Instytut Sterowania i Systemów Informatycznych
Uniwersytet Zielonogórski

# Integracja systemów

Laboratorium 9
Prowadzący: Marek Kowal
(M.Kowal@issi.uz.zgora.pl)

## Handling Bad Data with the Fuzzy Lookup

More often than not, when you are working in the real world, data is not going to be perfect like it is in the **AdventureWorks2014** database. Real-world situations call for cleansing dirty data or data that has abnormalities like misspellings or truncation.

Imagine you are attempting to retrieve a foreign key from a dimension table, but, strangely, you find rows without a match. Upon investigation, you find bad data is being supplied to you. One technique might be to divert these rows without matches to a table to be dealt with later; another might be to just add the bad data regardless of misspellings and other mishaps that occur during data entry.

The **Fuzzy Lookup Transform**, and the **Fuzzy Grouping Transform** gives other alternatives to dealing with dirty data while reducing your number of unmatched rows. The **Fuzzy Lookup Transform** matches input records with data that has already been cleansed in a reference table. It returns the match and can also indicate the quality of the match. This way you know the likelihood of the match being correct.

**NOTE**: A best practice tip is to use the **Fuzzy Lookup Transform** only after trying a regular lookup on the field first. The **Fuzzy Lookup Transform** is a very expensive operation that builds specialized indexes of the input stream and the reference data for comparison purposes. Therefore, it is recommended to first use a regular **Lookup Transform** and then divert only those rows not matching to the **Fuzzy Lookup Transform**.

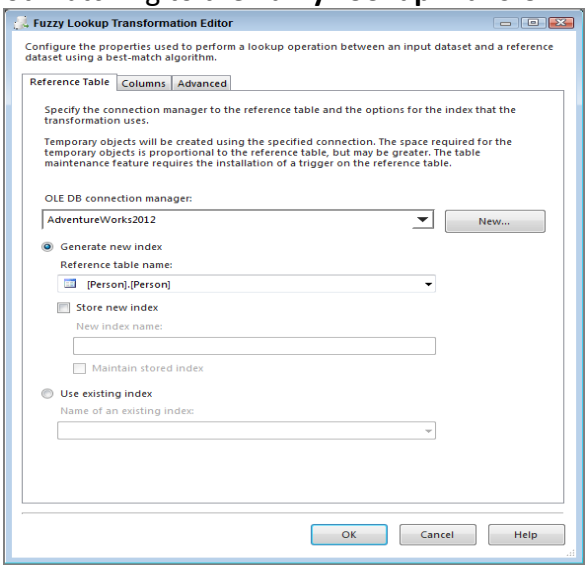


Fig. 1

Using the **Fuzzy Lookup Transform** requires at least one field to be a string, either a **DT_WSTR** or **DT_STR** data type. On the **Columns** tab in the editor, you need to map at least one text field from the input to the reference table for comparison.

The **Advanced** tab contains the settings that control the fuzzy logic algorithms. You can set the maximum number of matches to output per incoming row. The default is set to 1, which pulls only the best record out of the reference table that meets the similarity threshold. Incrementing this setting higher than the default might generate more results that you'll have to sift through, but it might be required if you have too many closely matching strings in your data. A slider controls the similarity threshold. When you are experimenting, a good strategy is to start this setting at 0.5 and move up or down as you review the results. This setting is normally decided based on a businessperson's review of the data, not the developer's review. If a row cannot be found that's similar enough, the columns that you checked in the **Columns** tab will be set to **NULL**. The token delimiters can also be set if, for example, you don't want the comparison process to break up incoming strings with a period (.) or spaces. The default for this setting is all common delimiters. See Fig. 2 for an example of an **Advanced** tab.

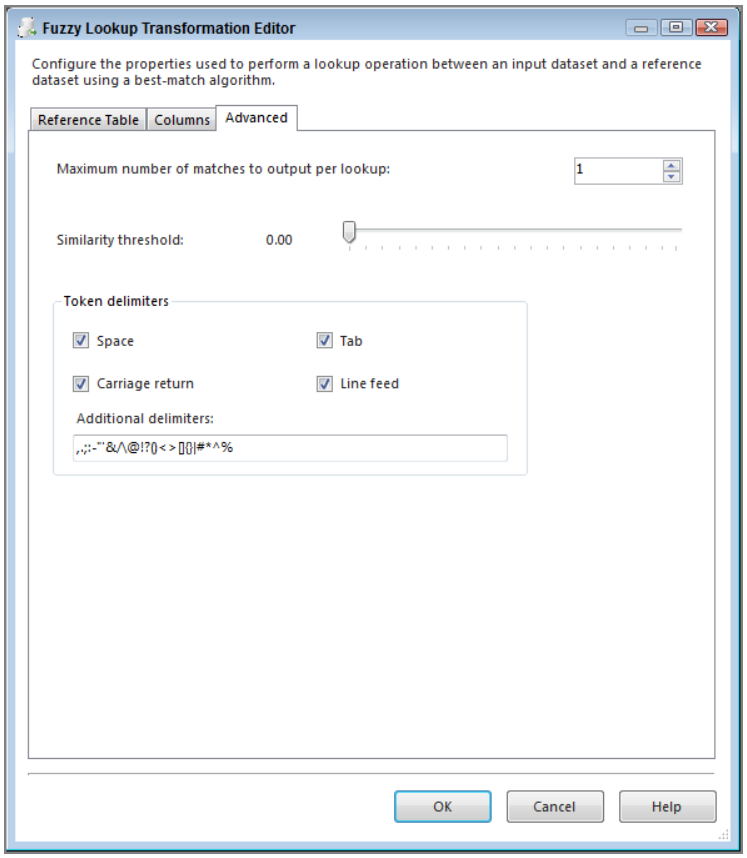


Fig. 2

The transform creates several output columns that you may or may not decide are useful to store in a table. Either way, they are important to understand:

- **Input and Pass-Through Field Names and Values**—This column contains the name and value of the text input provided to the **Fuzzy Lookup Transform** or passed through during the lookup.

- **Reference Field Name and Value**—This column contains the name and value(s) of the matched results from the reference table.

- **Similarity**—This column contains a number between 0 and 1 representing similarity. Similarity is a threshold calculated by comparing one word with another; you set this when configuring the **Fuzzy Lookup Transform**. The closer this number is to 1, the closer the two text fields match. A similarity of 1 would indicate an exact match.

- **Confidence**—This column contains a number between 0 and 1 representing confidence of the match relative to the set of matched results. Confidence is different from similarity; it is not calculated by comparing just one string against another, but rather by comparing the chosen string match against all the other possible matches. Confidence gets better the more accurately your reference data represents your subject domain, and it can change based on the sample of the data coming into the ETL process.

You may not want to use each of these fields, but it is important to appreciate the value they could provide.

1. **Cel ćwiczenia**

   You use the **Fuzzy Lookup Transform** to attempt to correct some bad data that you receive in a flat file. After this lesson, you should have an idea of how useful the **Fuzzy Lookup Transform** can be in cleansing your data.

2. **Przebieg ćwiczenia**

   1) Create a new package

   2) Drag a **Data Flow Task** onto your designer and name it **DFT - Fuzzy Lookup**.

   3) Create a new **Flat File Connection Manager** name it **New Employee**, and point it to **FuzzyExample.txt**. Check the Column names in the first data row option. The editor should look like Fig. 3.

   4) In the **Data Flow**, bring a new **Flat File Source** over and name it **New Employee Load**. Open the editor and make the connection manager the newly created **New Employee**.

   5) On the Columns tab, change the name of the output columns to **LastName**, **FirstName**, and **OccupationLabel**.

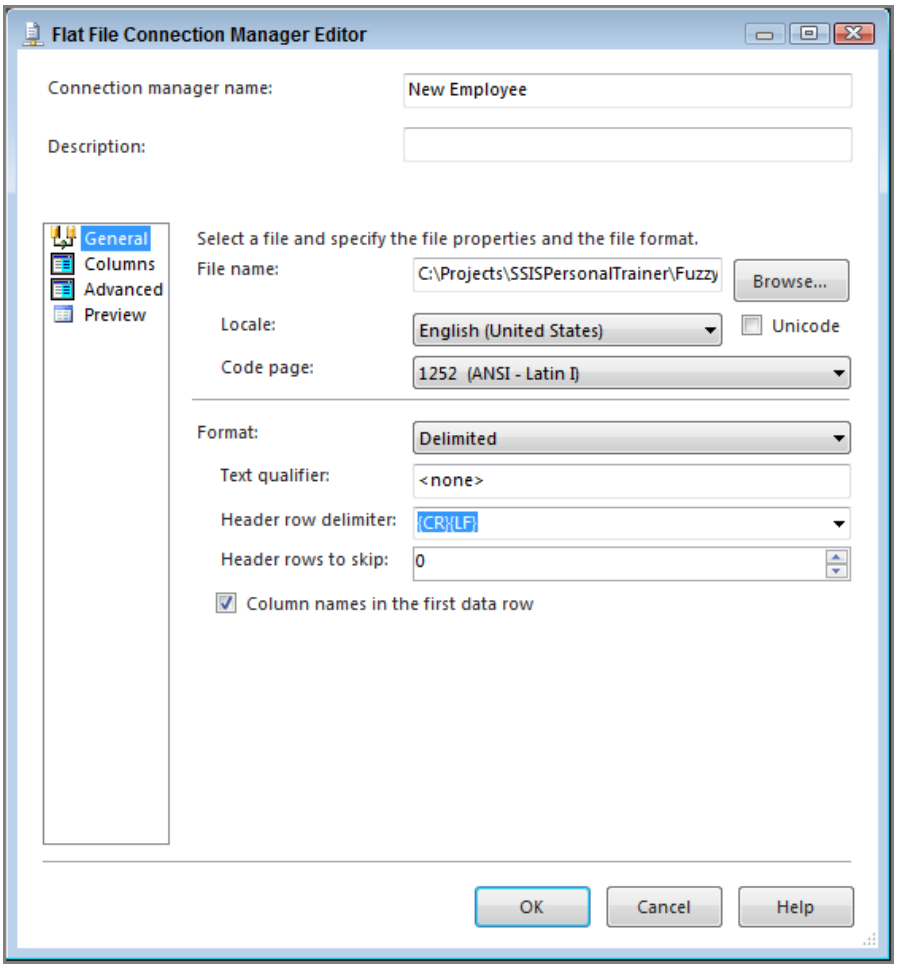


Fig. 3

6) Open **Management Studio** and run the following query to create a new table called **Occupation**.

```
CREATE TABLE [dbo].[Occupation](
	[OccupationID] [smallint] IDENTITY(1,1) NOT NULL,
	[OccupationLabel] [varchar](50) NOT NULL,
CONSTRAINT [PK_Occupation_OccupationID] PRIMARY KEY CLUSTERED
(
	[OccupationID] ASC
) ON [PRIMARY]
) ON [PRIMARY]
GO
INSERT INTO [dbo].[Occupation] Select 'CUSTOMER SERVICE REPRESENTATIVE'
INSERT INTO [dbo].[Occupation] Select 'SHIFT LEADER'
INSERT INTO [dbo].[Occupation] Select 'ASSISTANT MANAGER'
INSERT INTO [dbo].[Occupation] Select 'STORE MANAGER'
INSERT INTO [dbo].[Occupation] Select 'DISTRICT MANAGER'
INSERT INTO [dbo].[Occupation] Select 'REGIONAL MANAGER'
```

7) Next, create another connection manager, this time an **OLE DB Connection Manager**, using the **AdventureWorks2014** database.

8) Drag a **Lookup Transform** on the design surface and use the new **[dbo].[Occupation]** table to select the **OccupationID** based on the **OccupationLabel** that exists in both the source and the reference table. Fig. 4 shows what your mapping should look like. Lastly, before closing the editor, make sure to specify in the **General** tab that non-matching entries should redirect rows to no match output.

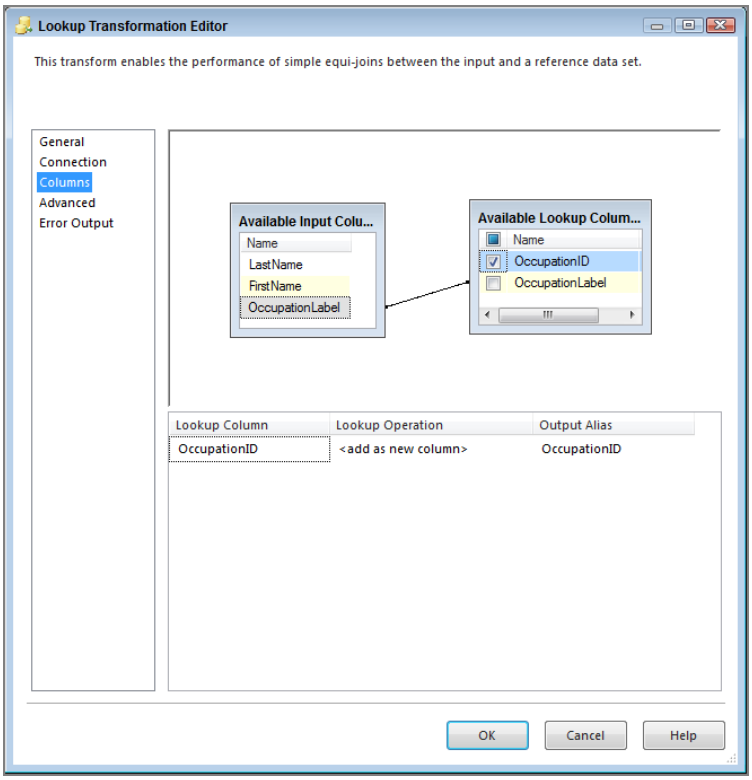


Fig. 4

9) You already know from the lesson description that the source data is dirty, so now you're going to use a **Fuzzy Lookup Transform** to catch all the bad data the regular Lookup doesn't recognize. Drag a new **Fuzzy Lookup Transform** in the **Data Flow** and connect the blue no match output arrow from the **Lookup Transform** to it.

10) Open the **Fuzzy Lookup** and select **[dbo].[Occupation]** for the **Reference table name** property. Fig. 5 shows the **Fuzzy Lookup Transformation Editor** using the **Occupation** table as the reference table.

11) The **Columns** tab should be joined by **OccupationLabel** as shown in Fig. 6. It should also return the **OccupationID** and **OccupationLabel** from the reference table, which you can ensure by checking the boxes in the **Available Lookup Columns** box. The **OccupationLabel** from the reference table should replace the same column from the input stream to correct bad data. To do this, uncheck the **OccupationLabel** column from the **Available Input Columns**

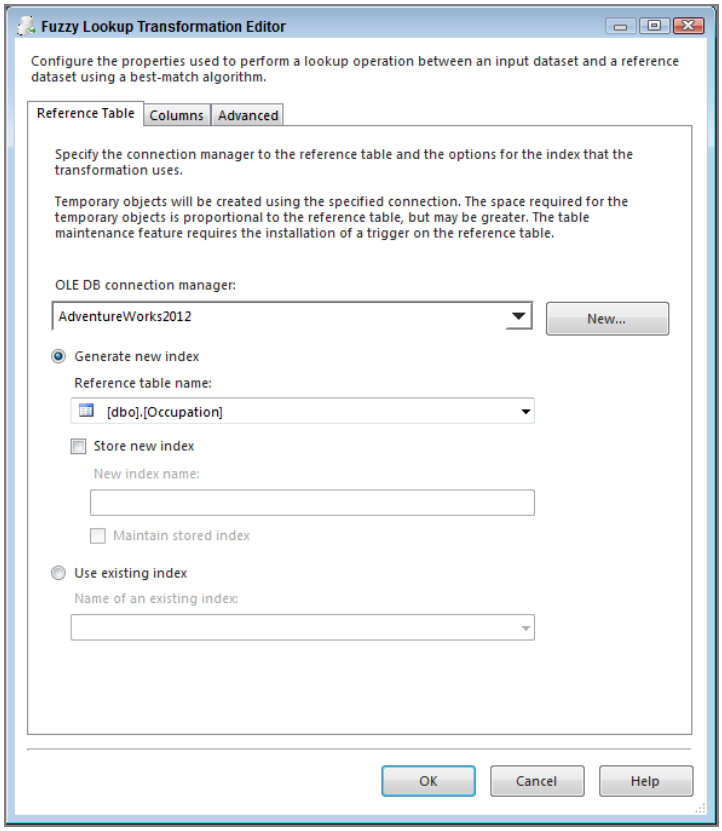
Fuzzy Lookup Transformation Editor
Configure the properties used to perform a lookup operation between an input dataset and a reference dataset using a best-match algorithm.
Reference Table
Columns
Advanced
Specify the connection manager to the reference table and the options for the index that the transformation uses.
Temporary objects will be created using the specified connection. The space required for the temporary objects is proportional to the reference table, but may be greater. The table maintenance feature requires the installation of a trigger on the reference table.
OLE DB connection manager:
AdventureWorks2012
New...
Generate new index
Reference table name:
[dbo].[Occupation]
Store new index
New index name:
Maintain stored index
Use existing index
Name of an existing index:
OK
Cancel
Help

Fig. 5

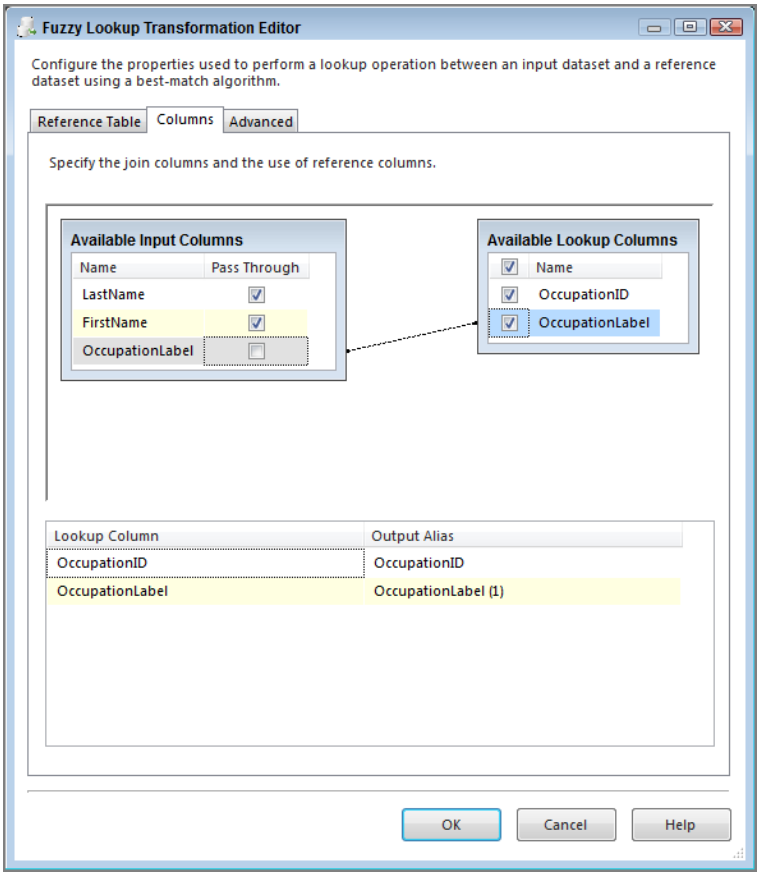
Fuzzy Lookup Transformation Editor
Configure the properties used to perform a lookup operation between an input dataset and a reference dataset using a best-match algorithm.
Reference Table
Columns
Advanced
Specify the join columns and the use of reference columns.
Available Input Columns
Name
Pass Through
LastName
FirstName
OccupationLabel
Available Lookup Columns
Name
OccupationID
OccupationLabel
Lookup Column
Output Alias
OccupationID
OccupationID
OccupationLabel
OccupationLabel (1)
OK
Cancel
Help

Fig. 6

12) Next, in the **Advanced** tab, leave the **Similarity threshold** at the default setting and change the token delimiters to use only a period in the **Additional delimiters box**, as reflected in Fig. 7. Also, modify the **Similarity threshold** to **0.50** and then click **OK**.

13) To bring together the data from both lookup transforms, drag a **Union All** over and connect the two lookups to it. First, connect the blue arrow from the **Fuzzy Lookup Transform** and then connect the blue arrow from the regular **Lookup Transform**. Then open the **Union All Transformation Editor** and delete the unneeded columns by right-clicking and selecting **Delete** on the columns that are not pictured in Fig. 8. You may also need to rename the output of **OccupationLabel** to not include (1) in the name.

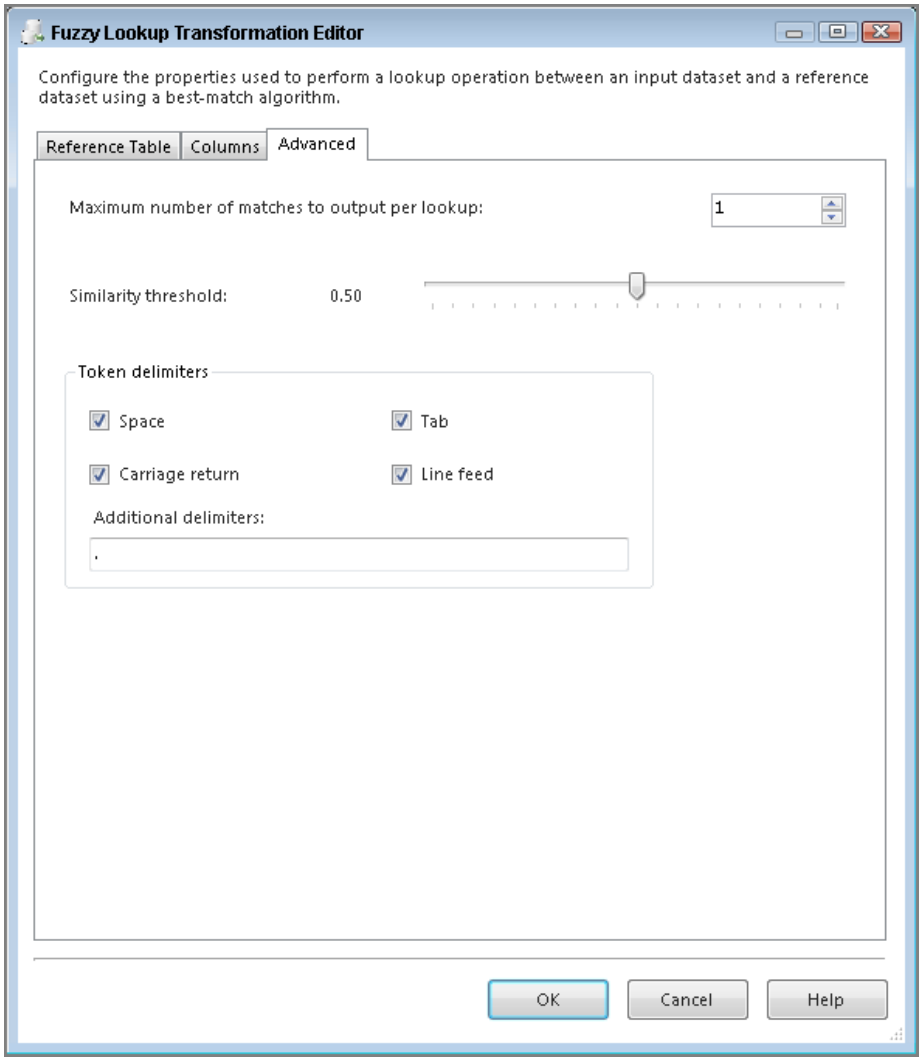


Fig. 7

14) To finish off this package you need to load the results into a new table. Bring an **OLE DB Destination** onto the design surface, and from within the editor, select **New** to create a new table. Use the following code to create the **EmployeeRoster** table:

```
CREATE TABLE [EmployeeRoster] (
      [EmployeeID] [smallint] IDENTITY(1,1) NOT NULL,
      [LastName] varchar(50),
      [FirstName] varchar(50),
      [OccupationID] smallint,
      [OccupationLabel] varchar(50)
)
```

15) Once the mapping has been set in the destination, click **OK** and your package is complete. A successful run of this package should look like Fig. 9. Compare the **EmployeeRoster** table to the original flat file you started with, and you will see the **Fuzzy Lookup** using the reference table corrected 10 rows of dirty data.

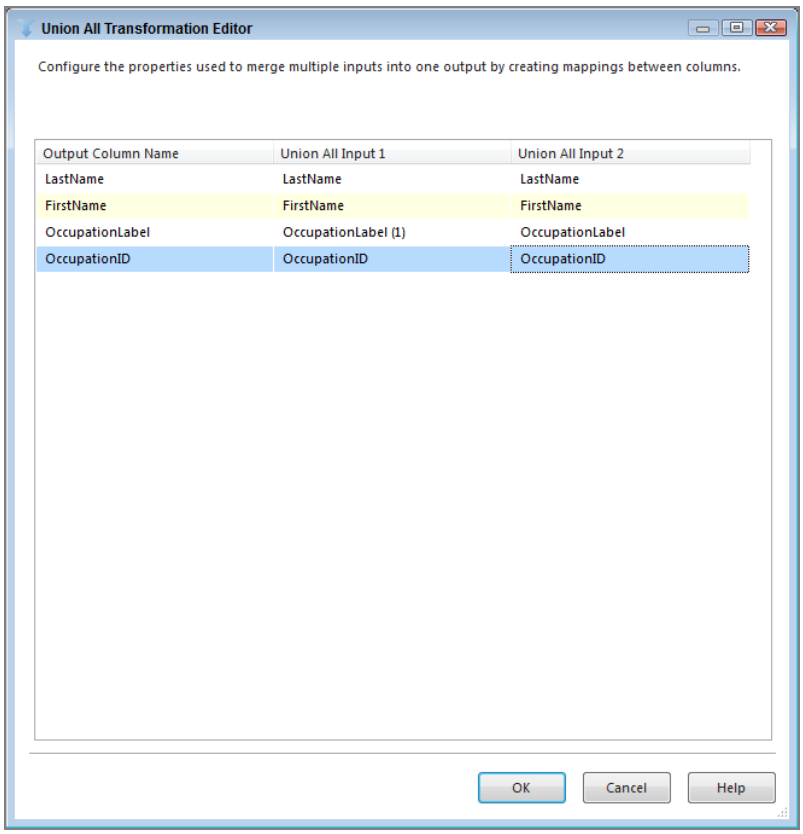


Fig. 8

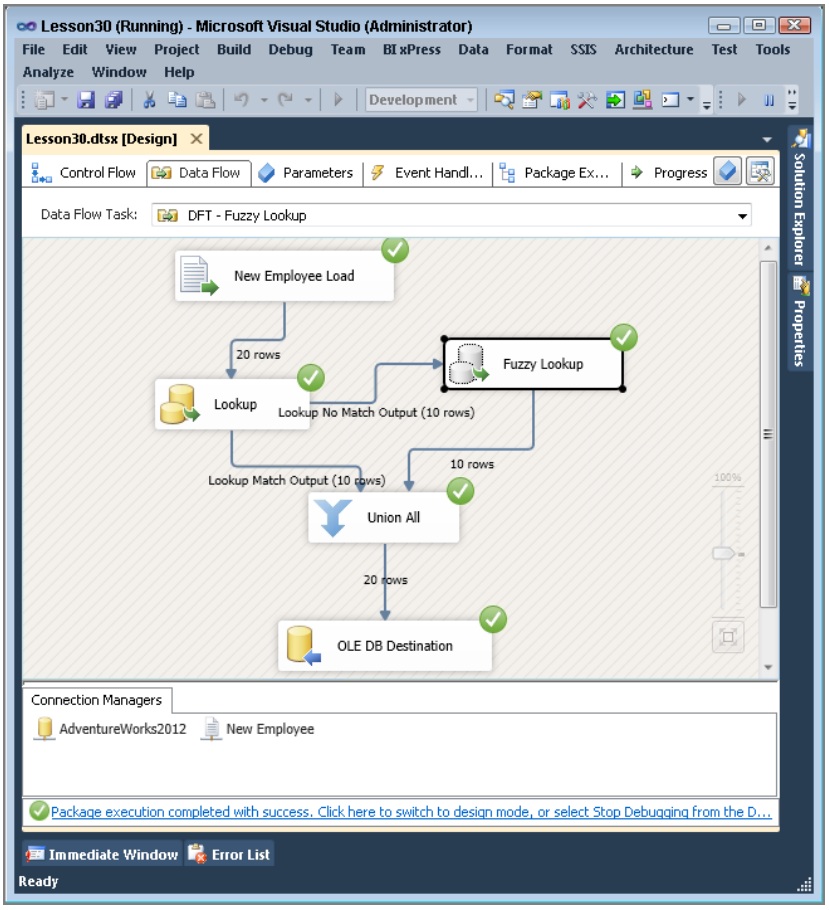
Lesson30 (Running) - Microsoft Visual Studio (Administrator)
File Edit View Project Build Debug Team BIxPress Data Format SSIS Architecture Test Tools
Analyze Window Help
Development
Lesson30.dtsx [Design]
Control Flow
Data Flow
Parameters
Event Handl...
Package Ex...
Progress
Data Flow Task:
DFT - Fuzzy Lookup
New Employee Load
20 rows
Lookup
Lookup No Match Output (10 rows)
Fuzzy Lookup
Lookup Match Output (10 rows)
10 rows
Union All
20 rows
OLE DB Destination
Connection Managers
AdventureWorks2012
New Employee
Package execution completed with success. Click here to switch to design mode, or select Stop Debugging from the D...
Immediate Window
Error List
Ready
Solution Explorer
Properties

Fig. 9

**Removing Duplicates with the Fuzzy Grouping Transform**

In the previous exercise, you saw how to use the **Fuzzy Lookup Transform** to prevent bad data from being loaded in your dimension tables, but what if the bad data is already in your table or if you are just beginning to build your data warehouse?.

In these circumstances, you can use the **Fuzzy Grouping Transform** to examine the contents of suspect fields and provide groupings of similar words. You can use the matching information provided by this transform to clean up the table and eliminate redundancy.

The **Fuzzy Grouping Transform** uses the same logic as the **Fuzzy Lookup Transform**, and therefore requires many of the same things. It must have a connection to an **OLE DB Connection Manager** to generate temporary tables that the transform uses in its algorithm. At development time, the **Connection Manager** tab is where you make this setting.

Also, just as was the case with the **Fuzzy Lookup Transform**, this transform expects an input stream with a string, either **DT_WSTR** or **DT_STR** data type. The **Columns** tab of the **Fuzzy Grouping Transformation Editor** (which you open by double-clicking the transform), shown in Fig, 10, is where you select the string field that you want to be analyzed and grouped into logical matches. Notice in the top part of the **Columns** tab that you can also check **Pass Through** on each column, which means the data is not analyzed, but is accessible in the output stream. If you move down to the bottom part of the **Columns** tab, you see a table of options for each input column. You can choose the names of any of the output columns: **Group Output Alias**, **Output Alias**, and **Similarity Output Alias**. Often the only column you want from this group is the **Group Output Alias**. If you select more than one column to be analyzed, the minimum similarity evaluation is configurable at the column level.

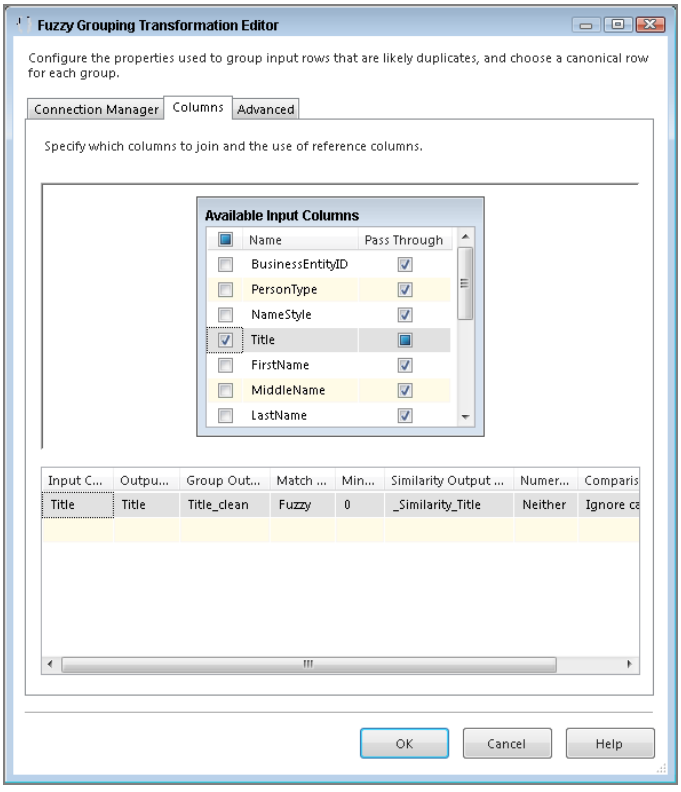


Fig. 10

The **Numerals** option enables you to configure the numbers in the input stream when grouping text logically. This may be necessary when comparing an address field, because, more than likely, it will have leading numerals; for example, the address **"834 West Elm Street"** has the leading numerals **"834."**
Comparison flags provide the options to ignore or pay attention to case, kana type (Japanese characters), nonspacing characters, character width, symbols, and punctuation.

The **Advanced** tab is where you see some of the familiar configurations you saw in the **Fuzzy Lookup Transform** that control the logic algorithm used for finding matches. A slider controls the similarity threshold. It is recommended you start this at **0.5** to test and move the slider up or down until you get the results you are looking for. This setting is normally decided based on a businessperson's review of the data, not the developer's review. The token delimiters can also be set if, for example, you don't want the comparison process to break up incoming strings with a period (.) or spaces. Fig. 11 shows the default settings for the **Advanced** tab.

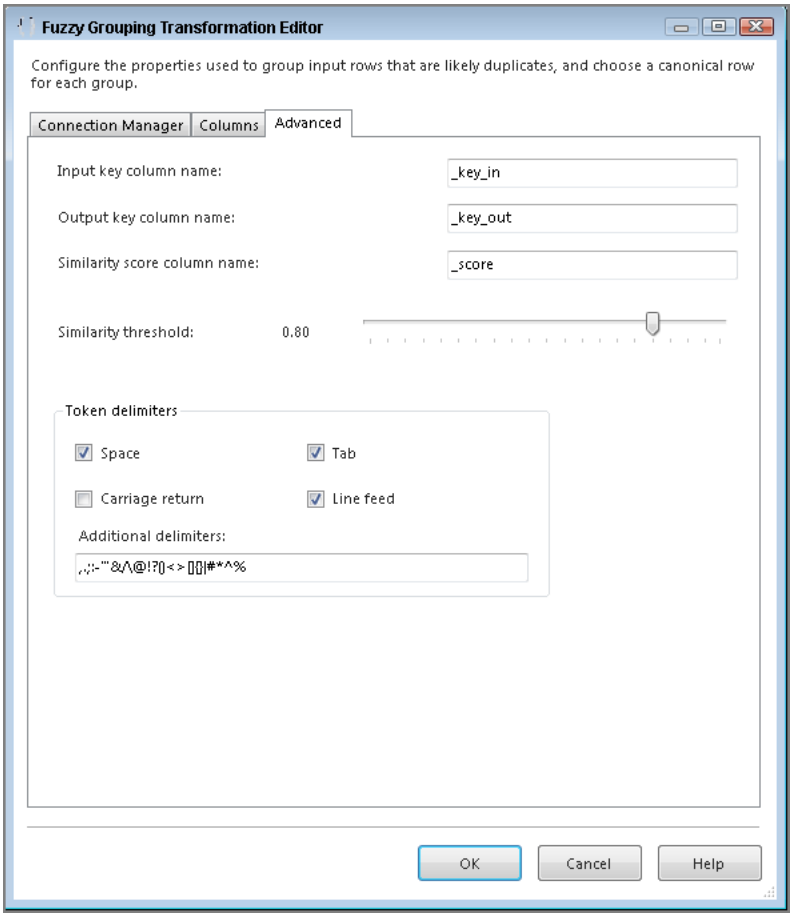

Fig. 11

One feature that was not in the **Fuzzy Lookup Transform** is the ability to set the names of the three additional fields that are added automatically to the output of this transform. By default, these fields are named **_key_in**, **_key_out**, and **_score**. These new outputs that will be added to the data stream are important to understand:

- **_key_in**—This column uniquely identifies each row in the stream.

- **_key_out**—This column identifies a group of duplicate rows. Any rows that have the same _key_out value are rows that are in the same group.
- **_score**—This column indicates the similarity of the row with a value between 0 and 1. A similarity of 1 would be an exact match.

## 1. Cel ćwiczenia

In this lab, you create a new dimension table and populate it with occupations for your company. The import file contains several different versions of the same occupation, and you need to determine which will be the best fit. After this lesson, you will have an understanding of how to use the **Fuzzy Grouping Transform** to remove duplicates.

## 2. Przebieg ćwiczenia

1) Create a new package.

2) Drag a **Data Flow Task** onto your designer and name it **DFT - Fuzzy Grouping**.

3) Create a new **Flat File Connection Manager**, name it **Occupations**, and point it to **FuzzyExample.txt**. Also, check the Column names in the first data row option. The editor should look like Fig. 12.

4) In the Data Flow, bring a new **Flat File Source** over and name it **Occupation Load**. Open the editor and make the connection manager the newly created Occupations.

5) On the **Columns** tab, select only the **TITLE** column to return, change the name of the output column to **OccupationLabel,** then click **OK**.

6) Next, create another connection manager, this time an **OLE DB Connection Manager**, using the **AdventureWorks2014** database.

7) Bring a **Fuzzy Grouping Transform** in the **Data Flow**, connect it to your **Flat File Source**, and open the editor. Set the OLE DB Connection Manager to **AdventureWorks2014**.

8) On the **Columns** tab, there is only one column to bring back, so check the **OccupationLabel** Fig. 13 shows what the **Columns** tab should look like now.

9) Next, in the **Advanced** tab, change the **Similarity threshold** to **0.50** and change the Token delimiters to reflect Fig. 14. Then click **OK**.

10) If you ran this now and loaded a table, you would have 20 rows of the clean data, but you would also have several duplicate records. Remember, you are trying to create a

dimension table, so to prevent duplicates in this package add a **Conditional Split Transform** with an **Output Name** of **Best Match** and a **Condition** of **_key_in == _key_out**. If these two values match, the grouped value is the best representative candidate for the natural key in a dimension table. All other rows are not needed, so you can name the **Default Output Name Delete**. Fig. 15 shows how your **Conditional Split Transform** should be configured.

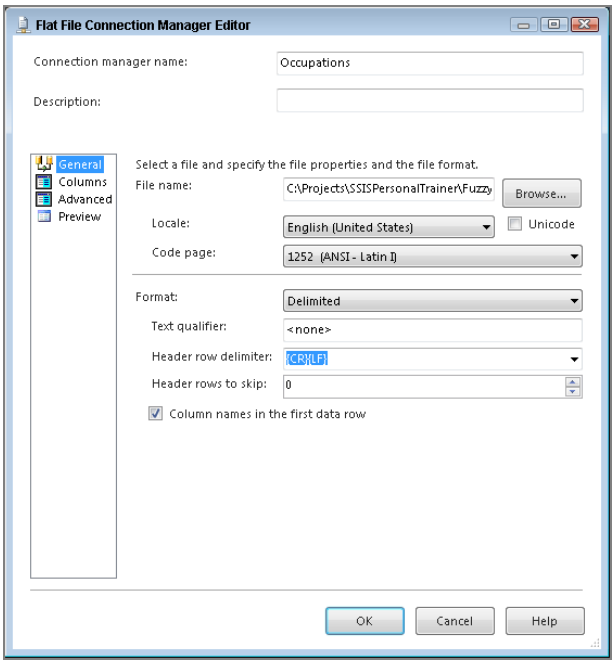


Fig. 12

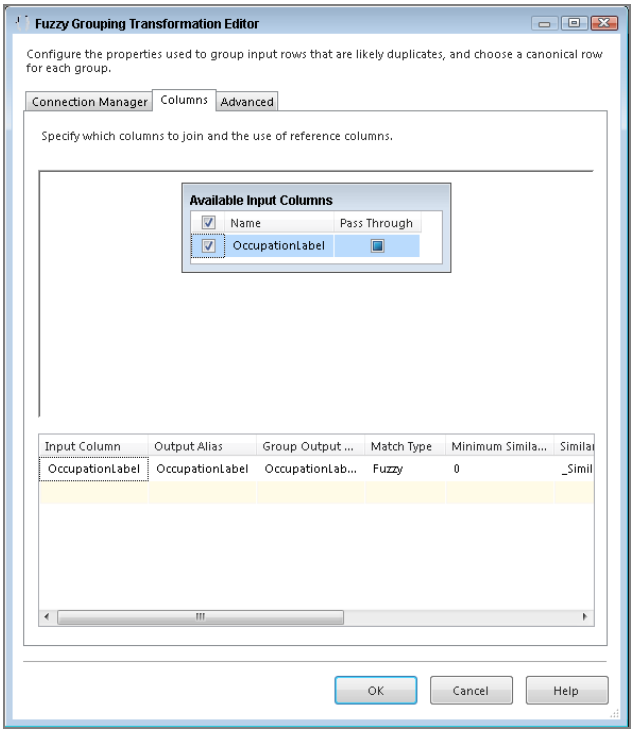


Fig. 13

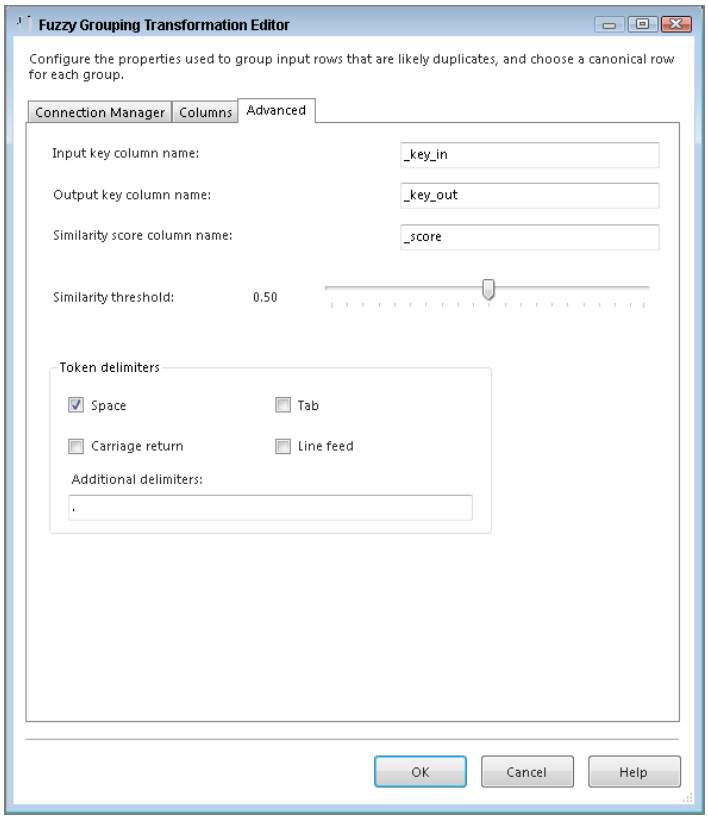
Fuzzy Grouping Transformation Editor
Configure the properties used to group input rows that are likely duplicates, and choose a canonical row for each group.
Connection Manager
Columns
Advanced
Input key column name:
_key_in
Output key column name:
_key_out
Similarity score column name:
_score
Similarity threshold:
0.50
Token delimiters
Space
Tab
Carriage return
Line feed
Additional delimiters:
,
OK
Cancel
Help

Fig. 14

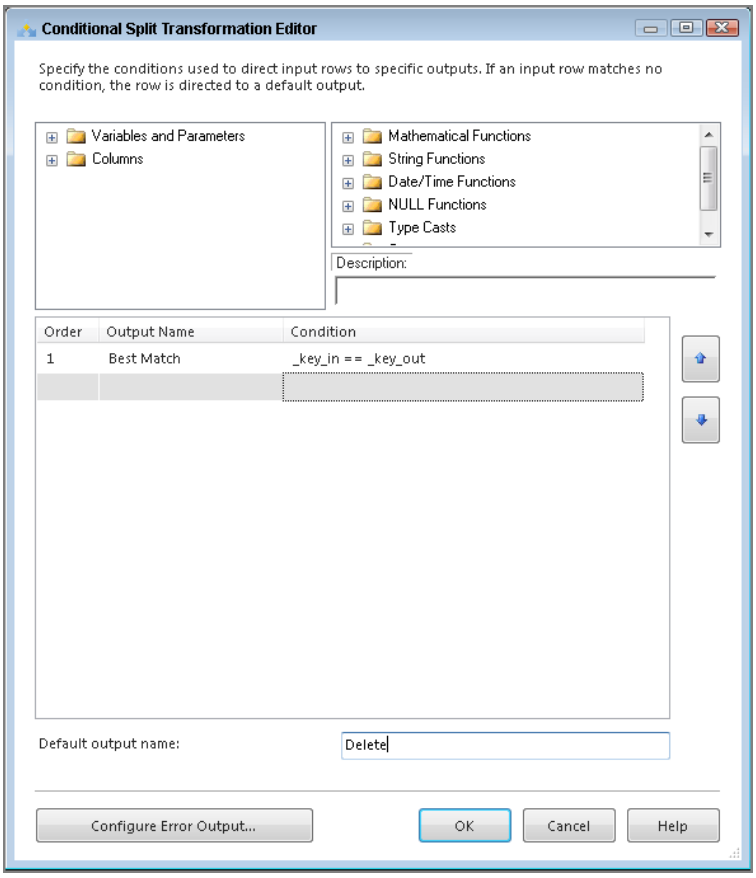
Conditional Split Transformation Editor
Specify the conditions used to direct input rows to specific outputs. If an input row matches no condition, the row is directed to a default output.
Variables and Parameters
Columns
Mathematical Functions
String Functions
Date/Time Functions
NULL Functions
Type Casts
Description:
Order
Output Name
Condition
1
Best Match
_key_in == _key_out
Default output name:
Delete
Configure Error Output...
OK
Cancel
Help

Fig. 15

11) To finish off this package, you need to load the results into a new table. Bring **an OLE DB Destination** onto the design surface and from within the editor select New next to Name of the table or name of the view to create a new table. Use the following code to create the **Occupation_FuzzyGrouping** table:

```
CREATE TABLE [dbo].[Occupation_FuzzyGrouping](
[OccupationID] [smallint] IDENTITY(1,1) NOT NULL,
[OccupationLabel] [varchar](50) NOT NULL
)
```

12) Remember from the beginning of this lesson that the **Fuzzy Grouping Transform** provides several output columns. These columns include a **Group Output Alias** column that you now use in the **Mappings** tab. Set **OccupationLabel_clean** to map to the **OccupationLabel** column in the destination. Once your **Mappings** tab looks like Fig. 16, click **OK.**

13) A successful run of this package should look like Fig. 17.

14) Fig. 18 shows the results in the **Occupation_FuzzyGrouping** table you just populated. If you completed previous lab, you might notice that you just created essentially the same table (aside from the order) that was used as a reference table in the previous lab.

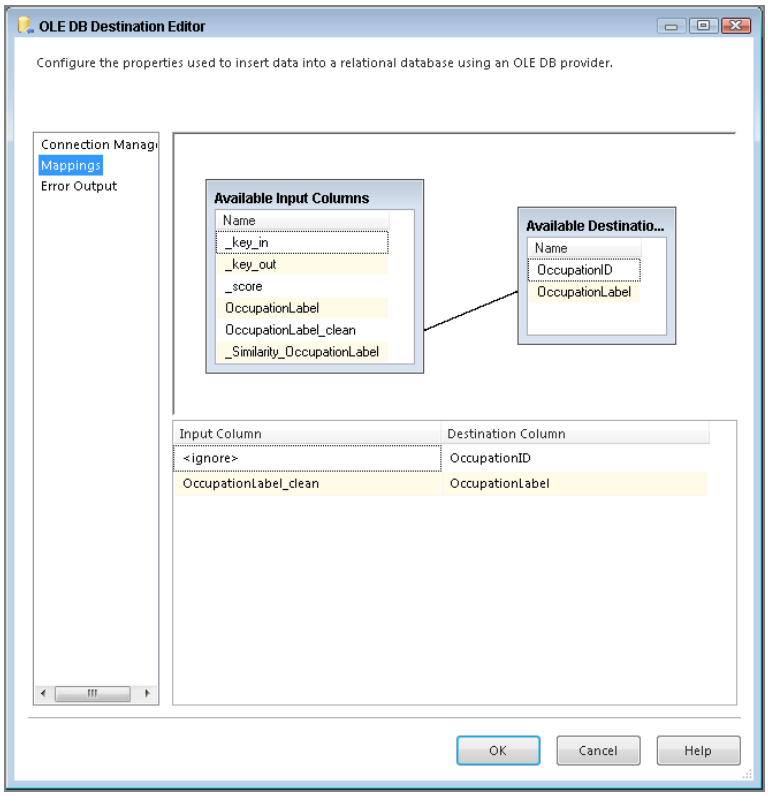


Fig. 16

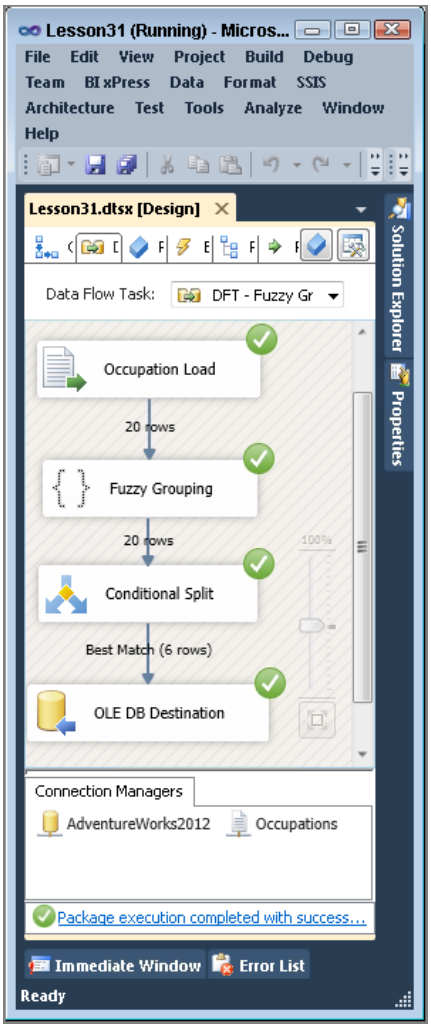


Fig. 17

| | OccupationID | OccupationLabel |
|---|---|---|
| 1 | 1 | CUSTOMER SERVICE REPRESENTATIVE |
| 2 | 2 | ASSISTANT MANAGER |
| 3 | 3 | REGIONAL MANAGER |
| 4 | 4 | STORE MANAGER |
| 5 | 5 | SHIFT LEADER |
| 6 | 6 | DISTRICT MANAGER |

Fig. 18

## Bibliografia


1) Knight B., Knight D., Davis M, Snyder W. (2013): Knight's Microsoft® SQL Server® 2012 Integration Services 24-Hour Trainer, John Wiley & Sons.
2) Knight B., Veerman E., Moss J.M., Davis M., Rock C. (2012): PROFESSIONAL Microsoft® SQL Server® 2012 Integration Services, John Wiley & Sons.
3) http://www.wrox.com/WileyCDA/Section/id-814197.html
4) https://msdn.microsoft.com/library/ms169917(SQL.120).aspx
5) Tok W-H., Parida R. Masson M. Ding X. Sivashanmugam (2012): Microsoft SQL Server 2012 Integration Services, Promise (tłumaczenie j. polski).
6) Kimball R. (2004): The Data Warehouse ETL Toolkit. John Wiley & Sons