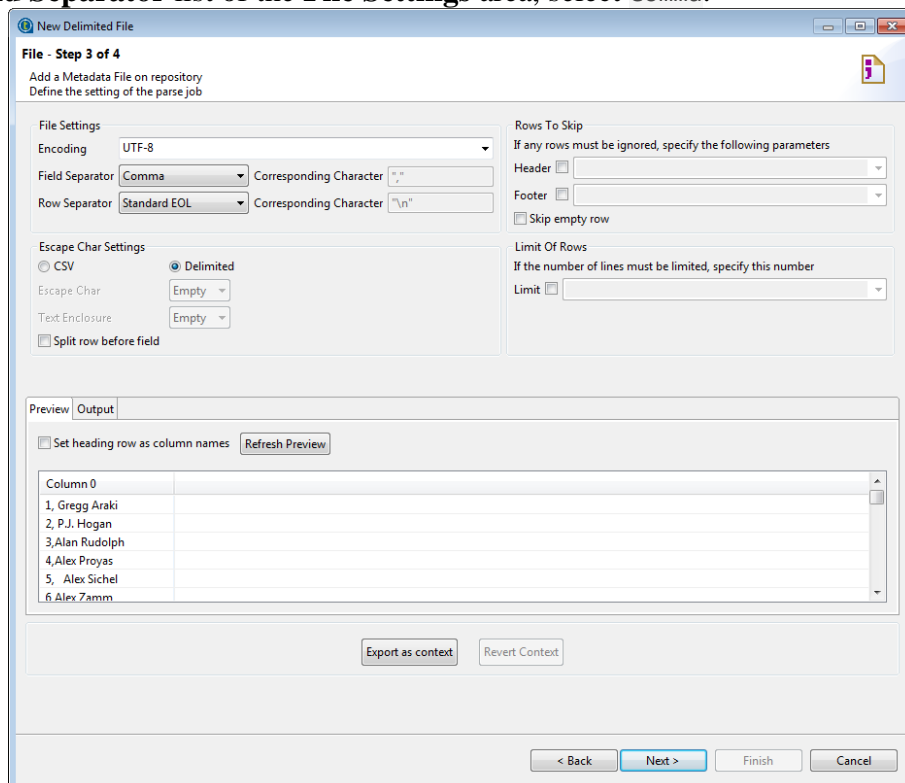
< Back Next > Finish Cancel

3. In the **File** field specify the path of the source file, or click **Browse** to browse to the file.



The file is loaded, and the **File Viewer** area displays an abstract of the file, allowing you to check the file consistency, the presence of header and more generally the file structure.

4. Select **Windows** from the **Format** list, and click **Next** to parse the file.
5. From the **Field Separator** list of the **File Settings** area, select Comma.



6. Click **Next** to retrieve the file schema.

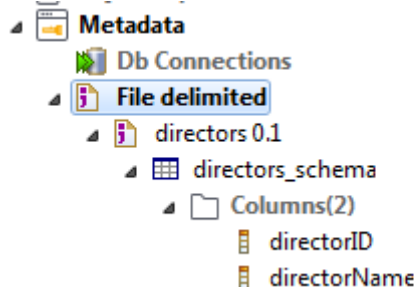
The **Description of the Schema** table displays the generated file schema.

7. Name the schema `directors_schema` and rename the columns to `directorID` and `directorName` respectively, and change the data type of the `directorID` columns from Integer to String.

Column	Key	Type	N.	Date Pattern (Ctrl+Sp...	Length	Precision	Default	Comment
directorID	<input type="checkbox"/>	String	<input checked="" type="checkbox"/>		2	0		
directorName	<input type="checkbox"/>	String	<input checked="" type="checkbox"/>		20	0		

8. Click **Finish** to validate the schema close the wizard.

The created file metadata is shown in the **Repository** tree view.



Results

You now have the directors file metadata ready for use when you set up the component to read the reference file.

Duplicating the existing Job

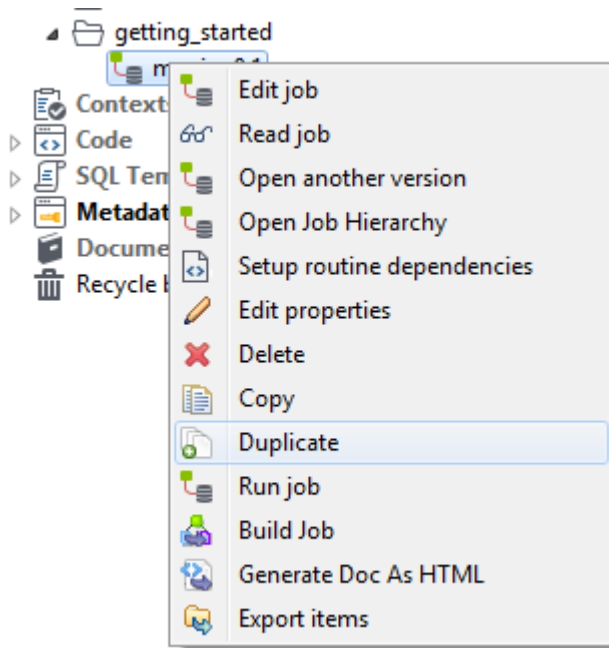
This procedure shows how to create a Job based on an existing Job.

Before you begin

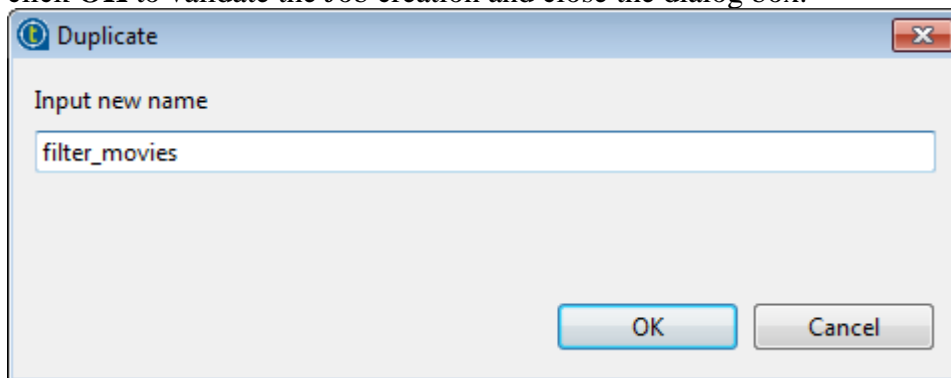
- You have created and successfully executed the Job named *movies* as described in [Reading movies information from a CSV file](#).

Procedure

1. In the **Repository** tree view, right-click the Job named *movies* and select **Duplicate** from the contextual menu.



2. In the **Duplicate** dialog box, enter a name for the new Job, `filter_movies` in this example, and click **OK** to validate the Job creation and close the dialog box.



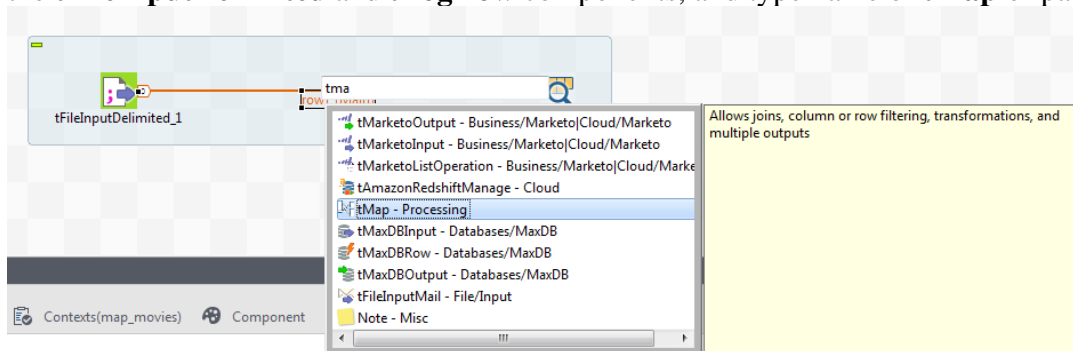
The Job named `filter_movies` is created, which is a duplicate of the Job named `movies`.

Adding a mapping component

The procedure below shows how to add a mapping component by typing the component name directly on the existing connection.

Procedure

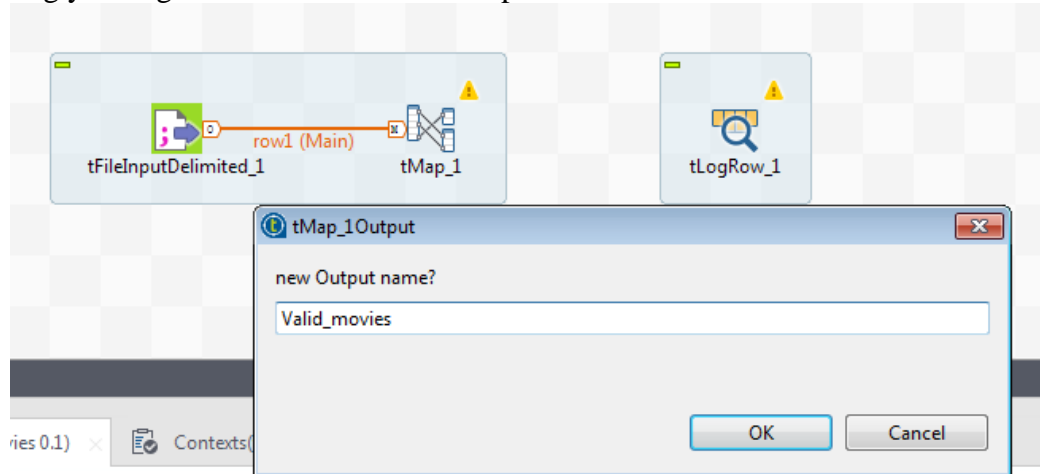
1. In the new Job named `filter_movies`, select the **Row** connection linking the **tFileInputDelimited** and **tLogRow** components, and type name of **tMap** or part of it.



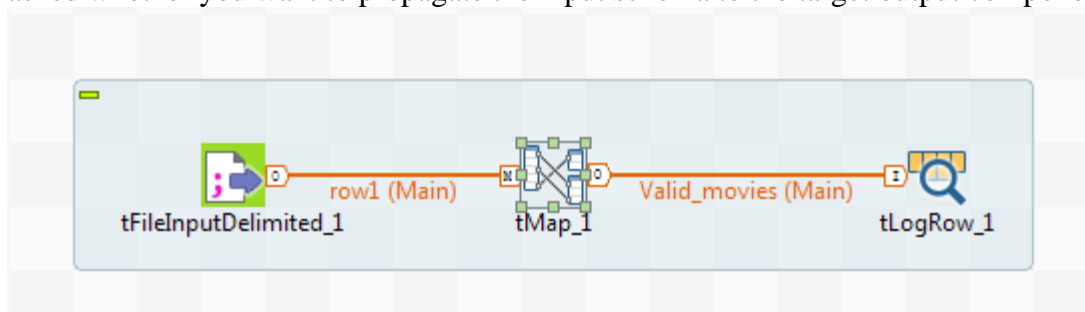
When you start typing the component name, a list of components that match your input appears. You can select a component to view its description besides the component list.

2. Double-click **tMap** on the list to add it onto the connection.

The newly added **tMap** component is now connected with the input component, and a dialog box opens asking you to give a name to the new output connection.



3. Enter a name for the new output connection, `Valid_movies` in this example, and click **OK**. When asked whether you want to propagate the input schema to the target output component, click **Yes**.



Results

The **tMap** component is now added to the Job and connected with the two existing components via **Row > Main** connections.

Adding a lookup component

The procedure below shows how to add a lookup input component from the **Repository**, connect it to the **tMap**, and enable column trimming in the component.

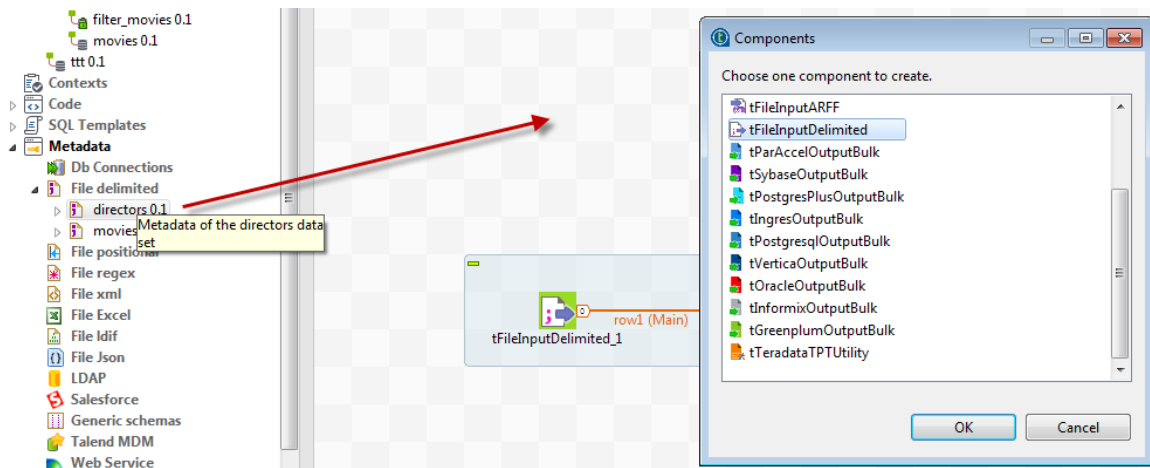
Before you begin

- You have centralized the metadata for `directors.txt` in the **Repository** as described in [Preparing directors file metadata](#).

Procedure

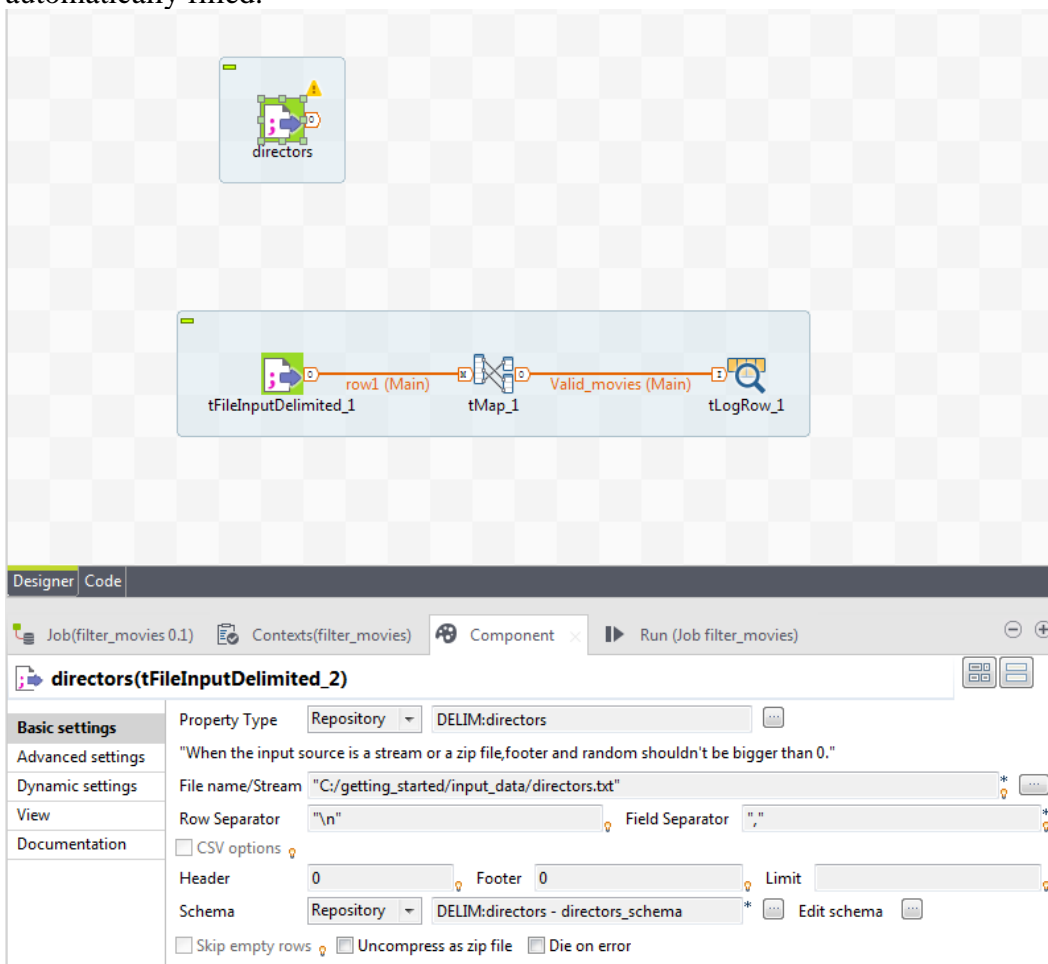
1. In the **Repository** tree view, expand **Metadata > File delimited**, drag and drop the file connection **directors** or its schema **directors_schema** onto the design workspace.

The **Components** dialog box opens, showing a list of components you can add to the Job from this metadata item.



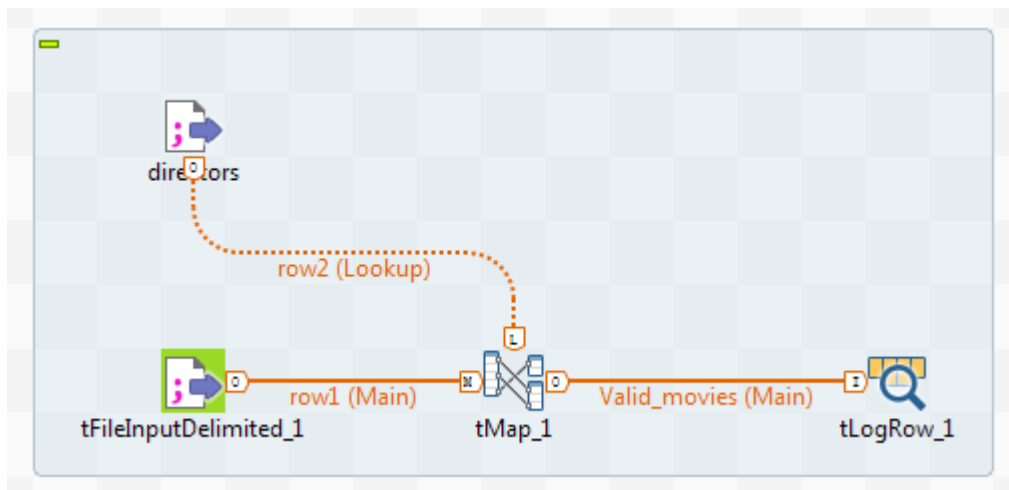
2. Select **tFileInputDelimited** and click **OK**.

A **tFileInputDelimited** labelled **directors** is added to the design workspace, with its basic settings automatically filled.



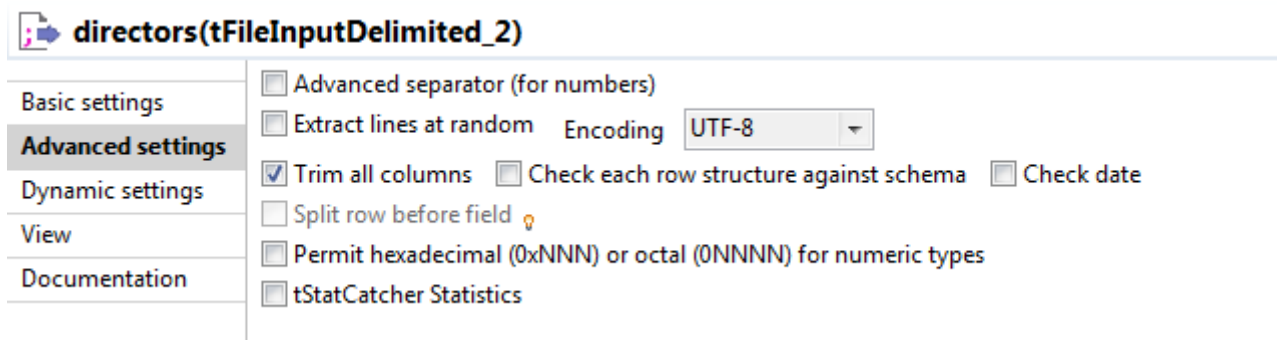
3. Right-click the newly added **tFileInputDelimited** component, select **Row > Main** from the contextual menu, and click the **tMap** component.

The **tFileInputDelimited** is connected to the **tMap** via a lookup connection now.



4. In the **Advanced settings** tab of the new **tFileInputDelimited** component, and select the **Trim all columns** check box.

Some records of the reference input file `directors.txt` contains leading white spaces. This option allows you to remove such white spaces from the lookup input flow when the Job is executed.



Results

You have now all the components in the Job needed for filtering the movies information. Next you'll need to configure mappings in the **tMap** component to filter the main input flow against the lookup flow and output the desired information.

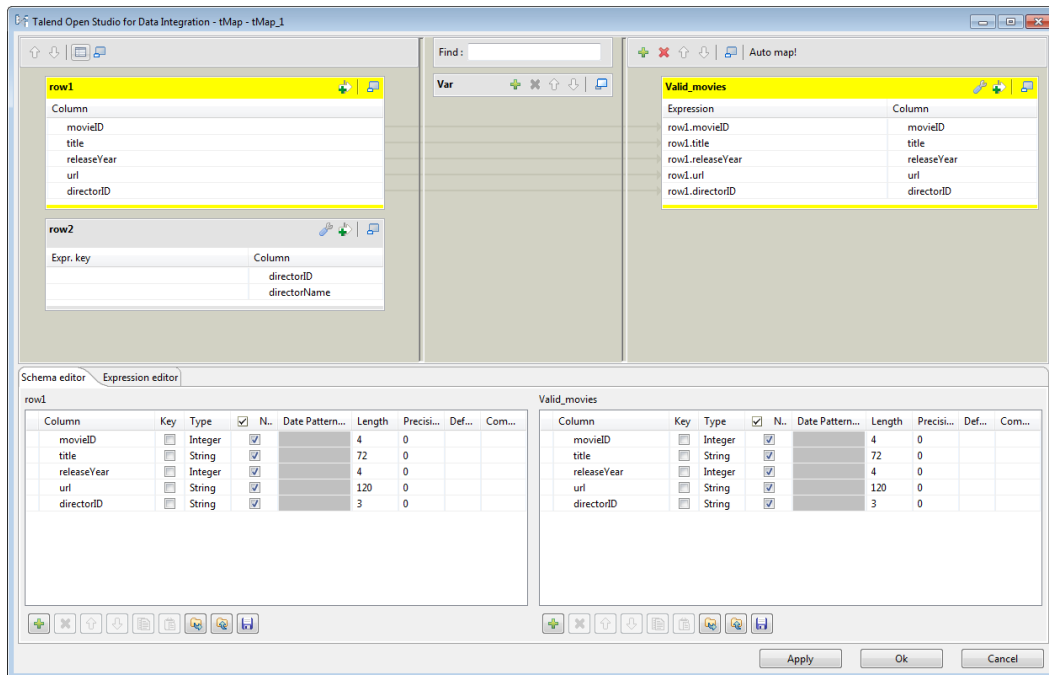
Configuring mappings and executing the Job

The procedure below shows how to configure mappings and an inner join to output movies information with valid director IDs.

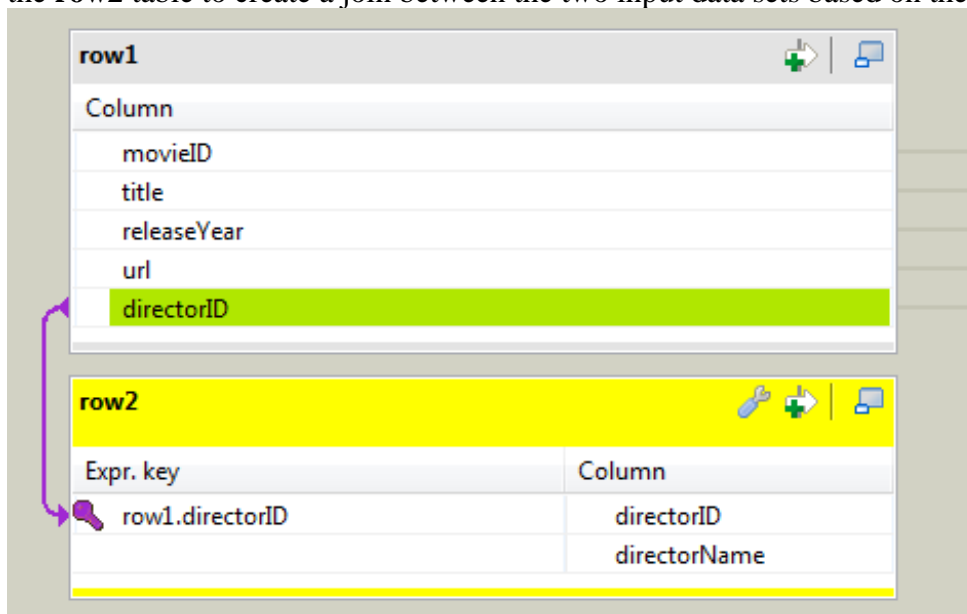
Procedure

1. Double-click the **tMap** component to open the map editor.

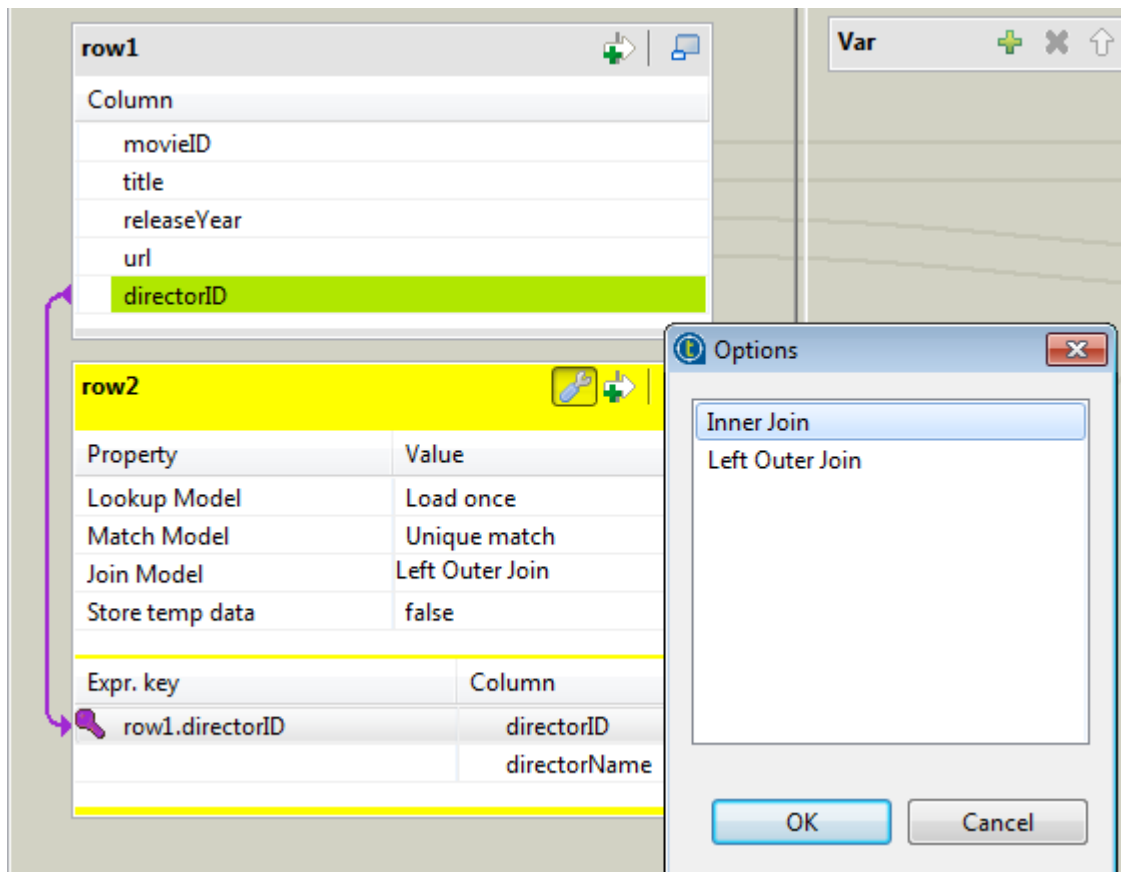
The map editor shows three tables, named **row1**, **row2** and **Valid_movies** in this example, corresponding respectively to the movies file schema, the directors file schema, and the schema of the output for valid movies information, and columns in the **row1** table are already mapped to the columns in the **Valid_movies** table.



2. Select the **directorID** column in the **row1** table, and drop it onto the **directorID** column in the **row2** table to create a join between the two input data sets based on the director IDs.



3. Click the **tMap settings** button, then click **Value** field for **Join Model**, and then click the [...] button that appears to open the **Options** dialog box. In the dialog box, select **Inner Join** and click **OK** to define the join as an inner join.



With this setting, only the movie records with the director IDs matching with those in the reference file will be passed to the output.

4. In the **Schema editor** at the bottom of the map editor, select **directorID** column of the output schema, **Valid_movies** in this example, and click the **X** button to remove it.
5. Click the **[+]** button beneath the output table to add a new column, name it **directedBy**, set its length to **20**, and move it up so that it's between the **title** and **releaseYear** columns.

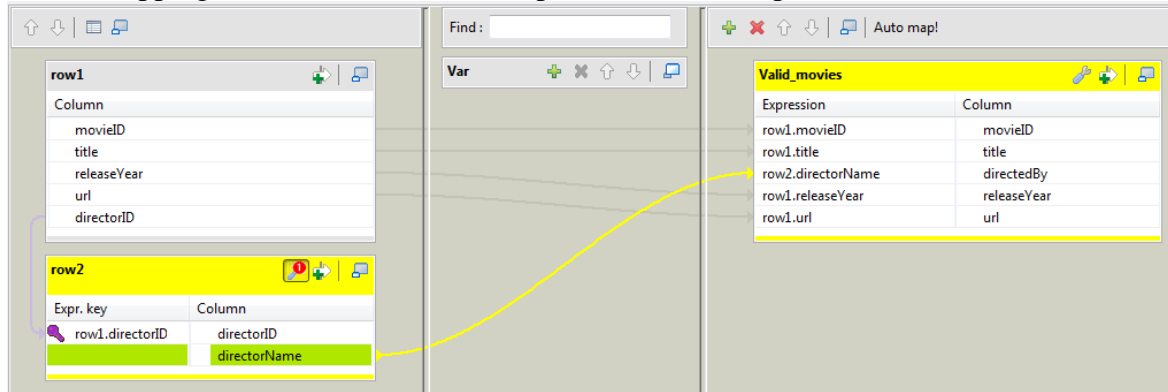
Valid_movies

Column	Key	Type	<input checked="" type="checkbox"/>	N..	Date Pattern...	Length	Precisi...	Def...	Com...
movieID	<input type="checkbox"/>	Integer	<input checked="" type="checkbox"/>			4	0		
title	<input type="checkbox"/>	String	<input checked="" type="checkbox"/>			72	0		
directedBy	<input type="checkbox"/>	String	<input checked="" type="checkbox"/>			20			
releaseYear	<input type="checkbox"/>	Integer	<input checked="" type="checkbox"/>			4	0		
url	<input type="checkbox"/>	String	<input checked="" type="checkbox"/>			120	0		

Below the table are several icons: a green plus sign, a red X, a blue up arrow, a blue down arrow, a document icon, a folder icon, a magnifying glass icon, and a save icon.

6. Select the **directorName** column in the **row2** table, and drop it to the **Expression** field corresponding to the **directedBy** column in the output table.

A new mapping is created between lookup table and the output table.



- Click **OK** to validate the mappings and close the map editor, and click **Yes** when asked whether to propagate the changes.

The mapping configurations are saved and the output schema is synchronized to the output component **tLogRow**.

- Press **F6** or click the **Run** button on the **Run** view to execute your Job.

The screenshot shows the execution console for the job 'filter_movies'. The console displays the following output:

```

movieID      192
title        Raging Bull
directedBy   Martin Scorsese
releaseYear   1980
url          http://us.imdb.com/M/title-exact?Raging%20Bull%20(1980)
-----
#255. tLogRow_1
-----
key          value
movieID      200
title        Shining, The
directedBy   Stanley Kubrick
releaseYear   1980
url          http://us.imdb.com/M/title-exact?Shining,%20The%20(1980)
-----
[statistics] disconnected
Job filter_movies ended at 10:51 21/04/2016. [exit code=0]
  
```

At the bottom, there are checkboxes for 'Line limit' (set to 100) and 'Wrap' (checked).

Results

Only movie records with valid director information are displayed on the **Run** console.

Gathering rejected movies information and saving processing results to a database

Based on the scenario described in [Filtering the movies information](#), this scenario further extends the Job to gather movies data missing director information and writes both valid and invalid data to a MySQL database.

This scenario demonstrates:

- How to add a component by typing on the design workspace or dragging from an existing component. See [Adding database output components to your Job](#) for details.

- How to configure mappings for rejected information in **tMap**. See [Configuring mappings for rejected data](#) for details.
- How to configure database outputs. See [Configuring MySQL database outputs](#) for details.

Adding database output components to your Job

In the example below we will create a new Job from the Job **filter_movies** and add two **tMySQLOutput** components. These components will be used to write the processed movies information to the specified database tables.

Before you begin

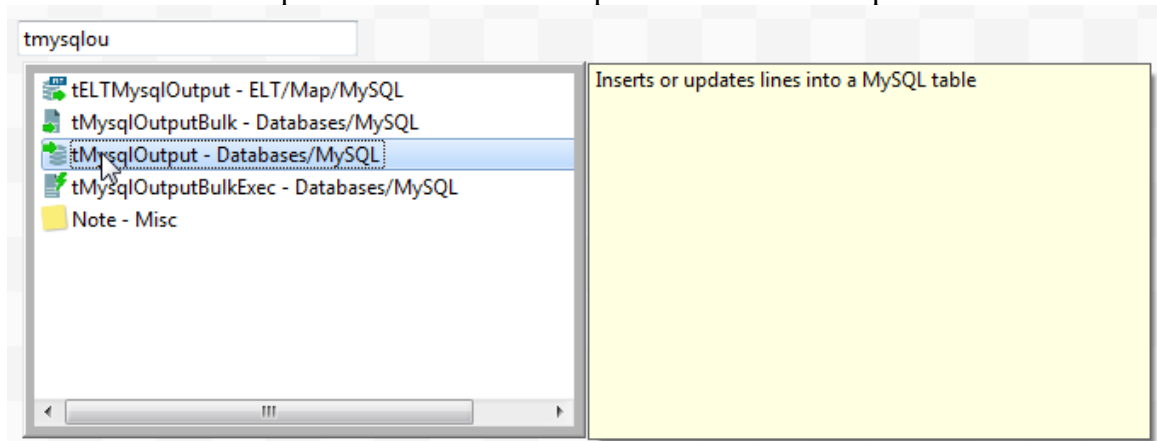
- You have created and successfully executed the Job **filter_movies** as described in [Filtering the movies information](#).

Procedure

1. Create a new Job by duplicating the Job created in the previous scenario, and name the new Job **write_movies_to_db**, and then double-click the Job to open it in the design workspace.
2. Right-click the **tLogRow** component and select **Delete** from the contextual menu to delete it.
3. Click where the **tLogRow** was on the design workspace and type the name of **tMySQLOutput** or part of it, and then select and double-click **tMySQLOutput** on the list to add it onto the design workspace.

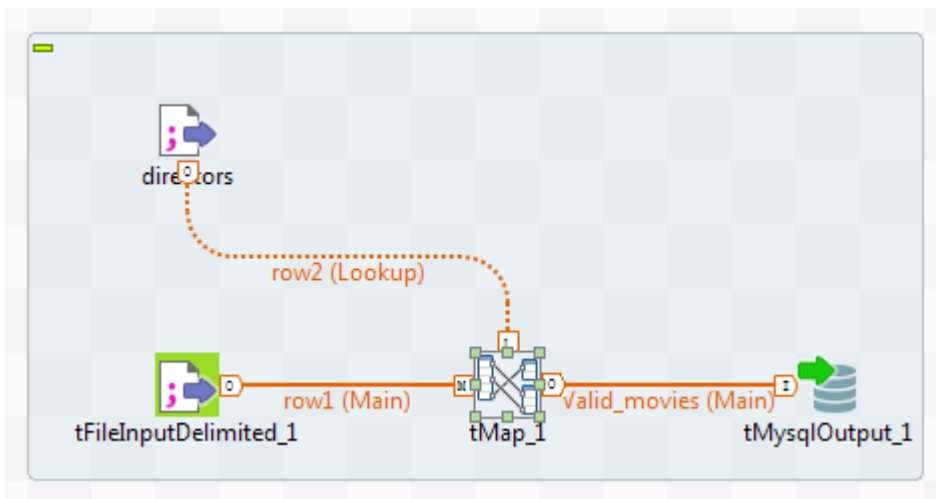
When you start typing the component name, a list of components that match your input appears.

You can select a component to view its description besides the component list.



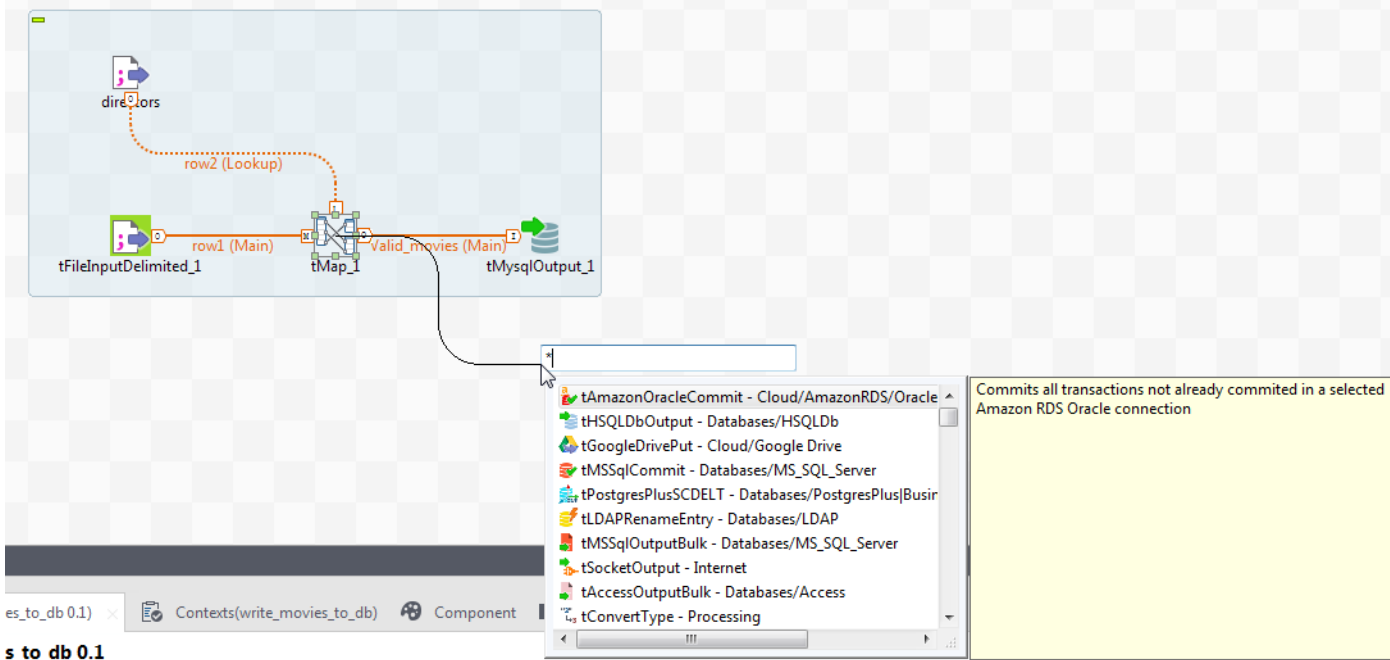
4. Right-click the **tMap** component, select **Row** > **Valid_movies** from the context menu, and click the **tMySQLOutput** to link it with the **tMap**.

The connection name **Valid_movies** corresponds to the name of the existing output table in **tMap**.



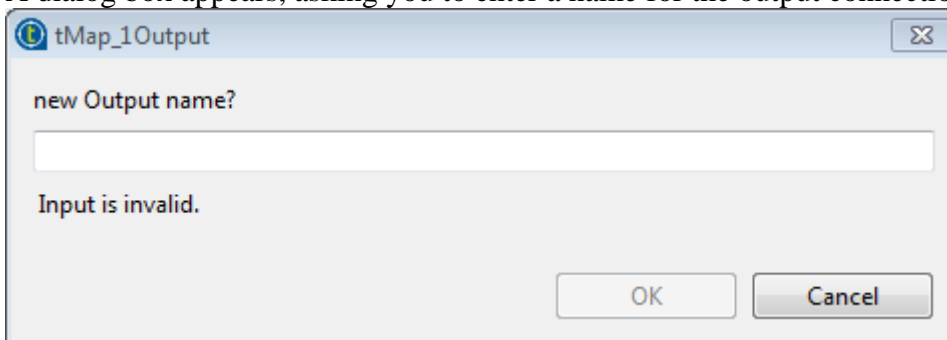
5. Click the **tMap** component, and drag and drop the arrow icon displayed onto the design workspace.

A text field and a list of suggested components appear. You can select a component to view its description besides the component list.

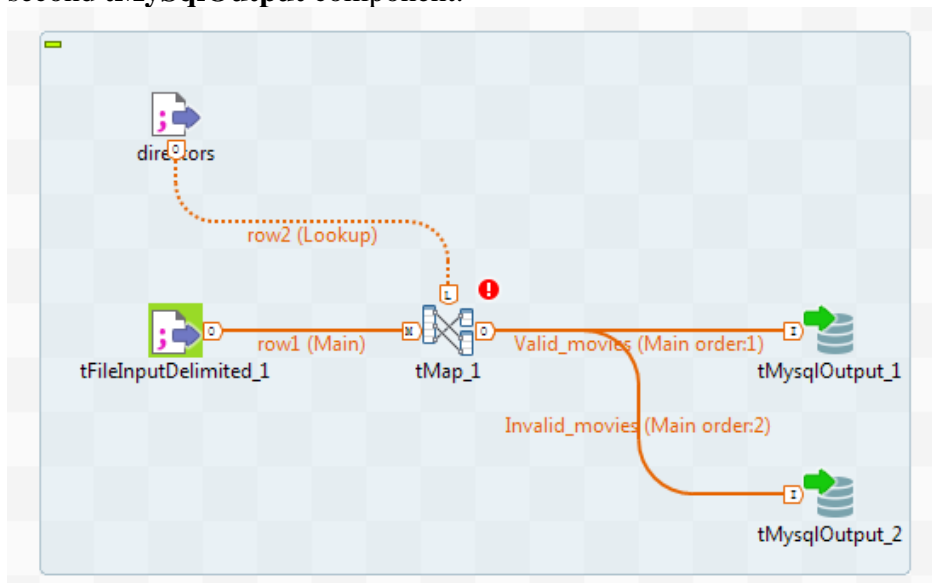


6. In the text field, type the name of **tMysqlOutput**, select the component on the list, and press **Enter** to add another **tMysqlOutput** component onto the design workspace.

A dialog box appears, asking you to enter a name for the output connection.



7. In the dialog box, enter `Invalid_movies` and click **OK** to connect **tMap** to the second **tMySQLOutput** component.



Results

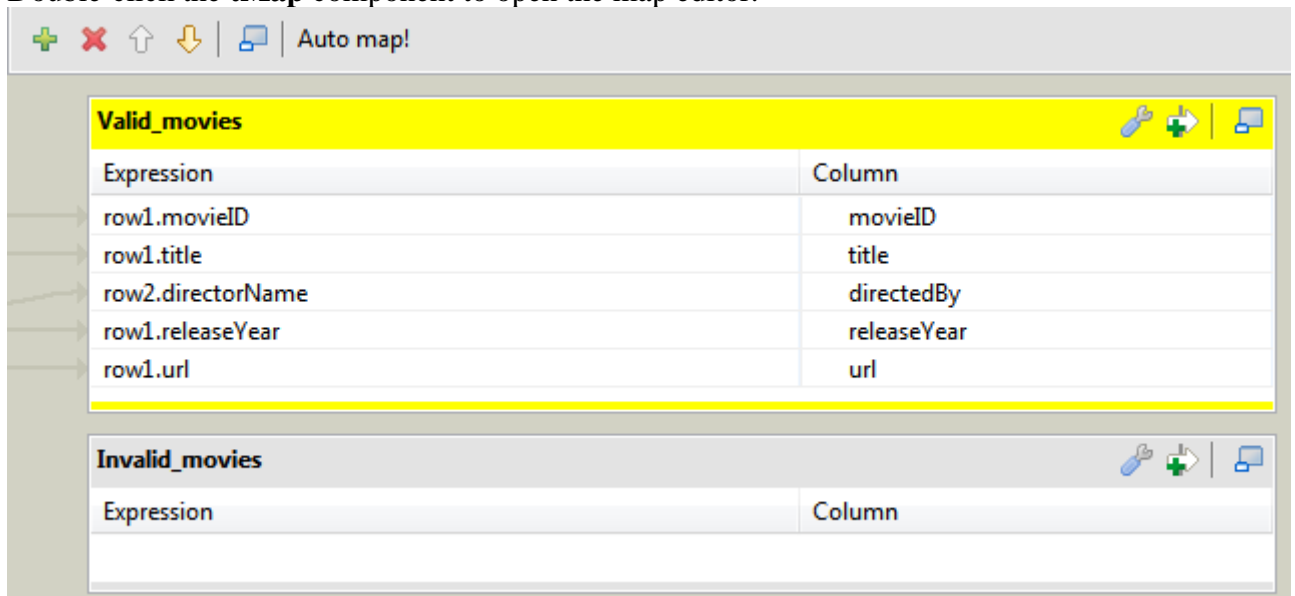
Now you have added and connected the database output components you need to write the processed movies information to a MySQL database. Next, you'll need to configure new mappings in the **tMap** and database settings in the **tMySQLOutput** components.

Configuring mappings for rejected data

This procedure shows how to configure mappings to gather rejected information.

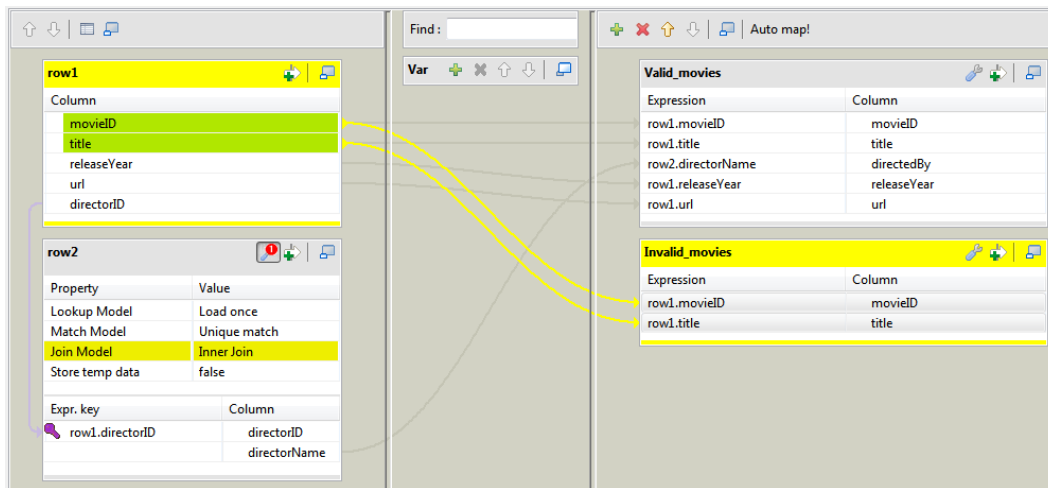
Procedure

1. Double-click the **tMap** component to open the map editor.

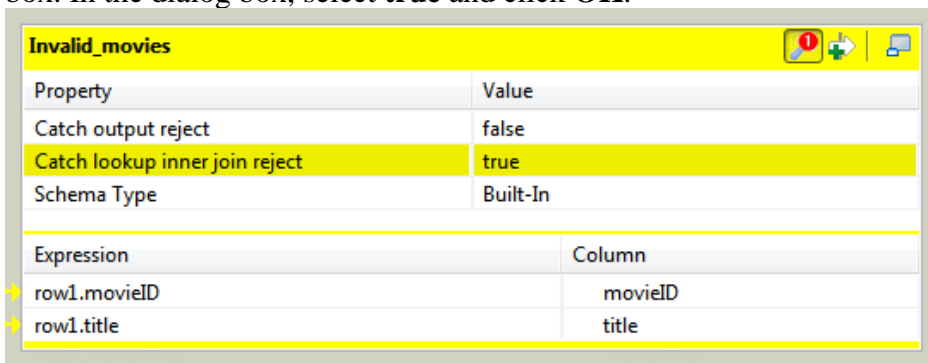


An second output table named **Invalid_movies** has been automatically created.

2. Drop the **movieID** and **title** columns from the **row1** table to the **Invalid_movies** table.



- Click the **tMap settings** button on the **Invalid_movies** table, click the **Value** field for **Catch lookup inner join reject**, and then click the [...] button that appears to open the **Options** dialog box. In the dialog box, select **true** and click **OK**.



With this setting, any records without director IDs or with director IDs that do not match with those in the reference file will be passed to this output.

- Click **OK** to validate the mappings and close the map editor, and click **Yes** when asked whether to propagate the changes.

The mapping configurations are saved and the output schema is synchronized to the output component.

Results

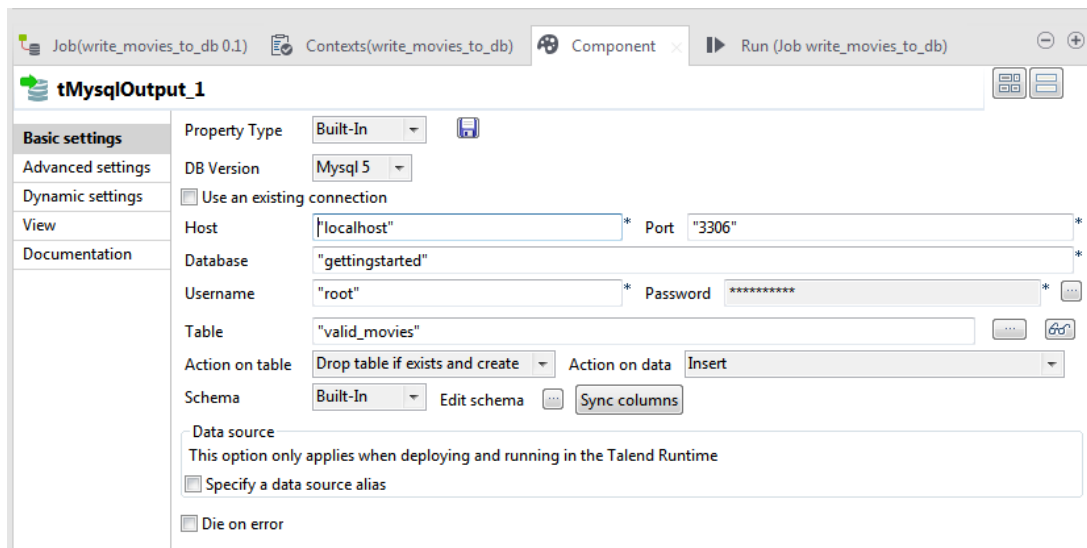
Now you have configured mappings for the rejected output. Next, you'll need to configure the output components to write the output flows to database tables.

Configuring MySQL database outputs

This procedure shows how to configure database output components to write movies information to MySQL database tables.

Procedure

- Double-click the first **tMySQLOutput** component to open its **Basic settings** in the **Component** view.



2. Provide the connection details needed to access your database, including the host name or IP address, port number, database name, user name and password, in the relevant fields.
When entering your password, you need first to click the [...] button next to the **Password** field to open a dialog box, enter your password between double quotation marks in the text field, and then click **OK**.
3. In the **Table** field, enter the name of the target database table.
In this example, the table for valid movies information is `valid_movies`.
4. Select the **Action on table** and **Action on data** options according to your needs.
In this example, we want to remove the table first if it already exists and then create an empty one, and use the default option for the action on data.
5. In the **Basic settings** of the second **tMySQLOutput** component, use the same settings as in the first **tMySQLOutput** except the name for the target database table.
In this example, the table for invalid movies information is `invalid_movies`.
6. Press **F6** or click the **Run** button on the **Run** view to execute your Job.

Results

The movies records with valid director information are saved to the database table named `valid_movies`, and those without valid director information are saved to the database table named `invalid_movies`.