Paper 031-31

Project Duplication: Eradication Techniques

Monal Kohli, I-behavior NY

ABSTRACT

The performance of any statistical model depends largely on clean and reliable set of input data.Duplication of data is a very common problem in everyday working scenario and can lead to ambiguous results and incorrect predictions .There are straightforward techniques from SAS® which help eradicate duplication quickly.But at times, it gets very risky to delete data without actually looking at it. This paper talks in detail about the various techniques which help in identifying duplicates and removing them efficiently. The techniques I will be talking about are NODUPKEY option, NODUPRECS option, By Group processing and hash tables.

INTRODUCTION

Duplication is mainly repetition of values of one or more variables in multiple observations in a dataset. Techniques like NODUPKEY, NODUPRECS or By group processing comes in very handy to remove duplicates from the data. Let's look at an example which shows how duplication affects the results of analysis:

Considering we have a dataset "cust_info" with these variable names:
Cust_No, Name, State, Zip, Ord_no , Ord_total

The data looks like below:

cust_no	name	state	zip	ord_no	ord_total
10001	SHERYL MATHEWS	VA	22030	A0001	85
12003		TN	37312	A0005	125
10001	SHERYL MATHEWS	VA	22030	A0001	85
15623	MARY JACOB	NY	10303	A0009	55
12003		TN	37312	A0005	125
17485	NANY PERKINS	CA	94506	A0003	90
15623	MARY JACOB	NY	10303	A0009	55
17485	NANY PERKINS	CA	94506	A0008	75
12003	JOHN SMITH	TN	37312	A0005	125
17485		CA	94506	A0008	75
10001	SHERYL MATHEWS	VA	22030	A0002	100
17485	NANY PERKINS	CA	94506	A0003	90

To calculate the average order per customer we use the code below:

```
proc sort data=cust_info ;
by cust_no ORD_NO;
run;

DATA cust_info1;
   SET cust_info;
   BY CUST_NO ORD_NO;
   IF FIRST.ORD_NO THEN ORD_NET=0;
   ORD_NET+ord_total;
   IF LAST.ORD_NO THEN OUTPUT;
RUN;

PROC UNIVARIATE DATA=cust_info1;
   VAR ORD_NET;
RUN;
```

A quick snapshot of univariate procedure looks like below:

The SAS System

The UNIVARIATE Procedure
Variable: ORD_NET

Moments

N	6	Sum Weights	6
Mean	180.833333	Sum Observations	1085
Std Deviation	100.32032	Variance	10064.1667
Skewness	1.90347057	Kurtosis	4.0483061
Uncorrected SS	246525	Corrected SS	50320.8333
Coeff Variation	55.4766748	Std Error Mean	40.9555993

Let's look at the average order per customer after removing duplicates:

The UNIVARIATE Procedure
Variable: ORD_NET

Moments

N	6	Sum Weights	6
Mean	88.3333333	Sum Observations	530
Std Deviation	23.5937845	Variance	556.666667
Skewness	0.26141057	Kurtosis	0.73550504
Uncorrected SS	49600	Corrected SS	2783.33333
Coeff Variation	26.7099447	Std Error Mean	9.63212218

The average order per customer with the duplication is $**180** whereas the average order per customer without duplication is $88.This is a perfect example of how duplication can lead to invalid results during analysis. I will be hereby reviewing the techniques individually to remove duplication efficiently. This dataset CUST_INFO will be used throughout the paper for explanation of different techniques.

NODUPKEY OPTION

This option is used in conjunction with PROC SORT. Before deleting duplication, it is imperative to find the variables by which the duplication occurs. Those variables contribute to the BY variables group of the PROC SORT . If this option is specified with PROC SORT,all the BY values for each observation is compared to those for the previous observation written to the output data set. If an exact match is found, the observation is not written to the output data set.

Let's apply the NODUPKEY option to the above data and see the results:

```
Proc sort data=cust_info  nodupkey out =temp;
By cust_no ord_no;
run;
Proc print data=temp;
run;
```

This is how the dataset temp looks after execution of the above steps:

cust_no	name	state	zip	ord_no	ord_total
10001	SHERYL MATHEWS	VA	22030	A0001	85
10001	SHERYL MATHEWS	VA	22030	A0002	100
12003		TN	37312	A0005	125
15623	MARY JACOB	NY	10303	A0009	55
17485	NANY PERKINS	CA	94506	A0003	90
17485	NANY PERKINS	CA	94506	A0008	75

After going through the results, the cust_no "12003" seems to be missing the name even though the dataset CUST_INFO has a value "JOHN SMITH" for this cust_no. That's tricky now!!!!

Let's analyze what happened in the step above. When we sort the dataset CUST_INFO, SAS keeps the first occurrence of cust_no,ord_no.Considering this cust_no "12003", the first occurrence has the name missing and that is the observation which is kept after removing duplication.The normal sort order in SAS is "ascending to descending" or in other words "smallest to largest". To reverse the sort order SAS provides with a DESCENDING option in the BY statement.

DESCENDING OPTION

This option is specified on the BY statement .The DESCENDING option should be specified before the variable which needs to be sorted in the descending order. We need to select the observation which has a name populated at the top, so that is the observation which would be selected.

Let's apply NODUPKEY option with the DESCENDING option:

```
Proc sort data=cust_info  out=temp1;
By cust_no ord_no descending name;
run;
Proc sort data=temp1  nodupkey out =temp;
By cust_no  ord_no;
run;
Proc print data=temp;
run;
```

This is how the dataset temp1 looks after sorting by descending name:

cust_no	name	state	zip	ord_no	ord_total
10001	SHERYL MATHEWS	VA	22030	A0001	85
10001	SHERYL MATHEWS	VA	22030	A0001	85
10001	SHERYL MATHEWS	VA	22030	A0002	100
12003	JOHN SMITH	TN	37312	A0005	125
12003		TN	37312	A0005	125
12003		TN	37312	A0005	125
15623	MARY JACOB	NY	10303	A0009	55
15623	MARY JACOB	NY	10303	A0009	55
17485	NANY PERKINS	CA	94506	A0003	90
17485	NANY PERKINS	CA	94506	A0003	90
17485	NANY PERKINS	CA	94506	A0008	75
17485		CA	94506	A0008	75

This is how the dataset temp looks after the final sort step:

cust_no	name	state	zip	ord_no	ord_total
10001	SHERYL MATHEWS	VA	22030	A0001	85
10001	SHERYL MATHEWS	VA	22030	A0002	100
12003	JOHN SMITH	TN	37312	A0005	125
15623	MARY JACOB	NY	10303	A0009	55
17485	NANY PERKINS	CA	94506	A0003	90
17485	NANY PERKINS	CA	94506	A0008	75

After using the descending option before deduping the data, we have the observations with the name populated at the top, so those observations with the populated names will be kept after NODUPKEY is executed.Thus the nodupkey option ensures that there is only one unique observation for each cust_no and ord_no.**NODUPKEY was one of the techniques to remove duplicates.Lets look into another technique to perform the same task.**

NODUPRECS OPTION

We learned above that the NODUPKEY option compares only the variables specified in the BY statement to remove duplication.The way NODUPRECS option differs from NODUPKEY is that is checks all the variables in the previous observation that was written to the output dataset to the observation in the PDV before writing out that observation.If they are exactly same then the observation in the PDV is not written to the output dataset.Lets apply NODUPRECS to CUST_INFO dataset and see the results:

```
Proc sort data=cust_info  out=temp1;
By cust_no  ord_no descending name;
run;
Proc sort data=temp1  noduprecs  out =temp;
By cust_no  ord_no;
run;
Proc print data=temp;
run;
```

This is how the datastep temp looks after the execution of above steps:

cust_no	name	state	zip	ord_no	ord_total
10001	SHERYL MATHEWS	VA	22030	A0001	85
10001	SHERYL MATHEWS	VA	22030	A0002	100
12003	JOHN SMITH	TN	37312	A0005	125
12003		TN	37312	A0005	125
15623	MARY JACOB	NY	10303	A0009	55
17485	NANY PERKINS	CA	94506	A0003	90
17485	NANY PERKINS	CA	94506	A0008	75
17485		CA	94506	A0008	75

Well it doesn't look like all the duplicates have been removed. There are two sets of cust_no,ord_no which still have duplicates. They are (12003 ,A0005) and (17485,A0008). Let's analyze the results:

This is the dataset we have after sorting the name in descending order:

cust_no	name	state	zip	ord_no	ord_total	OBS
10001	SHERYL MATHEWS	VA	22030	A0001	85	_N_=1
10001	SHERYL MATHEWS	VA	22030	A0001	85	_N_=2
10001	SHERYL MATHEWS	VA	22030	A0002	100	_N_=3
12003	JOHN SMITH	TN	37312	A0005	125	_N_=4
12003		TN	37312	A0005	125	_N_=5
12003		TN	37312	A0005	125	_N_=6
15623	MARY JACOB	NY	10303	A0009	55	_N_=7
15623	MARY JACOB	NY	10303	A0009	55	_N_=8
17485	NANY PERKINS	CA	94506	A0003	90	_N_=9
17485	NANY PERKINS	CA	94506	A0003	90	_N_=10
17485	NANY PERKINS	CA	94506	A0008	75	_N_=11
17485		CA	94506	A0008	75	_N_=12

Let's look at how the NODUPRECS executes:

The first observation is read into PDV and written to the output dataset.The second observation is read into PDV and compared to _N_=1 observation,they are exactly same ,so this observation is skipped.Third observation _N_=3 is read into the PDV and compared to _N_=1 which is in the output dataset,since they are different _N_=3 is written to the output dataset.Fourth observation _N_=4 is read into the PDV and compared to _N_=3 ,since they are different fourth observation is written to the dataset. Now fifth observation (_N_=5) is read into the PDV and compared to _N_=4 observation, since they are not exactly same this observation is also written to the dataset.

Sixth observation (_N_=6) is exactly similar to the fifth observation which was written out to the dataset. This is an explanation of few steps, but do we know what the issue here is? Do we???
Here's what's happening.The observation 5th and 6th with cust_no 12003 and ord_no A0005 are exactly same ,so one of them is written out.But we also know that they are "JOHN SMITH's" orders since the customer number are same.
So how do we remove the duplicate records here? This is what we need to do.
If one of the cust_no has name populated in it, we can populate all the observation for that cust_no with that name.
Let's see how to implement it.

```
Proc sort data=cust_info out=temp1;
By cust_no ord_no  descending name;
run;
data temp1;
set temp1;
By cust_no ord_no descending name;
retain fullname;
if first.cust_no then fullname=name;
name=fullname;
drop fullname;
run;
```

This is how the dataset temp1 looks like:

cust_no	name	state	zip	ord_no	ord_total
10001	SHERYL MATHEWS	VA	22030	A0001	85
10001	SHERYL MATHEWS	VA	22030	A0001	85
10001	SHERYL MATHEWS	VA	22030	A0002	100
12003	JOHN SMITH	TN	37312	A0005	125
12003	JOHN SMITH	TN	37312	A0005	125
12003	JOHN SMITH	TN	37312	A0005	125
15623	MARY JACOB	NY	10303	A0009	55
15623	MARY JACOB	NY	10303	A0009	55
17485	NANY PERKINS	CA	94506	A0003	90
17485	NANY PERKINS	CA	94506	A0008	75
17485	NANY PERKINS	CA	94506	A0003	90
17485	NANY PERKINS	CA	94506	A0008	75

We do see that name is populated for all the observations, so lets look apply NODUPRECS to temp1:

```
Proc sort data=temp1 noduprecs out =temp;
By cust_no  ord_no;
run;
Proc print data=temp;
run;
```

This is how the dataset temp looks like:

cust_no	name	state	zip	ord_no	ord_total
10001	SHERYL MATHEWS	VA	22030	A0001	85
10001	SHERYL MATHEWS	VA	22030	A0002	100
12003	JOHN SMITH	TN	37312	A0005	125
15623	MARY JACOB	NY	10303	A0009	55
17485	NANY PERKINS	CA	94506	A0003	90
17485	NANY PERKINS	CA	94506	A0008	75

Finally, the results are same as NODUPKEY but we have to add additional steps to get the same results.
Let's look into another powerful technique to handle duplication.
BY GROUP PROCESSING

By group processing is a very powerful feature of SAS.In BY Group processing SAS processes data in groups. The variables by which how the data is grouped is specified on the by statement. The BY group processing can be used if the variables defined on the BY statement are sorted in that order or if an index is created on those variables or if the data that is read is grouped in that order. When a BY statement is used with a set statement, SAS creates two temporary variables FIRST.variable and LAST.variable.
Since we have sorted the cust_info by cust_no ord_no descending name ,lets look at the BY group processing for the same dataset.

```
Data temp1;
Set  temp1;
By cust_no ord_no;
If first.cust_no then first_cust_no=1;
If first.ord_no then first_ordno=1;
If last.cust_no then last_custno=1;
If last.ord_no then last_ordno=1;
Run;
```

cust_no	name	state	zip	ord_no	ord_total	first_custno	last_custno	first_ordno	last_ordno
10001	SHERYL MATHEWS	VA	22030	A0001	85	1		1	
10001	SHERYL MATHEWS	VA	22030	A0001	85				1
10001	SHERYL MATHEWS	VA	22030	A0002	100		1	1	1
12003	JOHN SMITH	TN	37312	A0005	125	1		1	
12003		TN	37312	A0005	125				
12003		TN	37312	A0005	125		1		1
15623	MARY JACOB	NY	10303	A0009	55	1		1	
15623	MARY JACOB	NY	10303	A0009	55		1		1
17485	NANY PERKINS	CA	94506	A0003	90	1		1	
17485	NANY PERKINS	CA	94506	A0003	90				1
17485	NANY PERKINS	CA	94506	A0008	75			1	
17485		CA	94506	A0008	75		1		1

As we know the first and last members of (cust_no ord_no) group we can separate the duplicates to another dataset.How do we do that?We use conditional processing with first and last variables like this:

```
Data  nodups dups;
Set temp1;
By cust_no ord_no;
If first.ord_no and last.ord_no eq 1 then output nodups;
Else output dups;
Run;
```

There is only one observation in nodups dataset and 11 observations in the dups dataset.So now we have seen how to separate the duplicates from the nonduplicates.Let's see how to get a count of duplicates for each cust_no and ord_no group.

```
Proc sort data=cust_info  out=temp1;
By cust_no ord_no  descending name;
run;

data temp1(keep=cust_no ord_no cnt_duplicates);
set temp1;
By cust_no ord_no  ;
if first.ord_no then cnt_duplicates=0;
cnt_duplicates+1;
if last.ord_no then output;
run;
```

This is the output of the above step:

cust_no	ord_no	cnt_duplicates
10001	A0001	2
10001	A0002	1
12003	A0005	3
15623	A0009	2
17485	A0003	2
17485	A0008	2

Now that we know how to get the counts for duplicate order_no ,let's look at how to eliminate the duplication from cust_info dataset. Let's see how to accomplish this:

```
Proc sort data=cust_info  out=temp1;
By cust_no ord_no  descending name;
run;
data temp1;
set temp1;
By cust_no ord_no  descending name;
data cust_info_nodups;
set temp1;
by cust_no ord_no ;
if first.ord_no;
run;
```

This is how the final dataset cust_info_nodups looks like:

cust_no	name	state	zip	ord_no	ord_total
10001	SHERYL MATHEWS	VA	22030	A0001	85
10001	SHERYL MATHEWS	VA	22030	A0002	100
12003	JOHN SMITH	TN	37312	A0005	125
15623	MARY JACOB	NY	10303	A0009	55
17485	NANY PERKINS	CA	94506	A0003	90
17485	NANY PERKINS	CA	94506	A0008	75

So we have successfully removed all the duplication from cust_info dataset.Let's look at another new technique to remove duplication.

HASH TABLES

Hash tables are one of the predefined component objects which were introduced in SAS version 9.Another component object is the hash iterator.The hash table is a quick way of storing,searching and retrieving data by using keys.The keys are dataset variables and data is stored in key data pair. So the key is used to search or store data in memory. A hash table is basically a table in memory which facilitates for better lookup techniques. A key is used to point to a location in the table where actual data will be stored. The key can be both character and numeric. We can also create composite keys. A component object interface is used to manipulate these component objects using statements and methods. We use an object dot notation to manipulate the objects. With all this basic information about component objects let's look at the solution for removing duplication from cust_info dataset. The methods used in the solution are REPLACE( ),DEFINEKEY( ),DEFINEDATA( ),DEFINEDONE( ) ,NEXT( ) ,FIRST( )and OUTPUT( ). Let's look at the implementation now:

```
Proc sort data=cust_info out=temp;
by cust_no ord_no name ;
run;
data nodups ;
      dcl hash cust   () ;
      cust.definekey  ('cust_no', 'ord_no') ;
      cust.definedata ('name', 'state', 'zip','ord_total'  ) ;
      cust.definedone () ;
        do _n_ = 1 by 1 until (last.ord_no) ;
         set temp;
         by cust_no ord_no ;
         cust.replace() ;
         cust.output();
      end ;
run ;
```

proc print data=nodups;run;

This is how the dataset nodups looks like after the above step executes:

cust_no	name	state	zip	ord_no	ord_total
10001	SHERYL MATHEWS	VA	22030	A0001	85
10001	SHERYL MATHEWS	VA	22030	A0002	100
12003	JOHN SMITH	TN	37312	A0005	125
15623	MARY JACOB	NY	10303	A0009	55
17485	NANY PERKINS	CA	94506	A0003	90
17485	NANY PERKINS	CA	94506	A0008	75

In the above implementation ,we basically create a hash table cust with a composite key consisting of cust_no and ord_no .The data to be linked to the key is Name,State,Zip and Ord_total.The function replace makes sure that only unique keys are added to the hash table.In the above step ,the dataset cust_info is sorted by cust_no,ord_no and name.So the name is sorted as ascending to descending and as a result the populated record is at the bottom and that is the record added to the hash table.

This is one of the best methods to remove duplication .This technique can also be used if the data is grouped together since the By statement can use the NOTSORTED option.

CONCLUSION

The Nodupkey option removes duplication in one PROC SORT step whereas NODUPRECS option might take additional steps based on the data to remove duplication completely.By group processing gives an option of saving the duplicates for future reference unlike nodupkey or noduprecs option where there is no way to track the data deleted.Thus BY Group processing can turn out to be more helpful in cases where it is hard to sort the entire dataset and the datasets is either grouped by the "By variables" or it has index defined on those variables.Hash table is a very new feature in SAS but it surely is a powerful way of removing duplication.Both BY group processing and Hashing can prove very helpful in removing duplicates in cases where data is grouped together and it is difficult to sort the data.

REFERENCES

SUGI-30: Paul M. Dorfman Koen Vyverman
DATA Step Hash Objects as Programming Tools
SUGI 29: Paul Dorfman Koen Vyverman
Hash Component Objects: Dynamic Data Storage and Table Look-Up
SUGI 28: Paul M. Dorfman Gregg P. Snell
Hashing: Generations
SUGI27: Paul M. Dorfman Gregg P. Snell
Hashing Rehashed

CONTACT INFORMATION

Author Name	Monal Kohli
Company	I-behavior
Address	500 Mammaroneck ave
City state ZIP	Harrison NY 10528
Work Phone:	9147987752
Fax:	9147777731
Email:	monal@i-behavior.com
Web:	