Paper 1850-2014

Adding the Power of DataFlux® to SAS® Programs Using the DQMATCH Function

Pat Taylor, TIMES, University of Houston
Lee Branum-Martin, Georgia State University

ABSTRACT

The SAS Data Quality Server® allows SAS programmers to integrate the power of DataFlux into their data cleaning programs. The power of SAS Data Quality Server enables programmers to efficiently identify matching records across different datasets when exact matches are not present. During a recent educational research project, the DQMATCH function proved very capable when trying to link records from disparate data sources. Two key insights led to even greater success in linking records. The first insight was acknowledging that the hierarchical structure of data can greatly improve success in matching records. The second insight was that the names of individuals can be restructured to improve the chances of successful matches. This paper provides an overview of how these insights were implemented using the DQMATCH function to link educational data from multiple sources.

INTRODUCTION

Many companies and other entities are facing a proliferation of data that is staggering. Unfortunately, much of this data comes from disparate sources. Often times it is necessary to merge these sources to make the best use of the data. Unfortunately, there is not a universal set of standards for how data should be structured. Names, for instance, might be stored as LAST, FIRST or FIRST LAST. Alternatively, first and last names might be separate variables. As given these structures are easily manipulated to easily accommodate merging. However, problems can arise when the LAST, FIRST structure does not include the comma or when nicknames are used. These types of problems can best be addressed by fuzzy matching.

SAS® offers several tools to accomplish fuzzy matching. These include functions in base SAS (COMPCOST, COMPGED, COMPLEV, SOUNDEX, and SPEDIS). Several papers from SAS conferences discuss fuzzy matching with these functions (e.g. Foley, 1999; Staum, 2007). Additional tools for fuzzy matching are provided through the SAS® Data Quality Server. Specifically, the DQMATCH function returns a match code that can be used to match similar values. Use of the DQMATCH function for fuzzy matching was previously discussed by Krumenaker et. al. (2004) and Kincheloe (2007). This paper will discuss the strengths and weaknesses of the DQMATCH function and then offer suggestions to maximize the success rate of fuzzy matching using DQMATCH.

THE DQMATCH FUNCTION

The DQMATCH function takes four arguments and returns a character match code. The first argument is the character value that is the object of the matching. This must be character value in the form of a constant, variable, or expression. The second argument is the match definition which guides the match code generation. Valid values for the match code include 'NAME', 'ADDRESS', 'TEXT', and others. The third argument, the sensitivity level, is an integer value ranging from 50 to 95. The default sensitivity level is 85. Higher values result in stricter matches while lower values allow for matching of strings with greater differences. The final argument is the locale which aids the match definition in parsing the first argument. The following line shows the use of the DQMATCH function to create a match code called mcName85 using the variable TEACHER with a sensitivity setting of 85, the match code NAME, and the locale ENUSA.

```
mcName85 = dqmatch(teacher,'NAME',85,'ENUSA');
```

Table 1 shows the resulting match codes for a set of teacher names. Notice that all values of mcTeacher85 are the same despite the differences in the structure of the name values. The DQMATCH function even recognized the typical shortening of Timothy to Tim. This is true for many combinations of names and nicknames such as Charles-Chuck, Richard-Dick, William-Bill, and Kathleen-Kathy. As a result, DQMATCH works particularly well at the default sensitivity setting when the data is relatively clean. In fact, the names in Table 1 would have resulted in identical match codes even at the highest sensitivity level. However, there are cases when the default level is not as successful.

Teacher	mcTeacher85
Timothy Jones	CB$$$$$$$$$~7B$$$$$$
Tim Jones	CB$$$$$$$$$~7B$$$$$$
Jones Tim	CB$$$$$$$$$~7B$$$$$$
Jones, Tim	CB$$$$$$$$$~7B$$$$$$

Table 1: Match codes for teacher names.

Table 2 shows the results of similar calls to DQMATCH for two different teachers, Mrs. Gray and Mrs. Guerra. Note that at the default sensitivity of 85 (mcTeacher85) identical match codes are generated for names that appear to be quite distinct. Raising the sensitivity level to 95 makes a distinction between the two teachers when middle initials are present. Raising the sensitivity level does not distinguish the two teachers when their first names are shortened to initials. In fact the two names (Maria Gray and Maria Guerra) are not distinguishable when the middle initial is absent even at the highest sensitivity level. Note also in Table 2 that the separate records for each individual were not linked until the sensitivity level was dropped to 60. At that point the sensitivity was so low that all four records received identical match codes. The takeaway here is that more information in the source string results in more accurate clustering.

Teacher	mcTeacher95	mcTeacher85	mcTeacher60
MARIA A GRAY	FY&$$$$$$$$B&YR$&\|$$	FY&$$$$$$$$B&YR$$$$$	FY&$$$$$$$$B&$$$$$$$
MARIA T GUERRA	FY&$$$$$$$$B&YR$~\|$$	FY&$$$$$$$$B&YR$$$$$	FY&$$$$$$$$B&$$$$$$$
M GRAY	FY&$$$$$$$$B$$$$$$$$	FY&$$$$$$$$B$$$$$$$$	FY&$$$$$$$$B$$$$$$$$
M GUERRA	FY&$$$$$$$$B$$$$$$$$	FY&$$$$$$$$B$$$$$$$$	FY&$$$$$$$$B$$$$$$$$

Table 2: Match codes at different sensitivities for two distinct teachers.

The examples presented so far have dealt with relatively clean data. Unfortunately, data is often less than ideal and this presents additional challenges. The data in Table 3 shows teacher names for three distinct teachers (C. Serrato, D. Serrata, and Amanda Zurita) and the groupings created by match codes at different sensitivity levels.

Teacher	95	85	55	50
SERRATA	1	1	1	1
SERRATO	1	1	1	1
ZURITA	1	1	1	1
AMANDA ZURITA	2	2	2	1
C SERRATO	3	3	3	1
SERRATO C	3	3	3	1
SERRATO C(	3	3	3	1
SERRATO C(BI	4	3	3	1
D SERRATA	5	4	4	1
D. SERRATA	5	4	4	1
SERRATA D	5	4	4	1
SERRATA D(BI	6	4	4	1
SERRATA D(BIL)	6	4	4	1

Table 3: Grouping of names for three teachers.

At the default sensitivity level of 85 the DQMATCH function does a good job of distinguishing the teachers when there is information beyond the last name. The function was able to disregard the extraneous information to the right of the "(" character. However, the only adjustment to the sensitivity level that would connect one of the first three records to the its associated records was the lowest level. At that point all 13 records received identical match codes resulting in one group where there should be three. This example suggests a message similar to the prior example. The sensitivity level cannot be adjusted to overcome a lack of information.

The examples in this section illustrate the exceptional ability of the DQMATCH function to create fuzzy matches when there is sufficient information available. DQMATCH does not need the names to be in a consistent format to successfully match names. Even shortened versions of given names are easily matched to the longer versions of those names. However, when there are missing portions of a name or names are shortened to initials then the lack of information quickly diminishes the effectiveness of DQMATCH to appropriately match records. There are even cases when apparently distinct values with full first and last names can result in identical match codes (e.g. Maria Gray and Maria Guerra). The next section will discuss techniques to overcome these issues.

HELPFUL TECHNIQUES

While the DQMATCH function does an excellent job of fuzzy matching, understanding your data can provide opportunities to enhance the results. Often data can be considered hierarchical. That is, sub-units are nested within units. When you know your data is nested in this fashion it is possible to use this information in a manner that seems to augment the information available in the object of a fuzzy match.

Hierarchical Data

One very common example of hierarchical data is found in the field of educational research. This is to say that students are nested within teachers, teachers are nested within grade levels, grade levels within campuses, and campuses within districts. It is true that teachers may change grade levels or even schools across years and students can change teacher, school, or district at any time. However, at any given point in time, the hierarchical structure is steady. In fact, within a year, the structure remains relatively stable. This knowledge can help control the pool of available data when searching for matches.

Similar examples can be found in other fields. Individuals might be nested within a department which in turn is nested within a region. Another example would include families nested within a street, streets nested within a zip-code, zip-codes within a city, cities within counties, and counties within states. Regardless of the context, if data exists in a hierarchical structure then this structure can be used to improve the performance of the DQMATCH function. Consider the data in Table 4.

Teacher	85	55	50
HERNANDEZ	1	1	1
HERNANDEZ	1	1	1
C HERNANDEZ	2	2	1
C. HERNANDEZ	2	2	1
HERNANDEZ C	2	2	1
C HERNANDEZ	2	2	1
C HERNANDEZ	2	2	1
CORIN HERNANDEZ	3	2	1
CARME HERNANDEZ	3	2	1

Table 4: Fuzzy matches ignoring data hierarchy.

At the default sensitivity level, the DQMATCH function has done a nice job of matching all the variations of C HERNANDEZ but the matching of CORIN and CARME HERNANDEZ seems problematic. In addition, the two records without first names are not matched to any other records until the sensitivity level is dropped the minimum. At this point all records are grouped together as one individual. This echoes the lessons learned in the earlier example. The major problem here is that the hierarchical structure of the data has been ignored. Creating matches while leveraging the information available due to nesting will allow for matches to be created on a smaller pool of potential names.

Matches created at the minimum sensitivity level in conjunction with the information regarding school and grade vastly improves the performance of the DQMATCH function. Table 5 show the results from this process using the same names as listed in Table 4. Even at the lowest level of sensitivity, all variations of the names are correctly matched when school and grade are included in the process. This process correctly identified four unique teachers while the process that ignored school and grade identified at most 3 teachers and incorrectly matched CORIN and CARME HERNANDEZ.

School	Grade	Teacher	50
R001D001S001	2	HERNANDEZ	1
R001D001S001	2	C HERNANDEZ	1
R001D001S001	2	C. HERNANDEZ	1
R001D001S001	2	HERNANDEZ C	1
R001D001S018	K	CORIN HERNANDEZ	2
R001D001S029	2	C HERNANDEZ	3
R001D001S029	2	CARME HERNANDEZ	3
R001D001S032	2	HERNANDEZ	4
R001D001S032	2	C HERNANDEZ	4

Table 5: Fuzzy matches exploiting data hierarchy.

This process can be easily implemented by using the FIRST. notation. The section of code below demonstrates how this can be accomplished. First a data step is required to create the match code. Next the data is sorted by increasing restrictive levels of the hierarchy and the match code. Finally, a second data step creates a group variable that is incremented at the first instance of a unique combination of school, grade, and match code.

```
* create a match code for the variable name;
data teachers;
set teachers;
name50 = dqmatch(name,'NAME',50,'ENUSA');
run;

* sort the data by increasingly restricted hierarchy levels and then the match
code;
proc sort data = teachers;
by school grade name50;
run;

* create group numbers using first. notation to increment group counter;
data teachers;
set teachers;
by school grade name50;
length Group 8;
retain Group 0;
if first.name50 then Group + 1;
run;
```

This process allows the user to overcome a lack of information in the target variable by diminishing the potential pool of matching values. This has the effect of increasing the likelihood of accurate matches that would otherwise be lost if the hierarchical structure of the data was ignored. While this technique is quite powerful, the potential exists for individuals to exist within the lowest level of a hierarchy that still resist proper matching by the DQMATCH function.

Name Games

Sometimes names vary due to reasons other than typographical errors or inconsistent data structures. One such instance is the case of hyphenated names that can arise due to marriage. Under such circumstances it is sometimes impossible for the DQMATCH function to properly match records without a little help. For example, the names in Table 6 represent a distinct individual. Even at the lowest sensitivity level the DQMATCH function still returns two distinct match codes for these names.

Name	Group
LASH	1
JESSI LASH	1
J LASHLEDEZ	1
JESSIC LASHLEDEZ	1
J LASHLEDEZMA	1
J. LASH-LEDEZMA	2
JESSICA LASH-LEDEZMA	2

Table 6: Marriage induced name variations.

One way to correct this problem is to augment the list of names by making multiple records from those records that would not match the majority group. In this case those records contain a hyphen. Table 7 shows the augmented name list with a record id that corresponds to a record's position in Table 6. The augmented records were created by creating a new name based on whatever information was present for the first name plus one part of the second name. A new record was created for each part of the second name. When a match code is created for the augmented name list there are still two resulting groups. However, when the data are sorted by record id and group it is clear that at least one new record from each augmented record belongs in the majority group. When sorted in this fashion, the first record in each record id contains the final group number for that record. Take caution when using this technique. Depending on the value of the match code, the final group number may be the last record in each record id.

Original Name	Name	Record ID	Group
LASH	LASH	1	1
JESSI LASH	JESSI LASH	2	1
J LASHLEDEZ	J LASHLEDEZ	3	1
JESSIC LASHLEDEZ	JESSIC LASHLEDEZ	4	1
J LASHLEDEZMA	J LASHLEDEZMA	5	1
J. LASH-LEDEZMA	J. LASH	6	1
J. LASH-LEDEZMA	J. LEDEZMA	6	2
JESSICA LASH-LEDEZMA	JESSICA LASH	7	1
JESSICA LASH-LEDEZMA	JESSICA LEDEZMA	7	2

Table 7: Grouping of augmented name list.

Table 8 shows the final grouping of names bused on taking the first occurring group number for each record id. Several variations of this technique exist. For instance some names were seemingly not recognized by DQMATCH. We believe it was a limitation of the locale that we had licensed. Other locales may have recognized the name. Regardless of the cause, this name was entered as FIRST LAST and LAST, FIRST but was not matched. The records were augmented by creating a record of the original order and then a record with the name order reversed. Subsequent processing proceeded as described here for the hyphenated names.

Original Name	Name	Record ID	Group	Final Group
LASH	LASH	1	1	1
JESSI LASH	JESSI LASH	2	1	1
J LASHLEDEZ	J LASHLEDEZ	3	1	1
JESSIC LASHLEDEZ	JESSIC LASHLEDEZ	4	1	1
J LASHLEDEZMA	J LASHLEDEZMA	5	1	1
J. LASH-LEDEZMA	J. LASH	6	1	1
J. LASH-LEDEZMA	J. LEDEZMA	6	2	1
JESSICA LASH-LEDEZMA	JESSICA LASH	7	1	1
JESSICA LASH-LEDEZMA	JESSICA LEDEZMA	7	2	1

Table 8: Final grouping of augmented name list.

CONCEPTUAL EXAMPLE

Research at TIMES (Texas Institute for Measurement, Evaluation, and Statistics) has generally focused on educational studies. For the majority of these studies, TIMES personnel have been in control of data collection procedures. This level of involvement has allowed us to maintain control over the assignment of ID's to students, teachers, schools, and districts. In addition we are typically able to control data collection procedures such that ID values are assigned at the time of a data collection event. However, in one specific study our level of involvement did not allow any degree of control for the majority of the data collection. For TIMES, this project involves aggregating data from different testing vendors for all students in Kindergarten, 1st, 2nd, and 3rd grade from over 700 campuses in Texas for a period of five years.

The major challenge in this project is the accurate linking of educational records for students from various sources. This is a challenge because identifying information for each observation is recorded differently by each vendor. In addition, while each vendor tends to have unique ID's within a school year the ID's are not always unique across years. Adding to this challenge is the fact that not all students are tested by each vendor. In fact, end of year testing is handled by two separate vendors or the state. This challenge is compounded by size of the project. The vast number of records precludes manual checking of questionable records. Given these challenges, it was necessary for data management at TIMES to develop new data cleaning procedures. It was decided that the approach should leverage the powerful data cleansing capabilities of DataFlux® as made available through the SAS® Data Quality Server. Specifically, the DQMATCH function was used to create fuzzy matches for students across the various data sources.

State records were used as a master list and all other data was linked to that list. The linking began at the highest level of the hierarchy which was district. Direct merging using data steps was first performed. Any records that did not match exactly through the merge were then linked with fuzzy matching by DQMATCH. Once the district names were matched a district id was assigned and then the process moved on to the next level of the hierarchy. We proceeded

through each level of the hierarchy down to the student level implementing the two techniques above when appropriate.

Attempts to link the data through direct merges in data steps resulted in less than 50% matches with no matches at all for data from one vendor. Using the DQMATCH function we were able to link all but 11,065 records of the 5,059,877 records. This represents a success rate of 99.8% and an improvement of about 50%.

CONCLUSION

This paper focused on fuzzy matching of names but the DQMATCH performs admirably for other data types as well. The DQMATCH function provides a versatile tool for performing fuzzy matching. While performance can suffer due to poor data quality or lack of information, careful attention to the available information in your data can enhance performance. Specifically, taking advantage of any hierarchical structure in your data and augmenting your data by transforming problematic records can yield large dividends. By taking full advantage of you data you can use DQMATCH to optimize matching efforts.

REFERENCES

- Foley, Malachy (1999), "Fuzzy Merges: Examples and Techniques" *Proceedings of the Twenty Fourth Annual SAS Users Group International Conference,* 46*.*
- Kincheloe, Faron (2007), "Squeaky Clean Data with SAS® Data Quality Server" *Proceedings of the 2007 SAS Global Forum Conference,* 106*.*
- Krumenaker, Michael; Bukhbinder, George; and Yang, Xiaoyan (2004), "SAS® Data Quality – Cleanse: Techniques for Merge/Purge on Very Large Datasets" *Proceedings of the Twenty Ninth Annual SAS Users Group International Conference,* 14*.*
- Staum, Paulette (2007), "Fuzzy Matching using the COMPGED Function" *Proceedings of the Twentieth Annual NESUG Conference,* 23.

CONTACT INFORMATION

Your comments and questions are valued and encouraged. Contact the author at:

Name:	Pat Taylor
Organization:	TIMES – University of Houston
Address:	2151 W. Holcombe Blvd., Suite 224
City, State ZIP:	Houston TX 77204-5053
Email:	pat.taylor@times.uh.edu