Paper 106-25

REMOVING DUPLICATES: PROC SQL CAN HELP YOU "SEE"

Vicki Siemers, Damark International, Inc., Minneapolis, MN
Brian Nash, Conseco Finance Corp., St. Paul, MN

Abstract

Duplication of information within data sets is a common occurrence. Data files may contain duplicate account numbers, telephone numbers, names, etc. A standard/accepted solution for removing duplicates is the NODUPKEY option of PROC SORT. Unfortunately, this procedure is often used blindly. The first duplicate observation is kept in the data set while all subsequent occurrences are deleted. But what if the third occurrence should have been kept and the first and second occurrences deleted? PROC SQL, a component of Base SAS, can help SAS® users investigate duplicates so that a more informed choice can be made when deleting observations. This paper is intended for users with an intermediate SAS programming skill set.

Introduction

Generally, the first phase in any analysis project is to check the integrity of the data (e.g., remove duplicates, ensure valid values, etc.). The project presented in this paper involved both a test and control group. The first step was to verify that the groups were mutually exclusive (i.e., the same observation did not appear in both groups). In an ideal situation, the test and control groups would always be mutually exclusive. However, this is often not the case. In this project's data set, it was found that the test and control groups were not mutually exclusive. Therefore, PROC SQL was used to ensure that, if a duplicate occurred, the observation was kept in the desired group and removed from the other group.

Data

The following data set, DATAACCT.DAT, will be used throughout this paper:

The variables contained in this data set are:

- 5 Digit Account Number
- First Name
- Last Name
- State
- Telephone Number
- Status

The duplicate observations are highlighted.

```
----+----1----+----2----+----3----+----4---
12345 AMBER    ERICKSON MN 507-012-3456 TEST
67890 JOAN     HALLEY   WI 608-001-2345 CNTL
98765 KEVIN    TRACEN   TX 817-098-7654 TEST
00001 KYLE     RANGER   FL 941-555-1234 TEST
76543 HEATHER  SAMPSON  FL 941-555-1235 TEST
33333 ALICE    ANDERSON FL 941-555-1236 CNTL
44455 HOWARD   SMITH    GA 706-555-0001 TEST
12345 AMBER    ERICKSON MN 507-012-3456 CNTL
55666 GEORGE   WILLIAMS GA 706-555-0002 CNTL
66777 NANCY    BECK     GA 706-555-0003 CNTL
77788 MARIA    ALVAREZ  TX 817-003-4567 TEST
98765 KEVIN    TRACEN   TX 817-098-7654 CNTL
88899 TINA     HOLT     OR 503-004-5678 CNTL
67890 JOAN     HALLEY   WI 608-001-2345 TEST
99900 WALTER   DENNIS   MN 612-009-2345 TEST
```

PROC SORT (The "Blind" Method)

The initial approach to check the data was:

1. Read in the data file.
2. Verify that there are no duplicate account numbers by applying the NODUPKEY option of PROC SORT.
3. Check the notes in the SAS Log.

```
* READ IN THE DATA FILE.;
DATA ACCTDAT;
   INFILE "DATAACCT.DAT";
   INPUT @1  ACCT   $CHAR5.
         @7  FNAME  $CHAR7.
         @15 LNAME  $CHAR8.
         @24 STATE  $CHAR2.
         @27 PHONE  $CHAR12.
         @40 STATUS $CHAR4.;

* REMOVE DUPLICATES BLINDLY USING PROC SORT.;
PROC SORT DATA=ACCTDAT NODUPKEY;
   BY ACCT;
```

Below is the partial log:

```
28          PROC SORT DATA=ACCTDAT NODUPKEY;
29             BY ACCT;

NOTE: 3 observations with duplicate key values
      were deleted.
NOTE: The data set WORK.ACCTDAT has 12
      observations and 6 variables.
NOTE: The PROCEDURE SORT used 0.01 CPU seconds
      and 3037K.
```

At this point, the duplicate account numbers had been identified. However, it was unknown if the duplicates existed only in the test group, only in the control group, or if identical account numbers appeared in both groups. Further investigation was needed.

PROC SQL: An Investigative Tool

PROC SQL was a logical choice for researching the observations with duplicate account numbers because of these three features:

- The FREQ summary function counts the number of times an account number occurs within the group.
- The GROUP BY clause allows PROC SQL to process the FREQ summary function on a specified group (i.e., account number).
- The HAVING clause tells PROC SQL to print the results only when an account number appears multiple times in the file.

The above process (shown in the example below) identifies all occurrences of the duplicate accounts. That is, if an account number appears in the file twice, both observations are printed.

```
* READ IN THE DATA FILE.;
DATA ACCTDAT;
   INFILE "DATAACCT.DAT";
   INPUT @1  ACCT   $CHAR5.
         @7  FNAME  $CHAR7.
         @15 LNAME  $CHAR8.
         @24 STATE  $CHAR2.
         @27 PHONE  $CHAR12.
         @40 STATUS $CHAR4.;

* IDENTIFY DUPLICATES BY COUNTING OCCURRENCES.;
TITLE "DATA FILE - ACCOUNT INFO";
TITLE2 "DUPLICATES IDENTIFIED BY COUNTING # OF
        OCCURRENCES";
```

```
PROC SQL;
   SELECT ACCT,
          FNAME,
          LNAME,
          STATE,
          PHONE,
          STATUS,
          FREQ(ACCT) AS NUMACCT /*# x Acct in file*/
   FROM ACCTDAT
   GROUP BY ACCT
   HAVING NUMACCT GE 2;
```

The output below shows each occurrence of the duplicate account numbers. By looking at the variable STATUS, it was easy to identify whether the particular observation existed in the test or control group.

DATA FILE - ACCOUNT INFO
DUPLICATES IDENTIFIED BY COUNTING # OF OCCURRENCES

ACCT	FNAME	LNAME	STATE	PHONE	STATUS	NUMACCT
12345	AMBER	ERICKSON	MN	507-012-3456	TEST	2
12345	AMBER	ERICKSON	MN	507-012-3456	CNTL	2
67890	JOAN	HALLEY	WI	608-001-2345	CNTL	2
67890	JOAN	HALLEY	WI	608-001-2345	TEST	2
98765	KEVIN	TRACEN	TX	817-098-7654	TEST	2
98765	KEVIN	TRACEN	TX	817-098-7654	CNTL	2

The output shows that each duplicate account number occurred once in the test group and once in the control group.

PROC SORT (The In"Sight"ful Method)

After speaking with the project leader, it was discovered that the duplicates occurred because a portion of the control group inadvertently received the test offer. Because the customer received the test offer, they had to be removed from the control group. The following process was used to correct the problem before the analysis continued.

1. Sort the data set by ascending account number and descending status. Because STATUS is a character variable, sorting in descending order will organize the column in reverse alphabetical order (i.e., T before C). Thus, the test observation will occur before the control observation for each duplicate account.
2. At this point, PROC SORT with the NODUPKEY option was used because, by default, it always keeps the first occurrence and drops everything else. Thus, for every duplicate account number, the test observation was kept while the control observation was deleted.

```
* READ IN THE DATA FILE.;
DATA ACCTDAT;
   INFILE "DATAACCT.DAT";
   INPUT @1  ACCT   $CHAR5.
         @7  FNAME  $CHAR7.
         @15 LNAME  $CHAR8.
         @24 STATE  $CHAR2.
         @27 PHONE  $CHAR12.
         @40 STATUS $CHAR4.;

* KEEP OBSERVATION WITH TEST STATUS IF THERE ARE
DUPLICATES BY ACCOUNT NUMBER.;
PROC SORT DATA=ACCTDAT;
   BY ACCT DESCENDING STATUS;

PROC SORT DATA=ACCTDAT NODUPKEY;
   BY ACCT;

PROC PRINT DATA=ACCTDAT NOOBS;
   TITLE "DATA FILE - ACCOUNT INFO";
   TITLE2 "DUPS CORRECTLY REMOVED BY PROC SORT";
```

DATA FILE - ACCOUNT INFO
DUPS CORRECTLY REMOVED BY PROC SORT

ACCT	FNAME	LNAME	STATE	PHONE	STATUS
00001	KYLE	RANGER	FL	941-555-1234	TEST
12345	AMBER	ERICKSON	MN	507-012-3456	TEST
33333	ALICE	ANDERSON	FL	941-555-1236	CNTL
44455	HOWARD	SMITH	GA	706-555-0001	TEST
55666	GEORGE	WILLIAMS	GA	706-555-0002	CNTL
66777	NANCY	BECK	GA	706-555-0003	CNTL
67890	JOAN	HALLEY	WI	608-001-2345	TEST
76543	HEATHER	SAMPSON	FL	941-555-1235	TEST
77788	MARIA	ALVAREZ	TX	817-003-4567	TEST
88899	TINA	HOLT	OR	503-004-5678	CNTL
98765	KEVIN	TRACEN	TX	817-098-7654	TEST
99900	WALTER	DENNIS	MN	612-009-2345	TEST

Conclusion

There are many techniques for investigating data. The method used above is a straightforward, intuitive approach to assist in verifying data integrity. This paper presented one practical application of PROC SQL. It is the authors' hope that SAS users will see how easy and powerful PROC SQL can be and will investigate its use in other applications.

References

SAS Institute Inc., *SAS Language: Reference, Version 6, First Edition*, Cary, NC 1990
SAS Institute Inc., *SAS Procedures Guide: Version 6, Third Edition*, Cary, NC 1990
SAS Institute Inc., *Getting Started with the SQL Procedure, Version 6, First Edition*, Cary, NC 1994
SAS Institute Inc., *SAS Guide to the SQL Procedure: Usage and Reference, Version 6, First Edition*, Cary, NC 1989

Contact Information

Vicki Siemers
Damark International, Inc.
Marketing Analysis
7101 Winnetka Ave N
Minneapolis, MN 55428
(612) 531-3110
(612) 531-3095 FAX
vicki.siemers@damark.com

Brian Nash
Conseco Finance Corp.
332 Minnesota Street
Suite 610
St. Paul, MN 55101
(651) 265-7368
(651) 292-2239 FAX
brian_f_nash@consecofinance.com