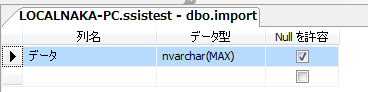
Demo1

新しいデータベースを作成する

ssistest

新しいテーブルを作成する



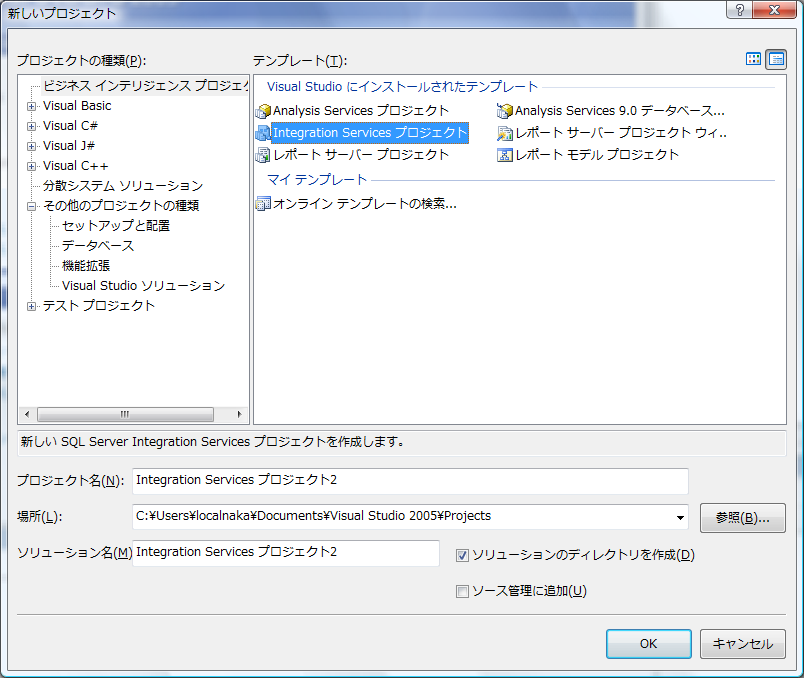
テーブル名はimport

元ネタ.txtを見せる

Visual Studioを立ち上げる

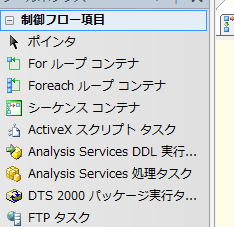
作成-プロジェクト

ビジネスインテリジェンス-ISプロジェクト

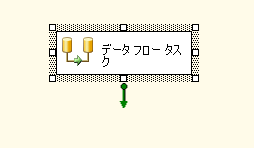


デフォルトで作成

ツールボックスの中身を見せて



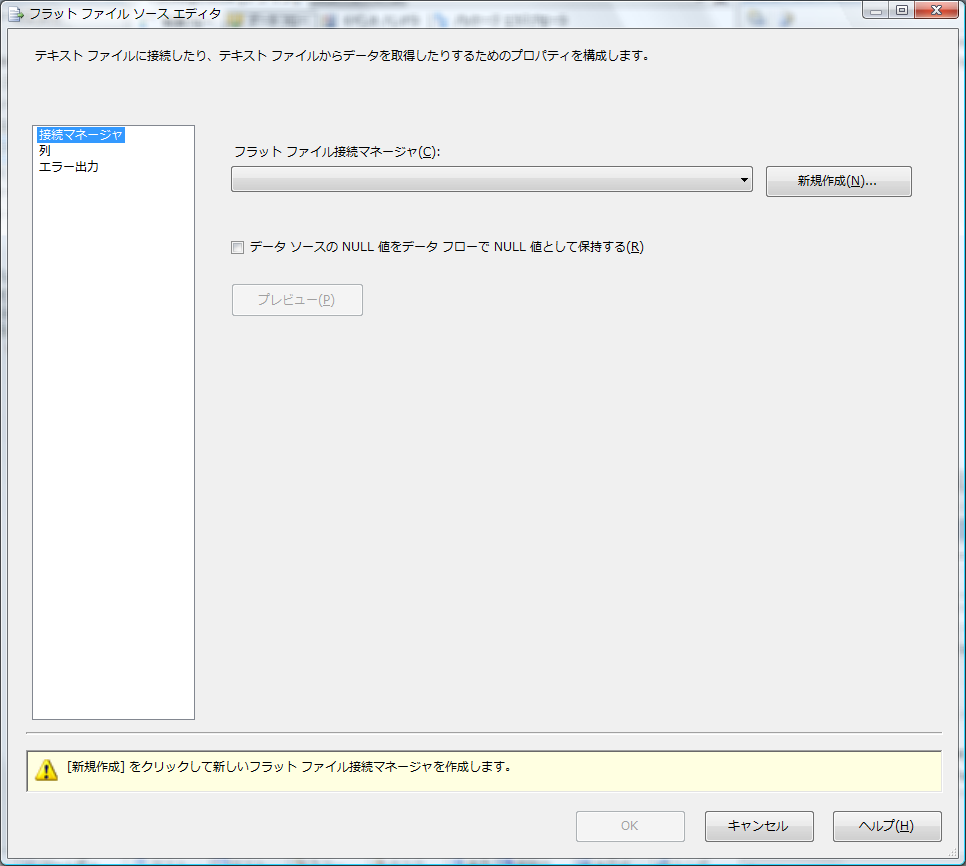
データフローを落とす



ダブルクリックして

フラットファイルから読み込むので、変換元にフラットファイルソースを配置

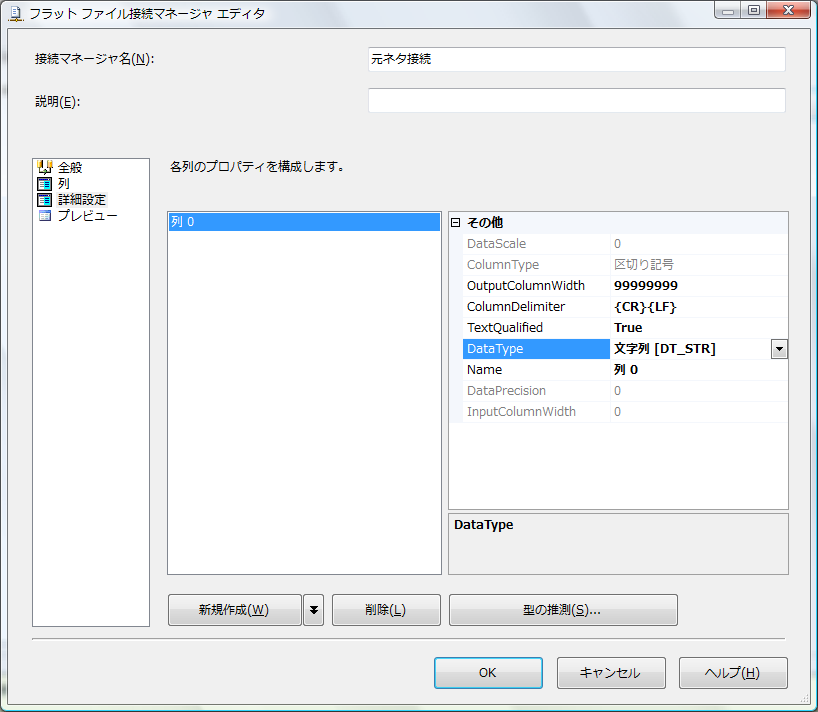
ダブルクリックで編集



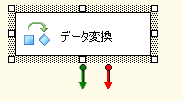
接続マネージャがないので、新規作成

各種設定をするが、詳細設定列だけ入念に

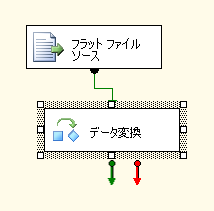
OutputColumnWidthは8000にしておく



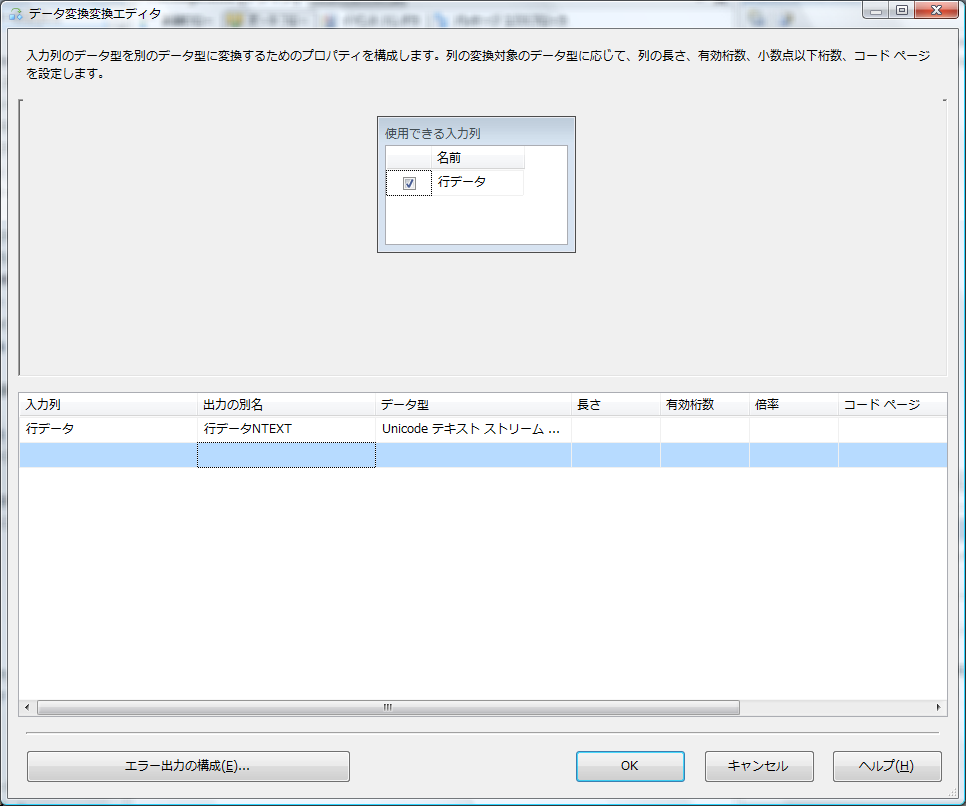
次にデータ変換タスクを置く



つなぐ



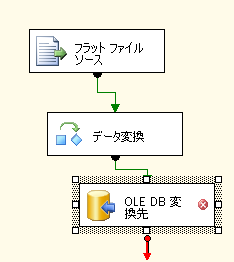
データ変換でN\_TEXTに変換する



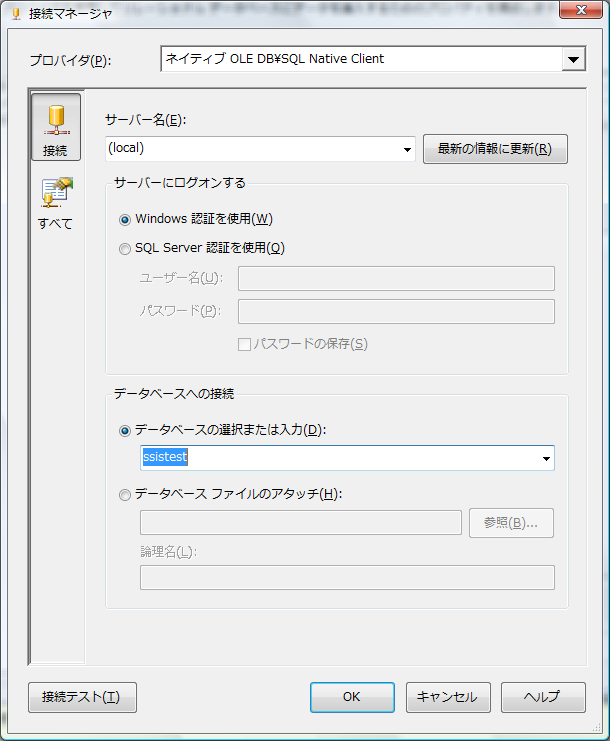
次はOLEDB変換先を置く



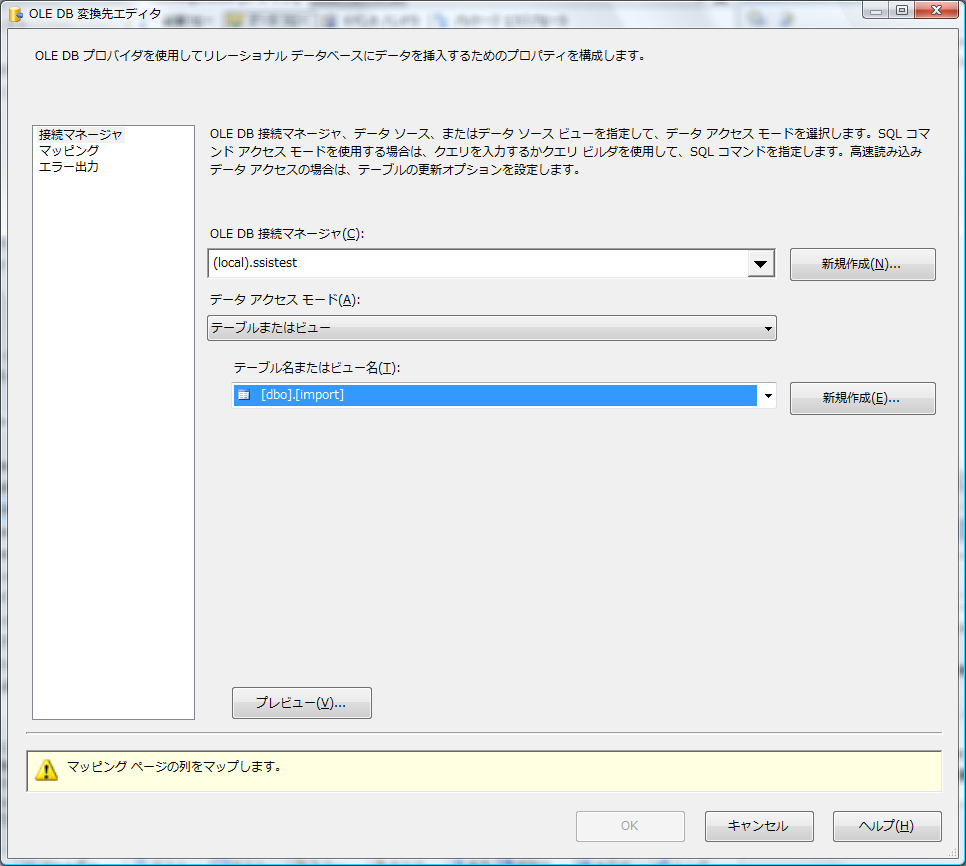
つなぐ



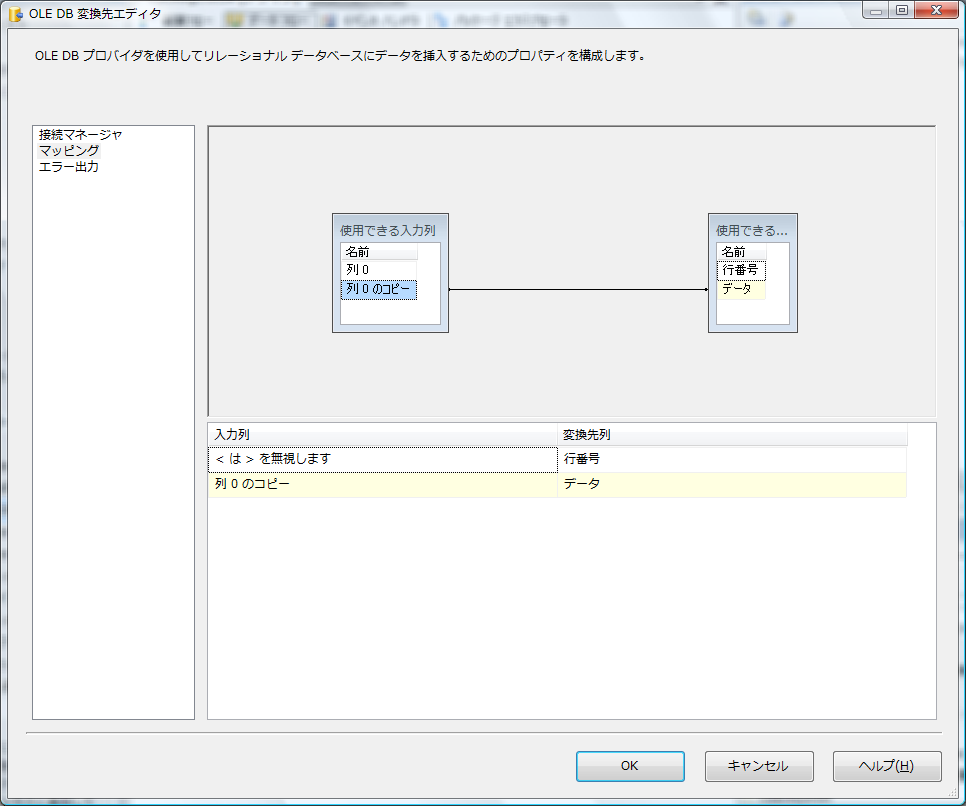
接続先を設定する



カンタンチン



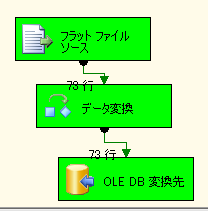
マッピング



実行や

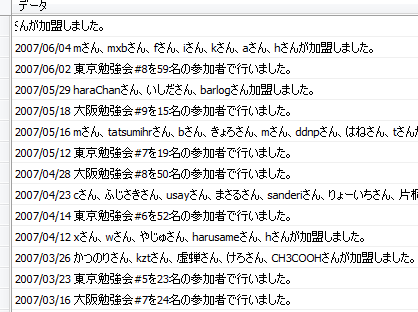


緑は成功



SSMSで確認

よっしゃできてるで。

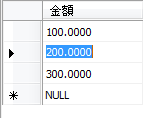


Demo2

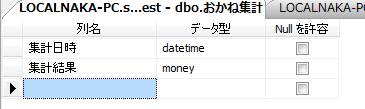
おかねテーブル



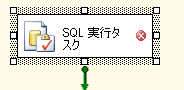
テストデータを入れておく



おかね集計テーブル



SQL実行タスクをぽとぺた



コネクションを設定する

コマンドを設定する

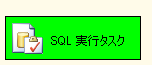
insert into おかね集計

select getdate(), sum(金額) from おかね

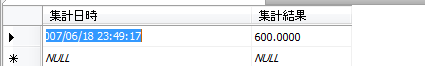
実行



とりあえず成功



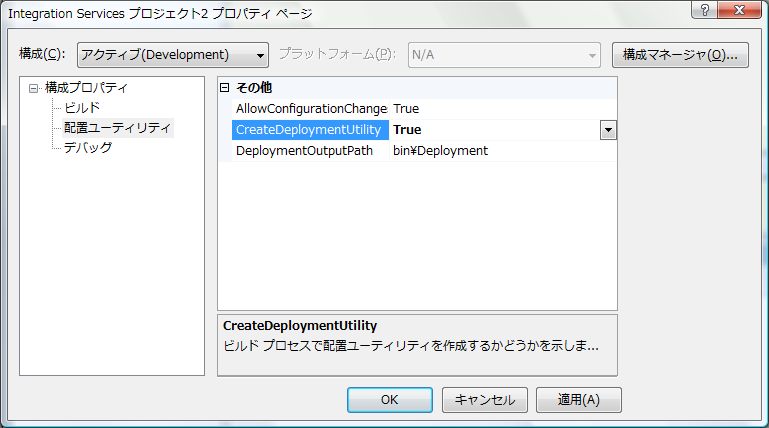
はいってまーす



パッケージをインストールしましょう

プロジェクト→プロパティ

CreateDeploymentUtility=True



Ctrl+Shift+B

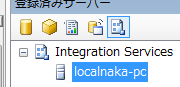
Explorerでさぐりあてる

.SSISDeploymentManifestをクリック

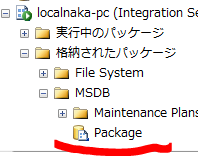


完了

Integration Serviceに接続



入っていることを確認する



エージェントの有効

exec sp\_configure 'Agent XPs', 1

go

reconfigure

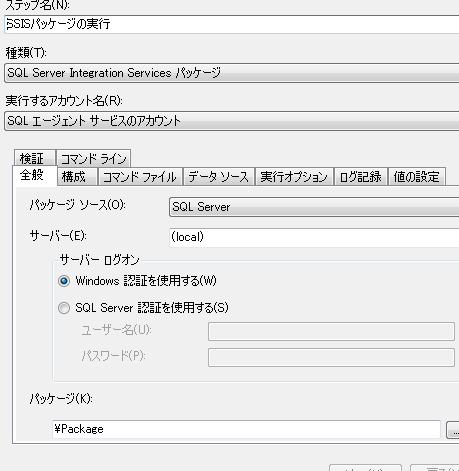
go

エージェントを起動して

ジョブの新規作成



ステップの新規作成

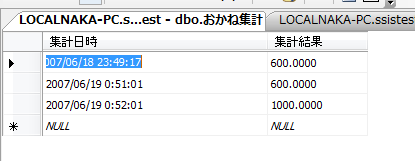


スケジュールの作成

定期的,毎日,間隔1分

これで１分に１回流れ始めます。

おかねを追加してみたりして、集計が変わることを確認しましょう

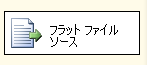


はい大成功

ジョブとパッケージは削除しておきましょう。

デモ３

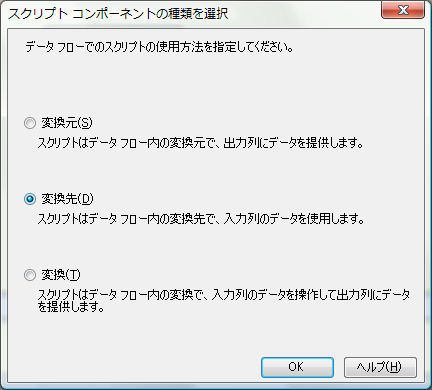
もう一度フラットファイルソースを作成する。



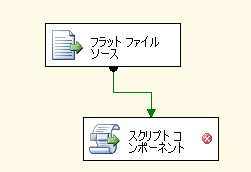
スクリプトコンポーネントをDD



変換を選ぶ

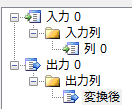


つなぐ



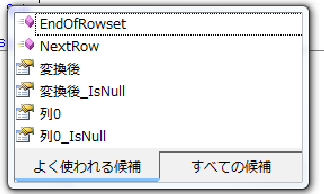
出力列を作る

型はNTEXT



スクリプトのデザイン

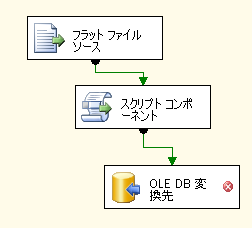
Row.



参照設定もできるよー

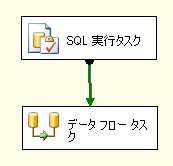
適当に編集するようにする

OLE DBに出力する



もちろんマッピングを忘れずに

あそうそう、データが残っちゃってるから、毎回消しましょうね。

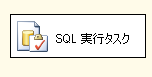


実行



デモ4

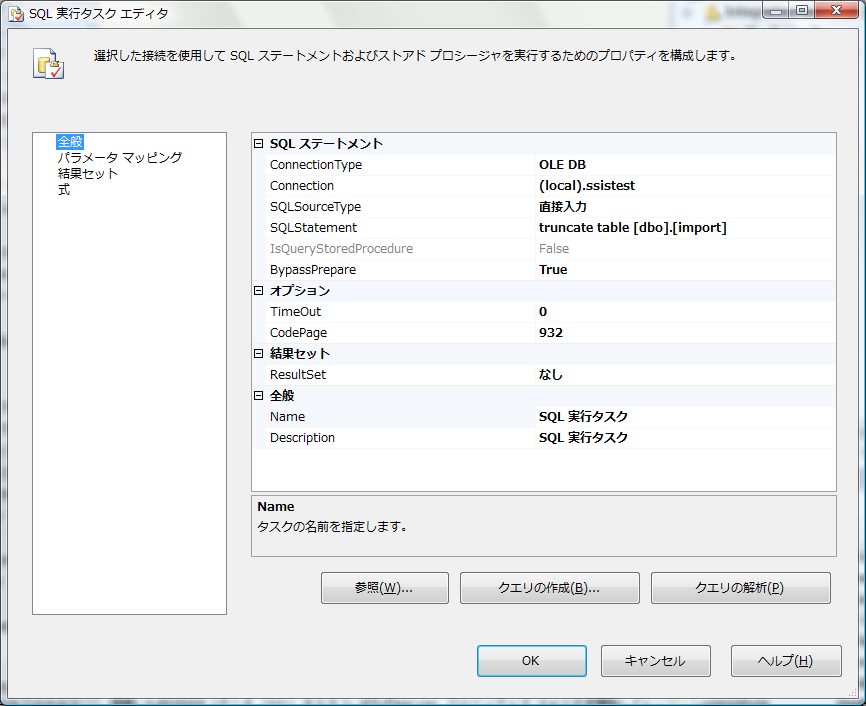
SQL実行タスクを追加



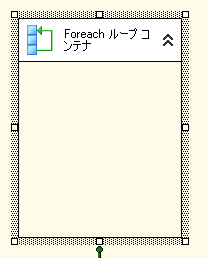
Connectionを設定

SQLStatementにtruncate table [dbo].[import]

を記述

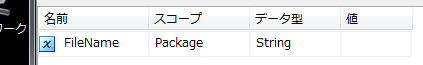


For-Eachコンテナを置く

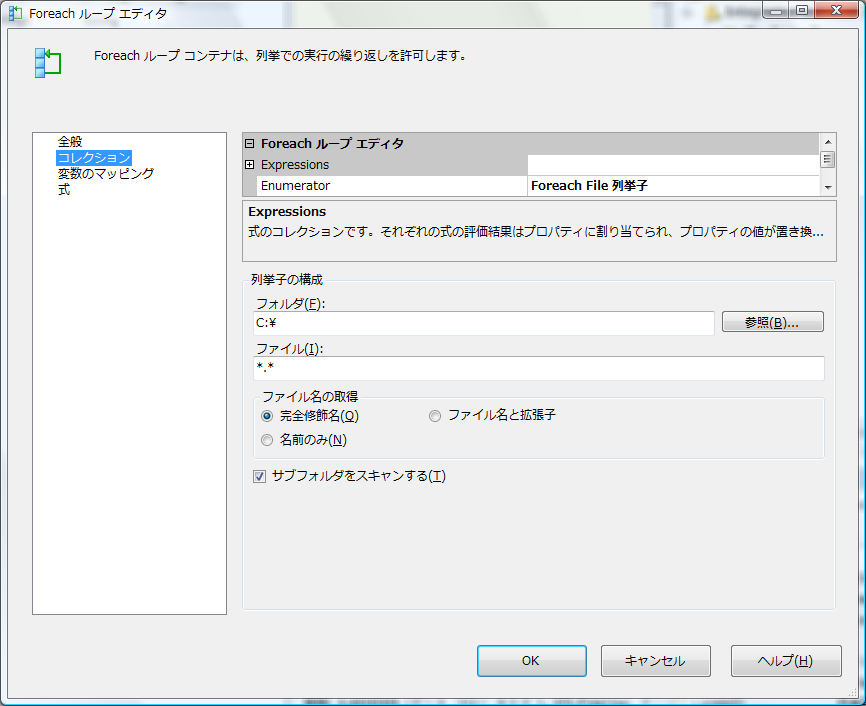


つなぐ

変数を作成する



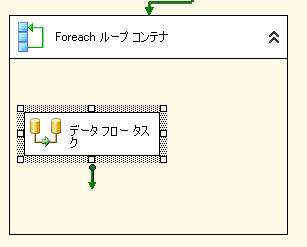
ファイルの一覧を取得するところを設定する



変数のマッピングをする

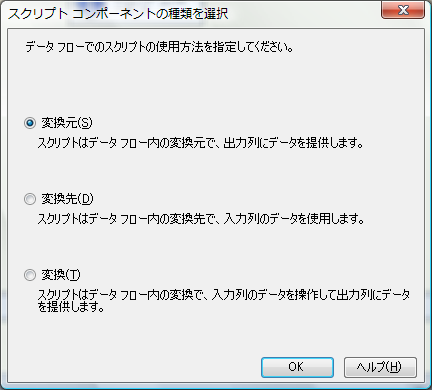


次にコンテナ内にデータフロータスクを入れる

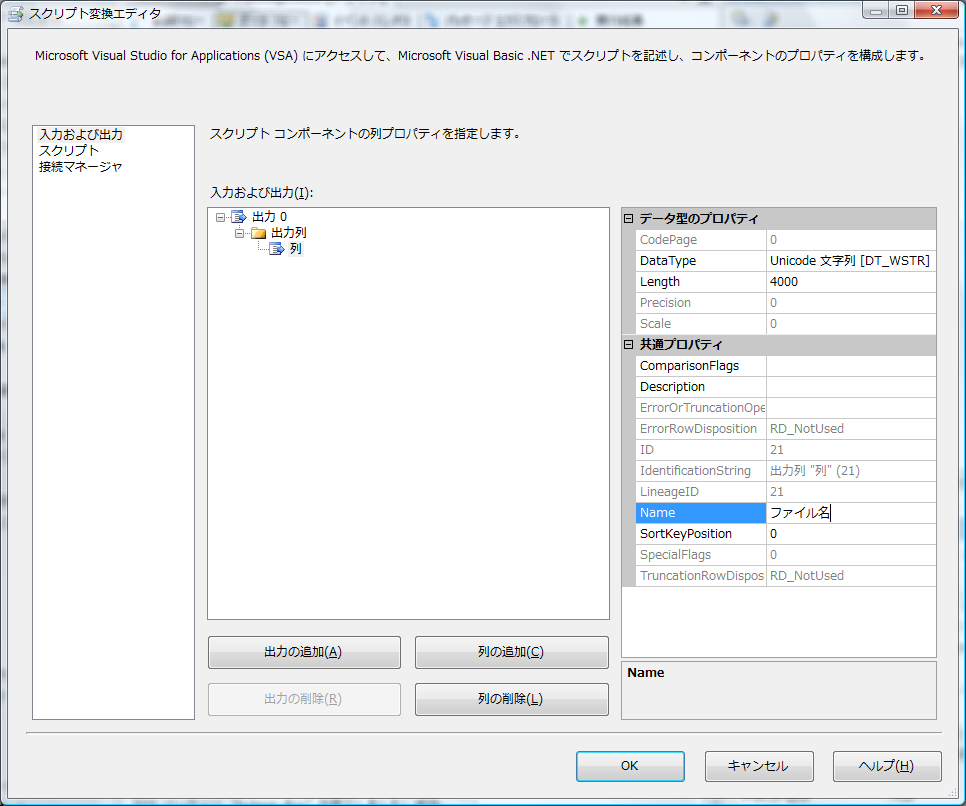


スクリプトコンポーネントを置く

今回は変換元

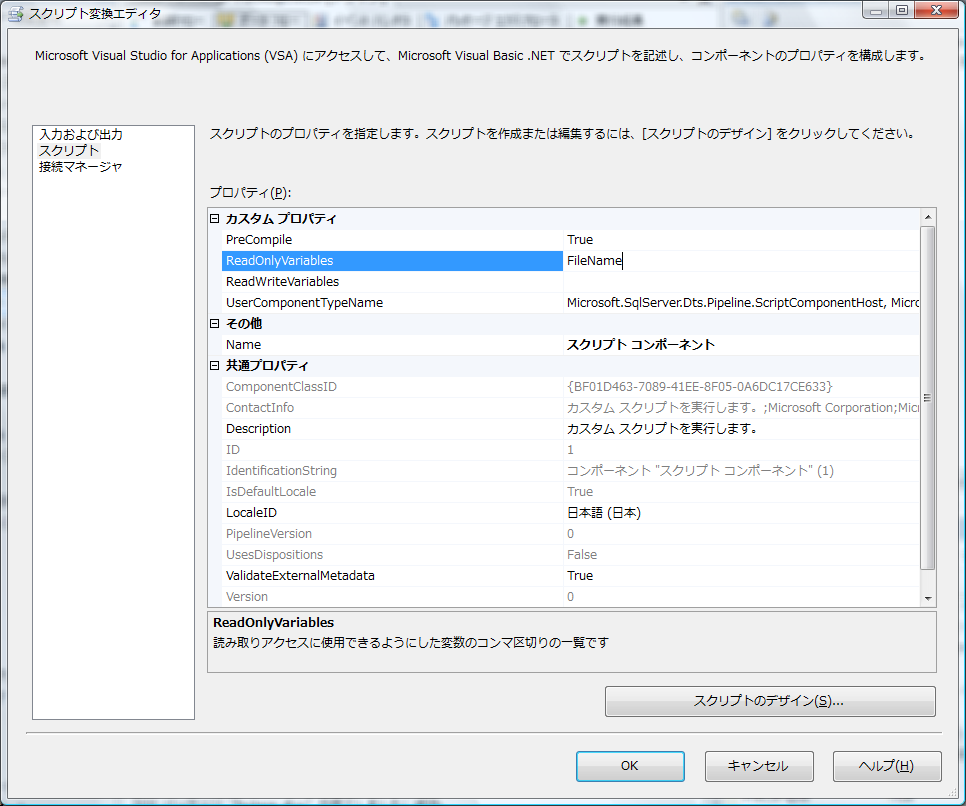


列追加



今回はDT\_WSTR 4000 Name=ファイル名

ReadOnlyVariablesにFileNameを記述



スクリプトに移動

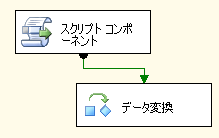
コードを書く

Me.出力Buffer.AddRow()

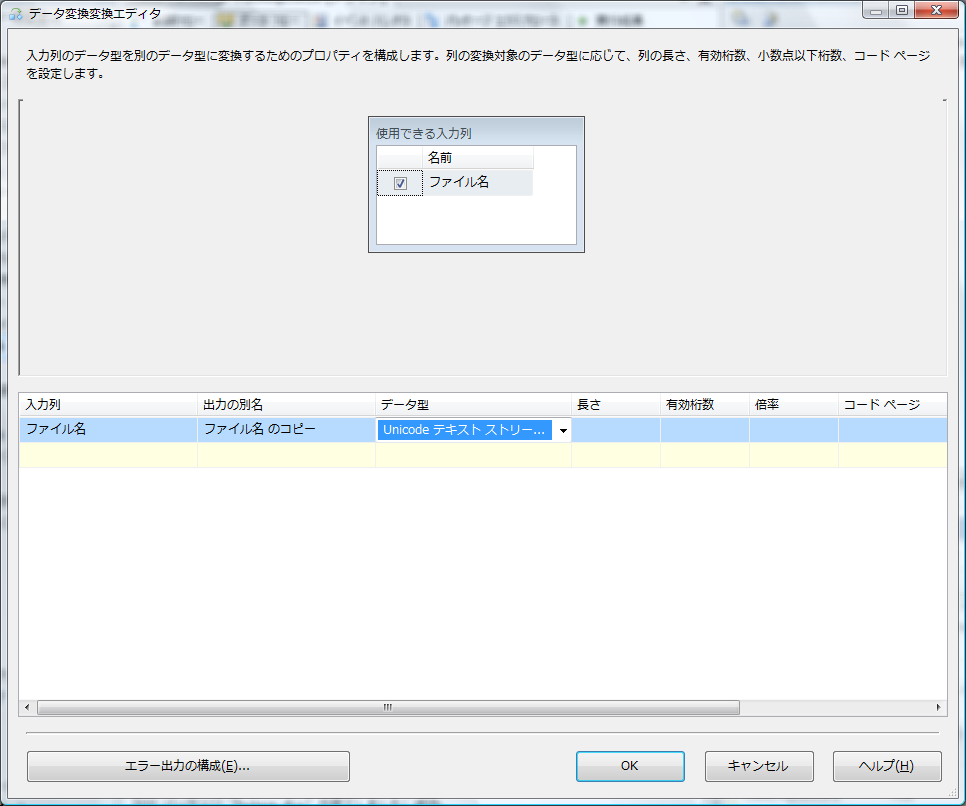
Me.出力Buffer.ファイル名= Me.Variables.FileName

AddRow忘れるな

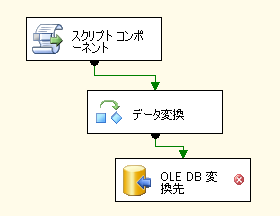
データ変換を置いてつなぐ



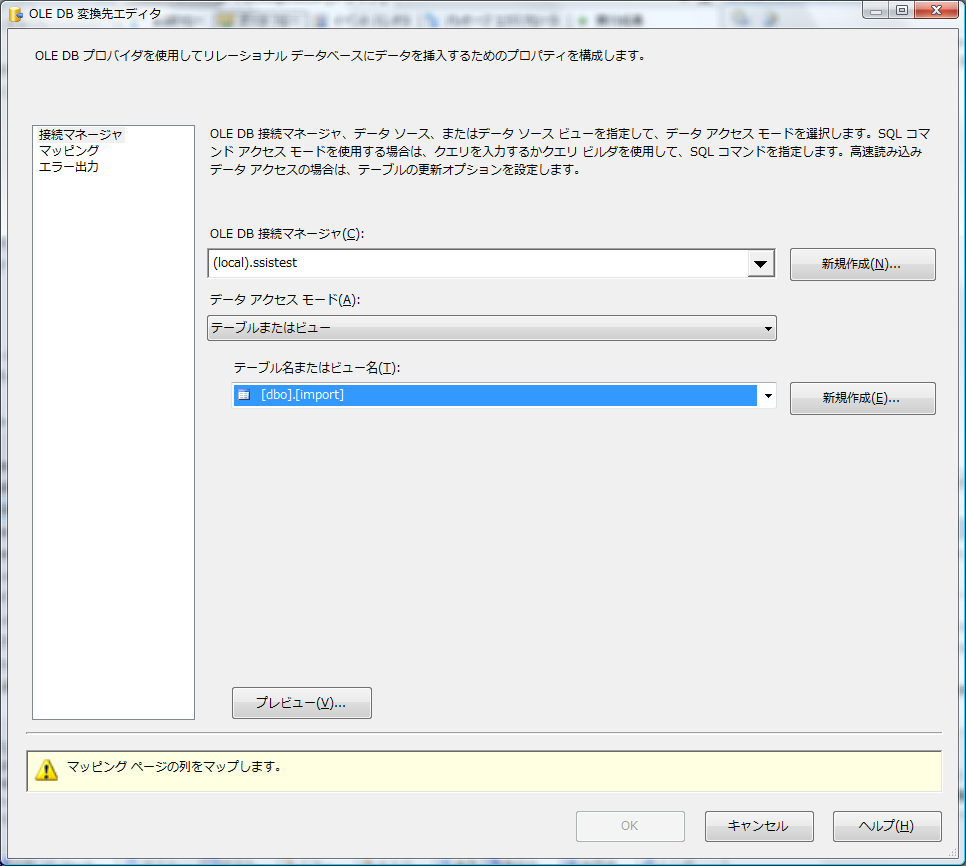
DT\_WSTRをDT\_NTEXTに変換



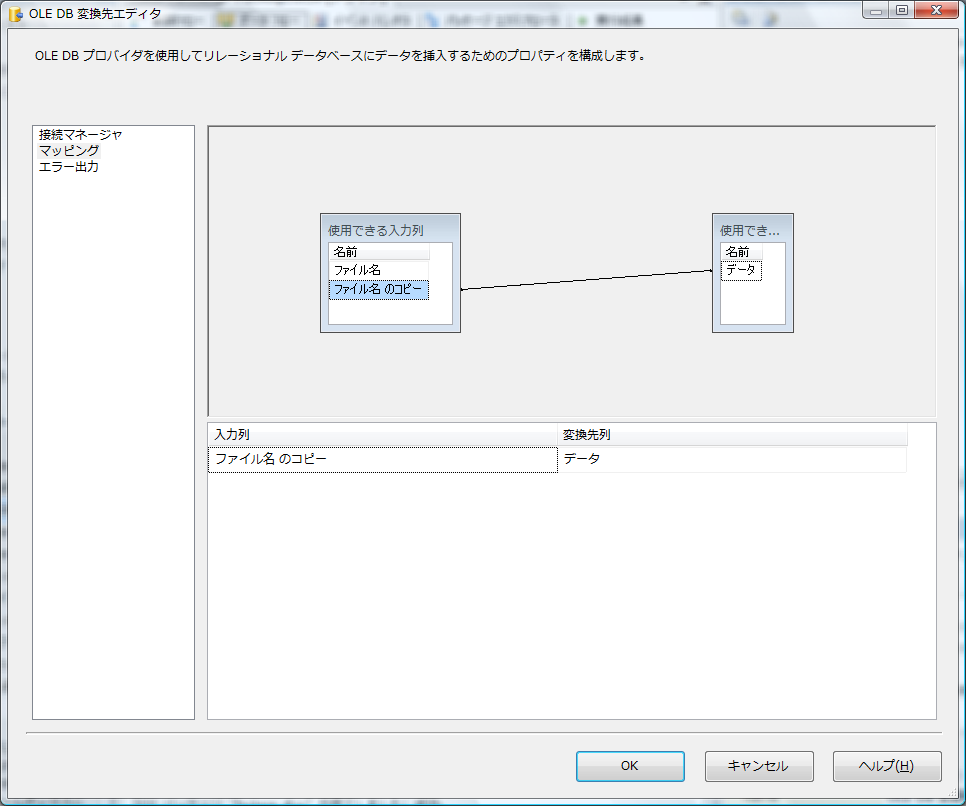
OLEDB変換先を置いて、つなぐ



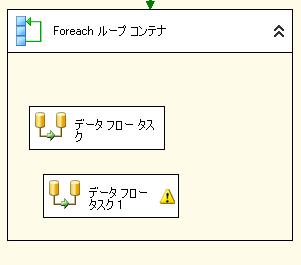
テーブルimportを指定



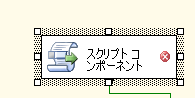
マッピングする



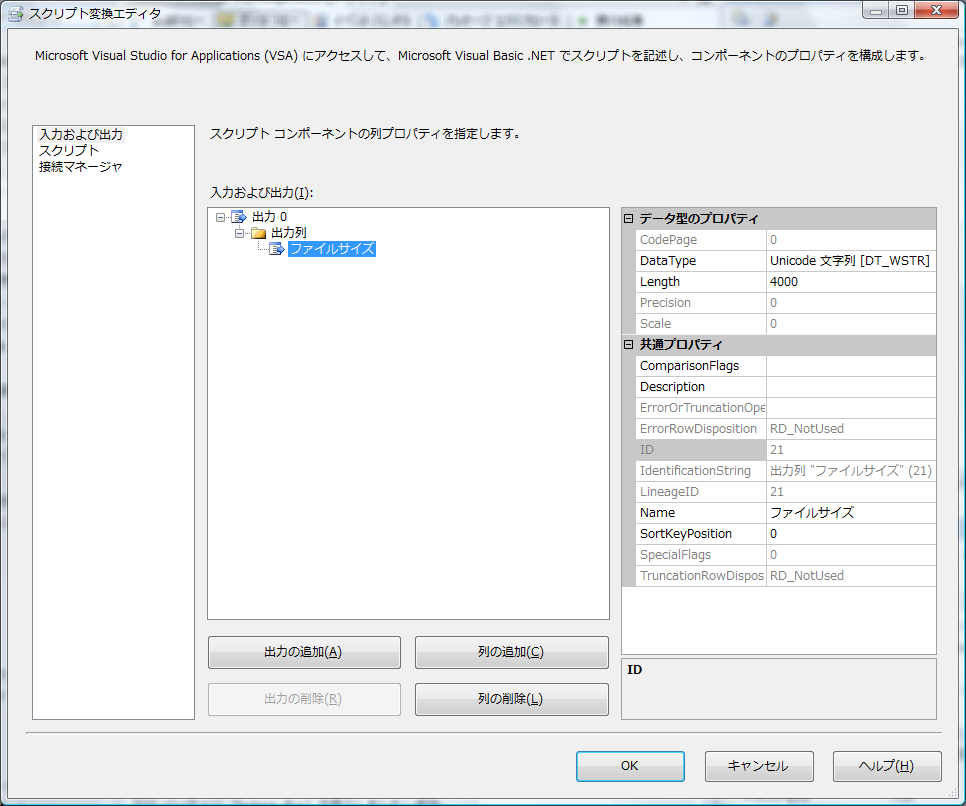
制御フローに戻って、データフローをこぴぺ



スクリプトを開く



出力列をファイル名からファイルサイズに変更



コードを書く

Me.出力Buffer.AddRow()

Me.出力Buffer.ファイルサイズ= New System.IO.FileInfo(Me.Variables.FileName).Length.ToString()

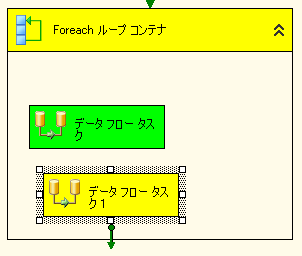
データ変換で変数名を変更しましょうね。

出力先も確認しましょう。・

実行



じらじら並列実行



こんな感じで完全に並列処理成功

