Purge those MERGE Problems: Realistic solutions for accurate and more complete matching of inexact data fields that occur in the everyday world

Judy Palermo, MN Center for Health Statistics, St. Paul, MN

Abstract

This paper describes techniques for match-merging two datasets on common character and numeric data fields used in marketing, health and other application areas- person's name and date of birth. These common data fields are commonly imprecise, as well, with errors or variations in reporting making typical linking by exact agreement insufficient . Studies typically need a very high percentage of successful matches to eliminate possibility of bias etc. Using efficient steps and techniques combining the INDEX , COMPRESS, TRIM , UPCASE, SUM functions and the Soundex Algorithm, many valid matches can be captured that previously could only be made with an individual's exhausting visual inspection.

Introduction

Linking records from datasets created from different origins, each gathering up their own pertinent client information, offers a wealth in new possibilites of variable combinations for analysis by researchers.

Since these disparate datasets usually do not share a unique identifier, it becomes a challenge to link records for a very high rate of matching accuracy as well as for the highest possible match percentage for those two datasets. Linking the correct records together is an obvious requirement; to approach 100% of the data being matched successfully is important to eliminate any bias from having a significant amount of data elements unable to be matched (e.g. if a community clinic in a very impoverished section of town had somewhat imprecise data management, then an inordinate number of these data elements having poverty as a common trait would be disproportionately removed from your analysis).

Using to good advantage the attributes of Proc SQL in Base SAS 6.11 software, executed on a SUN Sparc2 workstation (but applicable on all host operating systems), a process is described here using techniques to match-merge some commonly found fields in database records: name and date of birth. These variables carry strong weight regarding their specific identifying value, but often errors or variations in reported names can occur. Because two datasets have different sources of data collection and data capture, inherently there will be different levels in precision and accuracy. Their types of inaccuracies may differ as well. The typical computer linking procedures requiring exactness have been previously shown to miss a substantial fraction of valid matches, as in a 1994 study linking Medicaid claims data to California vital statistics records.[1] In that study, manual inspection to check myriad variations on last name was made on a small sample (1440) of the total matches made by linkage agreement over a wide range of variables, but there was no way for them to resolve name variations as a whole.

In the real world, names from different database records that do not agree precisely to a computer are considered no match, but a "human eye" reviewing could tell that the names were really of the same person. Manual inspection of the name, which is most accurate in determining matches on inexact entries, is time-consuming, costly and a drain on human energy.

Techniques are described here that can replicate the 'human eye' to successfully create such matches that are inexact, and still keep a high accuracy rate, avoiding false positives, and keep necessary manual inspections to a minimum.

Procedure

For this study we perform a common task in the

health field: linking vital statistics records with managed health care claims data. A successful and reliable linkage offers great opportunities for researchers to do follow-up studies of health status at birth with the level of care needed at a later date.

Data Description

Two simplified subsets of origin files are shown, one of birth certificate records, and the other of managed-health care claims data. In viewing this data to attempt to match-merge the records, examples are shown of common inexact fields such as name, which do not agree precisely, and are not usually matched by the computer, but that the names are really of the same person.

Note that both files originally contained over 100 variables each; for this simplified short example only the variables used for matching, first and last name, and date of birth, are displayed, as well as a unique ID number for each observation per file. This ID number becomes quite useful in our matching process, but also is initially important-variables unneccessary to the matching process do not need to be carried along - a smaller file of each origin containing only the variables used for matching can be created for the task, and later linked by the ID variable to capture the vast remaining information.

Utilizing PROC SQL has great advantages over typical match-merging methods in Base SAS software. For one, no sorting by possible matching variables need be done initially. This alone greatly enhances the freedom to try different combinations of variables to match upon during consecutive passes. It is a very powerful tool , and the techniques in this paper highly benefit from its use.

Data Familiarity

Assumptions are made for this procedure that the programmer has some familiarity of the data files to be matched i.e. which variable or variables in one file may be compared to one in another file; an overall grasp or suspicion of the precision of the data input, and its format. It is also generally important to understand which of the linking variables carry greater weight in uniquely identifying down to an individual, such as first and last name , than one such as birth date, which is strong evidence for a match, but is not by itself uniquely confirming. In this example it was known at the outset that common name variations and inaccuracies may be present, and that while month of birth was considered precise, day of birth may not always have been accurately recorded. We do not need to look at year of birth, since both files are only of 1996 births.

For this simple example, the primary focus is on linking techniques for variations in last name; in reality similar steps needed to be done with first name variables as well. Separate iterations can be made for each combination of exact and inexact first and last names if it is deemed important to know which variable combination created a successful match. However in general this information has no additive value, and typically a single pass, searching both first and last names using inexact methods is sufficient and equally precise.

Conclusion

After studying the following procedure code, and utilizing some of the concepts from it, the advantages and limitations will become apparent. The ideas in this paper are not meant to be an overall solution to the myriad variations that may occur to data in the everyday world, but are eminently practical in providing concrete and workable aids to at least begin to disseminate, and lessen the individual’s visual workload.

Judy Palermo, MN Center for Health Statistics
Phone: (612) 297-1085
Fax: (612) 296-9362
Internet: judy.palermo@health.state.mn.us

Procedure Code

```
data bthcert;
input fname $ 1-10 lname $ 12-25 bcert bmo bday byr;
cfname=upcase(compress(fname,"-' "));
clname=upcase(compress(lname,"-' "));       **A**;
cards;
john            acre kendall      98001 12 25 96
evander         conley jr         76899    8  6 96
donna           lee allison       48234    7 12 96
Erida           Afdjeie           45233    5  5 96
Edwin           Cay               56422   11  9 96
RABEKAH     PALERMO       11233   6 20 96
sarah           o'connell         78767   8 13 96
;
run;

data hlthcare;
 input firstn $ 1-20 lastn $ 22-39 hlthcert $ mo day yr;
cfirstn=upcase(compress(firstn,"-' "));
clastn =upcase(compress(lastn,"-' "));          **A**;
cards;
Linda       Macayrena          HP435688 09  15 96
John        Acre-Kendall       HP211889 12  25 96
Ed          Caye               HP000229 11  9  96
evander     conley             HP676598   8  6  96
Rebecca     Palermo            HP111211   6 20  96
DONNA    LEE ALLISON     HP431230   7 12  96
Erida       Okerstrom Afdjeie  HP557655   5  5   96
Sarah       Oconnoll           HP892133   8 13  96
;
run;

PROC SQL;                                   **B**;
create table pass1 as
select ichi.*,ni.* from
bthcert as ichi full join hlthcare as ni
on ichi.clname=ni.clastn and
   ichi.cfname=ni.cfirstn and
   ichi.bmo   =ni.mo      and
   ichi.bday  =ni.day;

proc print data= PASS1;
var clname clastn cfname cfirstn bcert hlthcert;

title 'PASS 1- no problems here- any matches made
here are valid';
run;
data goodone bthcert2 hlthcar2; set pass1;   **C**;
if bcert=. then output hlthcar2; else
if hlthcert=' '  then output bthcert2;else output
goodone;proc sql;
```

A - At this point of reading in the data from both files we clean up all punctuation irregularities, nested blanks (note that within our COMPRESS function we need to list a "blank" along with the "-" and " ' "). We also make upper-case uniform for matching purposes. Nesting the functions within the one equation makes only one permutation necessary, which becomes crucial after awhile when working with so much detail.

B Now we begin our linking process by the typical method of matching by exactness within the comparable fields.
Resulting dataset is printed in *table 1*.
**Pass1 is a work dataset consisting of good,exact matches in our 1st iteration, plus observations from file bthcert that could not be matched to an observation in file hlthcare, and observations from file hlthcare that could not be matched to an observation in file bthcert.

C This step is to retain and keep all the good,exact matches (GOODONE), separate out the remaining 'leftover' observations i.e. those with missing values for the ID variables bcert and hlthcert for files bthcert and hlthcare, respectively. These were created when one file observation did not find a match with the other.Here is where retention of ID variables from each file becomes important. Those observations with a missing value for bcert become the new origin file of healthcare data, and the observations with a missing value for hlthcert become the new origin file of birth certificate data. In this way we avoid unnecessary attempts to match already matched observations in either the birth certificate or health care files.

```
**D**;
create table pass2 as
select
ichi.clname,ichi.bcert,ichi.bmo,ichi.bday,ichi.cfname,
       ni.clastn,ni.hlthcert,ni.mo,ni.day,ni.cfirstn
from
bthcert2 as ichi full join hlthcar2 as ni
on
sum(index(ichi.clname,trim(ni.clastn)),
   index(ni.clastn,trim(ichi.clname)))>0
and
ichi.bmo=ni.mo and
ichi.bday=ni.day and
ichi.cfname=ni.cfirstn;
proc print data=pass2;

title'Pass2-More good matches made but what if
possible duplicate occurs?';
run;
```

D We perform another iteration, loosening our constraint by using the clever INDEX function that allows us to keep high accuracy. The INDEX function will search for name value in one file within the name value field in the other file. First create numerous possible pairs by linking with exact constraints of month and day and first name, which then can be condensed down, using the INDEX function.
(In actuality, at this point this technique was also simultaneously used for the first name variables as well, and pairs were created only by exact match on month and day).
Notice that we need to, as separate constraints, search for the entire character value of **clname** somewhere in the field of **clastn**, and return the position of its first character within **clastn** . At the same time we must do the converse, and search for the character value of **clastn** within the field of **clname**, and note its first position. Taking their sum is a simple way of checking if either of the positions are non-zero, which would indicate a valid link.)
Note that the TRIM function is crucial here to remove trailing blanks from the default record length.

```
data good2   bthcert3 hlthcar3; set pass2;    **E**;
if bcert=. then output hlthcar3;else
if hlthcert=' '  then output bthcert3;else output good2  ;
run;
```

E This is the same process as first shown in **C**. As further iterations are performed and matches are verified and accepted, the files each containing birth certificate records and managed care records yet to be matched become smaller and smaller.

```
data good2x;set good2;                    **F**;
marker= _N_;
proc sort data=good2x ; by bcert;
proc print data=good2x ; title'good2x';
data good2a good2b;set good2x;
by bcert;
if first.bcert then output good2a;
else output good2b;
```

F What if in our various iterations we create more than one link for an observation, in that for one observation in file bthcert2, we find we have created two possible matches in file hlthcar2, or vice-versa?
This method checks for such a possibility, and can be included at every iteration, if desired:
the created variable MARKER is a simple tag we put to good use for observation viewing.
GOOD2A contains all 1st sightings of variable BCERT; GOOD2B shows any 2nd or more sightings of variable BCERT. This process may then be repeated to check for more than one match of HLTHCERT, the identifier variable for

```
proc print data= good2a;title 'good2a';     **G**;
proc print data= good2b;title 'good2b';
run;

data goodtwo;set good2a good2b;
if marker=3 then delete;

proc print data=goodtwo;
title 'goodtwo-the good matches from 2nd iteration';
run;

proc sql;                                   **H**;
create table pass3 as
select ichi.clname,ichi.bcert,ichi.bmo,ichi.bday,
ichi.cfname,
ni.clastn,ni.hlthcert,ni.mo,ni.day,ni.cfirstn
from
bthcert3 as ichi full join hlthcar3 as ni
on
ichi.clname =* ni.clastn and
ichi.cfname =* ni.cfirstn and
ichi.bmo=ni.mo and
ichi.bday=ni.day;

proc print data=pass3;
title'Pass3-Using the Soundex algorithm to good use';
run;
data goodthre; set pass3;                   **I**;
data finmatch; set goodone goodtwo goodthre;
keep bcert hlthcert bmo bday mo day clname cfname
clastn cfirstn;
 run;

                                            **J**;
indexc(ni.clastn,trim(ichi.x1name),trim(ichi.x2name),tr
im(ichi.x3name));

                                            **K**;
**a copy of **D**, but with arange(+ -2) for
numeric value day allowed for match-merging**;
proc sql;
create table pass2 as
select
ichi.clname,ichi.bcert,ichi.bmo,ichi.bday,ichi.cfname,
       ni.clastn,ni.hlthcert,ni.mo,ni.day,ni.cfirstn
from
bthcert2 as ichi full join hlthcar2 as ni
on
sum(index(ichi.clname,trim(ni.clastn)),
   index(ni.clastn,trim(ichi.clname)))>0 and
ichi.bmo=ni.mo and
abs(ichi.bday-ni.day)<=2 and
```

observations in dataset HLTHCAR2.
G At this point a visual check may be made to see which matched pair should be deleted from the file GOODTWO. The variable MARKER is easier to use than having to refer to the unique BCERT and HLTHCERT identifier combination in order to eliminate the bad match.

H The previous iterations performed solved the common problem of names imbedded due to hypenation or truncation; for the typical problem of slight misspelling of vowels or names with varied spellings, the **Sounds-like Operator** is highly useful.

I At this final point the successful matches are all grouped together, and intermediary markers are removed.

More useful choices:

J Searching for the character string of one variable when it may possibly be found in two or more variable fields in the file to which you wish to link can be done efficiently e.g. woman's last name on the managed-care claim might be found in either of 3 variables on the birth certificate file: child's last name, mother's last name, or mother's maiden last name. Using the INDEXC function the character values for all 3 variables are in one line of code searched for within the source variable to locate the first occurrence of any of the character strings.

K Linking with inexact numeric data is a simple process, one mainly used if known that a numeric entry may be imprecise. This requires some judgement call and knowledge about the history of the data e.g. in date of birth entries, it may be acknowledged that recorded day of birth may be imprecise. But should an entry of 5 instead of 3 be regarded as truly the same date?; couldn't an entry of 3 by human error be recorded as 13?

ichi.cfname=ni.cfirstn;

Pass1- no problems here - any matches made here are valid

OBS	CFNAME	CFIRSTN	CLNAME	CLASTN	MO	BMO	BDAY	DAY
1	JOHN	JOHN	ACREKENDALL	ACREKENDALL	12	12	25	25
2	ERIDA		AFDJEIE		.	5	5	.
3	EDWIN		CAY		.	11	9	.
4		EDWIN		CAYE	11	.	.	9
5		EVANDER		CONLEY	8	.	.	6
6	EVANDER		CONLEYJR		.	8	6	.
7	DONNA	DONNA	LEEALLISON	LEEALLISON	7	7	12	12
8		EDWIN		MACAYRENA	11	.	.	9
9	SARAH		OCONNELL		.	8	13	.
10		SARAH		OCONNOLL	8	.	.	13
11		ERIDA		OKERSTROMAFDJEIE	5	.	.	5
12	RABEKAH		PALERMO		.	6	20	.
13		REBECCA		PALERMO	6	.	.	20

Pass2-More good matches made- but what if possible duplicate occurs?

OBS	CFNAME	CFIRSTN	CLNAME	CLASTN	MO	BMO	BDAY	DAY
1	ERIDA	ERIDA	AFDJEIE	OKERSTROMAFDJEIE	5	5	5	5
2	RABEKAH		PALERMO		.	6	20	.
3		REBECCA		PALERMO	6	.	.	20
4	EVANDER	EVANDER	CONLEYJR	CONLEY	8	8	6	6
5		SARAH		OCONNOLL	8	.	.	13
6	SARAH		OCONNELL		.	8	13	.
7	EDWIN	EDWIN	CAY	MACAYRENA	11	11	9	9
8	EDWIN	EDWIN	CAY	CAYE	11	11	9	9

good2x

OBS	CFNAME	CFIRSTN	CLNAME	CLASTN	MO	BMO	BDAY	DAY	MARKER
1	ERIDA	ERIDA	AFDJEIE	OKERSTROMAFDJEIE	5	5	5	5	1
2	EDWIN	EDWIN	CAY	MACAYRENA	11	11	9	9	3
3	EDWIN	EDWIN	CAY	CAYE	11	11	9	9	4
4	EVANDER	EVANDER	CONLEYJR	CONLEY	8	8	6	6	2

good2a

OBS	CFNAME	CFIRSTN	CLNAME	CLASTN	MO	BMO	BDAY	DAY	MARKER
1	ERIDA	ERIDA	AFDJEIE	OKERSTROMAFDJEIE	5	5	5	5	1
2	EDWIN	EDWIN	CAY	MACAYRENA	11	11	9	9	3
3	EVANDER	EVANDER	CONLEYJR	CONLEY	8	8	6	6	2

good2b

OBS	CFNAME	CFIRSTN	CLNAME	CLASTN	MO	BMO	BDAY	DAY	MARKER
1	EDWIN	EDWIN	CAY	CAYE	11	11	9	9	4

goodtwo-the good matches from 2nd iteration

OBS	CFNAME	CFIRSTN	CLNAME	CLASTN	MO	BMO	BDAY	DAY
1	ERIDA	ERIDA	AFDJEIE	OKERSTROMAFDJEIE	5	5	5	5
2	EVANDER	EVANDER	CONLEYJR	CONLEY	8	8	6	6
3	EDWIN	EDWIN	CAY	CAYE	11	11	9	9

Pass3-Using the Soundex algorithm to good use

OBS	CFNAME	CFIRSTN	CLNAME	CLASTN	MO	BMO	BDAY	DAY
1	RABEKAH	REBECCA	PALERMO	PALERMO	6	6	20	20
2	SARAH	SARAH	OCONNELL	OCONNOLL	8	8	13	13

finmatch-total acceptable matches

OBS	BCERT	HLTHCERT	BMO	BDAY	MO	DAY	CLNAME	CFNAME	CLASTN	CFIRSTN
1	98001	HP211889	12	25	12	25	ACREKENDALL	JOHN	ACREKENDALL	JOHN
2	48234	HP431230	7	12	7	12	LEEALLISON	DONNA	LEEALLISON	DONNA
3	45233	HP557655	5	5	5	5	AFDJEIE	ERIDA	OKERSTROMAFDJEIE	ERIDA
4	76899	HP676598	8	6	8	6	CONLEYJR	EVANDER	CONLEY	EVANDER
5	56422	HP000229	11	9	11	9	CAY	EDWIN	CAYE	EDWIN
6	11233	HP111211	6	20	6	20	PALERMO	RABEKAH	PALERMO	REBECCA
7	78767	HP892133	8	13	8	13	OCONNELL	SARAH	OCONNOLL	SARAH

References

1. Bell Robert, Keesey Joan, Richards Toni,(1994), “The Urge to Merge: Linking Vital Statistics Records and Medicaid Claims”, Medical Care, 32, 1004-1018

2. SAS Institute Inc. (1990), SAS Guide to the SQL Procedure,Version 6, First Edition, Cary,NC:SAS Institute Inc.

3. SAS Institute Inc. (1990), SAS Language: Reference ,Version 6, First Edition, Cary,NC:SAS Institute Inc.

4. SAS Institute Inc. (1990), Getting Started with the SQL Procedure ,Version 6, First Edition, Cary,NC:SAS Institute Inc.