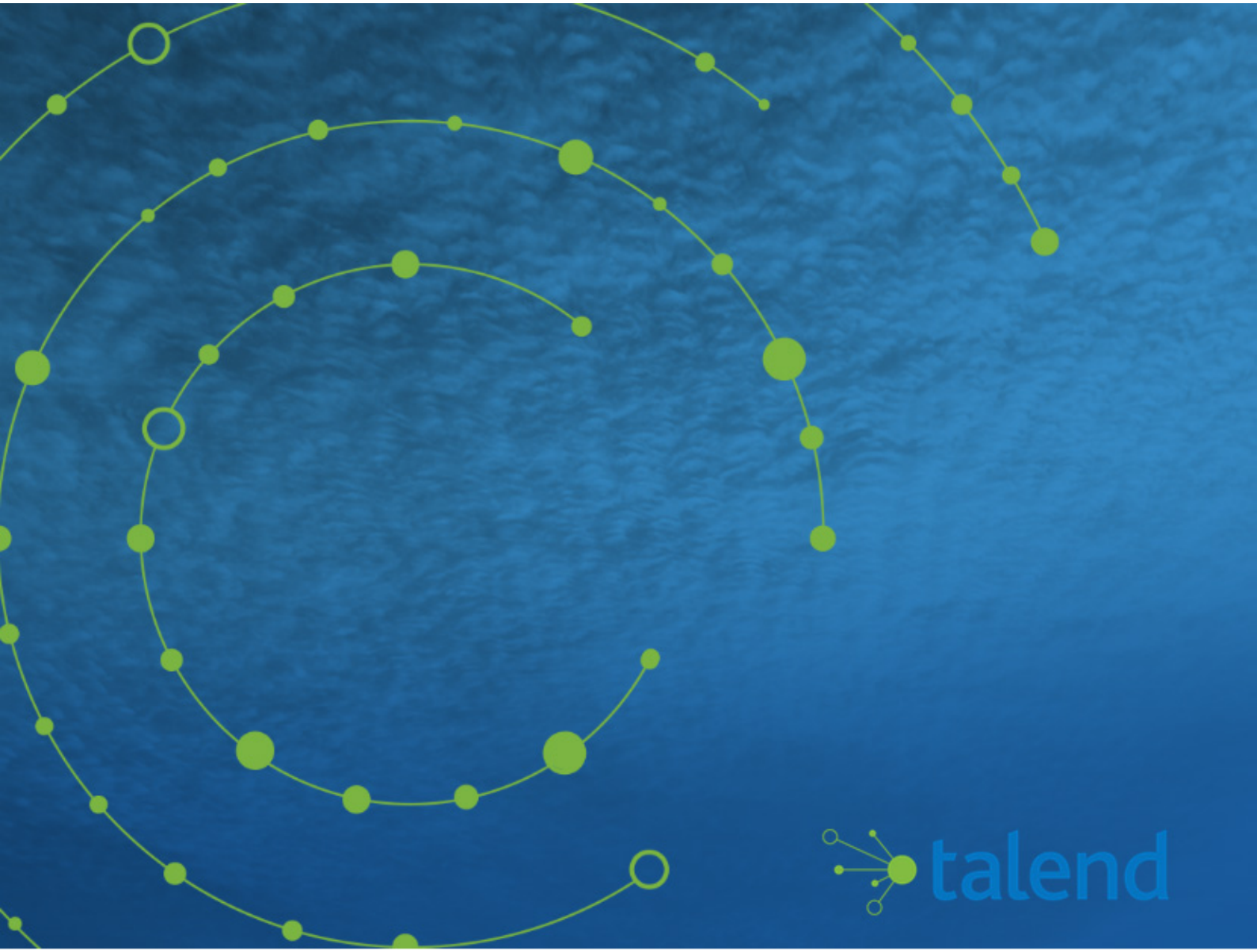


Exemples de Jobs d'intégration de données

7.0.1

Table des matières

Copyleft

Convient à la version 7.0.1. Annule et remplace toute version antérieure de ce guide.

Date de publication : 13 avril 2018

Exemple de Job comprenant un tMap

Pour illustrer le fonctionnement du Studio Talend, vous trouverez ci-dessous un scénario reflétant un cas d'utilisation réel. Dans ce scénario, vous devez charger un fichier dans une table MySQL en appliquant des transformations à la volée. Et dans une étape suivante, vous sélectionnez les données à charger en appliquant un filtre dynamique.

Avant de commencer le Job, vérifiez les données en entrée (Input) et les données attendues en sortie (Output).

Données en entrée

Le contenu du fichier en entrée est une liste des clients de toutes les régions de l'état de Californie. Ces données seront donc chargées dans une table de données.

La structure du fichier, communément appelée **Schéma** dans le Studio Talend comprend les colonnes suivantes :

- First name (prénom)
- Last name (nom)
- Address (adresse)
- City (ville)

Données en sortie

Vous souhaitez charger uniquement les données des clients habitant dans certaines régions (Counties) de la Californie dans la nouvelle base de données : les régions d'Orange et de Los Angeles.

La structure de la table est légèrement différente, ainsi les données devant être chargées dans la table de données doivent être structurées de la manière suivante :

- `Key` (Clé, Type entier)
- `Name` (Type chaîne, longueur max. 40)
- `Address` (Type chaîne, longueur max. 40)
- `County` (Type chaîne, longueur max. 40)

Pour charger cette table, vous devez utiliser les processus de mapping suivants :

La colonne `Key` est alimentée par un entier auto-incrémenté.

La colonne `Name` est renseignée avec une concaténation des données First Name (prénom) et Last Name (nom).

Les données de la colonne `Address` sont les mêmes que celles de la colonne `County` du fichier d'entrée et elles seront mises en majuscule avant d'être chargées.

La colonne `County` est alimentée par le nom de la région dans laquelle se situe la ville. Un fichier de référence vous aidera à filtrer les villes des régions d'Orange et de Los Angeles.

Données de référence

Etant donné que les données des régions d'Orange et de Los Angeles doivent être chargées dans la base de données, vous devez mapper les villes de Californie avec leur région respective, afin de pouvoir filtrer uniquement les villes d'Orange et de Los Angeles.

Pour cela, utilisez un fichier de référence contenant la liste des villes situées dans ces régions, par exemple :

City	County
Agoura Hills	Los Angeles
Alhambra	Los Angeles
Aliso Viejo	Orange
Anaheim	Orange
Arcadia	Los Angeles

Le fichier de référence de ce Job se nomme `LosAngelesandOrangeCounties.txt`.

Du scénario au Job

Pour mettre ce scénario en pratique, séparez ce Job en quatre étapes.

1. Création du Job, configuration des paramètres et lecture du fichier d'entrée
2. Mapping et transformations de données
3. Définition des paramètres du fichier de référence, mapping correspondant à l'aide du composant **tMap** et sélection du mode Inner Join.
4. Redirection des données en sortie dans une table MySQL

Etape 1 : Création du Job, définition des données d'entrée, lecture du fichier

Procédure

1. Après avoir lancé le Studio Talend, créez un projet en local ou importez un projet démo si vous lancez le Studio Talend pour la première fois.
2. Pour créer le Job, cliquez d'abord sur l'élément **Job Designs** du Référentiel avec le bouton droit de la souris et sélectionnez la première option du menu : **Create Job**.
3. Dans la boîte de dialogue qui apparaît alors à l'écran, seul le premier champ **Name** est obligatoire. Saisissez **California1** et cliquez sur **Finish**.

   Un Job vide s'ouvre ensuite dans la fenêtre principale et la **Palette** de composants techniques apparaît (par défaut, à droite du Studio) affichant une dizaine de familles de composants, notamment : Databases, Files, Internet, Data Quality, etc. Plus de 400 composants sont disponibles actuellement.

4. Pour lire le fichier `California_Clients`, utilisez le composant **tFileInputDelimited**. Ce composant se trouve dans la famille **File** > **Input** de la **Palette**. Cliquez sur ce composant et placez-le à la gauche de l'espace de modélisation.
5. Définissez maintenant les propriétés de lecture de ce composant : chemin d'accès, séparateur de colonnes, encodage, etc. Pour ce faire, utilisez le **Metadata Manager**. Cet outil possède de nombreux assistants qui vous aideront à définir les paramètres nécessaires et vous permettront de conserver ces propriétés qui pourront être réutilisées en un seul clic dans de futurs Jobs.
6. Puisque votre fichier d'entrée est un fichier plat délimité, cliquez sur **Metadata** > **File Delimited** dans le Référentiel et dans le menu contextuel du nœud **File Delimited**, sélectionnez l'option **Create file delimited**.

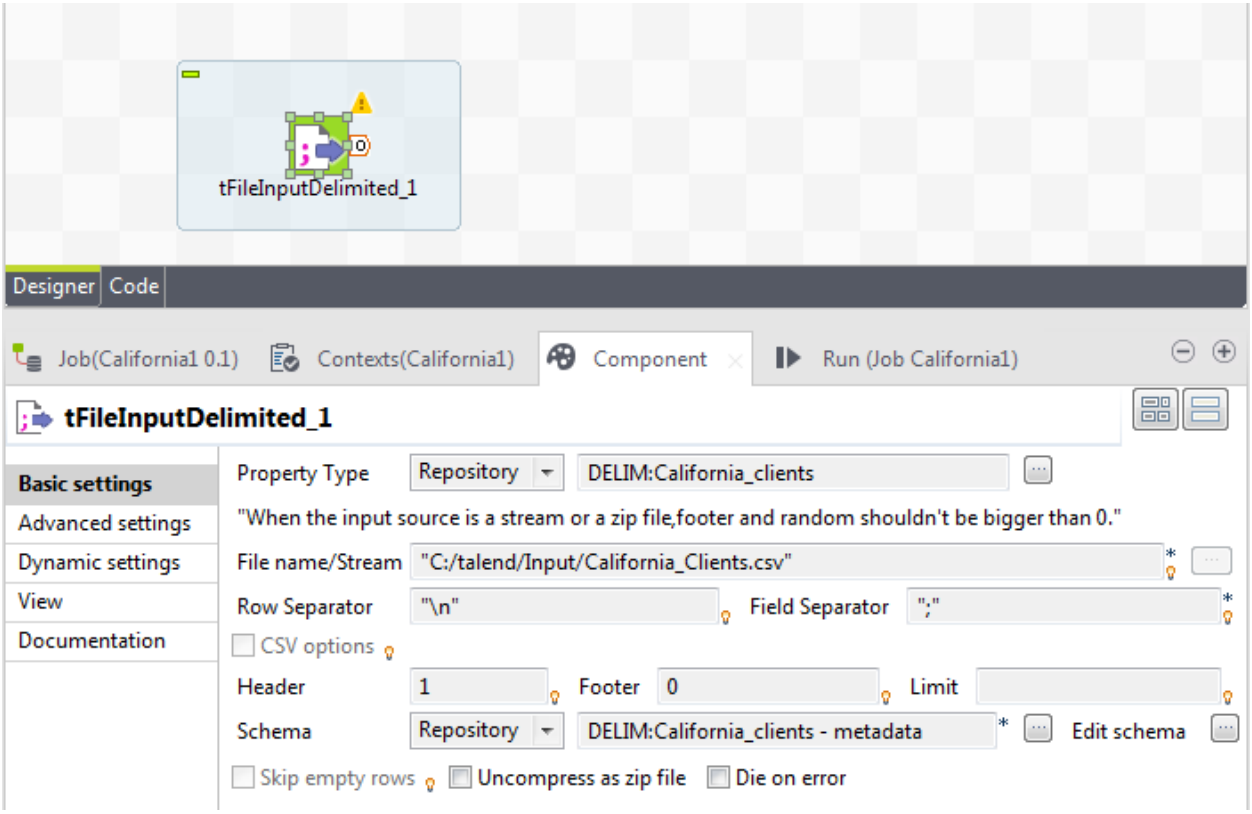
   L'assistant spécifique aux fichiers délimités s'ouvre :

   - A l'étape 1, seul le champ **Name** est obligatoire : saisissez le nom **California_clients** et passez à l'étape suivante.
   - A l'étape 2, sélectionnez le fichier d'entrée (`California_Clients.csv`) via le bouton **Browse...**. Un extrait du fichier apparaît immédiatement dans la zone **File viewer** en bas de l'assistant, afin que vous puissiez en vérifier le contenu. Cliquez sur **Next**.
   - A l'étape 3, définissez les paramètres du fichier : encodage, séparateurs de colonnes et de lignes, etc. Puisque votre fichier d'entrée est standard, vous pouvez garder la plupart des valeurs par défaut. La première ligne de votre fichier est un en-tête contenant les noms des colonnes. Pour récupérer automatiquement ces noms, cochez la case **Set heading row as column names** et cliquez sur **Refresh Preview**. Cliquez sur **Next** pour passer à l'étape suivante.
   - A l'étape 4, définissez chaque colonne de votre fichier. L'assistant intègre des algorithmes qui essayent de deviner le type et la longueur des données contenues dans les colonnes du fichier en analysant les premières lignes. La description des données (appelé schéma dans le Studio Talend) peut être modifiée à tout moment. Pour ce scénario particulier, ces informations peuvent être gardées telles quelles.

   La métadonnée **California_clients** est maintenant définie.

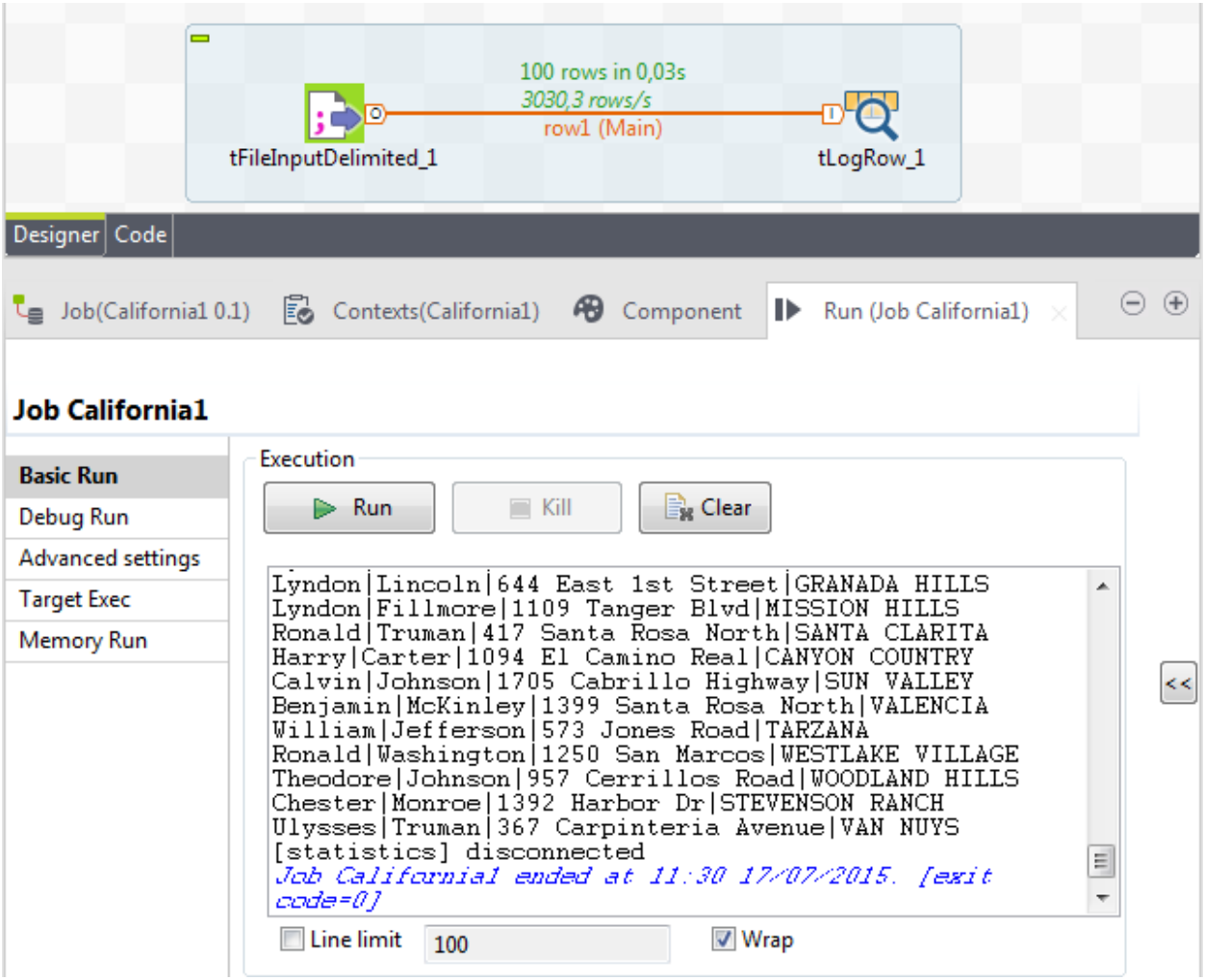
   Vous pouvez donc l'utiliser dans votre composant d'entrée. Sélectionnez le composant **tFileInputDelimited** que vous avez déposé dans l'espace de modélisation et sélectionnez la vue **Component Settings** dans le bas de la fenêtre.
7. Sélectionnez l'onglet vertical **Basic settings**. Dans cet onglet, vous trouverez toutes les propriétés techniques nécessaires au composant. Au lieu de les saisir une à une, utilisez la métadonnée que vous venez de créer.
8. Sélectionnez **Repository** dans la liste déroulante **Property type**. Un nouveau champ apparaît : cliquez sur le bouton **[...]** et sélectionnez la métadonnée correspondante dans la liste, **California_clients**.

   Notez que tous les paramètres sont automatiquement renseignés.



A cette étape, terminez votre flux en envoyant tout simplement les données lues dans le fichier d'entrée vers une sortie standard (StdOut).

9. Pour ce faire, ajoutez un composant **tLogRow** (de la famille **Logs & Errors**). Pour lier ces deux composants, cliquez-droit sur le composant d'entrée et sélectionnez **Row** > **Main**. Puis cliquez sur le composant de sortie **tLogRow**.
10. Ce Job est maintenant prêt à être exécuté. Pour l'exécuter, sélectionnez la vue **Run** dans le bas de la fenêtre.
11. Activez les statistiques en cochant la case **Statistics** dans l'onglet **Advanced settings** de la vue **Run**, puis exécutez le Job en cliquant sur le bouton **Run**, dans l'onglet **Basic Run**.



Le contenu du fichier d'entrée apparaît dans la console de la vue **Run**.

Etape 2 : Mapping et transformations

Vous allez maintenant enrichir votre Job en ajoutant des transformations à la volée. Pour effectuer ces transformations, utilisez le composant **tMap** dans votre Job. Ce composant est multiple et peut gérer des :

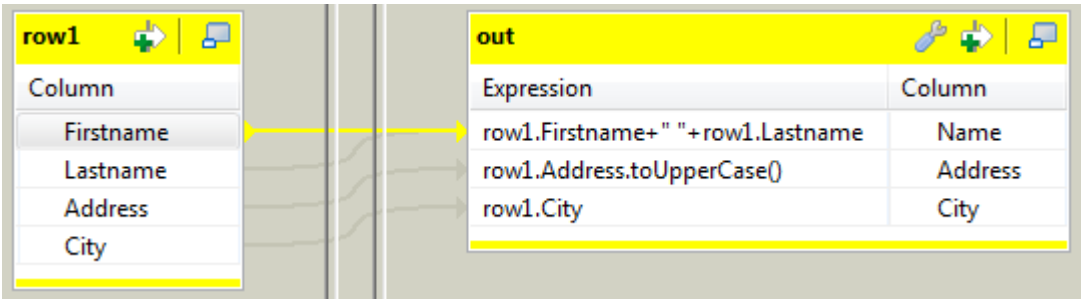
- entrées et sorties multiples,
- recherches de référence (simple, produit cartésien, première et dernière correspondance, etc.),
- jointures (inner join, outer join),
- transformations,
- rejets,
- etc.

Procédure

1. Supprimez la connexion reliant vos deux composants via un clic-droit sur cette connexion et en sélectionnant l'option **Delete**. Puis placez le **tMap** entre les deux autres composants et reliez-le au composant d'entrée comme vous l'avez fait précédemment.

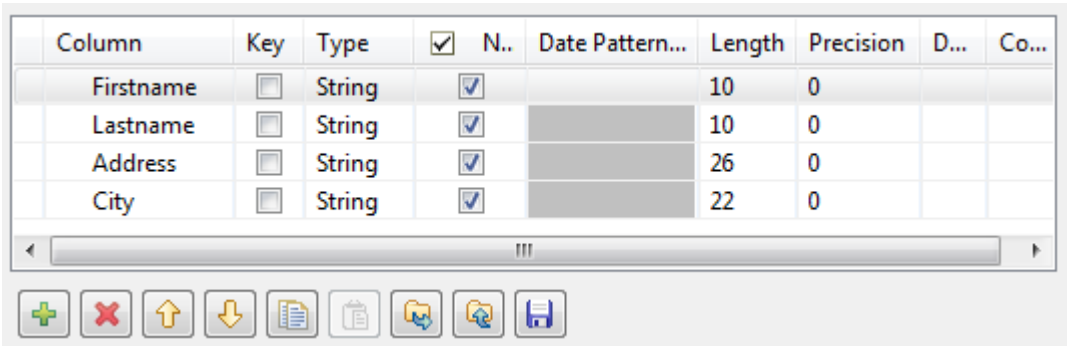
2. Enfin, pour lier le composant **tMap** à la sortie standard, cliquez-droit sur le **tMap** et sélectionnez **Row** > ***New Output* (Main)** et cliquez sur le composant **tLogRow** pour créer la connexion. Saisissez `out1` dans la boîte de dialogue. Logiquement, une boîte de dialogue apparaît (pour la rétro-propagation des schémas), ignorez-la en cliquant sur **No**.
3. Maintenant, double-cliquez sur le **tMap** pour accéder à son interface.

   A gauche, vous trouverez le schéma (description) de votre fichier d'entrée (**row1**). A droite, votre sortie est encore vide pour le moment (**out1**).
4. Déposez les colonnes **FirstName** et **LastName** de la gauche vers la droite dans la colonne **Name**, comme le montre la capture d'écran suivant. Puis déposez les autres colonnes **Address** et **City** dans leur ligne respective.



5. Puis effectuez les transformations suivantes sur chaque colonne :
   - Changez les données de la colonne **Name** de la manière suivante : `row1.Firstname + " " + row1.LastName`.

     Cette action concatène les colonnes **Firstname** et **Lastname** dans une seule colonne.
   - Changez les données de la colonne **Address** de la manière suivante : `row1.Address.toUpperCase()`. Cette action met l'adresse en majuscule.
6. Puis supprimez la colonne LastNameLastName de la table out1out1, et augmentez la longueur des colonnes restantes. Pour cela, cliquez sur l'onglet Schema Editor situé en bas de l'éditeur du [Map Editor] et procédez comme suit :



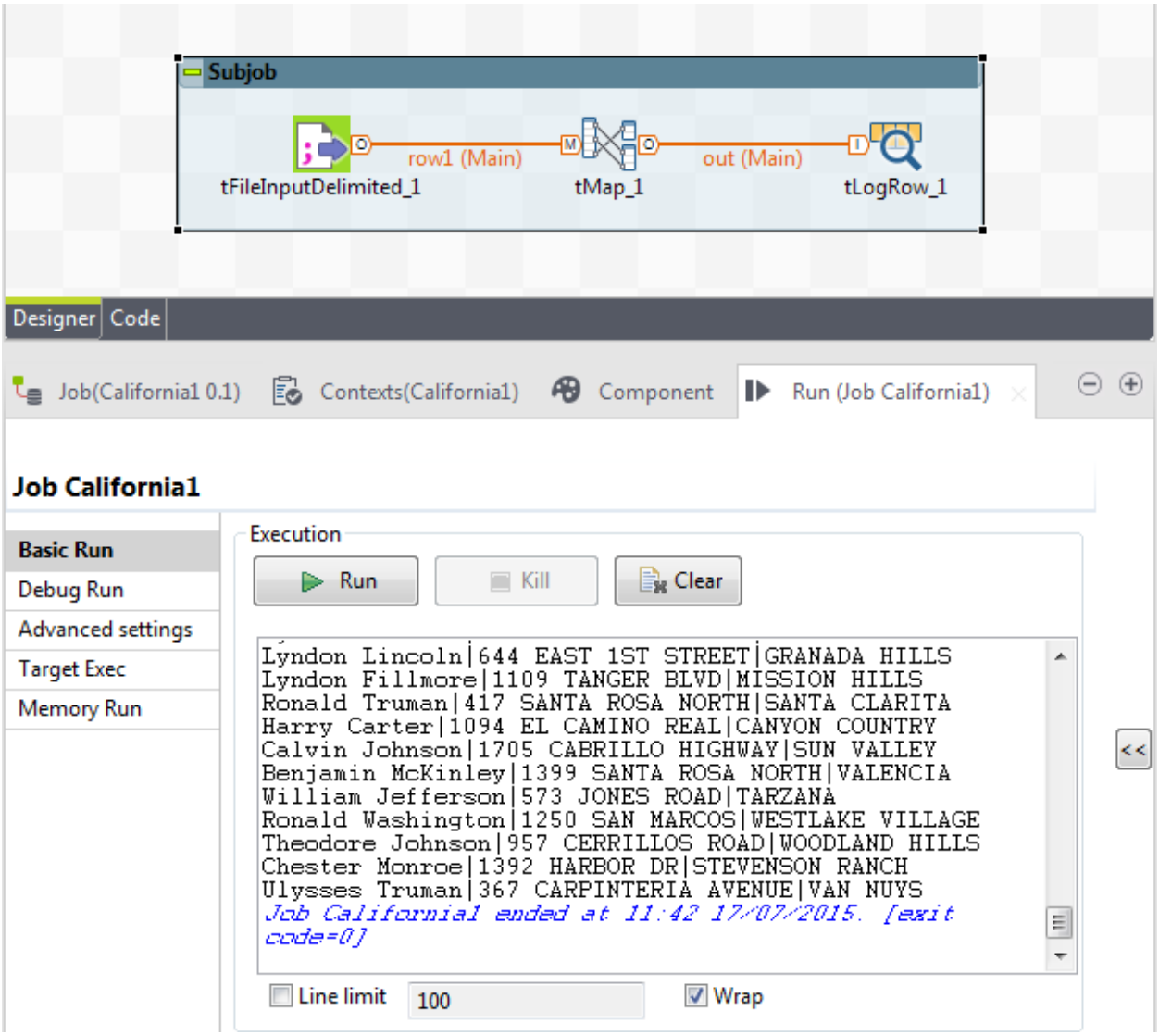
   1. Sélectionnez la colonne à supprimer du schéma, et cliquez sur l'icône représentant une croix rouge.
   2. Sélectionnez la colonne dont vous souhaitez augmenter la longueur.
   3. Saisissez la longueur que vous voulez dans la colonne **Length**. Dans cet exemple, modifiez la longueur de chaque colonne restante en 40.

Remarque :

Comme les noms et prénoms des clients sont concaténés, il est nécessaire d'augmenter la longueur de la colonne *name*name, afin de prendre en compte la longueur complète du nom.

Aucune transformation n'est effectuée sur la colonne **City**.

7. Cliquez sur **OK** pour valider les modifications et fermer l'éditeur.
8. Si vous exécutez votre Job à cette étape (via l'onglet **Run**, comme précédemment), vous remarquerez que les changements que vous avez apportés ont été implémentés.



L'adresse a été mise en majuscule et les prénoms et noms ont été regroupés dans une seule colonne.

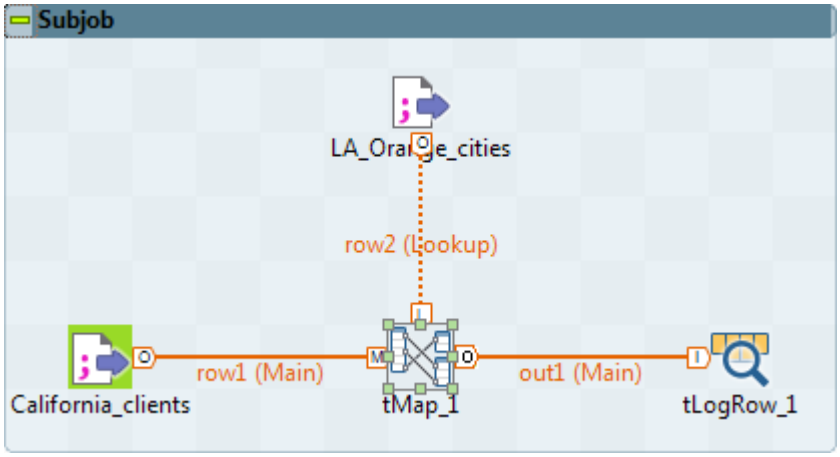
Etape 3 : Définition du fichier de référence, mapping des données de référence, sélection du mode Inner Join

Procédure

1. Définissez la métadonnée correspondant au fichier `LosAngelesandOrangeCounties.txt` à l'aide de l'assistant, comme vous l'avez fait dans l'étape 1 avec le fichier `California_clients`.

   A l'étape 1 de l'assistant, nommez cette métadonnée `LA_Orange_cities`.

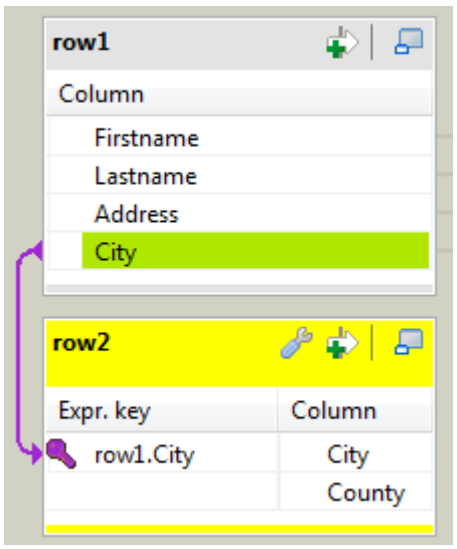
2. Puis déposez cette nouvelle métadonnée en haut de l'espace de modélisation, cela créera automatiquement un composant de lecture pointant vers cette métadonnée.
3. Reliez ce composant au **tMap**.



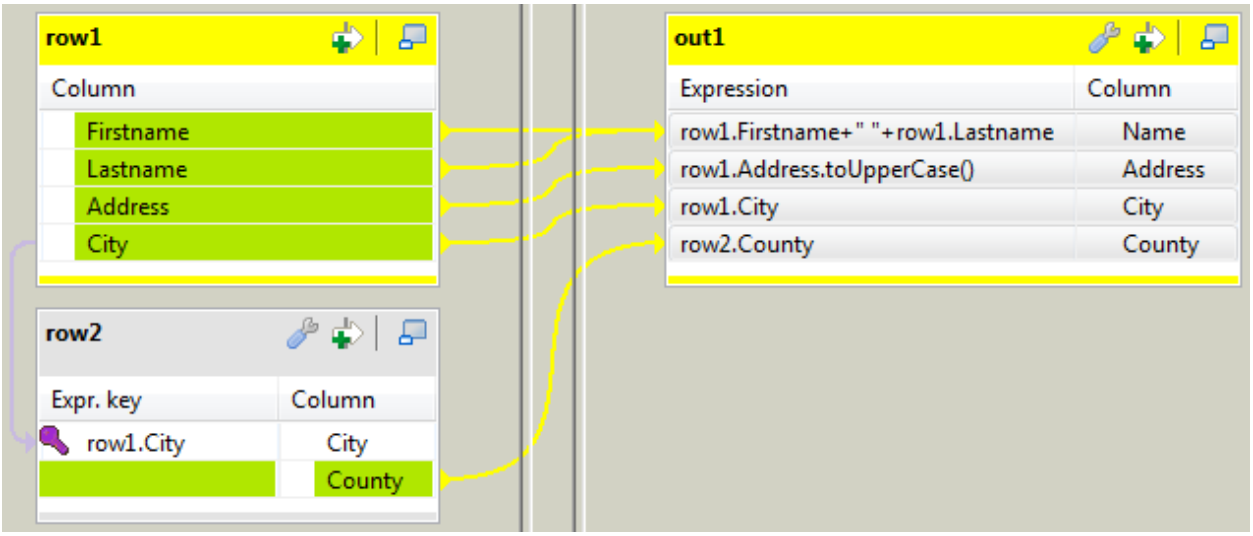
4. Double-cliquez de nouveau sur le composant **tMap** pour ouvrir son interface. Notez que la table de référence (**row2**) correspondant au fichier `LosAngelesandOrangeCounties.txt`, apparaît à gauche de la fenêtre dans la zone **Input** d'entrée, juste en dessus de votre flux d'entrée principal (**row1**).
5. Maintenant, définissez la jointure entre le flux principal et le flux de référence.

   Dans ce scénario, la jointure est simple à définir puisque la colonne **City** est présente dans les deux fichiers d'entrée et que les données correspondent parfaitement. Mais si ça n'avait pas été le cas, il aurait été possible de rapprocher les données (padding, changement de casse, etc.) directement à ce niveau.

   Pour établir la jointure, déposez la colonne **City** de la première table d'entrée vers la colonne **City** de la table de référence. Un lien violet apparaît pour matérialiser cette jointure.

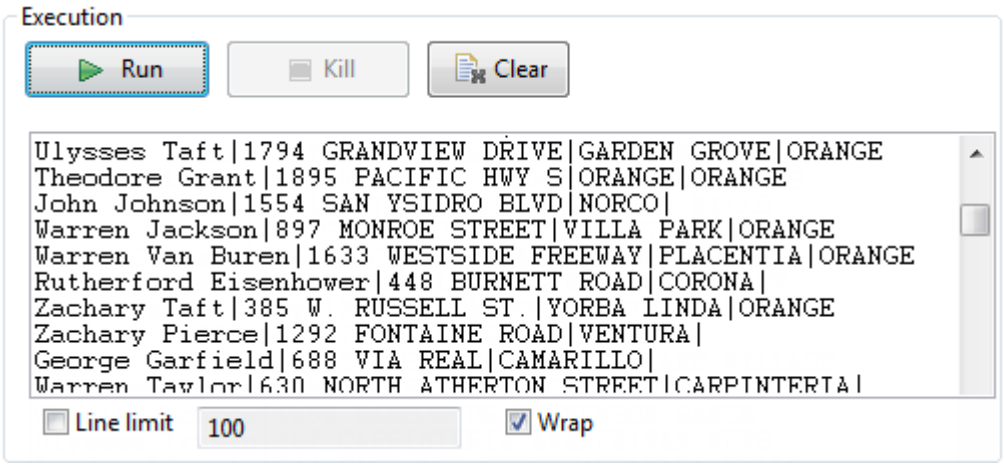


   Maintenant, vous pouvez utiliser la colonne **County** de la table de référence dans la table de sortie (**out1**).



6. Enfin, cliquez sur le bouton **OK** pour valider les modifications et exécutez ce nouveau Job.

   La sortie suivante s'affichera dans la console :



   Comme vous pouvez le voir, la dernière colonne ne contient que les villes des régions d'Orange et de Los Angeles. Pour les autres villes, cette colonne reste vide. Ceci est dû au fait que par défaut le **tMap** établit une jointure Left Outer Join. Si vous souhaitez appliquer un filtre permettant de n'afficher que les données pour lesquelles une correspondance a été trouvée par le **tMap**, cliquez sur le bouton **tMap settings** et sélectionnez **Inner Join** dans la liste **Join Model** sur la table de référence (**row2**).

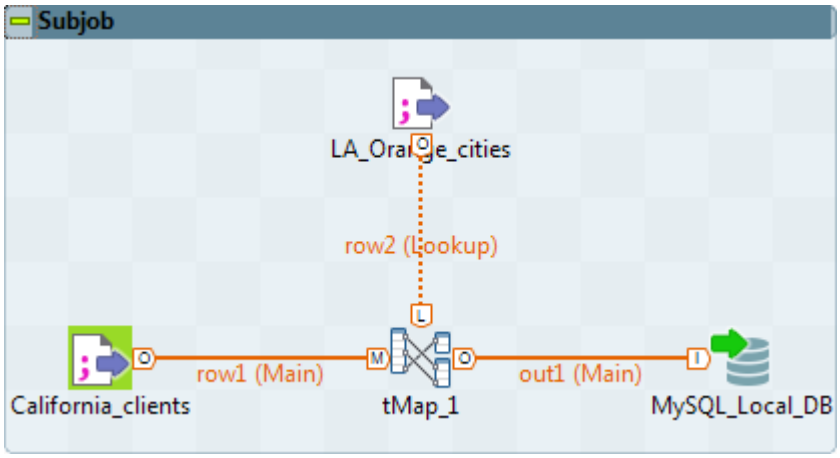
Etape 4 : Sortie vers une table MySQL

Votre Job fonctionne à merveille. Pour le finaliser, redirigez le flux de sortie vers une table MySQL.

Procédure

1. Pour cela, créez tout d'abord une métadonnée décrivant la connexion à la base de données MySQL. Double-cliquez sur **DemoMySQL** dans le répertoire **Metadata** > **MySQL** du Référentiel (à condition que vous ayez bien importé le projet **Demo**) pour lancer l'assistant **[Metadata]**.
2. A l'étape 2 de l'assistant, renseignez les paramètres de connexion à la base de données. Vérifiez la validité de cette connexion en cliquant sur le bouton **Check**. Enfin, validez vos modifications en cliquant sur **Finish**.

3. Déposez cette métadonnée à droite de l'espace de modélisation en maintenant la touche **Ctrl** enfoncée pour créer automatiquement un composant **tMysqlOutput**.
4. Supprimez le composant **tLogRow** de votre Job.
5. Reconnectez le flux de sortie out1out1 du tMap vers le composant tMysqlOutput.



6. Dans l'onglet **Basic settings** de ce composant :
   a) Saisissez `LA_Orange_clients` dans le champ **Table** pour nommer votre table cible qui va être créée à la volée.
   b) Sélectionnez l'option **Drop table if exists and create** dans le champ **Action on table**.
   c) Cliquez sur **Edit Schema** et sur le bouton **Reset DB type** (le bouton en forme de base de données dans la barre d'outils) pour renseigner automatiquement le type de base de données, si nécessaire.
7. Exécutez à nouveau le Job.

Résultats

La table cible devrait être automatiquement créée et remplie en moins d'une seconde.

Dans ce scénario, seuls quatre composants différents sont utilisés, mais la **Palette** en contient plus de 450 (bases de données, Webservices, FTP, etc.).

D'autres composants, réalisés cette fois par la communauté, sont disponibles sur le site communautaire : talendforge.org.

Utilisation de la fonctionnalité Use Output Stream

Le scénario suivant a pour objectif de montrer comment utiliser la fonctionnalité de flux de sortie dans un certain nombre de composants, afin d'améliorer considérablement les performances en sortie.

Dans ce scénario, un fichier .csv prédéfini contenant des informations client est chargé dans une table d'une base de données. Les données chargées sont sélectionnées à l'aide d'un composant **tMap** et écrites dans un fichier de sortie local, ainsi que dans la console, via la fonctionnalité **Output stream**.

Pour consulter les principes de la fonctionnalité **Use Output Stream**, consultez la section *Utiliser la fonctionnalité Use Output Stream* du Guide utilisateur du Studio Talend à l'adresse https://help.talend.com.

Données d'entrée

Le fichier d'entrée, dont les données seront chargées dans la table de la base de données, contient des informations clients variées.

La structure du fichier appelée **Schema** dans le Studio Talend comprend les colonnes suivantes :

- id (Type : Integer)
- CustomerName (Type : String)
- CustomerAge (Type : Integer)
- CustomerAddress (Type : String)
- CustomerCity (Type : String)
- RegisterTime (Type : Date)

Données de sortie

Le composant **tMap** est utilisé pour sélectionner les colonnes **id**, **CustomerName** et **CustomerAge** dans les données d'entrée. Les données sélectionnées sont écrites en sortie via la fonctionnalité de flux de sortie.

Les données attendues en sortie doivent avoir la structure suivante :

- id (Type : Integer)
- CustomerName (Type : String)
- CustomerAge (Type : Integer)

Ces trois colonnes proviennent des colonnes des données d'entrée.

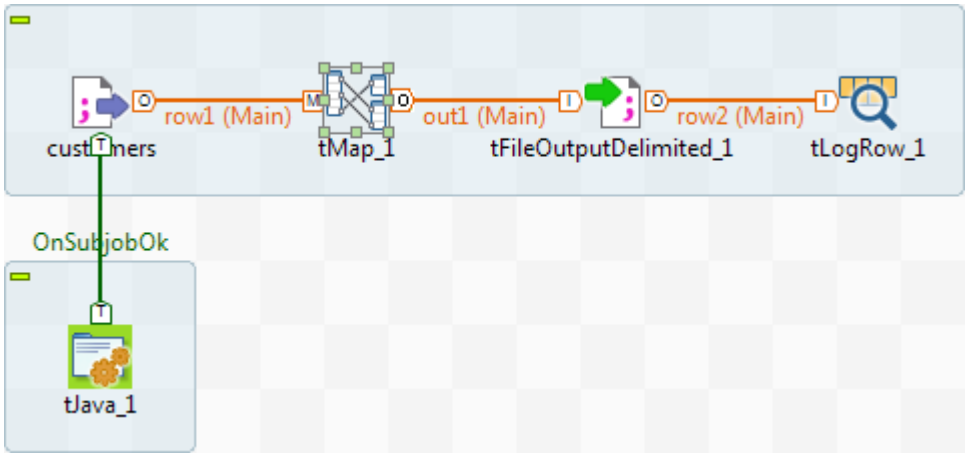
Création du Job

Pour créer ce Job, vous devez effectuer les quatre étapes suivantes :

1. Création du Job, configuration du schéma d'entrée et lecture du fichier d'entrée selon le schéma défini.

2. Définition de la commande activant la fonctionnalité de flux de sortie.
3. Mapping des données via le composant **tMap**.
4. Ecriture en sortie du flux de données sélectionné.

Vous pouvez voir le Job terminé dans la capture d'écran ci-dessous. Pour consulter les instructions détaillées relatives à la création du Job, lisez les sections suivantes.

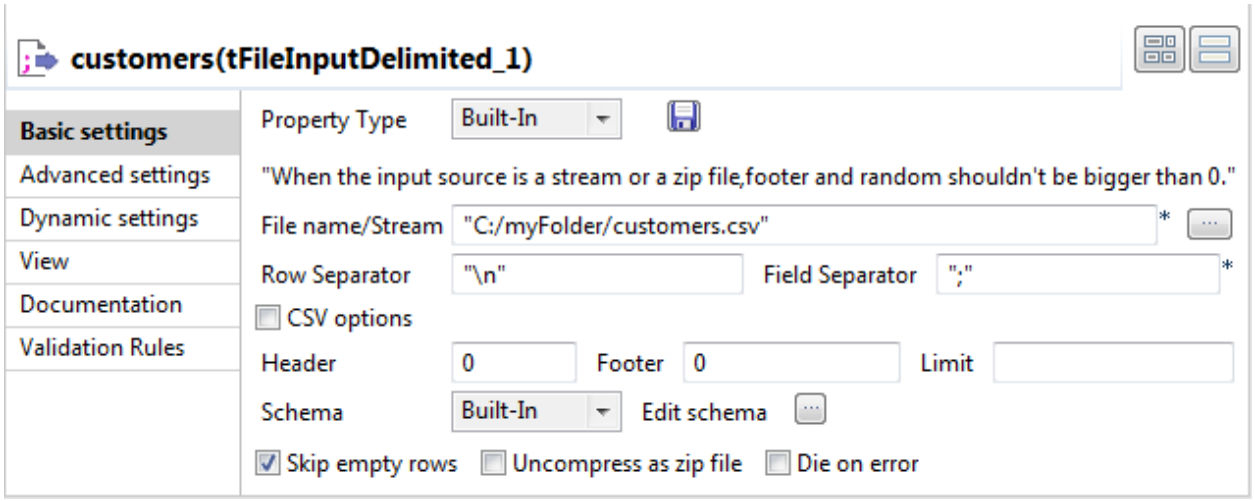


Étape 1 : Lire les données d'entrée d'un fichier local

Utilisez le composant **tFileInputDelimited** pour lire le fichier `customers.csv` contenant les données d'entrée. Ce composant se trouve dans la famille **File/Input** de la **Palette**. Cliquez sur ce puis déposez-le dans l'espace de modélisation graphique.

Procédure

1. Double-cliquez sur le composant **tFileInputDelimited** afin d'ouvrir sa vue **Basic settings** et définir ses propriétés de base.



2. Cliquez sur le bouton **[...]** à côté du champ **File name/Stream** et parcourez votre système jusqu'à votre fichier d'entrée. Vous pouvez également saisir manuellement le chemin d'accès à ce fichier.
3. Cliquez sur **Edit schema** pour ouvrir une boîte de dialogue dans laquelle configurer la structure du fichier d'entrée.
4. Cliquez six fois sur le bouton **[+]** pour ajouter six colonnes, puis, dans la colonne **Type**, sélectionnez **Integer** pour les colonnes **id** et **CustomerAge**, **String** pour les colonnes

CustomerName, **CustomerAddress** et **CustomerCity**. Sélectionnez **Date** pour la colonne **RegisterTime**.

Column	Key	Type	N..	Date Patt...	Length	Pre...	D...	Co...
id	☐	Integer	☑					
CustomerName	☐	String	☑					
CustomerAge	☐	Integer	☑					
CustomerAddress	☐	String	☑					
CustomerCity	☐	String	☑					
RegisterTime	☐	Date	☑	"dd-MM...				

5. Cliquez sur **OK** pour fermer la boîte de dialogue.

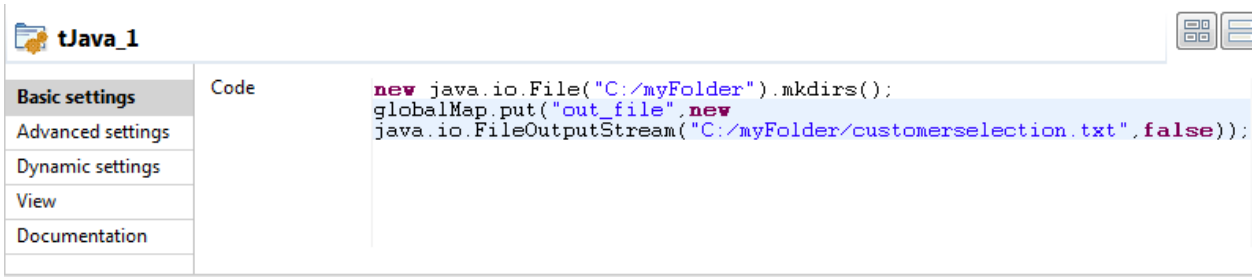
Étape 2 : Configurer la commande pour activer la fonctionnalité Output Stream

Utilisez le **tJava** pour définir la commande de création d'un fichier de sortie et un répertoire contenant le fichier de sortie.

Pour ce faire, déposez un composant **tJava** dans l'espace de modélisation graphique.

Procédure

1. Double-cliquez sur le **tJava** pour ouvrir sa vue **Basic settings** et définir ses propriétés.

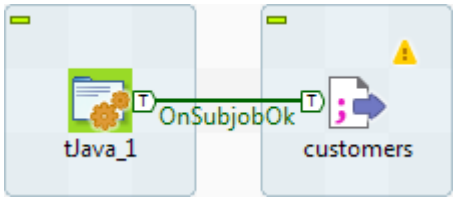


2. Dans le champ **Code**, saisissez la commande suivante :

```
new java.io.File("C:/myFolder").mkdirs();
globalMap.put("out_file",new java.io.FileOutputStream("C:/myFolder/cust
omerselection.txt",false));
```

> **Remarque :**
>
> La commande saisie ci-dessus créé un nouveau répertoire `C:/myFolder` pour sauvegarder le fichier de sortie `customerselection.txt`. Vous pouvez personnaliser la commande selon vos besoins.

3. Reliez le **tJava** au **tFileInputDelimited** à l'aide d'un lien **Trigger** > **On Subjob Ok**. Cela déclenche le **tJava** lorsque le sous-job commençant par le **tFileInputDelimited** est correctement exécuté.

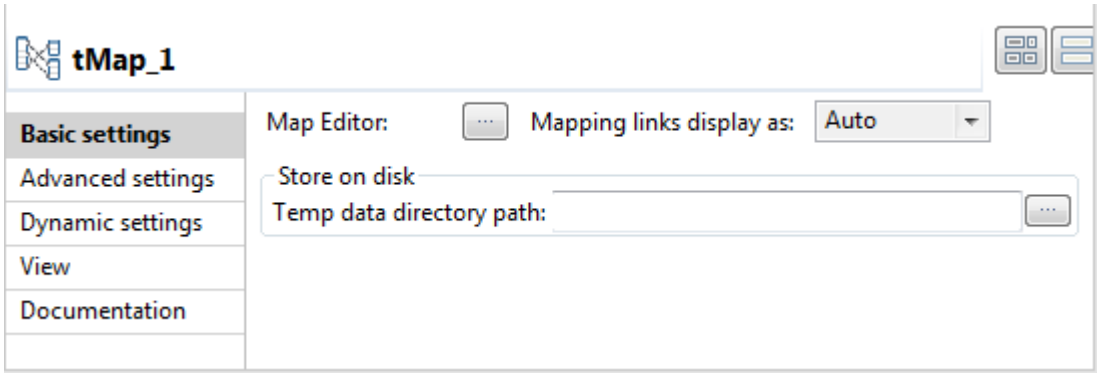


Étape 3 : Mapper les données à l'aide du composant tMap

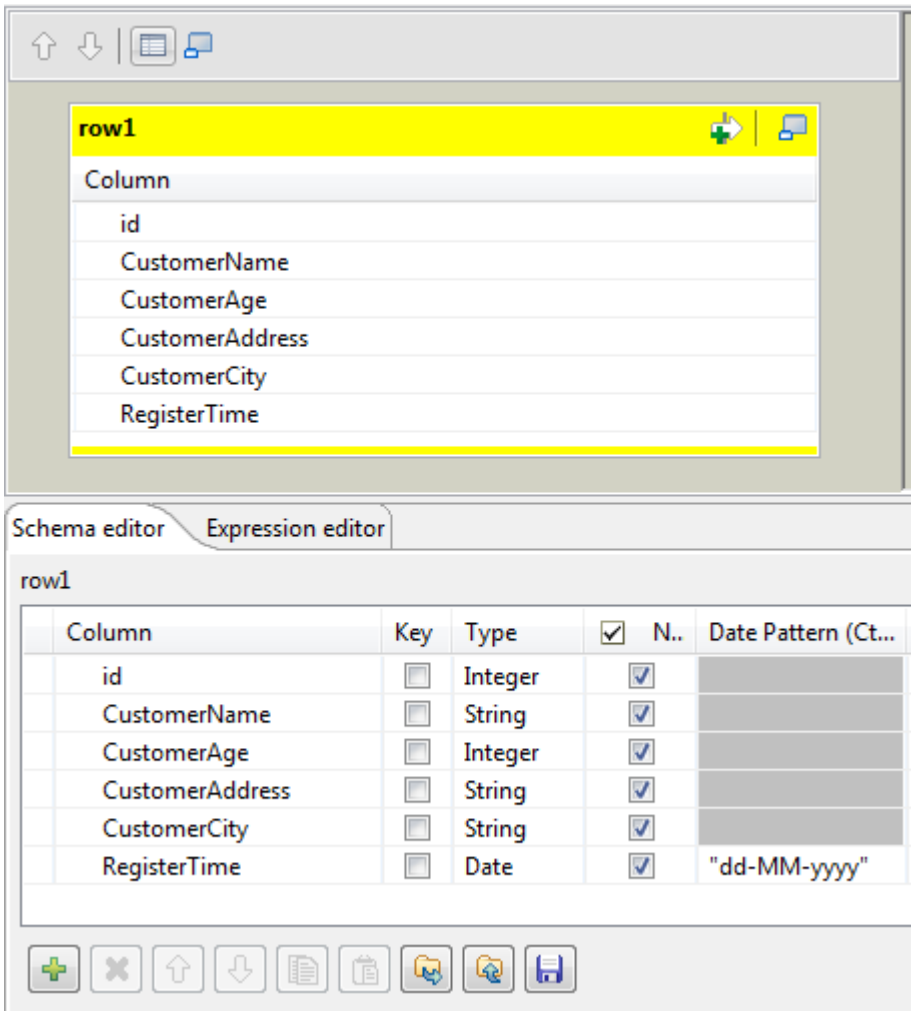
Déposez un composant **tMap** dans l'espace de modélisation graphique.

Procédure

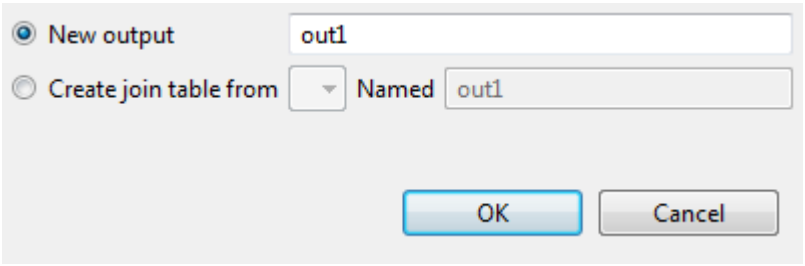
1. Double-cliquez sur le **tMap** afin d'afficher sa vue **Basic settings** et configurer ses propriétés de base.



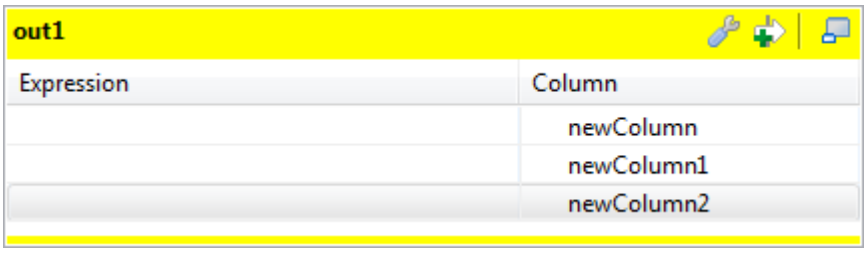
2. Cliquez sur le bouton **[...]** à côté du **Map Editor** pour ouvrir une boîte de dialogue dans laquelle configurer le mapping.
3. Cliquez sur le bouton **[+]** de gauche pour ajouter six colonnes au schéma d'entrée, ces colonnes devant être **id**, **CustomerName**, **CustomerAge**, **CustomerAddress**, **CustomerCity**, **RegisterTime**.



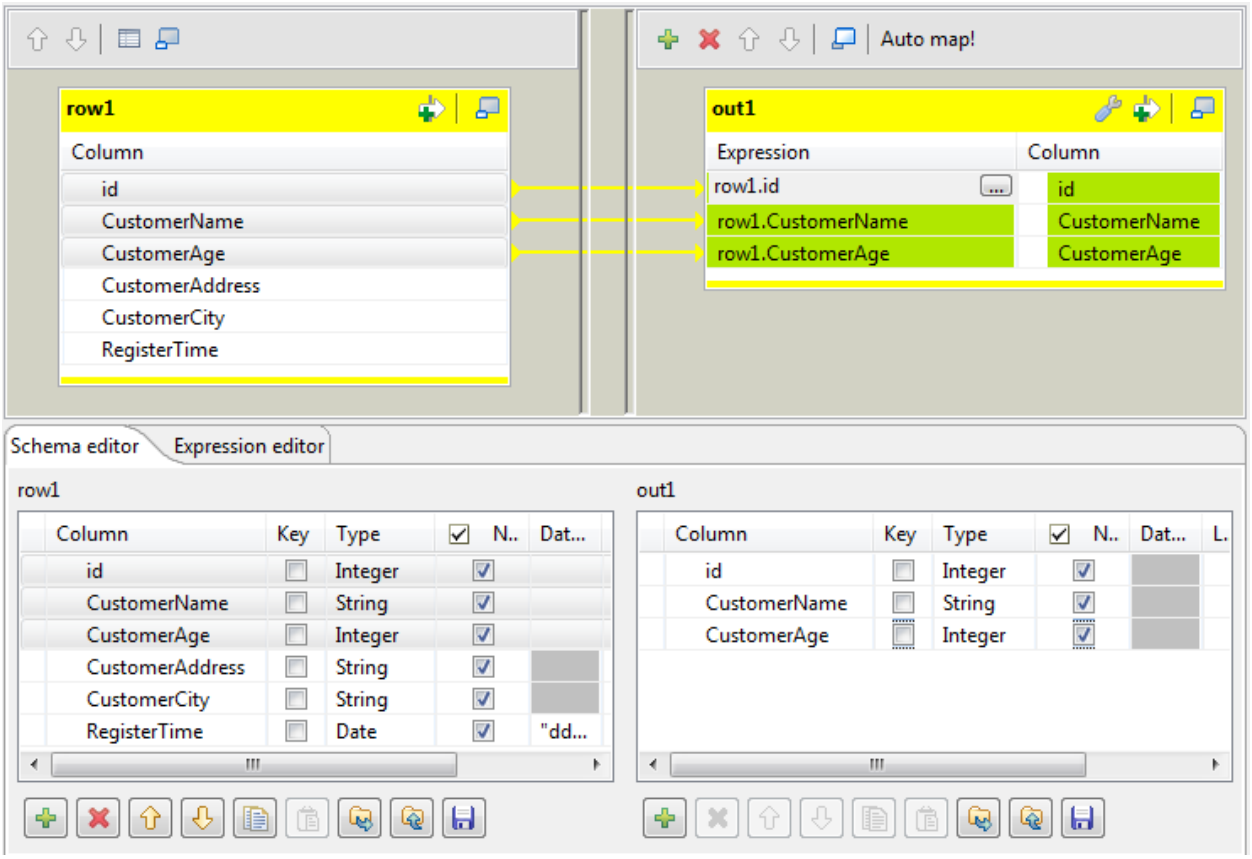
4. Cliquez sur le bouton **[+]** à droite pour ajouter un schéma de sortie.



5. Sélectionnez **New output** et cliquez sur **OK** pour sauvegarder votre schéma de sortie. Le schéma de sortie est vide.
6. Cliquez sur le bouton **[+]** sous la table **out1** pour ajouter trois colonnes aux données de sortie.



7. Déposez les colonnes **id**, **CustomerName** et **CustomerAge** de la gauche à la droite, dans leurs lignes respectives.



8. Cliquez sur **OK** pour sauvegarder les paramètres.

Étape 4 : Ecrire en sortie le flux de données sélectionné

Procédure

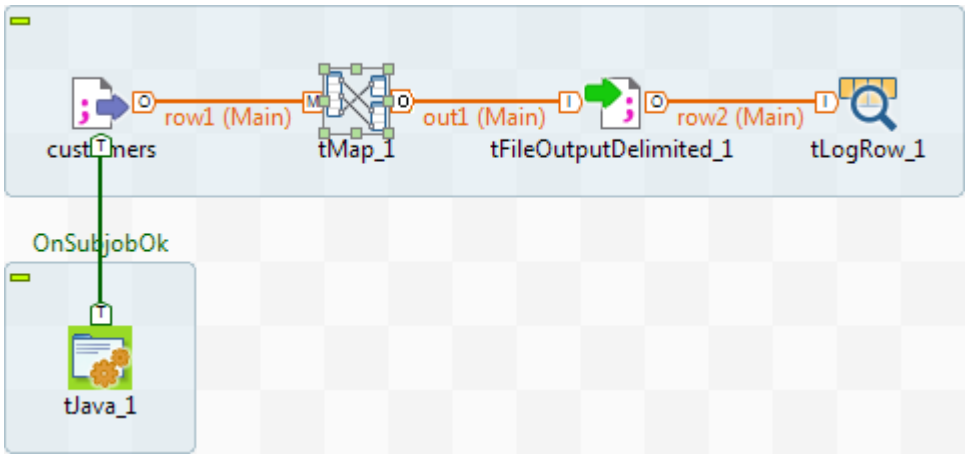
1. Déposez un composant **tFileOutputDelimited** dans l'espace de modélisation graphique, et double-cliquez sur le **tFileOutputDelimited** afin d'ouvrir sa vue **Basic settings** et configurer ses propriétés de base.
2. Cochez la case **Use Output Stream** pour activer le champ **Output Stream** et saisissez dans le champ **Output Stream** la commande suivante :

```
(java.io.OutputStream)globalMap.get("out_file")
```

> **Remarque :**
>
> Vous pouvez personnaliser la commande dans le champ **Output Stream** en appuyant sur les touches **Ctrl+Espace** pour utiliser l'autocomplétion et sélectionner des commandes built-in. Vous pouvez également saisir manuellement votre commande. Dans ce scénario, la commande utilisée dans le champ **Output Stream** appelle la classe `java.io.OutputStream` pour écrire le flux de données filtré dans un fichier local, spécifié dans la zone **Code** du **tJava**.

3. Reliez le **tFileInputDelimited** au **tMap** à l'aide d'un lien **Row** > **Main** puis reliez le **tMap** au **tFileOutputDelimited** à l'aide du lien **Row** > **out1**, défini dans le **Map Editor** du **tMap**.
4. Cliquez sur le bouton **Sync columns** pour récupérer le schéma du composant précédent.
5. Déposez un **tLogRow** de la famille **Logs & Errors** dans l'espace de modélisation graphique. Double-cliquez sur ce composant afin d'ouvrir sa vue **Basic settings**
6. Sélectionnez l'option **Table** dans la zone **Mode**.
7. Reliez le **tFileOutputDelimited** au **tLogRow** à l'aide d'un lien **Row** > **Main**.
8. Cliquez sur **Sync columns** pour récupérer le schéma défini dans le composant précédent.

   Ce Job est maintenant prêt à être exécuté.



9. Appuyez sur les touches **Ctrl+S** afin de sauvegarder votre Job et appuyez sur **F6** pour l'exécuter.

   Le contenu de données sélectionnées s'affiche dans la console.

```
Starting job OutputStream at 17:31 19/10/2011.

[statistics] connecting to socket on port 4059
[statistics] connected
.-----+------------+-----------.
|           tLogRow_1            |
|=====+============+===========|
|id   |CustomerName|CustomerAge|
|=====+============+===========|
|10001|Warren      |67         |
|10002|Woodrow     |68         |
|10003|Grover      |77         |
|10004|Abraham     |74         |
|10005|Chester     |78         |
|10006|Calvin      |63         |
|10007|Zachary     |53         |
|10008|Chester     |36         |
|10009|Chester     |60         |
|10010|Woodrow     |57         |
'-----+------------+-----------'
[statistics] disconnected
Job OutputStream ended at 17:31 19/10/2011. [exit code=0]
```

Les données sélectionnées sont également écrites dans le fichier local `customerselection.txt`.

customerselection.txt

```
1  id;CustomerName;CustomerAge
2  10001;Warren;67
3  10002;Woodrow;68
4  10003;Grover;77
5  10004;Abraham;74
6  10005;Chester;78
7  10006;Calvin;63
8  10007;Zachary;53
9  10008;Chester;36
10 10009;Chester;60
11 10010;Woodrow;57
```

Utilisation de la fonctionnalité de chargement implicite des contextes

La configuration d'un Job selon les variables de contexte vous permet d'orchestrer et d'exécuter vos Jobs dans différentes contextes et environnements. Vous pouvez définir les valeurs de vos variables de contexte lors de leur création ou charger dynamiquement vos paramètres de contexte, explicitement ou implicitement, lors de l'exécution des Jobs.

Le scénario ci-dessous décrit comment utiliser la fonctionnalité de chargement implicite des contextes (Implicit Context Load) de votre Studio Talend afin de charger dynamiquement les paramètres de contexte lors de l'exécution du Job.

Le Job de ce scénario comprend deux composants. Il lit les données des employés stockées dans deux bases de données MySQL, l'une en test et l'autre en production. Les paramètres de connexion pour accéder à ces bases de données sont stockés dans une autre base de données MySQL. Lorsqu'il est exécuté, le Job charge dynamiquement ces paramètres de connexion afin de se connecter aux deux bases de données.

Créer le Job et définir des variables de contexte

Avant de commencer

Créez deux tables nommées `db_testing` et `db_production`, respectivement, dans une base de données MySQL nommée `db_connections`, pour contenir les paramètres de connexion permettant d'accéder aux bases de données susmentionnées, `testing` et `production`. Chaque table doit contenir seulement deux colonnes : `key` et `value`, de type VARCHAR. Voici un exemple du contenu des tables des bases de données :

db_testing :

key	**value**
host	localhost
port	3306
username	root
password	talend
database	testing

db_production :

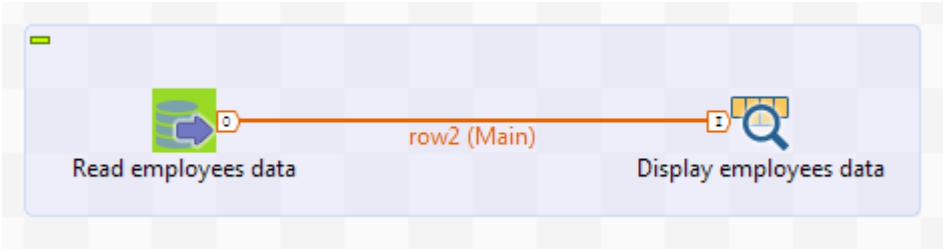
key	**value**
host	localhost
port	3306
username	root
password	talend

key	value
database	production

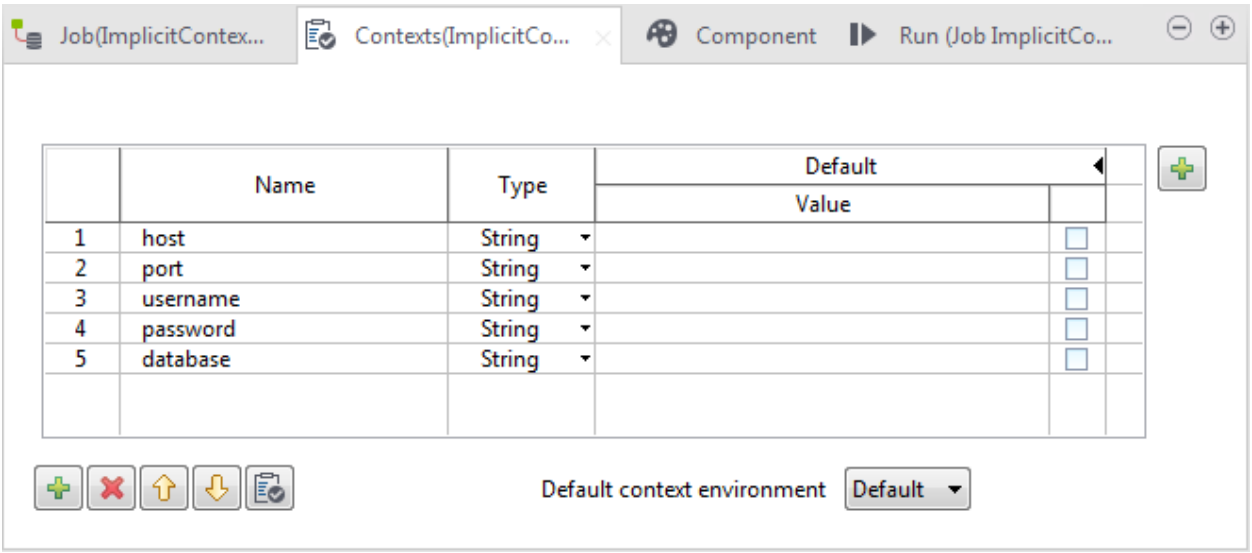
Vous pouvez créer ces tables de bases de données à l'aide d'un autre Job Talend contenant un **tFixedFlowInput** et un **tMysqlOutput**.

Procédure

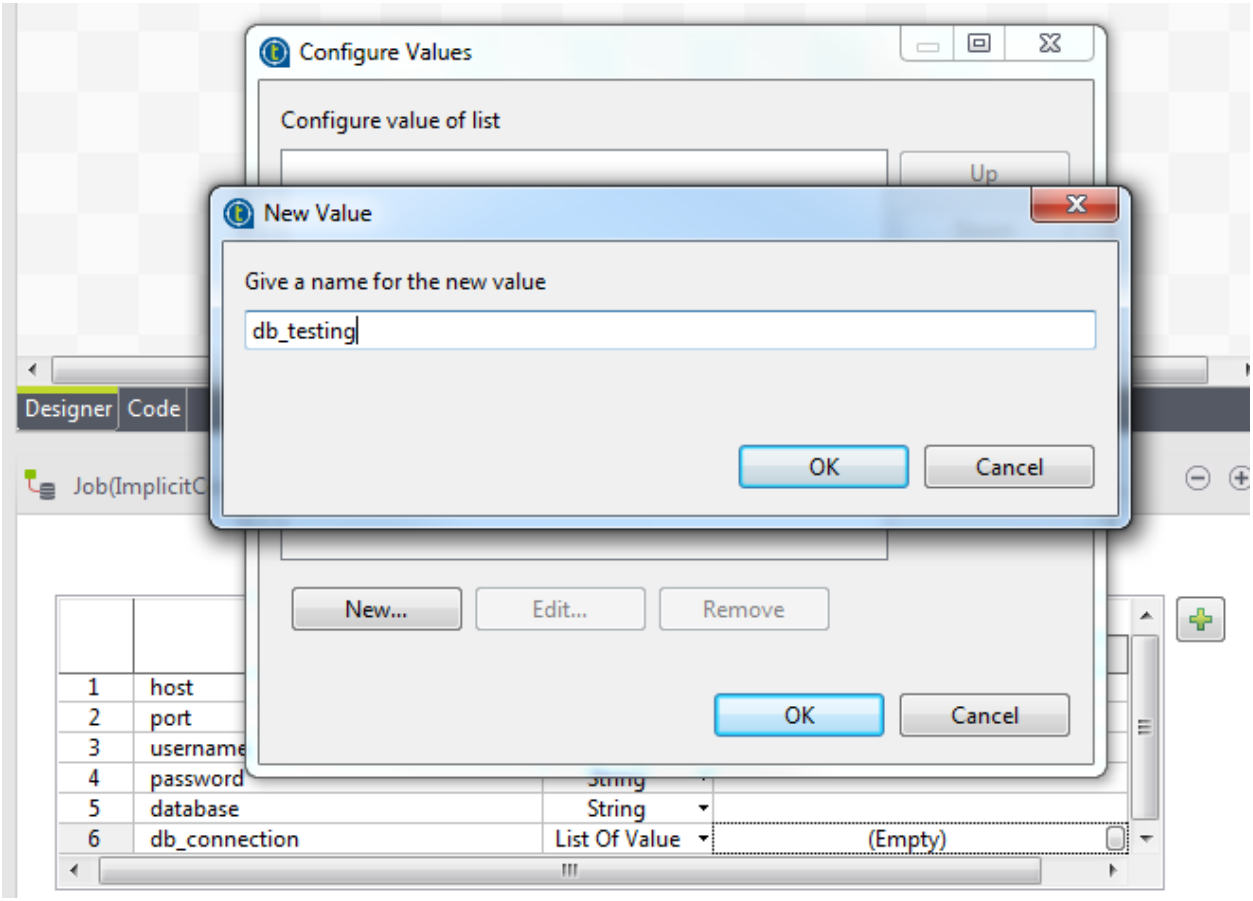
1. Créez un Job et ajoutez un composant **tMysqlInput** et un **tLogRow** dans l'espace de modélisation graphique et reliez-les à l'aide d'un lien **Row** > **Main**.



2. Sélectionnez la vue **Contexts** du Job et cliquez cinq fois sur le bouton **[+]** au bas de la vue, pour ajouter cinq lignes à la table et définir les variables de contexte suivantes, toutes de type **String**. Ne configurez pas les valeurs, car elles seront chargées dynamiquement lors de l'exécution du Job : `host`, `port`, `username`, `password`, et `database`.



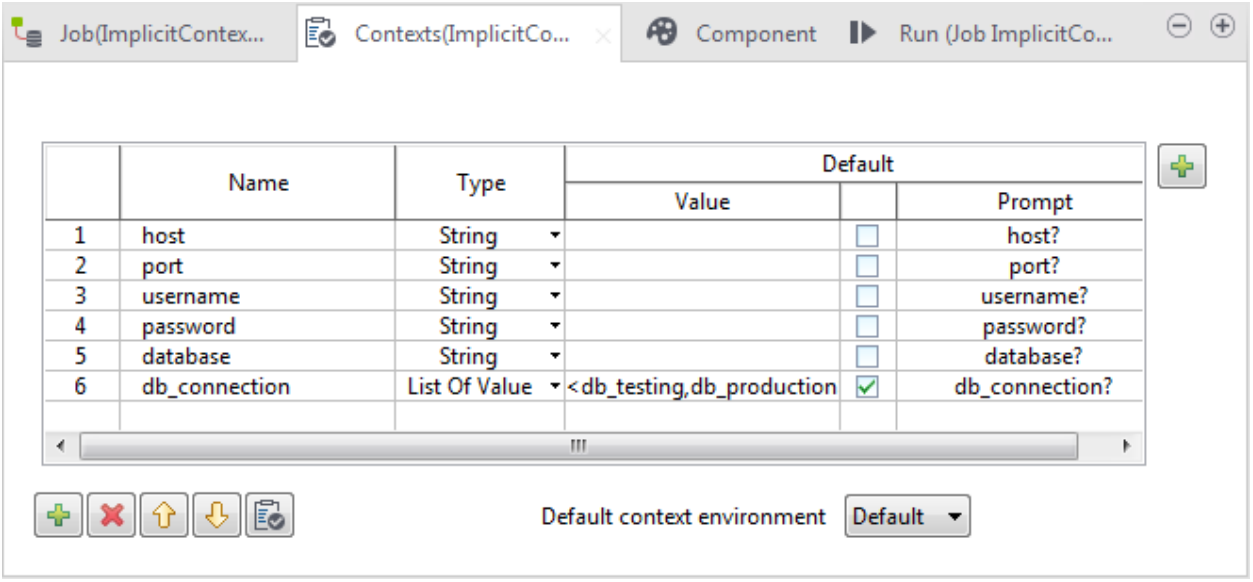
3. Créez une autre variable nommée `db_connection` de type **List Of Value**.
4. Cliquez dans le champ **Value** de la nouvelle variable créée et cliquez sur le bouton qui s'affiche dans la boîte de dialogue **Configure Values**. Cliquez sur **New...** pour ouvrir la boîte de dialogue **New Value**. Saisissez le nom d'une des tables des bases de données contenant les informations de connexion à la base de données puis cliquez sur **OK**.



5. Cliquez sur à nouveau sur **New...** pour définir l'autre table contenant les informations de connexion à la base de données. Cela fait, cliquez sur **OK** afin de fermer la boîte de dialogue **Configure Values**.

   La variable `db_connection` contient une liste de valeurs `db_testing` et `db_production`, les tables de base de données desquelles charger les paramètres de connexion.

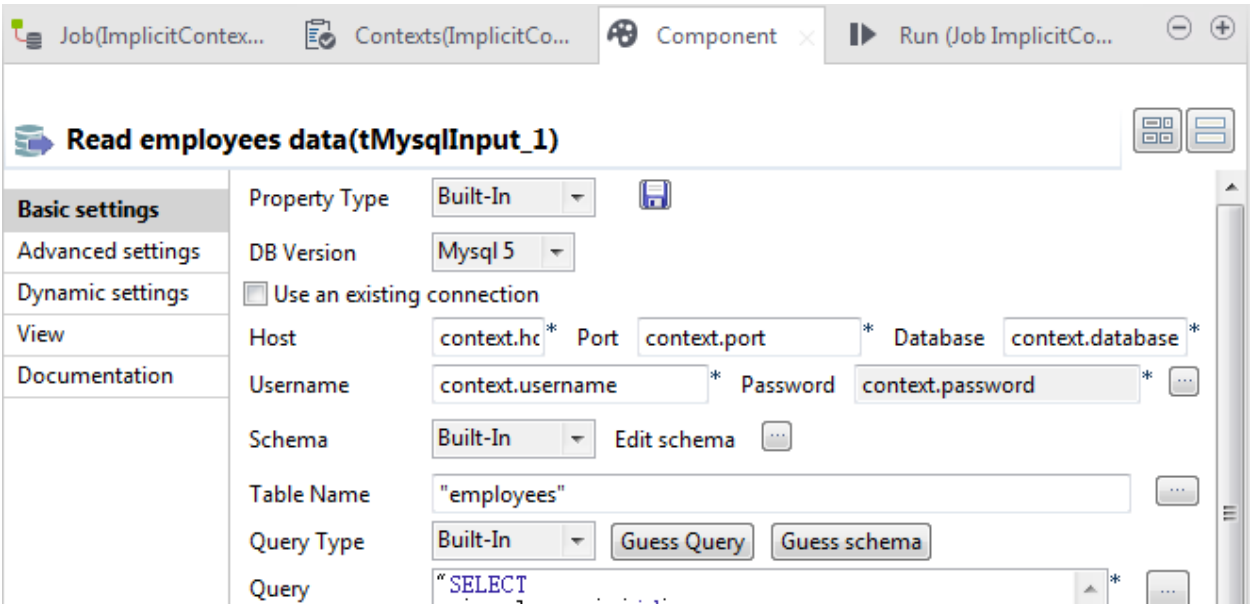
6. Cochez la case **Prompt** à côté du champ **Value** de la variable `db_connection` pour afficher les champs **Prompt** et saisissez le message à afficher lors de l'exécution.



Configurer les composants

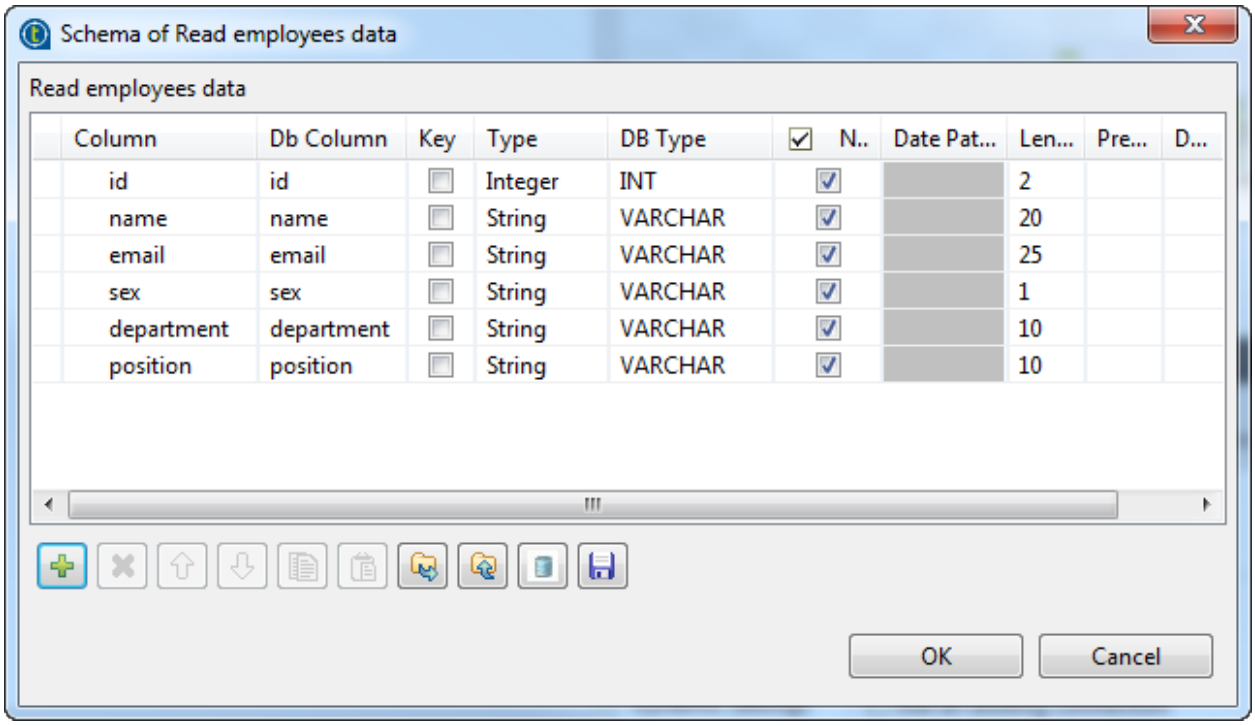
Procédure

1. Double-cliquez sur le **tMysqlInput** pour ouvrir sa vue **Basic settings**.
2. Renseignez les champs **Host**, **Port**, **Database**, **Username**, **Password** et **Table Name** avec les variables correspondantes définies dans l'onglet **Contexts** : `context.host`, `context.port`, `context.database`, `context.username` et `context.password` dans cet exemple.



3. Renseignez le champ **Table Name** en saisissant `employees`, le nom de la table contenant les informations des employés, dans les deux bases de données, dans cet exemple.
4. Renseignez le **Schema**. Si vous avez stocké le schéma dans le **Repository**, vous pouvez le récupérer en sélectionnant **Repository** ainsi que l'entrée correspondante dans la liste.

   Dans cet exemple, le schéma des deux tables des bases de données à lire se compose de six colonnes : `id` (INT, de longueur 2), `name` (VARCHAR, de longueur 20), `email` (VARCHAR, de longueur 25), `sex` (VARCHAR, de longueur 1), `department` (VARCHAR, de longueur 10) et `position` (VARCHAR, de longueur 10).



5. Cliquez sur **Guess Query** afin de récupérer toutes les colonnes de la table, qui seront affichées dans l'onglet **Run**, via le composant **tLogRow**.
6. Dans la vue **Basic settings** du composant **tLogRow**, sélectionnez l'option **Table** pour afficher les enregistrements sous forme de tableau.

Configurer la fonctionnalité de chargement implicite des contextes

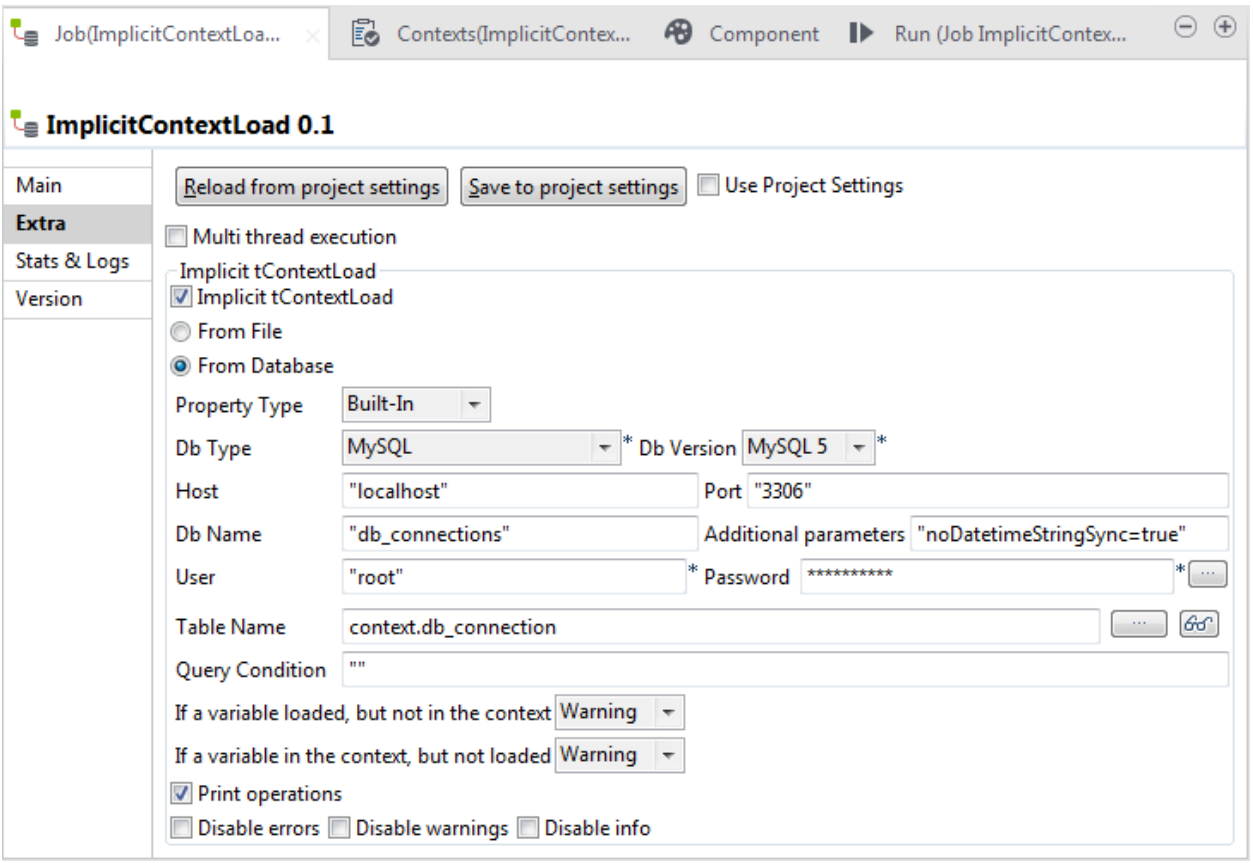
Vous pouvez configurer la fonctionnalité de chargement implicite des contextes, soit dans les paramètres du projet, pour pouvoir l'utiliser dans tous les Jobs du projet, soit dans l'onglet **Job** pour un Job spécifique.

L'exemple suivant explique comment configurer cette fonctionnalité dans la vue **Job** pour un Job en particulier. Si vous souhaitez configurer la fonctionnalité pour la réutiliser dans différents Jobs, sélectionnez **File** > **Edit Project properties** dans le menu pour ouvrir la boîte de dialogue **Project Settings**. Dans **Job Settings** > **Implicit context load**, cochez la case **Implicit tContextLoad** et configurez les paramètres en suivant les étapes 2 à 6, ci-dessous. Dans la vue **Job**, cochez la case **Use Project Settings** pour appliquer les paramètres au Job.

Procédure

1. Dans la vue **Job**, cliquez sur l'onglet **Extra** et cochez la case **Implicit tContextLoad** pour activer le chargement explicite des contextes à l'aide du composant **tContextLoad** dans le Job.
2. Sélectionnez la source de laquelle charger les paramètres de contexte. Une source de contextes peut être un fichier plat à deux colonnes ou une table de base de données contenant deux colonnes. Dans ce scénario, les informations de connexion à la base de données sont stockées dans des tables de bases de données. Sélectionnez donc l'option **From Database**.
3. Configurez les détails de la connexion à la base de données comme vous configurez les paramètres simples d'un composant d'entrée de base de données.

Dans cet exemple, tous les paramètres de connexion sont utilisés pour ce Job en particulier, sélectionnez **Built-In** dans la liste **Property Type** et renseignez manuellement les informations de connexion.



4. Renseignez le champ **Table Name** avec la variable de contexte nommée `db_connection` définie dans la vue **Contexts** du Job, afin de pouvoir choisir la table de base de données de laquelle charger dynamiquement les paramètres de contexte lors de l'exécution du Job.
5. Vous allez récupérer sans condition tous les détails de la connexion depuis les tables de la base de données, laissez donc le champ **Query Condition** vide.
6. Cochez la case **Print operations** afin de lister les paramètres de contexte chargés lors de l'exécution du Job.

Exécuter le Job

Procédure

1. Appuyez sur les touches **Ctrl+S** afin de sauvegarder le Job et appuyez sur **F6** pour exécuter le Job.
2. Une boîte de dialogue s'ouvre et vous demande de sélectionner une base de données. Sélectionnez une base de données et cliquez sur **OK**.

   Les paramètres de contexte chargés ainsi que le contenu de la table "employees" de la base de données sélectionnée s'affichent dans la console **Run**.
3. Appuyez sur la touche **F6** pour exécuter le Job à nouveau et sélectionnez l'autre base de données lorsque cela vous est proposé.

   Les paramètres de contexte chargés, ainsi que le contenu de la table "employees" de l'autre base de données, sont affichés dans la console de la vue **Run**.

Utilisation de la fonctionnalité Exécution en multi thread pour exécuter des Jobs en parallèle

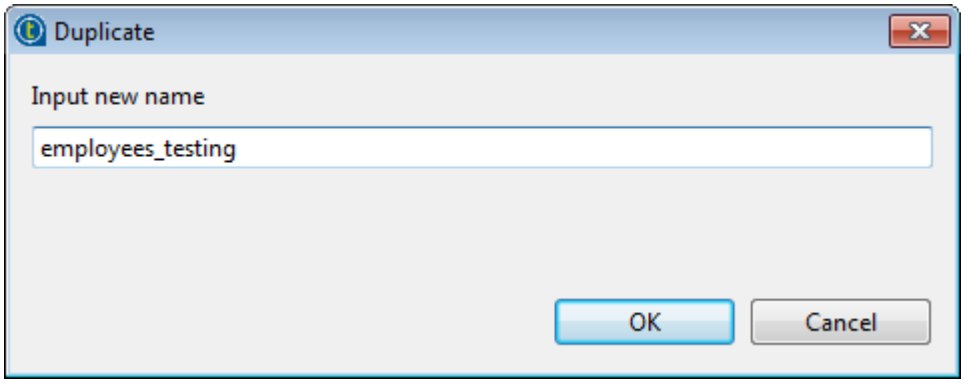
Basé sur le cas d'utilisation précédent Utilisation de la fonctionnalité Use Output Stream à la page 14, ce scénario donne un exemple d'utilisation de la fonctionnalité d'exécution en multi thread pour exécuter deux Jobs en parallèle afin d'afficher des informations des employés dans l'environnemen t de test en même temps que dans l'environnement de production. Lorsqu'il faut gérer de grandes quantités de données, cette fonctionnalité peut considérablement optimiser les performances d'exécution du Studio Talend.

Pour plus d'informations sur la fonctionnalité d'exécution en multi thread, consultez la section *Exécuter plusieurs sous-jobs en parallèle* du Guide utilisateur du Studio Talend à l'adresse https://help.talend.com.

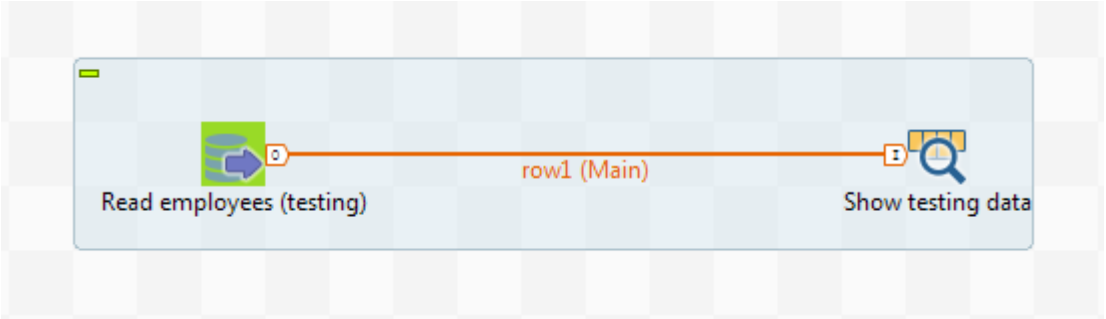
Préparer les Jobs pour lire des données des employés dans différents contextes

Procédure

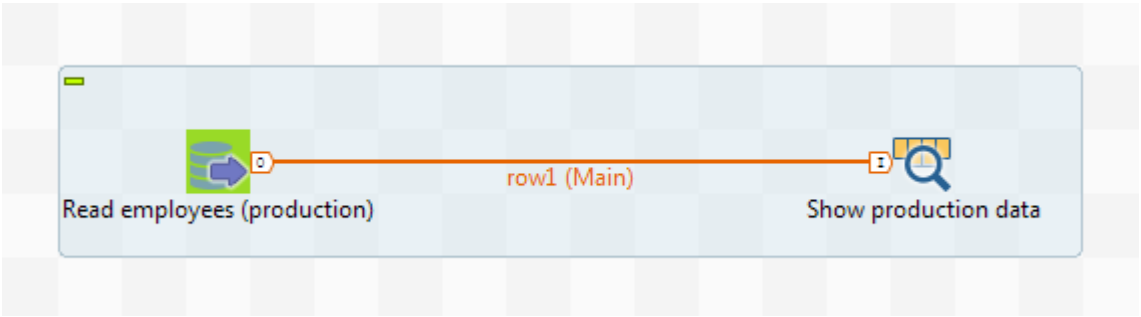
1. Dans la vue **Repository**, cliquez-droit sur le Job créé dans le cas d'utilisation Utilisation de la fonctionnalité Use Output Stream à la page 14 et sélectionnez **Duplicate** dans le menu contextuel. Puis, dans la boîte de dialogue **[Duplicate]**, saisissez un nouveau nom pour le Job, `employees_testing` dans cet exemple, puis cliquez sur **OK**.



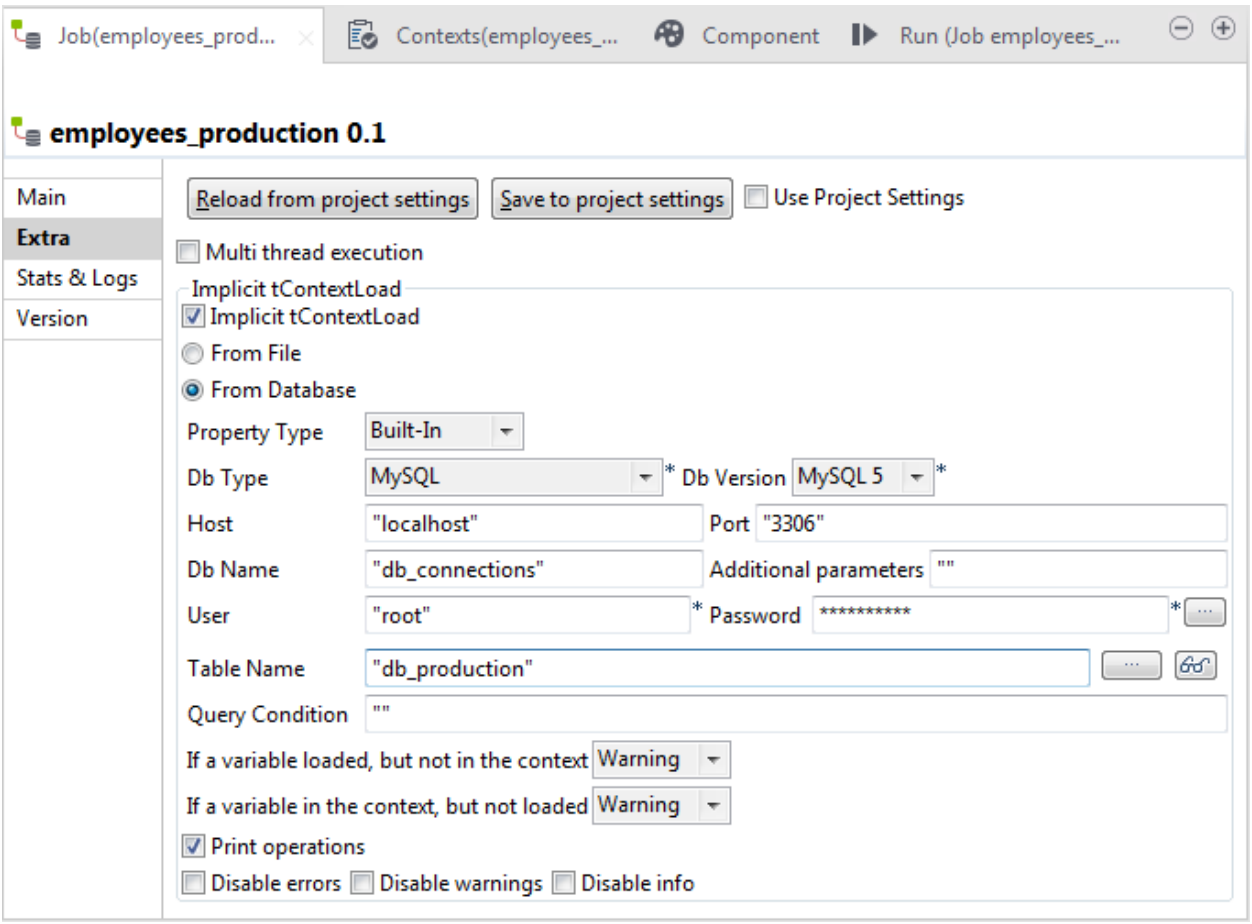
2. Ouvrez le nouveau Job et renommez les composants pour mieux identifier le rôle de chacun.



3. Créez un autre Job nommé `employees_production` en répétant les étapes ci-dessus.



4. Dans la vue **Contexts** des deux Jobs, supprimez la variable **db_connection**.
5. Dans l'onglet **Extra** de la vue **Job** du Job **employees_testing**, renseignez le champ **Table Name** en saisissant `db_testing`. Dans l'onglet **Extra** de la vue **Job** du Job **employees_production**, renseignez le champ **Table Name** en saisissant `db_production`.

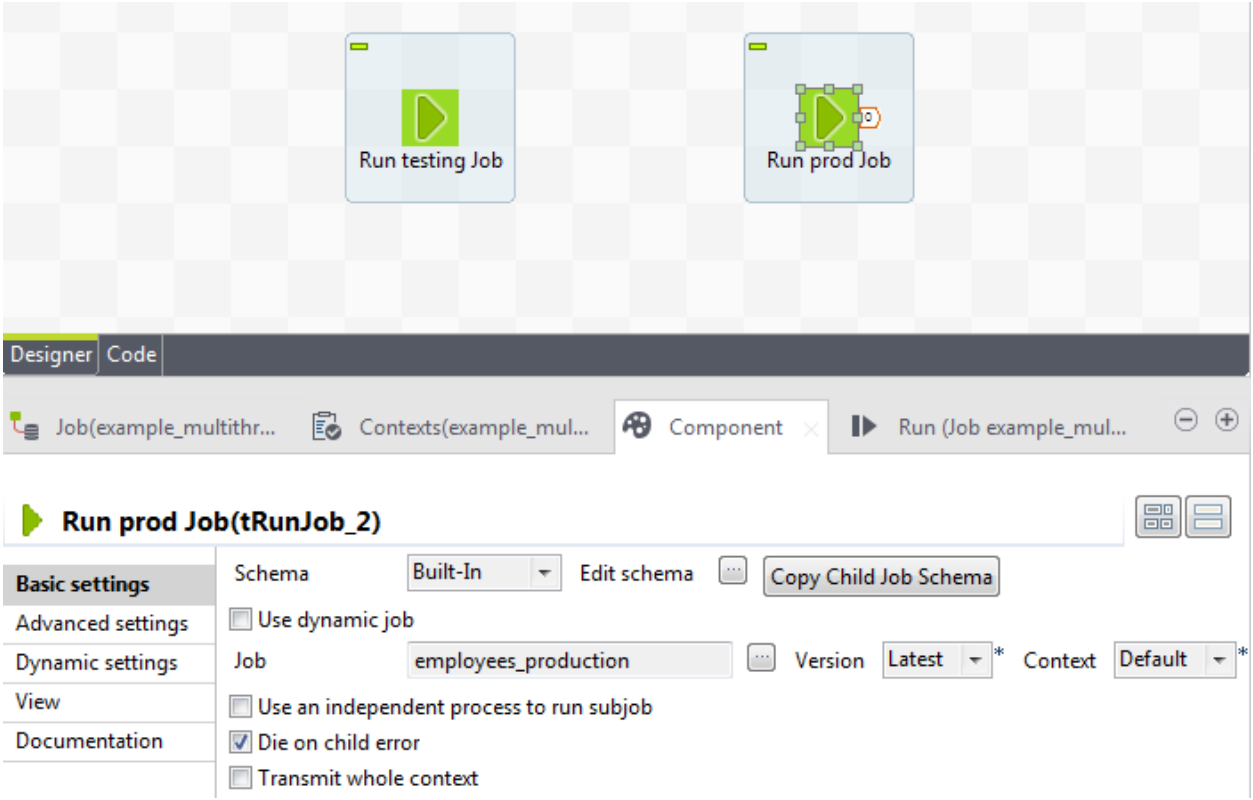


Mettre en place un Job parent pour exécuter les Jobs en parallèle

Procédure

1. Créez un nouveau Job, ajoutez deux composants **tRunJob** dans l'espace de modélisation graphique et renommez ces composants pour mieux identifier le rôle de chacun.

2. Dans la vue **Component** du premier composant **tRunJob**, cliquez sur le bouton **[...]** à côté du champ **Job** et spécifiez le Job à exécuter, `employees_testing` dans cet exemple.
3. Configurez l'autre composant **tRunJob** pour exécuter l'autre Job, `employees_production`.



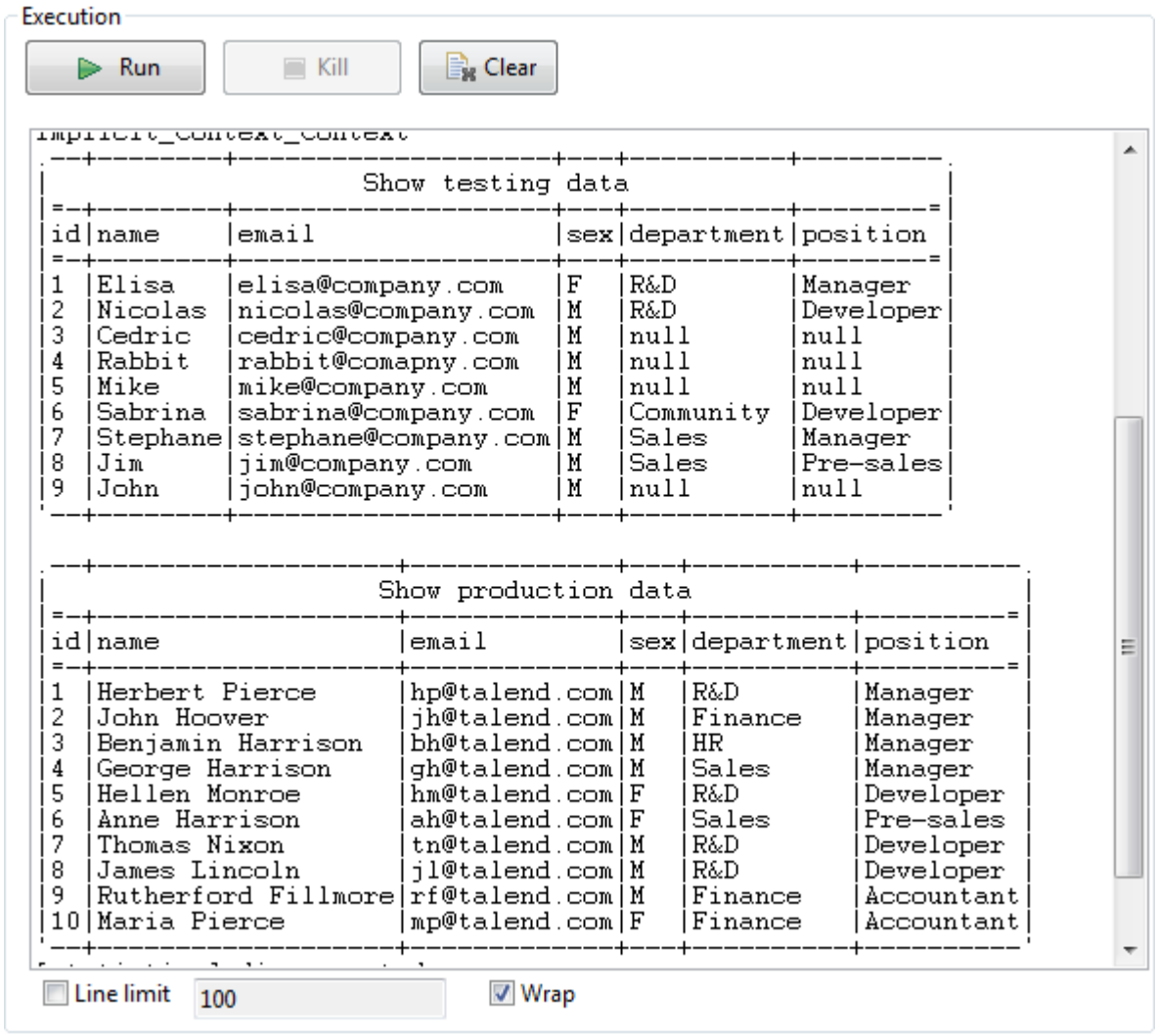
4. Dans l'onglet **Extra** de la vue **Job**, cochez la case **Multi thread execution** pour activer l'exécution en multi thread.

Exécuter les Jobs

Procédure

1. Sauvegardez chaque Job en appuyant sur les touches **Ctrl+S**.
2. Dans le Job parent, appuyez sur **F6** ou cliquez sur **Run** dans la vue **Run** pour démarrer l'exécution des Jobs enfants.

   Les Jobs enfants sont exécutés en parallèle, lisent les données des employés à partir des deux bases de données et affichent ces données dans la console.


Execution
Run
Kill
Clear
Show testing data
id|name|email|sex|department|position
1|Elisa|elisa@company.com|F|R&D|Manager
2|Nicolas|nicolas@company.com|M|R&D|Developer
3|Cedric|cedric@company.com|M|null|null
4|Rabbit|rabbit@comapny.com|M|null|null
5|Mike|mike@company.com|M|null|null
6|Sabrina|sabrina@company.com|F|Community|Developer
7|Stephane|stephane@company.com|M|Sales|Manager
8|Jim|jim@company.com|M|Sales|Pre-sales
9|John|john@company.com|M|null|null
Show production data
id|name|email|sex|department|position
1|Herbert Pierce|hp@talend.com|M|R&D|Manager
2|John Hoover|jh@talend.com|M|Finance|Manager
3|Benjamin Harrison|bh@talend.com|M|HR|Manager
4|George Harrison|gh@talend.com|M|Sales|Manager
5|Hellen Monroe|hm@talend.com|F|R&D|Developer
6|Anne Harrison|ah@talend.com|F|Sales|Pre-sales
7|Thomas Nixon|tn@talend.com|M|R&D|Developer
8|James Lincoln|jl@talend.com|M|R&D|Developer
9|Rutherford Fillmore|rf@talend.com|M|Finance|Accountant
10|Maria Pierce|mp@talend.com|F|Finance|Accountant
Line limit
100
Wrap