

XSILB

Auteur	Type de Modification	Dernier Date de Modification	Version Doc
Larid Boudjema	Mise à jour globale	03/11/09	1.4

FORMATION TALEND - Travaux Pratiques

SOMMAIRE

1	PRÉSENTATION DE LA FORMATION.....	3
---	-----------------------------------	---

2	SPÉCIFICATIONS	3
2.1	Sources	4
2.2	Chargements.....	6
2.2.1	Description des actions à effectuer	6
2.2.2	Structure de l'espace de travail	6
2.2.3	Liste des chargements de l'espace de travail	6
2.2.4	Structure du datawarehouse	9
2.2.5	Liste des chargements du datawarehouse	9
2.3	Environnement de travail.....	12
2.4	Règles de Développement	12
2.4.1	Nommage des jobs.....	12
2.4.2	Jobs unitaires	12
2.4.3	Gestion des erreurs	12
2.4.4	Enchaînement des traitements	12
2.4.5	Préconisations de design.....	12
3	TRAVAUX PRATIQUES : EXERCICES	14
3.1	Préparation.....	14
3.1.1	Charger la structure des tables	14
3.1.2	Définir la structure des fichiers plats	21
3.2	Chargement de l'espace de travail.....	21
3.2.1	Charger les directions	22
3.2.2	Charger les distances	27
3.2.3	Charger les produits	27
3.2.4	Charger les offres.....	28
3.2.5	Charger les clients	29
3.2.6	Charger les appels.....	32
3.2.7	Réalisation du Job Sequencer ou du batch d'extraction	34
3.3	Chargement du datawarehouse	38
3.3.1	Charger les directions et les produits	38
3.3.2	Charger les distances	39
3.3.3	Charger les offres.....	42
3.3.4	Charger les clients	43
3.3.5	Charger les appels.....	45
3.3.6	Réalisation du Job Sequencer ou du batch d'intégration	46
3.4	Chargement des agrégats	46
3.4.1	Charger les agrégats mensuels groupés sur les distances	47
3.4.2	Charger les agrégats mensuels groupés sur les produits	47
3.4.3	Réalisation du Job Sequencer ou du batch d'agrégation.....	48

1 PRESENTATION DE LA FORMATION

La formation a été construite de manière à vous mettre dans une situation proche de celle d'un projet. C'est à dire de partir de spécifications et de mettre en place, étape par étape, une solution complète.

Ces différentes étapes permettront d'aborder les fonctionnalités les plus utilisées de Talend.

Il s'agit également de sensibiliser à des problématiques d'intégration de données communes à une majorité de projet et valable quelque soit l'outil utilisé.

2 SPECIFICATIONS

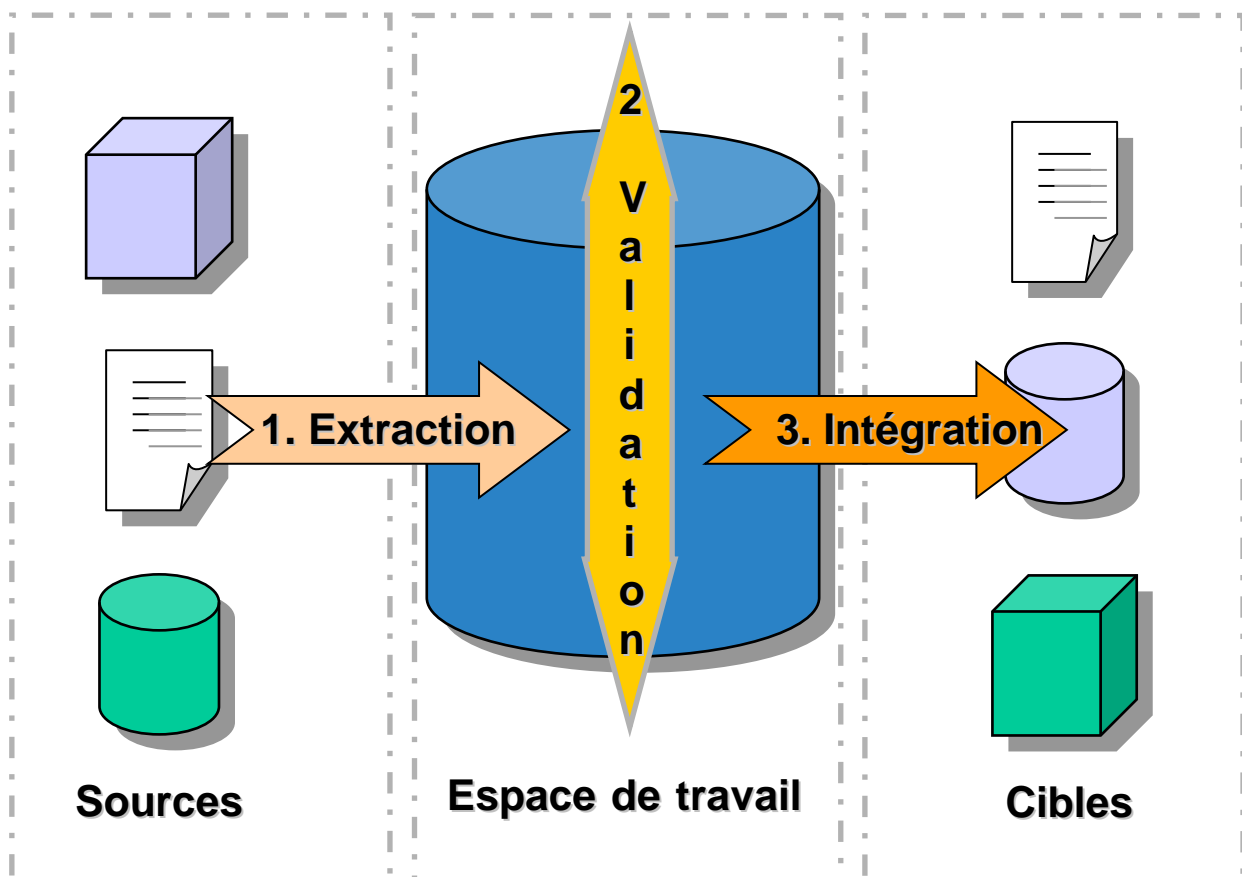
Il s'agit de mettre en place un petit datawarehouse pour un opérateur de téléphonie mobile.

L'architecture de l'alimentation est composée de trois phases :

- Extraction : Chargement des données sources dans un espace de travail
- Validation : Contrôle des données
- Intégration : Alimentation de la base de restitution (DWH) à partir de l'espace temporaire

Cette architecture permet de contrôler la qualité des données avant de les charger définitivement et d'avoir des données stabilisées en cas d'incident.

NB : Les spécifications doivent être lues entièrement avant de commencer tout développement.



2.1 Sources

Ce chapitre décrit les structures des sources qui vont être manipulées durant la formation.

Les sources de données sont hétérogènes (base de données MySQL ou autres, et fichiers plats).

Dans le cadre de ce TP, Vous pouvez être amenés à créer et charger les tables sources. Vous trouverez les scripts prévus à cet effet dans le répertoire

... \FORMATION_TALEND\Formation\Environnement\Scripts\

De ce fait vous pouvez choisir la base de données que vous voulez, mais Il faut adapter les scripts à la base choisie. Dans vos projets réels, vous aurez à traiter diverses Bases.

Données de référence : Client

Source : fichier plat – SRC_RefClient.csv

Le séparateur de champ est le point virgule. La première ligne du fichier est le nom des champs.

Champs	Type	Description
Id_Client	NUMBER(10)	Identifiant du client
Nom_Client	VARCHAR(15)	Nom du client
Prenom_Client	VARCHAR(20)	Prénom du client
Numéro	VARCHAR(10)	Numéro de téléphone du client
Id_Offre	NUMBER(3)	Identifiant de l'offre souscrite par le client
Date_abonnement	DATE	Date de souscription de l'offre

Données de référence : Offre

Source : fichier plat – SRC_RefOffre.csv

Le séparateur de champ est le point virgule. La première ligne du fichier est le nom des champs.

Champs	Type	Description
Id Offre	NUMBER(3)	Identifiant de l'offre
Lib_Offre	VARCHAR(15)	Libellé de l'offre
Desc_Offre	VARCHAR(30)	Description de l'offre

Données de référence : Direction

Source : table MySQL – SRC_REFDIRECTION

Champs	Type	Description
Id Direction	NUMBER(1)	Identifiant de la direction
Lib_Direction	VARCHAR(15)	Libellé de la direction

Données de référence : Distance

Source : table MySQL – SRC_REFDISTANCE

Champs	Type	Description
Id Distance	NUMBER(2)	Identifiant de la distance
Lib_Distance	VARCHAR(15)	Libellé de la distance
Desc_Distance	VARCHAR(50)	Description de la distance

Données de référence : Produit

Source : table MySQL – SRC_REFPRODUIT

Champs	Type	Description
Id_Produit	NUMBER(1)	Identifiant du produit
Lib_Produit	VARCHAR(8)	Libellé du produit
Desc_Produit	VARCHAR(20)	Description du produit

Données de fait : Appels

Source : fichier plat – SRC_FaitAppels.txt

Le séparateur de champ est le "pipe" (";" ou ASCII 59). La première ligne du fichier est le nom des champs.

Champs	Type	Description
Id_Client	NUMBER(10)	Identifiant du client
Date_appel	DATE	Date de l'appel
Heure_appel	VARCHAR(5)	Heure de l'appel
No_appelant	VARCHAR(20)	Numéro appelant
No appele	VARCHAR(20)	Numéro appelé
Id_Direction	NUMBER(1)	Id de la direction de l'appel
Id_Produit	NUMBER(1)	Identifiant du produit de l'appel
Id_Distance	NUMBER(2)	Identifiant de la distance de l'appel
Durée	NUMBER(5)	Durée de l'appel

2.2 Chargements

Ce chapitre décrit les actions et contrôles à effectuer pour le chargement des données.

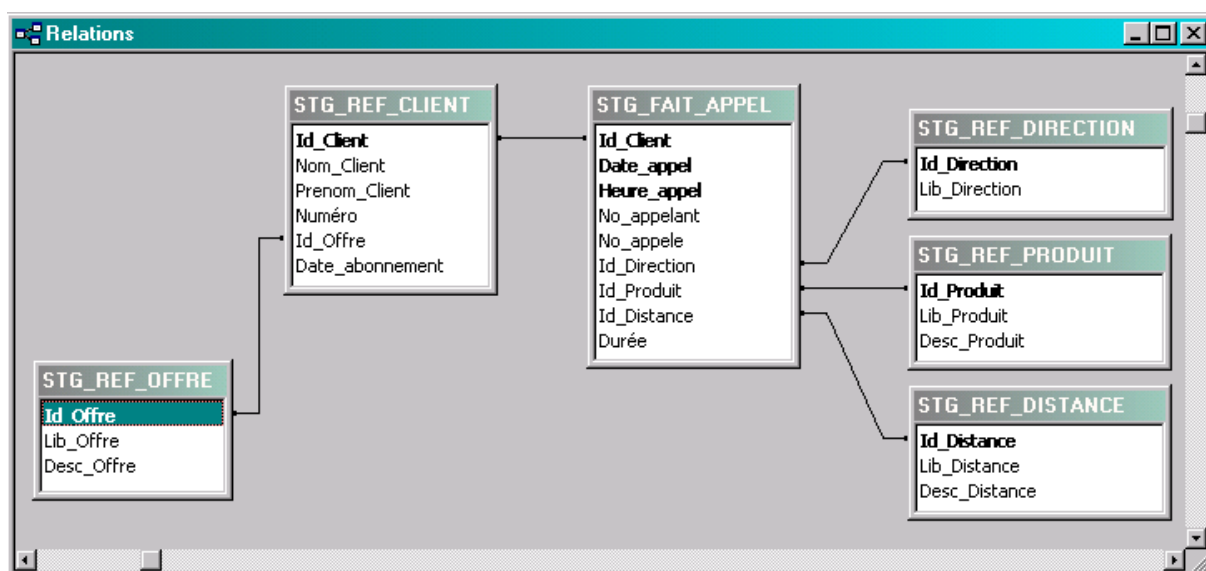
2.2.1 Description des actions à effectuer

Ce tableau récapitule les différents types d'actions rencontrés dans les associations entre les données sources et les données cibles.

Action	Description
Filtre	L'occurrence est filtrée (et non enregistrée)
Rejet	L'occurrence est rejetée dans un fichier
Alerte	L'occurrence est intégrée mais l'erreur doit être signalée via les logs

2.2.2 Structure de l'espace de travail

Les données seront préalablement chargées dans un espace de travail avec des tables temporaires dont la structure est la même que les sources. Et ce dans le but d'appliquer des règles de formatage et de faire des contrôles de cohérence sur les données.



2.2.3 Liste des chargements de l'espace de travail

Le tableau suivant donne la correspondance Source / espace temporaire et la stratégie de chargement à appliquer dans chacun des cas.

NB : ce tableau ne représente pas l'ordonnancement des alimentations. La définition de l'ordre fait partie de l'exercice.

Source	Cible	Stratégie
SRC RefOffre.csv	STG_REF_OFFRE	MAJ/insertion
SRC RefClient.csv	STG_REF_CLIENT	MAJ/insertion
SRC FaitAppels.txt	STG_FAIT_APPEL	Insertion
SRC REFDIRECTION	STG_REF_DIRECTION	Annule et remplace
SRC REFDISTANCE	STG_REF_DISTANCE	Annule et remplace
SRC REFPRODUIT	STG_REF_PRODUIT	MAJ/insertion

Les tableaux suivants fournissent la liste des contrôles à effectuer ainsi que les actions à effectuer.

Offre

Colonne	Contrôles et Règles de chargement		Action
Id_Offre	(C41)	Doit être un entier positif	Rejet
Lib_Offre	(R41)	Mettre en majuscule Si NULL, mettre à "Inconnu"	-
Desc_Offre	(R42)	Si champ trop long pour la colonne, tronquer la donnée.	-

Client

Colonne	Contrôles et Règles de chargement		Action
Id_Client	(C51)	Doit être un entier positif	Rejet
Nom_Client	(R51)	Mettre en majuscule	-
Prenom_Client	(R52)	Première lettre en majuscule	-
Numéro	(C52)	Doit être renseigné	Rejet
	(C53)	Doit être au format "06" suivi de 8 chiffres	Alerte
Id_Offre	(C54)	Doit exister dans STG_REF_OFFRE	Rejet
Date_abonnement	(R53)	Si vide, mettre à "2002-01-01"	-

Appel

Colonne	Contrôles et Règles de chargement		Action
Id_Client	(C61)	Doit être renseigné et doit exister dans STG_REF_CLIENT	Rejet
Date_appel	(C62)	Doit être renseigné	Rejet
Heure_appel	(R61)	Si vide, mettre "12:00"	-
No_appelant	(C63)	Doit être au format "10 chiffres" ou au format "+" suivi de 19 chiffres max	Alerte
No_appelle	(C64)	Doit être au format "10 chiffres" ou au format "+" suivi de 19 chiffres max	Alerte
Id_Direction	(C65)	Doit être renseigné et doit exister dans STG_REF_DIRECTION	Rejet
Id_Produit	(C66)	Peut être vide, sinon doit exister dans STG_REF_PRODUIT	Rejet
Id_Distance	(C67)	Doit être renseigné et doit exister dans STG_REF_DISTANCE	Rejet
Durée	(C68)	Doit être un entier positif sinon mettre à 0	-

Direction

Colonne	Contrôles et Règles de chargement		Action
Id_Direction	(C11)	Doit être un entier positif	Filtre
Lib_Direction	-	-	-

Distance

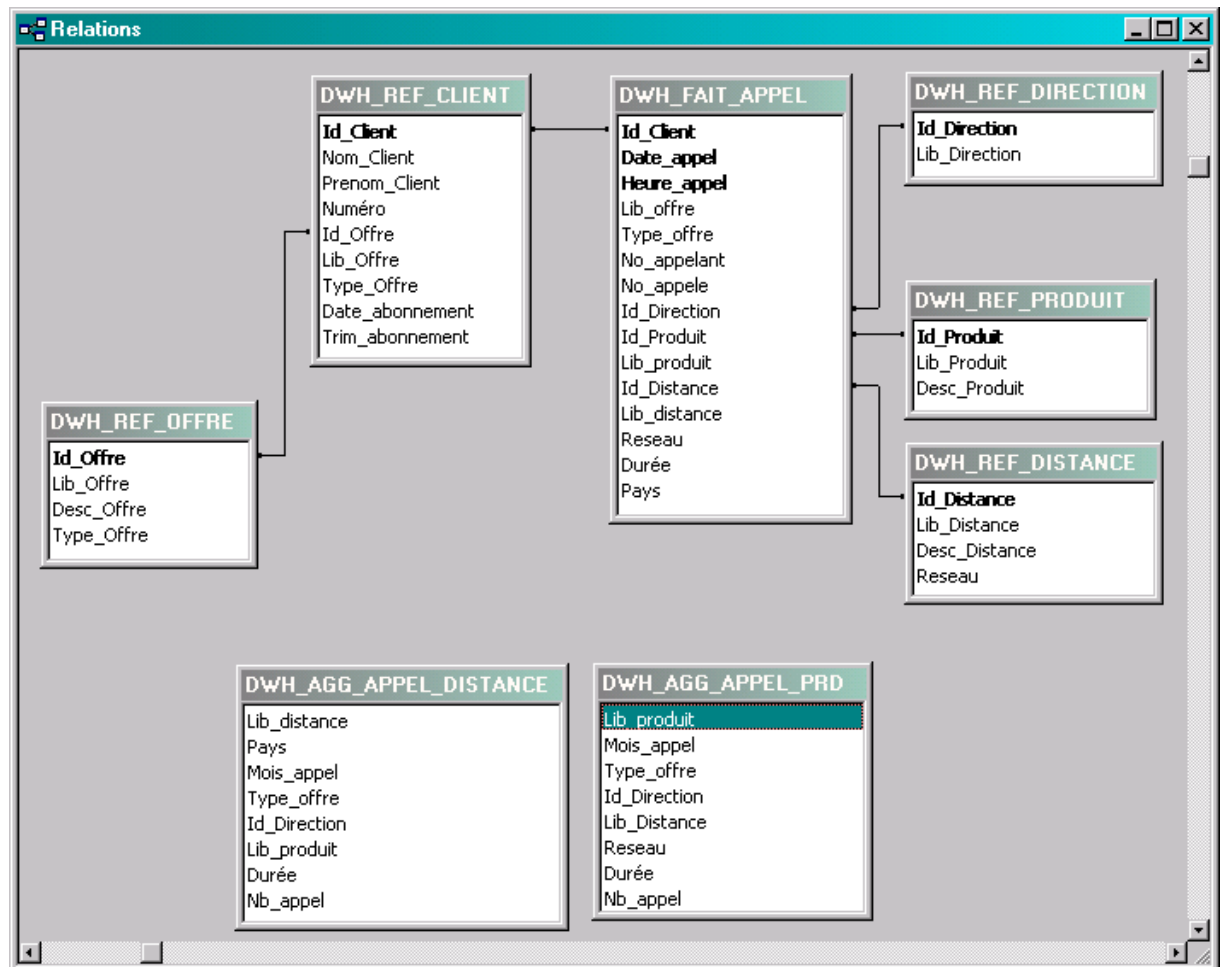
Colonne	Contrôles et Règles de chargement		Action
Id_Distance	(C21)	Doit être un entier positif	Filtre
Lib_Distance	(R21)	Si NULL, mettre à "Non renseigné"	-
Desc_Distance	-	-	-

Produit

Colonne	Contrôles et Règles de chargement		Action
Id_Produit	(C31)	Doit être un entier positif	Filtre
Lib_Produit	(R31)	Mettre en majuscule Si NULL, mettre à "Inconnu"	-
Desc_Produit	-	-	-

2.2.4 Structure du datawarehouse

Il s'agit de la base finale qui sera accessible aux utilisateurs. Elle contiendra les données atomiques (non agrégées, prise telles quelles de la source) enrichies ainsi que deux tables agrégées.



2.2.5 Liste des chargements du datawarehouse

Ce chapitre décrit les traitements et les règles de gestion à effectuer sur les colonnes de chaque table.

Chargement de DWH_REF_DIRECTION

Colonne	Mapping
Id_Direction	STG_REF_DIRECTION.Id_Direction
Lib_Direction	STG_REF_DIRECTION.Lib_Direction

Chargement de DWH_REF_PRODUIT

Colonne	Mapping
Id_Produit	STG_REF_PRODUIT.Id_Produit
Lib_Produit	STG_REF_PRODUIT.Lib_Produit
Desc_Produit	STG_REF_PRODUIT.Desc_Produit

Chargement de DWH REF DISTANCE

Colonne	Mapping
Id_Distance	STG_REF_DISTANCE.Id_Distance
Lib_Distance	STG_REF_DISTANCE.Lib_Distance
Desc_Distance	STG_REF_DISTANCE.Desc_Distance
Reseau	Analyse du champ STG_REF_DISTANCE.Desc_Distance : <ul style="list-style-type: none"> ▪ Si Desc_Distance contient le mot "fixe" (en majuscule ou en minuscule), alors mettre "FIXE" ▪ Sinon, si Desc_Distance contient le mot "portable" (idem), alors mettre "GSM" ▪ Sinon mettre "INCONNU"

Chargement de DWH REF OFFRE

Colonne	Mapping
Id_Offre	STG_REF_OFFRE.Id_Offre
Lib_Offre	STG_REF_OFFRE.Lib_Offre
Desc_Offre	STG_REF_OFFRE.Desc_Offre
Type_Offre	Analyse du champ STG_REF_OFFRE.Id_Offre : <ul style="list-style-type: none"> ▪ Si Id_Offre commence par "2", alors mettre "PREPAID" ▪ Si Id_Offre commence par "5", alors mettre "POSTPAID" ▪ Sinon mettre "INCONNU"

Chargement de DWH REF CLIENT

Colonne	Mapping
Id_Client	STG_REF_CLIENT.Id_Client
Nom_Client	STG_REF_CLIENT.Nom_Client
Prenom_Client	STG_REF_CLIENT.Prenom_Client
Numéro	STG_REF_CLIENT.Numéro
Id_Offre	STG_REF_CLIENT.Id_Offre
Lib_Offre	STG_REF_OFFRE.Lib_Offre par rapport à Id_Offre
Type_Offre	Analyse du champ STG_REF_CLIENT.Id_Offre : <ul style="list-style-type: none"> ▪ Si Id_Offre commence par "2", alors mettre "PREPAID" ▪ Si Id_Offre commence par "5", alors mettre "POSTPAID" ▪ Sinon mettre "INCONNU"
Date_abonnement	STG_REF_CLIENT.Date_abonnement
Trim_abonnement	Analyse du champ STG_REF_CLIENT.Date_abonnement : <ul style="list-style-type: none"> ▪ Définir le trimestre en fonction de la date ▪ Le trimestre doit être stocké au format YYYYTT avec TT = "01", "02", "03" ou "04"

Chargement de DWH FAIT APPEL

Colonne	Mapping
Id_Client	STG_FAIT_APPEL.Id_Client
Date_appel	STG_FAIT_APPEL.Date_appel
Heure_appel	STG_FAIT_APPEL.Heure_appel
Lib_Offre	DWH_REF_CLIENT.Lib_Offre par rapport à Id_Client
Type_Offre	DWH_REF_CLIENT.Type_Offre par rapport à Id_Client
No_appelant	STG_FAIT_APPEL.No_appelant
No appele	STG_FAIT_APPEL.No appele
Id_Direction	STG_FAIT_APPEL.Id_Direction

Colonne	Mapping
Id Produit	STG_FAIT_APPEL.Id Produit
Lib Produit	DWH_REF_PRODUIT.Lib Produit par rapport à Id Produit
Id Distance	STG_FAIT_APPEL.Id Distance
Lib Distance	DWH_REF_DISTANCE.Lib Distance par rapport à Id Distance
Reseau	DWH_REF_DISTANCE.Reseau par rapport à Id Distance
Duree	STG_FAIT_APPEL.Duree
Pays	Analyse de Id Distance / No_appelant / No_appele : <ul style="list-style-type: none"> ▪ Si Id Distance n'est pas "5", alors mettre "France" ▪ Sinon analyse du numéro international (No_appelant ou No_appele selon cas) : <ul style="list-style-type: none"> ▪ Si numero commence par "30", mettre "Grèce" ▪ Si numero commence par "33", mettre "France" ▪ Si numero commence par "352", mettre "Luxembourg" ▪ Si numero commence par "377", mettre "Monaco" ▪ Si numero commence par "41", mettre "Suisse" ▪ Si numero commence par "44", mettre "Royaume Uni" ▪ Si numero commence par "45", mettre "Danemark" ▪ Si numero commence par "49", mettre "Allemagne" ▪ Sinon mettre "Autre"

Chargement de DWH_AGG_APPEL_PRD

Agrégation de la table DWH_FAIT_APPEL.

Colonne	Mapping
Lib Produit	GROUPEMENT sur DWH_FAIT_APPEL.Lib Produit
Mois_appel	GROUPEMENT sur le mois et l'année de DWH_FAIT_APPEL.Date appel
Type Offre	GROUPEMENT sur DWH_FAIT_APPEL.Type Offre
Id Direction	GROUPEMENT sur DWH_FAIT_APPEL.Id Direction
Lib Distance	GROUPEMENT sur DWH_FAIT_APPEL.Lib Distance
Reseau	GROUPEMENT sur DWH_FAIT_APPEL.Reseau
Duree	MOYENNE de DWH_FAIT_APPEL.Duree
Nb appel	COMPTAGE du nombre de lignes groupées

Chargement de DWH_AGG_APPEL_DISTANCE

Agrégation de la table DWH_FAIT_APPEL.

Colonne	Mapping
Lib Distance	GROUPEMENT sur DWH_FAIT_APPEL.Lib Distance
Pays	GROUPEMENT sur DWH_FAIT_APPEL.Pays
Mois_appel	GROUPEMENT sur le mois et l'année de DWH_FAIT_APPEL.Date appel
Type Offre	GROUPEMENT sur DWH_FAIT_APPEL.Type Offre
Id Direction	GROUPEMENT sur DWH_FAIT_APPEL.Id Direction
Lib Produit	GROUPEMENT sur DWH_FAIT_APPEL.Lib Produit
Duree	MOYENNE de DWH_FAIT_APPEL.Duree
Nb appel	COMPTAGE du nombre de lignes groupées

2.3 Environnement de travail :

Le produit Talend dispose de plusieurs versions compatibles avec divers systèmes d'exploitations (Win, Linux, MacOS), que ce soit sur 32 ou sur 64bit. Nous utiliserons dans ce TP, le TOS-Win32-r26090-VX.Y (Talend Open Studio) ; sur un monoposte (PC) 32bit, équipé de windows 32 (XP ou ultérieur). Afin de simplifier le cas, on installera la base de données MySql en local ; sur le même poste. On aurait, bien évidemment, pu l'installer sur une autre machine, et passer par un accès réseau. Talend dispose de connecteurs natifs qui permettent de se connecter à la plupart des bases du marché ; en local ou à distance.

2.4 Règles de Développement

2.4.1 Nommage des jobs

Les noms de jobs seront formatés comme suit : SOURCE&CIBLE&ROLE

Avec SOURCE / CIBLE codés sur 3 caractères comme suit :

Code	Signification
Fic	Fichier plat
Src	Base de données source
Stg	Espace de travail
Dwh	Datawarehouse

Par exemple, pour le chargement des directions dans l'espace de travail, on aura le job SrcStgDirection.

2.4.2 Jobs unitaires

- Chaque table cible doit faire l'objet d'un job au moins (décomposition des traitements).
- Les jobs unitaires doivent être nommés de manière à pouvoir identifier aisément la source, la cible et le rôle du job.

2.4.3 Gestion des erreurs

- Les lignes rejetées doivent contenir un champ supplémentaire permettant de savoir la raison du rejet (cf. code de la règle ou du contrôle défini précédemment).

2.4.4 Enchaînement des traitements

- Le chargement de l'espace de travail doit être achevé avant de commencer le chargement du datawarehouse. Si le chargement de l'espace de travail échoue, le chargement du datawarehouse ne doit pas être lancé.
- Les jobs n'ayant pas de liens fonctionnels peuvent être exécutés en parallèle.

2.4.5 Préconisations de design

Quatre règles simples peuvent être appliquées pour améliorer la lisibilité (et donc la maintenance !) des flux.

Règle 1 :

Le flux doit être orienté de la gauche vers la droite (source à gauche et cible à droite).

Règle 2 :

Le flux doit être simple et manipuler des process (sources et cibles) de mêmes niveaux. « Evitez d'avoir plus de 15 composants dans le meme job »

Règle 3 :

Les références doivent être placées au dessus des tMap afin de les identifier rapidement.

Règle 4 :

Respecter la symétrie d'un flux lorsqu'il est découpé en plusieurs liens.

3 TRAVAUX PRATIQUES : EXERCICES

3.1 Préparation

Tout d'abord, il est nécessaire de préparer l'environnement de travail. Cela consiste en la récupération des schémas de données et en la préparation des différents environnements d'exécutions (différentes valeurs de contextes) à savoir Dev, Rec. et Prod.

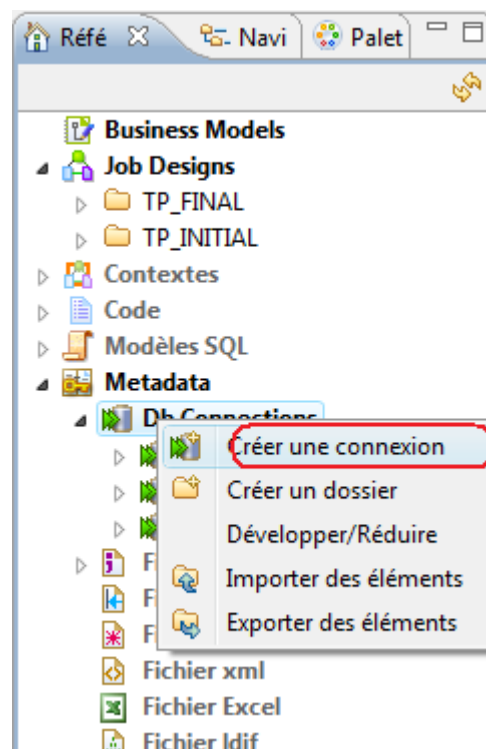
! NB : On rappelle que là à notre niveau, les structures de tables sources, STG et DWH doivent être créées.

3.1.1 Charger les structures (schémas) de tables

Pour récupérer et charger les schémas de données dans le référentiel de développement, suivez les étapes ci-dessous :

a. Créer et configurer une connexion :

- 1) Allez dans la vue Référentiel (Repository) → Metadata → DbConnection
→ cliquez sur Créer une nouvelle connexion.



- 2) Suivez et Saisissez les informations demandées dans l'assistant de création connexion comme ci-dessous.

Connexion Base de Données

Nouvelle connexion

Définir les propriétés **Le Nom est obligatoire**

Nom: MySQL_TalendFormation

Objectif:

Description: **Zone descriptive à commenter facultativement ; mais conseillée .**

Créé par : blad@blad.com

Verrouillé par :

Version: 0.1 M M

Statut:

Chemin d'accès: Sélectionner

< Retour Suivant> Terminer Annuler

Connexion Base de Données

Nouvelle connexion à la base de données - Etape 2/2

Définir les paramètres de connexion

Paramètres de la base de données

Type DB: **1** MySQL

Chaîne de connexion: jdbc:mysql://localhost:3306/?noDatetimeStringSync=true

Identifiant: **2** root

Mot de passe: **3**

Serveur: **4** localhost

Port: **5** 3306

Datasource: **6** test

Schéma:

Paramètres additionnels: noDatetimeStringSync=true

7 Vérifier **cliquez pour tester si les paramètres sont bons**

Propriétés de la base de données

Syntaxe SQL: SQL 92

Séparateur de chaîne de caractère: " Caractère Null: 000

8 Exporter comme Contexte Dissocier du Contexte

< Retour Suivant> Terminer Annuler

- 3) Cliquez sur « Exporter comme contexte » pour avoir les paramètres de connexion à cette base en variables de contexte de référentiel (groupe de contexte).
- 4) Saisissez un Nom à ce Groupe de contexte ; ex « MySQL_TalendFormation ».

Créer / Editer un groupe de contextes

Etape 1 sur 2
Renseigner les propriétés

Nom:

Objectif:

Description:

Créé par:

Verrouillé par:

Version:

Statut:

Chemin d'accès:

- 5) En cliquant sur suivant on verra toutes les variables que contient ce groupe de contexte.

Créer / Editer un groupe de contextes

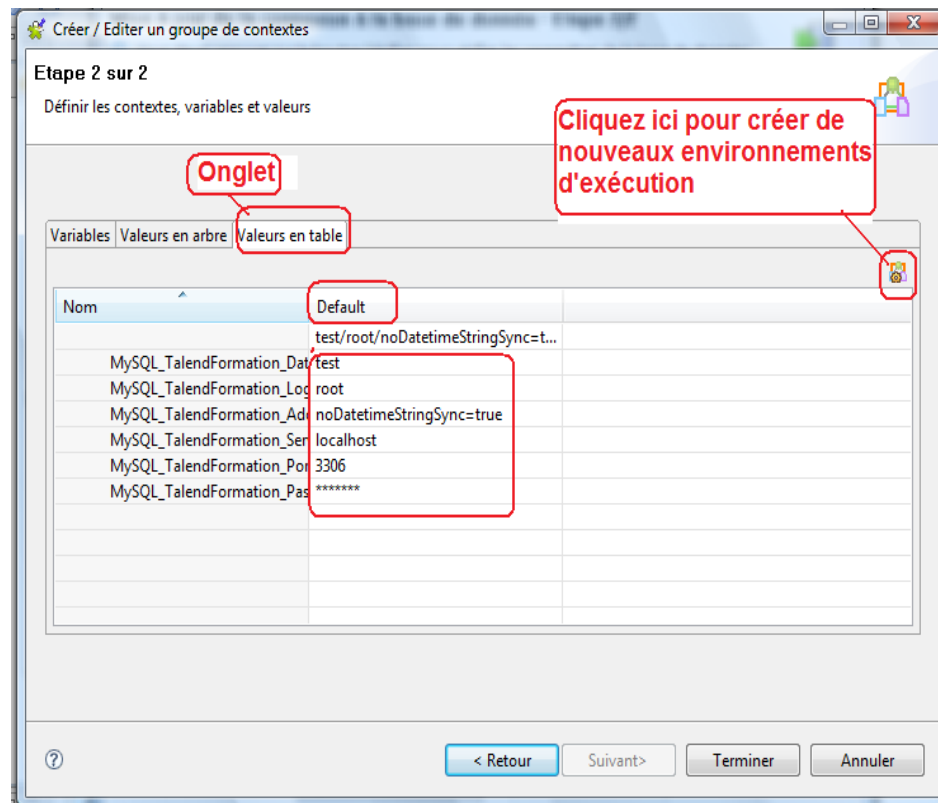
Etape 2 sur 2
Définir les contextes, variables et valeurs

Variables | Valeurs en arbre | Valeurs en table

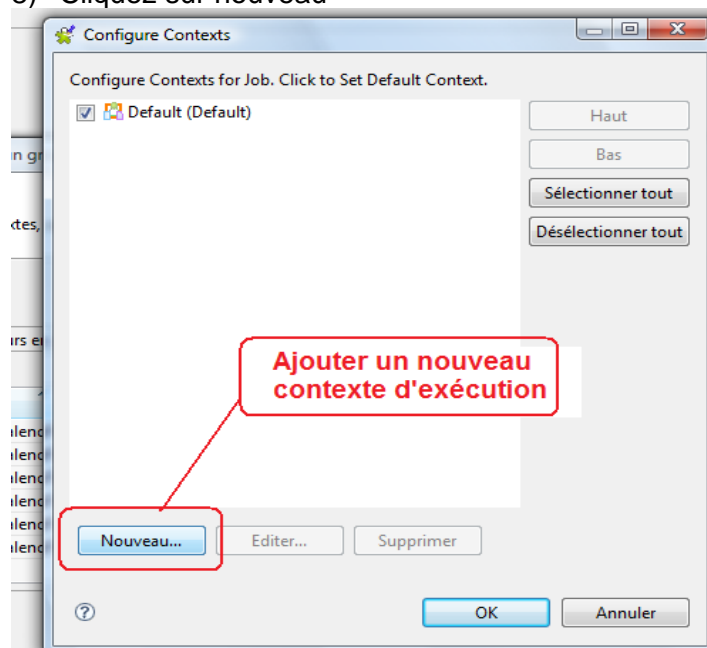
Nom	Type	Code script	Commentaire
MySQL_TalendFor	String	context.MySQL_TalendFormation_Additiona...	
MySQL_TalendFor	String	context.MySQL_TalendFormation_Database	
MySQL_TalendFor	String	context.MySQL_TalendFormation_Login	
MySQL_TalendFor	Password	context.MySQL_TalendFormation_Password	
MySQL_TalendFor	String	context.MySQL_TalendFormation_Port	
MySQL_TalendFor	String	context.MySQL_TalendFormation_Server	

☐ Ordre d'origine

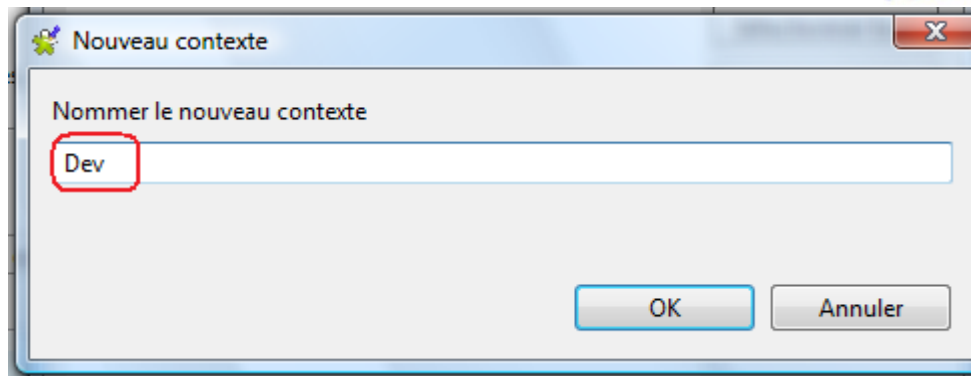
- 6) Dans l'onglet Valeurs en tables, on voit même les valeurs de ses variables, attribué pour le premier environnement d'exécution généré par défaut, et s'appelle « Default ».
- 7) Pour créer les différents environnements d'exécutions (Dev, Rec et Prod) cliquez sur le bouton en haut à droite, comme indiqué sur la copie d'écran ci-dessous.



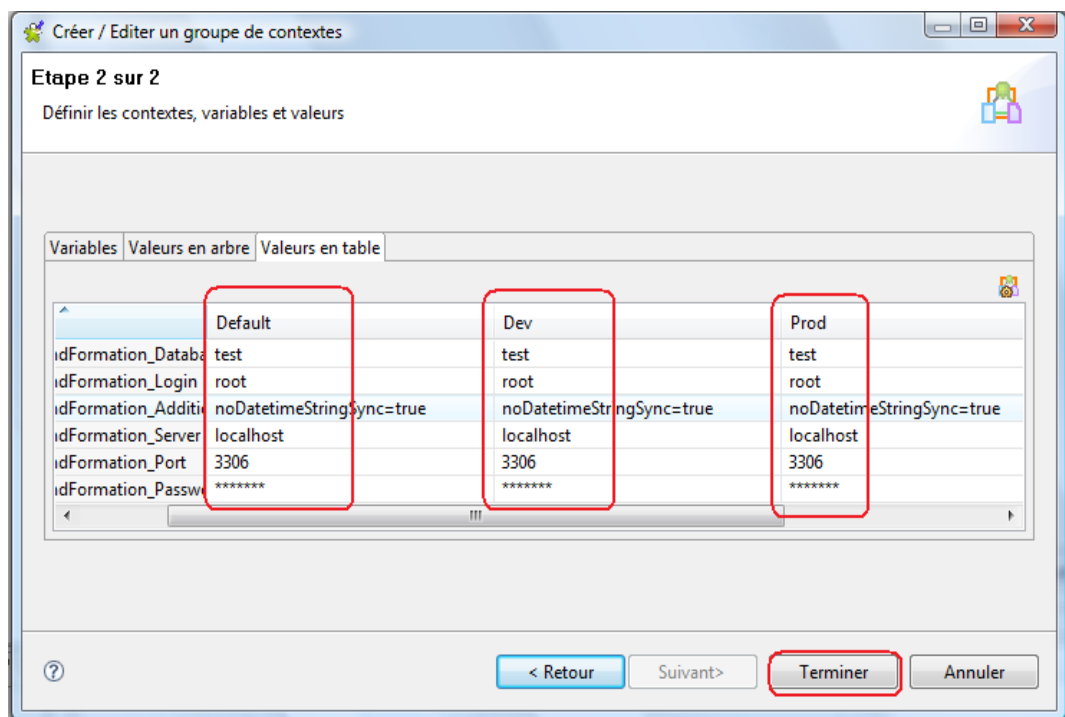
- 8) Cliquez sur nouveau



- 9) Saisissez le Nom du premier contexte , à savoir « Dev », puis Ok !



10) Répétez les mêmes étapes pour « Rec » et « Prod » puis Ok !



11) A ce stade, vous verrez les différents contextes ayant pris les mêmes valeurs que le « Default », Vous pouvez saisir les bonnes valeurs qui vont avec les différents contextes.

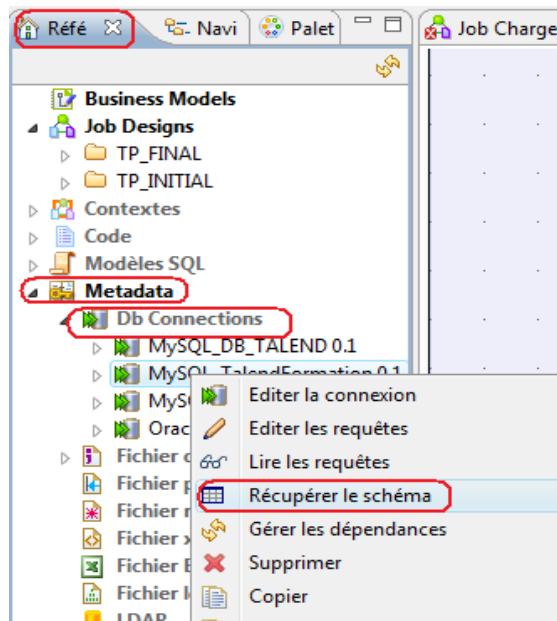
12) Terminer l'export des paramètres comme contexte

13) Enfin, au retour à la première fenêtre, cliquez sur Terminer pour finir la définition de la connexion à la base !

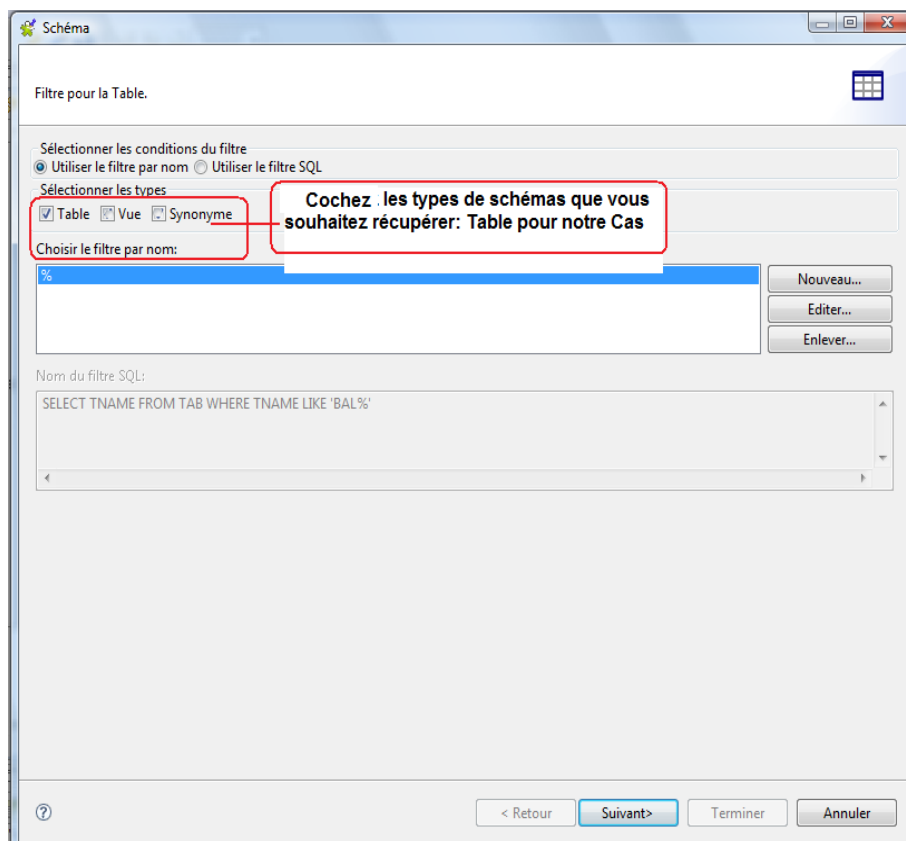
NB : Grâce à l'utilisation des contextes, à chaque exécution, vous verrez la possibilité de choisir dans quel environnement (contexte) vous souhaitez lancer le Job.

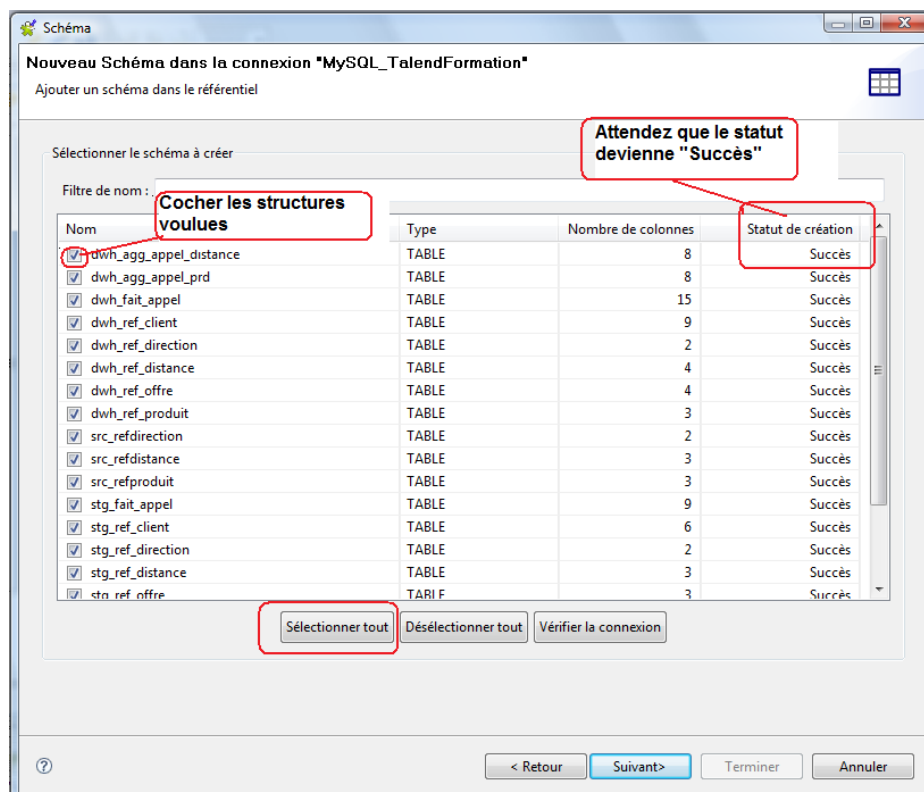
b. Récupérer les structures des tables :

- 1) Allez dans la vue Référentiel (Repository) → Metadata → DbConnection → MySQL_TalendFormation (qu'on vient de créer) → Récupérer le schéma

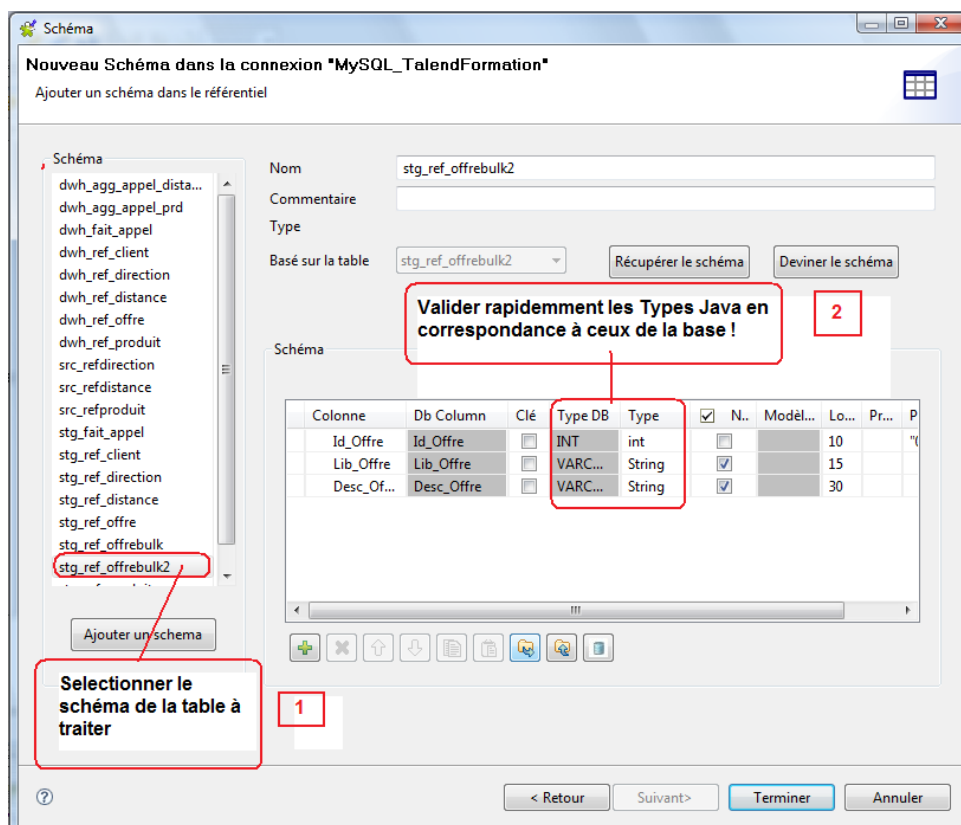


- 2) Suivez et Saisissez les informations demandées dans l'assistant de récupération de schéma comme ci-dessous.



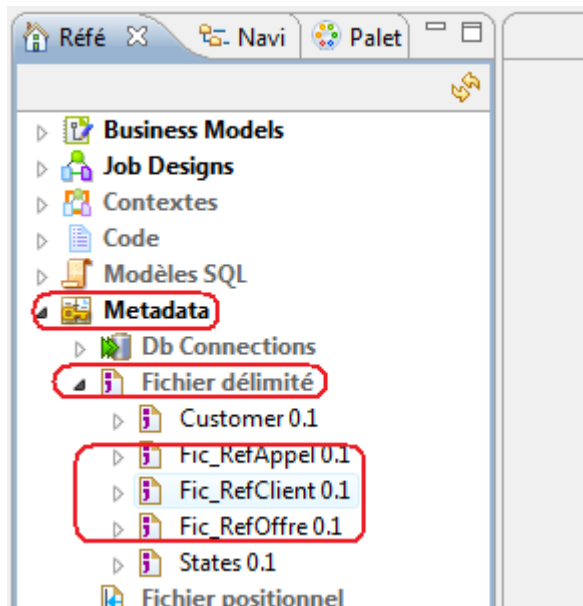


3) Suivez la dernière étape de validation puis cliquer sur Terminer



3.1.2 Définir la structure des fichiers plats

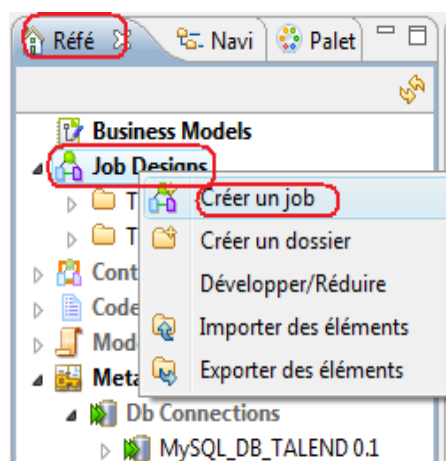
En suivant les mêmes étapes réalisées dans le TP initial à l'exo5, récupérez les schémas des fichiers plats, dont vous aurez besoin pour la suite du TP.



3.2 Chargement de l'espace de travail

Créez un Job pour chacune des tables sources.

Dans la vue Référentiel (repository) → Job Designs → Créer un job.



3.2.1 Charger les directions

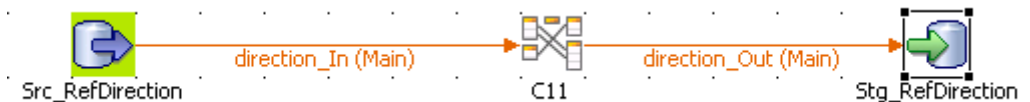
Vous devez créer le job de chargement des directions. *La création de ce premier job est guidée pas à pas.*

1. Mise en place du Design

- Le composant d'accès aux bases MYSQL. Il en faut un pour la source et un pour la cible.
- Pour les contrôles et les règles de gestion, il faut mettre un tMap qui permet de faire des tests, des transformations sur les champs et des filtres sur les flux de sortie.
- Tirer des liens entre les composants dans le sens du flux (extraction des données source, contrôle des données, intégration des données dans la cible).



- Les composants (connecteurs) et liens doivent être nommés et paramétrés (cliquez sur le composant, puis sur l'onglet composant → View → Label Format, pour le lien, cliquez 2 fois sur le bouton gauche) afin que le job soit compréhensible.



NB : La bonne pratique est de récupérer les composants (d'entrée et de sorties) du référentiel (glisser/coller). En procédant ainsi, vos composants seraient, automatiquement nommés, paramétrés et attachés au référentiel.

2. Paramétrage du job

- Les paramètres nécessaires à ce job sont les informations de connexion aux bases de données source et cible.
(Pour plus de détails, référez vous à la doc TalendOpenStudio_UG_32_a_FR.pdf page 89-91)

a. Paramétrage des Composant d'accès à la base (Connecteurs) :

Disposez les connecteurs comme présentés sur les copies d'écran ci-dessous. Pour le schéma, allez le chercher dans le référentiel et faites des glisser/déposer (comme vous avez la possibilité de le définir manuellement en Intégré (built-in), toutefois, cette méthode reste déconseillée).

"src_refdirection" (tMysqlInput_1)

Type de propriétés: **Référentiel** | BD (MYSQL):MySQL_DB_TALEND

☐ Utiliser une connexion existante

Hôte: context.MySQL_DB_TALEND1_Server | Port: context.MySQL_DB_TALEND1_Port

Base de données: conte | La variable associée à ce paramètre est : __HOST__

Utilisateur: context.MySQL_DB_TALEND1_Login | Mot de passe: context.MySQL_DB_TALEND1_Password

Schéma: **Référentiel** | BD (MYSQL):MySQL_DB_TALEND - src_refdir | Editer le schéma

Nom de table: "src_refdirection"

Type de requête: **Intégré** | Guess Query | Guess schema

Requête: "SELECT src_refdirection.Id_Direction, src_refdirection.Lib_Direction FROM src_refdirection"

✓ **tMysqlInput**

Diagram illustrating the data flow in a Talend job:

```

graph LR
    src_refdirection["src_refdirection"] --> direction_in["direction_in (Main)"]
    direction_in --> C11["C11"]
    C11 --> direction_out["direction_out (Main)"]
    direction_out --> STG_REF_DIRECTION["STG_REF_DIRECTION"]
  
```

Below the diagram, the configuration for the **"src_refdirection" (tMysqlInput_1)** component is shown:

Type de propriétés: **Référentiel** | BD (MYSQL):MySQL_DB_TALEND **1**

☐ Utiliser une connexion existante

Hôte: context.MySQL_DB_TALEND1_Server | Port: context.MySQL_DB_TALEND1_Port

Base de données: conte | La variable associée à ce paramètre est : __HOST__ **2**

Utilisateur: context.MySQL_DB_TALEND1_Login | Mot de passe: context.MySQL_DB_TALEND1_Password

Schéma: **Référentiel** | BD (MYSQL):MySQL_DB_TALEND - src_refdir | Editer le schéma

Nom de table: "src_refdirection"

Type de requête: **Intégré** | Guess Query | Guess schema

Requête: "SELECT src_refdirection.Id_Direction, src_refdirection.Lib_Direction FROM src_refdirection"

Si besoin, on peut personnaliser la requete

✓ **tMysqlOutput**

Choisissez l'action sur la table avant le transfert du flux: création, suppression, vidage etc

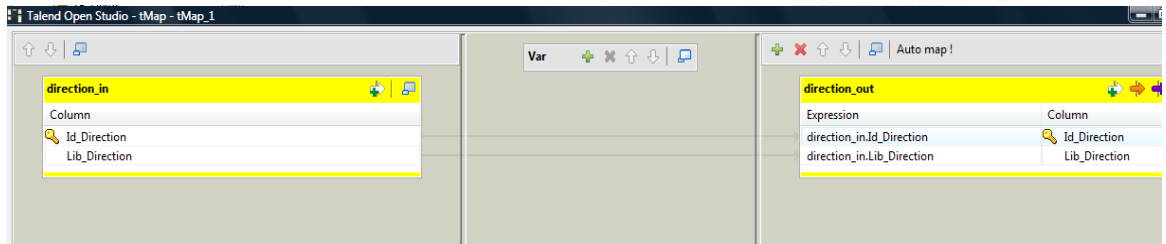
Précisez le type d'action que vous voulez sur les données: Insertion, Update, etc

b. Paramétrage du tMap :

Après avoir configuré les propriétés des composants source et cible, les colonnes apparaissent dans le tMap. Il reste à définir les règles de transfert des données (ici, la règle est : pas de transformation, mais un filtre à positionner).

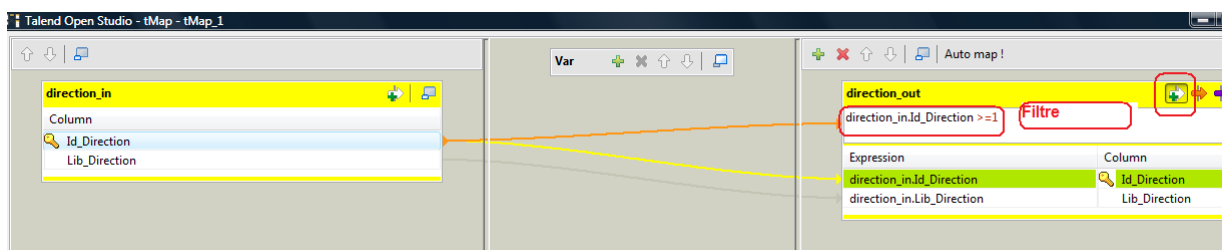
➤ **Passage des données**

Les données source sont transférées vers les champs cible sans transformation. Sélectionner les champs source et les faire glisser vers les champs cibles afin de remplir les Expressions.



Filtre

Accéder au filtre en cliquant sur le bouton représentant la flèche avec un « + » vert

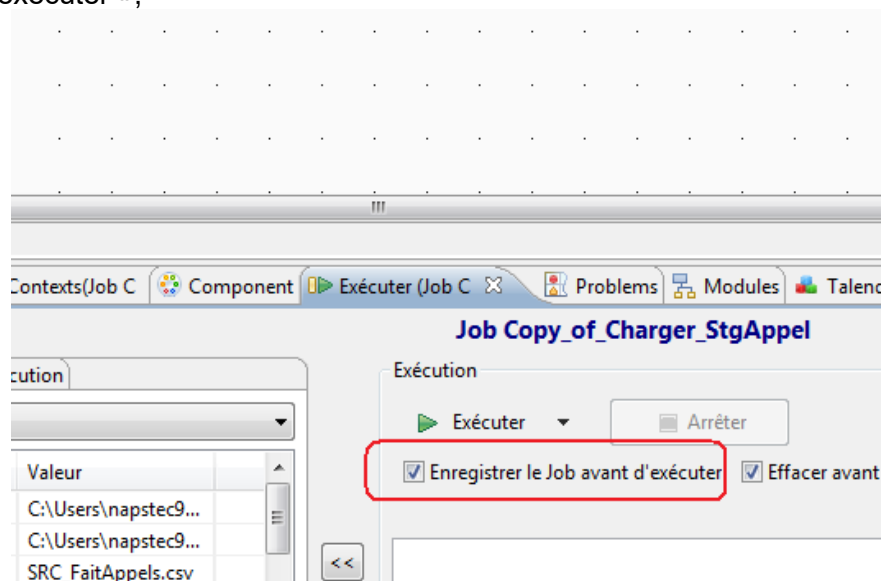


NB : En cliquant une deuxième fois sur le bouton de filtre (+), on désactive le filtre. Ainsi, pour des fins de tests, on peut le désactiver/activer sans avoir à l'effacer et le réécrire.

5. Sauvegarde et compilation du job

La sauvegarde peut se faire :

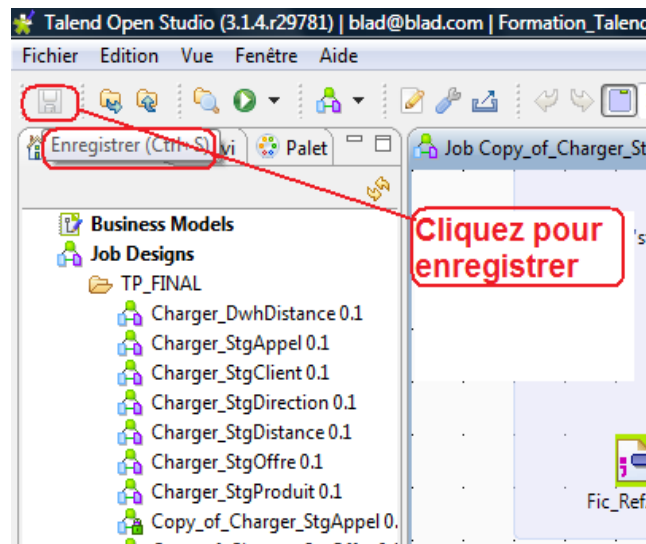
Soit automatiquement avant chaque exécution, si l'option « enregistrer le job avant d'exécuter »,



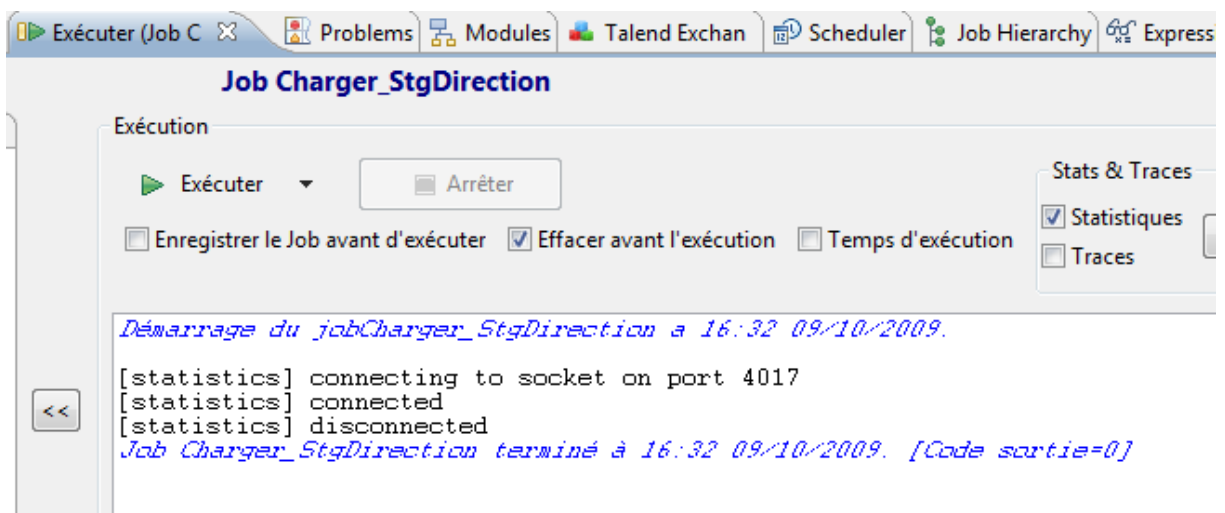
Soit manuellement en cliquant sur l'icône d'enregistrement. Dans le deuxième cas, cliquez seulement sur l'icône d'enregistrement (ou avec la combine Ctrl+s).

Pour exécuter un job, allez Dans l'onglet « Exécuter » et cliquez sur le bouton Exécuter « ou Run ». Ce faisant, les étapes suivantes s'ensuivent :

- 1) Le code du job est sauvegardé (si l'option « enregistrer avant exécution » est cochée)
- 2) Le code est compilé
- 3) Enfin, le Job est exécuté



Après si l'exécution s'est bien passé on obtient un log avec une sortie [exit code 0]




Résultats attendus :

Les cardinalités doivent être :

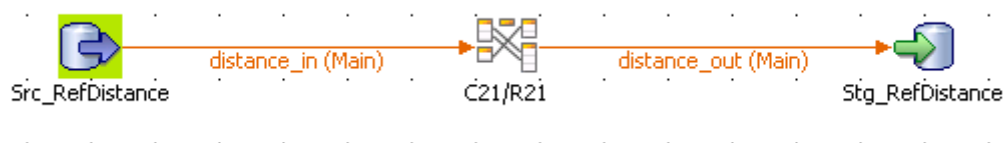
SOURCE		CIBLE		
SRC_REFDIRECTION		STG_REF_DIRECTION		
Lues	Filtrées	Supprimées	Mises à jour	Insérées
4	2	0	0	2

3.2.2 Charger les distances

La structure du job de chargement des distances est la même que pour les directions. On va donc repartir du job précédent et l'adapter.

Pour cela, allez dans le Repository -> job designs, sélectionnez le job précédent et copiez le (click droit puis Copy). Ouvrez et renommez le dans Properties -> main -> name et faites l'actualisation .

- Renommez les composants et les liens
- Les paramètres de connexion aux bases de données sont les mêmes → pas besoin de changer les paramètres du job



- Mettez le nom et le schéma des tables appropriées sur les composants source et cible.
- Remplissez les différents flux (main, rejets etc.) en sortie du tMap.

Ajouter la contrainte sur l'identifiant (C21)

- Modifier la dérivation pour le champ "LIB_DISTANCE" afin d'ajouter la règle de gestion (R21)
- lancer le job

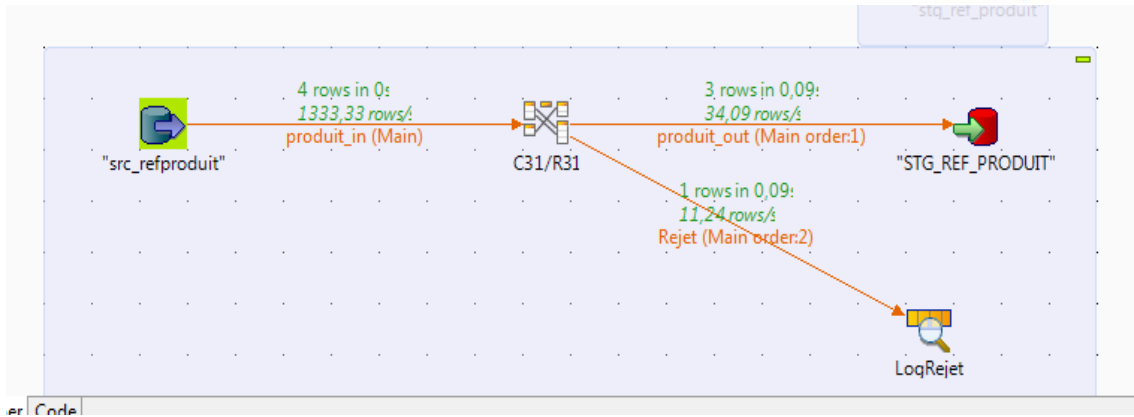
Résultats attendus :

Les cardinalités doivent être :

SOURCE		CIBLE		
SRC_REFDISTANCE		STG_REF_DISTANCE		
Lues	Filtrées	Supprimées	Mises à jour	Insérées
6	1	0	0	5

3.2.3 Charger les produits

Procéder comme précédemment en copiant un job existant et en l'adaptant au chargement des produits. Utiliser la fonction **StringHandling.UPCASE()** sur le champ "Lib_Produit" dans le tMap, pour mettre le libelle en majuscule.



Attention ! NullPointerException) :

Pensez systématiquement à mettre en place des tests de (!= null) sur les champs (nullable ; comme String) avant de leur appliquer une fonction (méthode). A titre d'exemple :

produit_in.Lib_Produit!=null && !produit_in.Lib_Produit.trim().equals("") ?

StringHandling.UPCASE(produit_in.Lib_Produit) : "Inconnu"

Résultats attendus :

Les cardinalités doivent être :

SOURCE		CIBLE		
SRC_REFPRODUIT		STG_REF_PRODUIT		
Lues	Filtrées	Supprimées	Mises à jour	Insérées
4	1	0	0	3

3.2.4 Charger les offres

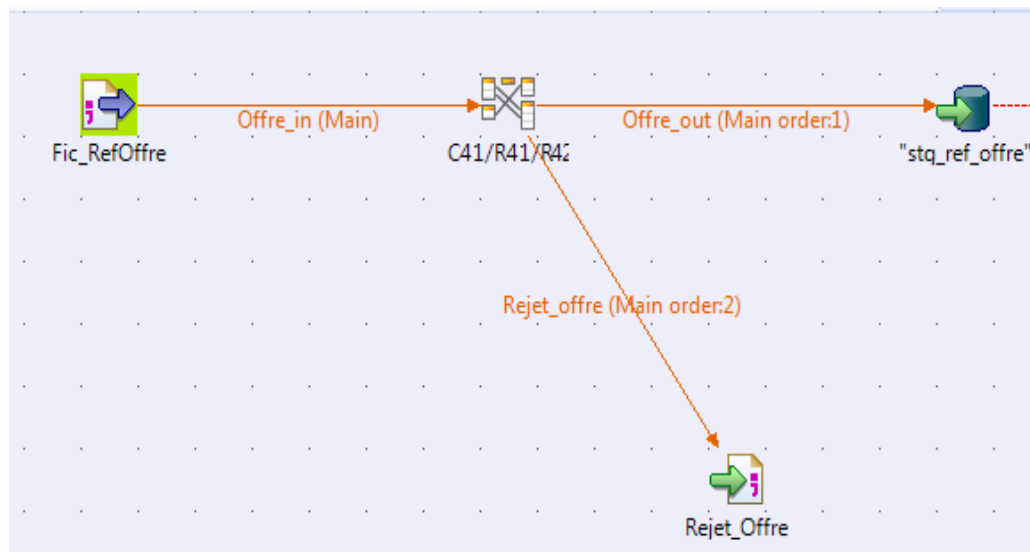
Procédez comme précédemment en copiant un job existant et en l'adaptant au chargement des offres.

Deux nouveautés dans ce job :

- ❖ La source est ici un fichier plat et non plus une table de base de données
- ❖ Le rejet impose, à la différence du filtre où les données filtrées ne sont pas prises en compte, de rediriger les données en erreur. Nous redirigerons ces données dans un fichier plat. Toutefois, il est possible de les mettre dans une table. Ce choix dépendra de la volonté de livraison des données et de la capacité à intervenir sur leur contenu.

Il faut donc :

- ❖ Il faut juste, à la place du composant d'entrée, récupérer le composant InputDelimitedFile avec un glisser/coller à partir du référentiel (repository)→Métadonnées→FichierDélimité>→<FicRefOffre>
- ❖ Ajouter un deuxième lien vers le fichier plat de rejets en sortie du tMap. Dans le fichier de rejet, vous devez récupérer, toutes les colonnes sources ainsi qu'un champ supplémentaire "cause_rejet" précisant la raison du rejet (par exemple "Identifiant de l'offre négatif").



Résultats attendus :

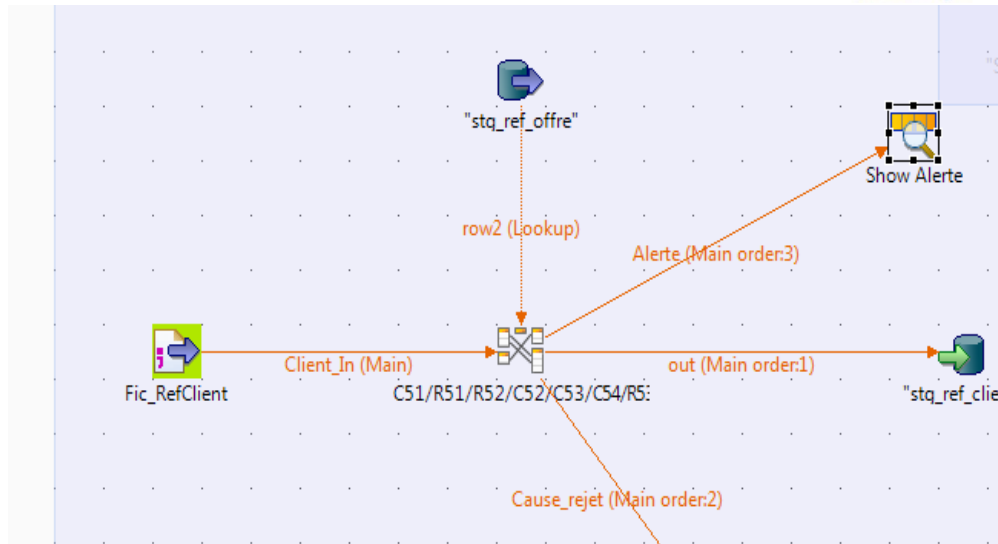
Les cardinalités doivent être :

SOURCE		CIBLES		
SRC_RefOffre.csv		STG_REF_OFFRE		
Lues	Filtrées	Supprimées	Mises à jour	Insérées
10	0	0	0	9
		RJT_OFFRE		
		Supprimées	Mises à jour	Insérées
		0	0	1

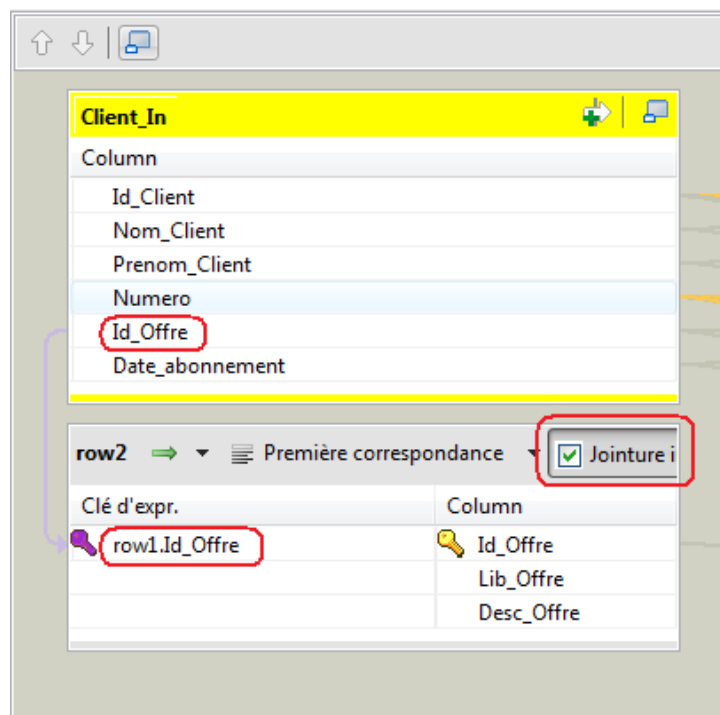
3.2.5 Charger les clients

Deux nouveautés dans ce job :

- ❖ Le rejet des données peut avoir plusieurs raisons
- ❖ Une recherche de valeur de référence (lookup) doit être faite sur la table des offres qui vient d'être chargée dans le job précédent. Ceci vérifiera l'intégrité référentielle des clients par rapport aux offres.



- Construisez un job à l'image de celui des offres ou dupliquez ce dernier et réadaptez le en reprenant les composants adéquats du repository. Vous devez reprendre en lookup la table stg_ref_offre (source de la table de STG qui est chargée dans le Job Précédent). Reliez cette table avec le tMap par un lien de type lookup et le fichier source (Fic_RefClient) par un lien main ; c'est le flux principal.
- A l'intérieur du tMap, vous devriez réaliser la jointure comment ci-dessous ; avec glisse souris, et **sans oublier de cocher la jointure interne**.

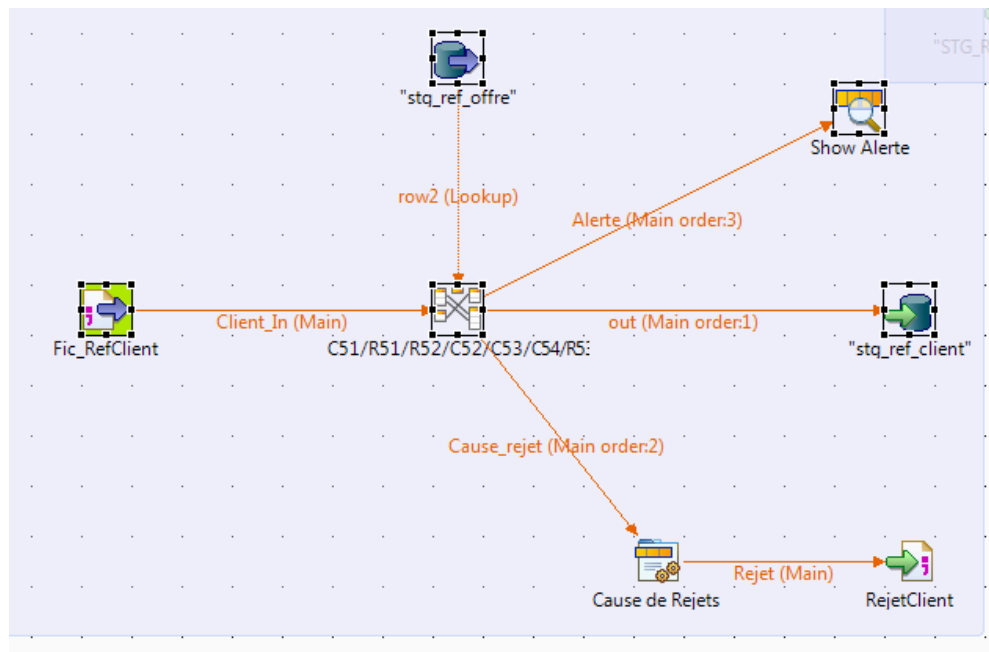


Remarques :

Vous avez plusieurs difficultés liées à l'implémentation de ce genre de problématiques avec Talend :

1. Attention aux incompatibilités des types entre la lookup et la source.
2. Utilisez le tJavaRow pour gérer les causes de rejet. Cette cause serait la concaténation des codes (C51 C52 et C54).

→EN ajoutant la gestion des types de rejet, notre Job deviendra comme suite :



Voilà un exemple code Java (à mettre dans le tJavaRow) qui gère les causes de rejets.

```
String CauseRejet="";

//input_row.Cause_rejet;
if (input_row.Id_Client.intValue() < 0)
{
    CauseRejet = "Id_Client negatif" ;
} else {
    if ( input_row.Numero == null || input_row.Numero.trim().equals("") ) {
        CauseRejet="Numero non renseigné";
    } else {
        CauseRejet="id_offre inexistant";
    }
}

output_row.Id_Client = input_row.Id_Client;
output_row.Nom_Client = input_row.Nom_Client;
output_row.Prenom_Client = input_row.Prenom_Client;
output_row.Numero = input_row.Numero;
output_row.Id_Offre = input_row.Id_Offre;
output_row.Date_abonnement = input_row.Date_abonnement;
output_row.Cause_rejet = CauseRejet ;
```

Résultats attendus :

Les cardinalités doivent être :

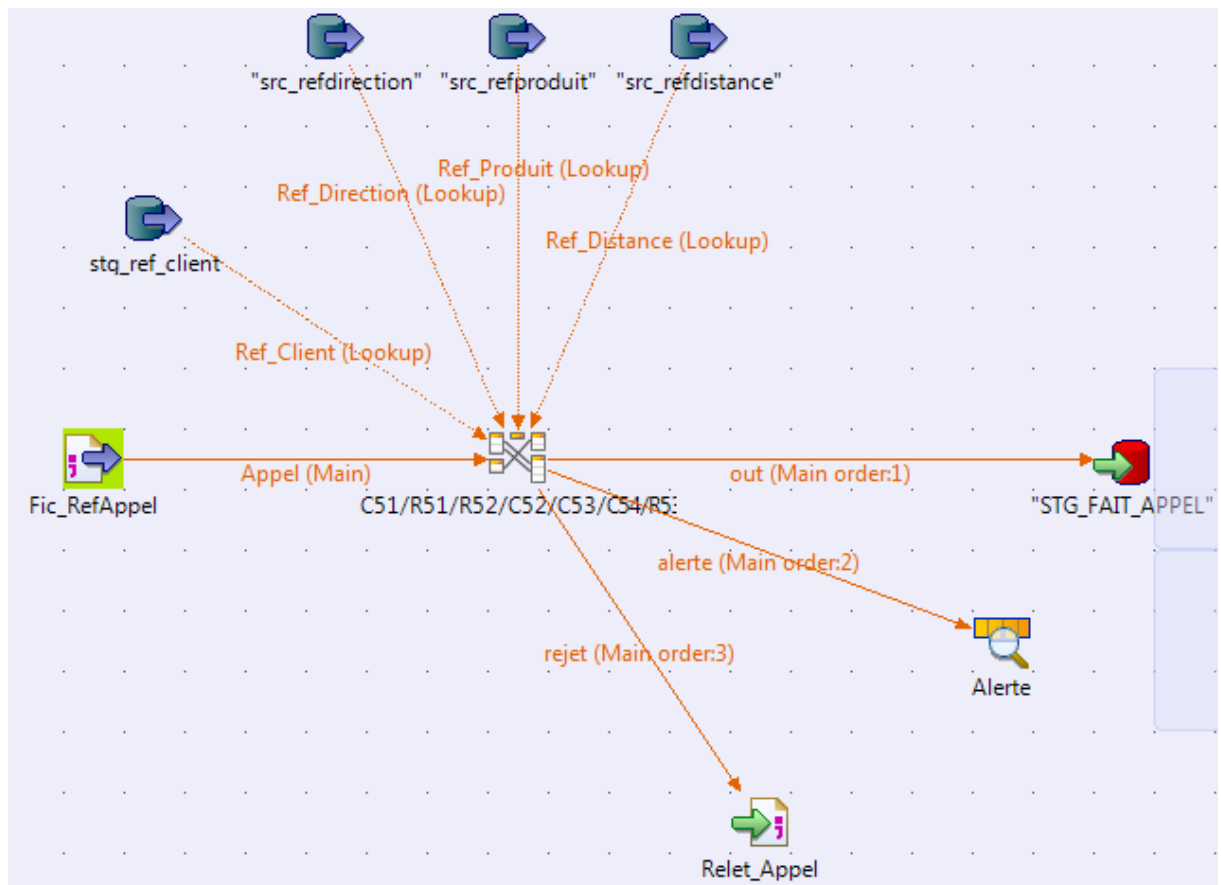
SOURCE		CIBLES		
SRC_RefClient.csv		STG_REF_CLIENT		
Lues	Filtrées	Supprimées	Mises à jour	Insérées
25	0	0	0	21
		RJT_CLIENT		
		Supprimées	Mises à jour	Insérées
		0	0	4
		ALT_CLIENT		
		Supprimées	Mises à jour	Insérées
		0	0	6

3.2.6 Charger les appels

Toutes les fonctionnalités nécessaires à ce job ont déjà été utilisées (lookup, rejet, alerte).

Ce job reprend le même principe que le précédent si ce n'est qu'il y a quatre références : Produit, Direction, Distance et Client.

Conseil : Il n'est pas nécessaire de poser autant de tMap qu'il y a de lookup mais de les connecter au même (au dépend des problématiques des jointures multiple ; c'est toujours plus simple de tracer une jointure à deux flux qu'une jointure à plusieurs flux). Ceci afin d'obtenir les meilleures performances. Le Job doit ressembler à ceci :



Pour contrôler le format des numéros de téléphone No_appelant et No_appelé, utilisez les **Expressions régulières**. Dans le cas de java, voici comment on peut réaliser C63 et C64 :

```
Appel.No_appelant !=null && ( Appel.No_appelant.matches("[0-9]{10}") ||  
Appel.No_appelant.matches("^\\+[0-9]{19}") )
```

NB : Dans le tMap, quand on doit réaliser des transformations compliquées et factorisables (dont le résultat va être utilisé dans plusieurs champs cibles), il est de bonne pratique d'utiliser des variables temporaires.

Ex : Dans notre cas, pour implémenter C63 et C64, je déclare deux variables temporaires de type booléen que j'assigne avec les valeurs résultantes des tests de correspondance entre les numéros et les formats voulus. Concernant la correspondance, je la réalise en utilisant les expressions régulières.

The screenshot shows the Talend Open Studio tMap interface. On the left, the 'Appel' source table is mapped to the 'out' target table. The 'Var' section in the center lists several variables, with two highlighted: 'Appel.No_appelant !=null && (Appel.No_appelant.matches("[0-9]{10}") || Appel.No_appelant.matches("^\\+[0-9]{19}"))' and 'Appel.No_appelé !=null && (Appel.No_appelé.matches("[0-9]{10}") || Appel.No_appelé.matches("^\\+[0-9]{19}"))'. These are labeled 'C63 et C64'. The 'out' table on the right shows columns like 'Id_Client', 'Date_appel', 'Heure_appel', 'No_appelant', 'No_appelé', 'Id_Direction', 'Id_Produit', 'Id_Distance', and 'Duree'. The 'alerte' and 'rejet' tables also show columns for alerts and rejections. A red box highlights the variable declarations with the text: 'Quand on a des transformations compliquées et factorisables (à réutiliser plusieurs fois), il est judicieux d'utiliser des variables temporaires: Dans notre cas (C63 et C64), je déclare deux variables booléennes que j'assigne avec les valeurs des tests de correspondance des numéros aux formats voulus. La correspondance, elle même, je la vérifie avec les expressions régulières'.

Résultats attendus :

Les cardinalités doivent être :

SOURCE		CIBLES		
SRC_FaitAppel.csv		STG_FAIT_APPEL		
Lues	Filtrées	Supprimées	Mises à jour	Insérées
160	135	0	0	135
		RJT_APPEL		
		Supprimées	Mises à jour	Insérées
		0	0	25
		ALT_APPEL		
		Supprimées	Mises à jour	Insérées
		0	0	14

3.2.7 Réalisation du Job Sequencer ou du batch d'extraction

Le batch réunira les actions et jobs nécessaires à l'extraction des données source vers l'espace temporaire (STG). Il est maintenant nécessaire de déterminer les enchaînements des exécutions et de déterminer les niveaux de parallélisme possibles.

La règle à cela est :

- Si on a une table fille qui référence, par une Foreign Key (clé étrangère), une table mère, on ne peut pas charger la fille avant que la mère soit préalablement chargée.
- Si les tables sont complètement indépendantes (pas d'ordre de causalité qui les régit) , on peut les charger en parallèle .

Exercice !

Avant de voir la solution plus loin dans ce TP, pensez, en se basant sur les règles ci-dessous, à établir le bon ordre de chargement des différentes tables.

***Solution

Il est nécessaire de charger les références suivantes en premier (avec possibilité de parallélisme entre les quatre premiers) :

- Direction
- Distance
- Produit
- Offre

Puis de charger les références référencées par d'autres références :

- Client

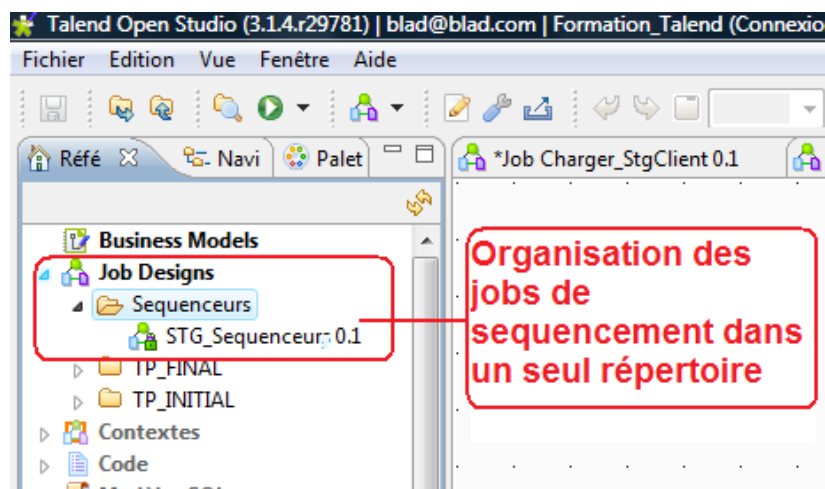
Puis de charger les faits

- Appel

Nous pourrions lancer en série ces jobs, mais il paraît plus efficace de lancer tous les jobs indépendants au sein d'un même niveau en parallèle. Ainsi Les directions, distances, produits et offres ne sont liés ni par les sources ni par les cibles et il n'y a aucune contrainte d'intégrité (référentielle externe ; clé étrangère) qui les rend interdépendants. Il est donc parfaitement possible de les lancer en parallèle.

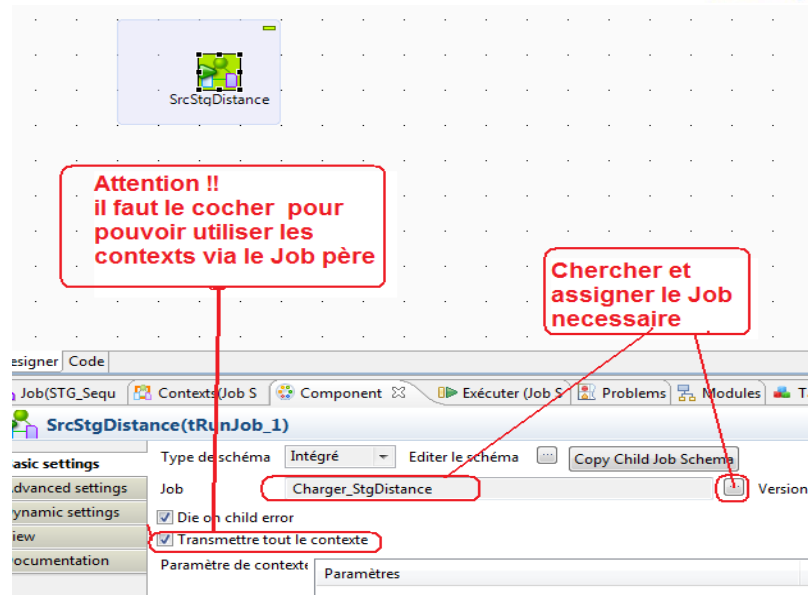
La méthode consiste à créer, dans le référentiel, un répertoire « Séquenceurs » où, on va regrouper les jobs de séquencement.

Dans ce répertoire, créez le premier job de séquencement des jobs d'alimentation de la source vers l'espace temporaire (STG), ce qui donnera ceci :

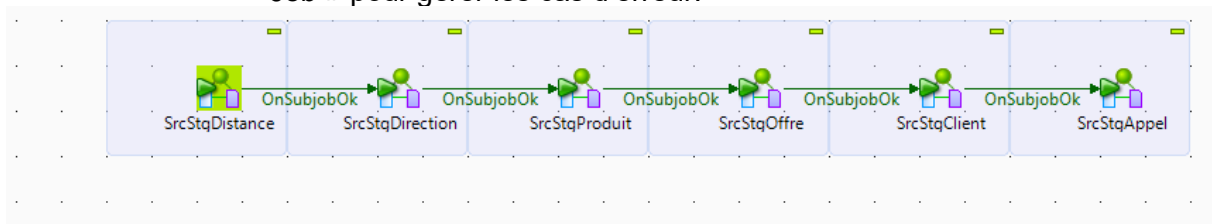


❖ Composition du Job STG_sequencEUR :

- 1) Ouvrez ce Job, encore vide.
- 2) Vous avez dans la « palette→Système » le composant tRunJob qui vous permet d'enchaîner et d'exécuter Les jobs que vous avez dans le référentiel (Repository).
- 3) Pour chaque Job il faut prendre un tRunJob et le configurer.
 - Pour le Job des distance, prenez un tRunJob et renommer le par « SrcStgDistance ». Faites le pointer sur le Job SrcStgDistance du repository puis cochez « transmettre tout le contexte »
 - Pour les autres, suivez les mêmes étapes pour les implémenter.

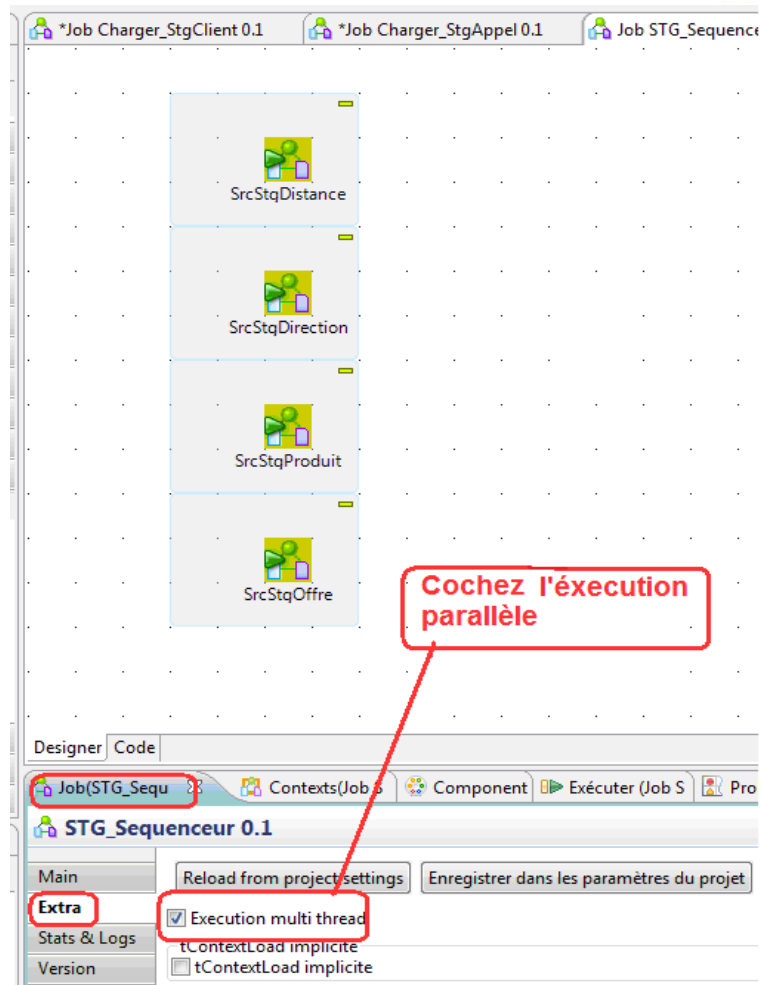


- 4) Enchaînez les avec les lien (Triggers) « sur réussite du Sous-Job ».
Pour information, on peut aussi utiliser les liens « sur erreur du Sous-Job » pour gérer les cas d'erreur.



NB : Pour paralléliser un ensemble de sous job, il faut :

- ✓ Ne pas les relier avec les liens triggers
- ✓ Cochez l'option « Execution multi thread » ; Onglet Job→Extra.



Remarque :

Grace au tRunJob, on peut réaliser toute forme d'hierarchie et d'enchaînement de Job qu'on peut concevoir. Vous pouvez, à titre d'exercice, tenter d'établir une organisation de Jobs de sorte à pouvoir paralléliser les quatre premiers, puis les enchaîner avec les deux derniers.

3.3 Chargement du datawarehouse

Toujours par le même principe, vous devez adapter les jobs précédents pour créer les jobs d'alimentation du datawarehouse.

Remarque : Dans le schéma d'architecture présenté précédemment, il était fait mention d'une étape de contrôle et de validation de données. L'étendu du projet de la formation ne nécessite pas de réelle validation de donnée, cette étape n'a donc pas lieu d'exister, nous pouvons donc passer directement à la troisième étape.

3.3.1 Charger les directions et les produits

Il s'agit d'un transfert de base à base sans transformation.

Reprenez les Jobs d'alimentation de la source vers l'espace temporaire Stg («Charger_StgDirection »,« Charger_StgDirection ») et renommez les respectivement en « Charger_DwhDirection » et « Charger_DwhDirection » puis, pointez les connecteurs de base sur les bons schémas (Stg en source et Dwh en cible) et surtout, à ne pas oublier, rafraîchissez les requêtes.

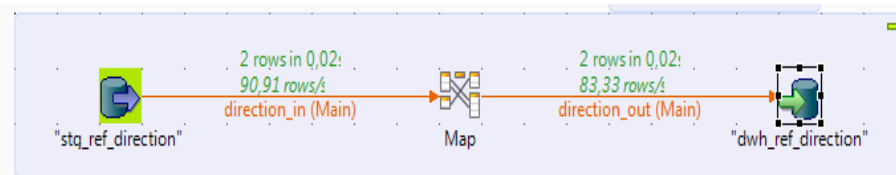
Dans le Job de produit, remplacez la transformation par un transfert directe.

Remarque :

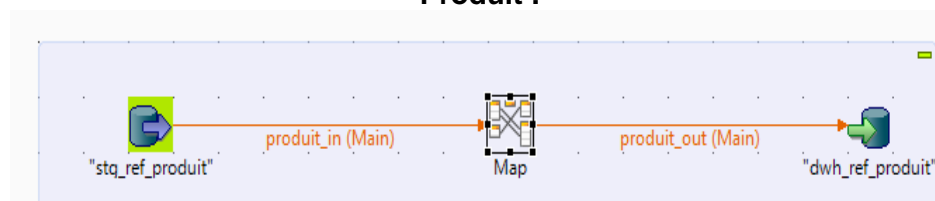
Bien qu'il soit possible de mettre les deux flux dans le même job, il est recommandé pour favoriser la maintenance de créer deux flux distincts.

Le mode de transfert est mise à jour / insertion. Il vous faut choisir l'option (insert or update) dans le composant cible.

Direction :



Produit :



Résultats attendus :

Les cardinalités doivent être :

SOURCE		CIBLES		
STG_REF_DIRECTION		DWH_REF_DIRECTION		
Lues	Filtrées	Supprimées	Mises à jour	Insérées
2	0	0	0	2
STG_REF_PRODUIT		DWH_REF_PRODUIT		
Lues	Filtrées	Supprimées	Mises à jour	Insérées
3	0	0	0	3

3.3.2 Charger les distances

Il s'agit de transférer les données sources et d'enrichir la cible d'une nouvelle information (champ Réseau) définie sur les colonnes existantes.

Remarque :

Deux possibilités s'offrent à vous pour transformer la description DESC_DISTANCE :

- Soit directement dans le tMap (ce qui est envisageable vu la simplicité du cas) avec l'opérateur ternaire de Java

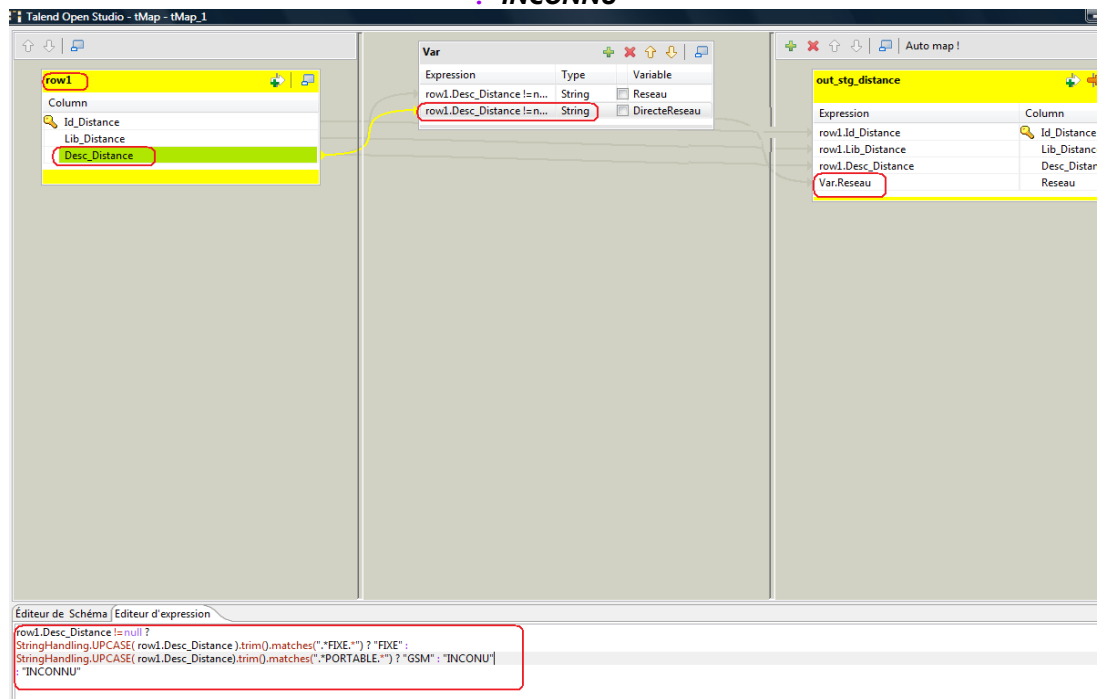
Condition ? valeur_Si_vrai : valeur_si_faux

Ex :

row1.Desc_Distance != null ?

StringHandling.UPCASE(row1.Desc_Distance).trim().matches(".*FIXE.*") ? "FIXE" :

StringHandling.UPCASE(row1.Desc_Distance).trim().matches(".*PORTABLE.*") ? "GSM" : "INCONNU"
: "INCONNU"



- Soit en définissant une routine dans le référentiel, puis en l'appelant dans le tMap.

La deuxième est nettement plus industrielle, d'autant plus si on a possibilité d'avoir à faire plusieurs fois cette transformation (pas dans notre cas).

Exercice !

En guise d'exercice, essayez d'écrire une routine qui fait cette transformation. Puis, comparez là à la solution présentée plus bas.

La Solution avec la Routine :

Référez vous à la Doc TalendOpenStudio_UG_32a_FR. En voilà ci-dessous un exemple de code pour cette transformation.

```
/**
 * TypeReseau:  retourne le type de reseau , .
 *
 *
 * {talendTypes} String
 *
 * {Category} User Defined
 *
 * {param} string("portable") input: The string need to be printed.
 *
 * {example} TypeReseau ("portable") # GSM !.
 */
public static String TypeReseau(String DescDistance) {
    String Ret_Str="";
    if (DescDistance == null || DescDistance.trim().equals("")) {
        Ret_Str = "INCONNU"; //
    }

    if (DescDistance.toLowerCase().trim().matches(".*fixe.*")){
        Ret_Str = "FIXE";
    }else if (DescDistance.toLowerCase().trim().matches(".*portable.*")){
        Ret_Str = "GSM";
    }

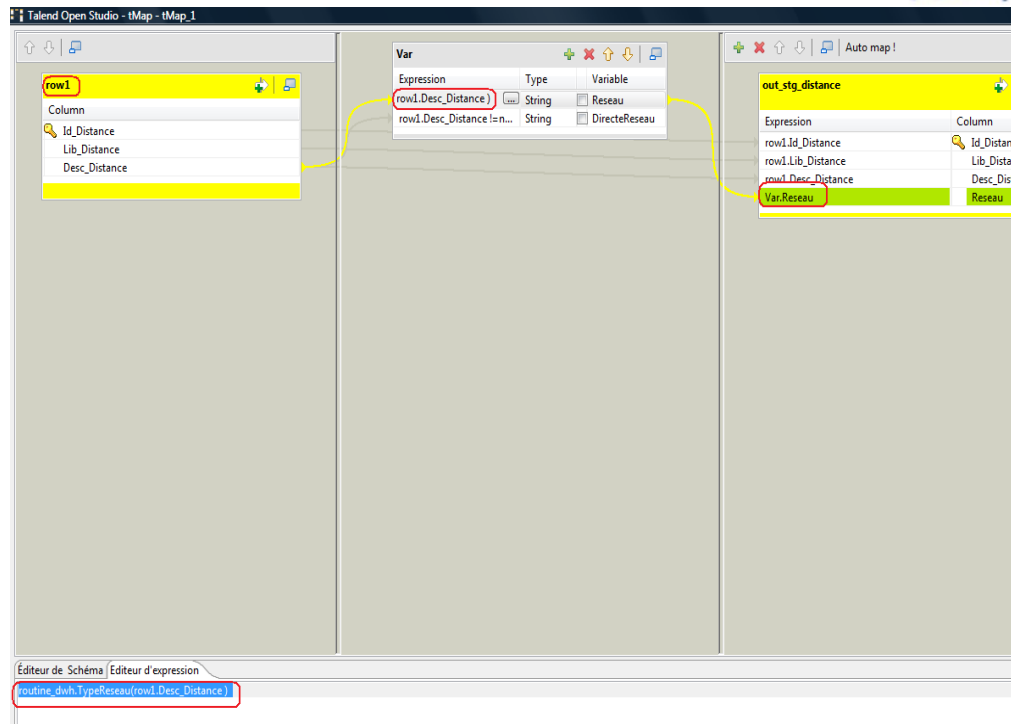
    return Ret_Str;
}
```

**Partie en JavaDoc à saisir
obligatoirement pour visualiser cette
routine dans la bibliotheque des
fonction, catégorie : User Defined**

- L'appel de routine se fait par CTRL+Espace, puis choisir notre routine dans la liste qui défile.

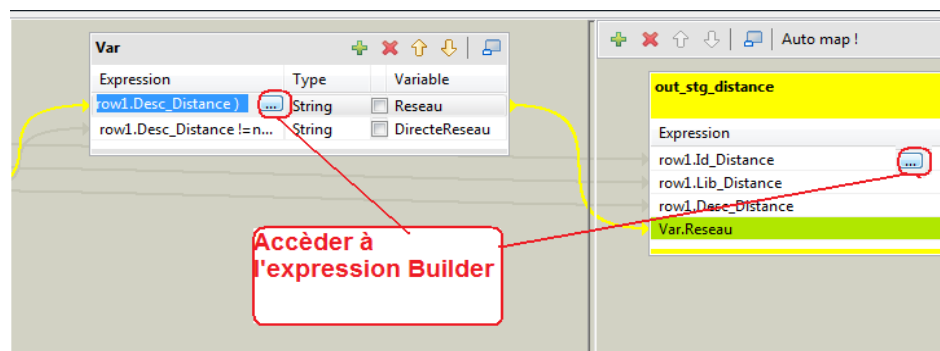
Ex : **row1.Desc_Distance !=null ? routine_dwh.TypeReseau(row1.Desc_Distance) : "INCONNU"**

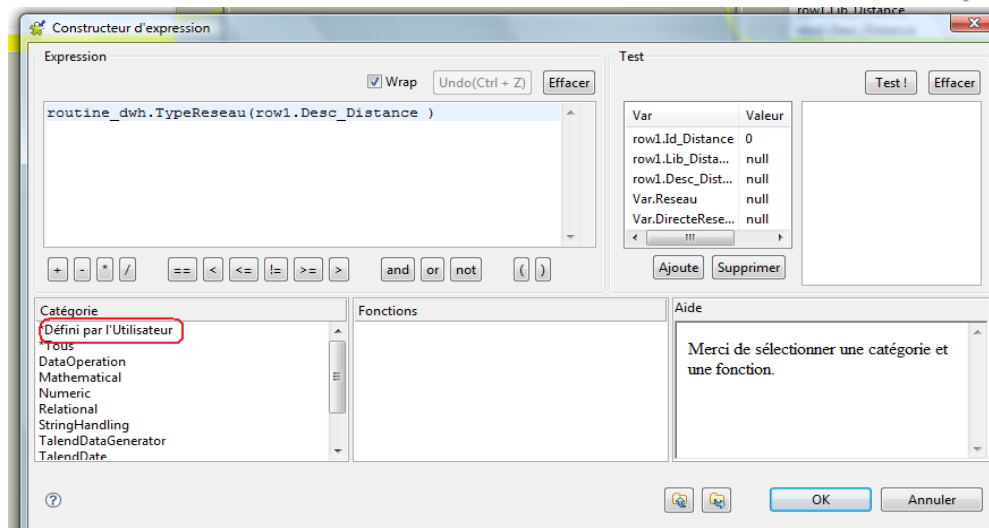
- La routine s'appelle routine_dwh
- La méthode (fonction) s'appelle **TypeReseau ()**
- Elle prend une chaîne de caractère en paramètre (Description de distance) et retourne une autre chaîne de caractère , type de réseau (GSM etc.)



NB : La routine doit être bien commenté avec la JavaDoc (commentaire en bleu qui commence par `/**`) pour qu'elle soit accessible via CTRL+Espace (ou dans le builder d'expression, via le tMap

Accéder à « **l'expression Builder** » ou « **Constructeur d'expression** »





Résultats attendus :

Les cardinalités doivent être :

SOURCE		CIBLES		
STG_REF_DISTANCE		DWH_REF_DISTANCE		
Lues	Filtrées	Supprimées	Mises à jour	Insérées
6	0	0	0	6

Deux lignes de type GSM, trois de type FIXE, et dernier de type INCONNU.

3.3.3 Charger les offres

Comme précédemment, on transfère les données existantes et on crée une nouvelle colonne. On va également utiliser une routine pour implémenter la règle de gestion

Exemple de routine :

```
/**
 * @Author: Boudejma Larid
 *
 * TypeReseau:  retourne le type de reseau , .
 *
 *
 * {talendTypes} String
 *
 * {Category} User Defined
 *
 * {param} int(2) input: The string need to be printed.
 *
 * {example} TypeOffre(2) # PREPAID !.
 * @return: PREPAID si IdOffre commence par 2,POSTPAID si IdOffre commence par 5,INCONNU sinon
 *
 */
public static String  TypeOffre(int IdOffre) {

    String Ret_Str= "INCONNU"; //

    if (String.valueOf(IdOffre).startsWith("2")){
        Ret_Str = "PREPAID";
    }else if (String.valueOf(IdOffre).startsWith("5")){
        Ret_Str = "POSTPAID";
    }

    return Ret_Str;
}
```

Résultats attendus :

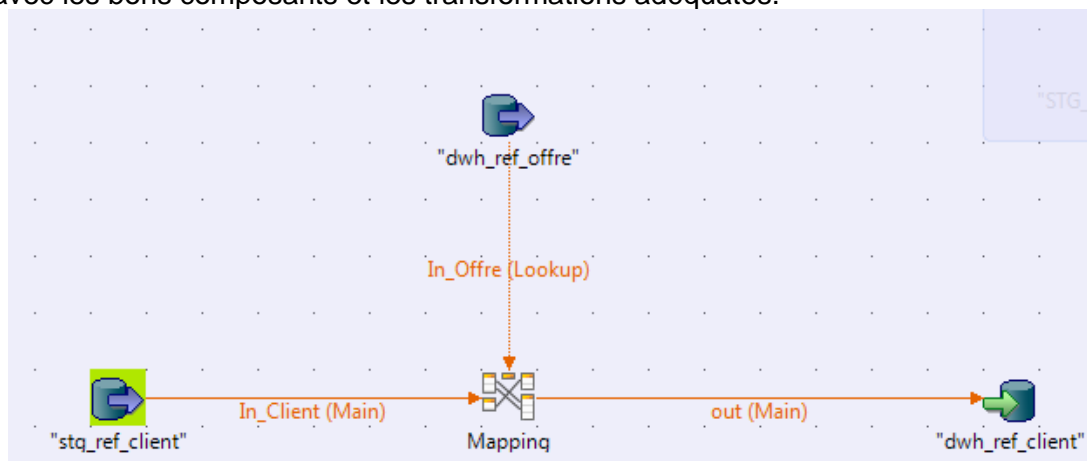
Les cardinalités doivent être :

SOURCE		CIBLES		
STG_REF_OFFRE		DWH_REF_OFFRE		
Lues	Filtrées	Supprimées	Mises à jour	Insérées
9	0	0	0	9

Trois lignes de type PREPAID, trois de type POSTPAID et trois de type INCONNU.

3.3.4 Charger les clients

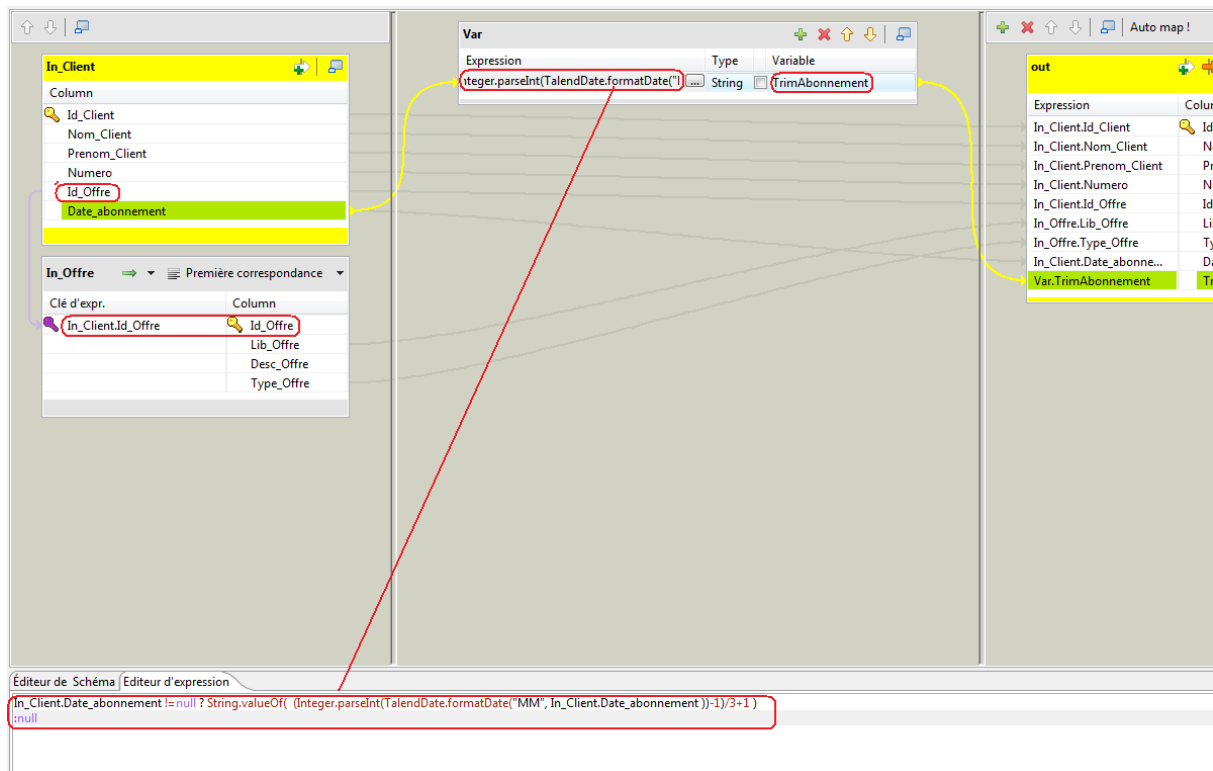
Vous devez réaliser jointure (lookup) avec de table « dwh_ref_offre » pour récupérer, entre autres, le libellé de l'offre. Vous pouvez dupliquer le « Charger_Stg_Client », le renommer en « Charger_Dwh_Client », puis l'adapter avec les bons composants et les transformations adéquates.



Une solution pour le calcul du trimestre est :

À l'intérieur du tMap, créez une variable temporaire « TrimAbonnement » qui sera assignée par le résultat de la transformation ci-dessous.

In_Client.Date_abonnement != null ? String.valueOf((Integer.parseInt(TalendDate.formatDate("MM", In_Client.Date_abonnement))-1)/3+1) : null



Résultats attendus :

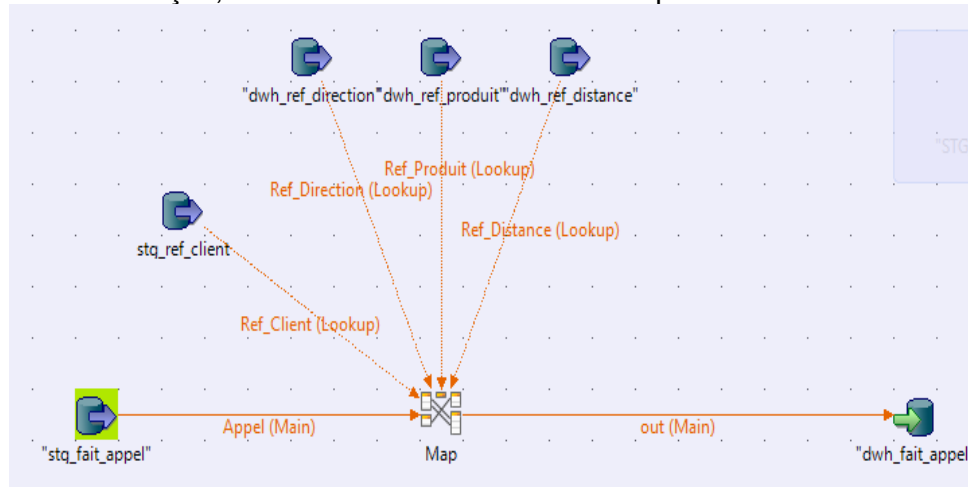
Les cardinalités doivent être :

SOURCE		CIBLES		
STG_REF_CLIENT		DWH_REF_CLIENT		
Lues	Filtrées	Supprimées	Mises à jour	Insérées
21	0	0	0	21

Septe lignes de type PREPAID et onze de type POSTPAID et Trois INCONNU.

3.3.5 Charger les appels

De la même façon, réalisez ce Job à l'instar de la copie écran ci-dessous.



Pour le Champ Pays, vous pouvez développer une routine (une méthode dans la même routine précédemment développée) et l'appeler dans le tMap. En voici un exemple :

```
public static String PaysCall(int IdDistance, String NumeroTelephone) {

    String Ret_Str= "France"; //

    if (IdDistance == 5 && NumeroTelephone!=null ){

        if (NumeroTelephone.trim().matches("30.*")){
            Ret_Str = "Grèce";
        }
        if (NumeroTelephone.trim().matches("33.*")){
            Ret_Str = "France";
        }
        if (NumeroTelephone.trim().matches("352.*")){
            Ret_Str = "Luxembourg";
        }
        if (NumeroTelephone.trim().matches("377.*")){
            Ret_Str = "Monaco";
        }
        if (NumeroTelephone.trim().matches("41.*")){
            Ret_Str = "Suisse";
        }
        if (NumeroTelephone.trim().matches("44.*")){
            Ret_Str = "Royaume Uni";
        }
        if (NumeroTelephone.trim().matches("45.*")){
            Ret_Str = "Danemark";
        }
        if (NumeroTelephone.trim().matches("49.*")){
            Ret_Str = "Allemagne";
        }

        if (Ret_Str.equals("France")){ // Si Autre, i.e Ret_Str n'a pas été changé
            Ret_Str = "Autres";
        }

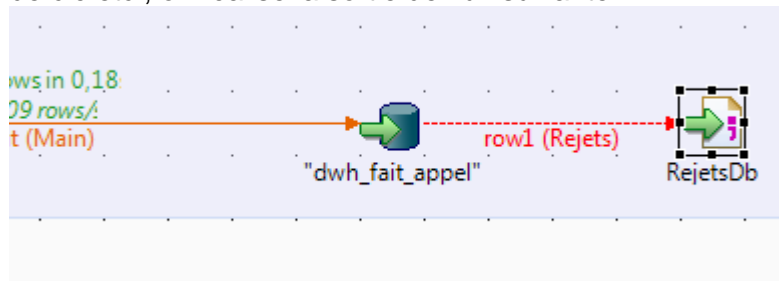
    }

    return Ret_Str;
}
```

Remarque :

Avec Talend, en mode ETL, On peut réaliser des jointure avec des tables de Différents schéma, et même, de différentes base

Pour récupérer les éventuels rejets de base de donnée, si par exemple il y a doublon de clé etc., on réalise la sortie de flux suivante :



Résultats attendus :

Les cardinalités doivent être :

SOURCE		CIBLES		
STG_FAIT_APPEL		DWH_FAIT_APPEL		
Lues	Filtrées	Supprimées	Mises à jour	Insérées
135	0	0	0	135
		RJT_APPEL		
		Supprimées	Mises à jour	Insérées
		0	0	0
		ALT_APPEL		
		Supprimées	Mises à jour	Insérées
		0	0	0

3.3.6 Réalisation du Job Sequencer ou du batch d'intégration

A l'instar du job de séquençement de l'extraction vers la STG, mettez en avant les possibilités de parallélisme dans l'enchaînement des jobs.

3.4 Chargement des agrégats

Cette partie est distincte de celle de l'intégration car l'espace source en est distinct. Les agrégats seront compris dans les calculs de type Datamarts.

Remarque : Encore une fois, il existe plusieurs solutions afin d'obtenir le bon résultat.

3.4.1 Charger les agrégats mensuels groupés sur les distances

Un des critères d'agrégation est le mois d'appel, on a besoin de le calculer à partir de la date d'appel.

➤ Trouvez une solution de votre choix !

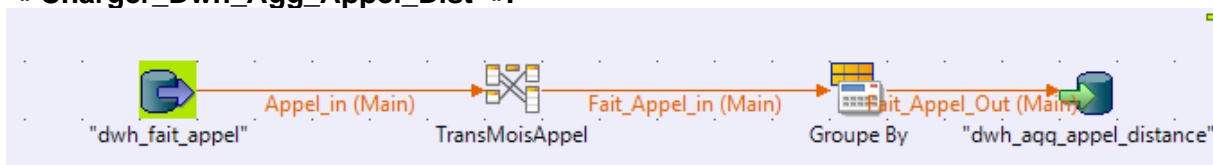
Voici un exemple en utilisant le tMap. Cependant, n'oubliez pas que le tMap est très consommateur de ressource.

Appel_in.Date_appel != null ? TalendDate.formatDate("MM", Appel_in.Date_appel) : null

➤ D'autres Solutions possibles :

- ✓ Transformation dans la requête, plus efficace mais nécessite le changement du schéma
- ✓ Utiliser un composant moins lourd que le tMap (ex. tJavaRow etc), pour calculer le mois.

Prenez le composant tAggregateRow et réalisez le Job
« **Charger_Dwh_Agg_Appel_Dist** ».



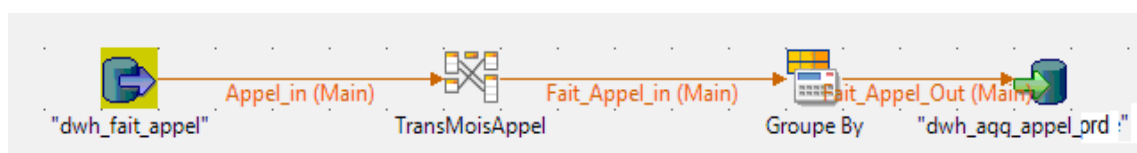
Résultats attendus :

Les cardinalités doivent être :

SOURCE		CIBLES		
STG_FAIT_APPEL		DWH_FAIT_APPEL		
Lues	Filtrées	Supprimées	Mises à jour	Insérées
135	0	0	0	33

3.4.2 Charger les agrégats mensuels groupés sur les produits :

Dupliquez le Job « **Charger_Dwh_Agg_Appel_Dist** », renommez le en « **Charger_Dwh_Agg_Appel_Prd** » et changez la cible avec celle appropriée.



Résultats attendus :

Les cardinalités doivent être :

SOURCE		CIBLES		
STG_FAIT_APPEL		DWH_FAIT_APPEL		
Lues	Filtrées	Supprimées	Mises à jour	Insérées
135	0	0	0	33

Remarque : Le composant stage tAggregateRow n'est à utiliser que pour des volumétries raisonnables, autrement ses performances seront faibles.

3.4.3 Réalisation du Job Sequencer ou du batch d'agrégation

Démarche similaire à ce qui a été développé dans le cadre de l'insertion dans le DWH.

NB :

A la fin de votre TP, Vous pouvez importer le Zip du Projet « **TalendFormationProject.zip** », c'est un exemple de développement des Jobs du TP.