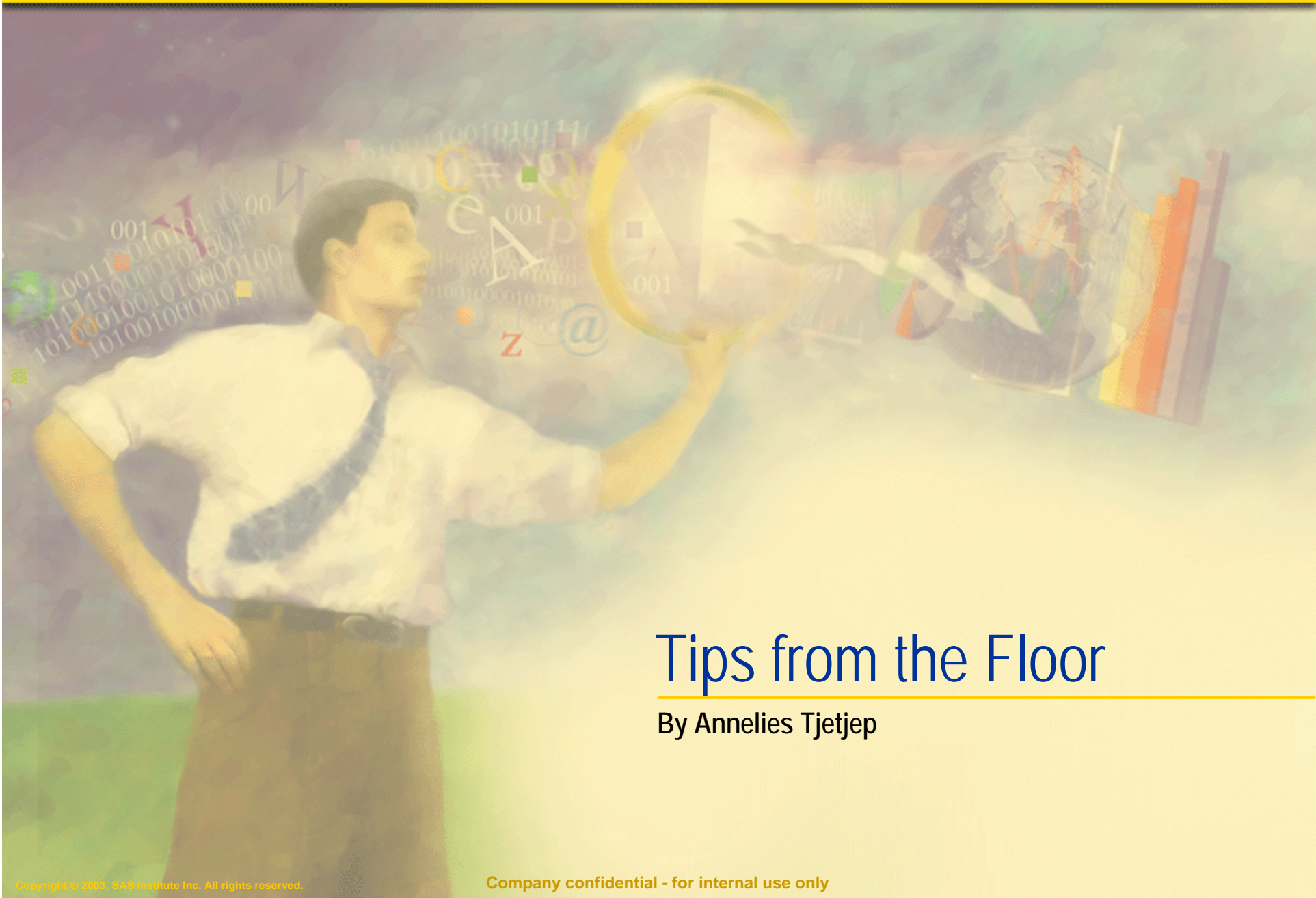




The Power to Know.



Tips from the Floor

By Annelies Tjetjep

Overview

- Features of PROC SORT
- Functions to Watch Out for
- Formats & The Case of the Date/Time Format
- Some EXTRA bits and pieces

PROC SORT

- Ways of improving System Efficiency when Sorting data
- The new DUPOUT= option
- A little look at PROC SQL
- Comments from the Technical Support team

A Couple of Ways to Improve Efficiency

- “Save Time on Sorts”
- **The NOEQUALS option**

```
proc sort data=fred noequals;
```

```
    by var;
```

```
run;
```

- Does not “necessarily preserve” the relative order within the BY groups

For Example:

SASHELP.CLASS is already sorted BY Name

```
proc sort data=sashelp.class out=work.sort;  
by sex;  
run;
```

Output:

Name	Sex	Age	Height	Weight
Alice	F	13	56.5	84
Barbara	F	13	65.3	98
Carol	F	14	62.8	102.5
Jane	F	12	59.8	84.5

```
proc sort data=sashelp.class out=work.sort noequals;  
by sex;  
run;
```

Output:

Name	Sex	Age	Height	Weight
Judy	F	14	64.3	90
Jane	F	12	59.8	84.5
Joyce	F	11	51.3	50.5
Barbara	F	13	65.3	98

Supplied by Stephen Hanks

- “For large Sorts”
- **The TAGSORT option**

```
proc sort data=fred tagsort;
```

```
    by var;
```

```
run;
```

- Temporarily stores only the BY variables and observation numbers (tags) then uses these tags to retrieve input data records in sorted order.

Supplied by Stephen Hanks

A New Feature of PROC SORT

- **The DUPOUT= option**

```
proc sort data=fred nodupkey dupout=dupes;
```

```
by sex;
```

```
run;
```

- Specifies an output data, “dupes”, set that contains duplicate observations
- **Undocumented**: dupout on proc sort does not work unless you also use nodupkey

Supplied by Brian Watts

PROC SQL

- To carry out the same basic sort:

PROC SQL;

```
CREATE TABLE WORK.CLASS AS SELECT CLASS.Name,  
    CLASS.Sex,  
    CLASS.Age,  
    CLASS.Height,  
    CLASS.Weight
```

```
FROM SASHELP.CLASS AS CLASS
```

```
ORDER BY CLASS.Sex;
```

QUIT;

Supplied by Enterprise Guide Query Builder

Comments

- New to SAS9: Multithreaded Sorting

When system option THREADS specified

- An Alternative Way to DUPOUT=

```
data dupes nondupes;
```

```
  set dataset;
```

```
  by var1 var2;
```

```
  if not (first.var2 and last.var2) then output dupes;
```

```
  else output nondupes;
```

```
run;
```

dupes contains the duplicate observations and nondupes contains the “clean” (unduplicated) observations.

Christopher Laurence

Functions

- Using the ROUND function to get Confirmation of “Expected Results”
- %UPCASE versus %LOWCASE
- Simulating a LOGNORMAL function
- FCMP

The Round Function

Binary Arithmetic vs Decimal Arithmetic

* without ROUND function ;

```
data _null_;
```

```
  x=0.1+0.1+0.1;
```

```
  if x = 0.3 then put "=0.3";
```

```
  if x ne 0.3 then put "<>0.3";
```

```
run;
```

```
LOG:
```

```
<>0.3
```

* with ROUND function ;

```
data _null_;
```

```
  x=0.1+0.1+0.1;
```

```
  if ROUND(x,.1) = 0.3 then put "=0.3";
```

```
  if ROUND(x,.1) ne 0.3 then put "<>0.3";
```

```
run;
```

```
LOG:
```

```
=0.3
```

Last Say:

Always useful to test calculation by using the
ROUND Function

Supplied By George W. Zhao

%UPCASE 'n' %LOWCASE

- %UPCASE is a **Macro Function**
 - Can be used as expected without any extraneous steps
- %LOWCASE is an **Autocall Macro**
 - MAUTOSOURCE System Option must be specified
 - AUTOCALL Library must be installed

Supplied by George W. Zhao

Simulating a Lognormal Function

- Use RANNOR function and some Mathematics

Lognormal distribution with mean= m & variance= v
from a Normal distribution with mean= μ &
variance= sig^2

Lognormal = $\exp(\mu + \sqrt{\text{sig}^2} * \text{rannor}(\text{seed}))$

Where $\mu = \ln(\text{msq} / \sqrt{v + \text{msq}})$,
 $\text{sig}^2 = \ln((v + \text{msq}) / \text{msq})$ and $\text{msq} = m^2$

Supplied by Annelies Tjetjep

The SAS Function Compiler (FCMP) Procedure

- Allows users to create, test, and store SAS functions and subroutines for use by certain other SAS procedures:
 - CALIS, COMPILE, DISTANCE, GA, GENMOD, MODEL, NLIN, NLMIXED, NLP, PHREG, RISK DIMENSIONS, ROBUSTREG, SIMILAR, SYLK

Documentation:

<http://support.sas.com/documentation/onlinedoc/base/91/fcmp.pdf>

Supplied By Brian Watts

Formats

Using the Picture Statement

Creates a template for printing numbers.

- Display Negative Numbers in Brackets

```
proc format;
```

```
PICTURE neg
```

```
low-<0 = '000,000.99)' (prefix='(')
```

```
0-high = '000,000.99';
```

```
run;
```

```
data test;  
a=1; b= -2;  
format a b neg.;  
run;
```

```
proc print data=test;  
run;
```

Output:

Obs	a	b
1	1.00	(2.00)

Supplied By Stephen Hanks

An Example of Coding Date/Time Formats/Informats: MMM DD YYYY HH:MM:SS

- **As a FORMAT:**

```
proc format ;
```

```
    PICTURE myfmt
```

```
    low-high='%0d-%0m-%0y %0H:%0M:%0S'  
    (datatype=datetime);
```

```
run;
```

■ **Example:**

```
data one;  
input sasdate ;  
format sasdate myfmt. ;  
put sasdate= ;  
cards ;  
9999999999  
7464646476  
3636363636  
252363  
 ;  
run;
```

■ **Output:**

```
sasdate=09-09-91 01:46:39  
sasdate=17-07-96 08:54:36  
sasdate=25-03-75 13:00:36  
sasdate=03-01-60 22:06:03
```

Supplied by Arif Bhuiyan

■ **As an Informat:**

/*The Date Part*/

proc format ;

PICTURE myfmt

low-high='%0b %0d %0Y '
(datatype=date);

run;

**/*Creating the dataset to be
used for Informats*/**

RHS ->

data infmt ;

retain fmtname "yyyymd" type "I" ;

do label = "01jan1960"d to
"01jan2005"d ;

start = put(label,myfmt.) ;

start = trim (left (start)) ;

output ;

end ;

run ;

proc format cntlin = infmt ;

run ;

■ **Example:**

```
data date;
infile datalines dlm=' ' missover;
input date & $20.
      ;
```

```
sasdate=input(substr(date,1,11), yyyymm.);
sastime=input(substr(date,13,8), time9.);
```

```
datalines;
SEP 09 1991 01:46:39
JUL 17 1996 08:54:36
MAR 25 1975 13:00:36
JAN 03 1960 22:06:03
;
run;
```

```
data datetime;
```

```
set date;
```

```
sasdatetime=dhms(sasdate,0,0,sastime);
```

```
format sasdatetime datetime17.;
```

```
put date= sasdate= sastime= sasdatetime;
```

```
run;
```

Supplied By Arif Bhuiyan

- Alternatively, **functions** can be used:
- **Example:**

```
data date;  
  infile datalines ;  
  input mon $ 1-3 day $ 5-6 year $ 8-11 @13 time time8.;  
  date=input(cats(day,mon,year),date9.);  
  datalines;  
  SEP 09 1991 01:46:39  
  JUL 17 1996 08:54:36  
  MAR 25 1975 13:00:36  
  JAN 03 1960 22:06:03  
  ;  
run;
```

Supplied By Brian Watts

Some Bits & Pieces

- **Numerical calculation on same prefix**

- Alex Tang

```
data work.table;
```

```
input id $ eccben eccfrm eccrsl;
```

```
datalines;
```

```
E101 10 1 4
```

```
E104 57 34 1
```

```
run;
```

```
data work.table_total;
```

```
set work.table;
```

```
/* sum all variables with prefix "ecc" */
```

```
total = sum(of ecc:);
```

```
run;
```

■ Boundary Case with Formats

- **Alex Tang**

proc format;

value test **10<-<20** = "Yeah"

other = "other";

run;

- 10 < - < 20 will be $10 < x < 20$
- 10 < - 20 will be $10 < x \leq 20$
- 10 - 20 will be $10 \leq x \leq 20$
- 10 - < 20 will be $10 \leq x < 20$

- **For Non-Coders**

- **Stephen Hanks**

“Use the provided point and click interfaces to SAS if you’re not used to the coding or syntax, eg the report builder, graph-n-go and in V9 Eguide”

- **Binders on Enterprise Guide**

- **Annelies Tjetjep**

Binders are useful for sharing data across servers on a network. They can contain EG projects, datasets and programs.

Enterprise Guide Administrator > File > New > Binder Icon

End of the Show

Please help yourself to the hors d'oeuvres and sparkling white wine.