# TP3 Correction - statistiques de température

la première partie de ce TP consiste à générer des données pour un ensemble de capteurs, ces données seront stockées dans des fichiers csv avec ces colonnes [*temperature,datetime*], pour cela nous allons générer une liste de noms de capteurs et nous allons itérer sur eux pour générer les données de chaque capteur.

## 1. Génération des noms de capteurs

Créer d'un nouvel Job "*tp3_datageneration*", Sur l'espace de travail, ajouter un composant de type *tRowGenerator*, sur l'interface de paramétrage du composant, ajouter une colonne avec le nom "*nom_capteur"* de type String et sur la colonne Functions sélectionner les trois points, sur l'onglet paramètres de la fonction :

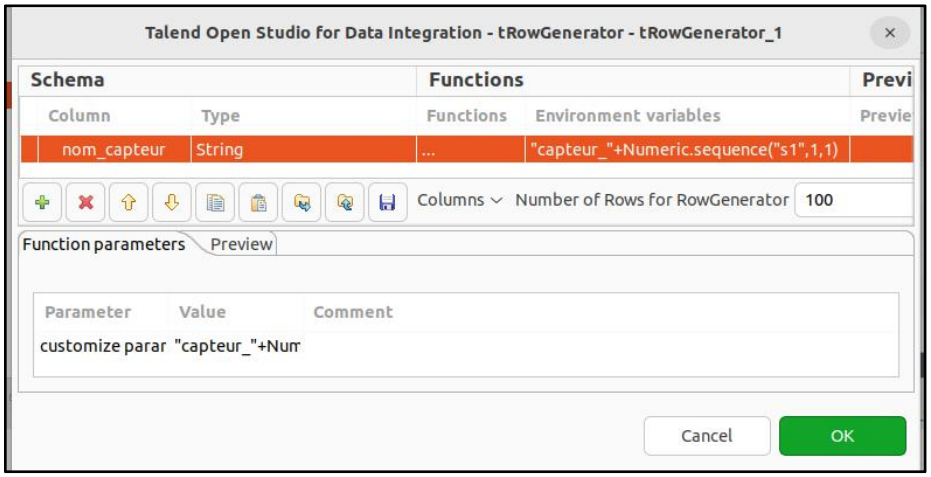


Image 1

Dans l'interface de paramétrage des fonctions, choisir la fonction *sequence* dans la catégorie Numeric et concaténer celle-ci avec la chaîne "*capteur_*".

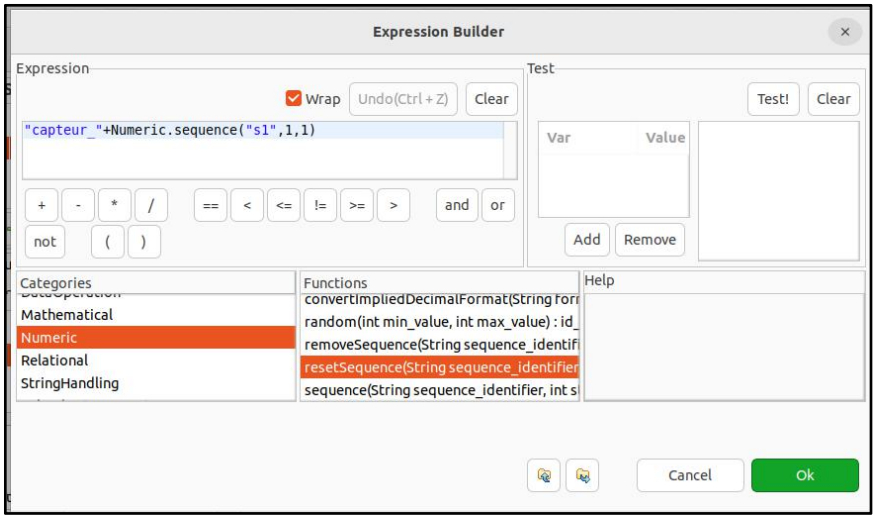

Image 2

## 2. Itérer sur chaque nom de capteur

Ajout d'un nouveau composant de type *tFlowToIterate* pour itérer sur chaque nom de capteur généré par l'étape précédente

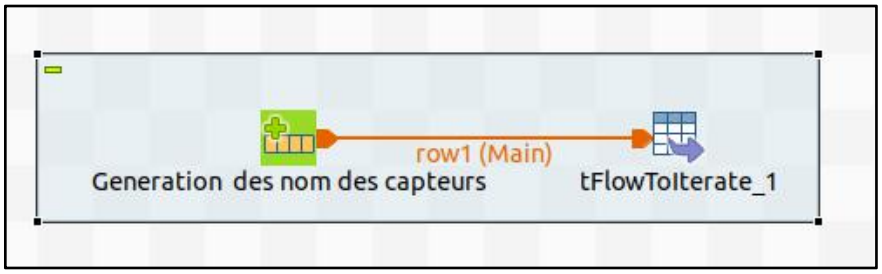

Image 3

Notez que le nom du capteur est stocké dans la variable globale globalMap accessible avec l'expression "*globalMap.get("row1.nom_capteur"))*"

## 3. Génération de données des capteurs

Ajouter un nouveau composant de type *tRowGenerator*, dans les paramètres du composant ajouter deux colonnes, la première est la "temperature" de type Integer avec la fonction *Numeric.Random* et les paramètres min value -20 max value 100, la seconde est "*datetime*" de type Date avec la fonction *TalendDate.getRandomDate* et les paramètres min "2022-01-01" et max "2022-12-31".

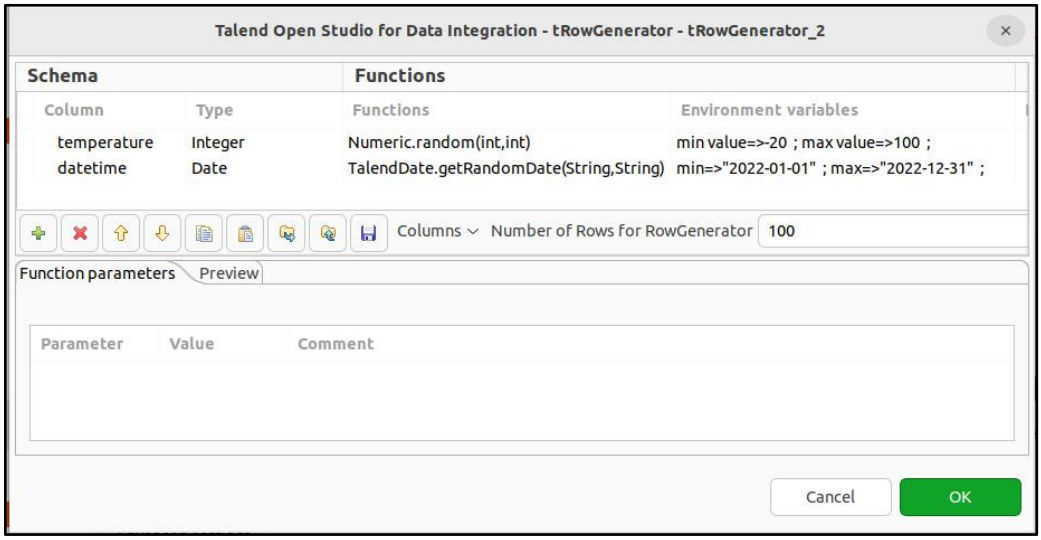

Image 4

## 4. Tri les valeurs générées

Ajouter un nouveau composant de type *tSortRow* pour trier les valeurs générées par date, avec les paramètres suivants:

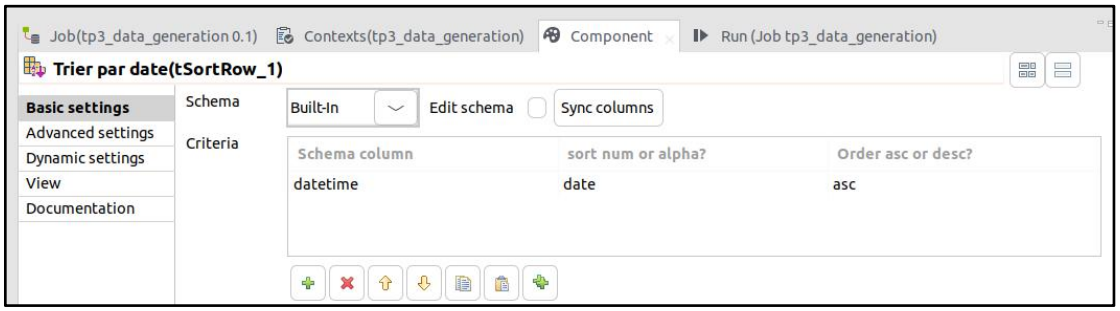

Image 5

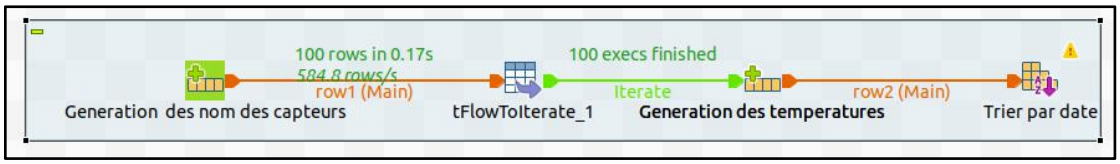

Image 6

## 5. Sauvegarder les données sur des fichiers csv

Ajouter un nouveau contexte avec le nom de "*tp3_context*" et ajouter une variable avec le nom "*output_dir"* de type Directory et choisir un répertoire où vous voulez sauvegarder les fichiers générés

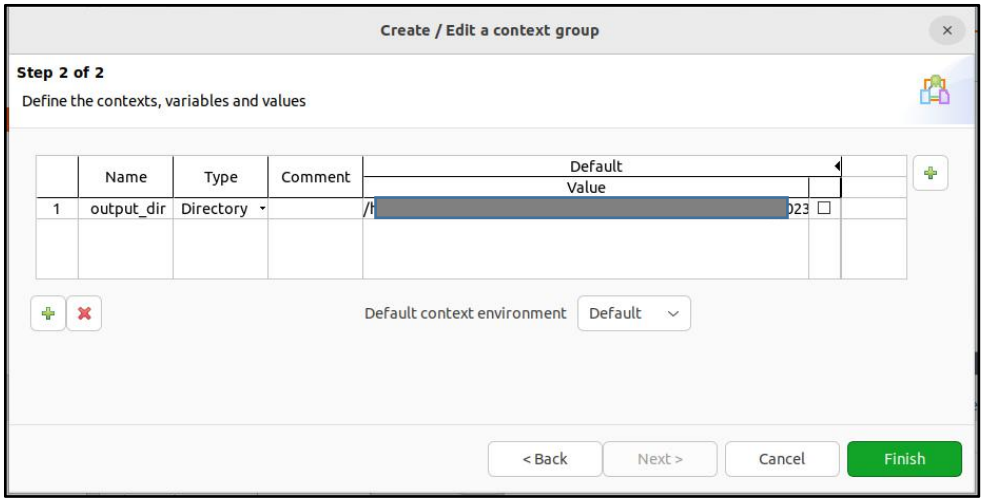


Image 7

Sur l'espace de travail, dans l'onglet contextuel, ajoutez la variable *output_dir* du *tp3_context*.

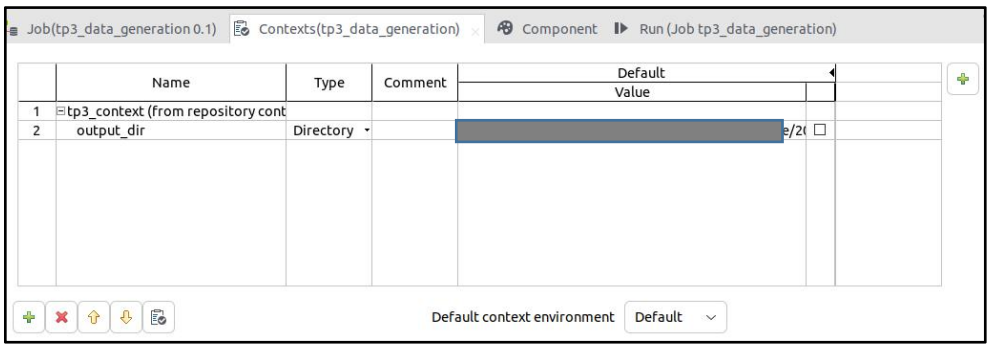


Image 8

Ajouter un nouveau composant de type *tFileOutputDelimited* avec les paramètres suivants:

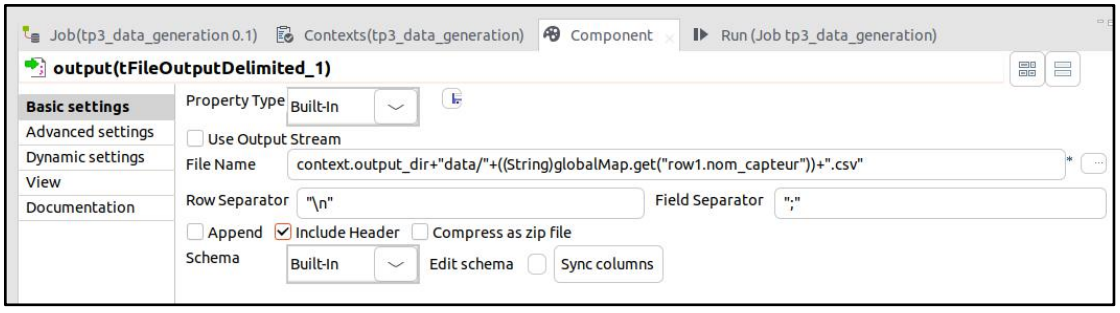


Image 9

Notez que le nom du fichier est généré par l'expression suivante :

*context.output_dir+"data/"+((String)globalMap.get("row1.nom_capteur"))+".csv"*

où *context.output_dir* est une variable qui contient un répertoire et *globalMap.get("row1.nom_capteur"))* contient le nom du capteur.

à la fin, vous devez avoir le travail suivant:

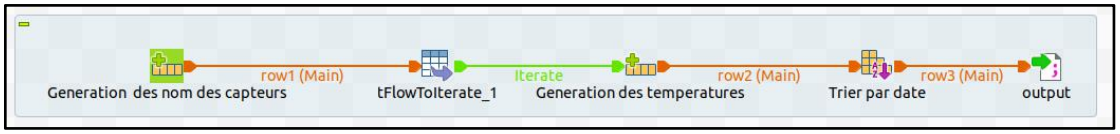


Image 10

Le but de la seconde partie du TP est de nettoyer et de calculer certaines statistiques (max, min et moyenne) à partir des données déjà générées, et de sauvegarder ces données dans un fichier csv, ainsi que de sauvegarder toutes les données aberrantes dans un second fichier csv.

Pour mettre en œuvre ce Job, nous allons suivre les étapes suivantes: tout d'abord nous allons itérer sur tous les fichiers dans le répertoire output_dir et nous allons filtrer les valeurs de chaque fichier selon les critères suivants (-10 < temperature < 70) après cela, nous allons calculer les statistiques et les enregistrer dans un fichier csv, pour les valeurs aberrantes, nous allons les obtenir à partir des valeurs rejetées sur l'étape de filtrage et simplement les enregistrer dans un fichier csv.

## 6. Lecture des fichiers de capteurs

Créer d'un nouvel Job "*tp3*". Sur l'espace de travail, dans l'onglet contextuel, ajoutez la variable *output_dir* du *tp3_context* et ajouter un nouveau composant de type *tFileList* avec le paramètre suivant:

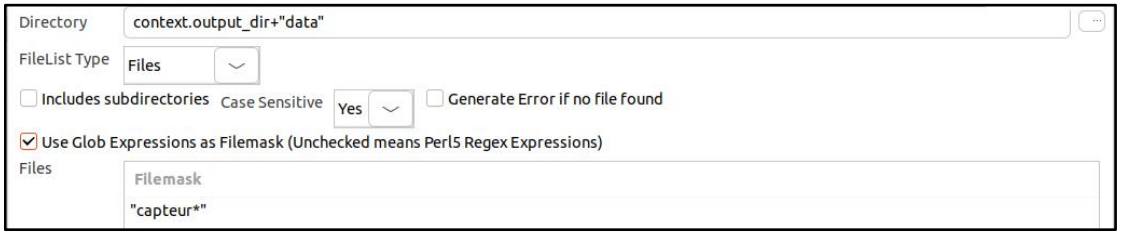


Image 11

Notez que nous avons filtré les fichiers à lire pour ne retenir que ceux qui commencent par "*capteur*".

Ajouter une métadonnée de type "*file Delimited*" nommé "*tp3_capteur*" à partir d'un des fichiers générés:

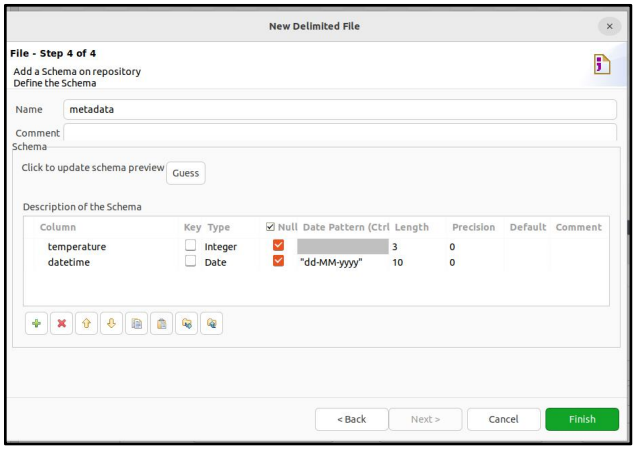


Image 12

Ajouter un nouveau composant de type *tFileOutputDelimited* avec le paramètre suivant:

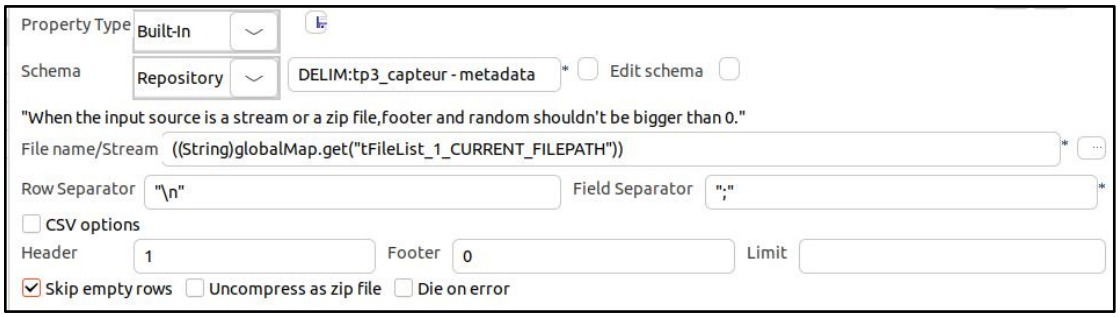


Image 13

Notez que nous avons lu le nom du fichier à partir de la variable globale *CURRENT_FILEPATH* générée par le composant tfileList.

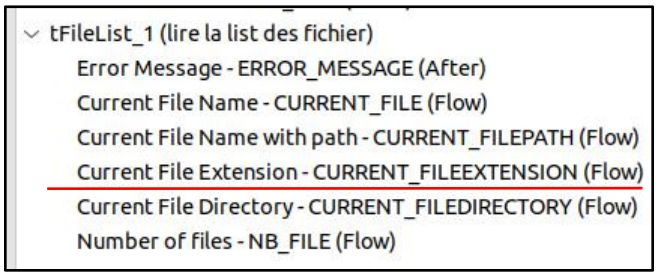


Image 14


Image 15

# 7. Filtrer les données

Tout d'abord nous allons ajouter une nouvelle colonne nommée "*capteur*" et l'initialiser avec le nom du capteur, lu à partir du nom du fichier, pour cela nous allons ajouter un nouveau composant de type *tJavaRow* avec la configuration suivante:

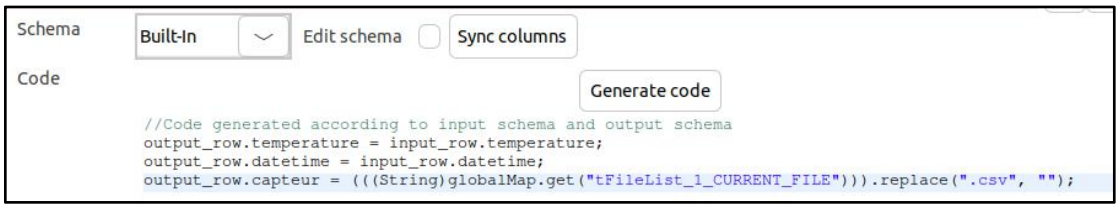


Image 16

et ajouter la nouvelle colonne "*capteur*"sur le schéma de sortie:

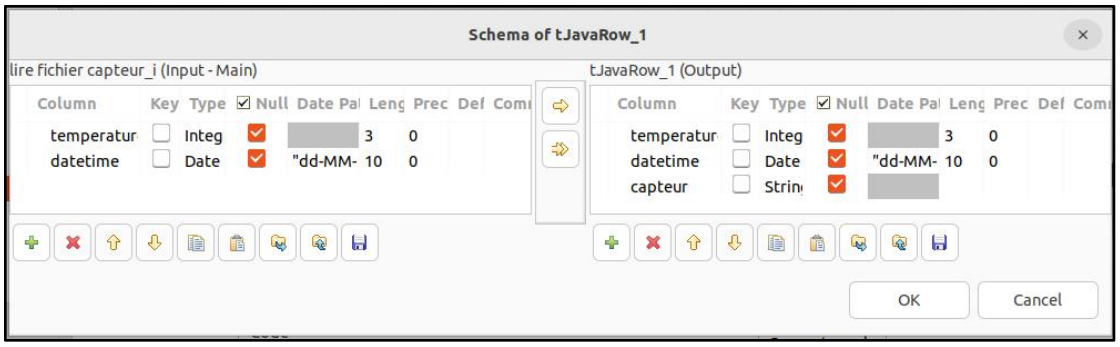


Image 17

Ajouter un nouveau composant de type *tFilterRow* avec le paramètre suivant:

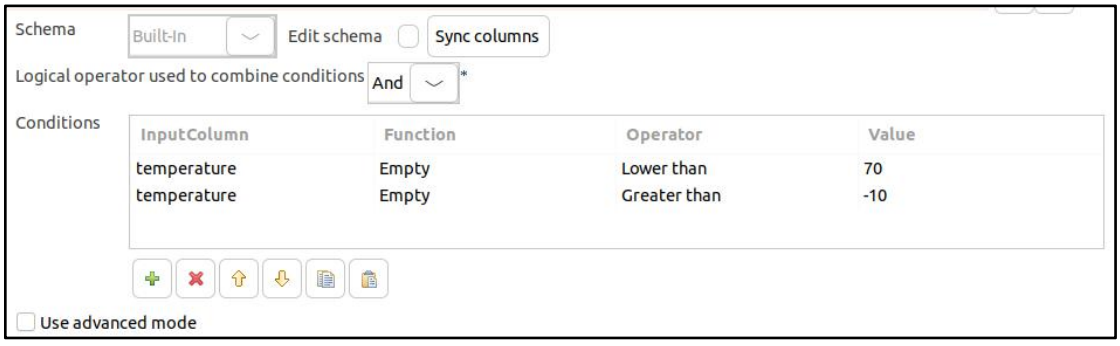


Image 18

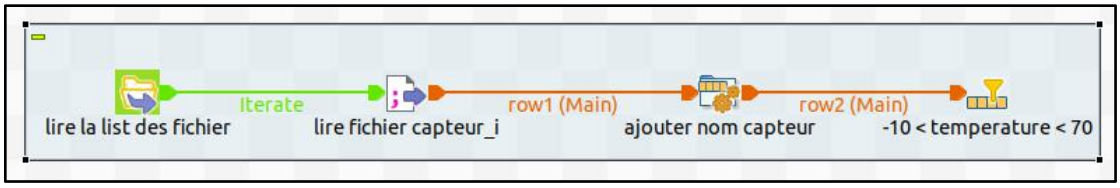


Image 19

## 8. Calcul des statistiques

Ajouter un nouveau composant de type *tAggregationRow* et le connecter à l'étape de filtrage par la connexion "*Filter*" et ajouter les colonnes *min max avg* au schéma de sortie:

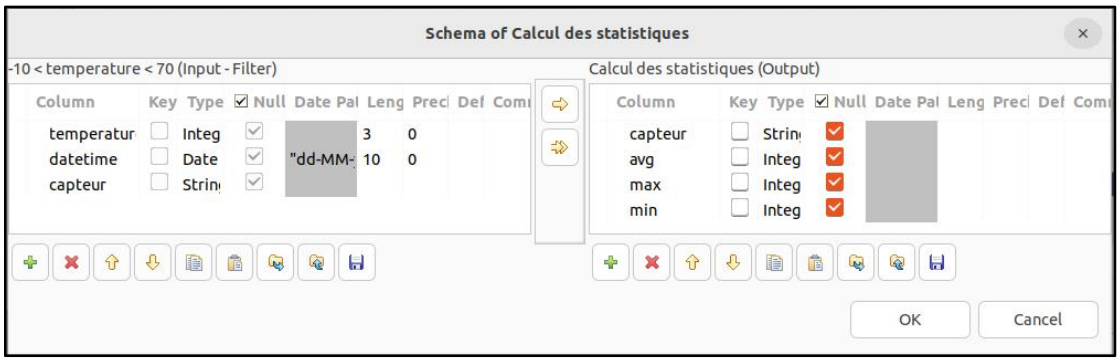


Image 20

Ajoutez la configuration suivante au neveau composant :

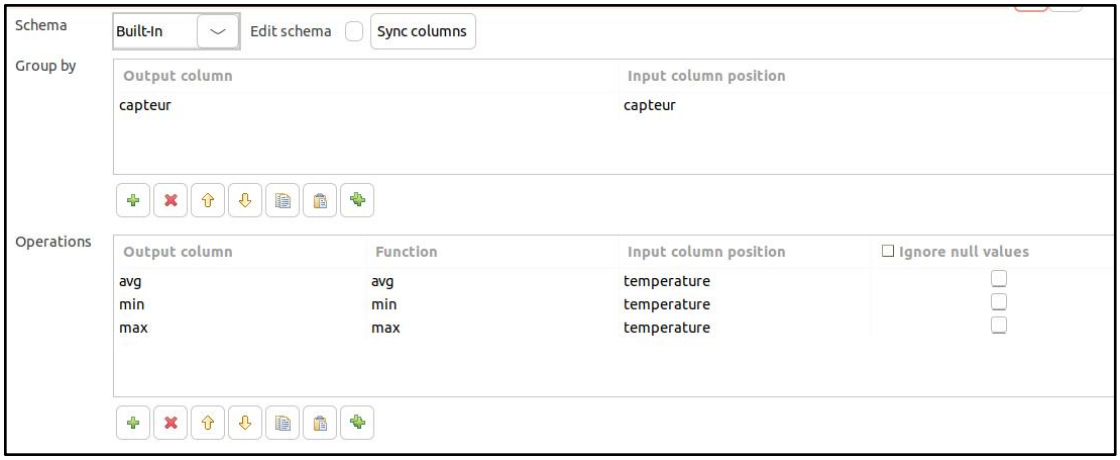


Image 21

Ajouter un nouveau composant de type *tFileDelimited* avec la configuration suivante:


Image 22

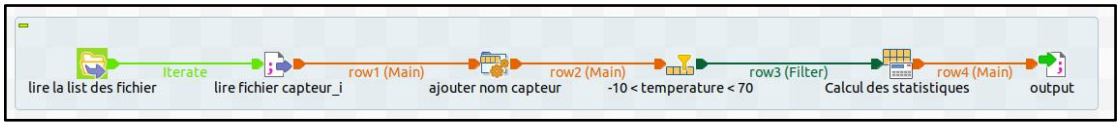


Image 23

## 9. Sauvegarder les valeurs aberrantes

Ajouter un nouveau composant de type *tFileDelimited* et le connecter à l'étape de filtrage par la connexion "*Reject*" avec la configuration suivante:

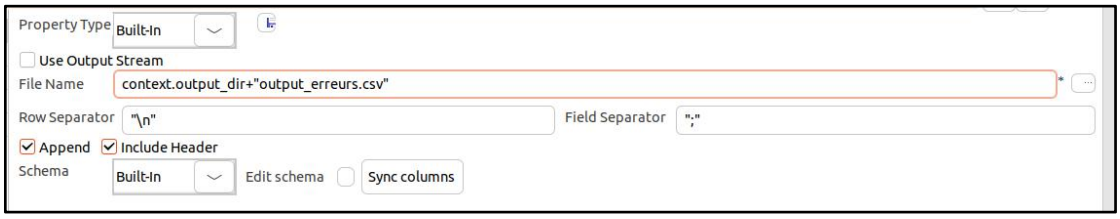


Image 24

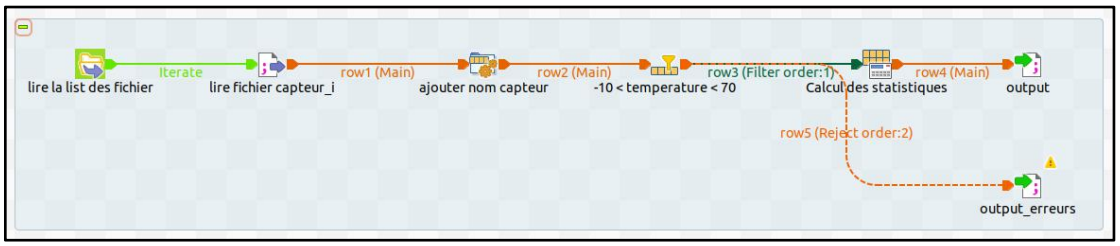


Image 25

## 10. Afficher le message "job terminé"

Ajouter un nouveau composant de type *tMsgBox* et relié à la *tFileList* par la connexion du trigger "*OnSubjobOk*" avec la configuration suivante:


Image 26

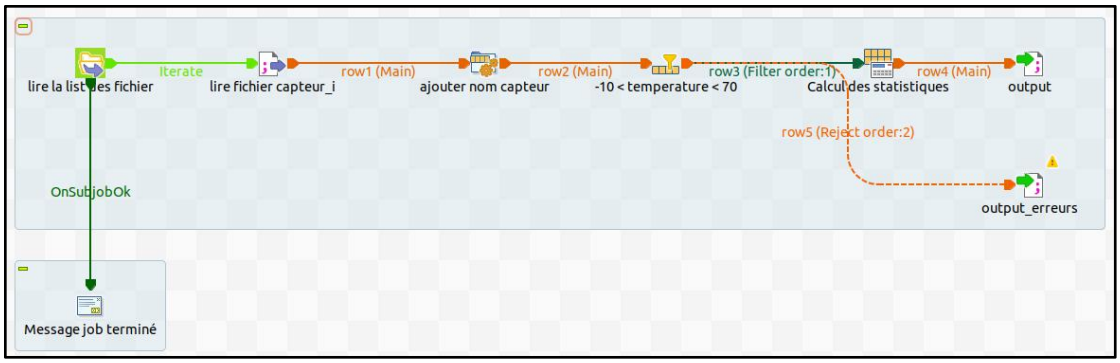


Image 27

# 11. Calcul les statistiques des valeurs aberrantes

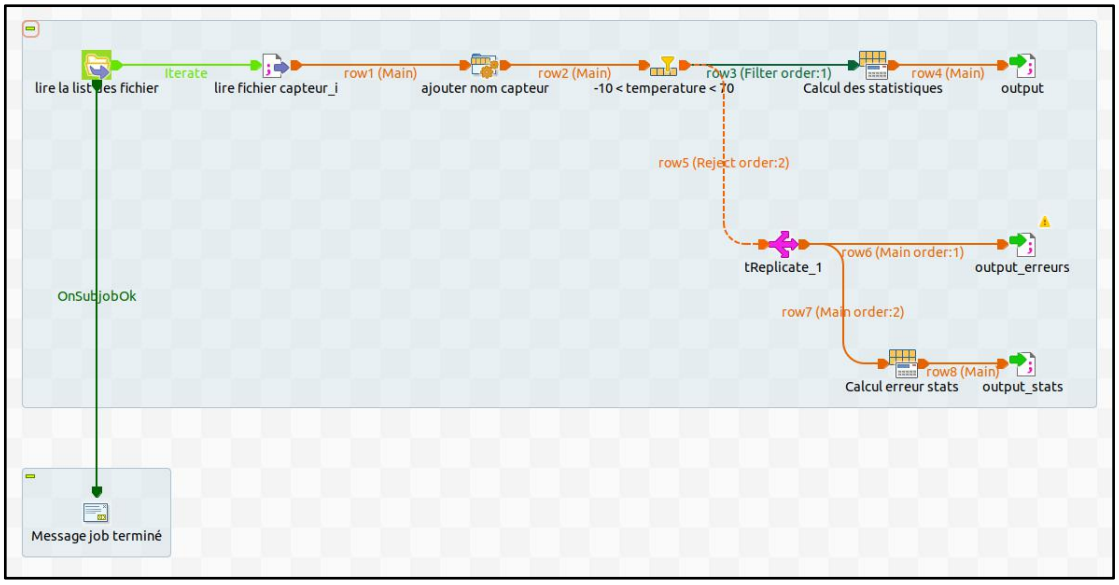


Image 28