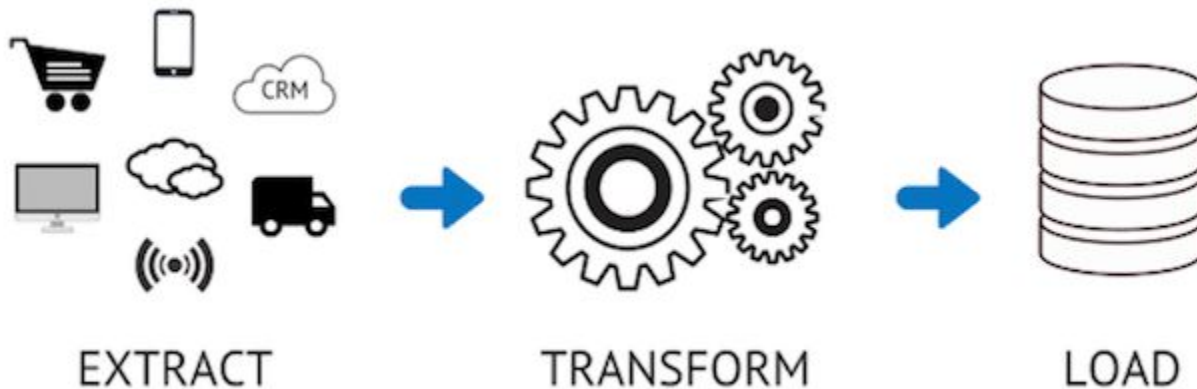


ETL con Talend

Introduzione ad ETL e Talend Open Studio DI

Cos'è ETL

ETL è l'acronimo per **Extract**, **Trasform** and **Load** e indica un processo di estrazione, trasformazione e caricamento dei dati in un sistema di sintesi.



I dati possono essere estratti da database relazionali, comuni file di testo o da altri sistemi informatici.

Durante la trasformazione dei dati avviene anche la fase di **Data Quality**, più complessa se si ha la necessità di estrarre dati da database disordinati e poco omogenei. Le tecniche di Data Quality permettono di pulire i dati disomogenei, eliminare duplicati e derivare informazioni calcolate; tutto ciò ha lo scopo di rendere conformi dati provenienti da sorgenti diverse e di renderli il più aderente possibile alla logica di business del sistema di analisi per cui vengono estratti.

In quali ambiti vengono utilizzati i processi ETL

- Migrazione dei dati da un'applicazione a un'altra;
- Replica dei dati per l'esecuzione di backup o analisi della ridondanza;
- Processi operativi quali il trasferimento di dati da un sistema CRM in un ODS (Operational Data Store) per l'ottimizzazione e l'arricchimento, per poi restituire i dati al sistema CRM;
- Inserimento dei dati in un Data Warehouse per l'assimilazione, l'ordinamento e la trasformazione in business intelligence;
- Migrazione delle applicazioni locali in infrastrutture Cloud, Cloud ibride o multi-Cloud;
- Sincronizzazione di sistemi.

L'obiettivo di un processo ETL è ottenere dati puliti e accessibili da utilizzare per attività di analisi o processi di business.

I dati grezzi vengono inizialmente estratti da una vasta gamma di origini. Successivamente abbiamo la fase più critica del processo ETL e cioè la fase di trasformazione. Durante la trasformazione, i dati grezzi vengono opportunamente “convertiti” nei formati richiesti. Se i dati non sono puliti, applicare le regole aziendali di segnalazione diventa complicato. L'ultima fase del processo ETL prevede, in genere, il caricamento dei dati estratti e trasformati in una nuova destinazione. I dati possono essere caricati in un data warehouse in due diversi modi: tramite caricamento completo o incrementale.

Talend Open Studio for Data Integration

Talend è una software-house statunitense che fornisce servizi e software di integrazione, gestione di dati e big data. Per la realizzazione di software ETL viene fornito il framework Talend Studio for Data Integration, un software basato su linguaggio Java, con lo scopo di fornire un'interfaccia grafica e un insieme di connettori utili allo sviluppo di flussi ETL.

Attraverso una semplice GUI, il programma produce automaticamente scripts in Java per il caricamento e la trasformazione dei dati. È possibile scegliere tra più di 900 componenti, connettersi a 40 database diversi, oltre a fogli Excel, file CSV, JSON, XML e tanti altri. Talend Data Integration permette di programmare i flussi ETL in modo visuale, tramite operazioni di drag&drop sugli elementi dalla palette alla griglia rappresentante il corpo del nostro programma, inoltre, permette la divisione di grandi flussi in unità più piccole, in modo da rendere il lavoro più semplice e scalabile.

Scaricare TOS DI

Talend Open Studio for Data Integration può essere scaricato gratuitamente dal sito ufficiale nella versione “installabile” e “portable”.

La versione installabile è disponibile per i SO Windows e MacOS al seguente indirizzo:

<https://www.talend.com/it/products/data-integration/data-integration-open-studio/>



La versione portable (TOS_DI-20200219_1130-V7.3.1.zip), invece, la potete trovare al seguente indirizzo:

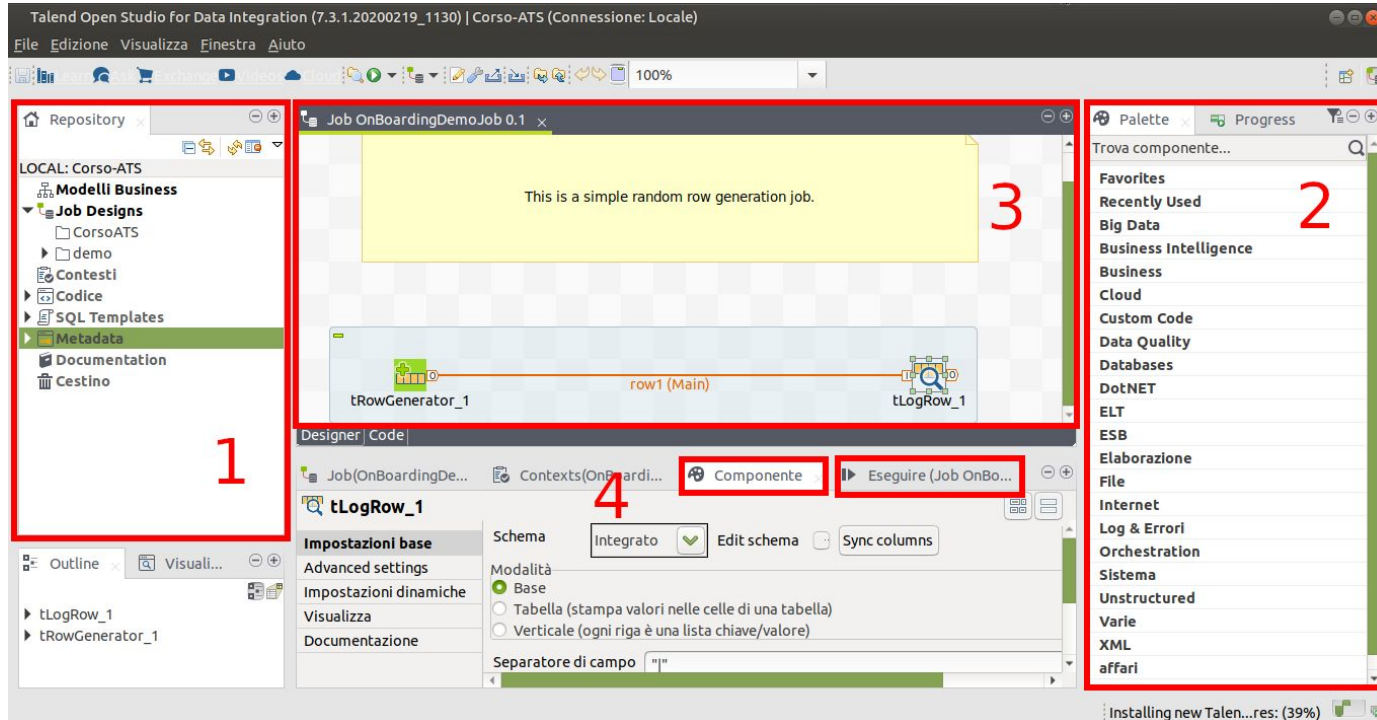
<https://www.talend.com/it/products/data-integration-manuals-release-notes/>

TOS_DI-20200219_1130-V7.3.1-osx-installer.dmg	7.3.1	26 febbraio 2020	Main	MAC	852MB	US and Europe
TOS_DI-20200219_1130-V7.3.1.zip	7.3.1	26 febbraio 2020	Main	Unix_Linux Windows MAC	904MB	US and Europe
TOS_DI-Win32-20200219_1130-V7.3.1.exe	7.3.1	26 febbraio 2020	Main	Windows	750MB	US and Europe

Vi consiglio l'utilizzo dell'ultima versione stabile contrassegnata dal valore **Main** nella colonna **Tipo release**.

Per il download basta selezionare il link che trovate nell'ultima colonna presente nella tabella mostrata dall'immagine precedente (US and Europe).

L'interfaccia di TSO DI



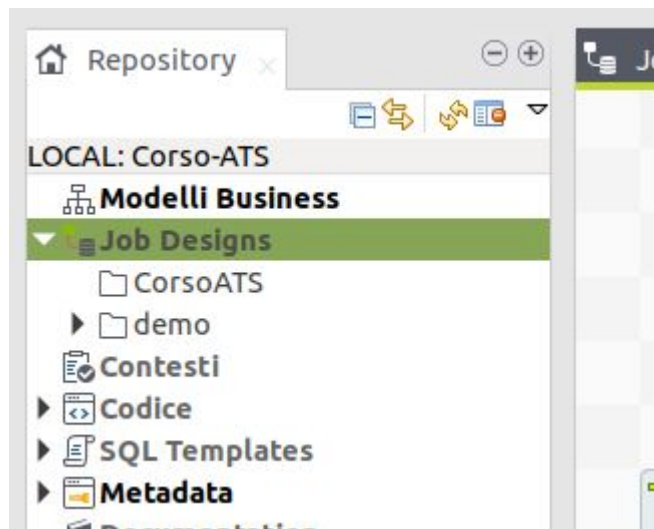
Talend Studio consente di creare programmi ETL mediante un'interfaccia utente grafica. Tali programmi sono chiamati job DI (Data Integration) o di integrazione dei dati.

L'interfaccia utente di Talend Studio include diversi pannelli chiamati viste

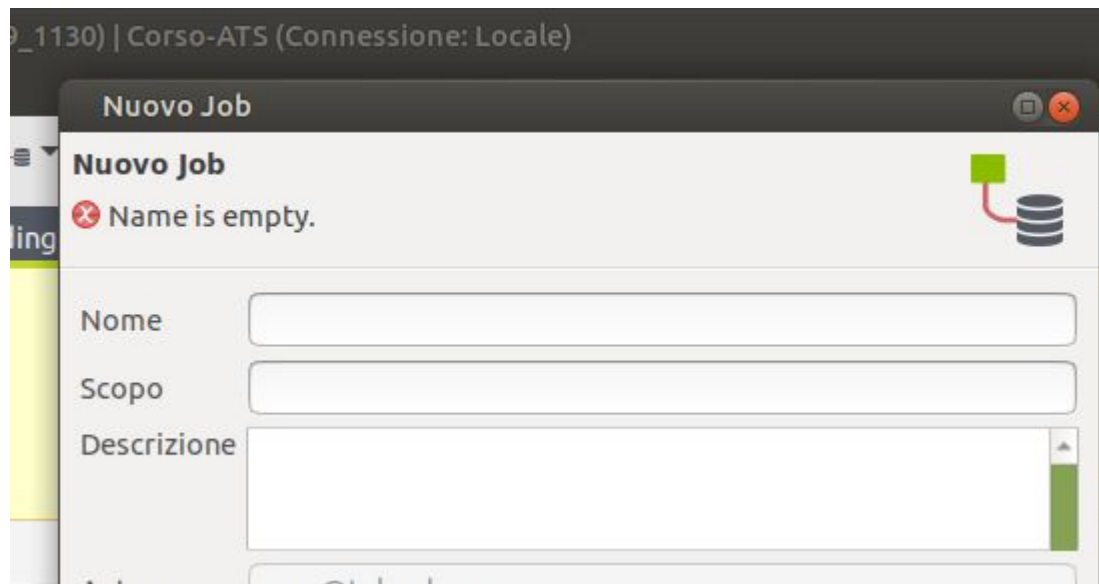
1. Nella vista Project Repository (Repository progetti) sono elencate tutte le voci di progetto, come job (programmi ETL Java), servizi, codice, metadati e documentazione dei progetti;
2. Nell'area Palette (Tavolozza) sono elencati tutti i componenti disponibili, organizzati in cartelle;
3. La vista Job Designer (Progettazione job) è la vista principale di Talend Studio, dove i vari componenti vengono utilizzati per creare job ETL;
4. Nell'area Component (Componente) sono visualizzati tutti i parametri necessari per configurare un componente. Le informazioni visualizzate in quest'area dipendono dalle selezioni effettuate in Job Designer (Progettazione job). Dalla vista Run (Esegui) è possibile attivare l'esecuzione di un job Talend e visualizzare

Creazione del primo job

Project Repository (Repository progetti), fai clic con il pulsante destro del mouse su **Job Designs** (Progetti job).



Per aprire la procedura guidata di creazione di un nuovo job, fai clic su **Create Job** (Crea job standard).



The screenshot shows a software window titled "Nuovo Job" (New Job) with a dark header bar. Below the header, the title "Nuovo Job" is repeated. A red error icon and the message "Name is empty." are displayed. To the right of the error message is a small icon of a database cylinder with a red line connecting it to a green square. Below the error message are three input fields: "Nome" (Name), "Scopo" (Purpose), and "Descrizione" (Description). The "Nome" field is currently empty. At the bottom of the window, a partially visible email address "xxx@tld.com" is shown.

Nel campo Name (Nome) della procedura guidata, inserisci il nome del job, in questo caso **testJob**.

Nel campo Purpose (Scopo), inserisci *Visualizzare un messaggio*.

Nel campo Description (Descrizione), digita *Questo tutorial impiega un componente per visualizzare una casella di testo con un messaggio personalizzato*.

Per chiudere la procedura guidata e creare il job, fai clic su **Finish** (Fine).

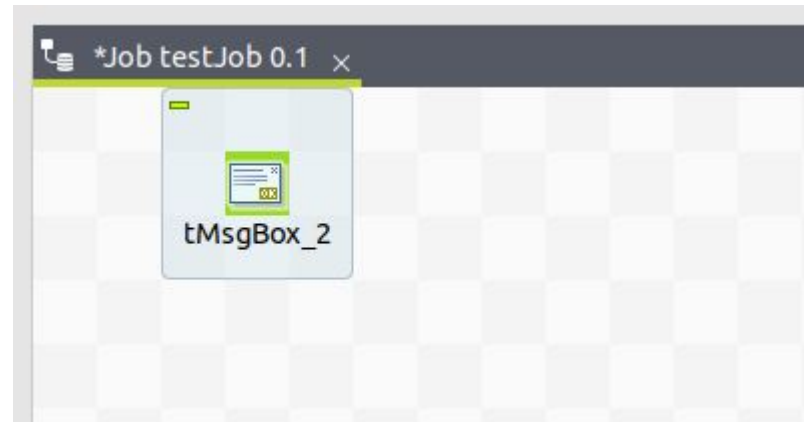
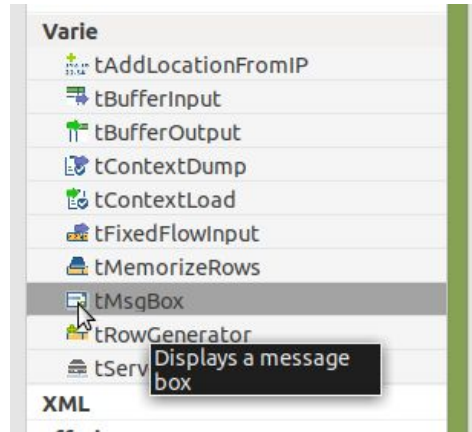
Viene aperto un nuovo job vuoto in **Job Designer** (Progettazione job).

I job DI impiegano componenti. Talend Studio offre una libreria completa di oltre 800 componenti per l'integrazione dei dati.

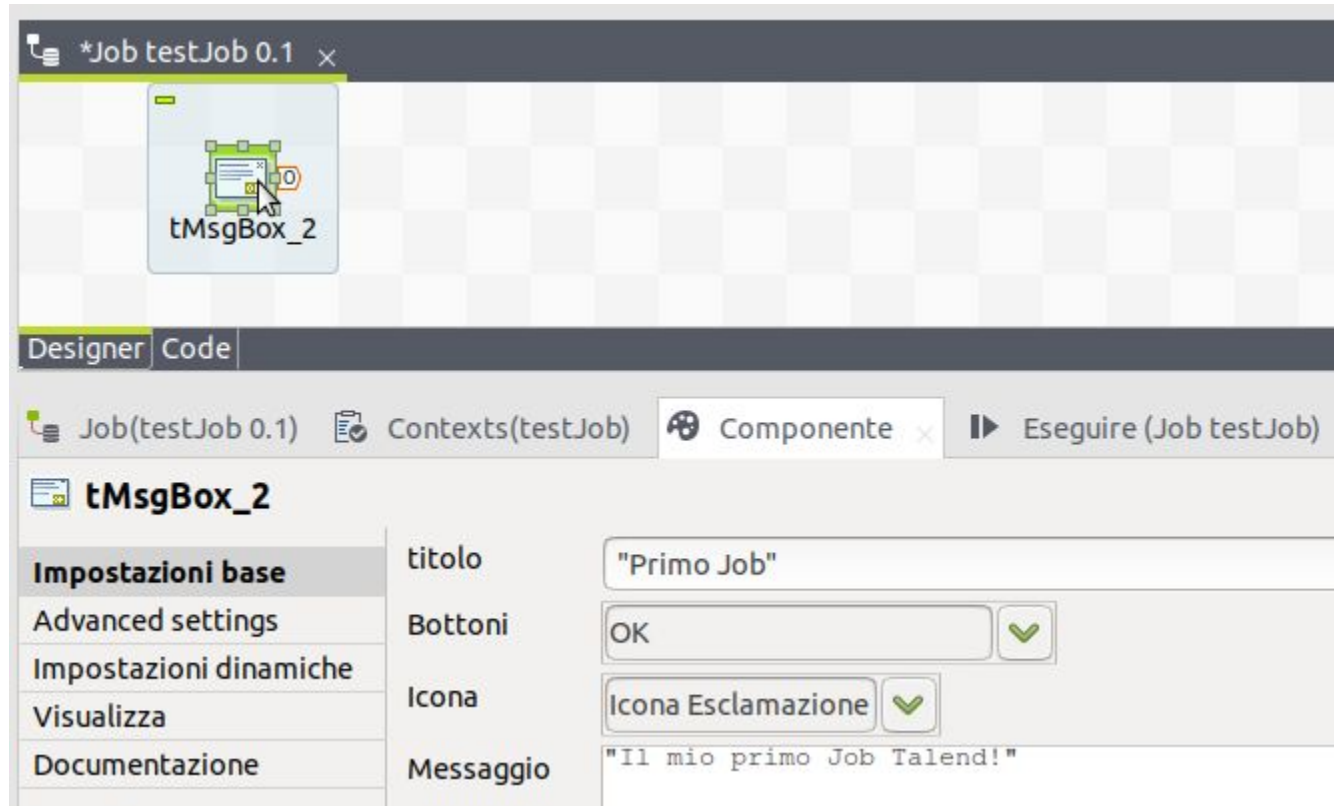


A questo punto possiamo provare ad aggiungere un componente al nostro Job. Come anticipato in TSO la creazione di un Job avviene tramite l'interfaccia visuale, è possibile aggiungere i nostri componenti all'interno del Job con una semplice operazione di drag and drop dall'area palette alla vista Job Designer.

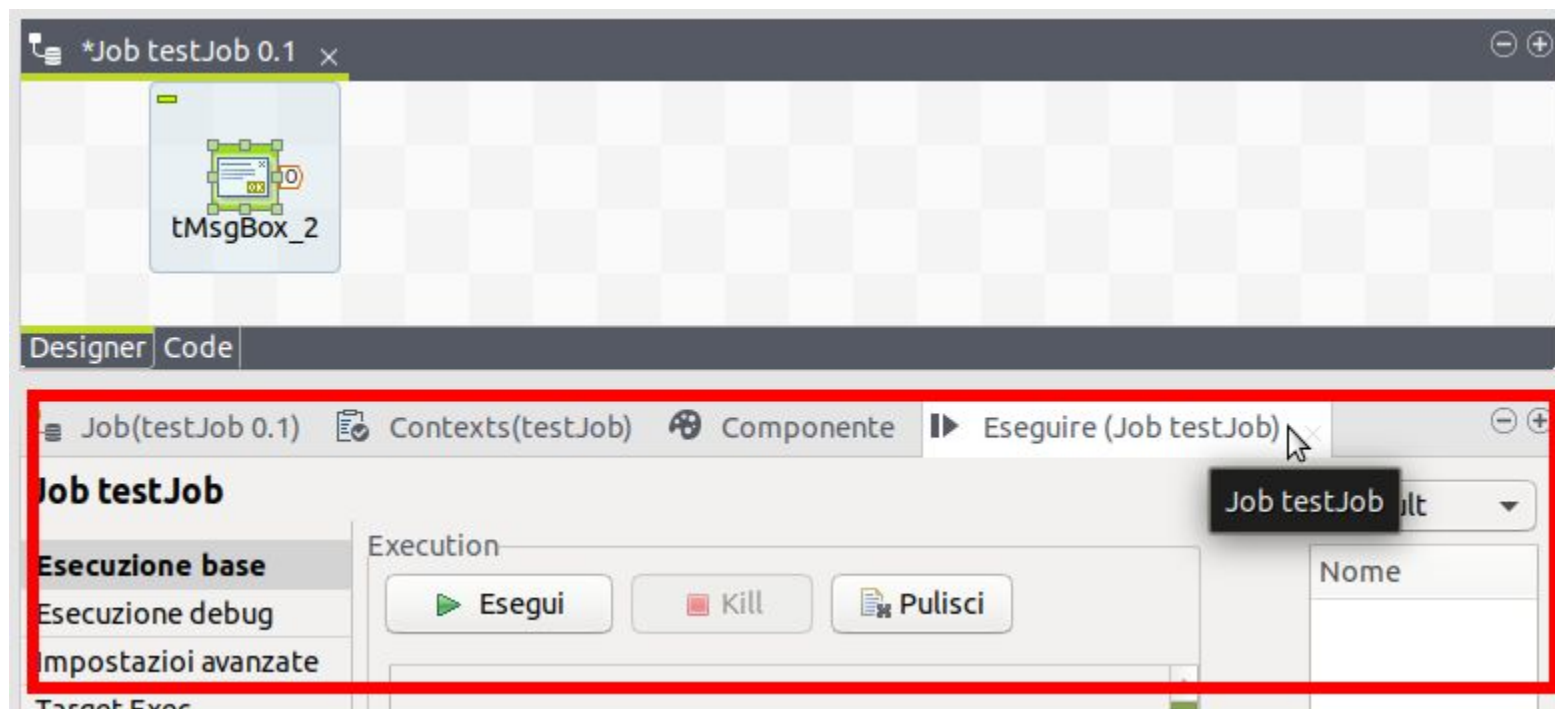
Aggiungi un componente **tMsgBox** al nostro Job. Per farlo lo seleziono in Palette, sotto la voce Varie e trascinalo in Job Designer (Progettazione job). Questo semplice componente, utile per i test, consente di visualizzare una casella di messaggio.



A questo punto puoi configurare il componente appena inserito. Seleziona il componente **tMsgBox_1** dalla view Job Designer e, dopo aver selezionato la view **Componente**, modifica le informazioni di base del componente.



Sei pronto per eseguire il tuo Job! Ora non devi fare altro che selezionare la vista “Esegui” e selezionare il bottone “Esegui”.



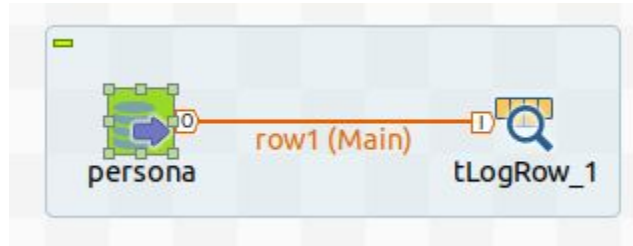
Dopo aver effettuato la compilazione del progetto Talend mostrerà il risultato del Job appena eseguito.



Un job è formato da un insieme di componenti che si scambiano informazioni tramite un flusso rappresentato da frecce direzionali.

Come già accennato, per importare un componente all'interno del progetto, è sufficiente eseguire un'operazione di Drag and Drop dalla palette alla griglia principale.

Per unire due componenti, è sufficiente cliccare col tasto destro sull'elemento sorgente, e dopo aver selezionato il tipo di riga, cliccare sull'elemento di destinazione



Le righe disponibili per collegare due connettori sono: **Principale, Su subjob OK / Su errore subjob, Su componente OK / Su errore componente e Esegui se**

Principale

Permette il transito di dati da un componente all'altro. Questo tipo di riga richiede che gli elementi di sorgente e destinazione abbiano tra le proprietà uno «schema». Lo schema è un insieme di attributi che possono essere scambiati tra due elementi.

Ad esempio, nel caso di un connettore che esegue una query, lo schema sarà strutturato in modo da poter contenere il risultato della query. Sulla freccia Principale sarà definito uno schema.

Su subjob OK / Su errore subjob

Tutti i componenti uniti tramite una riga Principale fanno parte dello stesso **Subjob**. Il tipo di riga «**Su subjob OK**» viene attivata solo al completamento del **Subjob**. Quindi, il componente di destinazione di questa freccia sarà eseguito solo quando l'intero **subjob** di sorgente sarà completo. La riga **Su errore subjob** funziona in modo simile, ma si attiva al verificarsi di un errore nel subjob di sorgente.

Su componente OK / Su errore componente

Il componente di destinazione di questa freccia sarà eseguito solo dopo l'esecuzione del componente sorgente.

La riga Su errore componente funziona in modo simile, ma si attiva al verificarsi di un errore nel componente sorgente.

Esegui se

Questo componente permette di specificare una condizione. L'elemento di destinazione viene eseguito se, una volta eseguito il componente sorgente, la condizione è verificata.

L'inserimento della condizione non è opzionale!

L'unico connettore che richiede uno schema è quello principale.

N.B. Durante l'esecuzione del flusso, le informazioni vengono passate dai componenti di input ad output una riga alla volta. Questo significa, ad esempio, che un componente che esegue una query invia il risultato al componente di output una riga alla volta.

L'iterazione sul risultato viene eseguita automaticamente!

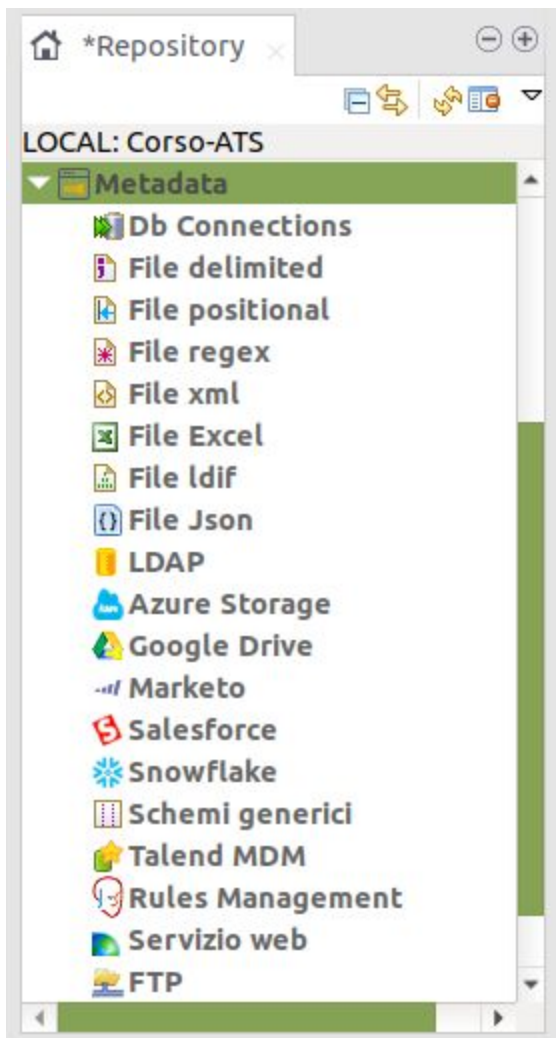
Metadati

I metadati permettono di gestire le connessioni verso i sistemi esterni.

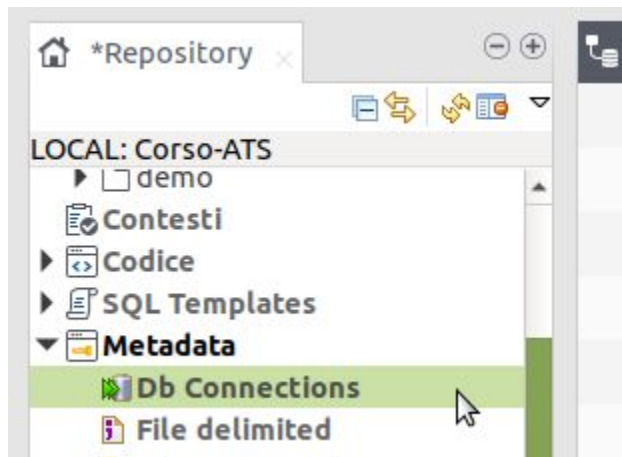
Una volta stabilita una connessione, è possibile parametrizzarla, modificarla, testarla e usufruire di tutte le funzionalità offerte da essa.

I metadati sono associati all'intero progetto e non al flusso, quindi possono essere dichiarati una sola volta, e utilizzati in tutti i job del processo. Le eventuali modifiche si rifletteranno automaticamente su Job/componenti interessati.

Talend può stabilire connessioni verso tutti i sistemi indicati all'interno del menù **metadata** presente nella view Repository.



Creiamo una connessione verso un database. All'interno della view **Repository** fare click con il pulsante destro del mouse su **Db Connections**.



e, dal menu contestuale che si apre, selezioniamo **crea connessione**.

Inseriamo le informazioni di base sulla connessione (nome, scopo e descrizione) e selezioniamo il bottone **Next**.

Nel passo 2/2 selezionare il tipo di database (nel nostro caso MySQL). La prima volta che selezioniamo un tipo di database ci verrà chiesto di scaricare i driver.

Connessione Database

Nuova Connessione al Database - Passo 2/2

Definire parametri di connessione



Tipo DB MySQL

Versione Db

MySQL 8

Stringa di connessione

jdbc:mysql://accademy.chdaxtz2ghx5.eu-central-1.rds.amazonaws.com:3306/acca

Login

accademy_user

Password

.....

Server

accademy.chdaxtz2ghx5.eu-central-1.rds.amazonaws.com

Porta

3306

DataBase

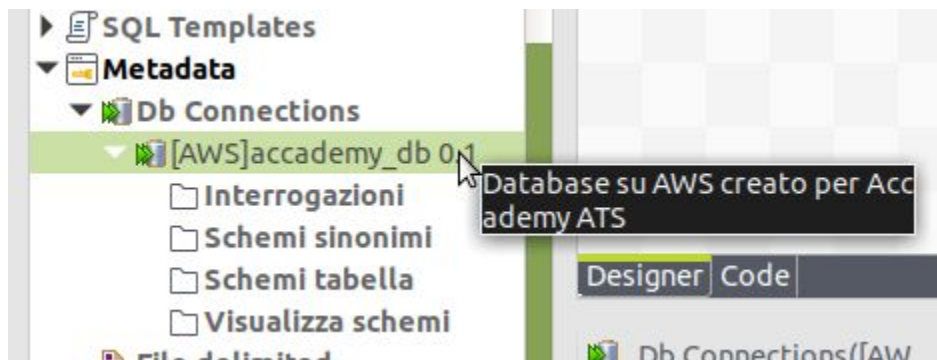
accademy_db

Test connection

v

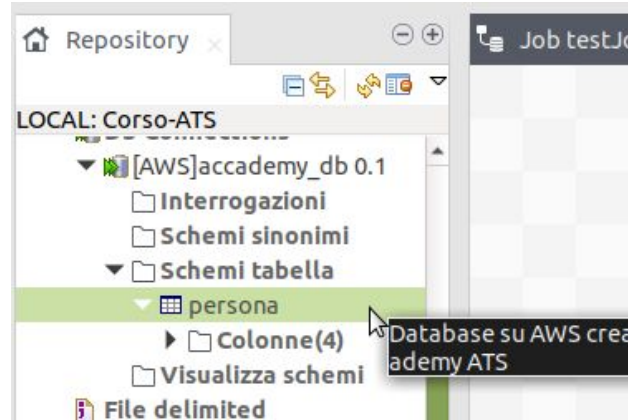
Una volta inserite tutte le informazioni proviamo ad effettuare un test della connessione con il bottone **Test Connection**. Se la connessione va a buon fine possiamo terminare la procedura di creazione con il bottone **Finish**

Una volta terminata la procedura dovremmo ritrovarci all'interno della sezione **metadata**



Per recuperare automaticamente tutti gli schemi delle tabelle, facciamo clic con il pulsante destro del mouse sui metadati **[AWS]accademy_db 0.1** nel Project Repository quindi selezioniamo **Recupera schema** che ci consentirà di recuperare la

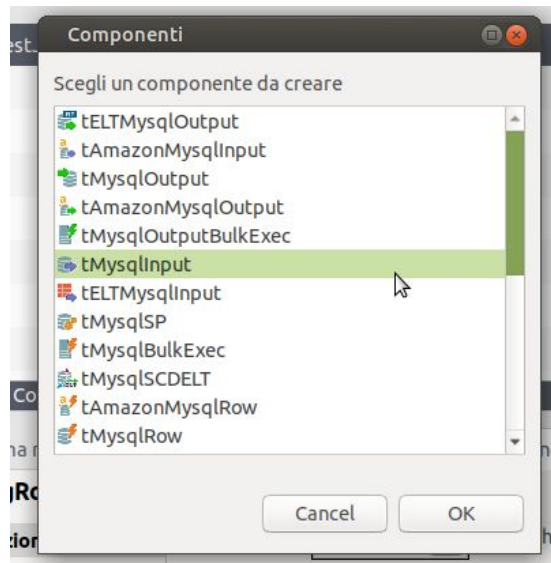
Per leggere i dati di una tabella nell'elenco, seleziona la tabella e trascinala in Job Designer.



Nella finestra Components (Componenti), fai clic su **tMySQLInput**, quindi su **OK**.

Viene creato un componente **tMySQLInput** con le informazioni del repository. Viene utilizzata la connessione **[AWS]accademy_db 0.1** e per lo schema vengono utilizzate le informazioni del repository acquisite dalla tabella dei metadati **persona**.

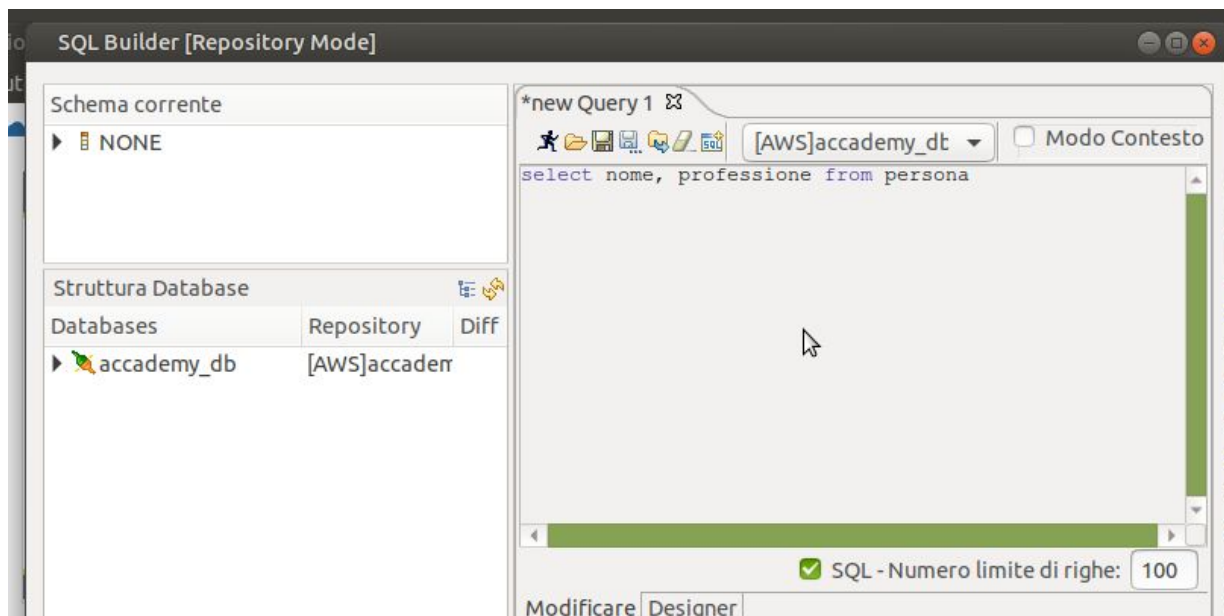
Inoltre, Talend genera la query SQL e la invia alla tabella **persona**.



Per visualizzare i dati della tabella, aggiungi il componente **tLogRow** e collega il componente **persona** al componente **tLogRow_1**.

Per eseguire il job, fai clic su **Esegui** nella vista **Eseguire**. I dati della tabella **persona** vengono visualizzati.

È possibile creare e salvare delle query custom, per interrogare il database cliccando con il tasto destro sulla connessione creata e selezionando la voce modifica query dal menu contestuale.



Le query create saranno presenti nella cartella «Interrogazioni», e potranno essere modificate o lanciate.

Creiamo un metadato verso un file Excel. Nella sezione metadata facciamo click con il tasto destro del mouse su **File Excel** e selezioniamo la voce **Crea file Excel**.

A questo punto selezioniamo un nome per il nostro metadata e andiamo avanti con **next**.

Nel passo successivo sarà necessario selezionare il file Excel dal nostro file system tramite il pulsante **naviga**.



A questo punto sarà necessario selezionare i fogli (sheets) del file Excel che vogliamo importare.

Nei passi successivi è possibile determinare i caratteri di escape, la presenza di una riga con le intestazioni di colonna e definire i tipi di dato presenti nelle varie colonne.

La lista con i tipi di dato viene autogenerata ma è possibile modificarla.

Nel modificare le impostazioni delle tipologie di dato noterete che, selezionando la checkbox **nullable** il tipo di dato viene automaticamente impostato con la classe Wrapper mentre definendo un campo che non ammette valori null verrà utilizzato il tipo primitivo.

Una volta creato il Metadata, è possibile modificarlo semplicemente cliccando su di esso col tasto destro e selezionando la voce «Modifica connessione». Così facendo si avrà la possibilità di modificare tutte le impostazioni del Metadata. Cliccando sul tasto destro, sarà disponibile anche la voce «Cancella», che elimina il metadata.

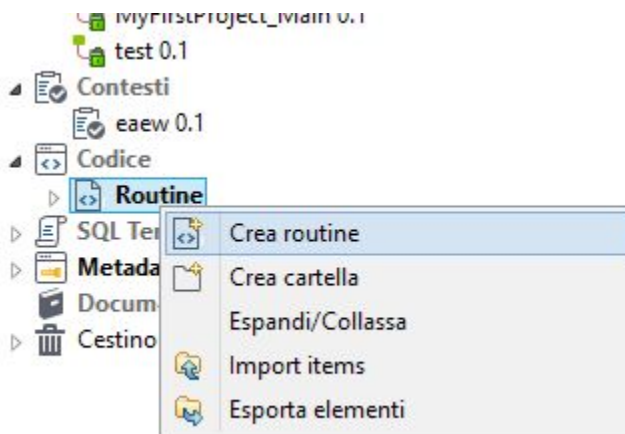
Esercitazioni

- Creare una connessione a un database MySQL, ed estrarne lo schema.
- Creare una connessione a un file CSV ed estrarne lo schema.

Routine

Talend permette di scrivere codice custom, in modo da poter risolvere anche problematiche non previste, o particolarmente complesse.

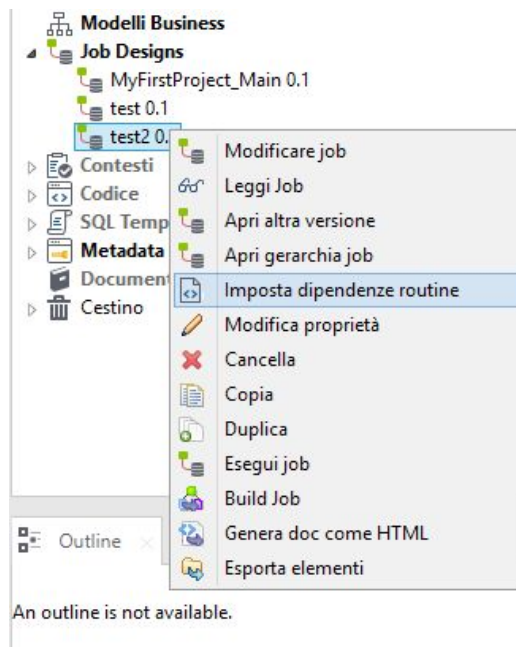
Per creare una classe, è sufficiente cliccare su «**Codice**», poi su «**Routine**» col tasto destro e selezionare la voce «**Crea routine**».



Il linguaggio di programmazione utilizzato per lo sviluppo di routine è il Java.

Le classi create godono quindi di tutte le caratteristiche relative a una classe Java.

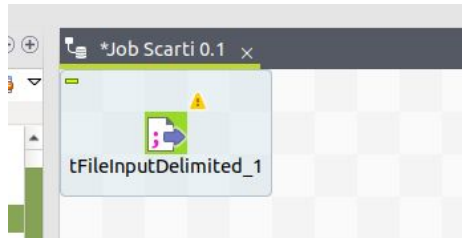
Per utilizzare una routine all'interno di un job è necessario dichiarare la dipendenza verso di essa



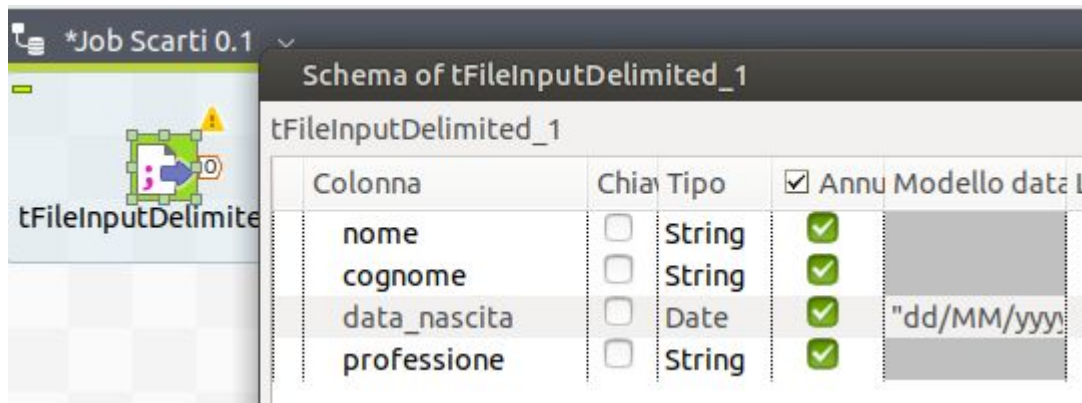
Flusso “Scarti”

Il flusso **Scarto (reject)** consente di gestire eventuali dati non conformi allo schema impostato per un determinato componente. In alcuni casi, a seconda dei requisiti applicativi, gli scarti non sono accettabili. In questi casi, i flussi di **Scarto** dovrebbero essere disabilitati ed il nostro job dovrebbe fallire.

Creiamo un nuovo Job, ed aggiungiamo un componente **tFileInputDelimited**. Impostiamo il componente appena inserito per caricare i dati presenti nel file **persone.csv** presente nel path **datasets/reject** del repository del corso e selezioniamo il checkbox “Interrompi se rilevato errore” nelle impostazioni di base del componente.

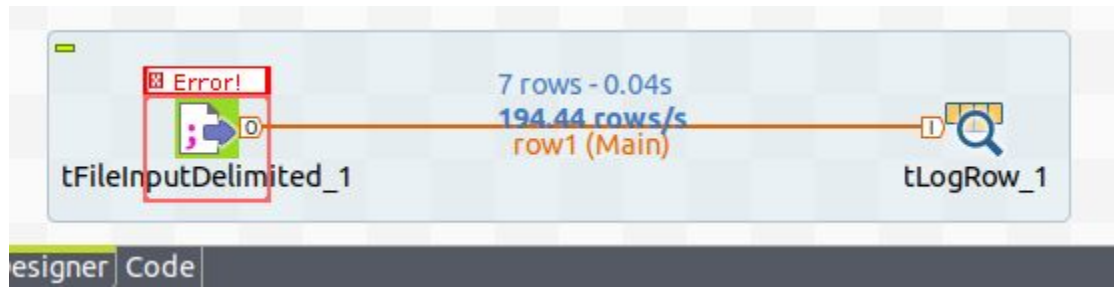


Aggiungiamo lo schema per la gestione delle informazioni presenti all'interno del file caricato



facendo attenzione ad impostare correttamente il tipo del campo data_nascita e fornendo il giusto pattern utilizzato per la parserizzazione.

A questo punto aggiungiamo un componente **tLogRow** per stampare le informazioni contenute nel file e proviamo a lanciare il nostro Job.



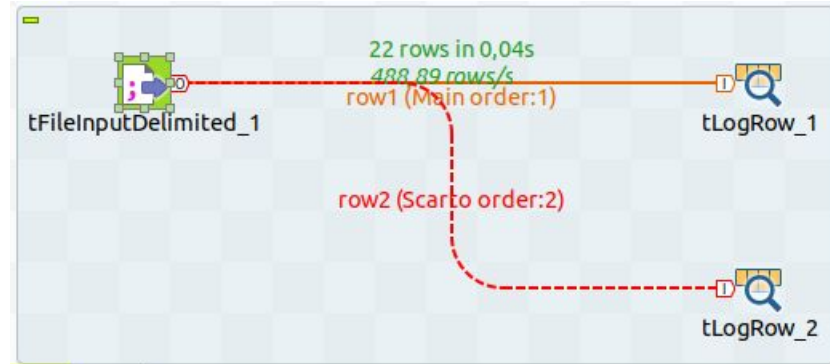
Come possiamo notare il nostro Job è andato in errore. Controllando i log, in effetti, possiamo notare che il nostro dataset contiene delle informazioni che non possono essere parserizzate correttamente nella colonna `data_nascita`.

Se volessimo far sì che il nostro Job completi la sua esecuzione anche in presenza di record che non possono essere validati sullo schema fornito possiamo deselezionare il controllo “Interrompi se rilevato un errore”.

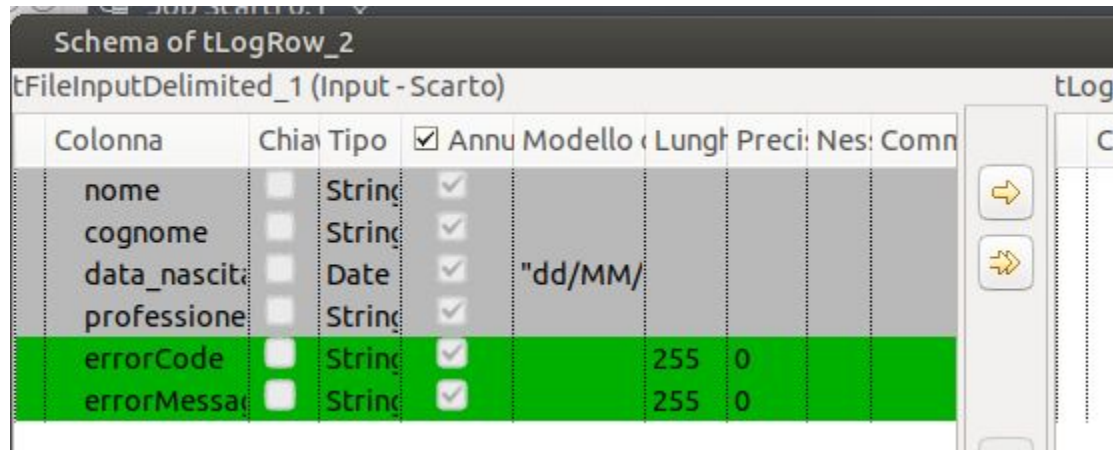
In effetti rilanciando il nostro job notiamo che questa volta non va in errore ed il nostro log mostra delle eccezioni relativamente ai record “scartati”.

In queste circostanze, se sono ammissibili gli scarti, potrebbe essere utile salvare in apposite strutture (file e/o database) i record che non sono stati processati correttamente al fine di poterne tenere traccia ed, eventualmente, apportare le opportune modifiche al dataset di input.

Aggiungiamo un nuovo componente **tLogRow** al nostro Job e colleghiamolo al componente che gestisce l'input tramite il flusso **scarto**. Per farlo selezionate il componente di input con il tasto **destro** -> **riga** -> **scarto** e selezionate nuovo componente **tLogRow**.



Eseguendo la nuova versione del nostro Job noterete che la nostra console non ha più errori ma i due componenti effettuano 2 stampe separate. Infatti i record conformi allo schema sono stati stampati dal componente **tRowMap** collegato mediante il flusso **principale** mentre i record che non sono stati parserizzati correttamente vengono passati al flusso **scarto** e stampate dal relativo componente. Inoltre potete notare che il componente legato al flusso di scarto ha delle colonne aggiuntive che non fanno parte dello schema principale.

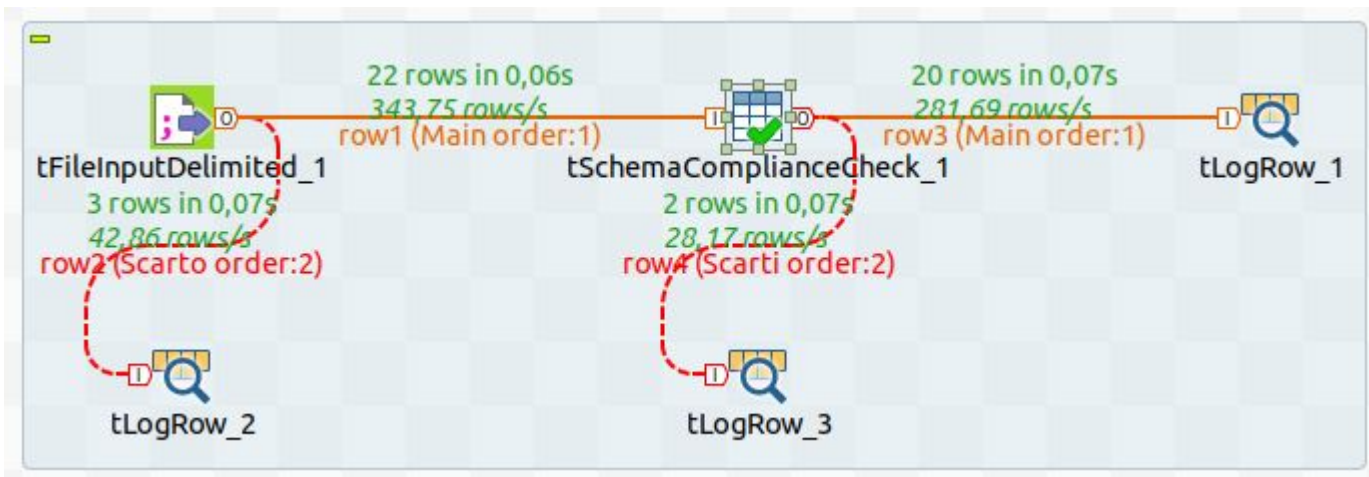


Schema of tLogRow_2

tFileInputDelimited_1 (Input - Scarto)

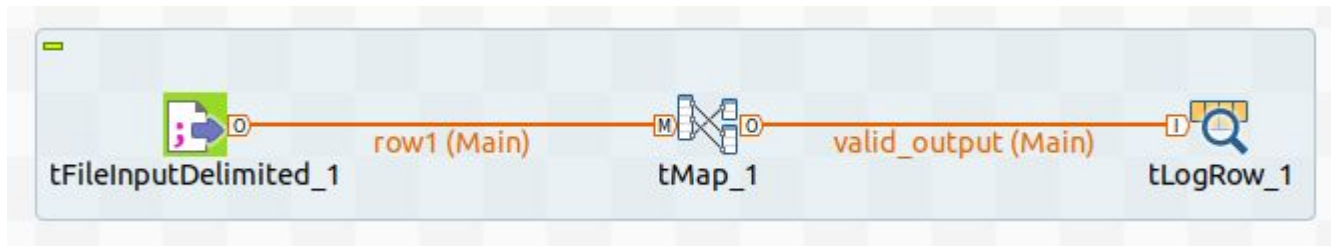
Colonna	Chia	Tipo	<input checked="" type="checkbox"/> Annu	Modello	Lung	Preci	Nes	Comm
nome	<input type="checkbox"/>	String	<input checked="" type="checkbox"/>					
cognome	<input type="checkbox"/>	String	<input checked="" type="checkbox"/>					
data_nascita	<input type="checkbox"/>	Date	<input checked="" type="checkbox"/>	"dd/MM/				
professione	<input type="checkbox"/>	String	<input checked="" type="checkbox"/>					
errorCode	<input type="checkbox"/>	String	<input checked="" type="checkbox"/>		255	0		
errorMessage	<input type="checkbox"/>	String	<input checked="" type="checkbox"/>		255	0		

Un altro componente utile alla validazione dei dati **tSchemaComplianceCheck**.

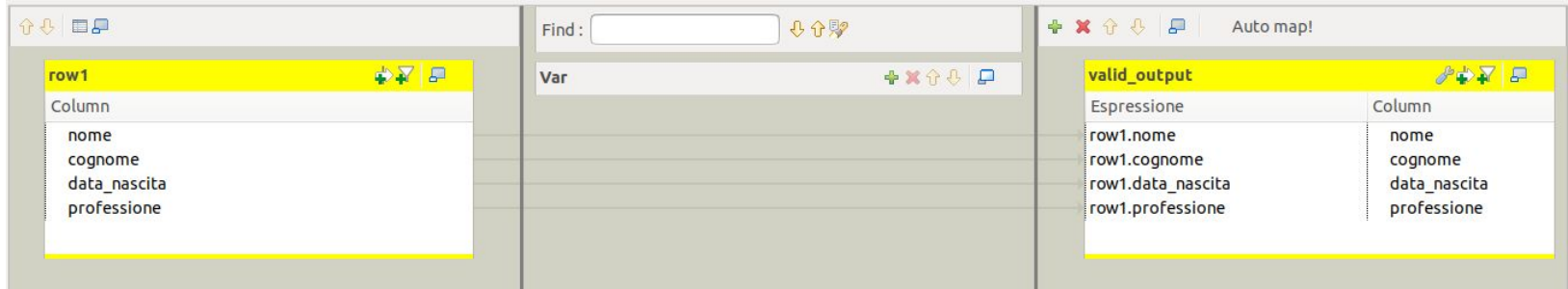


Anche un componente **tMap** può essere utilizzato per effettuare una verifica dei dati provenienti da una sorgente filtrando eventuali record. Questo comportamento può essere ottenuto con un **filtro** oppure con una **regola di validazione**.

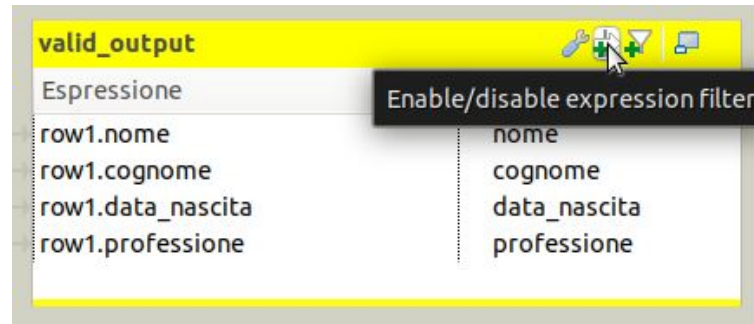
Creiamo un Job che abbiamo come input il nostro file **AST_ETL/datasets/reject/personone.csv**. Colleghiamo il file di input ad un componente di tipo **tMap** e, quest'ultimo, ad un componente di tipo **tLogRow**.



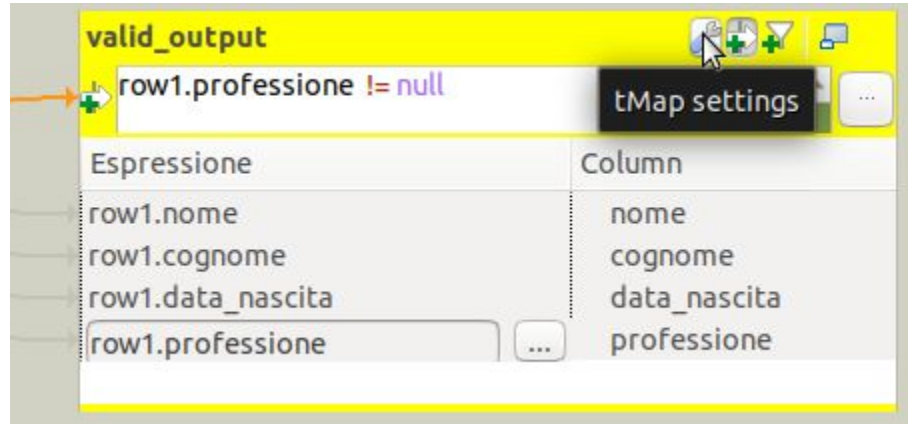
A questo punto apriamo il nostro componente **tMap** con doppio click sul componente in job designer e colleghiamo input ed output senza effettuare trasformazioni.



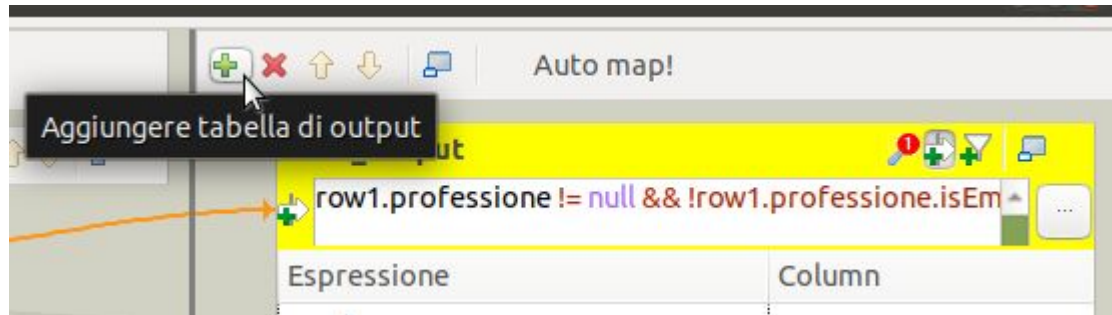
A questo punto, nella sezione output, abilitiamo il campo “expression filter”



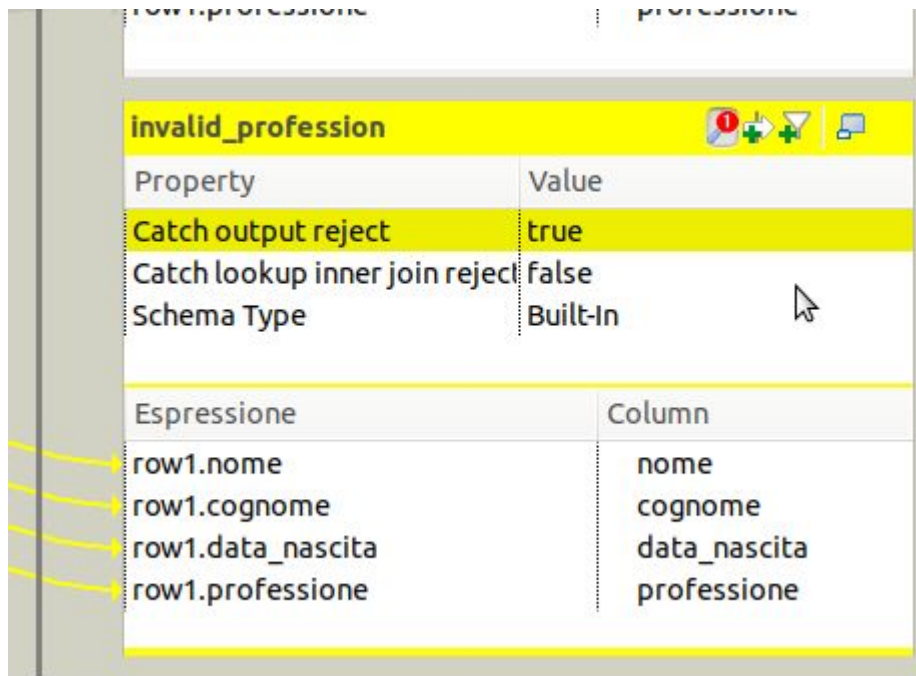
ed aggiungiamo il nostro filtro.



Aggiungiamo una nuova tabella di output nel nostro componente di tMap



Una volta aggiunta la nostra tabella di output selezioniamo il controllo **tMap setting**.



e selezioniamo il valore **true** per la proprietà **Catch output reject**

Nell'esempio precedente abbiamo, quindi, configurato un componente **tMap** per dividere l'input in due outputs separati in base ad una condizione (una sorta di IF ELSE).

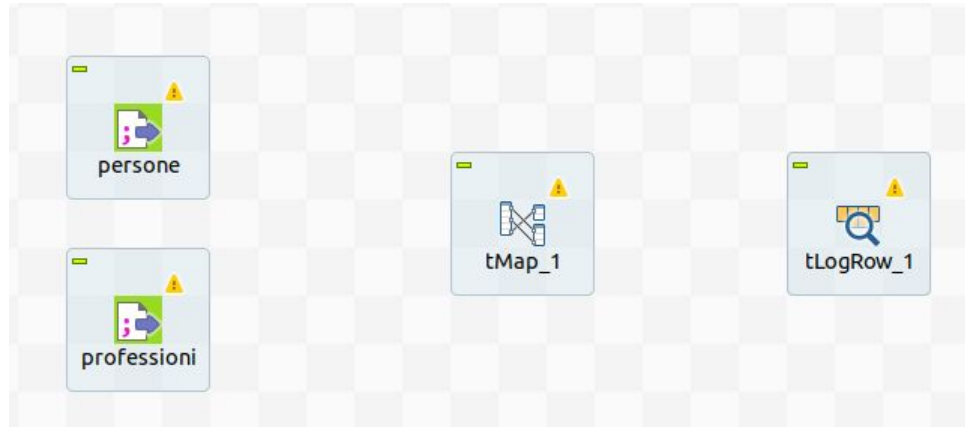
Tutte le righe che soddisfano la condizione verranno inviate al flusso **valid_input** mentre, avendo abilitato l'opzione **Catch output reject** sul secondo output che abbiamo creato, le righe che non rispettano la condizione impostata vengono scritte su **invalid_profession**.

Questa modalità può essere utilizzata per alimentare più di due output a seconda di una determinata condizione. Ricordatevi solo di effettuare il catch per quei record che non rientrano in nessuna delle condizioni configurate. Le condizioni dei vari output vengono controllate dall'alto al basso come in un costrutto if ... else.

Un altro modo per validare il contenuto di una colonna con un componente **tMap** potrebbe essere effettuato sulla base di valori contenuti in un altro input.

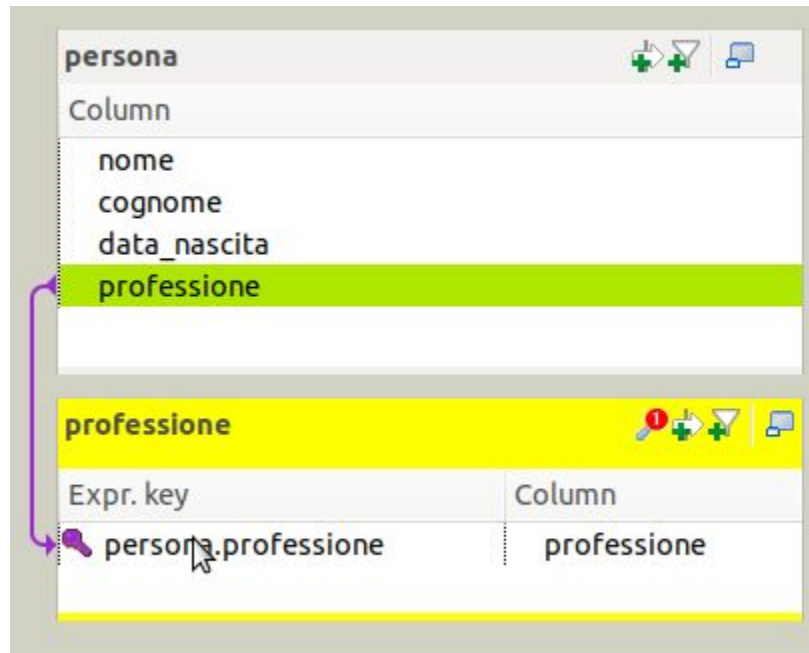
Ad esempio vorremmo recuperare tutte le persone, presenti nel file **persone.csv**, che svolgono una delle professioni presenti nel file **professioni.csv** (i due file sono nel path `AST_ETL/datasets/reject` del repository del corso).

Aggiungiamo all'interno del nostro Job i seguenti componenti: **tFileInputDelimited (persone)**, **tFileInputDelimited (professioni)**, **tMap** ed un **tLogRow**.

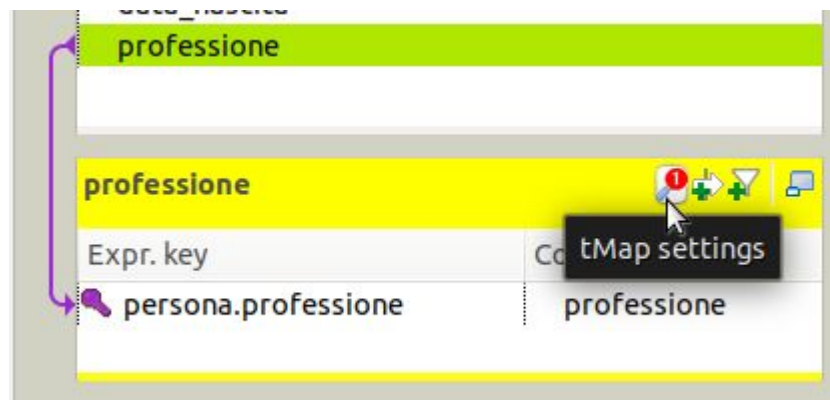


Collegiamo i nostri inputs al componente **tMap** ed apriamolo. Nella sezione relativa agli inputs vedremo che sono presenti entrambi gli schema dei nostri file.

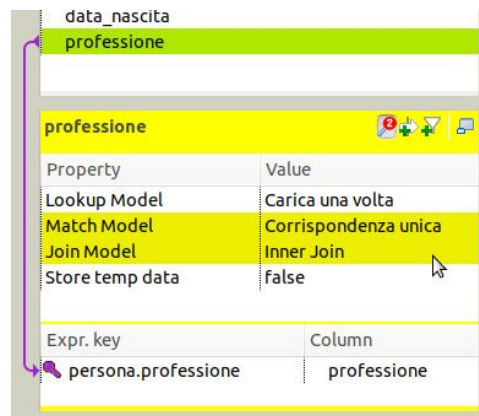
Collegiamo la proprietà professione di persone con la proprietà professione di professioni con un'operazione di Drag and Drop.




Selezioniamo il menu **tMap settings**




a questo punto selezionate per la voce **Join model** il valore **Inner join**



Aggiungiamo le due tabelle di output **valid_profession** ed **invalid_profession**

valid_profession			
Espressione	Column		
persona.nome	nome		
persona.cognome	cognome		
persona.data_nascita	data_nascita		
persona.professione	professione		

invalid_profession			
Espressione	Column		
persona.nome	nome		
persona.cognome	cognome		
persona.data_nascita	data_nascita		
persona.professione	professione		

Per la tabella di output **invalid_profession** impostiamo, all'interno del menu **tMap setting**, la proprietà **Catch lookup inner join reject** a **true** e colleghiamo questo secondo output ad un altro componente **tLogRow**

In questo modo abbiamo messo in relazione i due inputs tramite il campo con valori comuni utilizzando una **inner join** per questo motivo gli unici records del file persone.csv restituiti come “validi” saranno quelli che hanno come valore del campo professione un dei valori presenti all'interno del file professioni.csv . La proprietà **Catch lookup inner join reject** presente sull'output con le professioni non valide fa sì che in questo flusso vengano passate tutti quei records che non hanno una corrispondenza con i valori di professioni.csv (tutti i record scartati dalla join)

Molte volte le validazioni da fare su un determinato campo sono più complesse di un semplice confronto tra valori o magari richiedono delle operazioni di trasformazione del dato piuttosto che la verifica di più parametri.

In questi casi potrebbe essere più comodo, e leggibile, l'utilizzo di un metodo Java da riutilizzare all'interno dei componenti di Talend.

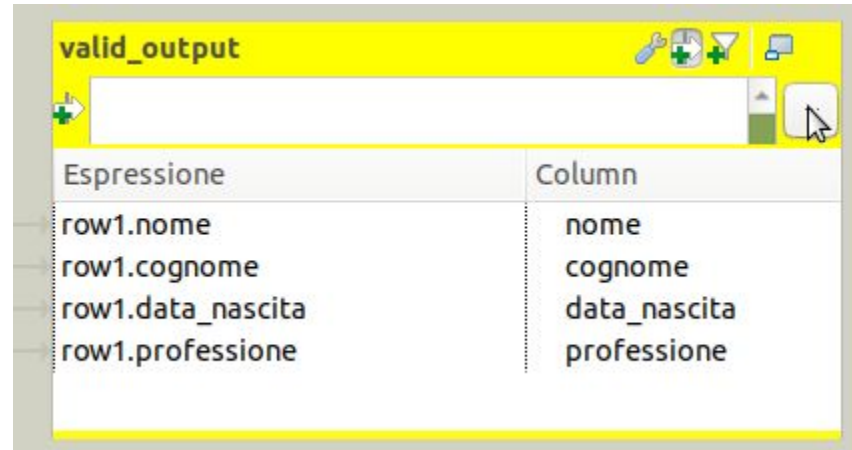
In questo caso ci tornano utili le **routine**. All'interno della view Repository, in **codice -> routine** creiamo la nuova routine **CheckRowUtils**. All'interno creeremo un metodo statico con la seguente firma

```
public static boolean check(String ... strings)
```

Attenzione al commento! Fate riferimento al commento presente sulla classe generata per sapere come scriverlo.

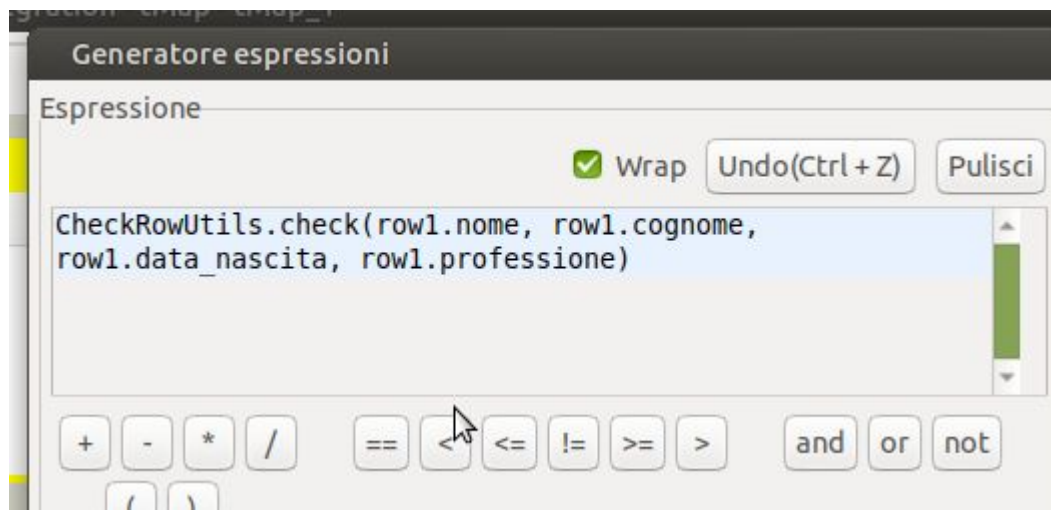
a questo punto riprendiamo il Job che abbiamo creato per filtrare il contenuto del file con il componente **tMap** (o creiamone uno con le stesse componenti).

Nel filtro del nostro tMap, però, non inseriamo il confronto del campo come fatto precedentemente ma selezioniamo il bottone accanto al campo (...)



Espressione	Column
row1.nome	nome
row1.cognome	cognome
row1.data_nascita	data_nascita
row1.professione	professione

Nella finestra di dialogo che si aprirà invochiamo il metodo che abbiamo creato precedentemente con la routine



salviamo e testiamo il nostro Job.

Contesti

Durante lo sviluppo di un software, capita sempre di dover impostare più variabili di connessione, le quali dovranno essere modificate quando il software sarà pronto ad essere rilasciato in Collaudo, e ancora una volta durante la fase di rilascio in Produzione.

Capita spesso che un software debba fare un passo indietro con conseguente modifica di tutte le variabili di connessione.

Distinguiamo quindi tre situazioni principali:

- **Locale:** solitamente le connessioni avvengono su localhost;
- **Collaudo:** le connessioni avvengono verso un server del tutto simile a quello di produzione, ma con dei dati adibiti alla fase testing

- **Produzione:** le connessioni avvengono verso il server su cui il cliente sta lavorando

Come si può risolvere, quindi, il problema di dover passare spesso da un ambiente all'altro?

Talend lo risolve grazie all'introduzione del concetto dei **Contesti**.

Ogni Contesto è visto come un insieme di variabili, capaci di assumere più valori, e quindi in grado di passare da un valore all'altro sotto istruzione dello sviluppatore.

Talend permette di creare, modificare ed eliminare contesti e gruppi contesti.

Componenti Talend

Talend offre centinaia di componenti, ognuno dei quali permette di collegarsi a sistemi esterni, svolgere operazioni di Data Quality, e molto altro.

I componenti sono selezionabili o dalla palette a destra dell'interfaccia di Talend, oppure cliccando sulla griglia interna al Job e scrivendo il nome del componente.

I componenti di input svolgono la prima parte di ciò che è un software ETL: Estrazione. Tali componenti permettono infatti di estrarre informazioni da una sorgente, tramite un flusso che potrà poi indirizzare i dati verso operazioni volte al Data Quality.

Per utilizzare come componente di input un database MySQL, è sufficiente trascinare nel Job la tabella da cui si vuole estrarre informazioni. Una volta trascinata la tabella, selezioniamo il componente **tMysqlInput**.

È possibile anche trascinare delle query create ad hoc, utilizzando lo stesso componente.

N.B. Quando si utilizza una query creata da noi, è necessario generare lo Schema tramite il pulsante Guess Schema. Ciò non è necessario trascinando nel Job l'intera tabella. Nel caso di componenti di input provenienti da un database, si può notare la presenza del campo query. Questo campo contiene la query che sarà invocata per generare il ResultSet che sarà passato attraverso la freccia Main. È possibile notare che la query è indicata tra doppi apici, come una qualsiasi stringa.

I componenti di output servono ad inserire o modificare i dati di un sistema di destinazione. I componenti di output sono collegati a una freccia di tipo Principale, associata, quindi, a uno Schema da cui prendono tutti i dati. Per utilizzare il componente di output di MySQL è sufficiente selezionare la tabella che dev'essere soggetta a un inserimento (o aggiornamento) e trascinarla all'interno del Job interessato. A quel punto basterà selezionare nella lista il connettore **tMysqlOutput**.