POLYTECH INTL
المدرسة الدولية الخاصة للتقنيات

DOCUMENTATION DU PROCESSUS ETL AVEC SSIS

Assyl Ghayeza

ANNÉE UNIVERSITAIRE : 2024-2025

SOMMAIRE

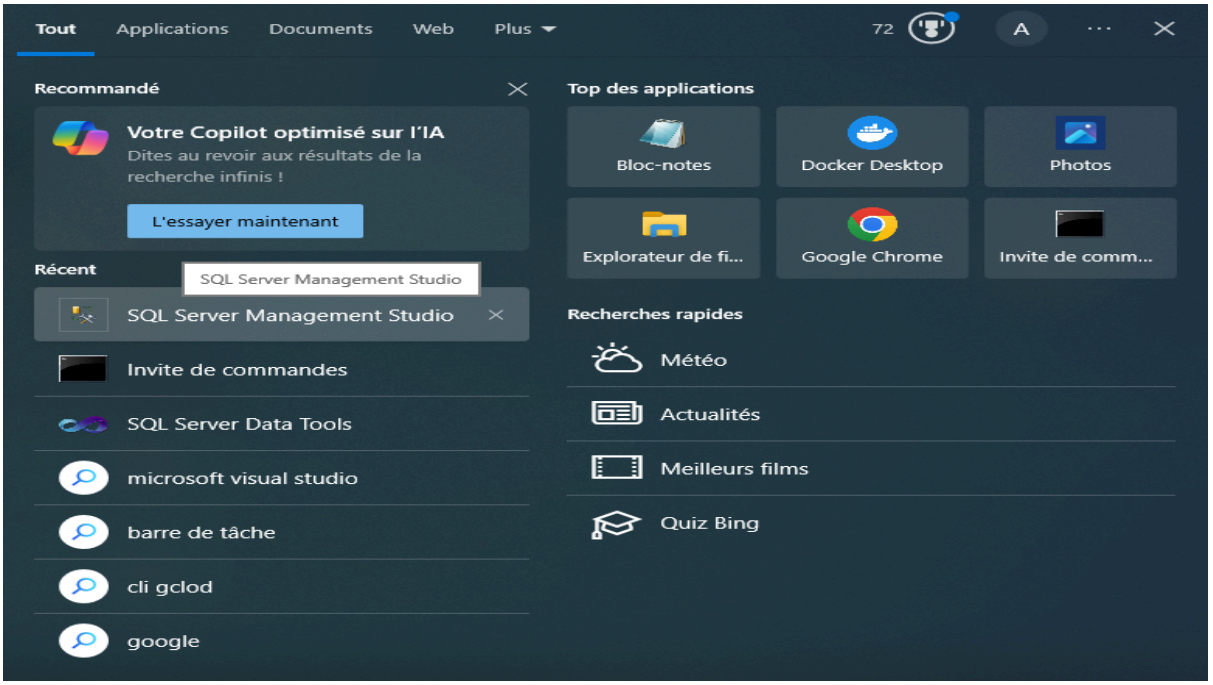
I- Finalité du Processus

L’objectif de ce processus SSIS est de centraliser plusieurs fichiers CSV dans une table, de nettoyer les données, puis de les exporter dans un nouveau fichier CSV et de les transmettre automatiquement par e-mail, tout en garantissant une exécution reproductible et fiable.

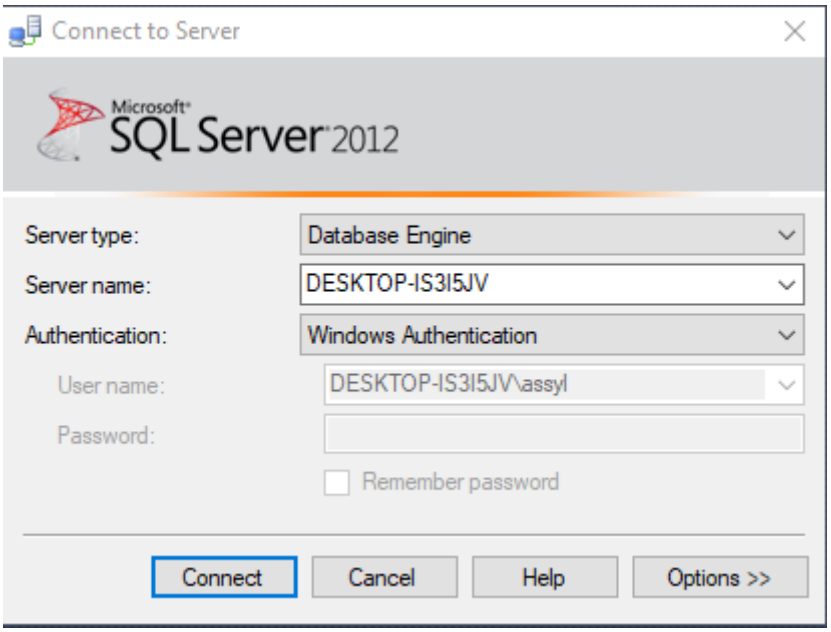
II. Création de la Base de Données

Avant de débuter avec SQL Server Integration Services (SSIS) pour l'extraction, la transformation et le chargement des données, il est important de créer une base de données destinée à accueillir ces données. Dans ce processus SSIS, une table nommée *Client* sera utilisée pour stocker les informations des clients. Voici les étapes pour créer la base de données et la table correspondante à l'aide de SQL Server Management Studio (SSMS).

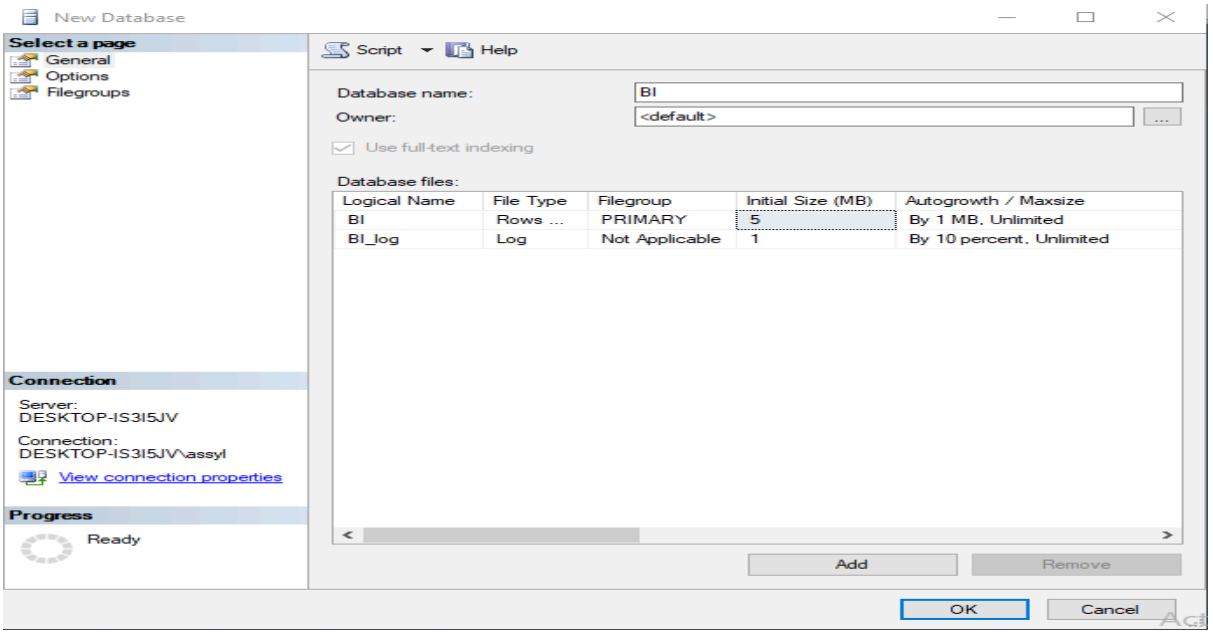
1. Ouvrir SQL Server Management Studio (SSMS) :
- Assurez-vous d'avoir installé SQL Server Management Studio.
- Lancez SSMS en recherchant **SQL Server Management Studio** dans le menu Démarrer ou en utilisant toute autre méthode d'accès.



2. Connexion au Serveur SQL :
- Une fois SSMS ouvert, connectez-vous à votre serveur SQL. Vous devrez fournir les informations de connexion appropriées.



3. Création de la base de données :
- Dans ***Object Explorer*** (panneau de gauche), développez le nœud correspondant à votre serveur SQL.
- Faites un clic droit sur **Databases**, puis sélectionnez **New Database**.
4.Dans la boîte de dialogue qui s'ouvre, donnez un nom à votre base de données. Par exemple, nommez-la BI



5. Création du Table :
- Soit d'écrire un script SQL en développant la fenêtre **NewQuery** pour créer la table
- Soit dans **Object Explorer**, développez le nœud de votre nouvelle base de données BI.
- Cliquez avec le bouton droit sur le dossier **Tables** et sélectionnez **New Table .**

6. Conception de la table Student **:**

-Dans *Object Explorer*, développez le nœud de votre nouvelle base de données et accédez à la table **Student**.

-Faites un clic droit sur la table, puis sélectionnez **Design**.

-Dans la vue de conception, une grille vide apparaîtra avec les en-têtes **Column Name**, **Data Type**, et **Allow Nulls**.

-Pour ajouter une colonne, cliquez dans la première ligne vide sous **Column Name** et saisissez le nom de la colonne.

-Dans la colonne **Data Type**, sélectionnez le type de données correspondant à la nature des données que la colonne stockera (par exemple : `varchar`, `int`, `varchar(max)`, etc.).

-Dans la colonne **Allow Nulls**, indiquez si la colonne peut contenir des valeurs nulles.

-Répétez ces étapes pour ajouter toutes les colonnes nécessaires à la table **Student**, en veillant à ce que la structure corresponde aux données du fichier CSV.

-Si votre table doit comporter des clés primaires ou d'autres contraintes, vous pouvez les définir dans cette vue de conception en utilisant les options de SSMS. Par exemple, pour définir une colonne comme clé primaire, faites un clic droit sur la colonne et sélectionnez **Set Primary Key**.

7. Définition de Valeurs par Défaut pour des Colonnes : Dans le cadre de la conception de la table "Client," il est nécessaire de définir des

valeurs par défaut pour certaines colonnes. Voici comment définir des valeurs par défaut pour les colonnes spécifiques :

- ID : Dans la partie **Column Properties**, développez la section **Identity Specification** changer le champ (**Is Identity**) en **Yes**

- FLAG : Dans **Default Value or Binding** de la boîte **Column Proprieties** mettez la valeur 0

- DEVELOPPEUR : Pour la colonne DEVELOPPEUR, vous pouvez définir la valeur par défaut comme le nom de l'utilisateur actuel en utilisant la fonction **SUSER_SNAME()**

- Date_Integration : Définir une valeur par défaut en utilisant la fonction **"CONVERT(varchar,GETDATE(),(112))"** ou tout autre format de date requis. 6 8. Enregistrement de la Table : Une fois que vous avez conçu la table, enregistrez-la en utilisant File>Save ou en utilisant la combinaison de touches Ctrl + S.

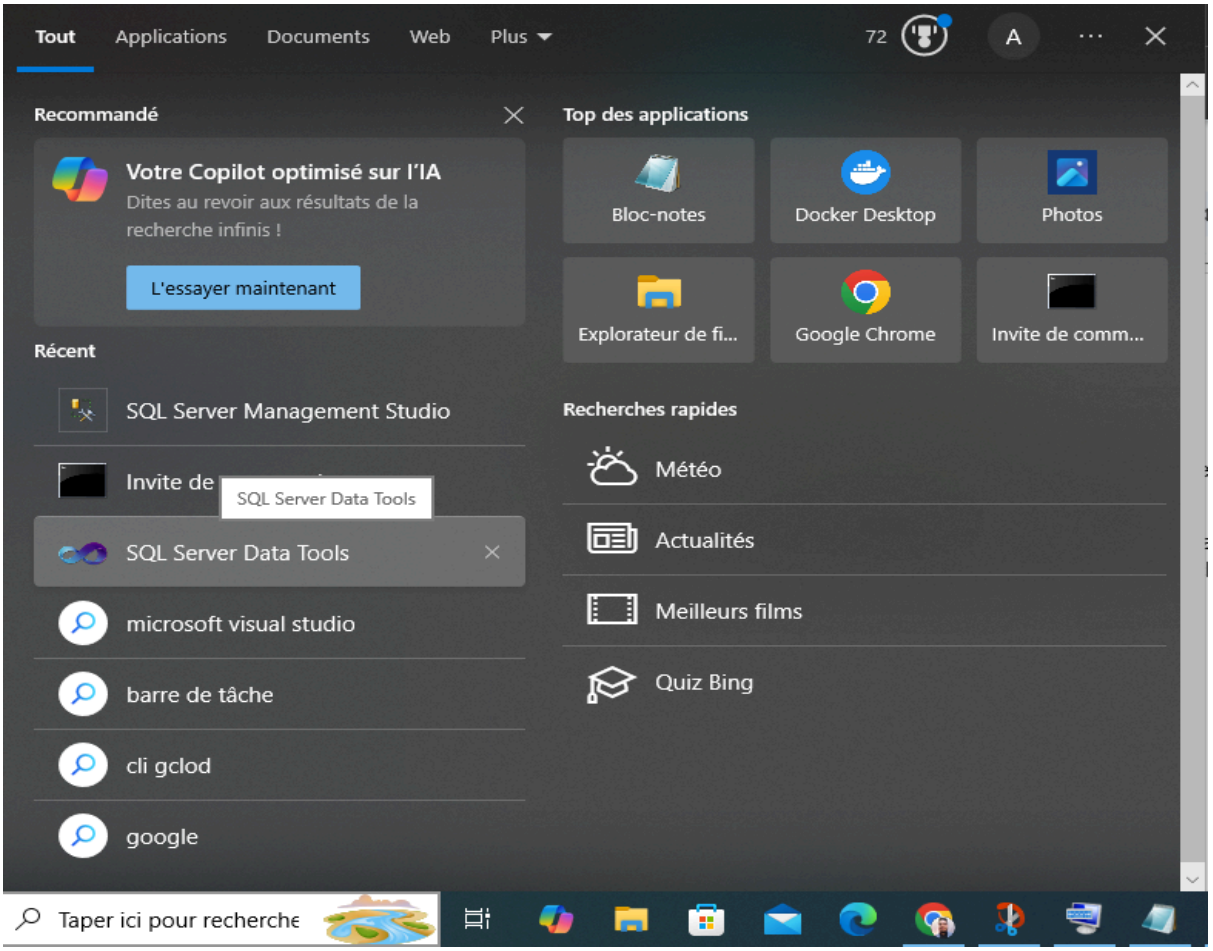
8. Enregistrement de la Table : Une fois que vous avez conçu la table, enregistrez-la en utilisant File>Save ou en utilisant la combinaison de touches **Ctrl + S.**

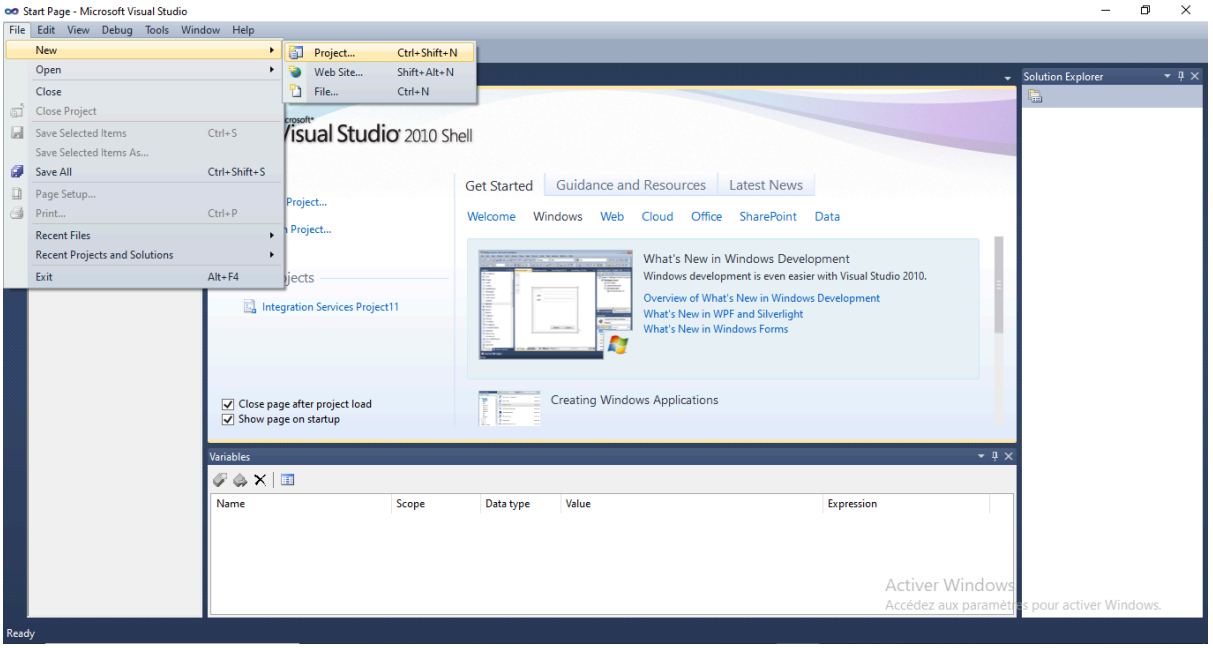
III. Création d'un nouveau Projet SSIS

Pour créer un nouveau projet SSIS, commencez par ouvrir l'outil SQL Server Data Tools (SSDT). Voici les étapes à suivre pour le lancer :

1. Cliquez sur le bouton **"Démarrer"** de votre ordinateur.
2. Saisissez **"SQL Server Data Tools"** ou **"SSDT"** dans la barre de recherche, puis appuyez sur **"Entrée"** pour effectuer la recherche.
3. Vous devriez voir une entrée correspondant à **"SQL Server Data Tools"** dans les résultats de la recherche. Cliquez sur l'entrée correspondant pour la lancer.

4. Dans SQL Server Data Tools vous avez deux option soit dans la page de démarrage cliquer sur New Project soit sélectionnez **File> New> Project**



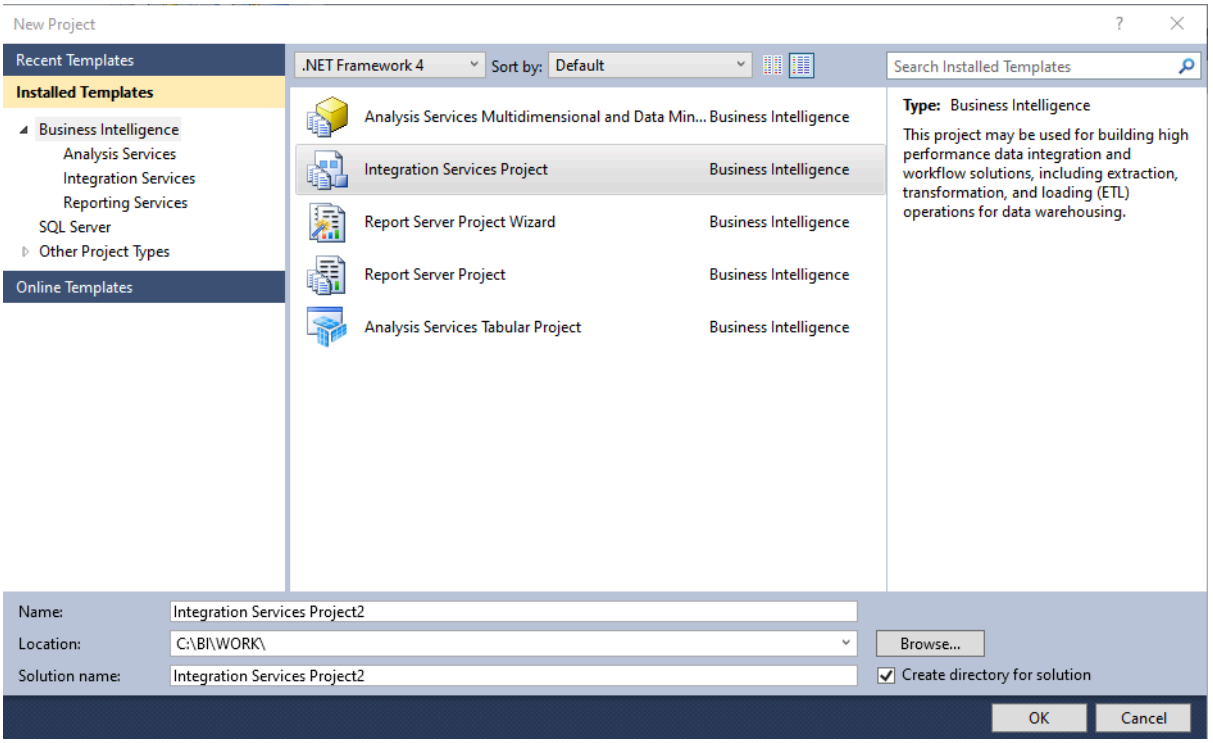


5. Dans la boîte de dialogue **New Project**, développez le nœud **Business Intelligence** sous **Installed Templates**, puis sélectionnez **Integration Services**.

6. Dans le champ **Name**, saisissez le nom souhaité pour votre projet. Si vous souhaitez utiliser un dossier existant, décochez l'option **Create directory for solution**.

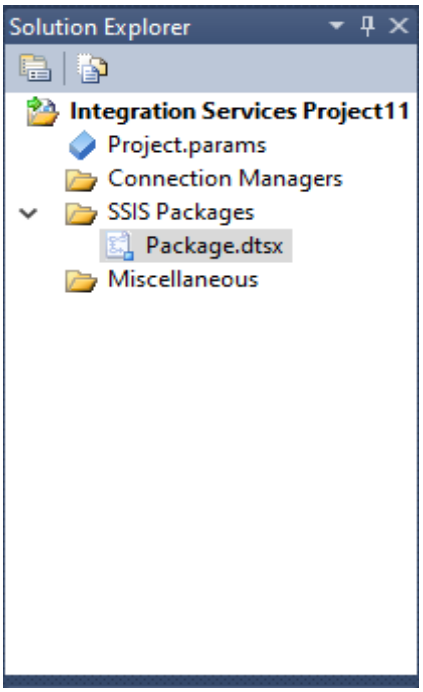
7. Conservez l'emplacement par défaut ou cliquez sur **Browse** pour choisir un dossier spécifique. Dans la boîte de dialogue **Project Location**, sélectionnez le dossier souhaité.

8. Cliquez sur **OK** pour valider.



9. Par défaut, un package vide nommé **Package.dtsx** est créé et ajouté à votre projet sous **SSIS Packages**.

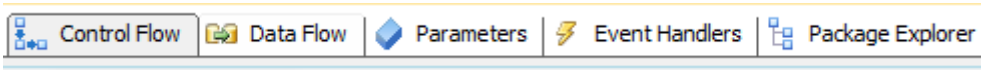
10. Dans **Solutions Explorer** , faites un clic droit sur **Package.dtsx**, sélectionnez **Renommer**, puis attribuez-lui le nom souhaité.



IV. Différentes fonctionnalités de l'interface de SQL Server Integration Services (SSIS)

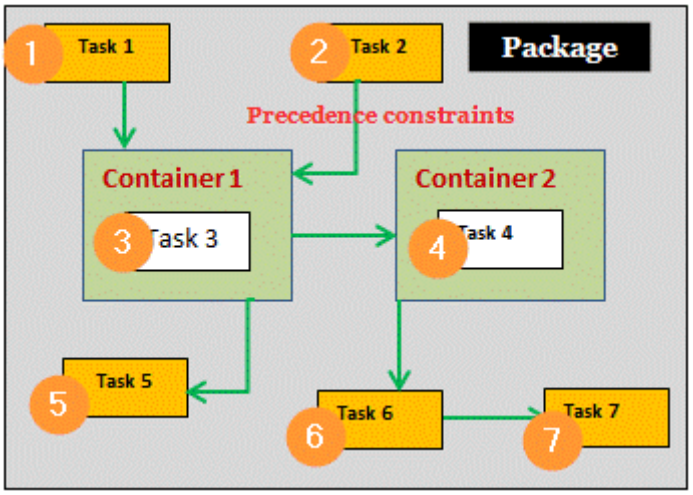
Une fois le projet SSIS créé, il est important de bien comprendre l'interface de SQL Server Integration Services (SSIS). Cette section vous présentera les principaux éléments de l'interface afin de vous permettre de travailler efficacement sur vos packages SSIS.

Le SSIS Designer propose de nombreuses fonctionnalités et vous permet d'exécuter les tâches suivantes :



1. Control Flow Interface :
   Un package se compose d'un flux de contrôle et, éventuellement, d'un ou plusieurs flux de données. SQL Server Integration Services propose trois types d'éléments de flux de contrôle : les conteneurs, qui fournissent des structures dans les packages , les tâches, qui assurent les fonctionnalités , et les contraintes de priorité, qui relient les exécutables, les conteneurs et les tâches dans un flux de contrôle organisé.
   Le diagramme suivant illustre deux flux de contrôle comportant deux conteneurs et sept tâches . Cinq de ces tâches sont définies au niveau du package, tandis que deux tâches sont définies au niveau du conteneur, étant situés à l'intérieur de celui-ci.

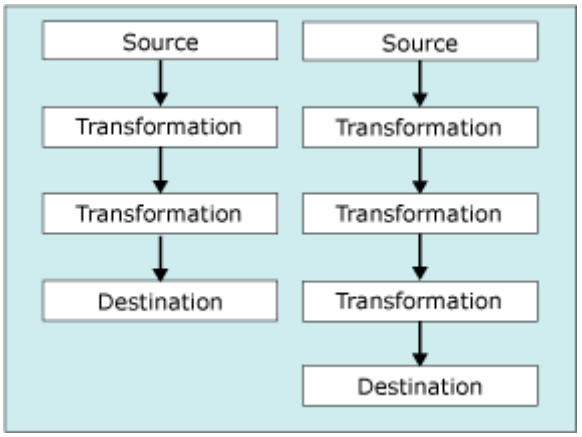
   L'architecture d'Integration Services permet l'imbrication des conteneurs, de sorte qu'un flux de contrôle peut comporter plusieurs niveaux de conteneurs imbriqués. Par exemple, un package peut inclure un conteneur, comme un conteneur **Foreach Loop**, qui peut à son tour contenir un autre conteneur **Foreach Loop**, et ainsi de suite.



2. Data Flow :

SQL Server Integration Services propose trois types de composants pour le flux de données : les sources, les transformations et les destinations.

- Les sources extraient les données à partir de divers magasins de données, tels que des tables et vues dans des bases de données relationnelles, des fichiers, et des bases de données des services d'analyse.
- Les transformations modifient, résument ou nettoient les données.
- Les destinations chargent les données dans des magasins de données ou créent des ensembles de données en mémoire.
  De plus, Integration Services fournit des chemins qui relient la sortie d'un composant à l'entrée d'un autre. Ces chemins définissent l'ordre des composants et permettent d'ajouter des annotations au flux de données ou de visualiser la source des colonnes.
  Le diagramme suivant illustre un flux de données comprenant une source, une ou plusieurs transformations avec une entrée et une sortie, ainsi qu'une destination.



3. Paramètres :

- Dans les outils de données SQL Server, vous pouvez créer, modifier ou supprimer les paramètres du projet via la fenêtre **Project.params**. Les paramètres permettent d'assigner des valeurs aux propriétés des packages lors de leur exécution.

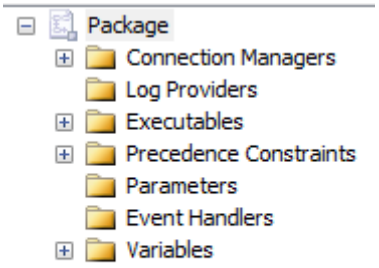
4. Event Handlers :

- Lors de l'exécution, les exécutables (tels que les packages, Foreach Loop, For Loop, Séquence et les conteneurs d'hôtes de tâches) déclenchent des événements. Par exemple, un événement **OnError** se déclenche lorsqu'une erreur survient. Vous avez la possibilité de créer des gestionnaires d'événements personnalisés pour ces événements, ce qui permet d'étendre les fonctionnalités des packages et de faciliter leur gestion pendant l'exécution.

5. Package Explorer :

Dans SSIS Designer, l'onglet **Package Explorer** offre une vue permettant d'explorer le package. Cette vue reflète la hiérarchie des conteneurs dans l'architecture d'Integration Services. Le conteneur de paquets se situe au sommet de la hiérarchie, et vous pouvez développer le package pour afficher les connexions, exécutables, gestionnaires d'événements, fournisseurs de journaux, contraintes de priorité et variables du package.

Le diagramme suivant présente une vue arborescente d'un package simple.



V. Processus D'Exécution

1. Création d’un Data Flow Task

Le **Data Flow Task** encapsule le moteur de flux de données, qui gère le transfert des données entre les sources et les destinations. Ajouter une tâche de flux de données à un package permet d'extraire, de transformer et de charger des données.

Un **Data Flow Task** comprend au moins un composant de flux de données, mais il s'agit généralement d'un ensemble de composants de flux de données interconnectés.

Ajouter Data Flow Task

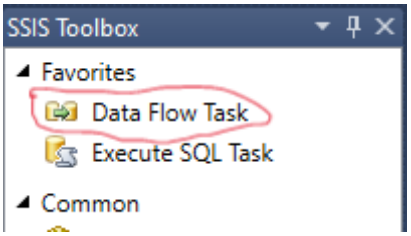
1. Dans Outils de données SQL Server (SSDT), ouvrez le projet Integration Services qui contient le package souhaité.

2. Dans l'Explorateur de Solutions, double-cliquez sur votre package pour l'ouvrir.

3. Cliquez sur le **Control Flow** onglet.

4. Dans le **SSIS Toolbox** volet, développer **Favorites**, et faire glisser un **Data Flow Task** sur la surface de conception de la **Control Flow** onglet.

5. Sur le **Control Flow** concevoir la surface, cliquez avec le bouton droit sur le nouveau **Data Flow**, sélectionnez **Rename** pour changer le nom par défaut.



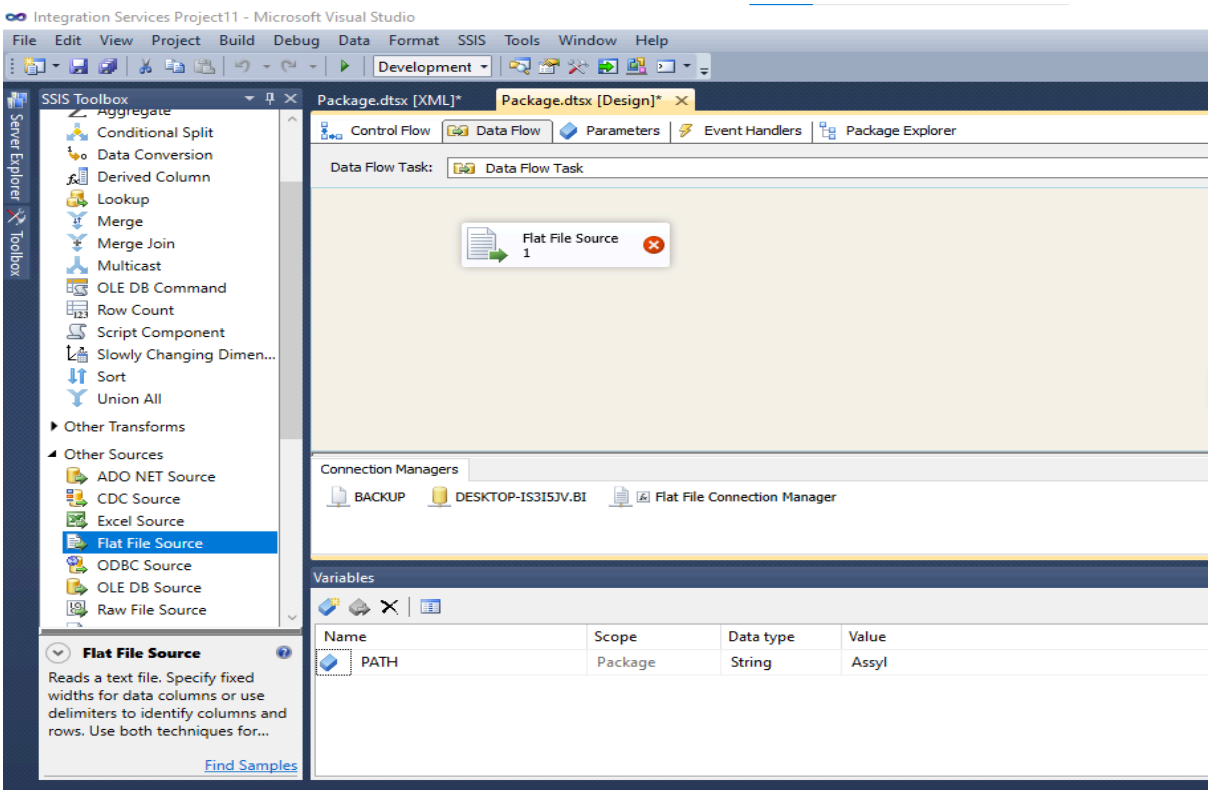
2. Flat File source

Dans notre processus SSIS, les données sources proviennent de fichiers CSV (Comma-Separated Values). Pour extraire ces données et les intégrer dans notre flux de données SSIS, nous utilisons le composant **"Flat File Source"**. Ce composant nous permet de lire le contenu des fichiers CSV et de le traiter dans nos packages SSIS.

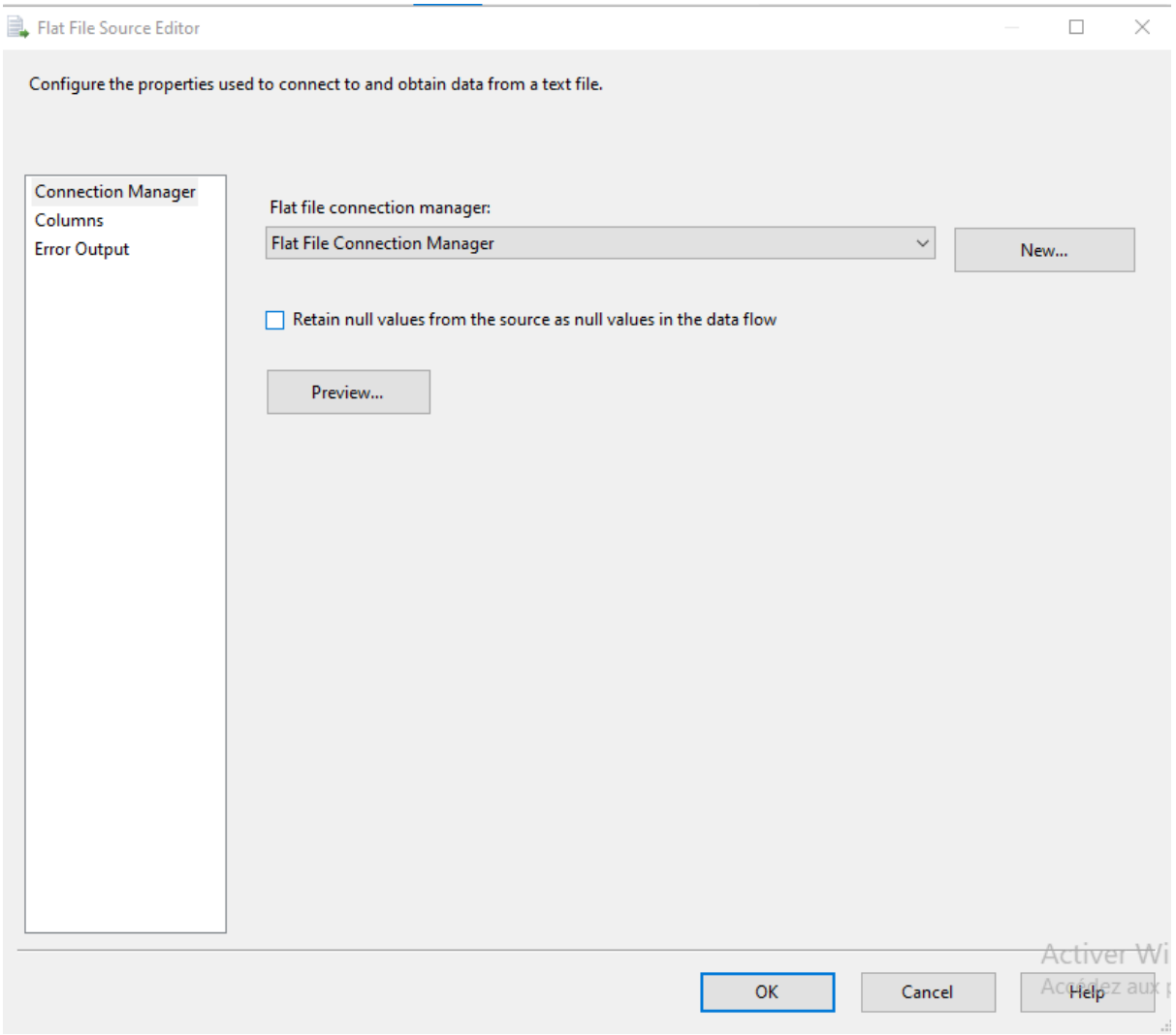
Pour cette tâche, nous devons configurer la source du fichier CSV à utiliser comme suit :

1. Pour ouvrir le **Data Flow designer**, double-cliquez sur le composant **Data Flow Task**, ou sélectionnez le **Data Flow onglet**.

2. Dans le **SSIS Toolbox**, développez **Other Sources**, puis faites glisser un **Flat File Source** sur la surface de conception de la **Data Flow** onglet.



3. Double-cliquez sur la **Flat File Source** pour ouvrir le **Flat File Source Editor**.



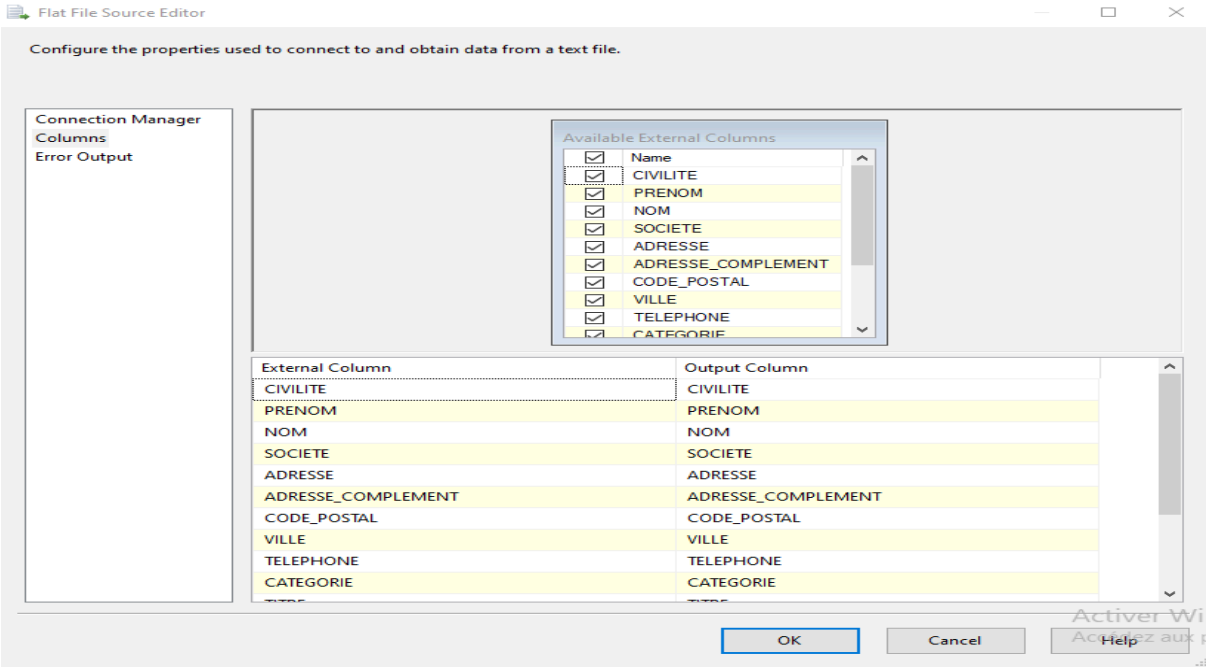
4. Dans le **Flat file connection manager** champ, cliquez sur **New** pour ouvrir le **Flat File Connection Manager Editor** ou sélectionner une connexion existante si vous en avez déjà configuré une.
5. Dans L'onglet General du Flat File Connection Manager Editor vous trouvez les champs suivants :
   - Connection manager name : C'est le nom de la connexion par défaut pré rempli avec le nom « Flat File Connection Manager »
   - Description : Champ facultatif pour ajouter une description de la source de données.
   - File name : Sélectionnez le fichier CSV en cliquant sur **Browse**.

   Les autres champs sont souvent automatiquement pré remplis en

fonction du fichier sélectionné, mais vous pouvez les ajuster si nécessaire.

- Local : Il s'agit de la configuration de la locale. Vous pouvez choisir la locale qui correspond à la langue, ce champ peut être pré rempli en fonction de la locale détectée dans le fichier.
- Code Page : Détermine comment les caractères sont encodés dans le fichier.
- Format : Choisissez **Delimited** pour les fichiers CSV, car les données sont séparées par des délimiteurs.
- Text qualifier : Spécifiez le délimiteur de texte, si nécessaire, pour les données entre guillemets, généralement utilisé pour des valeurs textuelles spéciales.
- Header row delimited : Permet de spécifier un délimiteur pour la ligne d'en-tête dans un fichier CSV.
- Header row skip : Indique combien de lignes à ignorer avant que les données réelles commencent. Par défaut, c'est souvent 0.
- Column name in the first data row : Cochez si le nom de colonne est inclus dans la première ligne de données.

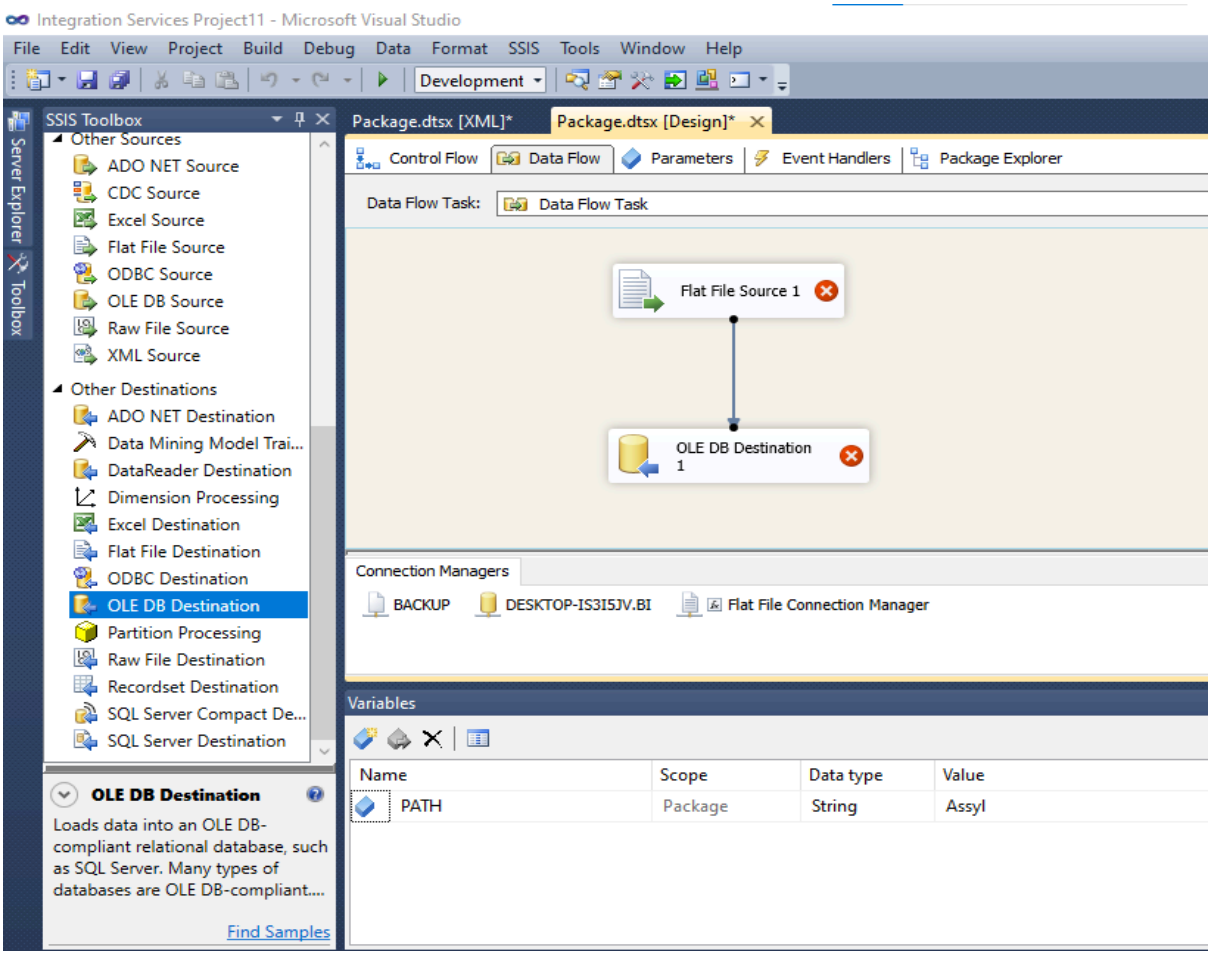
6. Sélectionner **Columns** et vérifiez que les noms des colonnes sont corrects. Dans L'onglet Columns tu peux changer le délimiteur des lignes et des colonnes.
7. Sélectionner **OK**.



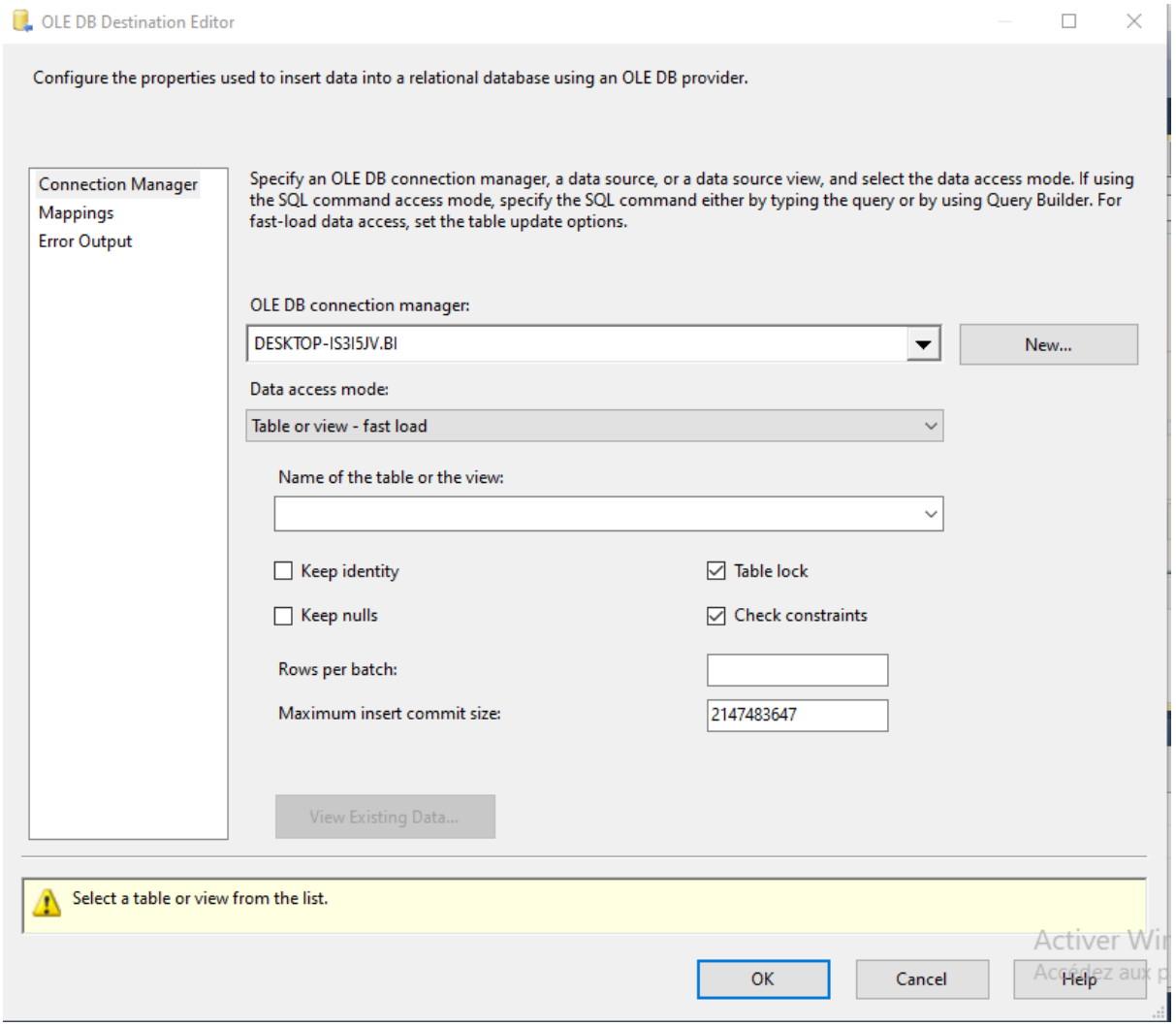
3. OLE DB Destination

Notre package est maintenant capable d'extraire les données depuis la source **Flat File**. L'étape suivante consiste à charger ces données dans une destination. Pour ce faire, ajoutez le composant **OLE DB Destination** au flux de données. Ce composant permet de charger les données dans une table de base de données, une vue ou via une commande SQL, et prend en charge une variété de bases de données compatibles avec OLE DB.

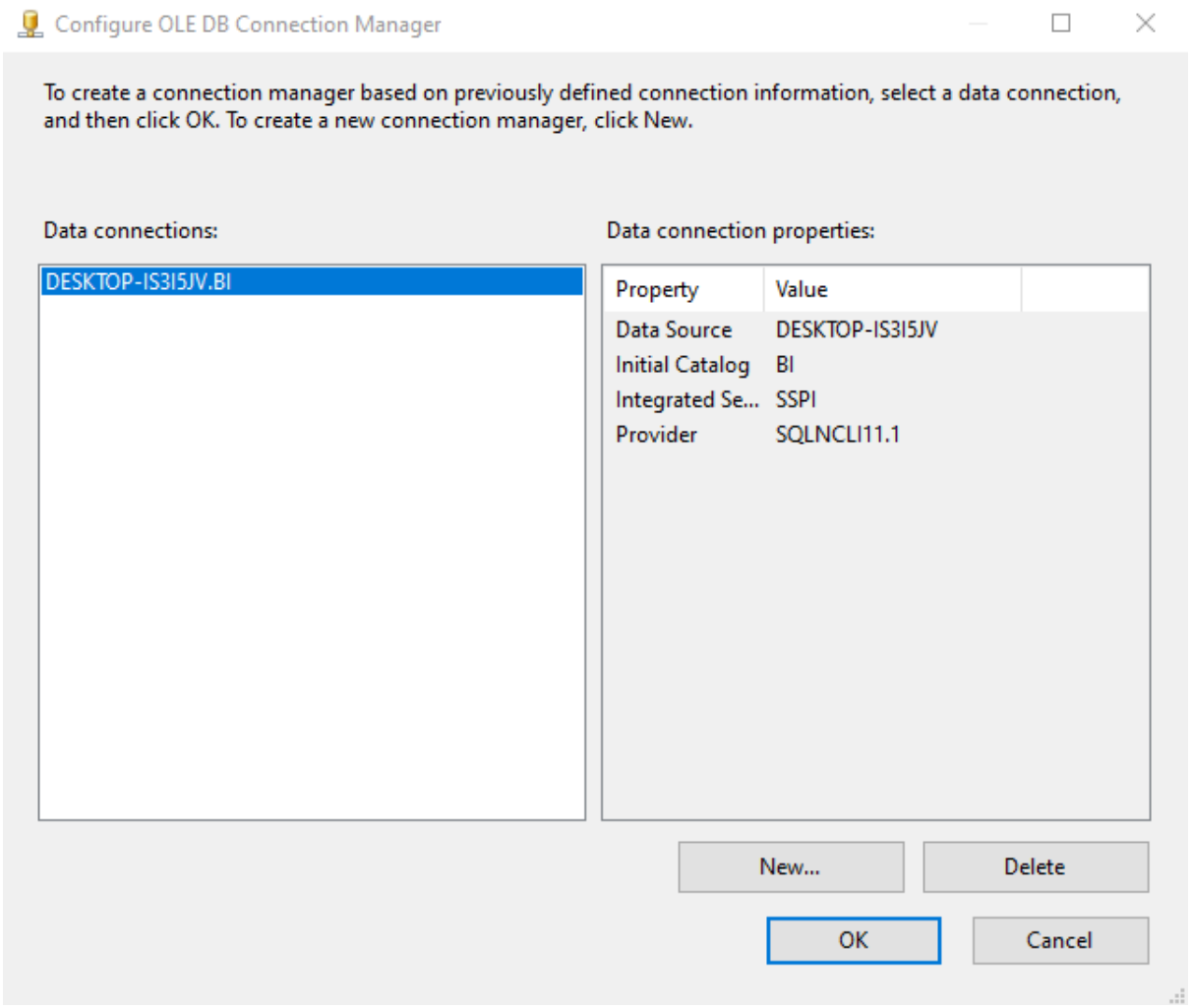
1. Dans le **SSIS Toolbox**, développez Other Destinations, et faites glisser **OLE DB Destination** sur la surface de conception **Data Flow**. Placez **OLE DB Destination** directement en dessous du **Flat File Source**
2. Sélectionnez **Flat File Source** et faites glisser sa flèche bleue vers le nouveau **OLE DB Destination** pour connecter les deux composants ensemble.



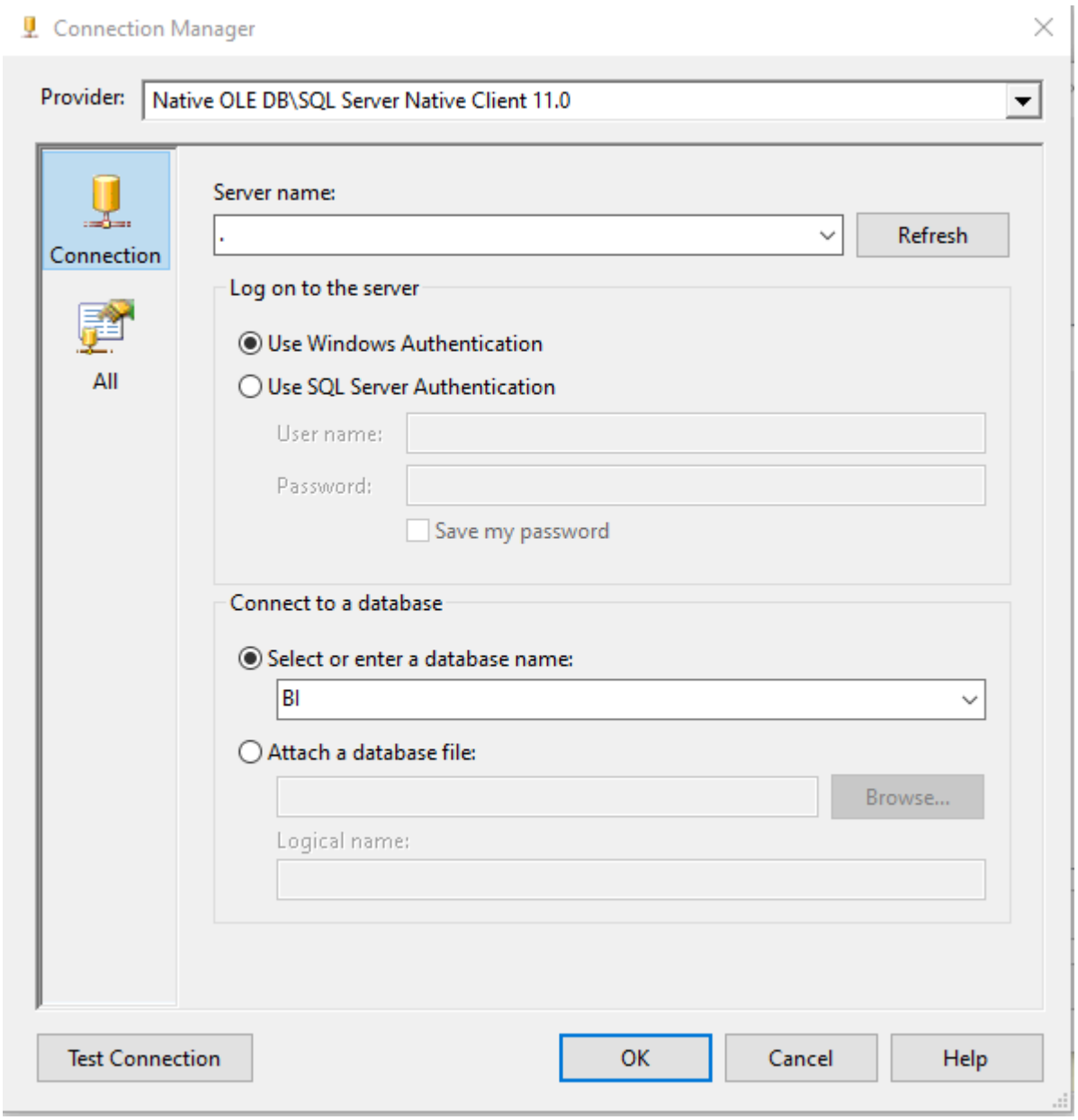
3. Double clic **OLE DB Destination** pour ouvrir la boîte **OLE DB Destination Editor**



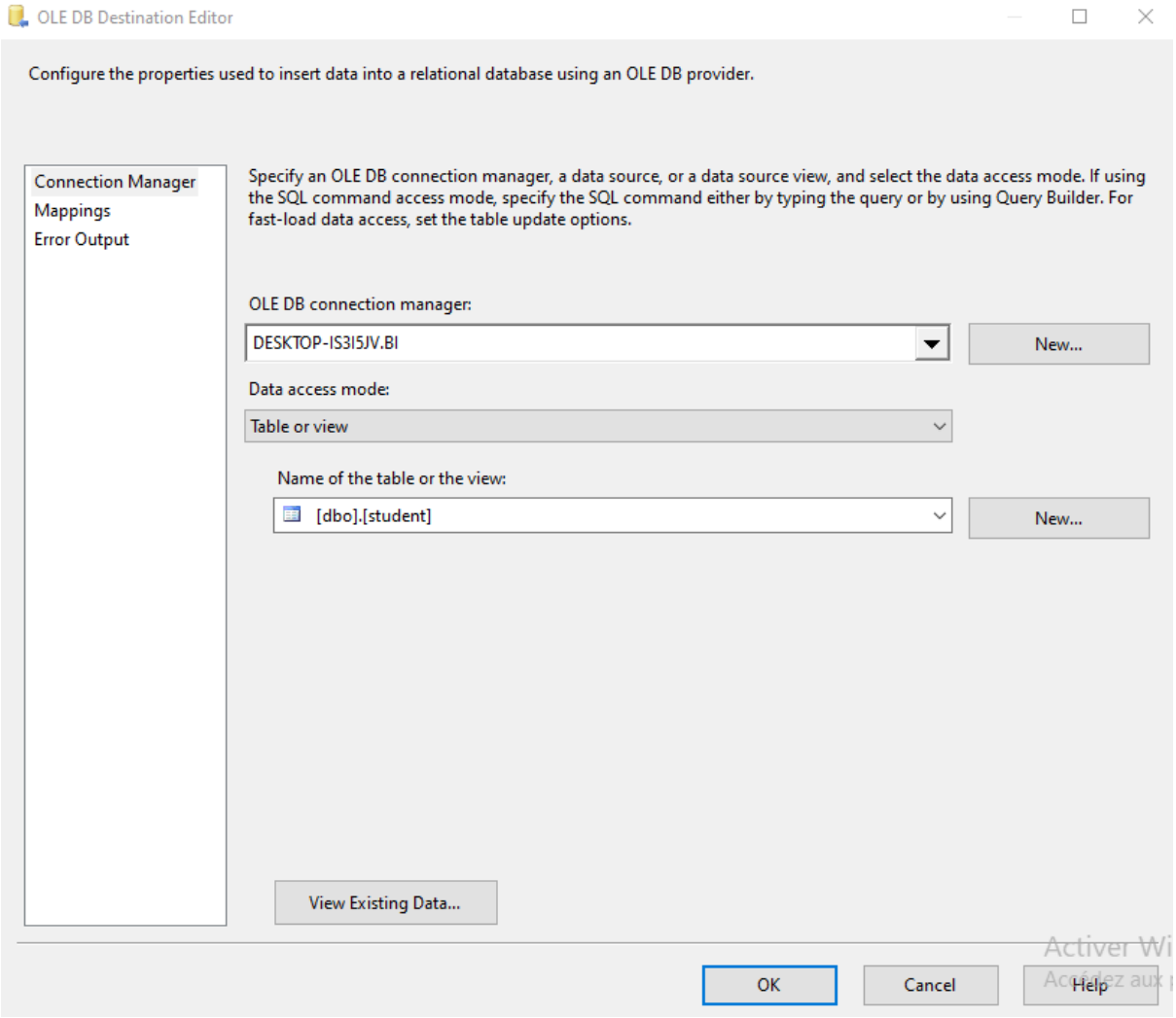
4. Dans l'onglet **OLE DB Destination Editor**, vous devrez configurer la connexion à la base de données de destination. Cliquez sur **New** pour créer une nouvelle connexion OLE DB ou sélectionnez une connexion existante si vous en avez déjà configuré une.
5. La boîte de dialogue **Configure OLE DB Connection Manager** s'ouvrira, Cliquez sur **New** pour créer une nouvelle connexion **OLE DB** ou sélectionnez une connexion existante si vous en avez déjà configuré une.



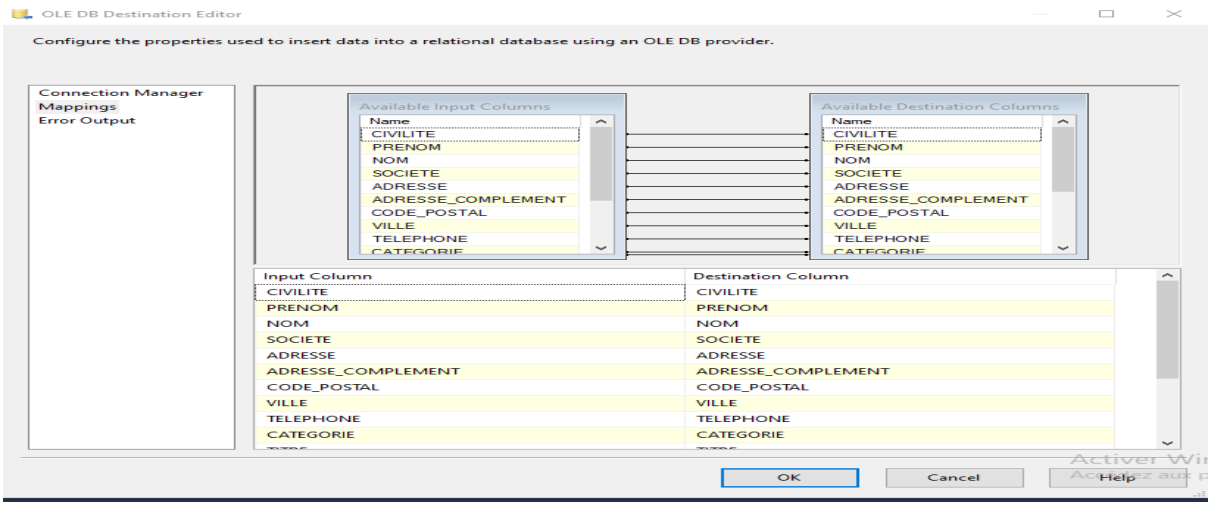
6. La boîte de dialogue **Connection Manager** s'ouvrira, dans le champ Server name pour spécifier le nom de serveur dans notre cas on va tapez « . » qui est utilisé pour représenter le serveur local. Cependant, si vous utilisez une instance distante de SQL Server, vous devrez spécifier le nom de cette instance ou son adresse IP dans le champ.
7. Si vous utilisez une authentification Windows, assurez-vous que l'option **Use Windows Authentication** est sélectionnée. Si vous utilisez une authentification SQL Server, vous devrez spécifier le nom d'utilisateur et le mot de passe approprié.
8. Choisissez de la liste déroulante ou saisissez le nom d'une base de données existante sur le serveur SQL.



9. Une fois que vous avez configuré tous les paramètres de la connexion, dans la boite **OLE DB Destination Editor**, dans le **Name of the table or the view** box, entrez ou sélectionnez **[db].[Student]**.



10. Sélectionner **Mappings**, assurez-vous que les colonnes correspondent en termes de nom.

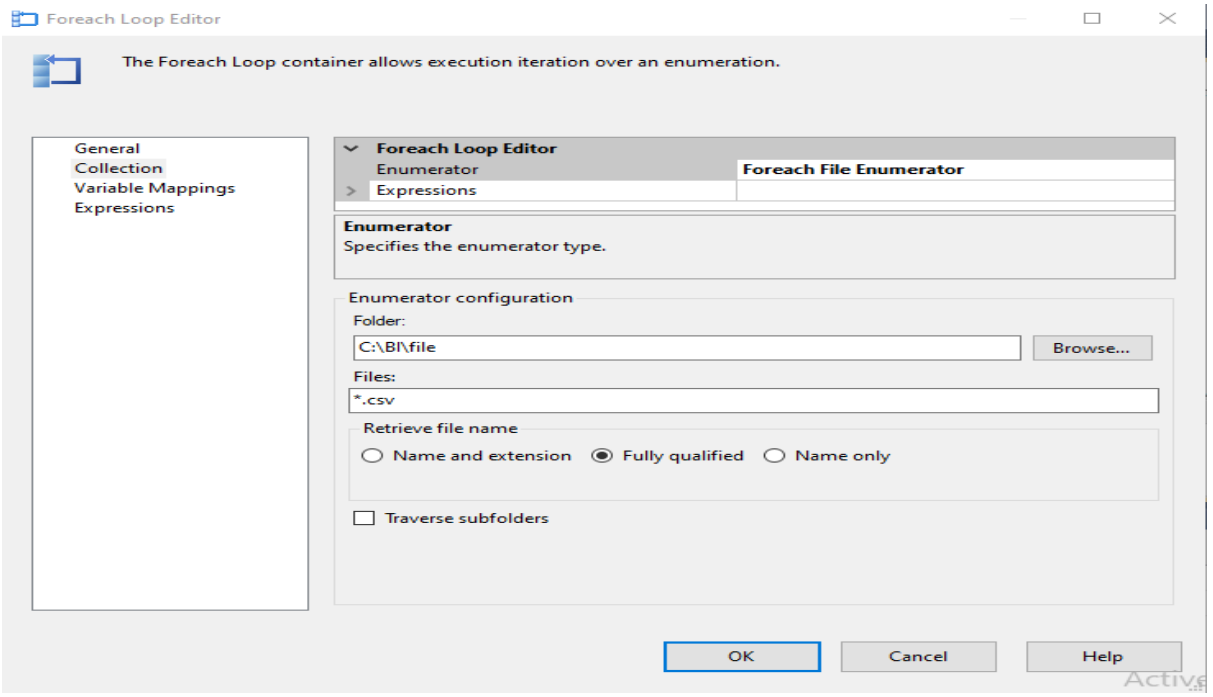


11. Sélectionner **OK**.

4. Foreach Loop Container

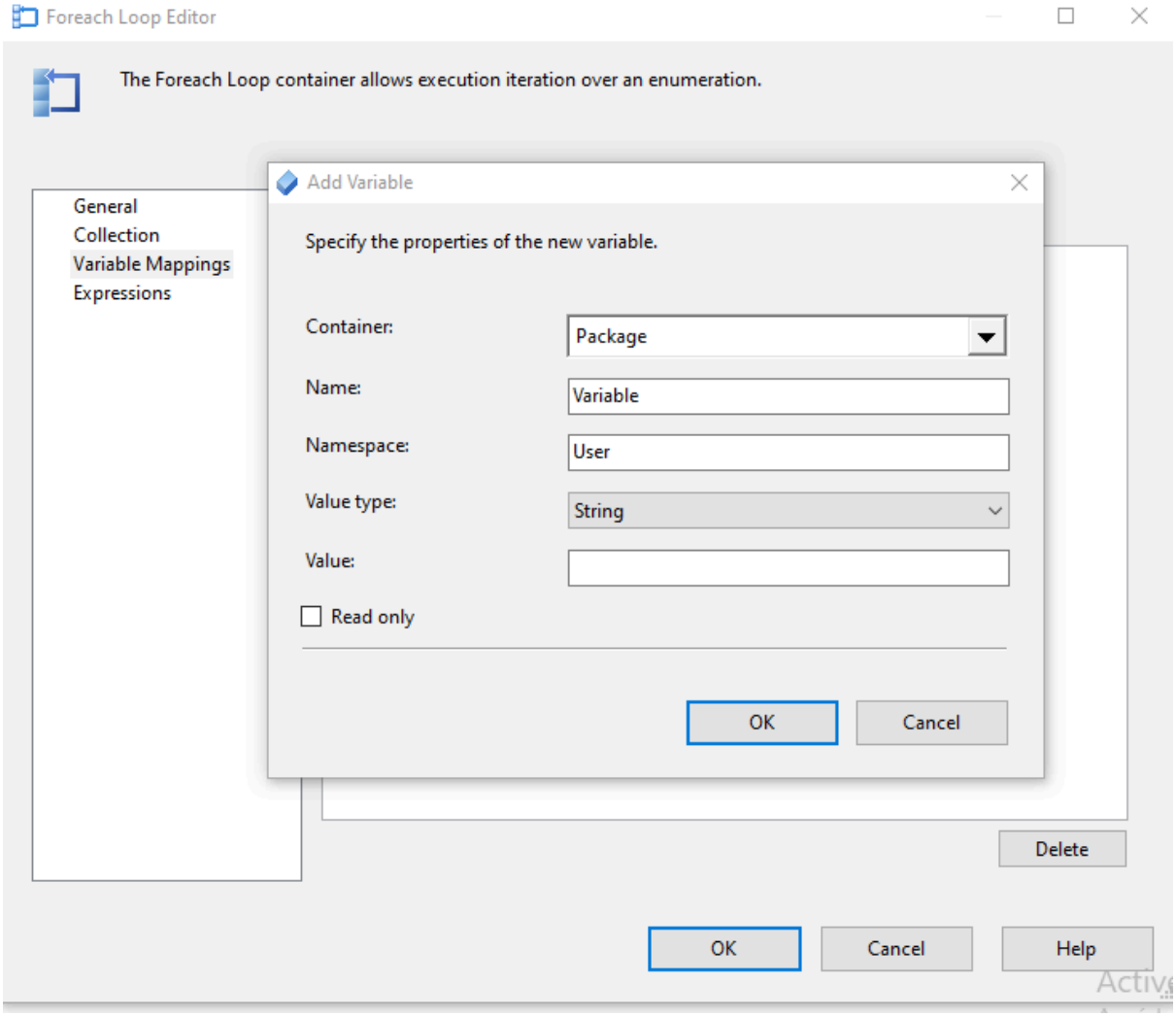
Le conteneur Foreach Loop est un élément essentiel dans SQL Server Integration Services (SSIS) pour traiter plusieurs fichiers ou éléments de données de manière itérative. Dans notre processus SSIS, nous l'utilisons pour parcourir une collection de fichiers CSV dans un répertoire donné et effectuer les mêmes opérations sur chaque fichier. Voici comment vous pouvez l'intégrer dans votre processus :

1. Dans SQL Server Data Tools, sélectionnez le **Control Flow** onglet.
2. Dans **SSIS Toolbox**, développez **Containers**, puis faites glisser **Foreach** sur la surface de conception de la **Controle Flow** onglet.
3. Double-cliquer **Foreach Loop Container** pour ouvrir le **Foreach Loop Editor .**
4. Sélectionner **Collection.**
5. Sur le Collection page, Dans le groupe **Enumerator configuration,** sélectionner **Browse.**
6. Dans la boîte de dialogue de recherche un dossier, localisez le dossier de votre machine contenant les CSV inclus avec les exemples de données.
7. Dans la boîte **Files**, entrez *.csv pour dire tous les fichiers CSV dans le dossier .

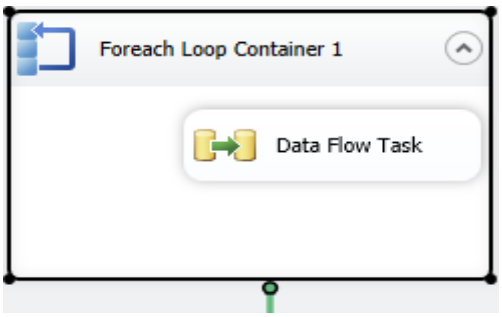


8. Sélectionner Variable **Mappings**

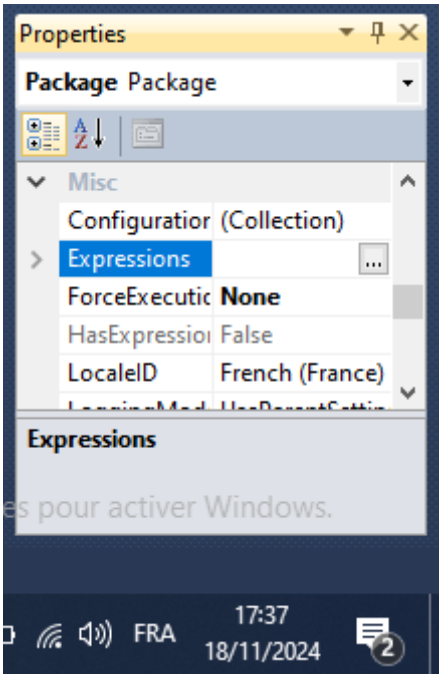
9. Sur la page **Variable Mappings**, dans la colonne Variable, sélectionnez la cellule vide et sélectionnez **< New Variable …>**
10. Dans la boîte de dialogue **Add Variable**, pour **Name** entrer le nom de la variable par exemple **Variable**, pour **Value type** sélectionner String .



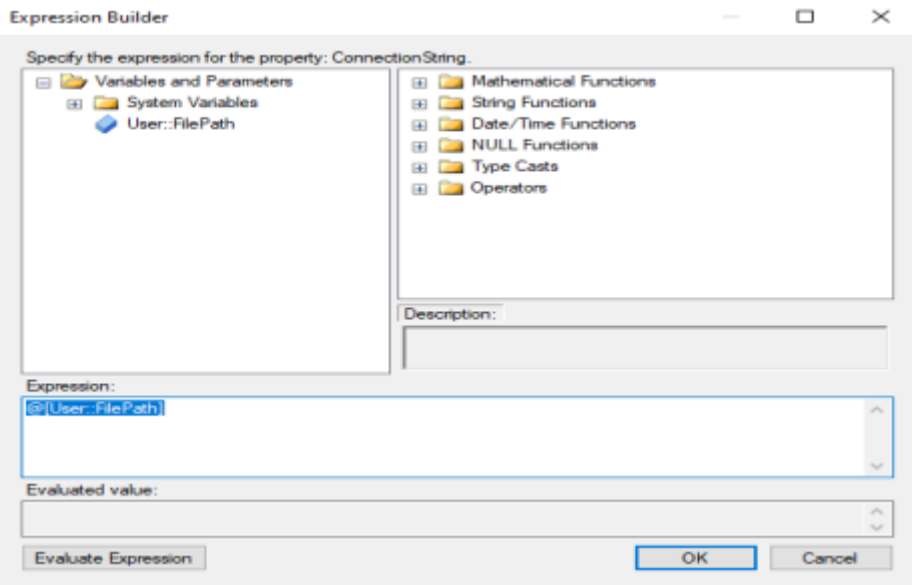
11. Sélectionner **OK.**
12. Sélectionner **OK** encore une fois pour sortir du **Foreach Loop Editor**
13. Faites glisser le **Data Flow Task** dans le composant **Foreach Loop Container .**



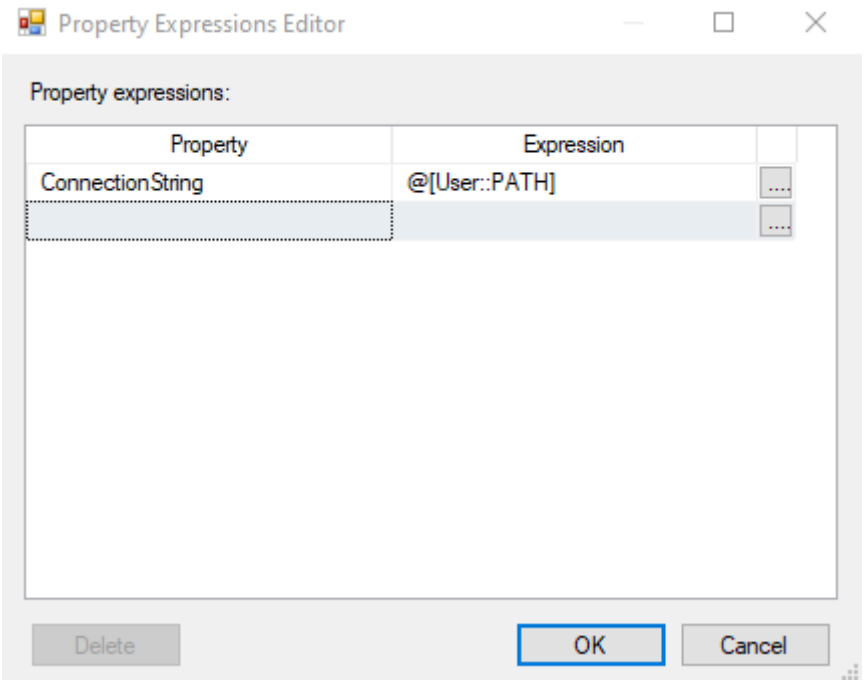
14. Dans le volet **Connection Managers**, cliquez avec le bouton droit sur le Flat File Connection Manager, puis sélectionnez **Properties.**
15. Dans la fenêtre **Properties**, pour **Expressions**, sélectionnez la cellule vide, puis le bouton **(...)**.



16. Dans la boîte de dialogue **Property Expressions Editor**, dans la colonne **Property**, sélectionnez **ConnectionString.**
17. Dans la colonne **Expression**, sélectionnez le bouton **(...)** pour ouvrir la boîte de dialogue **Expression Builder**.
18. Dans la boîte de dialogue **Expression Builder**, développez le nœud **Variables**.
19. Faites glisser la variable User::Variable créé précédemment dans la zone **Expression**.



20. Sélectionnez **OK** pour fermer la boîte de dialogue **Expression Builder.**
21. Sélectionnez **OK** pour fermer la boîte de dialogue **Property Expressions Editor .**



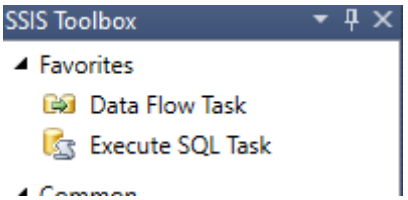
5) Gestion des Fichiers Processés

Lors de l'utilisation du conteneur Foreach Loop pour traiter plusieurs fichiers, il est nécessaire de gérer les fichiers déjà intégrés ou d'ignorer ceux qui ont déjà été traités. Pour ce faire, plusieurs approches peuvent être envisagées. Voici quelques solutions fréquemment utilisées :

Solution 1 : Vidage de la Table Existante

Une méthode couramment utilisée consiste à insérer une tâche *Execute **SQL Task*** avant le conteneur ***Foreach Loop***. Cette tâche exécute une requête SQL visant à vider la table cible, ce qui permet de supprimer les données existantes avant de procéder à l'intégration de nouveaux fichiers. Voici les étapes pour y parvenir :

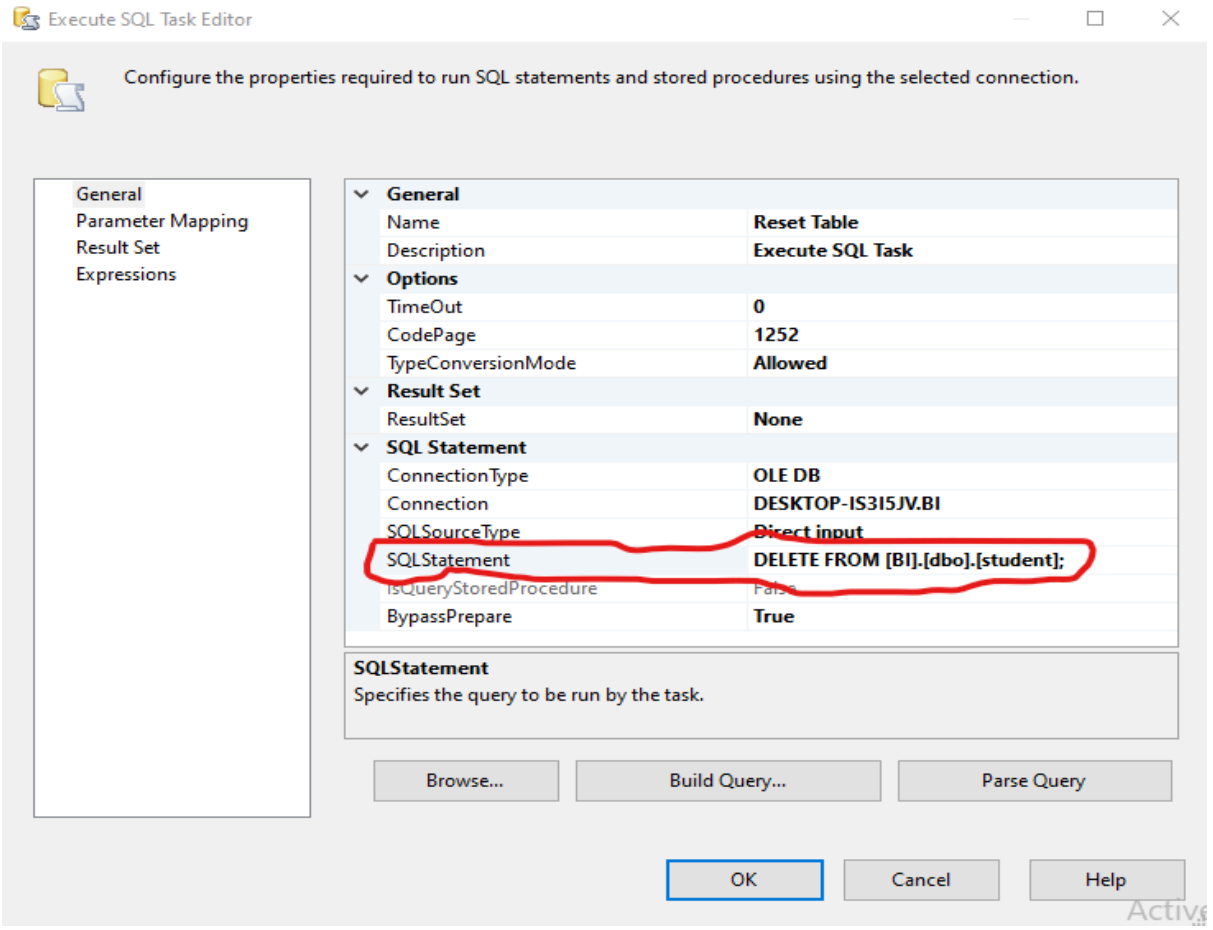
1. Dans SQL Server Data Tools, sélectionnez le **Controle Flow** onglet.
2. Dans le **SSIS Toolbox** volet, développer Favorites, et faites glisser **Execute SQL Task**



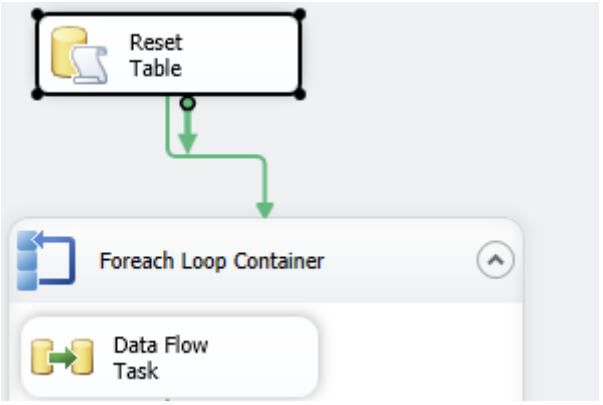
3. Double-cliquer pour ouvrir **Execute SQL Task** Editor .
4. Dans la section **Connection**, de l'onglet **Général** sélectionnez la connexion LocalHost.BI établie auparavant ou créez la connexion avec **< New Connection …> .**
5. Dans la section "SQL Statement," saisissez la requête SQL appropriée pour vider la table existante ou ouvrir la boîte Enter SQL Query .

   ```
   DELETE FROM [BI].[dbo].[Student]
   ```

6. Sélectionner **OK**.



7. Faites glisser le connecteur (la flèche verte) de l'**Execute SQL Task** vers le **Foreach Loop Container**.
8. Vous pouvez ainsi renommer le composant Execute SQL Task , je l'ai nommé Reset Table .

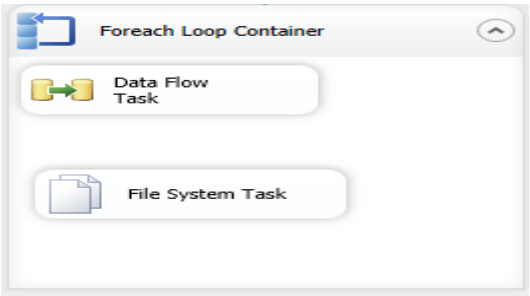


Solution 2 : Transférer les fichiers .

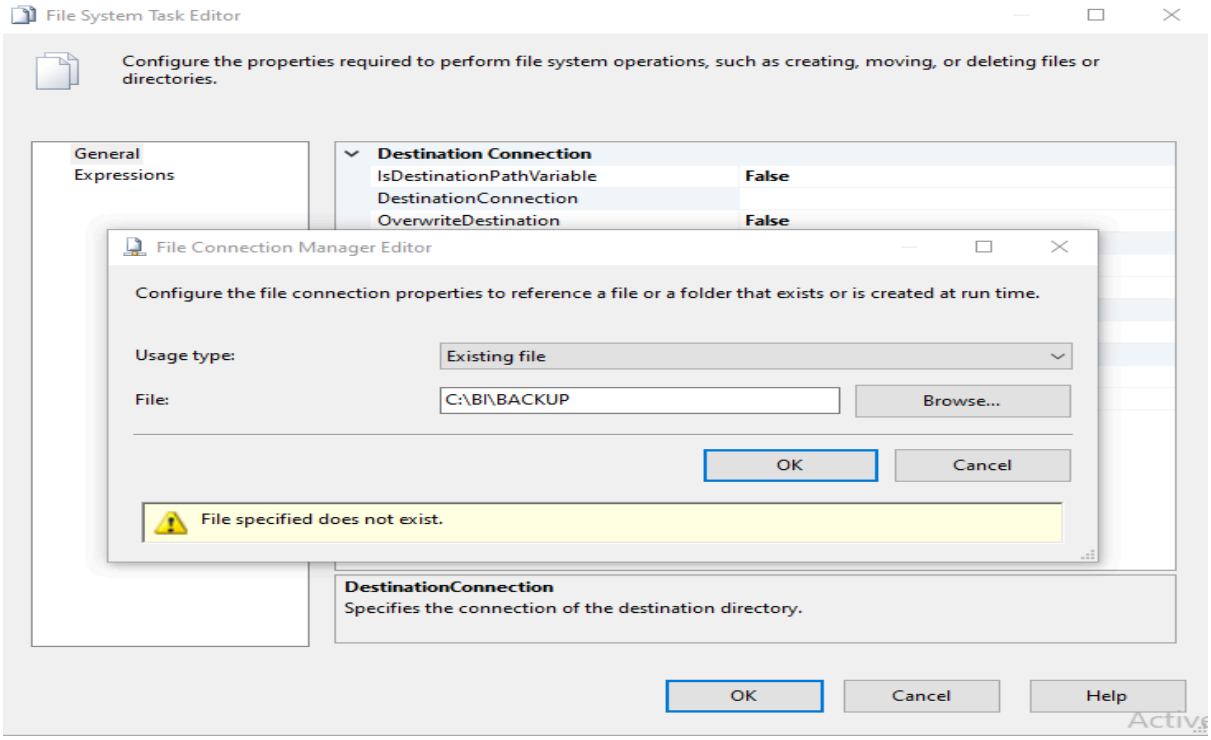
Une autre solution pour gérer les fichiers nouvellement traités consiste à les déplacer vers un dossier de sauvegarde après leur intégration. Cette méthode permet d'éviter que les fichiers déjà intégrés ne soient

réimportés à l'avenir. Voici les étapes à suivre pour mettre en œuvre cette approche :

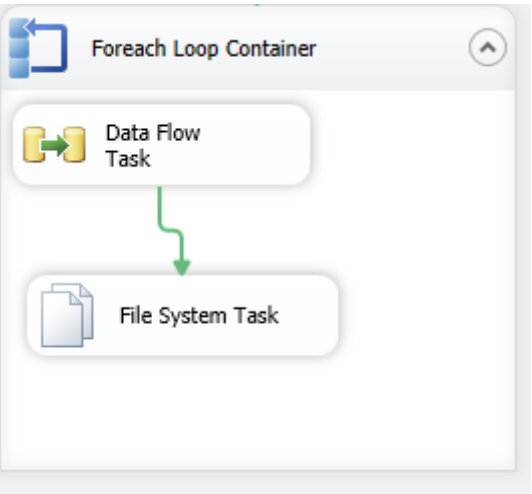
1. Créez un dossier dans lequel nous allons placer le fichier source par exemple Backup .
2. Dans le **SSIS Toolbox** volet, développer **Common**, et faites glisser **File System Task** dans le composant **Foreach Loop Container.**



3. Double-cliquer pour ouvrir **File System Task Editor**
4. Dans le champ **Opération**, sélectionnez **Move file**
5. Dans le champ **Destination Connection**, choisir **<New Connection…> .**
6. Dans **File Connection Manager Editor** dans le champ Usage Type choisir **Existing folder** si tu as créé le dossier Backup si nom choisir **Create folder .**
7. Cliquer sur Browse dans le champ Folder et entrer le chemin de dossier



8. Dans la partie **Source Connection**, mettez la valeur de la propriété **IsSourcePathVariable** à **True** et sélectionnez la variable **User::FilePath .**
9. Faites glisser le connecteur (la flèche verte) de **Data Flow Task** vers le **File System Task .**



Le transfert des fichiers vers un dossier de sauvegarde dans le cadre du processus SSIS constitue une solution efficace pour éviter la réintégration de fichiers déjà traités. Cependant, cette méthode comporte certains inconvénients, notamment une augmentation de la consommation d'espace disque, une gestion plus complexe des fichiers sauvegardés, ainsi qu'un risque accru d'erreurs humaines. Cela requiert une surveillance constante pour garantir un fonctionnement optimal.

Solution 3 : Vérification de l'Existence du Fichier .

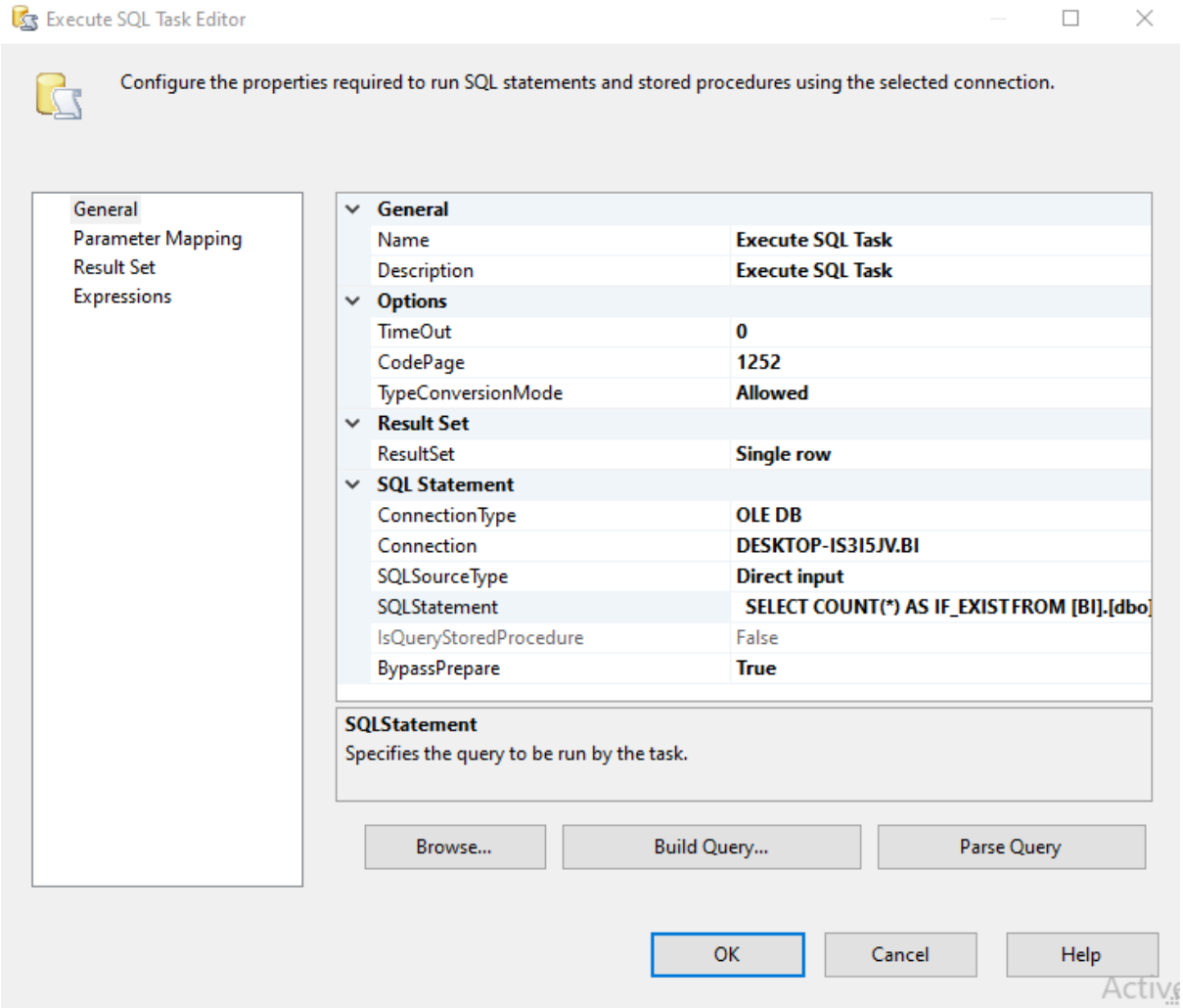
Une troisième solution pour gérer les nouveaux fichiers dans le cadre du processus SSIS consiste à vérifier préalablement leur existence dans la table *HISTORY*. Cette méthode garantit que seuls les fichiers absents de cette table seront intégrés. Voici les étapes à suivre pour mettre en œuvre cette approche :

1. Créer une table nommé par exemple **HISTORY** avec deux colonnes ID et FileName
2. Dans le SSIS Toolbox, sous Favorites faites glisser l'élément **Execute SQL Task** sur la surface de conception Control Flow dans le **Foreach Loop Container**
3. **Double-cliquez** sur la tâche **Execute SQL Task** pour ouvrir l'éditeur

4. Dans l'éditeur de la tâche **Execute SQL Task Editor**, sous l'onglet **Général** à la partie **SQL Statement**, sélectionnez ou créez une connexion à votre base de données cible.
5. Sélectionner **Single row** dans le champ **ResultSet .**
6. Dans le champ **SQLStatement**, saisissez la requête SQL suivante pour vérifier si le fichier existe dans la table HISTORY :

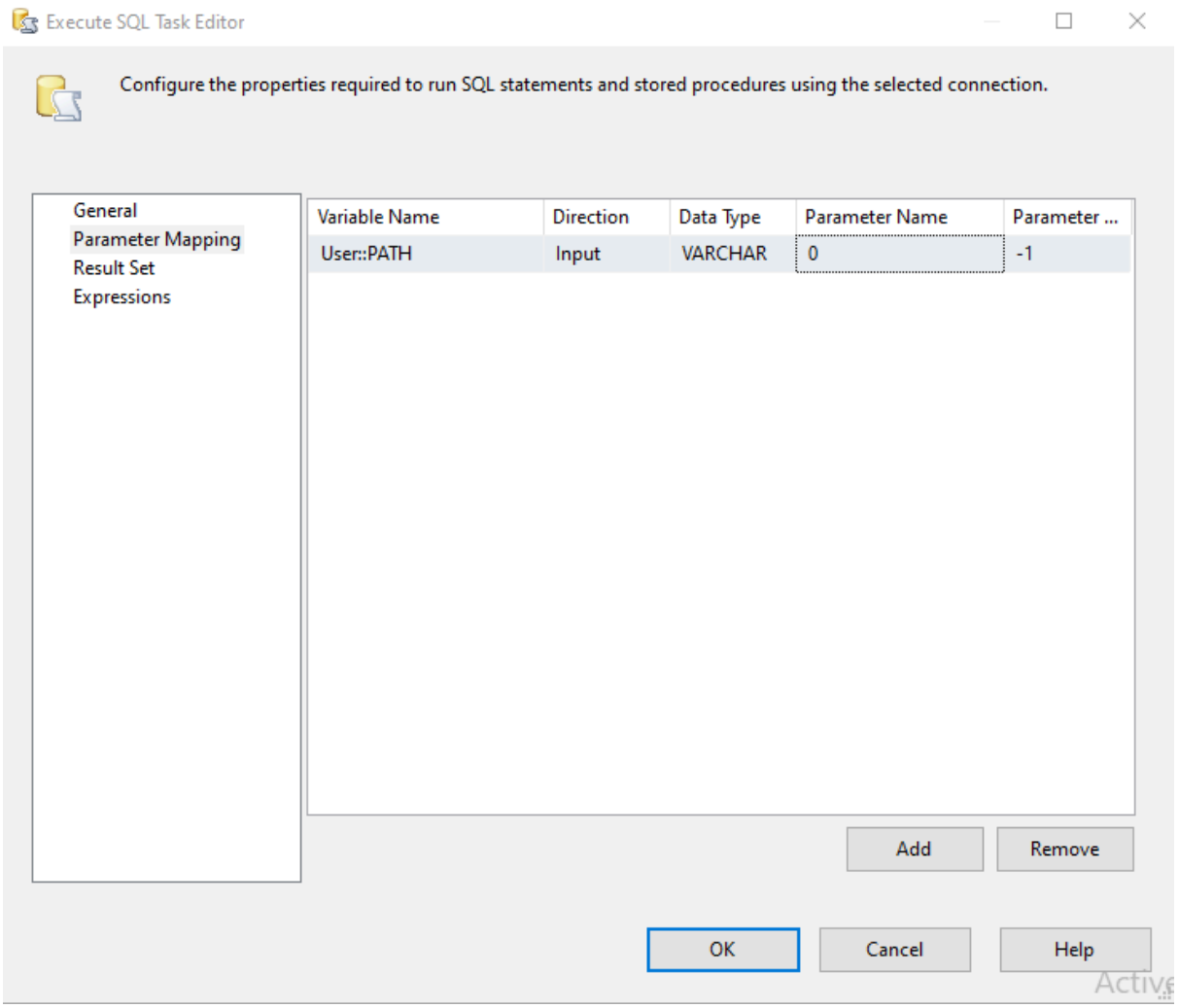
```
SELECT COUNT(*) AS IF_EXIST
 FROM [BI].[dbo].[HISTORY]
 WHERE FileName = ?
```

Cette requête permet de vérifier si un fichier donné est déjà présent dans la table `HISTORY` de la base de données `BI`. Elle utilise des paramètres pour rendre la vérification dynamique en fonction du nom du fichier.

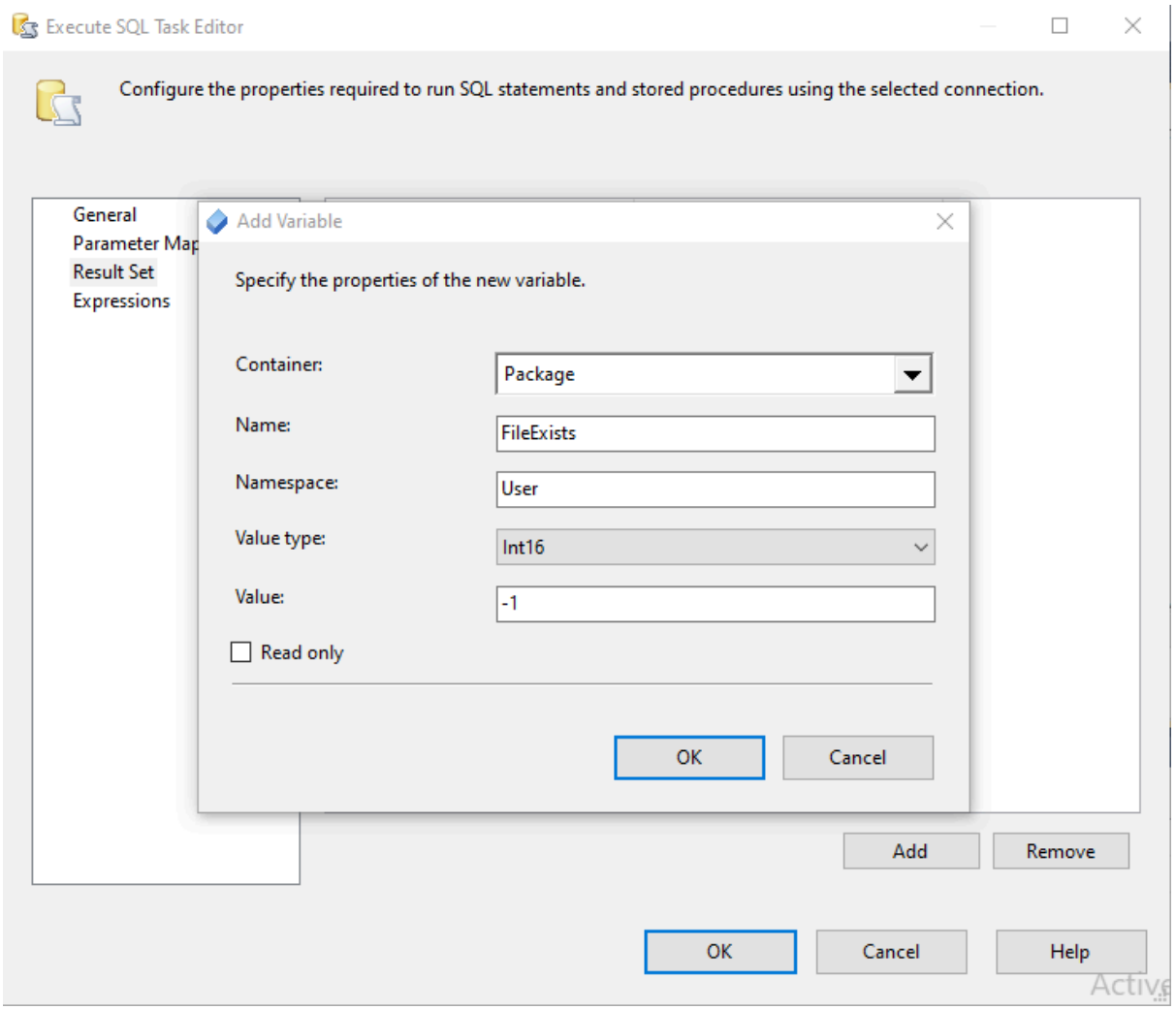


7. Allez à l'onglet **Parameter Mapping**, cliquez sur **Add** pour configurer le variable .

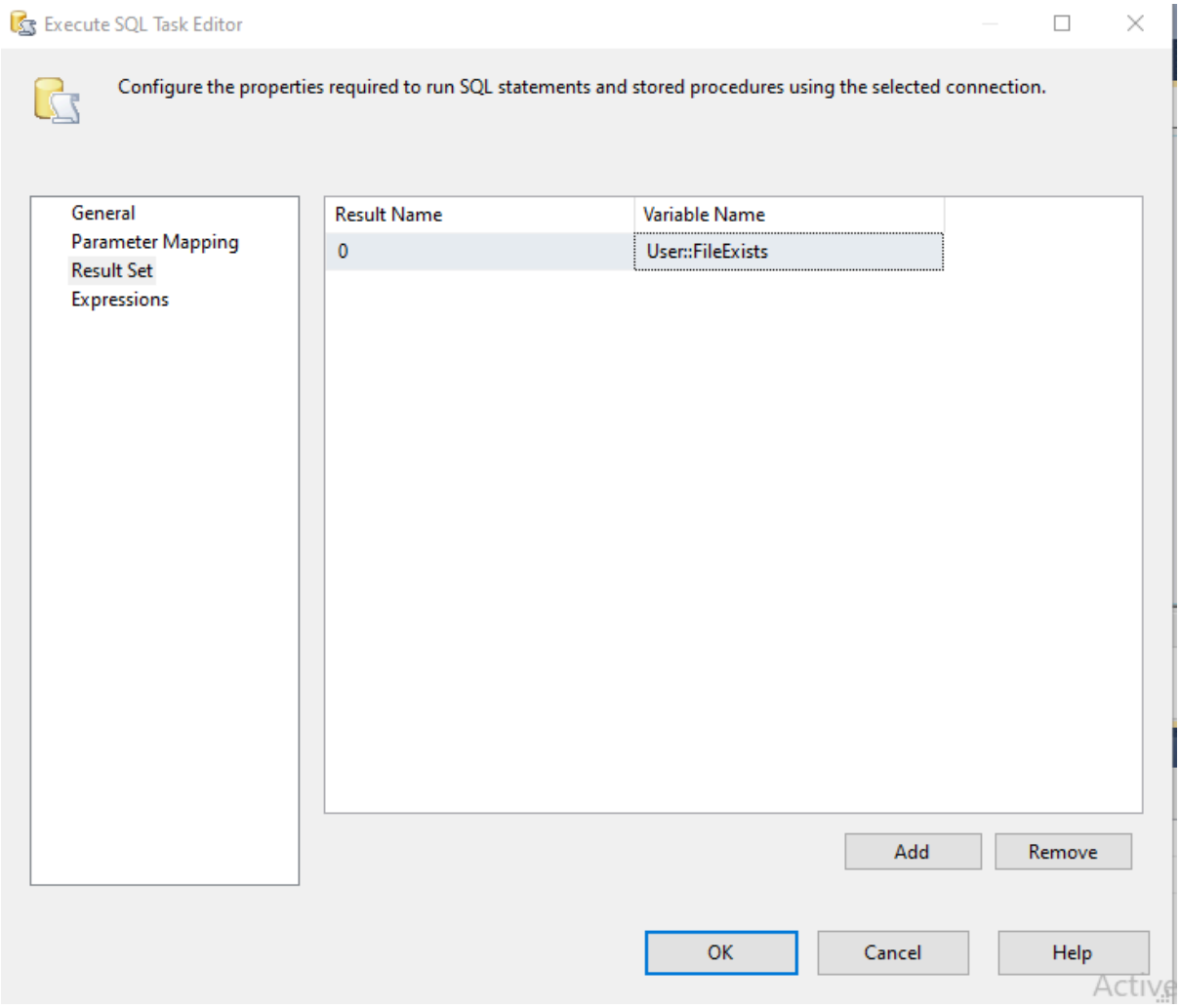
8. Dans la colonne **Variable Name**, sélectionnez la variable qui contient le nom du fichier à vérifier c'est le variable **Path** .
9. Dans la colonne **Direction**, choisissez **Input** pour indiquer que la variable est utilisée comme paramètre d'entrée dans la requête SQL.
10. Dans la colonne **Data Type**, sélectionnez le type de données approprié c'est **VARCHAR .**
11. Dans la colonne **Parameter Name**, utiliser le chiffre "0", cela signifie que vous faites référence au premier paramètre dans votre requête SQL.
12. Pour la colonne **Parameter Size**, L'utilisation de -1 est souvent appropriée lorsque la taille des données stockées dans le paramètre peut être inconnue à l'avance .



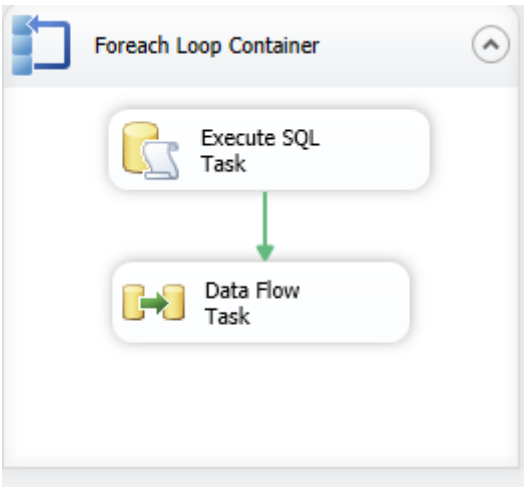
13. Allez à l'onglet **Result Set**, cliquez sur **Add .**
14. Dans la colonne **Result Name**, utiliser "0" comme nom de résultat, où "0" indique la première (et souvent la seule) ligne de résultat
15. Dans la colonne **Variable Name**, sélectionnez**< New Variable.>**
16. Dans la boîte de dialogue **Add Variable**, pour **Name** entrer le nom de la variable par exemple **FileExists**, pour **Value type** sélectionner Int16 , pour **Value** sélectionner -1 .



17. Sélectionner **OK** .

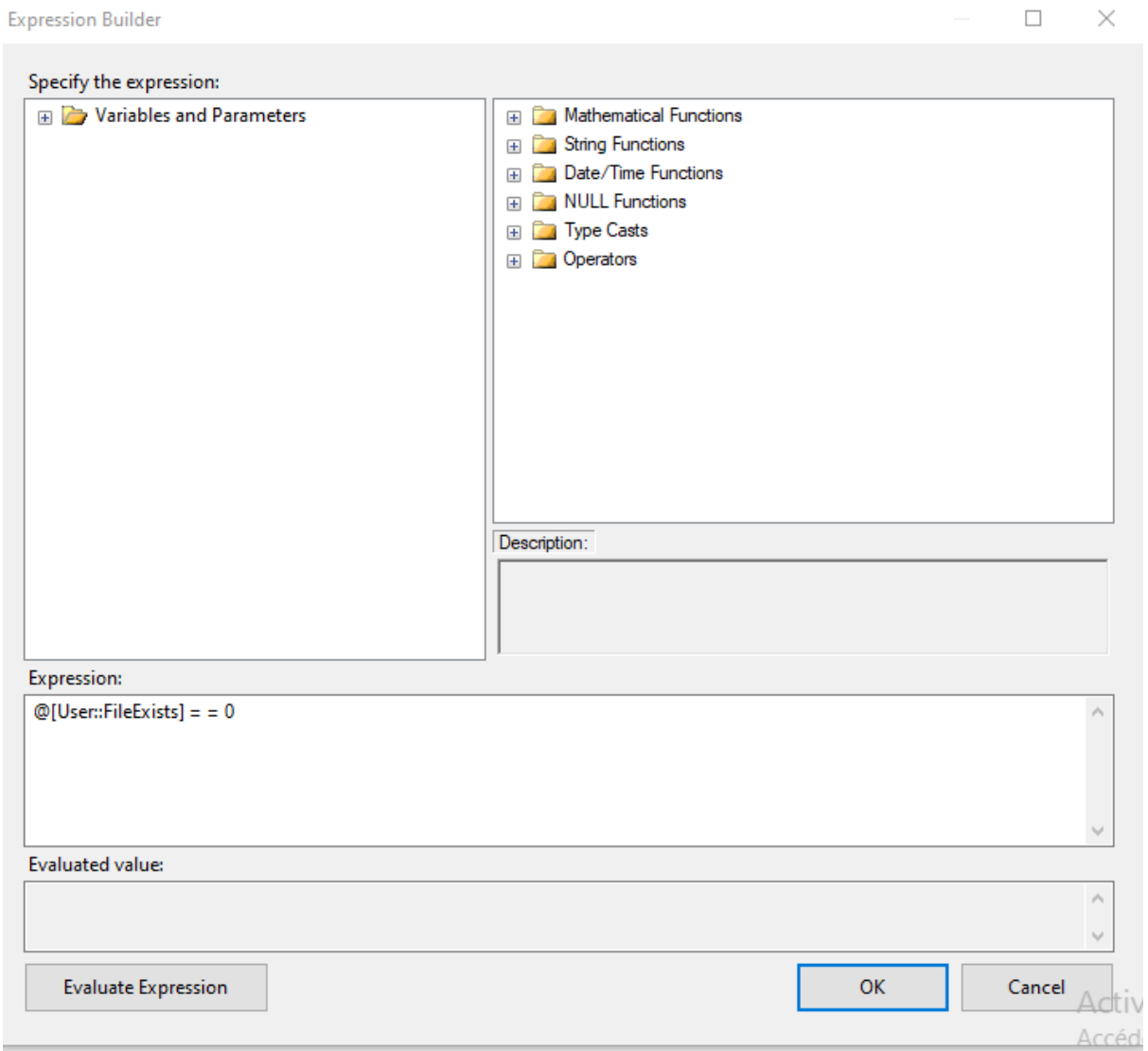


18. Sélectionner **OK .**
19. Faites glisser le connecteur (la flèche verte) depuis la tâche **Execute SQL Task** vers la tâche **Data Flow Task .**



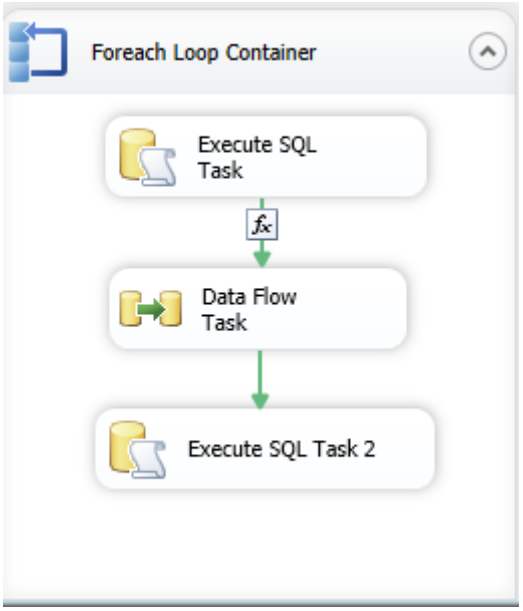
20. Faites un clic droit sur la flèche puis **Edit .**
21. La boite **Precedence Constraint Editor** s'ouvrira .

22. Dans la liste déroulante **Evaluation operation**, sélectionnez **Expression .**
23. Dans la zone **Expression**, sélectionnez le bouton (...) pour ouvrir la boîte de dialogue **Expression Builder**.
24. Dans la zone **Expression** tapez @[User::FileExists] = = 0 cette une expression conditionnelle qui doit être configurée pour s'assurer que le Data Flow Task ne s'exécute que lorsque la condition spécifiée est vraie.



25. .Sélectionnez **OK** pour fermer la boîte de dialogue **Expression Builder.**
26. Sélectionnez **OK** pour fermer la boîte de dialogue **Property Expressions Editor .**
27. Ajoutez un autre **Execute SQL Task** pour enregistrer le nom du fichier intégré dans la table HISTORY (répéter les étapes 2 à 4)

28. Dans le champ **SQLStatement**, saisissez la requête SQL suivante pour vérifier si le fichier existe dans la table HISTORY :
```
INSERT INTO [BI].[dbo].[HISTORY](FileName)
 VALUES (?)
```
29. Répéter les étapes de 7 à 12
30. Faites glisser le connecteur (la flèche verte) depuis la tâche **Data Flow Task** vers la deuxième **Execute SQL Task .**



Nous avons opté pour la troisième solution, qui consiste à vérifier la présence du fichier dans la table HISTORY avant de l'intégrer. Cette approche garantit que seuls les nouveaux fichiers sont traités, réduisant ainsi les doublons et optimisant l'efficacité du processus, tout en limitant la consommation d'espace disque. Bien que cela nécessite une gestion plus complexe, cela représente un compromis nécessaire pour préserver l'intégrité des données et la performance du système.

6. Nettoyage de données

Le nettoyage des données constitue une étape clé dans tout processus d'intégration, transformation et chargement de données, comme celui réalisé avec SQL Server Integration Services (SSIS). Cette phase a pour objectif de garantir que les données entrant dans le système soient fiables, de haute qualité, cohérentes et conformes aux normes définies. L'objectif principal du nettoyage des données est de supprimer les incohérences, les erreurs ainsi que les valeurs incorrectes ou manquantes, assurant ainsi une base de données robuste et fiable pour les étapes suivantes.

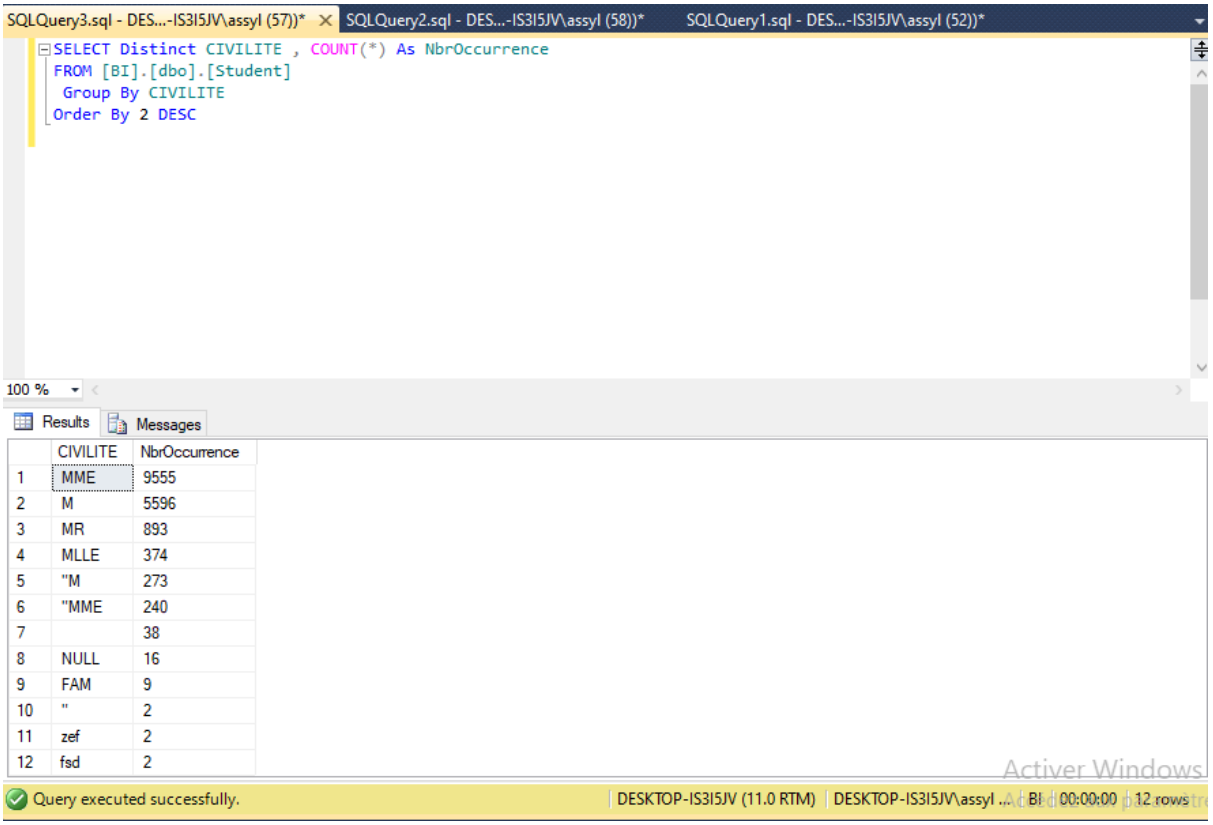
Nettoyage colonne civilité

La colonne "CIVILITE" dans la table "[BI].[dbo].[Student]'' présente plusieurs problèmes potentiels, notamment une diversité de valeurs, des valeurs incohérentes, des valeurs manquantes et des valeurs indésirables. Pour garantir la qualité et la cohérence des données, nous avons mis en place un processus de nettoyage de la colonne "CIVILITE". Voici les principales étapes de ce processus :

1. **Problématique**

Avant de nettoyer la colonne "CIVILITE", nous avons analysé les données existantes en utilisant la requête suivante :

```
SELECT Distinct CIVILITE , COUNT(*) As NbrOccurrence
FROM [BI].[dbo].[Student]
 Group By CIVILITE
Order By 2 DESC
```



Les résultats de cette requête ont révélé plusieurs problèmes :

- Diversité des valeurs : La colonne "CIVILITE" comporte un certain nombre de valeurs différentes, ce qui peut rendre la gestion et l'analyse des données plus complexes. Il est souhaitable de réduire cette diversité pour des raisons de cohérence et de facilité d'utilisation.
- Valeurs incohérentes : Les valeurs telles que "M", "MR", "MME", et "MLLE" semblent représenter des titres de civilité, mais elles sont mal orthographiées ou abrégées, ce qui peut rendre les données peu fiables et difficiles à interpréter.
- Valeurs manquantes : Les valeurs nulles (vide) et "NULL" sont présentes, ce qui indique des données manquantes ou incorrectes.
- Valeurs indésirables : Les valeurs "zef" et "fsd" semblent être des valeurs indésirables, probablement des erreurs ou des valeurs non conformes, qui devraient être supprimées ou corrigées.
- Manque de cohérence : Les valeurs "FAM" et "FAMILLE" semblent représenter la même catégorie, mais elles sont différentes. Il est souhaitable d'harmoniser ces valeurs pour une cohérence accrue.

2. **Utilisation de la Colonne "FLAG"**

Pour garantir que seules les nouvelles données sont affectées par les opérations de nettoyage, nous utilisons la colonne "FLAG" avec une valeur par défaut de "0". Cela signifie que toutes les nouvelles données insérées dans la table ont initialement un "FLAG" de "0". Lorsque nous effectuons les opérations de nettoyage, nous appliquons des filtres pour ne cibler que les enregistrements avec un "FLAG" égal à "0", garantissant que les données existantes, qui pourraient avoir été traitées auparavant, ne sont pas altérées.

3. **Nettoyage des Données.**

Pour remédier à ces problèmes, nous avons utilisé des instructions SQL pour effectuer les opérations de nettoyage suivantes :

- Les valeurs "MR" et "M" ont été mises à jour pour être "M".
- Les valeurs "MLLE" et "MME" ont été mises à jour pour être "MME".
- Les valeurs vides ("") ont été mises à jour pour être nulles ("NULL").
- La valeur "FAM" a été mise à jour pour être "FAMILLE".
- Les enregistrements avec des prénoms contenant " ET " ont été mis à jour avec "FAMILLE" comme civilité, sauf s'ils avaient déjà "FAMILLE" comme civilité.
- Les enregistrements avec une civilité vide ("") ont été mis à jour en fonction des prénoms correspondants (c'est-à-dire, si le même prénom avait une "M" ou "MME" ailleurs).
- Les valeurs "zef" et "fsd" ont été supprimées de la table.

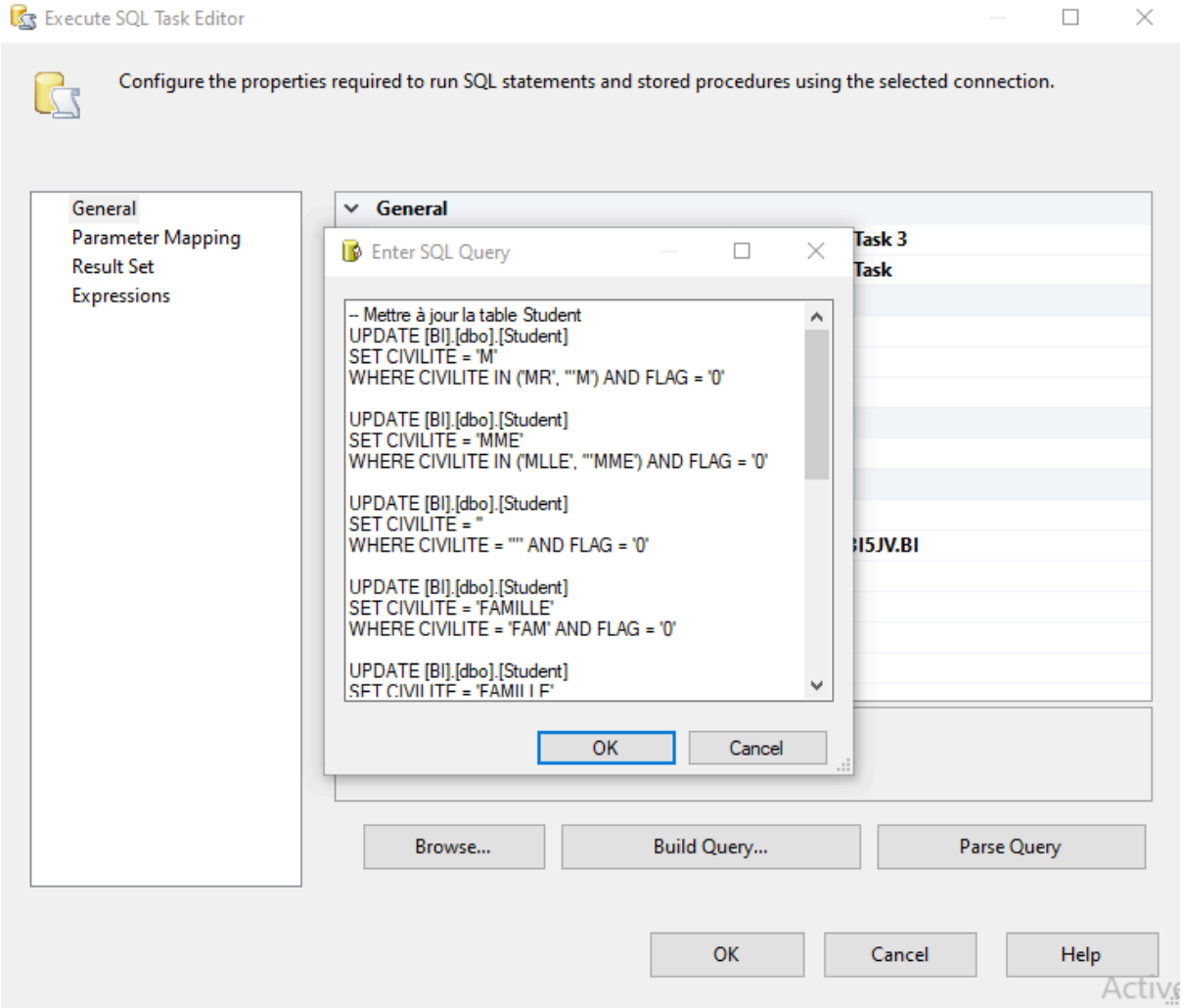
ÉTAPES À FAIRE :

1. Dans le SSIS Toolbox, sous Favorites faites glisser l'élément **Execute SQL Task** sur la surface de conception Control Flow
2. Double-cliquez sur la tâche **Execute SQL Task** pour ouvrir l'éditeur .

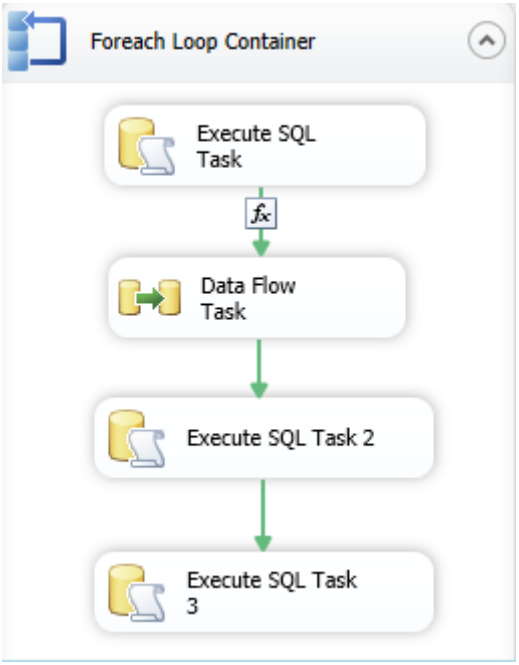
3. Dans l'éditeur de la tâche **Execute SQL Task Editor**, sous l'onglet **General** à la partie **SQL Statement**, sélectionnez ou créez une connexion à votre base de données cible.
4. Dans le champ **SQLStatement**, saisissez la requête SQL suivante pour nettoyer la colonne civilité :

```
-- Mettre à jour la table Student
UPDATE [BI].[dbo].[Student]
SET CIVILITE = 'M'
WHERE CIVILITE IN ('MR', '"M') AND FLAG = '0'

UPDATE [BI].[dbo].[Student]
SET CIVILITE = 'MME'
WHERE CIVILITE IN ('MLLE', '"MME') AND FLAG = '0'

UPDATE [BI].[dbo].[Student]
SET CIVILITE = ''
WHERE CIVILITE = '"' AND FLAG = '0'

UPDATE [BI].[dbo].[Student]
SET CIVILITE = 'FAMILLE'
WHERE CIVILITE = 'FAM' AND FLAG = '0'

UPDATE [BI].[dbo].[Student]
SET CIVILITE = 'FAMILLE'
WHERE PRENOM LIKE '% ET %' AND CIVILITE <> 'FAMILLE' AND
FLAG = '0'

UPDATE [BI].[dbo].[Student]
SET CIVILITE = 'M'
WHERE CIVILITE = ''
AND PRENOM IN (SELECT DISTINCT PRENOM FROM
[BI].[dbo].[Student] WHERE CIVILITE = 'M')
AND FLAG = '0'

UPDATE [BI].[dbo].[Student]
SET CIVILITE = 'MME'
WHERE CIVILITE = ''
```

```
AND PRENOM IN (SELECT DISTINCT PRENOM FROM
[BI].[dbo].[Student] WHERE CIVILITE = 'MME')
AND FLAG = '0'

-- Supprimer les lignes avec des valeurs non valides dans CIVILITE
DELETE FROM [BI].[dbo].[Student]
WHERE CIVILITE IN ('fsd', 'zef', 'NULL') AND FLAG = '0'
```



5. Faites glisser le connecteur (la flèche verte) depuis la tâche **Foreach Loop Container** vers le composant **Execute SQL Task**



Après ces opérations de nettoyage, la colonne "CIVILITE" est désormais homogène et contient des valeurs normalisées. Cela améliore la fiabilité des données, les rendant prêtes pour des analyses ou des rapports. L'utilisation de la colonne "FLAG" a permis de cibler uniquement les nouvelles données, garantissant que les données existantes ne soient pas modifiées.

Nettoyage colonne téléphone

La colonne "TELEPHONE" de la table Student peut également contenir des données problématiques. Pour garantir la qualité et la cohérence des données téléphoniques, un processus de nettoyage spécifique est mis en place. Voici les étapes clés de ce processus :

1. Problématique

Avant de procéder au nettoyage, une analyse préliminaire des données est réalisée avec les requêtes suivantes :

a. Sélection des valeurs non numériques :

```
SELECT DISTINCT TELEPHONE
```

```
 FROM [BI].[dbo].[Student]

 WHERE ISNUMERIC(TELEPHONE) = 0

GROUP BY TELEPHONE
```

b. Comptage des occurrences de longueurs de numéro de téléphone

```
SELECT LEN(TELEPHONE), COUNT(*) AS NbrOccurence

FROM [BI].[dbo].[Student]

GROUP BY LEN(TELEPHONE)
```

Les résultats de ces requêtes identifient les problèmes potentiels liés aux numéros de téléphone, tels que les numéros non numériques et les numéros de longueur inappropriée.

2. Nettoyage des Données

a. Les numéros de téléphone non numériques sont supprimés en affectant la valeur vide ("") aux enregistrements correspondants :

```
UPDATE [BI].[dbo].[Student]

 SET TELEPHONE = ' '

WHERE ISNUMERIC(TELEPHONE) = 0 AND FLAG = '0'
```

b. Les numéros de téléphone ayant une longueur inférieure à 9 caractères sont supprimés (affectation de la valeur vide) :

```
UPDATE [BI].[dbo].[Student]

 SET TELEPHONE = ''

WHERE LEN(TELEPHONE) < 9 AND FLAG = '0'
```

c. Les numéros de téléphone ayant des longueurs de 12 ou 13 caractères sont également supprimés :

```
UPDATE [BI].[dbo].[Student]

 SET TELEPHONE = ''
```

WHERE LEN(TELEPHONE) IN (12, 13) AND FLAG = '0'

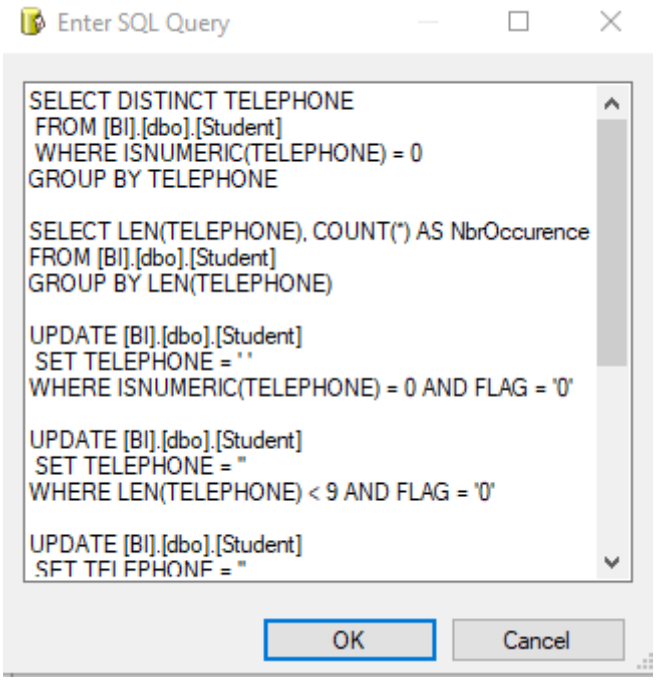
d. Les numéros de téléphone de 11 caractères, commençant par autre chose que "33", sont supprimés.

```
UPDATE [BI].[dbo].[Student]

 SET TELEPHONE = ''

WHERE LEN(TELEPHONE) = 11

 AND SUBSTRING(TELEPHONE,1,2)<>'33'AND FLAG = '0'
```

e. Les numéros de téléphone de 11 caractères commençant par "33" ont "33" supprimé et sont préfixés par "0" :

```
UPDATE [BI].[dbo].[Student]

 SET TELEPHONE = '0' + SUBSTRING(TELEPHONE, 3,

 WHERE LEN(TELEPHONE) = 11 AND FLAG = '0'
```

En appliquant ces opérations de nettoyage, la colonne "TELEPHONE" est rendue plus uniforme et ne contient que des numéros de téléphone valides, prêts à être utilisés dans des analyses ou des rapports.

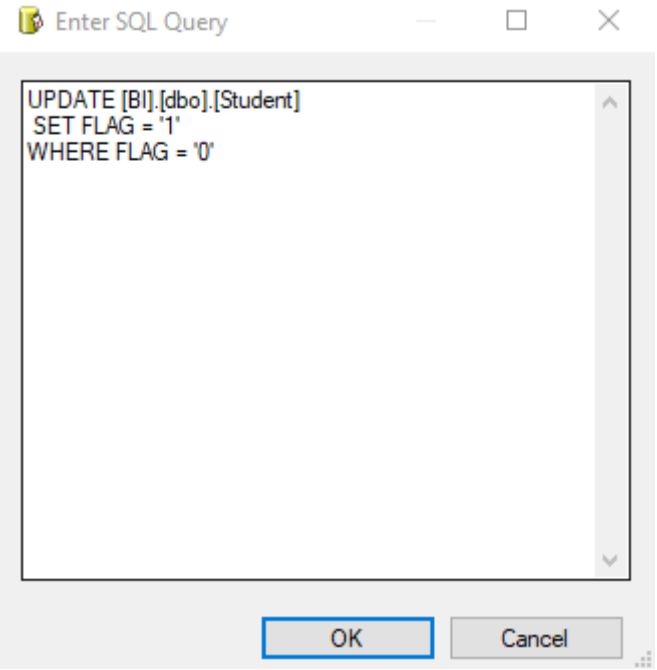


Mise à Jour de la Colonne FLAG

Après avoir effectué le nettoyage des colonnes "CIVILITE" et "TELEPHONE", il est essentiel de marquer les enregistrements concernés pour indiquer qu'ils ont été nettoyés. Cela permet d'éviter de traiter à nouveau ces enregistrements lors des opérations de nettoyage futures. Pour cela, nous utilisons une tâche SQL pour mettre à jour la colonne "FLAG". Voici comment cela fonctionne :

1. Ajouter un composant **Execute SQL Task** .
2. Ajouter la requête suivante :

```
UPDATE [BI].[dbo].[Student]

 SET FLAG = '1'

WHERE FLAG = '0'
```



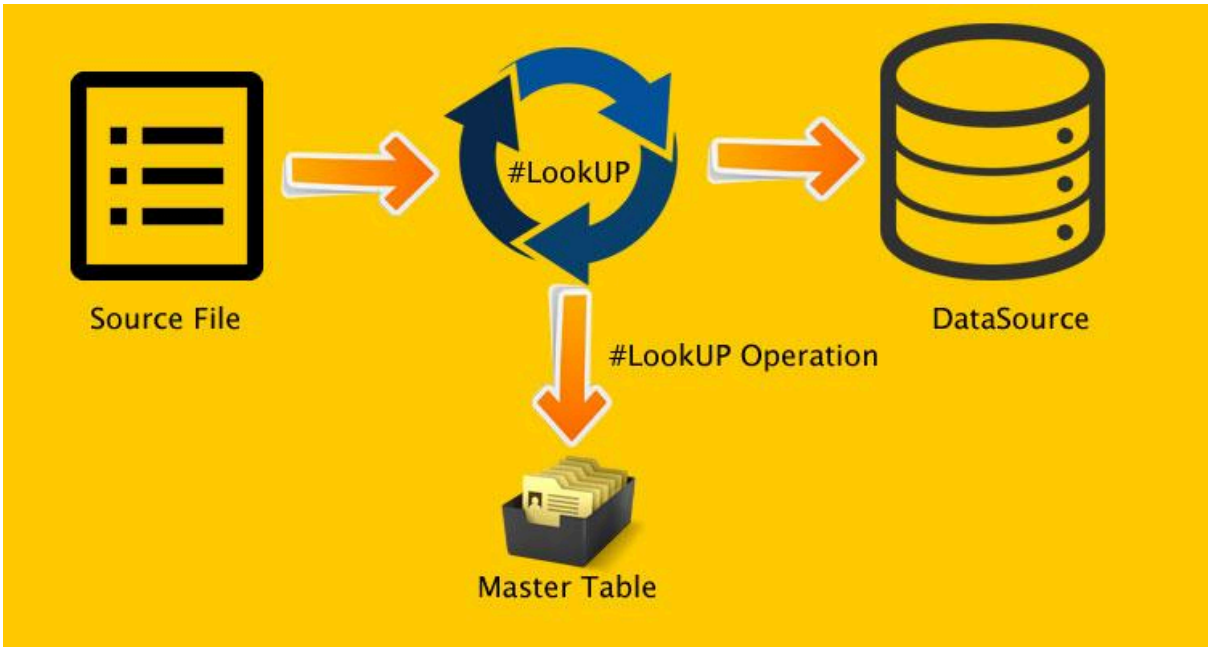
Cette requête met à jour tous les enregistrements ayant un "FLAG" égal à "0" en leur attribuant un "FLAG" de "1", indiquant ainsi que ces enregistrements ont été nettoyés.

Grâce à cette mise à jour de la colonne "FLAG", les enregistrements qui ont été nettoyés ne seront pas sujets à des opérations de nettoyage ultérieures, ce qui évite des opérations inutiles et contribue à maintenir l'intégrité et la qualité des données.

7. Transformation de données (LookUp)

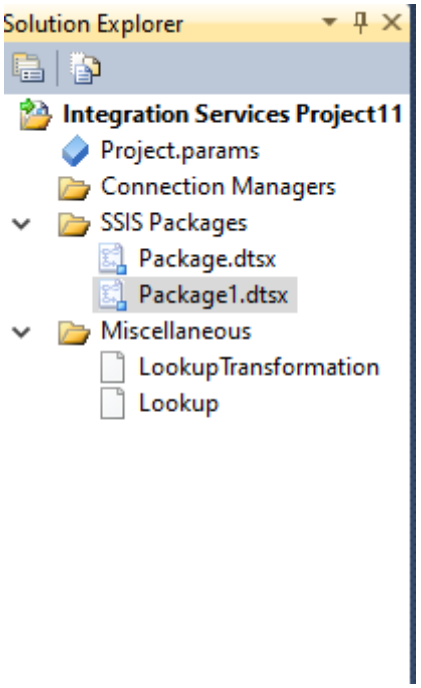
Vous pouvez configurer la transformation Lookup pour utiliser le mode de cache complet et un gestionnaire de connexion OLE DB. En mode cache complet, l'ensemble de données de référence est chargé dans le cache avant l'exécution de la transformation Lookup.

La transformation Lookup effectue des recherches en associant les données des colonnes d'entrée provenant d'une source de données connectée avec les colonnes d'un ensemble de données de référence.



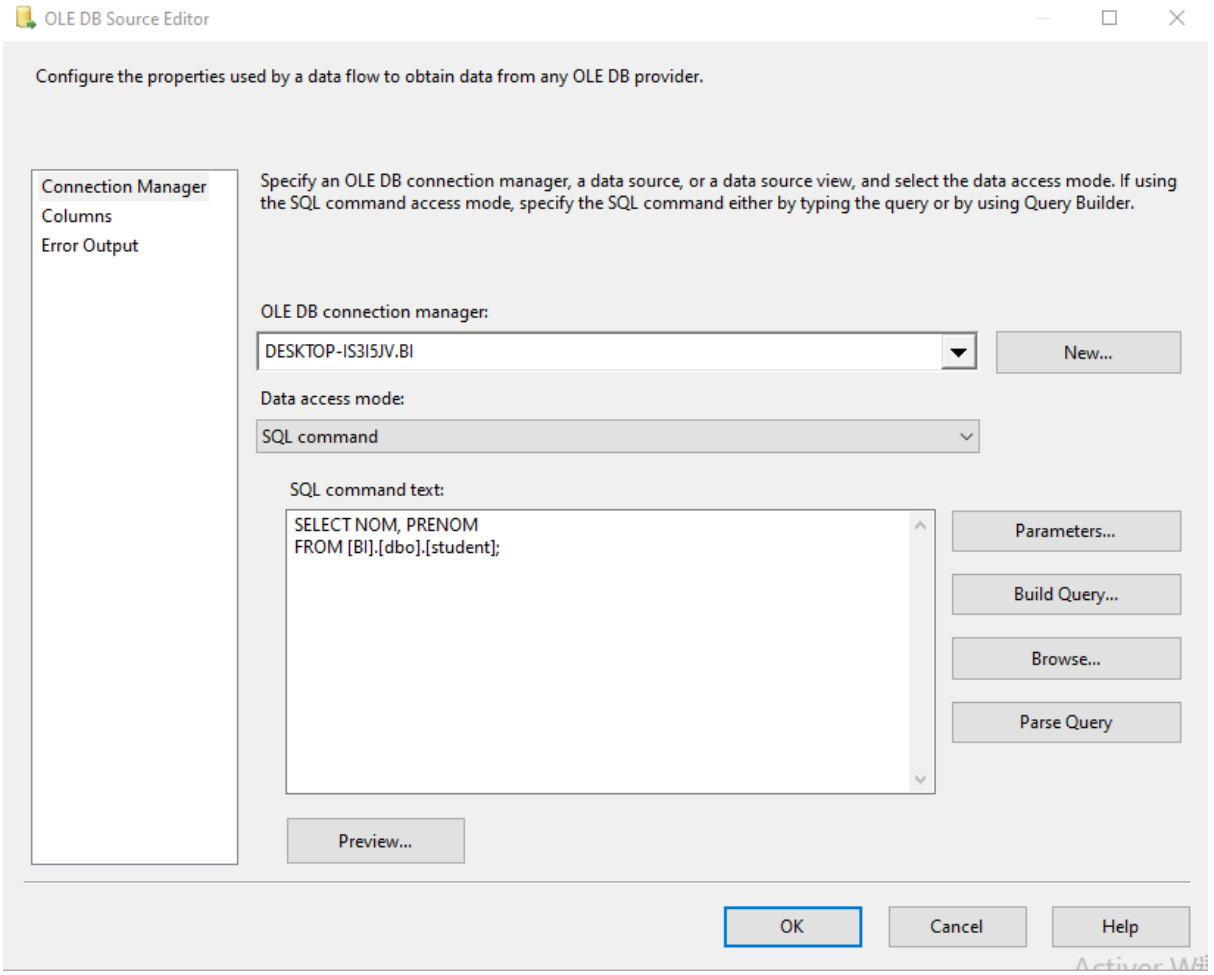
Maintenant l'objectif de l'utilisation de notre lookup est de pouvoir mettre à jour les noms ou les prénoms de la table Student tout en gardant les mises a jours affectés dans la table mère .

1. Dans le volet **Solutions explorer** faite un clic droit dans **packages** et **sélectionner Create a new package** , nommer le selon votre choix .



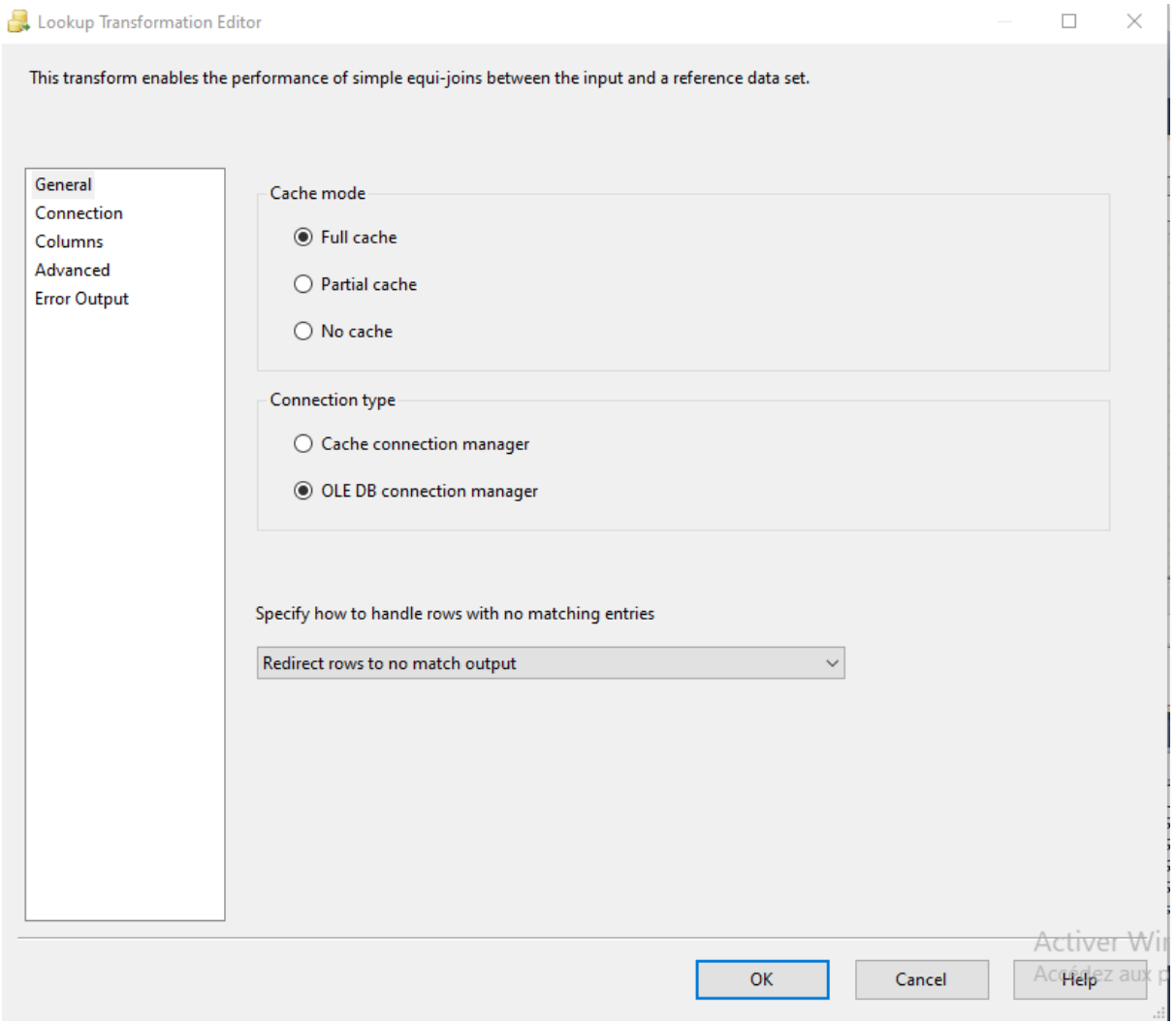
2. Ouvrez le package que vous avez créé et dans **control flow** faite glisser l'élément **dataflow task** et faites un double clique pour aller à **dataflow .**
3. faites glisser un **OLE BD SOURCE** et configurer votre connexion à votre base de donnés
4. Dans **OLEBD EDITOR** et dans le menu déroulant de **data access mode** choisissez **SQL COMMAND** et copiez cette requête qui permet de choisir seulement les colonnes NOM et PRENOM

   ```
   SELECT NOM, PRENOM
   FROM [BI].[dbo].[student];
   ```
5. Dans la section **COLUMNS** vérifier que les colonnes choisies ont été sélectionnées correctement .

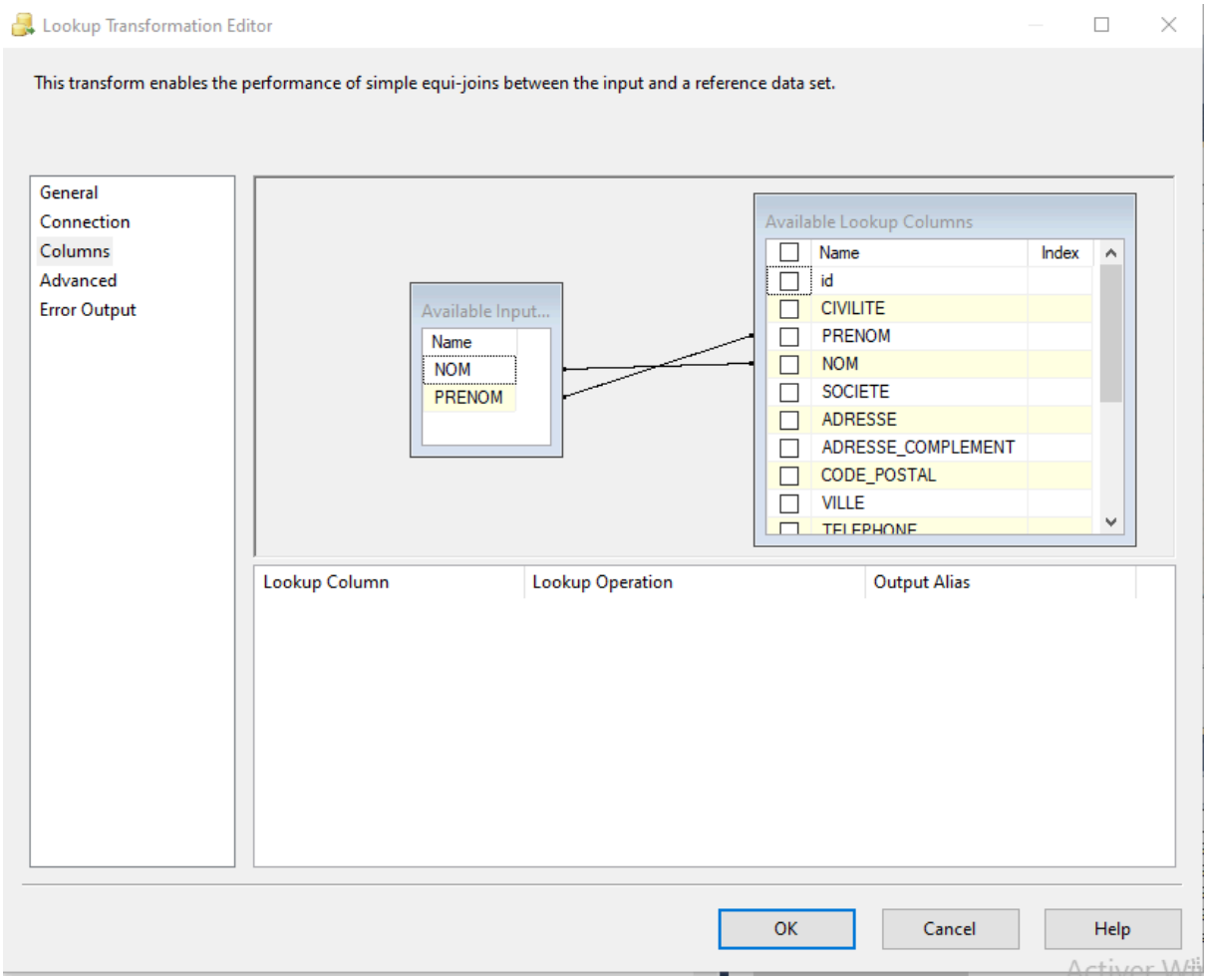


6. Maintenant dans le volet **dataflow** faites glisser un **lookup** et connecter le avec la flèche verte avec le **OLE DB SOURCE** et puis double cliquer sur **lookup** pour le configurer

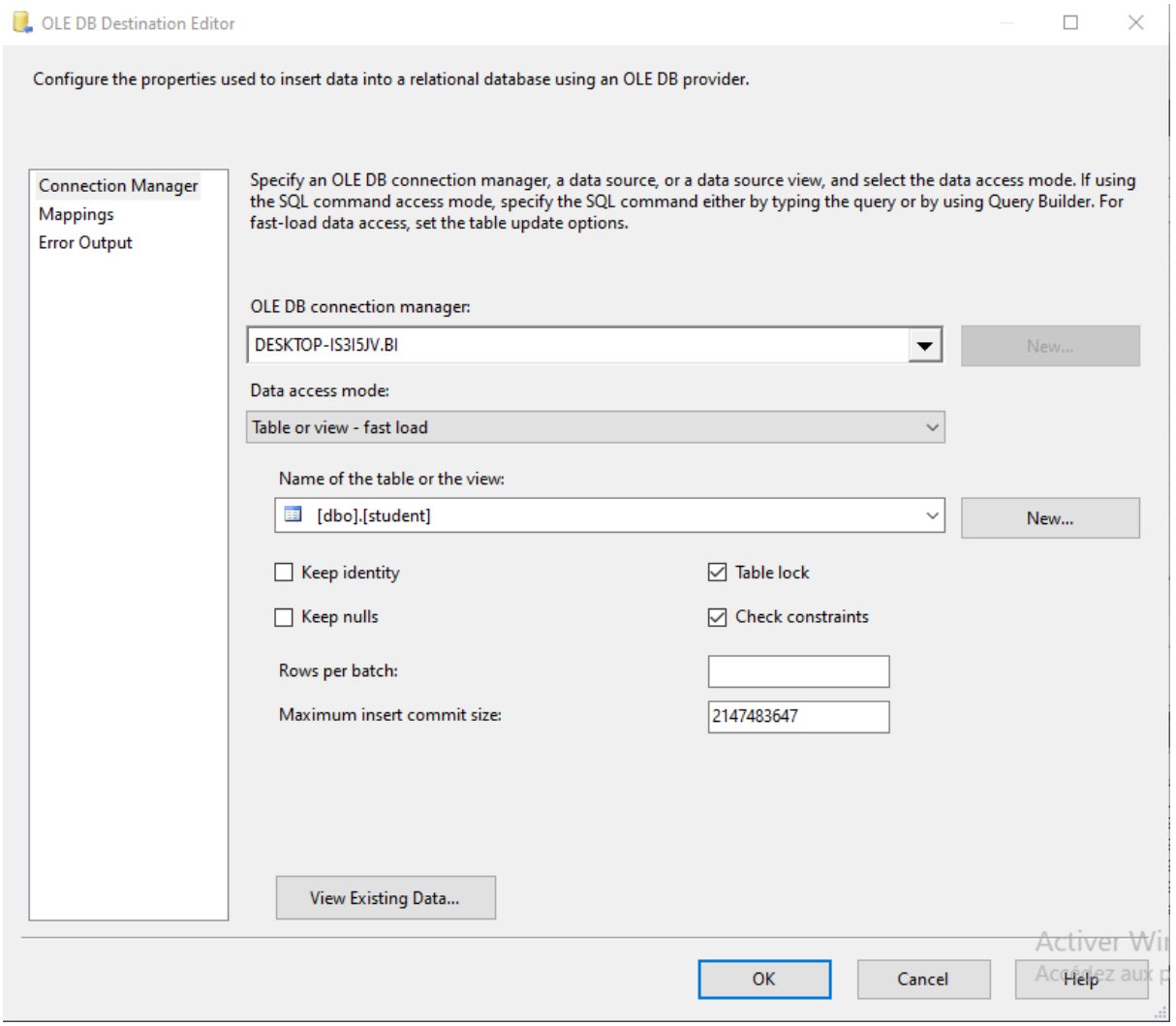
7. Dans **GENERAL** dans le menu déroulant de **specify how to handle rows with no matching entries** sélectionner **redirect rows with no much output**



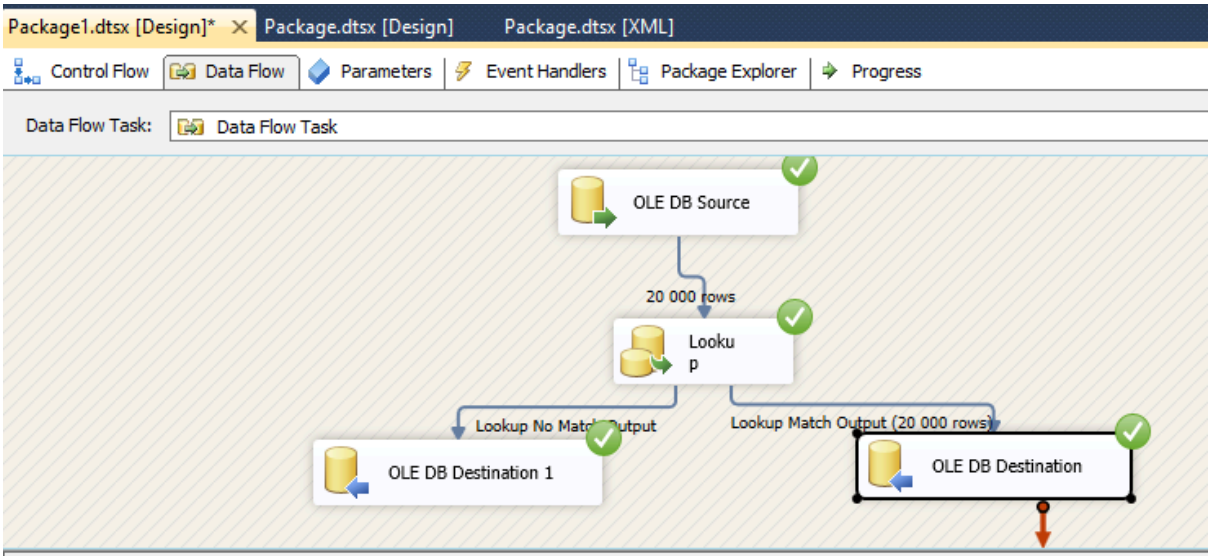
8. Dans **Connections** spécifier la base de donnés que vous souhaitez utiliser .
9. Dans **columns** faite glisser les colonnes NOM et PRENOM dans leur emplacement correspondant .



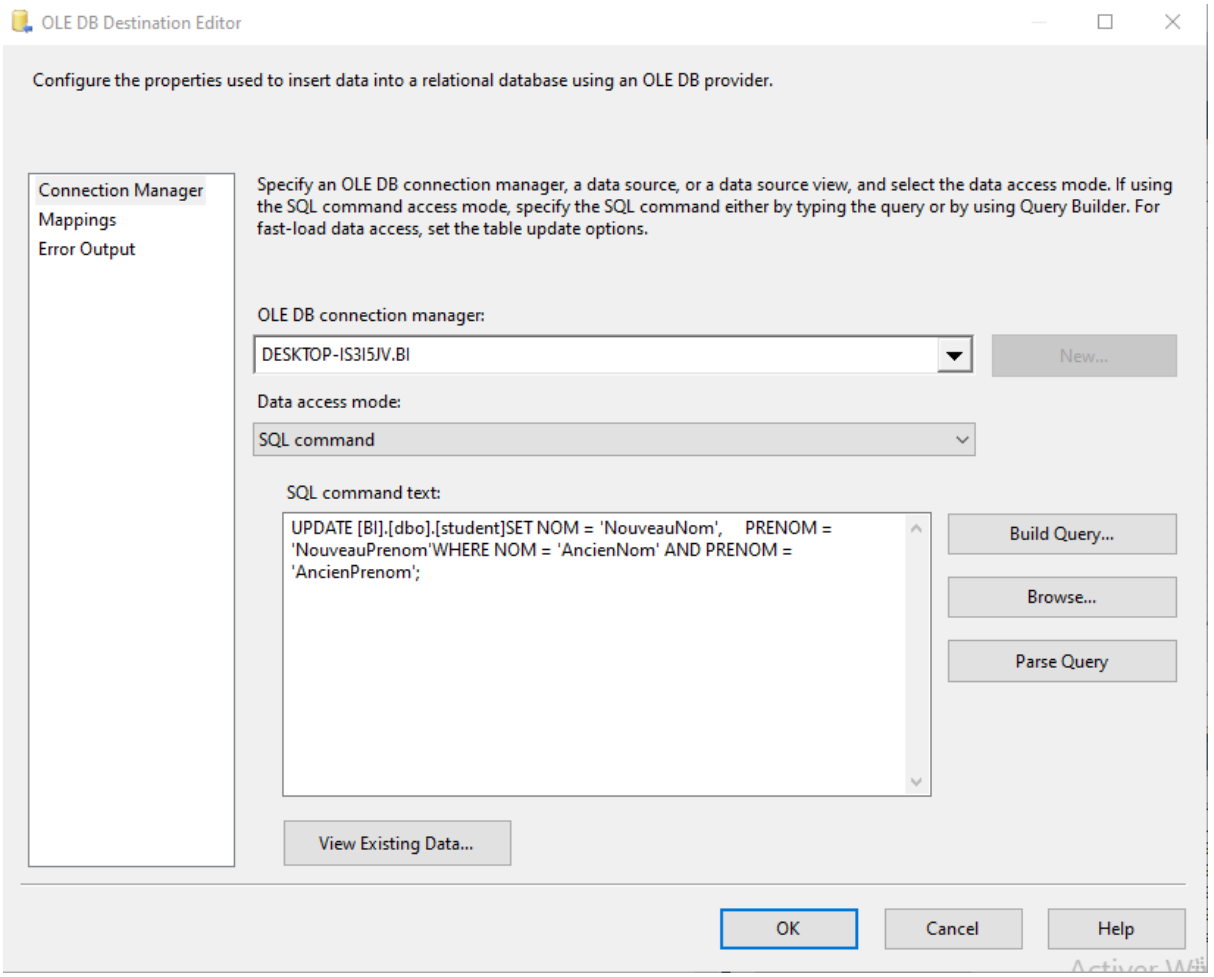
10. Maintenant dans le volet **Dataflow** faites glisser deux **OLE DB Destination** et les connecter avec le composant lookup avec les flèches bleues .
11. Double cliquez sur le premier **OLE DB Destination** et dans **connection manager** configurer le comme celle si



12. Faites la même chose pour le deuxième **OLE DB Destination** et le composant lookup va les diviser en des données correspondantes et non correspondantes .



13. Maintenant dans le **OLE BD destination** ou la valeur affiché est match double cliquez sur ce composant , dans **Connection manager** changer le **DATA ACCESS MODE** en **SQL COMMAND** et écrivez la requête SQL suivante



14. Cliquer sur **OK**

Maintenant si on souhaite mettre à jour le NOM ou le PRENOM d'un étudiant il suffit de mettre dans le **SQL COMMAND TEXT** l'ancien nom (ou prénom) et le nouveau nom (ou prénom) et exécuter le package .

```
UPDATE [BI].[dbo].[student]
SET NOM = 'NouveauNom',
    PRENOM = 'NouveauPrenom'
WHERE NOM = 'AncienNom' AND PRENOM = 'AncienPrenom';
```