APPLICATION NOTE

melissa

Cleaning, Validating and Enhancing the SQL Server Data Warehouse Contact Dimension

Problem:

Contact data is acquired from a variety of sources. You may collect contacts at trade shows or from your web site. You may purchase lists of contacts. In many cases the contact data may be incomplete and not in a standardized format. Cleaning, validating and standardizing the contact data is definitely a challenge.

In a SQL Server Data Warehouse, the Contact dimension may include existing customers as well as prospects. The goal of the Contact dimension is to have clean, valid and up-to-date data which can be used to communicate with contacts via email, mail and phone, as well as perform analysis on the contacts based on demographics. Unfortunately, the built-in SSIS components do not provide the kind of data cleansing, validating and enhancing of demographic data that you need. SSIS does provide the ability for you to create script components using .NET code to do these kinds of tasks.

How can we clean, validate and enhance our Contact dimension without writing code?

Solution:

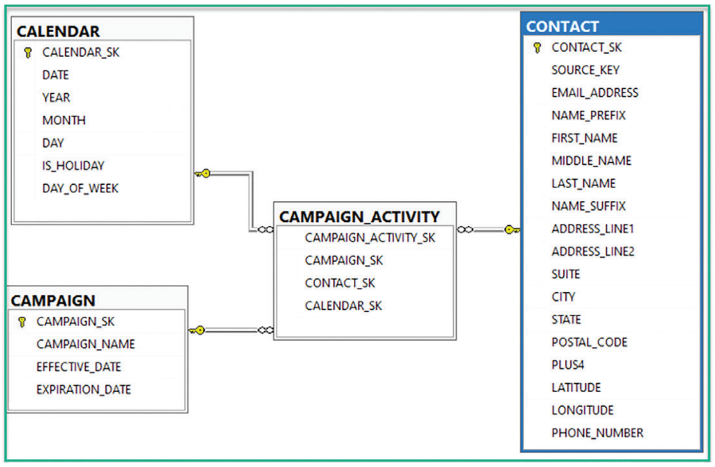
Melissa has a variety of tools available to clean, validate and enhance the Contact dimension in your SQL Server data warehouse. Specifically, Melissa's suite of SSIS Data Quality Components can be leveraged for this task. The Melissa SSIS components are plug and play; you simply drag and drop the components onto the Data Flow, configure the component properties, and you are ready to go. There is no coding required.

The Melissa SSIS components provide a wealth of capabilities including:

- SQL Server Data Import Options
- Poor Data Quality in Business is Expensive
- Correcting SQL Server Duplicates
- Resolving Out-of-Date SQL Server Data
- SQL Server Data Verification

Demo Solution

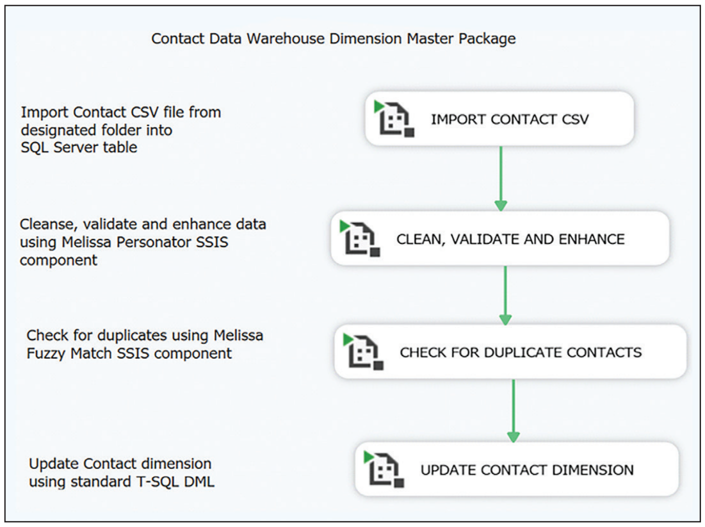
To set the stage, I will introduce the demo solution that I will use to show the Melissa SSIS components in action. The following is a simplified data warehouse; I will only focus on inserting and updating the Contact dimension.



The following are the main points for this hypothetical data warehouse:

- This is a simple star schema
- The CALENDAR table is a basic time dimension
- The CAMPAIGN table is a dimension that defines specific marketing campaigns
- The CAMPAIGN_ACTIVITY table is the fact table that tracks emails and letters sent to contacts for specific marketing campaigns
- Ultimately, I want to measure the effectiveness of the communications for each marketing campaign

I will use the following SSIS package to insert and update Contacts:


Contact Data Warehouse Dimension Master Package
Import Contact CSV file from designated folder into SQL Server table
IMPORT CONTACT CSV
Cleanse, validate and enhance data using Melissa Personator SSIS component
CLEAN, VALIDATE AND ENHANCE
Check for duplicates using Melissa Fuzzy Match SSIS component
CHECK FOR DUPLICATE CONTACTS
Update Contact dimension using standard T-SQL DML
UPDATE CONTACT DIMENSION

The following are the main points about the SSIS package:

- This is an example of a "master" package where each task is an Execute Package Task
- Contacts are received in a comma-separated values (CSV) file
- The IMPORT CONTACT CSV task inserts the records from the CSV file into a SQL Server table
- The CLEAN, VALIDATE AND ENHANCE task uses the Melissa Personator SSIS component to prepare the Contacts for loading into the Contact dimension
- The CHECK FOR DUPLICATE CONTACTS task uses the Melissa Fuzzy Match SSIS component to determine which Contacts already exist in the Contact dimension and which Contacts are new
- The UPDATE CONTACT DIMENSION task performs the inserts and updates for the Contact dimension

I will focus on the CLEAN, VALIDATE AND ENHANCE and CHECK FOR DUPLICATE CONTACTS tasks as these are the ones that use the Melissa SSIS components. I will touch on the UPDATE CONTACT DIMENSION task; this is just an Execute SQL task that inserts or updates rows in the Contact dimension.

Before drilling into the details of these tasks, I want to briefly discuss the Melissa SSIS components.

Melissa SSIS Components

The Melissa SSIS components are placed in the SSIS Data Flow. They accept input from any SSIS data source and they output results to any SSIS data destination. The components have very robust configuration capabilities. You can select the capabilities that you need and fine tune the many options available, as you will see in the later sections. You can save your configuration for use in other SSIS packages.

Personator

The Personator SSIS component provides cleaning, validating and enhancing of Contacts. It works on names, addresses, emails and phone numbers. Cleaning involves processes like standardizing upper/lower case, parsing and formatting. Validating involves checking; e.g. is the address deliverable? Enhancing provides all sorts of demographic data based on the address; e.g. latitude and longitude. Personator can parse full names and full addresses into their individual fields. Where address information is missing, Personator can provide it. For instance, if you have an address line and a ZIP Code, Personator can provide the city and state.

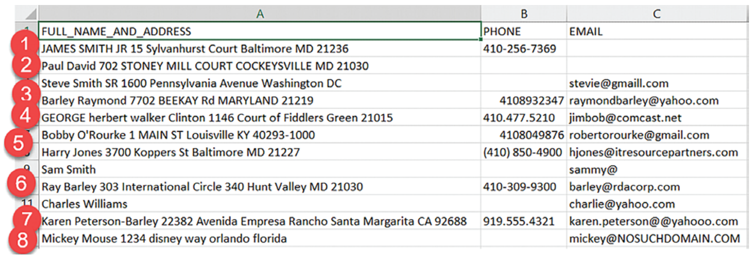
The main point about Personator is that it has domain-specific knowledge of Contact data. It leverages reference data to properly parse names and addresses.

Fuzzy Match

The Fuzzy Match SSIS component can determine whether Contacts are in fact new or already exist in the Contact dimension. It provides a large toolbox of state-of-the-art fuzzy matching algorithms that can be configured with each field comparison. Based on a calculated percentage match, the component can direct Contact inputs to Match, Possible Match and Non-Match outputs. This makes inserting or updating the Contact dimension very straight-forward.

Sample Input File

The following is the sample input file that I will use for the demo:



		A	B	C
		FULL_NAME_AND_ADDRESS	PHONE	EMAIL
1		JAMES SMITH JR 15 Sylvanhurst Court Baltimore MD 21236	410-256-7369	
2		Paul David 702 STONEY MILL COURT COCKEYSVILLE MD 21030		
		Steve Smith SR 1600 Pennsylvania Avenue Washington DC		stevie@gmaill.com
3		Barley Raymond 7702 BEEKAY Rd MARYLAND 21219	4108932347	raymondbarley@yahoo.com
4		GEORGE herbert walker Clinton 1146 Court of Fiddlers Green 21015	410.477.5210	jimbob@comcast.net
5		Bobby O'Rourke 1 MAIN ST Louisville KY 40293-1000	4108049876	robertorourke@gmail.com
		Harry Jones 3700 Koppers St Baltimore MD 21227	(410) 850-4900	hjones@itresourcepartners.com
	9	Sam Smith		sammy@
6		Ray Barley 303 International Circle 340 Hunt Valley MD 21030	410-309-9300	barley@rdacorp.com
	11	Charles Williams		charlie@yahoo.com
7		Karen Peterson-Barley 22382 Avenida Empresa Rancho Santa Margarita CA 92688	919.555.4321	karen.peterson@@yahooo.com
8		Mickey Mouse 1234 disney way orlando florida		mickey@NOSUCHDOMAIN.COM

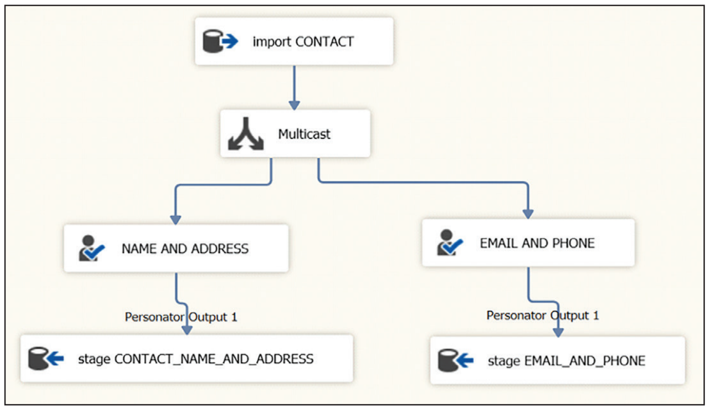
The sample file has some typical anomalies that we find in Contact data:

- The name should be in standard format with upper and lower case
- The address is an apartment building; the apartment number is missing
- The first name and last name are flipped; the city is missing
- The city and state are missing; this individual has two middle names
- The address is an office building; a suite number is required to deliver to the address
- The address is an office building; the suite number is there but the Ste abbreviation is missing
- How would you parse the address and city? The Personator SSIS component has the knowledge to do it
- Some people intentionally provide incorrect information

In the sections that follow, I will walk through the details of how the Personator SSIS component properly parses and corrects names and addresses. I will also walk through how the Fuzzy Match SSIS component determines duplicate Contacts.

CLEAN, VALIDATE AND ENHANCE Task

The following is the Data Flow for the CLEAN, VALIDATE AND ENHANCE task:



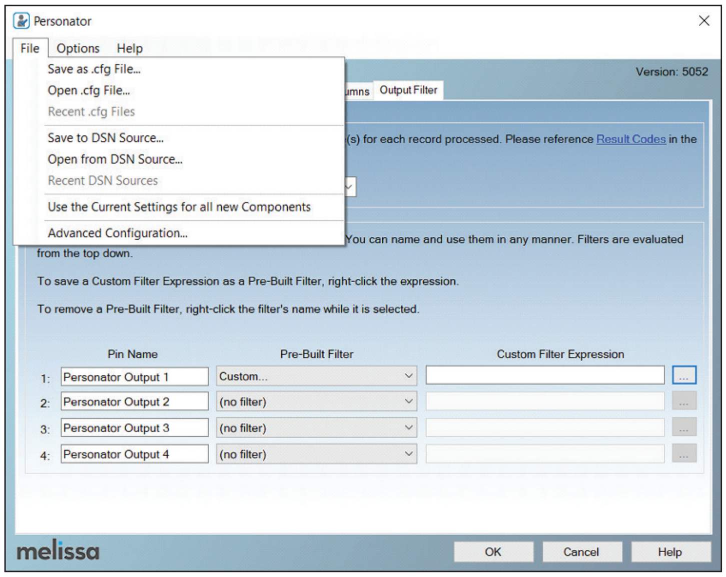
The following are the main points about the above Data Flow:

- Import CONTACT retrieves the Contact data to be processed from the SQL Server table import. CONTACT
- Multicast directs the Contact rows to the Melissa Personator SSIS components (NAME AND ADDRESS and EMAIL AND PHONE)
- NAME AND ADDRESS and EMAIL AND PHONE are instances of the Personator component that clean, validate and enhance the Contact data
- Stage CONTACT_NAME_AND_ADDRESS and stage EMAIL_AND_PHONE are SQL Server tables that store the output from the Melissa Personator components

In the next section, I will walk through the details to configure the Personator SSIS component. After that, I will review the results from executing the CLEAN, VALIDATE AND ENHANCE task.

Personator Configuration

The Personator SSIS component has very robust configuration capabilities. There are many options available. To start off, the following options are available from the File menu for saving and reusing your configuration:



(s) for each record processed. Please reference Result Codes in the

You can name and use them in any manner. Filters are evaluated from the top down.

To save a Custom Filter Expression as a Pre-Built Filter, right-click the expression.

To remove a Pre-Built Filter, right-click the filter's name while it is selected.

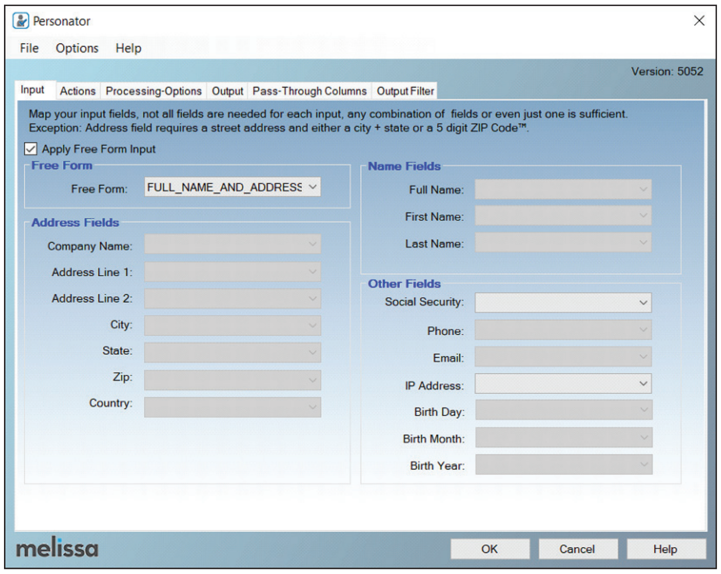
	Pin Name	Pre-Built Filter	Custom Filter Expression
1:	Personator Output 1	Custom...	
2:	Personator Output 2	(no filter)	
3:	Personator Output 3	(no filter)	
4:	Personator Output 4	(no filter)	

I will walk through the following tabs and discuss how it works:

- Input
- Output
- Pass-Through Columns
- Output Filter

Input

The Input tab is used to map the fields in your Contact input to the pre-defined fields in the Personator component. It is shown below:


Personator
File Options Help
Input Actions Processing-Options Output Pass-Through Columns Output Filter
OK Cancel Help
melissa

Version: 5052

Map your input fields, not all fields are needed for each input, any combination of fields or even just one is sufficient.
Exception: Address field requires a street address and either a city + state or a 5 digit ZIP Code™.

☑ Apply Free Form Input

Free Form

Free Form: FULL_NAME_AND_ADDRESS

Address Fields

Company Name:
Address Line 1:
Address Line 2:
City:
State:
Zip:
Country:

Name Fields

Full Name:
First Name:
Last Name:

Other Fields

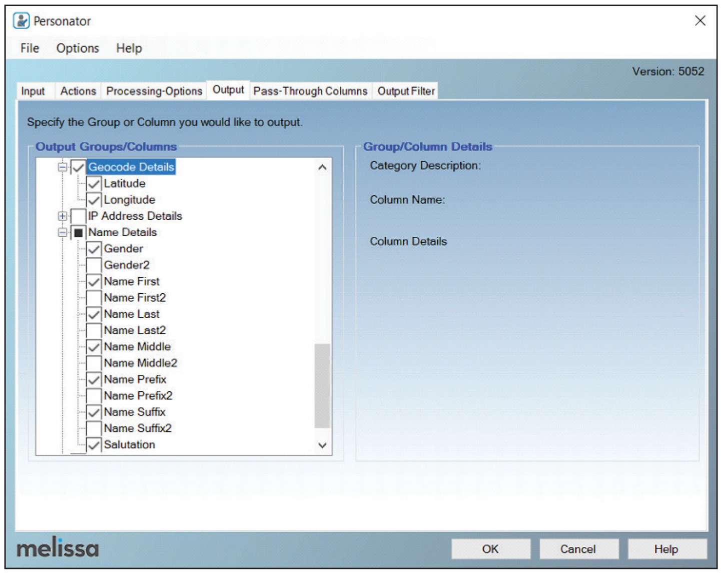
Social Security:
Phone:
Email:
IP Address:
Birth Day:
Birth Month:
Birth Year:

The following are the main points for the Input tab:

- Map whatever fields you have in your input to the pre-defined ones
- In my case, I want to parse the name and address from a single field so I check Apply Free Form Input and map the FULL_NAME_AND_ADDRESS field to the Free Form field
- When you choose Apply Free Form Input, the rest of the fields are disabled

Output

The Output tab allows you to specify which of the available fields from the Personator component that you want to retrieve. It is shown below:


Personator
File Options Help
Input Actions Processing-Options Output Pass-Through Columns Output Filter
OK Cancel Help
melissa

Version: 5052

Specify the Group or Column you would like to output.

Output Groups/Columns

- ☑ Geocode Details
  - ☑ Latitude
  - ☑ Longitude
- ☐ IP Address Details
- ■ Name Details
  - ☑ Gender
  - ☐ Gender2
  - ☑ Name First
  - ☐ Name First2
  - ☑ Name Last
  - ☐ Name Last2
  - ☑ Name Middle
  - ☐ Name Middle2
  - ☑ Name Prefix
  - ☐ Name Prefix2
  - ☑ Name Suffix
  - ☐ Name Suffix2
  - ☑ Salutation

Group/Column Details

Category Description:

Column Name:

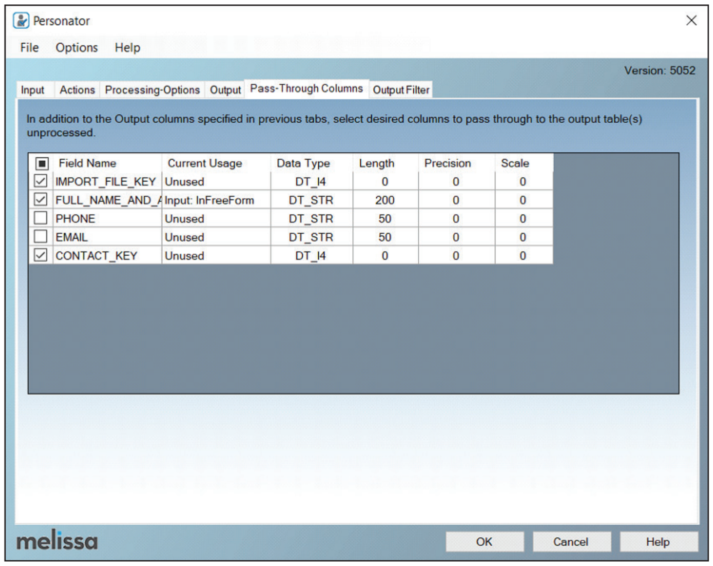
Column Details

The following are the main points for the Output tab:

- There are many fields available in the Output Groups/Columns list box; you simply check the ones you want
- I am selecting from the Name Details fields which are parsed from the FULL_NAME_AND_ADDRESS input field
- Gender and Salutation are examples of the additional demographic data that is available
- Personator is able to parse two names; for my demo, I am only specifying one name
- The Geocode Details are another example of additional data available; based on the address, you can retrieve the Latitude and Longitude

Pass-Through Columns

The Pass-Through Columns allow you to specify which of the fields in your input you want to include in the output from Personator. It is shown below:


Personator

File Options Help

Version: 5052

Input | Actions | Processing-Options | Output | Pass-Through Columns | Output Filter

In addition to the Output columns specified in previous tabs, select desired columns to pass through to the output table(s) unprocessed.

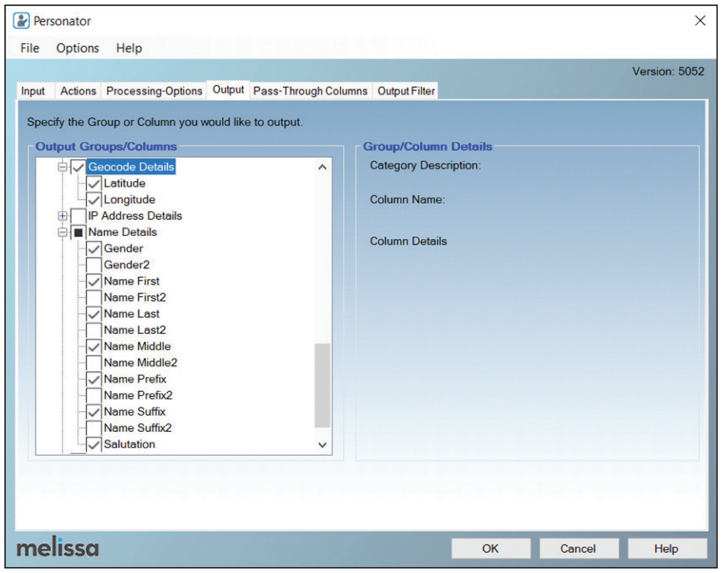
■	Field Name	Current Usage	Data Type	Length	Precision	Scale
✓	IMPORT_FILE_KEY	Unused	DT_I4	0	0	0
✓	FULL_NAME_AND_A	Input: InFreeForm	DT_STR	200	0	0
☐	PHONE	Unused	DT_STR	50	0	0
☐	EMAIL	Unused	DT_STR	50	0	0
✓	CONTACT_KEY	Unused	DT_I4	0	0	0

The following are the main points for the Output tab:

- The CONTACT_KEY is the primary key for the import.CONTACT table; when inserting or updating the Contact dimension, I want to include the CONTACT_KEY in order to identify the source of the Contact data
- I want to include the FULL_NAME_AND_ADDRESS in the output so I can compare it to the output from the Personator component
- The IMPORT_FILE_KEY is the primary key for a table that has a single row for each CSV file that I import

Output Filter

The Output Filter tab is the final tab in the Personator component configuration and has the options for the output you want from the component. It is shown below:


Personator

File Options Help

Version: 5052

Input | Actions | Processing-Options | Output | Pass-Through Columns | Output Filter

Specify the Group or Column you would like to output.

Output Groups/Columns

- ✓ Geocode Details
  - ✓ Latitude
  - ✓ Longitude
- ☐ IP Address Details
- ■ Name Details
  - ✓ Gender
  - ☐ Gender2
  - ✓ Name First
  - ☐ Name First2
  - ✓ Name Last
  - ☐ Name Last2
  - ✓ Name Middle
  - ☐ Name Middle2
  - ✓ Name Prefix
  - ☐ Name Prefix2
  - ✓ Name Suffix
  - ☐ Name Suffix2
  - ✓ Salutation

Group/Column Details

Category Description:

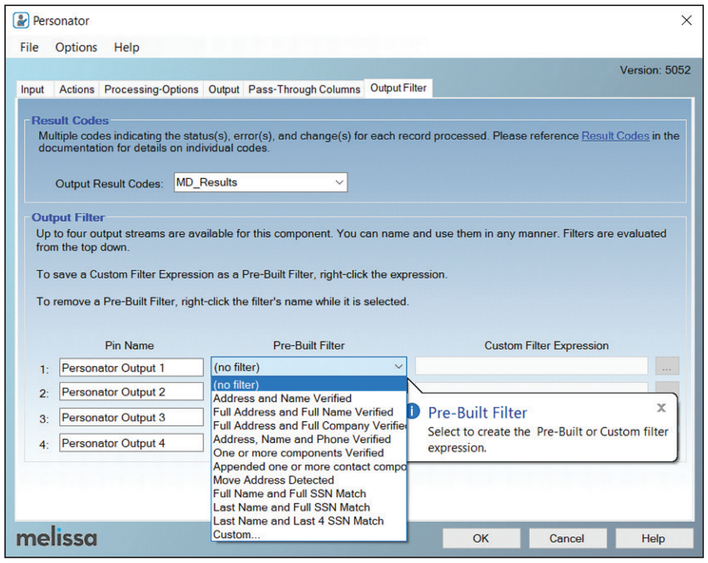
Column Name:

Column Details

The following are the main points for the Output Filter tab:

- The Personator component reports one or more "Result Codes" for each input record that it processes. The result codes provide the details on success or failure, and many other conditions. I will review the Result Codes in later sections
- Take a look at Result Codes for the details
- The Personator component has 4 outputs available to direct the Contact rows based on conditions; you can choose from the Pre-Built Filter or create your own Custom Filter Expression
- By default, all Contact rows will be directed to Output 1 unless you specify a filter
- In my demo, I went with the default

The Pre-Built Filter options are shown below:



Personator

File Options Help

Version: 5052

Input | Actions | Processing-Options | Output | Pass-Through Columns | Output Filter

Result Codes

Multiple codes indicating the status(s), error(s), and change(s) for each record processed. Please reference Result Codes in the documentation for details on individual codes.

Output Result Codes: MD_Results

Output Filter

Up to four output streams are available for this component. You can name and use them in any manner. Filters are evaluated from the top down.

To save a Custom Filter Expression as a Pre-Built Filter, right-click the expression.

To remove a Pre-Built Filter, right-click the filter's name while it is selected.

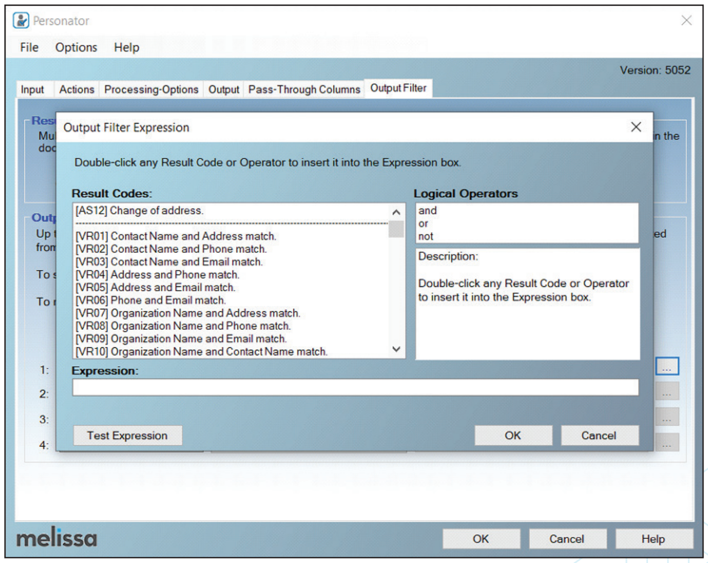
	Pin Name	Pre-Built Filter	Custom Filter Expression
1:	Personator Output 1	(no filter)	
2:	Personator Output 2		
3:	Personator Output 3		
4:	Personator Output 4		

(no filter)
Address and Name Verified
Full Address and Full Name Verified
Full Address and Full Company Verifie
Address, Name and Phone Verified
One or more components Verified
Appended one or more contact compo
Move Address Detected
Full Name and Full SSN Match
Last Name and Full SSN Match
Last Name and Last 4 SSN Match
Custom...

Pre-Built Filter

Select to create the Pre-Built or Custom filter expression.

When you select Custom… from the Pre-Built Filter, the Custom Filter Expression options are available and are shown below:



Personator

File Options Help

Version: 5052

Input | Actions | Processing-Options | Output | Pass-Through Columns | Output Filter

Output Filter Expression

Double-click any Result Code or Operator to insert it into the Expression box.

Result Codes:

[AS12] Change of address.
[VR01] Contact Name and Address match.
[VR02] Contact Name and Phone match.
[VR03] Contact Name and Email match.
[VR04] Address and Phone match.
[VR05] Address and Email match.
[VR06] Phone and Email match.
[VR07] Organization Name and Address match.
[VR08] Organization Name and Phone match.
[VR09] Organization Name and Email match.
[VR10] Organization Name and Contact Name match.

Logical Operators

and
or
not

Description:

Double-click any Result Code or Operator to insert it into the Expression box.

Expression:

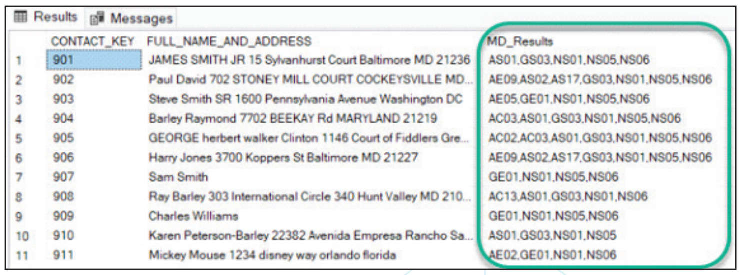
Next, I will review the results from executing the CLEAN, VALIDATE AND ENHANCE task.

CLEAN, VALIDATE AND ENHANCE Results

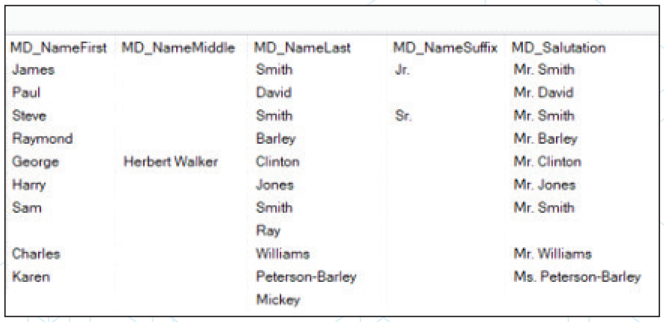
The CLEAN, VALIDATE AND ENHANCE task calls on the Personator component to clean, validate and enhance the Contact input. The Personator component outputs the results to the stage CONTACT_NAME_AND_ADDRESS and stage EMAIL_AND_PHONE SQL Server tables. The last step in the Data Flow is to combine the CONTACT_NAME_AND_ADDRESS and EMAIL_AND_PHONE tables into the stage FULL_CONTACT table.

I will review the following:

- Full Name results
- Full Address results
- Email and Phone results
- Combine tables



	CONTACT_KEY	FULL_NAME_AND_ADDRESS	MD_Results
1	901	JAMES SMITH JR 15 Sylvanhurst Court Baltimore MD 21236	AS01,GS03,NS01,NS05,NS06
2	902	Paul David 702 STONEY MILL COURT COCKEYSVILLE MD...	AE09,AS02,AS17,GS03,NS01,NS05,NS06
3	903	Steve Smith SR 1600 Pennsylvania Avenue Washington DC	AE05,GE01,NS01,NS05,NS06
4	904	Barley Raymond 7702 BEEKAY Rd MARYLAND 21219	AC03,AS01,GS03,NS01,NS05,NS06
5	905	GEORGE herbert walker Clinton 1146 Court of Fiddlers Gre...	AC02,AC03,AS01,GS03,NS01,NS05,NS06
6	906	Harry Jones 3700 Koppers St Baltimore MD 21227	AE09,AS02,AS17,GS03,NS01,NS05,NS06
7	907	Sam Smith	GE01,NS01,NS05,NS06
8	908	Ray Barley 303 International Circle 340 Hunt Valley MD 210...	AC13,AS01,GS03,NS01,NS06
9	909	Charles Williams	GE01,NS01,NS05,NS06
10	910	Karen Peterson-Barley 22382 Avenida Empresa Rancho Sa...	AS01,GS03,NS01,NS05
11	911	Mickey Mouse 1234 disney way orlando florida	AE02,GE01,NS01,NS06



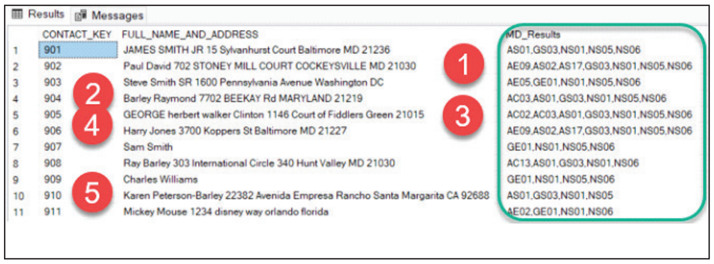
MD_NameFirst	MD_NameMiddle	MD_NameLast	MD_NameSuffix	MD_Salutation
James		Smith	Jr.	Mr. Smith
Paul		David		Mr. David
Steve		Smith	Sr.	Mr. Smith
Raymond		Barley		Mr. Barley
George	Herbert Walker	Clinton		Mr. Clinton
Harry		Jones		Mr. Jones
Sam		Smith		Mr. Smith
		Ray		
Charles		Williams		Mr. Williams
Karen		Peterson-Barley		Ms. Peterson-Barley
		Mickey		

The following are the main points:

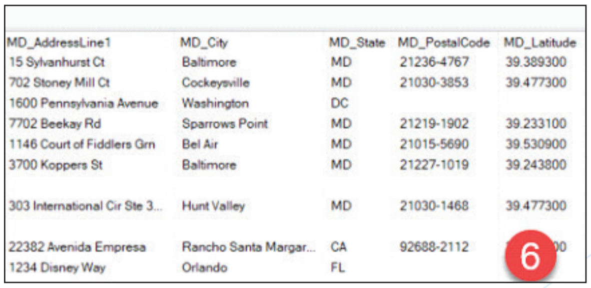
- The MD_Results column provides the details from the Personator component; it includes a comma-separated list of result codes for the name and address
- The full name gets parsed into the individual columns; salutation is an example of the additional data that is available
- NS01 tells us that name parsing was successful
- NS05 confirms that the FirstName was found in the census table of names and it is very likely to be a real first name
- NS06 confirms that the LastName was found in the census table of names and it is very likely to be a real last name

Full Address Results

The following shows the results for the full addresses:



	CONTACT_KEY	FULL_NAME_AND_ADDRESS	MD_Results
1	901	JAMES SMITH JR 15 Sylvanhurst Court Baltimore MD 21236	AS01,GS03,NS01,NS05,NS06
2	902	Paul David 702 STONEY MILL COURT COCKEYSVILLE MD 21030	AE09,AS02,AS17,GS03,NS01,NS05,NS06
3	903	Steve Smith SR 1600 Pennsylvania Avenue Washington DC	AE05,GE01,NS01,NS05,NS06
4	904	Barley Raymond 7702 BEEKAY Rd MARYLAND 21219	AC03,AS01,GS03,NS01,NS05,NS06
5	905	GEORGE herbert walker Clinton 1146 Court of Fiddlers Green 21015	AC02,AC03,AS01,GS03,NS01,NS05,NS06
6	906	Harry Jones 3700 Koppers St Baltimore MD 21227	AE09,AS02,AS17,GS03,NS01,NS05,NS06
7	907	Sam Smith	GE01,NS01,NS05,NS06
8	908	Ray Barley 303 International Circle 340 Hunt Valley MD 21030	AC13,AS01,GS03,NS01,NS06
9	909	Charles Williams	GE01,NS01,NS05,NS06
10	910	Karen Peterson-Barley 22382 Avenida Empresa Rancho Santa Margarita CA 92688	AS01,GS03,NS01,NS05
11	911	Mickey Mouse 1234 disney way orlando florida	AE02,GE01,NS01,NS06



MD_AddressLine1	MD_City	MD_State	MD_PostalCode	MD_Latitude
15 Sylvanhurst Ct	Baltimore	MD	21236-4767	39.389300
702 Stoney Mill Ct	Cockeysville	MD	21030-3853	39.477300
1600 Pennsylvania Avenue	Washington	DC		
7702 Beekay Rd	Sparrows Point	MD	21219-1902	39.233100
1146 Court of Fiddlers Grn	Bel Air	MD	21015-5690	39.530900
3700 Koppers St	Baltimore	MD	21227-1019	39.243800
303 International Cir Ste 3...	Hunt Valley	MD	21030-1468	39.477300
22382 Avenida Empresa	Rancho Santa Margar...	CA	92688-2112	[illegible]
1234 Disney Way	Orlando	FL		

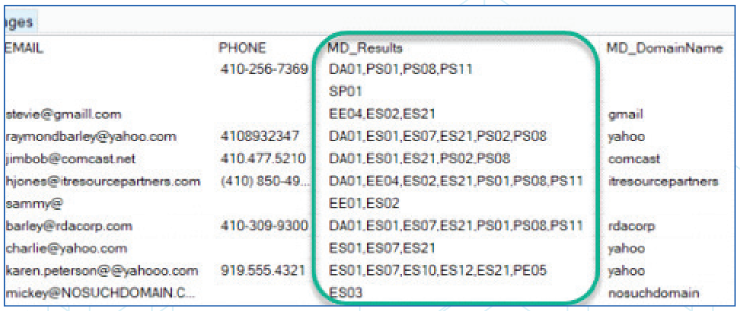
The following are the main points:

- The address is an apartment building; AE09 means the apartment number is missing
- The city is missing; AC03 means the city or municipality name was added or changed
- The city and state are missing; AC02 means the state or province was added or changed
- The address is an office building; AE09 means the suite is missing
- The address and city are properly parsed; Personator uses reference data to get the cities within the state and the streets within the cities
- Latitude is an example of additional data available

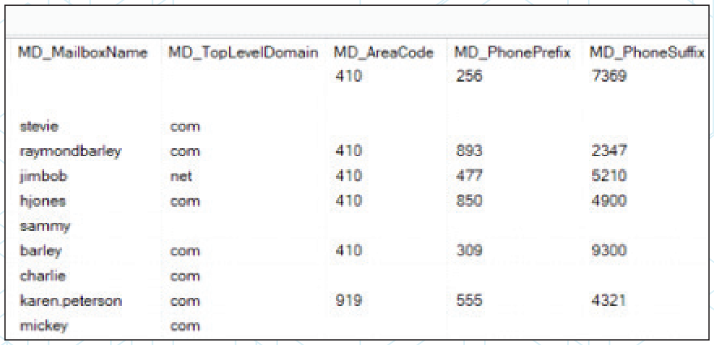
The AS01 result code means the address is valid and deliverable according to official postal agencies.

Phone and Email Results

The following shows the results for the email and phone:



EMAIL	PHONE	MD_Results	MD_DomainName
	410-256-7369	DA01,PS01,PS08,PS11	
		SP01	
stevie@gmaill.com		EE04,ES02,ES21	gmail
raymondbarley@yahoo.com	4108932347	DA01,ES01,ES07,ES21,PS02,PS08	yahoo
jimbob@comcast.net	410.477.5210	DA01,ES01,ES21,PS02,PS08	comcast
hjones@itresourcepartners.com	(410) 850-49...	DA01,EE04,ES02,ES21,PS01,PS08,PS11	itresourcepartners
sammy@		EE01,ES02	
barley@rdacorp.com	410-309-9300	DA01,ES01,ES07,ES21,PS01,PS08,PS11	rdacorp
charlie@yahoo.com		ES01,ES07,ES21	yahoo
karen.peterson@@yahooo.com	919.555.4321	ES01,ES07,ES10,ES12,ES21,PE05	yahoo
mickey@NOSUCHDOMAIN.C...		ES03	nosuchdomain



MD_MailboxName	MD_TopLevelDomain	MD_AreaCode	MD_PhonePrefix	MD_PhoneSuffix
		410	256	7369
stevie	com			
raymondbarley	com	410	893	2347
jimbob	net	410	477	5210
hjones	com	410	850	4900
sammy				
barley	com	410	309	9300
charlie	com			
karen.peterson	com	919	555	4321
mickey	com			

There are quite a few different result codes. The following are a sample of result codes and descriptions:

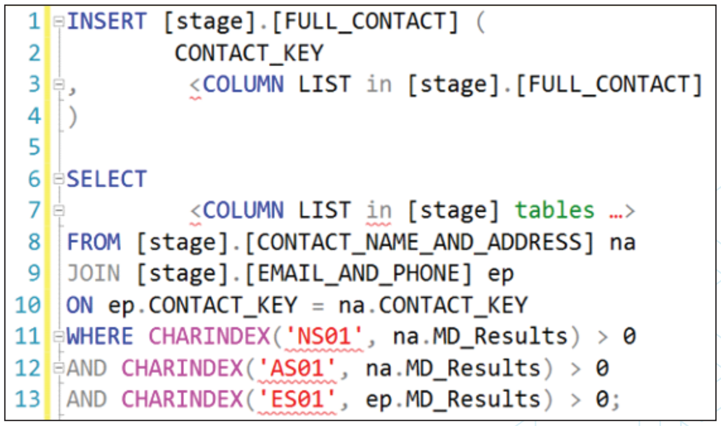
- ES01 is confirmed to be a valid email
- ES02 is confirmed to be an invalid email
- ES07 is an accept all server; every email is reported to be valid
- EE01 is a syntax error in the email address
- EE04 is an invalid mailbox
- PS01 is a valid phone
- PS08 was designated as a land line when activated
- PS11 is a business number

The level of detail in the email and phone result codes is really impressive.

Combine Tables

The final step in the CLEAN, VALIDATE AND ENHANCE task is to combine the CONTACT_NAME_AND_ADDRESS and EMAIL_AND_PHONE tables into the stage FULL_CONTACT table. There is no visual for this one; it is just an Execute SQL task.

The following is an abbreviated listing of the T-SQL:



```
INSERT [stage].[FULL_CONTACT] (
        CONTACT_KEY
,         <COLUMN LIST in [stage].[FULL_CONTACT]
)

SELECT
          <COLUMN LIST in [stage] tables …>
FROM [stage].[CONTACT_NAME_AND_ADDRESS] na
JOIN [stage].[EMAIL_AND_PHONE] ep
ON ep.CONTACT_KEY = na.CONTACT_KEY
WHERE CHARINDEX('NS01', na.MD_Results) > 0
AND CHARINDEX('AS01', na.MD_Results) > 0
AND CHARINDEX('ES01', ep.MD_Results) > 0;
```

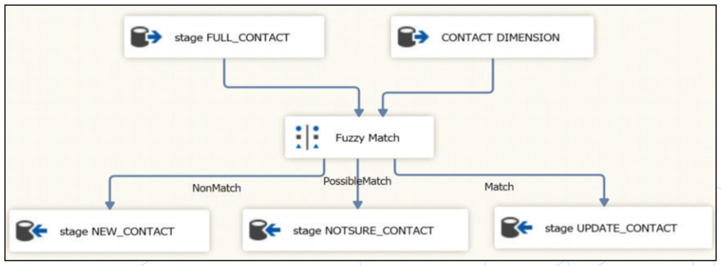
The following are the main points:

- The rows from the CONTACT_NAME_AND_ADDRESS and EMAIL_AND_PHONE tables are joined by the CONTACT_KEY column
- The rows are filtered by the WHERE clause so that only rows that have a successful name parse, a valid address and a valid email are inserted into the FULL_CONTACT table; this is a very conservative approach but I wanted to demonstrate how to only load good Contacts

CHECK FOR DUPLICATES Task

Now that we have cleaned, validated and enhanced our Contacts and loaded them to the FULL_CONTACT table, we need to check for duplicates. We do not want to add any Contact that is already in the Contact dimension. I will use the Fuzzy Match component to perform the duplicate check.

The following is the Data Flow with the Fuzzy Match component:



The following are the main points:

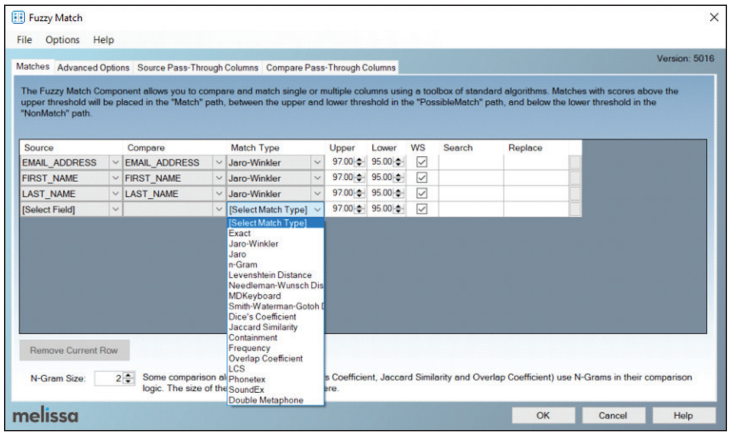
- The Fuzzy Match component take two inputs; they are referred to as the source and compare
- The source is the FULL_CONTACT table which has the Contacts that have just been run through the Personator component
- The compare is the existing Contact dimension
- The Fuzzy Match component compares every row in the source with every row in the compare
- Based on the component configuration (I will walk through that next) the Fuzzy Match directs rows to the three outputs shown above
- Each output is a SQL Server table in my example
- Stage NEW_CONTACT has the combination of source and compare rows that do not match; these are new Contacts
- Stage NOTSURE_CONTACT has the rows where the component is not sure whether the Contact matches or does not match an existing row in the Contact dimension
- Stage UPDATE_CONTACT has the rows where the Contact matches a row in the Contact dimension

I will walk through the configuration of the following tabs in the Fuzzy Match component:

- Matches
- Advanced Options
- Source-Pass Through Columns
- Compare Pass-Through Columns

Matches Tab

The following is the Matches tab:



The Fuzzy Match Component allows you to compare and match single or multiple columns using a toolbox of standard algorithms. Matches with scores above the upper threshold will be placed in the "Match" path, between the upper and lower threshold in the "PossibleMatch" path, and below the lower threshold in the "NonMatch" path.

Source	Compare	Match Type	Upper	Lower	WS	Search	Replace
EMAIL_ADDRESS	EMAIL_ADDRESS	Jaro-Winkler	97.00	95.00	✓		
FIRST_NAME	FIRST_NAME	Jaro-Winkler	97.00	95.00	✓		
LAST_NAME	LAST_NAME	Jaro-Winkler	97.00	95.00	✓		
[Select Field]		[Select Match Type]	97.00	95.00	✓		

[Select Match Type]
Exact
Jaro-Winkler
Jaro
n-Gram
Levenshtein Distance
Needleman-Wunsch Dis
MDKeyboard
Smith-Waterman-Gotoh
Dice's Coefficient
Jaccard Similarity
Containment
Frequency
Overlap Coefficient
LCS
Phonetex
SoundEx
Double Metaphone

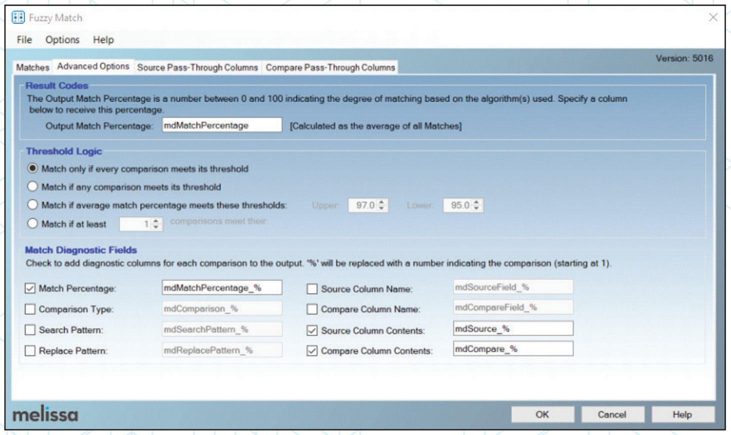
N-Gram Size: 2 Some comparison al s Coefficient, Jaccard Similarity and Overlap Coefficient) use N-Grams in their comparison logic. The size of the ere.

The following are the main points for the Matches tab:

- Select the fields to compare from the source (FULL_CONTACT table) and compare (Contact dimension)
- Select a Match Type for each comparison; choose from the available Fuzzy Match algorithms as shown above
- I am using the same algorithm for each; you are free to select different ones if you like
- Specify the Upper and Lower confidence values; I went with the defaults
- Check WS to ignore whitespace in the comparison

Advanced Options Tab

The following is the Advanced Options tab:



Result Codes

The Output Match Percentage is a number between 0 and 100 indicating the degree of matching based on the algorithm(s) used. Specify a column below to receive this percentage.

Output Match Percentage: mdMatchPercentage [Calculated as the average of all Matches]

Threshold Logic

- ● Match only if every comparison meets its threshold
- ○ Match if any comparison meets its threshold
- ○ Match if average match percentage meets these thresholds: Upper: 97.0 Lower: 95.0
- ○ Match if at least 1 comparisons meet their

Match Diagnostic Fields

Check to add diagnostic columns for each comparison to the output. '%' will be replaced with a number indicating the comparison (starting at 1).

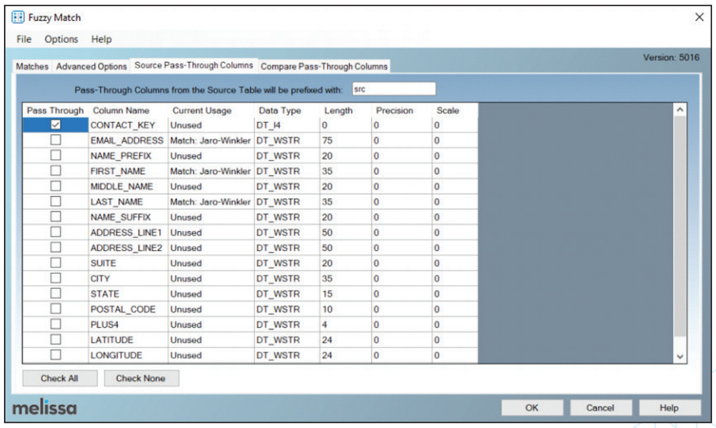
✓ Match Percentage:	mdMatchPercentage_%	☐ Source Column Name:	mdSourceField_%
☐ Comparison Type:	mdComparison_%	☐ Compare Column Name:	mdCompareField_%
☐ Search Pattern:	mdSearchPattern_%	✓ Source Column Contents:	mdSource_%
☐ Replace Pattern:	mdReplacePattern_%	✓ Compare Column Contents:	mdCompare_%

The following are the main points for the Advanced Options tab:

- The Output Match Percentage column will have the overall calculated match percentage for all of the specified comparisons (on the Matches tab); it will appear in every output row
- I chose Match if every comparison meets its threshold for the Threshold Logic; this is the most conservative approach; there are other options available
- The Match Percentage column will have the match percentage for each comparison; it will also appear in every output row
- The Source Columns Contents and Compare Column Contents send the value of the comparison columns to the output

Source Pass-Through Columns Tab

The following is the Source Pass-Through Columns tab:



Pass-Through Columns from the Source Table will be prefixed with: src

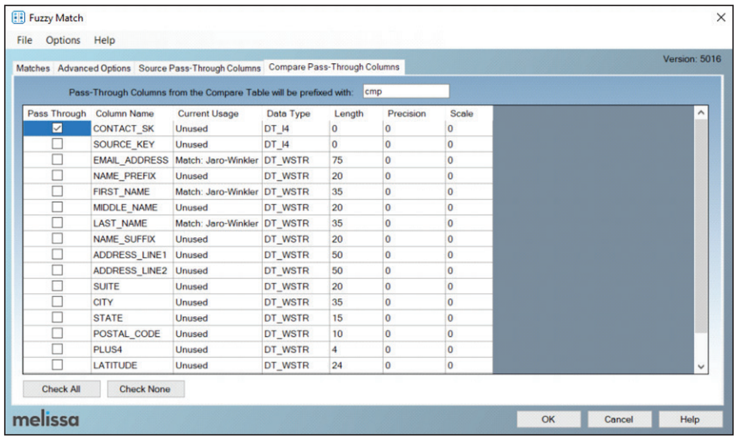
Pass Through	Column Name	Current Usage	Data Type	Length	Precision	Scale
✓	CONTACT_KEY	Unused	DT_I4	0	0	0
☐	EMAIL_ADDRESS	Match: Jaro-Winkler	DT_WSTR	75	0	0
☐	NAME_PREFIX	Unused	DT_WSTR	20	0	0
☐	FIRST_NAME	Match: Jaro-Winkler	DT_WSTR	35	0	0
☐	MIDDLE_NAME	Unused	DT_WSTR	20	0	0
☐	LAST_NAME	Match: Jaro-Winkler	DT_WSTR	35	0	0
☐	NAME_SUFFIX	Unused	DT_WSTR	20	0	0
☐	ADDRESS_LINE1	Unused	DT_WSTR	50	0	0
☐	ADDRESS_LINE2	Unused	DT_WSTR	50	0	0
☐	SUITE	Unused	DT_WSTR	20	0	0
☐	CITY	Unused	DT_WSTR	35	0	0
☐	STATE	Unused	DT_WSTR	15	0	0
☐	POSTAL_CODE	Unused	DT_WSTR	10	0	0
☐	PLUS4	Unused	DT_WSTR	4	0	0
☐	LATITUDE	Unused	DT_WSTR	24	0	0
☐	LONGITUDE	Unused	DT_WSTR	24	0	0

The following are the main points for the Source Pass-Through Columns tab:

- Select the columns from the source (FULL_CONTACT) to include in the output
- I chose the CONTACT_KEY so that I can insert or update the Contact dimension with this value and be able to trace any row in the Contact dimension back to the input

Compare Pass-Through Columns Tab

The following is the Compare Pass-Through Columns tab:



Pass-Through Columns from the Compare Table will be prefixed with: cmp

Pass Through	Column Name	Current Usage	Data Type	Length	Precision	Scale
✓	CONTACT_SK	Unused	DT_I4	0	0	0
☐	SOURCE_KEY	Unused	DT_I4	0	0	0
☐	EMAIL_ADDRESS	Match: Jaro-Winkler	DT_WSTR	75	0	0
☐	NAME_PREFIX	Unused	DT_WSTR	20	0	0
☐	FIRST_NAME	Match: Jaro-Winkler	DT_WSTR	35	0	0
☐	MIDDLE_NAME	Unused	DT_WSTR	20	0	0
☐	LAST_NAME	Match: Jaro-Winkler	DT_WSTR	35	0	0
☐	NAME_SUFFIX	Unused	DT_WSTR	20	0	0
☐	ADDRESS_LINE1	Unused	DT_WSTR	50	0	0
☐	ADDRESS_LINE2	Unused	DT_WSTR	50	0	0
☐	SUITE	Unused	DT_WSTR	20	0	0
☐	CITY	Unused	DT_WSTR	35	0	0
☐	STATE	Unused	DT_WSTR	15	0	0
☐	POSTAL_CODE	Unused	DT_WSTR	10	0	0
☐	PLUS4	Unused	DT_WSTR	4	0	0
☐	LATITUDE	Unused	DT_WSTR	24	0	0

The following are the main points for the Compare Pass-Through Columns tab:

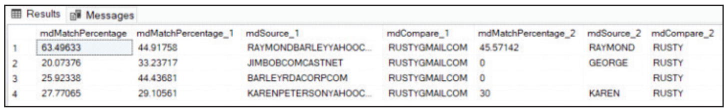
- Select the columns from the compare (Contact dimension) to include in the output
- I chose the CONTACK_SK which is the primary key in the Contact dimension; I use this to update the Contact dimension when a row matches an existing row in the Contact dimension

Next, I will review the results from executing the CHECK FOR DUPLICATES task.

CHECK FOR DUPLICATES Results

To review the results from the CHECK FOR DUPLICATES task, I will show the partial contents of the NEW_CONTACT and UPDATE_CONTACT SQL Server tables. Based on my sample data, the NOTSURE_CONTACT table is empty. The UPDATE_CONTACT table is also empty but only on the initial run.

The following are the partial results in the NEW_CONTACT table from the initial run of the CHECK FOR DUPLICATES task:


Results | Messages

	mdMatchPercentage	mdMatchPercentage_1	mdSource_1	mdCompare_1	mdMatchPercentage_2	mdSource_2	mdCompare_2
1	63.49633	44.91758	RAYMONDBARLEYYAHOOC...	RUSTYGMAILCOM	45.57142	RAYMOND	RUSTY
2	20.07376	33.23717	JIMBOBCOMCASTNET	RUSTYGMAILCOM	0	GEORGE	RUSTY
3	25.92338	44.43681	BARLEYRDACORPCOM	RUSTYGMAILCOM	0		RUSTY
4	27.77065	29.10561	KARENPETERSONYAHOOC...	RUSTYGMAILCOM	30	KAREN	RUSTY

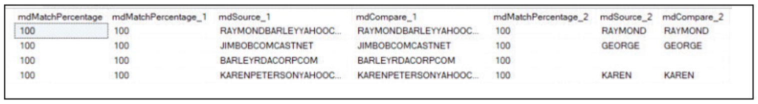
The following are the main points for the results:

- I started out with one row in the Contact dimension to highlight the fact that the Fuzzy Match component matches every row in the source with every row in the compare
- There were only four rows that were valid in the source; each source row gets compared with the one row in the Contact dimension as shown above
- The mdMatchPercentage is the overall match percentage for the three comparisons
- Based on the mdMatchPercentage, no row in the source matches a row in the compare so all rows are new Contacts

The final step in the sample SSIS package is the UPDATE CONTACT DIMENSION task. This is simply an Execute SQL task that inserts new Contacts in the Contact dimension and updates matching Contacts in the Contact dimension. Before reviewing the T-SQL in the UPDATE CONTACT DIMENSION task, I'm going to execute the entire SSIS package a second time. This second run will determine that every valid Contact matches an existing row in the Contact dimension.

Remember I mentioned earlier that the Fuzzy Match component compares every row in the source with every row in the compare. In my simplified example, this means that every valid Contact in the source will be a match for one row in the compare (i.e. the Contact dimension) but it will NOT match every other row in the Contact dimension. So, we will have matching rows in the UPDATE_CONTACT table but those same Contacts will appear in the NEW_CONTACT table because they do not match any other row in the Contact dimension.

The following shows the rows in the UPDATE_CONTACT table after running the sample SSIS package a second time:



mdMatchPercentage	mdMatchPercentage_1	mdSource_1	mdCompare_1	mdMatchPercentage_2	mdSource_2	mdCompare_2
100	100	RAYMONDBARLEYYAHOOC...	RAYMONDBARLEYYAHOOC...	100	RAYMOND	RAYMOND
100	100	JIMBOBCOMCASTNET	JIMBOBCOMCASTNET	100	GEORGE	GEORGE
100	100	BARLEYRDACORPCOM	BARLEYRDACORPCOM	100		
100	100	KARENPETERSONYAHOOC...	KARENPETERSONYAHOOC...	100	KAREN	KAREN

The following are the main points for the results:

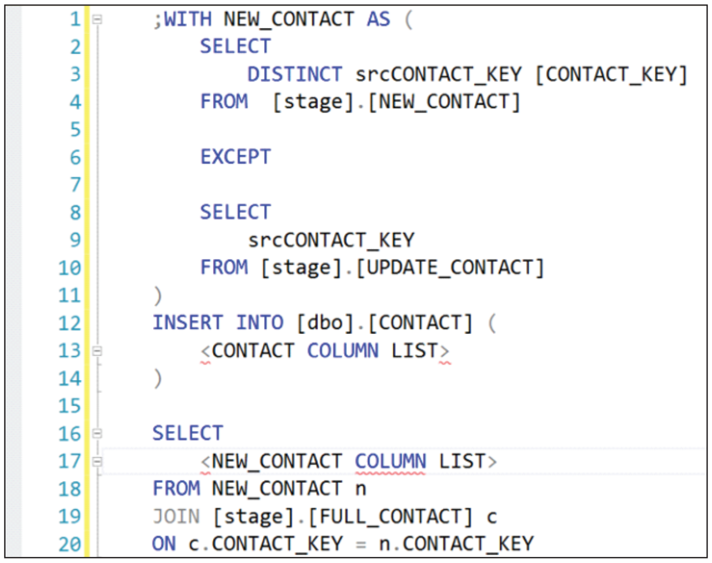
- The mdMatchPercentage for every row is 100%; this is expected since I ran the same input a second time
- Each row in the input is an exact match to one of the rows in the Contact dimension

Next, I will review the T-SQL in the UPDATE CONTACT DIMENSION task.

UPDATE CONTACT DIMENSION Task

This task inserts new Contacts into the Contact dimension and updates matching Contacts in the Contact dimension. For context, assume we are talking about the second run of the sample SSIS package where every Contact matches an existing row in the Contact dimension.

I have a stored procedure named UPSERT_CONTACT to implement this logic. The abbreviated INSERT T-SQL is shown below:



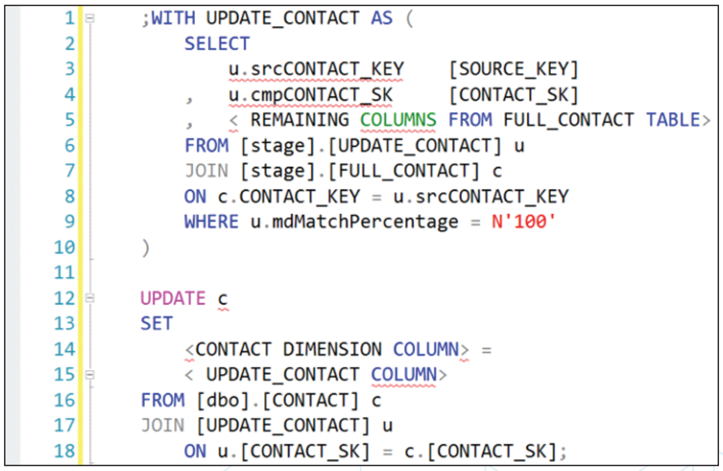
```
;WITH NEW_CONTACT AS (
    SELECT
        DISTINCT srcCONTACT_KEY [CONTACT_KEY]
    FROM  [stage].[NEW_CONTACT]

    EXCEPT

    SELECT
        srcCONTACT_KEY
    FROM [stage].[UPDATE_CONTACT]
)
INSERT INTO [dbo].[CONTACT] (
    <CONTACT COLUMN LIST>
)

SELECT
    <NEW_CONTACT COLUMN LIST>
FROM NEW_CONTACT n
JOIN [stage].[FULL_CONTACT] c
ON c.CONTACT_KEY = n.CONTACT_KEY
```

The following are the main points for the above T-SQL:

- I use a common table expression to get the list of CONTACT_KEY values to be inserted into the Contact dimension
- I get the DISTINCT list of CONTACT_KEY values from the NEW_CONTACT table; remember every Contact in the source is compared to every Contact in the compare so each CONTACT_KEY value will occur in as many rows as there are rows in the Contact dimension that it does NOT match
- I get the list of CONTACT_KEY values from the UPDATE_CONTACT table; these are the source rows that match a row in the compare table; i.e. these are the matching Contacts
- I use EXCEPT to remove each matching CONTACT_KEY value from the DISTINCT list of CONTACT_KEY values from the NEW_CONTACT table that is also in the UPDATE_CONTACT table
- I select every row from the FULL_CONTACT table that matches a CONTACT_KEY value from the common table expression and INSERT it into the Contact dimension
- CONTACT_KEY is in the NEW_CONTACT table because I specified it in the output pass-through columns in the Fuzzy Match component

I can tell you that it took a little bit of testing to figure this one out.

The abbreviated UPDATE T-SQL is shown below:



```
;WITH UPDATE_CONTACT AS (
    SELECT
        u.srcCONTACT_KEY     [SOURCE_KEY]
    ,   u.cmpCONTACT_SK      [CONTACT_SK]
    ,   < REMAINING COLUMNS FROM FULL_CONTACT TABLE>
    FROM [stage].[UPDATE_CONTACT] u
    JOIN [stage].[FULL_CONTACT] c
    ON c.CONTACT_KEY = u.srcCONTACT_KEY
    WHERE u.mdMatchPercentage = N'100'
)

UPDATE c
SET
    <CONTACT DIMENSION COLUMN> =
    < UPDATE_CONTACT COLUMN>
FROM [dbo].[CONTACT] c
JOIN [UPDATE_CONTACT] u
    ON u.[CONTACT_SK] = c.[CONTACT_SK];
```

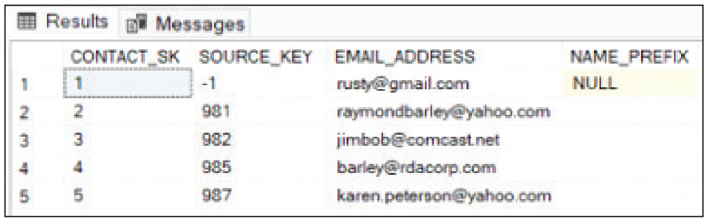
The following are the main points for the above T-SQL:

- I use a common table expression to get the rows from the FULL_CONTACT table that have a 100% match percentage in the UPDATE_CONTACT table
- I update the CONTACT dimension by joining to the common table expression results on CONTACT_SK
- CONTACT_SK is the primary key value in the Contact dimension
- CONTACT_SK is in the UPDATE_CONTACT table because I specified it in the output pass-through columns in the Fuzzy Match component

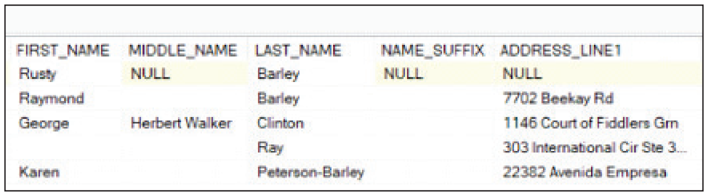
The last thing to look at are the results from executing the UPDATE CONTACT DIMENSION task.

UPDATE CONTACT DIMENSION Results

The following is the list of rows in the Contact dimension (partial list of columns):


Results | Messages

	CONTACT_SK	SOURCE_KEY	EMAIL_ADDRESS	NAME_PREFIX
1	1	-1	rusty@gmail.com	NULL
2	2	981	raymondbarley@yahoo.com	
3	3	982	jimbob@comcast.net	
4	4	985	barley@rdacorp.com	
5	5	987	karen.peterson@yahoo.com	



FIRST_NAME	MIDDLE_NAME	LAST_NAME	NAME_SUFFIX	ADDRESS_LINE1
Rusty	NULL	Barley	NULL	NULL
Raymond		Barley		7702 Beekay Rd
George	Herbert Walker	Clinton		1146 Court of Fiddlers Grn
		Ray		303 International Cir Ste 3...
Karen		Peterson-Barley		22382 Avenida Empresa

The following are the main points for the results:

- The SOURCE_KEY is the CONTACT_KEY from the input
- The initial value of the SOURCE_KEY is the CONTACT_KEY value from the input row that was inserted into the Contact dimension
- When a row in the Contact dimension is updated, the SOURCE_KEY value changes to the CONTACT_KEY value from the input row that updated the Contact dimension

Summary

I hope that I have presented a compelling option for cleaning, validating and enhancing your Contact data. I think the biggest takeaway is that you really want a solution that includes domain-specific knowledge of Contact data, yet is easy to implement in SQL Server Integration Services (SSIS) packages and does not require writing any code.

Next Steps

- To experiment with the demo code, you can download the SSIS Solution here.
- Check out other Melissa content on MSSQLTips.com in the links below
    - Improve Data Quality for SQL Server Reporting
    - Survival of the Fittest: Melissa's Matching Approaches for Golden Record Management
    - Powerful SQL Server Data Cleansing and Processing
    - Fundamentals of SQL Server Data Cleansing
    - Improving Data Quality with SQL Server Integration Services
    - Global Address Data Quality - Storage, Correction, and Verification
    - Make the most of SQL Server Integration Services Script Components

About the author

Ray Barley is a Principal Architect at IT Resource Partners and a MSSQLTips.com BI Expert.

Original Source - https://www.mssqltips.com/sqlservertip/6244/microsoft-sql-server-data-warehouse-data-quality-cleansing-verification-and-matching/

About MSSQLTips.com

MSSQLTips.com is a free community dedicated to SQL Server. Since 2006 our team has delivered value to millions of SQL Server DBAs, Developers and Business Intelligence Professionals on a daily basis. MSSQLTips.com helps solve real world problems and improve your SQL Server knowledge with free tips, tutorials, web casts, videos and more. The MSSQLTips.com team is comprised of **150+ expert authors** led by Greg Robidoux and Jeremy Kadlec. We are focused on sharing knowledge and improving the global SQL Server community every second of the day.

ABOUT MELISSA

Melissa is a leading provider of data quality, identity verification and address management solutions. Melissa helps businesses win and retain customers, validate and correct contact details, optimize their marketing ROI and manage risk. Since 1985, Melissa has been a trusted partner for key industries like retail, education, healthcare, insurance, finance, and government. For more information, visit www.melissa.com or call 1-800-Melissa.