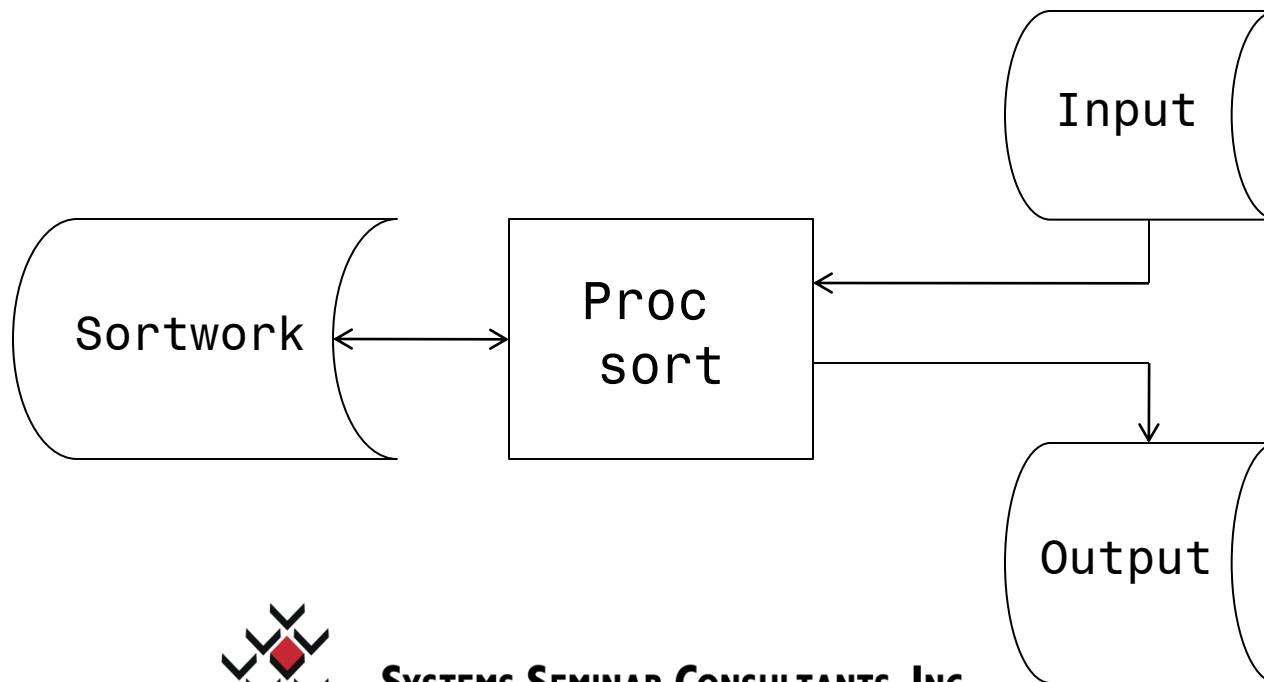




A Different View of PROC SORT



SYSTEMS SEMINAR CONSULTANTS, INC.

2997 Yarmouth Greenway Drive, Madison, WI 53711

Phone: (608) 278-9964 • Web: www.sys-seminar.com



Steven J. First **President**

- Over 30 years of SAS experience, including hundreds of manufacturing, retail, government, marketing, and financial applications.
- Over 25 years as President and Founder of SSC
- Founder of WISAS and WISUG, original MWSUG board member
- Invited speaker at local, regional, and international user groups

A Different View of PROC SORT



This paper was written by Systems Seminar Consultants, Inc. SSC specializes in SAS software and offers:

- Training Services
- Consulting Services
- Help Desk Plans
- Newsletter Subscriptions to *The Missing Semicolon*[™]

COPYRIGHT© 2010 Systems Seminar Consultants, Inc.

All rights reserved. Printed in the United States of America. No part of this publication may be reproduced, stored in a retrieval system, or transmitted, in any form or by any means, electronic, mechanical, photocopying, or otherwise, without prior written permission of SSC. SAS is a registered trademark of SAS Institute Inc. in the USA and other countries. *The Missing Semicolon* is a trademark of Systems Seminar Consultants, Inc.

Free SAS Newsletter



The Missing Semicolon™, shares SAS software solutions developed by our staff and provides additional technical assistance to our customers.

Visit www.sys-seminar.com to sign up for a free subscription:



Sorting Your Data



PROC SORT rearranges the order of the observations in a SAS dataset and replaces the dataset, or creates a new dataset.

Features:

- single or multiple sort variables
- ascending or descending sequence
- does not print any output
- missing values are lowest value to PROC SORT
- records that are exact duplicates or have duplicate keys can be ignored
- After sorting, BY value breaks can occur with formatted groups
- Original order can be preserved if not affected by sort sequence
- A alternate sort sequencing technique can be specified.

Using PROC SORT



Several options and statements can be used with PROC SORT.

Syntax:

PROC SORT *options*;

Options: (partial list)

DATA=	SAS dataset input to the sort
OVERWRITE	SAS deletes the original dataset before output
NODUPKEY	Eliminate rows with duplicate BY values
NODUPRECS	Eliminate consecutive duplicate records
OUT=	Create a new SAS dataset out of the sort

Using PROC SORT (continued)



Statements used with PROC SORT:

BY *<descending> variable-1 <descending> variable-n ;*

Or

Key *variable-1/option;*

Key *variable-2/option;*

Notes:

- KEY statements allow addition of individual options for each variable.

How Does PROC SORT Work?



Start



End

```
proc sort data=softsale;  
  by name;  
run;
```

Partial softsale

Name	Division	Years	Sales	Expense	State
CHRIS	H	2	233.11	94.12	WI
TOM	S	5	5669.12	798.15	MN
BETH	H	12	4822.12	982.10	WI

Sortwork
areas



A PROC SORT Example



Sort the SOFTSALE dataset into alphabetic order.

```
proc sort data=softsale;  
    by name;  
run;
```

```
proc print data=softsale;  
    title 'Softco Sales Summary';  
    footnote 'Sorted Alphabetically';  
run;
```

Notes:

- The original SAS dataset is replaced with the sorted dataset.

The Resulting Output



Softco Sales Summary

Obs	Name	Division	Years	Sales	Expense	State
1	ANDREW	S	24	1762.11	476.13	MN
2	BENJAMIN	S	3	201.11	25.21	IL
3	BETH	H	12	4822.12	982.10	WI
4	CHRIS	H	2	233.11	94.12	WI
5	JANET	S	1	98.11	125.32	WI
6	JENNIFER	S	1	542.11	134.24	IL
7	JOHN	H	7	678.43	150.11	WI
8	JOY	S	12	2442.22	761.98	WI
9	MARK	H	5	298.12	52.65	WI
10	MARY	S	14	5691.78	2452.11	WI
11	PAT	H	4	4009.21	322.12	IL
12	SARAH	S	6	301.21	65.17	MN
13	STEVE	H	21	6153.32	1507.12	WI
14	TOM	S	5	5669.12	798.15	MN
15	WILLIAM	H	11	3231.75	644.55	MN

Sorted Alphabetically

Making a Copy of the Input SAS Dataset



OUT= names a new output dataset and does not alter the input dataset.

```
proc sort data=softsale out=sortsoft;  
  by name;  
run;
```

```
proc print data=sortsoft;  
  title 'Softco Sales Summary';  
  footnote 'Sorted Alphabetically';  
run;
```

The Resulting Output



Softco Sales Summary

Obs	Name	Division	Years	Sales	Expense	State
1	ANDREW	S	24	1762.11	476.13	MN
2	BENJAMIN	S	3	201.11	25.21	IL
3	BETH	H	12	4822.12	982.10	WI
4	CHRIS	H	2	233.11	94.12	WI
5	JANET	S	1	98.11	125.32	WI
6	JENNIFER	S	1	542.11	134.24	IL
7	JOHN	H	7	678.43	150.11	WI
8	JOY	S	12	2442.22	761.98	WI
9	MARK	H	5	298.12	52.65	WI
10	MARY	S	14	5691.78	2452.11	WI
11	PAT	H	4	4009.21	322.12	IL
12	SARAH	S	6	301.21	65.17	MN
13	STEVE	H	21	6153.32	1507.12	WI
14	TOM	S	5	5669.12	798.15	MN
15	WILLIAM	H	11	3231.75	644.55	MN

Sorted Alphabetically

Notes:

- The SOFTSALE dataset is unchanged.

Another PROC SORT Example



Sort into DIVISION order.

```
proc sort data=softsale;  
  by division;  
run;
```

```
proc print data=softsale;  
  title 'Softco Sales Summary';  
  footnote 'Sorted by Division Alone';  
run;
```

The Resulting Output



Softco Sales Summary

Obs	Name	Division	Years	Sales	Expense	State
1	CHRIS	H	2	233.11	94.12	WI
2	MARK	H	5	298.12	52.65	WI
3	PAT	H	4	4009.21	322.12	IL
4	JOHN	H	7	678.43	150.11	WI
5	WILLIAM	H	11	3231.75	644.55	MN
6	STEVE	H	21	6153.32	1507.12	WI
7	BETH	H	12	4822.12	982.10	WI
8	SARAH	S	6	301.21	65.17	MN
9	ANDREW	S	24	1762.11	476.13	MN
10	BENJAMIN	S	3	201.11	25.21	IL
11	JANET	S	1	98.11	125.32	WI
12	JENNIFER	S	1	542.11	134.24	IL
13	JOY	S	12	2442.22	761.98	WI
14	MARY	S	14	5691.78	2452.11	WI
15	TOM	S	5	5669.12	798.15	MN

Sorted by Division Alone

Using OVERWRITE to Save Space



Delete the original dataset before copying the data to the output dataset.

Example:

```
proc sort data=softsale overwrite;  
    by name;  
run;
```

Notes:

- The resulting saving in workspace can be significant.
- Caution: Because the dataset is deleted before the new dataset is written out, data should be temporary, or backed up first.

How Does OVERWRITE Work?



Start



End

```
proc sort data=softsale  
    overwrite;  
    by name;  
run;
```

Partial softsale

Name	Division	Years	Sales	Expense	State
CHRIS	H	2	233.11	94.12	WI
TOM	S	5	5669.12	798.15	MN
BETH	H	12	4822.12	982.10	WI

Sortwork
areas



Multiple Keys and DESCENDING Option



Within each Division, sort highest to lowest Sales.

Example:

```
proc sort data=softsale;  
  by division descending sales;  
run;
```

```
proc print data=softsale;  
  title 'Softco Sales Summary';  
  footnote 'Sorted by Division and Descending Sales';  
run;
```

The Resulting Output



Softco Sales Summary

Obs	Name	Division	Years	Sales	Expense	State
1	STEVE	H	21	6153.32	1507.12	WI
2	BETH	H	12	4822.12	982.10	WI
3	PAT	H	4	4009.21	322.12	IL
4	WILLIAM	H	11	3231.75	644.55	MN
5	JOHN	H	7	678.43	150.11	WI
6	MARK	H	5	298.12	52.65	WI
7	CHRIS	H	2	233.11	94.12	WI
8	MARY	S	14	5691.78	2452.11	WI
9	TOM	S	5	5669.12	798.15	MN
10	JOY	S	12	2442.22	761.98	WI
11	ANDREW	S	24	1762.11	476.13	MN
12	JENNIFER	S	1	542.11	134.24	IL
13	SARAH	S	6	301.21	65.17	MN
14	BENJAMIN	S	3	201.11	25.21	IL
15	JANET	S	1	98.11	125.32	WI

Sorted by Division and Descending Sales

Duplicate Records



Sarah's data was accidentally entered twice.

Softdup Dataset						
Obs	Name	Division	Years	Sales	Expense	State
1	CHRIS	H	2	233.11	94.12	WI
2	MARK	H	5	298.12	52.65	WI
3	SARAH	S	6	301.21	65.17	MN
4	MARY	S	14	5691.78	2452.11	WI
5	PAT	H	4	4009.21	322.12	IL
6	JOHN	H	7	678.43	150.11	WI
7	WILLIAM	H	11	3231.75	644.55	MN
8	ANDREW	S	24	1762.11	476.13	MN
9	BENJAMIN	S	3	201.11	25.21	IL
10	JANET	S	1	98.11	125.32	WI
11	STEVE	H	21	6153.32	1507.12	WI
12	JENNIFER	S	1	542.11	134.24	IL
13	JOY	S	12	2442.22	761.98	WI
14	SARAH	S	6	301.21	65.17	MN
15	TOM	S	5	5669.12	798.15	MN
16	BETH	H	12	4822.12	982.10	WI

Deleting Duplicate Records



Specifying NODUPS deletes exact consecutive duplicate records after sorting.

```
proc sort data=softsale nodups;  
  by name;  
run;
```

Partial Log:

```
NOTE: 1 duplicate observations were deleted.  
NOTE: There were 16 observations read from the data set WORK.SOFTDUP.  
NOTE: The data set WORK.SOFTDUP has 15 observations and 6 variables.
```

The Resulting Output



Softdup Dataset after Nodups

Obs	Name	Division	Years	Sales	Expense	State
1	ANDREW	S	24	1762.11	476.13	MN
2	BENJAMIN	S	3	201.11	25.21	IL
3	BETH	H	12	4822.12	982.10	WI
4	CHRIS	H	2	233.11	94.12	WI
5	JANET	S	1	98.11	125.32	WI
6	JENNIFER	S	1	542.11	134.24	IL
7	JOHN	H	7	678.43	150.11	WI
8	JOY	S	12	2442.22	761.98	WI
9	MARK	H	5	298.12	52.65	WI
10	MARY	S	14	5691.78	2452.11	WI
11	PAT	H	4	4009.21	322.12	IL
12	SARAH	S	6	301.21	65.17	MN
13	STEVE	H	21	6153.32	1507.12	WI
14	TOM	S	5	5669.12	798.15	MN
15	WILLIAM	H	11	3231.75	644.55	MN

Deleting Records With Duplicate Keys



NODUPKEY will output only one record per value of BY variable.

Example: Select a single observation for each state.

```
proc sort data=softsale nodupkey;  
  by state;  
run;
```

```
proc print data=softsale;  
  title 'A Single Record Per State';  
run;
```

Partial Log:

```
NOTE: 12 observations with duplicate key values were deleted.  
NOTE: There were 15 observations read from the data set WORK.SOFTSALE.  
NOTE: The data set WORK.SOFTSALE has 3 observations and 6 variables.
```

The Resulting Output



A Single Record Per State

Obs	Name	Division	Years	Sales	Expense	State
1	PAT	H	4	4009.21	322.12	IL
2	SARAH	S	6	301.21	65.17	MN
3	CHRIS	H	2	233.11	94.12	WI

Notes:

- Under some operating systems you may not always get the first record input per key.

Additional PROC SORT Options



DATECOPY

Preserves dataset create date and time

DUPOUT=

specifies output data set for duplicate observations

EQUALS

for obs with identical BY values, EQUALS

maintains relative order of the obs

NOEQUALS

do not maintain this order for identical BY values

PRESORTED

checks before sorting for being already sorted

REVERSE

sorts character variables in reverse order

SORTSEQ=

specifies the collating sequence as an option, a translation table, an encoding, or LINGUISTIC.

TAGSORT

store only BY variables and obs no while sorting, After sorting, use tags to retrieve records directly.

THREADS |

enable/prevents multi-threaded sorting.

NOTHEADS



- Syncsort is a very fast and popular mainframe sort program.
- Your SAS installer may choose Syncsort to be the program to be used by SAS for sorts, but SAS still does the I/O. You code PROC SORT.
- PROC SYNCSORT is an additional product from Syncsort Inc. that replaces the SAS sort, and SYNCSORT does the I/O (fast). You could use PROC SYNCSORT or sometimes PROC SORT will also work.



SYSTEMS SEMINAR CONSULTANTS, INC.

SAS® Training, Consulting, & Help Desk Services
2997 Yarmouth Greenway Drive • Madison, WI 53711
(608) 278-9964 • Fax (608) 278-0065
www.sys-seminar.com



Steven First
President
sfirst@sys-seminar.com