STAT604 SAS Lesson 11

Using Custom Formats

The demonstration illustrates permanent data sets that have a custom format applied.

Using the NOFMTERR System Option

By default, the FMTERR system option is in effect. If you use a format that SAS cannot load, SAS issues an error message and stops processing the step.

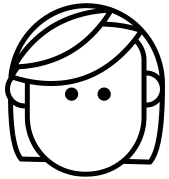
To prevent the default action, change the system option FMTERR to NOFMTERR.

```
OPTIONS FMTERR | NOFMTERR;
```

Location of Custom Formats

```
proc format;
    value $sttype
          'TS'='Tropical Storm'
          'SS'='Subtropical Storm'
          'ET'='Extratropical Storm'
          'DS'='Disturbance'
          'NR'='Not Reported';
run;
```

By default, custom formats are stored in the temporary **Work** library in a catalog named **formats**.



```
NOTE: Format $STTYPE has been output.
```

```
NOTE: Format $STTYPE is already on the library WORK.FORMATS.
NOTE: Format $STTYPE has been output.
```

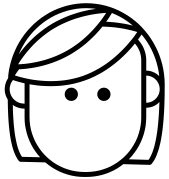
Creating Permanent Custom Formats

```
proc format library=pg2.myfmts;
```

```
proc format library=pg2;
```

```
proc format lib=pg2;
```

Use the LIBRARY= option to save formats in a permanent location.



Using the FMTSEARCH= System Option

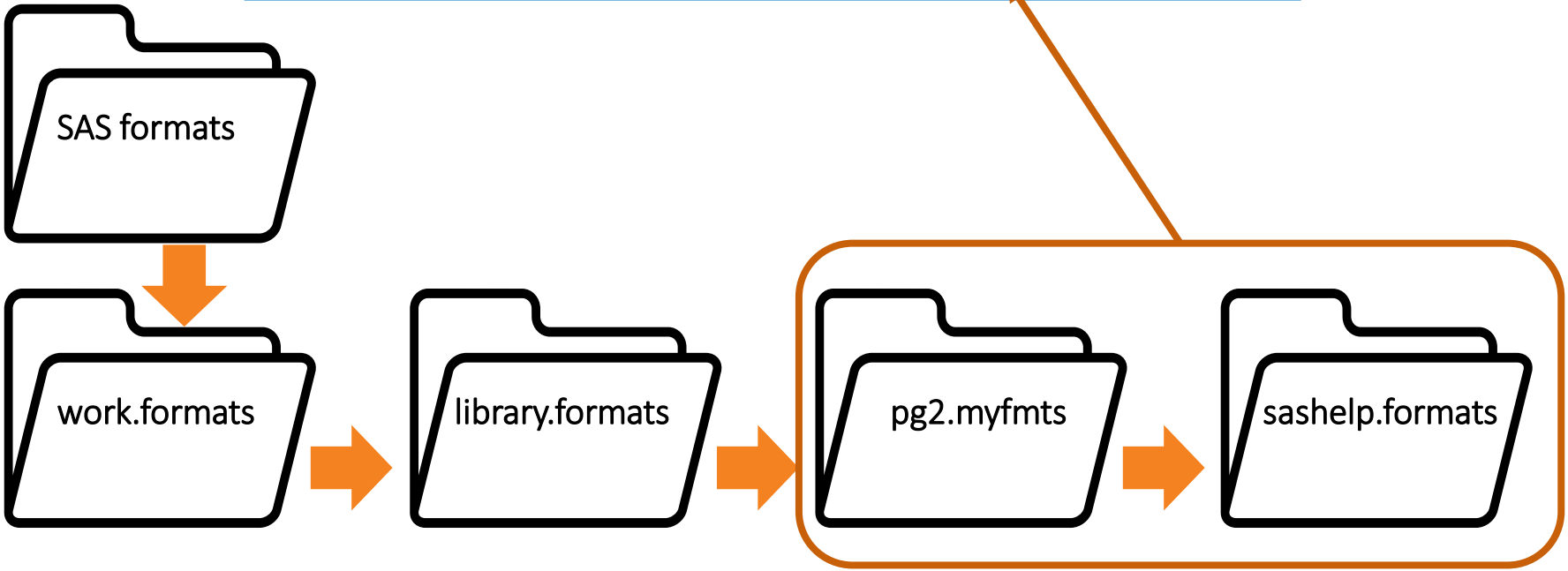
To use permanent formats or to search multiple catalogs, use the FMTSEARCH= system option to identify the catalog(s) to be searched for the format(s).

General form of the FMTSEARCH= system option:

```
OPTIONS FMTSEARCH = (item-1 item-2…item-n);
```

Searching for Custom Formats

```
options fmtsearch=(pg2.myfmts sashelp);
```



Using Permanent Custom Formats

The demonstration illustrates the storage and use of permanent formats.

Documenting Formats

You can use the FMTLIB option in the PROC FORMAT statement to document the format.

General form of the FMTLIB option:

```
PROC FORMAT LIBRARY = libref.catalog
                                FMTLIB;
        <SELECT format-name format-name...;>
        <EXCLUDE format-name format-name...;>
RUN;
```

Documenting Formats

```
proc format library=mylib fmtlib;
        select $col;
run;
```

```
        FORMAT NAME: $COL      LENGTH:    15   NUMBER OF VALUES:    4
   MIN LENGTH:    1  MAX LENGTH:   40  DEFAULT LENGTH:   15  FUZZ:        0
```

START	END	LABEL (VER. V7|V8 01NOV2020:15:59:52)
B	B	Sky Blue
G	G	Rain Cloud Gray
W	W	Moon White
Y	Y	Sunburst Yellow

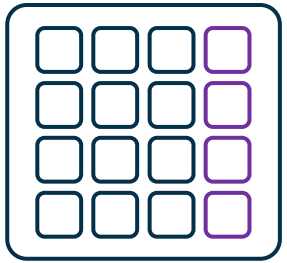
Using Permanent Custom Formats

The demonstration illustrates the documentation of formats.

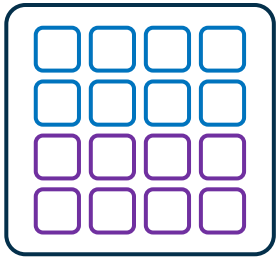
Summarizing Data

Creating an Accumulating Column – Prep Guide Page 148

Lesson Overview

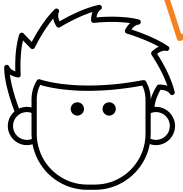


create an accumulating total



process data in groups

Understanding DATA step processing is critical as you perform more complex data manipulations.



Creating an Accumulating Column

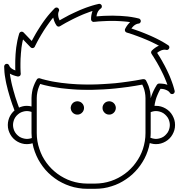
houston_rain

Date	DailyRain	YTDRain
01JAN2017	0.01	0.01
02JAN2017	1.29	1.3
03JAN2017	0	1.3
04JAN2017	0	1.3
05JAN2017	0	1.3
06JAN2017	0.27	1.57
07JAN2017	0	1.57

Create a new column that stores an accumulating total.

```
data houston_rain;
    set pg2.weather_houston;
    keep Date DailyRain YTDRain;
    YTDRain=YTDRain+DailyRain;
run;
```

Will this code produce the desired results? How will SAS process this assignment statement?



Creating an Accumulating Column

This demonstration illustrates using the DATA step debugger in PC SAS to observe how the default behavior of the PDV affects summary expressions.

Retaining Values in the PDV

Variable	Value
Station	USW00012960
Name	HOUSTON INTERCONTI…
Date	02JAN2017
Month	1
DailyRain	1.29
AvgWind	6.26
TempMin	52
TempMax	74
YTDRain	1.3
ERROR	0
_N_	2

To successfully create an accumulating column:

1) Set the initial value to 0.

2) Retain the value each time that the PDV reinitializes.

Retaining Values in the PDV

RETAIN *column <initial-value>*;

```
data houston2017;
    set pg2.weather_houston;
    retain YTDRain 0;
    YTDRain=YTDRain+DailyRain;
run;
```

1) retains the value each time that the PDV reinitializes
2) assigns an initial value

Date	DailyRain	YTDRain
01JUN2017	0.01	0.01
02JUN2017	0.22	0.23
03JUN2017	0.79	1.02
04JUN2017	0.97	1.99
05JUN2017	0.2	2.19
06JUN2017	0.02	2.21
07JUN2017	0	2.21

Activity

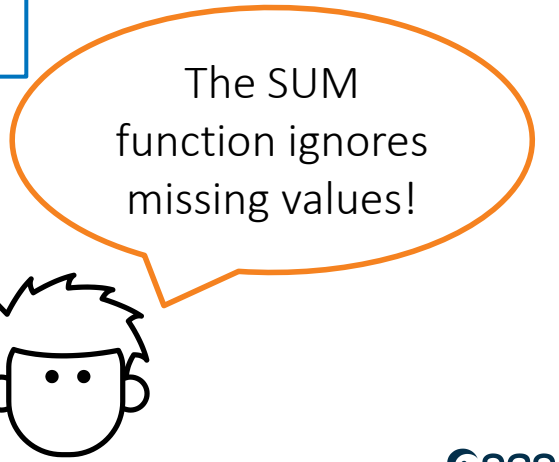
Return to the SAS program used in the demo and perform the following tasks:

1. Modify the program to retain **YTD_Fees** and set the initial value to 0.
2. Run the program and examine the results. Run a similar program using the **amounts** data set?
3. Change the assignment statement to use the SUM function instead of the plus symbol. Run the program again. Why are the results different?

Creating an Accumulating Column

```
data zurich2017;
    set pg2.weather_zurich;
    retain TotalRain 0;
    TotalRain=sum(TotalRain,Rain_mm);
run;
```

Date	Rain_mm	TotalRain
02JAN2017	.	0
03JAN2017	4.8	4.8
04JAN2017	39.9	44.7
05JAN2017	1.8	46.5
06JAN2017	4.3	50.8
07JAN2017	0	50.8



Using the Sum Statement

shortcut to what we just saw.

column+*expression*;

```
data zurich2017;
    set pg2.weather_zurich;
    TotalRain+Rain_mm;
run;
```

- creates **TotalRain** and sets the initial value to 0
- retains **TotalRain**
- adds **Rain_mm** to **TotalRain** for each row
- ignores missing values

Question

What sum statement would you add to this program to create the column named **DayNum**?

```
data zurich2017;
    set pg2.weather_zurich;
    YTDRain_mm+Rain_mm;
    ???
run;
```

PRECIP_MM	YTDPrecip_mm	DayNum
.	0	1
4.8	4.8	2
39.9	44.7	3
1.8	46.5	4
4.3	50.8	5
0	50.8	6

Correct Answer

What sum statement would you add to this program to create the column named **DayNum**?

```
data zurich2017;
    set pg2.weather_zurich;
    YTDRain_mm+Rain_mm;
    DayNum+1;
run;
```

PRECIP_MM	YTDPrecip_mm	DayNum
.	0	1
4.8	4.8	2
39.9	44.7	3
1.8	46.5	4
4.3	50.8	5
0	50.8	6

Activity

Return to the SAS program used in the demo and replace the previously used accumulating code with a single statement.

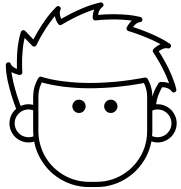
Summarizing Data

Processing Data in Groups – Prep Guide Chapter 8

Processing Data in Groups

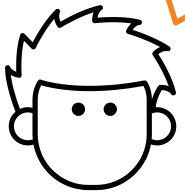
Basin	Name	MaxWindMPH	StartDate
NA	NATE	90	04OCT2017
NA	OPHELIA	115	09OCT2017
NA	PHILIPPE	60	28OCT2017
NA	RINA	60	06NOV2017
NI	MAARUTHA	45	15APR2017
NI	MORA	70	28MAY2017
NI	OCKHI	100	29NOV2017
SI	ALFRED	50	16FEB2017
SI	BLANCHE	65	02MAR2017
SI	CALEB	50	23MAR2017
SI	ERNIE	140	05APR2017
SI	FRANCES	75	21APR2017
SI	GREG	40	29APR2017
SI	CEMPAKA	40	22NOV2017
SI	DAHLIA	60	24NOV2017
SI	HILDA	60	24DEC2017
SP	DEBBIE	120	23MAR2017
SP	BART	45	19FEB2017
SP	COOK	100	06APR2017
SP	DONNA	125	01MAY2017
SP	ELLA	70	07MAY2017

If your data is sorted into groups, the DATA step can identify when each group begins and ends.



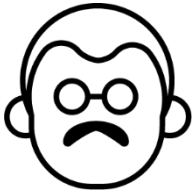
Processing Data in Groups

What is the maximum wind measurement for each storm?



When did the first storm occur in each basin?

Which storm names are used more than once within a season?



Processing Data in Groups

```
PROC SORT DATA=input-table
                <OUT=sorted-output-table>;
      BY <DESCENDING> col-name(s);
RUN;
```

sorts the table into groups

```
DATA output-table;
      SET sorted-output-table;
      BY <DESCENDING> col-name(s);
RUN;
```

processes the data in the sorted table by groups

Processing Data in Groups

```
data storm2017_max;
    set storm2017_sort;
    by Basin;
run;
```

First.*bycol*

Last.*bycol*

The BY statement creates **First./Last.** variables in the PDV that can be used to identify when each BY group begins and ends.

PDV

...other columns...	Basin	First.Basin	Last.Basin
		D	D

Processing Data in Groups

PDV

...other columns...	Basin	First.Basin	Last.Basin
	NA	1	0

first row where **Basin** is *NA*

PDV

...other columns...	Basin	First.Basin	Last.Basin
	NA	0	0

subsequent rows where **Basin** is *NA*

PDV

...other columns...	Basin	First.Basin	Last.Basin
	NA	0	1

last row where **Basin** is *NA*

Identifying the First and Last Row in Each Group

This demonstration illustrates using the DATA step debugger in PC SAS to observe how **First./Last.** variables are assigned values in the PDV during execution.

Short Answer Poll

What are the values for **First.Dept** and **Last.Dept** when the DATA step is processing the observation indicated
by the arrow?

Dept	Salary
ADMIN	20000
ADMIN	100000
ADMIN	50000
ENGINR	25000
FINANC	10000
FINANC	12000

First.Dept
?

Last.Dept
?

Short Answer Poll – Correct Answer

What are the values for **First.Dept** and **Last.Dept** when the DATA step is processing the observation indicated
by the arrow?

Dept	Salary
ADMIN	20000
ADMIN	100000
ADMIN	50000
ENGINR	25000
FINANC	10000
FINANC	12000

First.Dept
1

Last.Dept
1

First.Dept and Last.Dept are both 1. This happens when a group consists of a single observation.

Quiz

Suppose we only one to keep the last row from each group. Can the where statement be successfully added to this data step?

```
data storm2017_max;
    set storm2017_sort;
    by Basin;
    where last.Basin=1;
run;
```

Quiz – Correct Answer

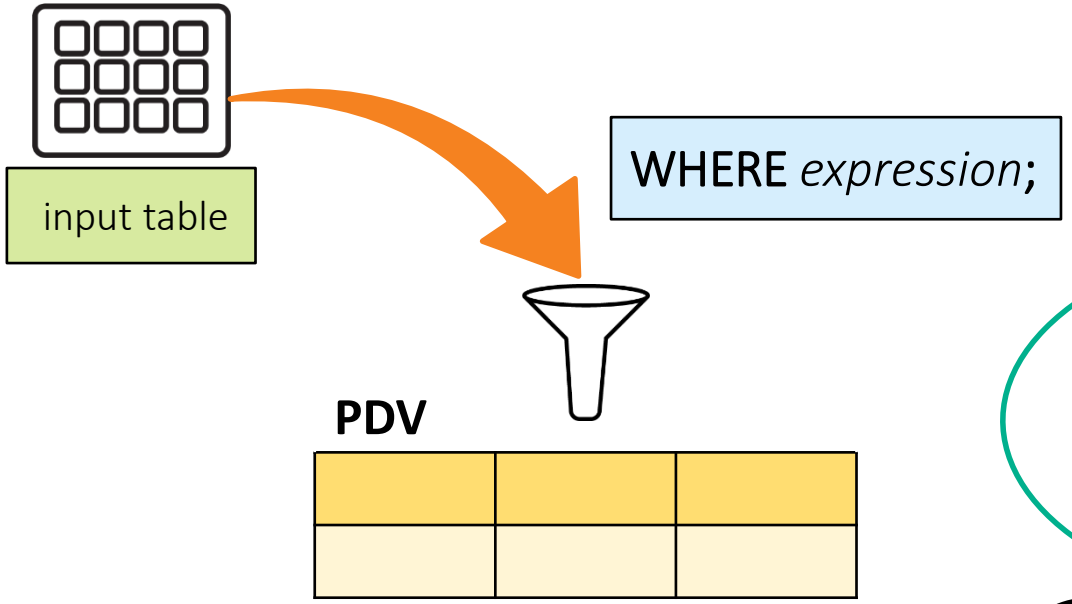
Can the where statement be successfully added to this data step?

```
data storm2017_max;
    set storm2017_sort;
    by Basin;
    where last.Basin=1;
run;
```

No, an error is generated in the log.

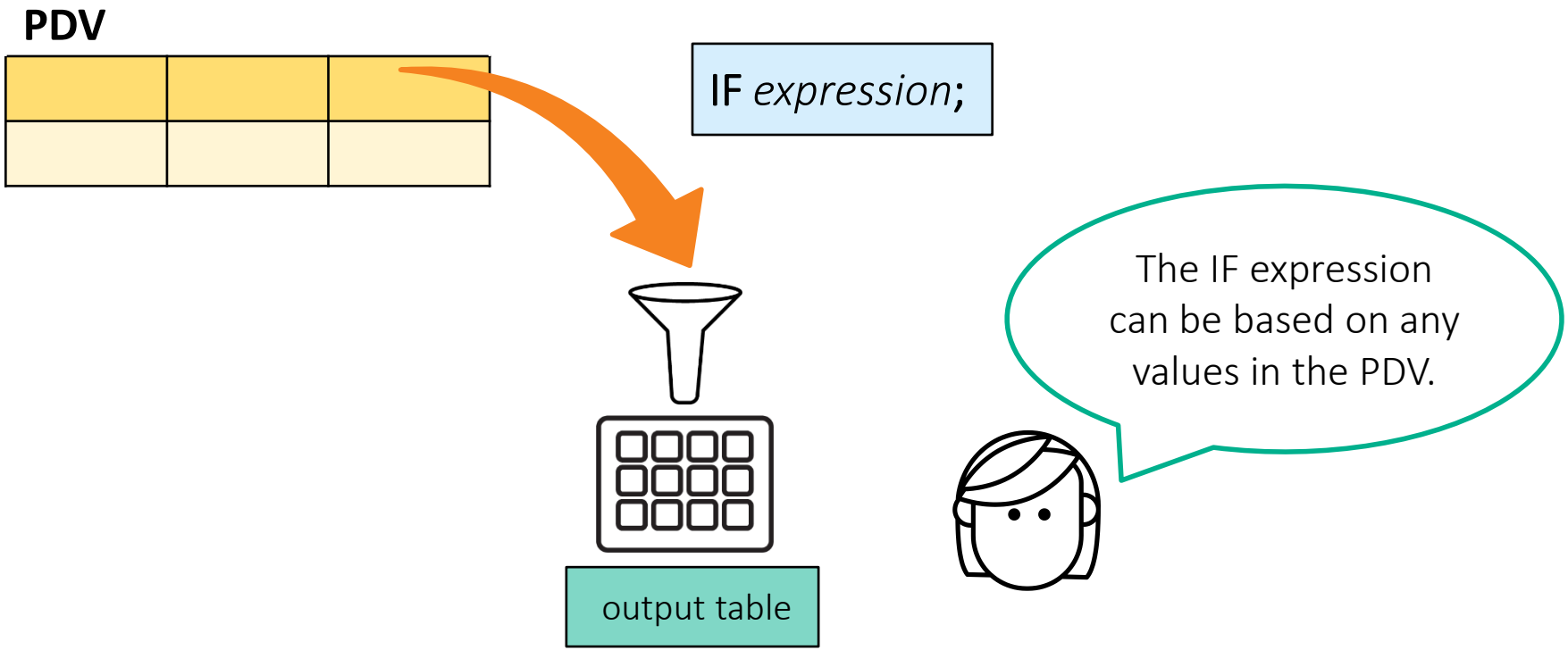
```
32                  where last.Basin=1;
                    __________
                    180
ERROR: Syntax error while parsing WHERE clause.
ERROR 180-322: Statement is not valid or it is used out of proper order.
```

Subsetting Rows in Execution (Review)



The WHERE expression must be based on columns in the input table.

Subsetting Rows in Execution (Review)



Subsetting Rows in Execution

```
data storm2017_max;
    set storm2017_sort;
    by Basin;
    if last.Basin=1;
    StormLength=EndDate-StartDate;
    MaxWindKM=MaxWindMPH*1.60934;
run;
```

true

When the IF condition is true, SAS continues processing statements for that row.

Implicit OUTPUT;
Implicit RETURN;

Subsetting in Execution

```
data storm2017_max;
    set storm2017_sort;
    by Basin;
    if last.Basin=1;
    StormLength=EndDate-StartDate;
    MaxWindKM=MaxWindMPH*1.60934;
run;
```

Implicit RETURN;

false

Implicit OUTPUT;

When the IF condition is false, SAS stops processing statements for that row and returns to the top of the DATA step.

Quiz

What difference does it make if the IF statement were moved to the bottom of the DATA step?

```
data storm2017_max;
    set storm2017_sort;
    by Basin;
    StormLength=EndDate-StartDate;
    MaxWindKM=MaxWindMPH*1.60934;
    if last.Basin=1;
run;
```

Quiz – Correct Answer

Both places produce the same output. However, consider the sequence of the statements in the execution phase. Where is the optimal placement of the subsetting IF statement?

```
data storm2017_max;
     set storm2017_sort;
     by Basin;
     if last.Basin=1;
     StormLength=EndDate-StartDate;
     MaxWindKM=MaxWindMPH*1.60934;
run;
```

Use the subsetting IF statement as early as possible so that SAS processes additional statements only for rows that will be written to the output table.

Keeping only one observation per group

This demonstration illustrates using the **Last.** variable to control output.

What Must Happen When?

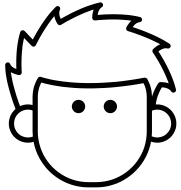
There is a three-step process for using the DATA step to summarize grouped data.

Step 1	Initialize: Set the accumulating variable to zero at the start of each BY group.
Step 2	Accumulate: Increment the accumulating variable with a sum statement (automatically retains).
Step 3	Output: Write only the last observation of each BY group.

Accumulating Column within Groups

Date	Month	DailyRain	MTDRain
01MAR2017	3	0	0
02MAR2017	3	0	0
03MAR2017	3	0	0
04MAR2017	3	0.05	0.05
05MAR2017	3	0.99	1.04
06MAR2017	3	0.24	1.28
24MAR2017	3	1.26	3.93
25MAR2017	3	0.02	3.95
26MAR2017	3	0	3.95
27MAR2017	3	0	3.95
28MAR2017	3	0	3.95
29MAR2017	3	1.68	5.63
30MAR2017	3	0	5.63
31MAR2017	3	0	5.63
01APR2017	4	0	0
02APR2017	4	0.09	0.09

How can we reset an accumulating column at the beginning of each group?



Summarizing Data by Groups

Step 1 Initialize

```
data deptsals(keep=Dept DeptSal);
   set SalSort;
   by Dept;
   if First.Dept then DeptSal=0;
   <additional SAS statements>
run;
```

The condition is considered true when **First.Dept** has a value of *1*.

Summarizing Data by Groups

Step 2 Accumulate

```
data deptsals(keep=Dept DeptSal);
   set SalSort;
   by Dept;
   if First.Dept then DeptSal=0;
   DeptSal+Salary;
   <additional SAS statements>
run;
```

Summarizing Data by Groups

Step 3 Output

Dept	Salary	DeptSal
ADMIN	20000	20000
ADMIN	100000	120000
ADMIN	50000	170000
ENGINR	25000	25000
ENGINR	20000	45000
ENGINR	23000	68000
ENGINR	27000	95000
FINANC	10000	10000
FINANC	12000	22000

Summarizing Data by Groups

Step 3 Output

```
data deptsals(keep=Dept DeptSal);
   set SalSort;
   by Dept;
   if First.Dept then DeptSal=0;
   DeptSal+Salary;
   if Last.Dept;
run;
```

Summarizing Data by Groups

```
proc print data=deptsals noobs;
run;
```

PROC PRINT Output

Dept	DeptSal
ADMIN	410000
ENGINR	163000
FINANC	318000
HUMRES	181000
SALES	373000

Partial SAS Log

```
NOTE: There were 39 observations read
      from the data set WORK.SALSORT.
NOTE: The data set WORK.DEPTSALS has 5
      observations and 2 variables.
```

Creating an Accumulating Column within Groups

This demonstration illustrates using the **First.** variable to reset an accumulating column.

Identifying and Removing Duplicate Rows

```
PROC SORT DATA=input-table <OUT=output-table>
                NODUPRECS <DUPOUT=output-table>;
      BY _ALL_;
RUN;
```

sorts by all columns to ensure that duplicate rows are adjacent

removes adjacent rows that are entirely duplicated

output table of duplicates

Identifying and Removing Duplicate Rows

```
proc sort data=pg1.class_test3
          out=test_clean
          noduprecs
          dupout=test_dups;
    by _all_;
run;
```

pg1.class_test3

Name	Subject	Test Score
Judy	Math	97
Judy	Reading	91
Barbara	Math	96
Barbara	Reading	86
Barbara	Math	96
Louise	Math	92

test_clean

Name	Subject	Test Score
Alice	Math	71
Alice	Reading	67
Barbara	Math	96
Barbara	Reading	86
Carol	Math	61
Carol	Reading	57

test_dups

Name	Subject	Test Score
Barbara	Math	96

Identifying and Removing Duplicate Key Values

```
PROC SORT DATA=input-table <OUT=output-table>
                NODUPKEY <DUPOUT=output-table>;
        BY <DESCENDING> col-name(s);
RUN;
```

keeps only the first occurrence of each unique value of the BY variable

This removes duplicate values of the column listed in the BY statement.

Identifying and Removing Duplicate Key Values

```
proc sort data=pg1.class_test2
          out=test_clean
          dupout=test_dups
          nodupkey;
    by Name;
run;
```

pg1.class_test2

Name	Subject	Test Score
Judy	Math	97
Judy	Reading	91
Barbara	Math	96
Barbara	Reading	86
Louise	Math	92
Louise	Reading	99
James	Math	90
James	Reading	85

test_clean

Name	Subject	Test Score
Alfred	Math	82
Alice	Math	71
Barbara	Math	96
Carol	Math	61
Henry	Math	85
James	Math	90
Jane	Math	84
Janet	Math	75

test_dups

Name	Subject	Test Score
Alfred	Reading	79
Alice	Reading	67
Barbara	Reading	86
Carol	Reading	57
Henry	Reading	86
James	Reading	85
Jane	Reading	76
Janet	Reading	71

Identifying and Removing Duplicate Values

This demonstration compares using the NODUPRECS and NODUPKEY options in PROC SORT with First. and Last. to identify and remove duplicates.

Lesson Quiz

1. Which of the following contains valid syntax?

a.
```
value grades 'A'='Excellent'
             'B'='Good';
```

b.
```
value qtrfmt 1,2,3='First'
             4,5,6='Second';
```

c.
```
value $grades. 'A'='Excellent'
               'B'='Good';
```

d.
```
value qtrfmt '1'-'3'='First'
             '4'-'6'='Second';
```

1. Which of the following contains valid syntax?

a.
```
value grades 'A'='Excellent'
             'B'='Good';
```

b.
```
value qtrfmt 1,2,3='First'
             4,5,6='Second';
```

c.
```
value $grades. 'A'='Excellent'
               'B'='Good';
```

d.
```
value qtrfmt '1'-'3'='First'
             '4'-'6'='Second';
```

2. What is the formatted value for the value of 100 given the following step?

```
proc format;
    value rates   1-<100='low'
                100<-<200='medium'
                 200-300 ='high'
                   other ='out of range';
run;
```

a. *low*

b. *medium*

c. *high*

d. *out of range*

2. What is the formatted value for the value of 100 given the following step?

```
proc format;
    value rates   1-<100='low'
                100<-<200='medium'
                 200-300 ='high'
                   other ='out of range';
run;
```

a. *low*

b. *medium*

c. *high*

d. *out of range*

3. In the FORMAT procedure, you specify the name of the format and the name of the column that will use the custom format.

a. True

b. False

3. In the FORMAT procedure, you specify the name of the format and the name of the column that will use the custom format.

a. True

b. False

5. Which one of the following does not have proper syntax for specifying a range in the VALUE statement?

a. **`500>-700`**

b. **`500-<700`**

c. **`'A'-'C'`**

d. **`'Horse'-'Mouse'`**

5. Which one of the following does not have proper syntax for specifying a range in the VALUE statement?

a. `500>-700`

b. `500-<700`

c. `'A'-'C'`

d. `'Horse'-'Mouse'`

8. Which option in the PROC FORMAT statement specifies a library to store a custom format?

a. CATALOG=
b. FMTLIB=
c. LIBRARY=
d. STORE=

8. Which option in the PROC FORMAT statement specifies a library to store a custom format?

a. CATALOG=

b. FMTLIB=

c. LIBRARY=

d. STORE=

9. What is the default search order that is used to locate formats?

a. LIBRARY.FORMATS ⇨ WORK.FORMATS
b. SASHELP.FORMATS ⇨ LIBRARY.FORMATS
c. SASHELP.FORMATS ⇨ WORK.FORMATS
d. WORK.FORMATS ⇨ LIBRARY.FORMATS

9. What is the default search order that is used to locate formats?

a. LIBRARY.FORMATS ⇨ WORK.FORMATS
b. SASHELP.FORMATS ⇨ LIBRARY.FORMATS
c. SASHELP.FORMATS ⇨ WORK.FORMATS
d. WORK.FORMATS ⇨ LIBRARY.FORMATS

10. Which of the following contains valid syntax for the FMTSEARCH= option?

a. `options fmtsearch=sashelp;`

b. `options fmtsearch=sashelp.formats;`

c. `options fmtsearch=(sashelp sashelp.fmts);`

d. `options fmtsearch=[sashelp.fmts sashelp];`

10. Which of the following contains valid syntax for the FMTSEARCH= option?

a. `options fmtsearch=sashelp;`

b. `options fmtsearch=sashelp.formats;`

c. `options fmtsearch=(sashelp sashelp.fmts);`

d. `options fmtsearch=[sashelp.fmts sashelp];`

1. Which statement contains valid syntax for the RETAIN statement?

a. `retain year 2018;`

b. `retain year*2018;`

c. `retain year=2018;`

d. `retain year{2018};`

1. Which statement contains valid syntax for the RETAIN statement?

a. `retain year 2018;`

b. `retain year*2018;`

c. `retain year=2018;`

d. `retain year{2018};`

2. Which statement is false concerning the sum statement?

a. The sum statement ignores missing values.
b. The sum statement initially sets the accumulator column to missing.
c. The sum statements adds a numeric value to an accumulator column.
d. The sum statement automatically retains the value of the accumulating column.

2. Which statement is false concerning the sum statement?

a. The sum statement ignores missing values.
b. The sum statement initially sets the accumulator column to missing.
c. The sum statements adds a numeric value to an accumulator column.
d. The sum statement automatically retains the value of the accumulating column.

3. What is the value of **Count** at the end of the third DATA step iteration?

work.nums

Tens
10
20
30
40

```
data newnums;
    set nums;
    retain Count 100;
    Count+Tens;
run;
```

a. . (missing)

b. 60

c. 130

d. 160

3. What is the value of **Count** at the end of the third DATA step iteration?

work.nums

Tens
10
20
30
40

```
data newnums;
    set nums;
    retain Count 100;
    Count+Tens;
run;
```

a. . (missing)

b. 60

c. 130

d. 160

4. What is the value of **Count** at the end of the third DATA step iteration?

work.nums

Tens
10
20
.
40

```
data newnums;
    set nums;
    Count+Tens;
run;
```

a. . (missing)

b. 20

c. 30

d. 70

4. What is the value of **Count** at the end of the third DATA step iteration?

work.nums

Tens
10
20
.
40

```
data newnums;
    set nums;
    Count+Tens;
run;
```

a. . (missing)

b. 20

c. 30

d. 70

5. Which step executes successfully without an error, given the input table sashelp.class?

Name	Sex	Age	Height	Weight
Alfred	M	14	69	112.5
Alice	F	13	56.5	84

a.
```
data new;
   set sashelp.class;
   Ratio=Height/Weight;
   where Sex='M' & Ratio>0.6;
run;
```

b.
```
data new;
   set sashelp.class;
   where Sex='M';
   Ratio=Height/Weight;
   where Ratio>0.6;
run;
```

c.
```
data new;
   set sashelp.class;
   where Sex='M’;
   Ratio=Height/Weight;
   if Ratio>0.6;
run;
```

d.
```
data new;
   set sashelp.class;
   if Sex='M’;
   Ratio=Height/Weight;
   where Ratio>0.6;
run;
```

5. Which step executes successfully without an error, given the input table sashelp.class?

Name	Sex	Age	Height	Weight
Alfred	M	14	69	112.5
Alice	F	13	56.5	84

a.
```
data new;
   set sashelp.class;
   Ratio=Height/Weight;
   where Sex='M' & Ratio>0.6;
run;
```

b.
```
data new;
   set sashelp.class;
   where Sex='M';
   Ratio=Height/Weight;
   where Ratio>0.6;
run;
```

c.
```
data new;
   set sashelp.class;
   where Sex='M';
   Ratio=Height/Weight;
   if Ratio>0.6;
run;
```

d.
```
data new;
   set sashelp.class;
   if Sex='M';
   Ratio=Height/Weight;
   where Ratio>0.6;
run;
```

6. Which statement is true given the following program?

```
data work.student;
    set sashelp.class;
    by Name;
run;
```

a. The PDV contains a temporary variable named **First.Name**.

b. The output table **work.student** contains a column named **Last.Name**.

c. The DATA step sorts the input table by the column **Name**.

d. An error is produced because the BY statement is not permitted in the DATA step.

6. Which statement is true given the following program?

```
data work.student;
    set sashelp.class;
    by Name;
run;
```

a. The PDV contains a temporary variable named **First.Name**.

b. The output table **work.student** contains a column named **Last.Name**.

c. The DATA step sorts the input table by the column **Name**.

d. An error is produced because the BY statement is not permitted in the DATA step.

7. What are the correct values for **First.Name** and **Last.Name** if the value of **Name** appears only once in the input table?

```
data work.student;
    set sashelp.class;
    by Name;
run;
```

a. **First.Name**=0 and **Last.Name**=0

b. **First.Name**=1 and **Last.Name**=1

c. **First.Name**=1 and **Last.Name**=0

d. **First.Name**=0 and **Last.Name**=1

7. What are the correct values for **First.Name** and **Last.Name** if the value of **Name** appears only once in the input table?

```
data work.student;
    set sashelp.class;
    by Name;
run;
```

a. **First.Name**=0 and **Last.Name**=0

b. **First.Name**=1 and **Last.Name**=1

c. **First.Name**=1 and **Last.Name**=0

d. **First.Name**=0 and **Last.Name**=1

8. Which DATA step statement indicates to continue processing the last row of a BY group?

a. `if First.JobTitle;`

b. `if Last.JobTitle;`

c. `where First.JobTitle=1;`

d. `where Last.JobTitle=1;`

8. Which DATA step statement indicates to continue processing the last row of a BY group?

a. `if First.JobTitle;`

b. `if Last.JobTitle;`

c. `where First.JobTitle=1;`

d. `where Last.JobTitle=1;`

9. Which statement needs to be added to the DATA step to reset the value of **TotalWt** for each new BY group?

```
data GenderWeight;
    set Students;
    by Gender;
    … add statement here …
    TotalWt+Weight;
    if Last.Gender=1 then output;
run;
```

a. `TotalWt=0;`

b. `if Last.Gender=0 then TotalWt=0;`

c. `if First.Gender=0 then TotalWt=0;`

d. `if First.Gender=1 then TotalWt=0;`

9. Which statement needs to be added to the DATA step to reset the value of **TotalWt** for each new BY group?

```
data GenderWeight;
    set Students;
    by Gender;
    … add statement here …
    TotalWt+Weight;
    if Last.Gender=1 then output;
run;
```

a. `TotalWt=0;`

b. `if Last.Gender=0 then TotalWt=0;`

c. `if First.Gender=0 then TotalWt=0;`

d. `if First.Gender=1 then TotalWt=0;`