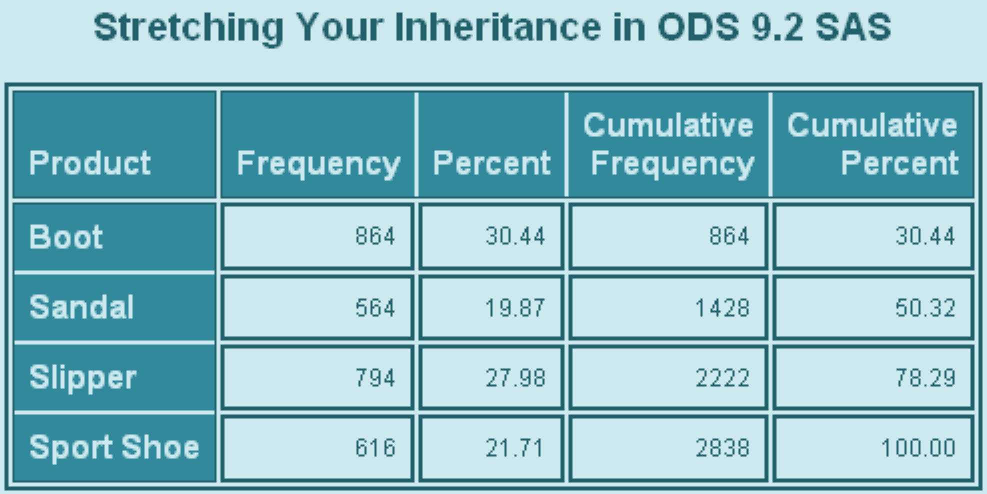
Add Style to ODS Output by Stretching Your Inheritance in Version 9.2 SAS®

Perry Watts, Stakana Analytics, Elkins Park, PA



Stretching Your Inheritance in ODS 9.2 SAS

Product	Frequency	Percent	Cumulative Frequency	Cumulative Percent
Boot	864	30.44	864	30.44
Sandal	564	19.87	1428	50.32
Slipper	794	27.98	2222	78.29
Sport Shoe	616	21.71	2838	100.00

Agenda

I. Start with the primary tool: a ***Lineage Tracer*** for ODS style element definitions in 9.2 SAS
   a) Define Terms: ***inheritance, lineage*** and ***tree***
   b) Use the definitions to develop the **Home Page** of the Lineage Tracer
   c) Connect **Lineage** in the Home Page to **Attribute**: the **"what"** of inheritance via **drill-down**

II. Use the Lineage Tracer to Define new Styles in ODS
   a) With the ***Class*** Statement
   b) With ***Style*** without ***From***
   c) With ***Style ... From***

III. How Styles are used in PROCS PRINT and REPORT (MEANS and FREQ are in the paper).

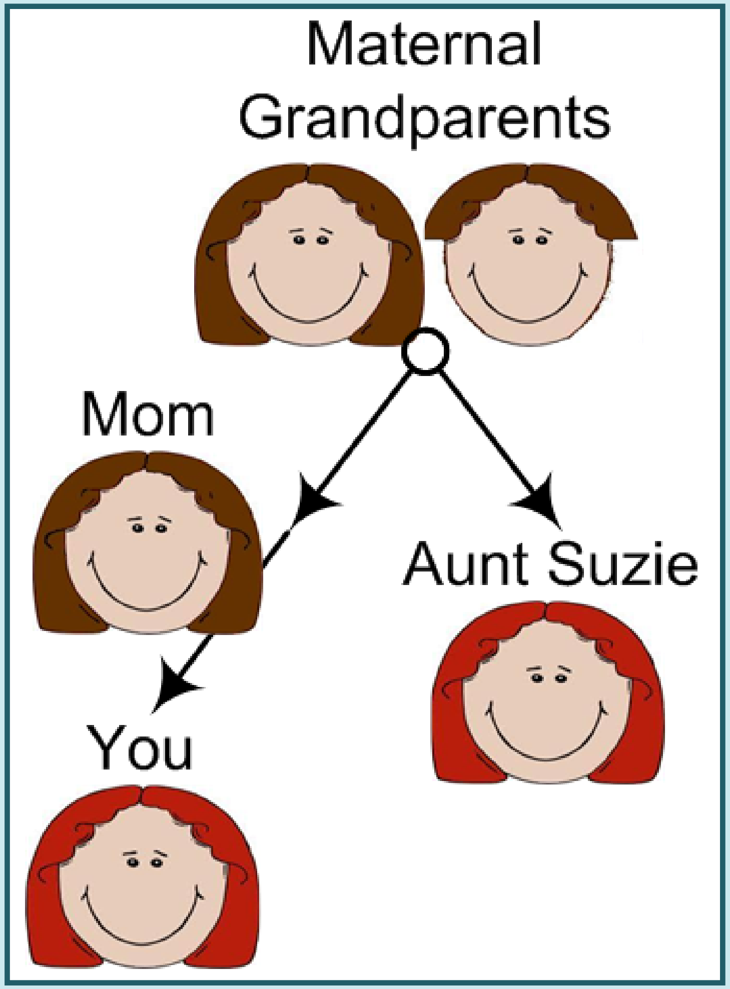
Define Terms: Inheritance, Lineage, Tree

- **Inheritance**: a thing that is inherited.
- **Inherit:** to derive (a quality or characteristic) genetically from one's ancestors.
- **Lineage**: lineal descent; ancestry, pedigree.
- **Lineal:** in the direct line of descent or ancestry; linear.
- **Family tree:** charts relationships and line**s** of descent.

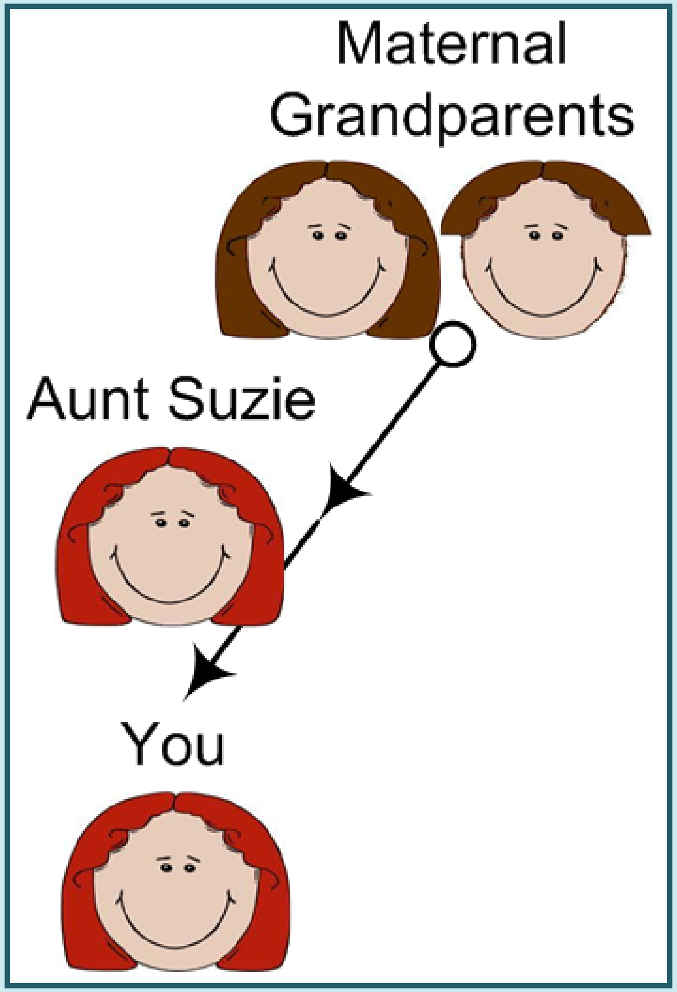
From The Concise Oxford Dictionary

From the above, a family tree is a collection of lineages that trace lines of descent. *Inheritance is confined to the lineage.*

Define Terms: Inheritance, Lineage, Tree



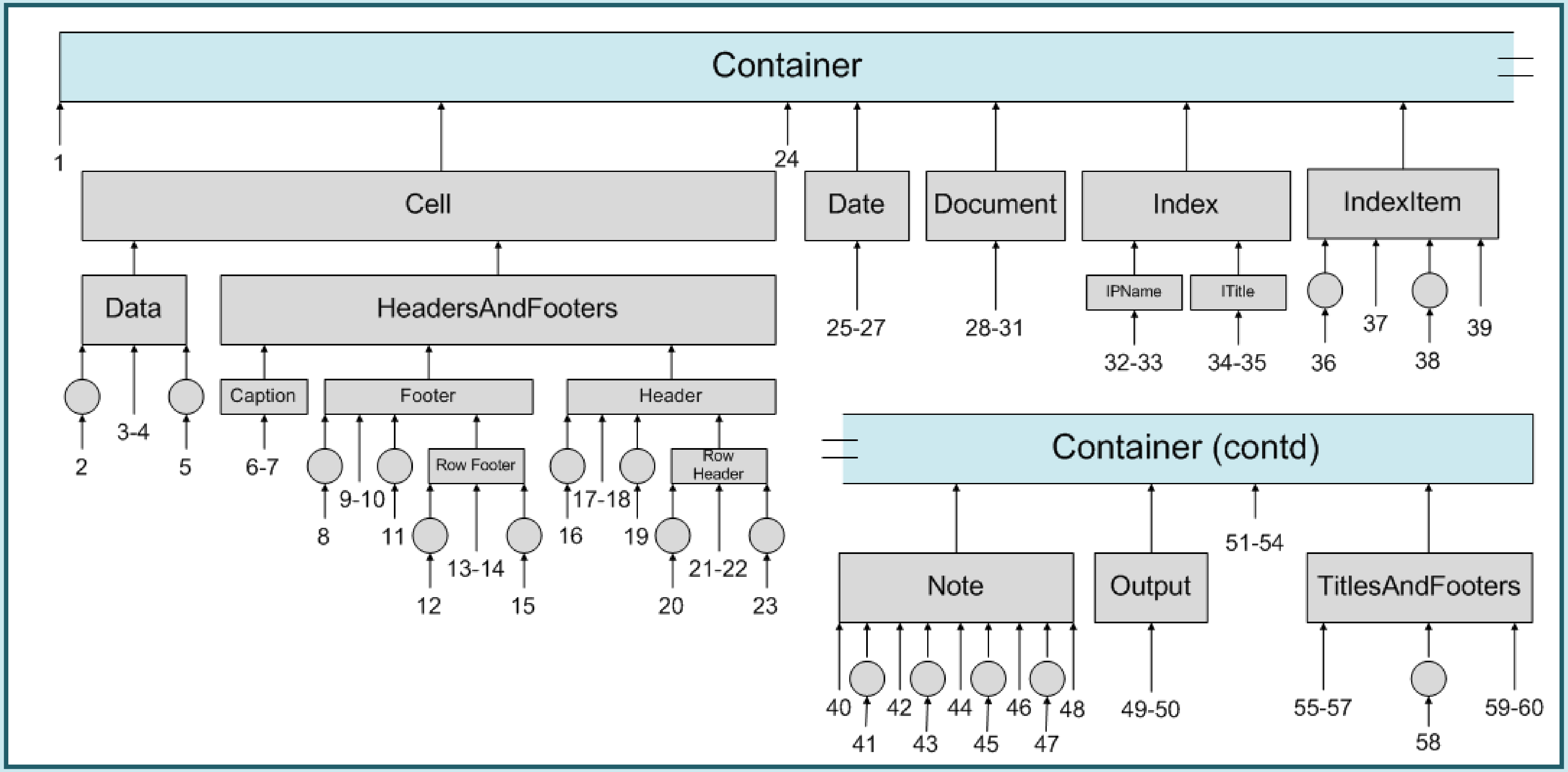
Human



As If ODS

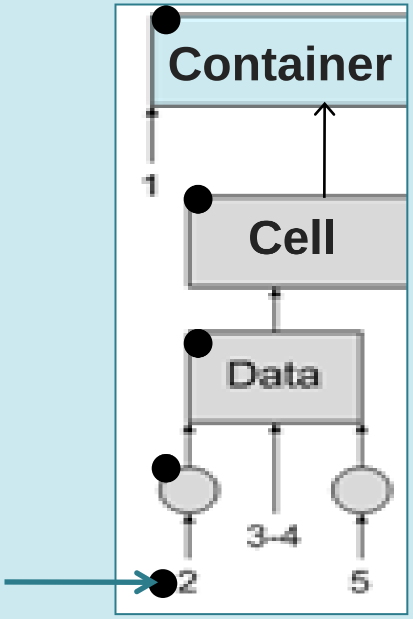
Public Domain Cartoon downloaded from http://www.clker.com/clipart-12187.html

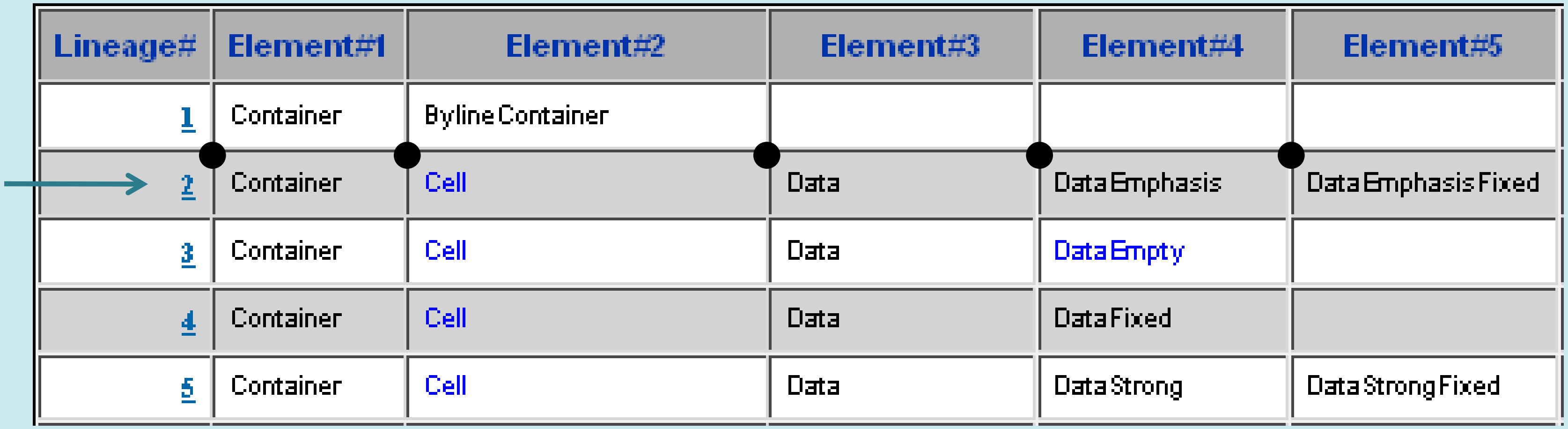
Container Lineages in a Conventional Family Tree



Container Lineages from the Styles.Default Template for 9.1.3 SAS

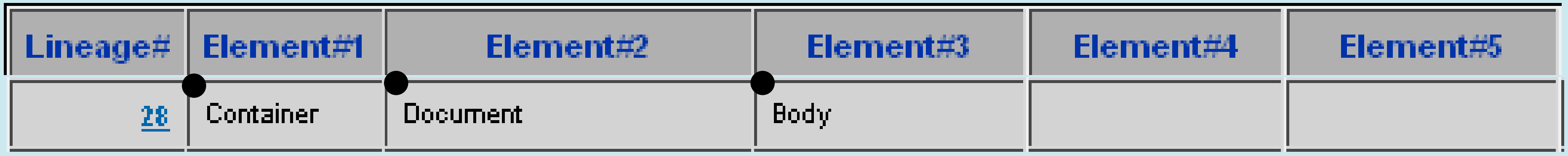
The Home Page of the Lineage Tracer Rotated 90˚





Lineage#	Element#1	Element#2	Element#3	Element#4	Element#5
1	Container	Byline Container			
2	Container	Cell	Data	Data Emphasis	Data Emphasis Fixed
3	Container	Cell	Data	Data Empty	
4	Container	Cell	Data	Data Fixed	
5	Container	Cell	Data	Data Strong	Data Strong Fixed

The Home Page comes from Base.Template.Style

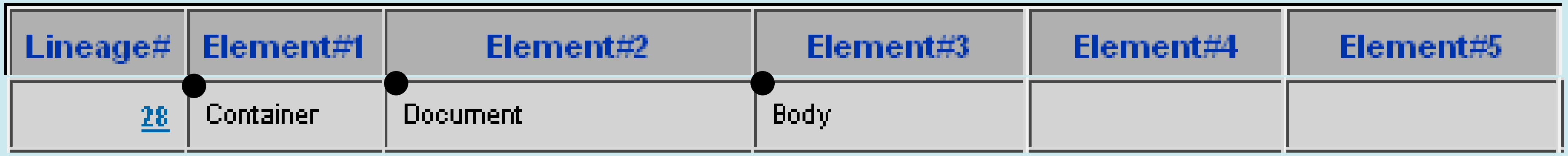


Lineage#	Element#1	Element#2	Element#3	Element#4	Element#5
28	Container	Document	Body		

Base.Template.Style is part of Sashelp.Tmplmst

```
proc template;
define style Base.Template.Style;
notes "Implicit parent for all style templates"
  ...
style Container
  "Controls all container oriented elements." /
   abstract =| on;
style Document from Container
  "Controls the various document bodies."
style Body from Document
  "Controls the Body/Frame/Contents/Page file.";
```

The Home Page comes from Base.Template.Style



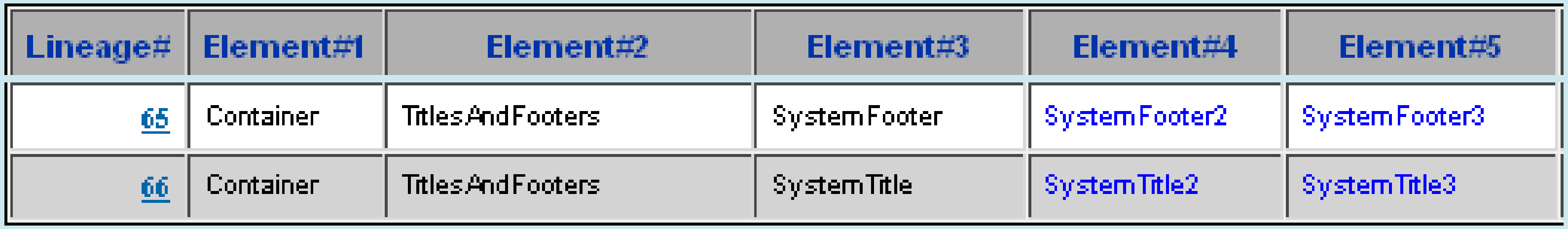
Lineage#	Element#1	Element#2	Element#3	Element#4	Element#5
28	Container	Document	Body		

Style Elements only appear ONCE: Need Redundancy. Supplied by Recrusion

```
proc template;
define style Base.Template.Style;
notes "Implicit parent for all style templates"
  ...
style Container ...;
style Document from Container ...;
style Body from Document ...;
...
style Date from Container
   "Controls how date fields look." /
   abstract =| on;
```

See Watts
Using Recursion to Trace Lineages in the SAS® ODS Styles.Default Template

What's with the Blue Text for SystemFooter2?



Lineage#	Element#1	Element#2	Element#3	Element#4	Element#5
65	Container	TitlesAndFooters	SystemFooter	SystemFooter2	SystemFooter3
66	Container	TitlesAndFooters	SystemTitle	SystemTitle2	SystemTitle3

Base.Template.Style AND Styles.Default are in Sashelp.Tmplmst

- In 9.1.3 SAS the Lineage Tracer was built exclusively from the **Styles.Default** template.
- Now **inheritance** is expressed in **Base.Template.Style. Attributes** continue to reside in the **Styles.Default** template.
- SystemFooter2 is **only** found in **Base.Template.Style**.
- There are 66 lineages in the 9.2 tracer: up from 60 in 9.1.3.
- A maximum of 12 style elements can be found in a single lineage: up from 7 in 9.1.3 SAS.

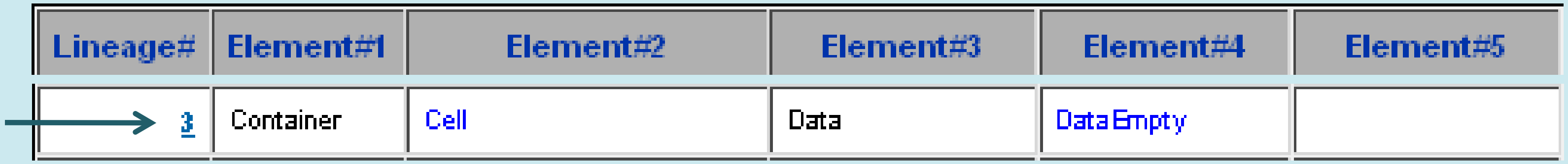
Connecting Lineage to Attribute

1) **Take the two SAS data sets created from BASE.TEMPLATE.STYLE and STYLES.DEFAULT and**
2) **Combine them with a PROC SQL FULL JOIN on STYLENAME = CLASSNAME**

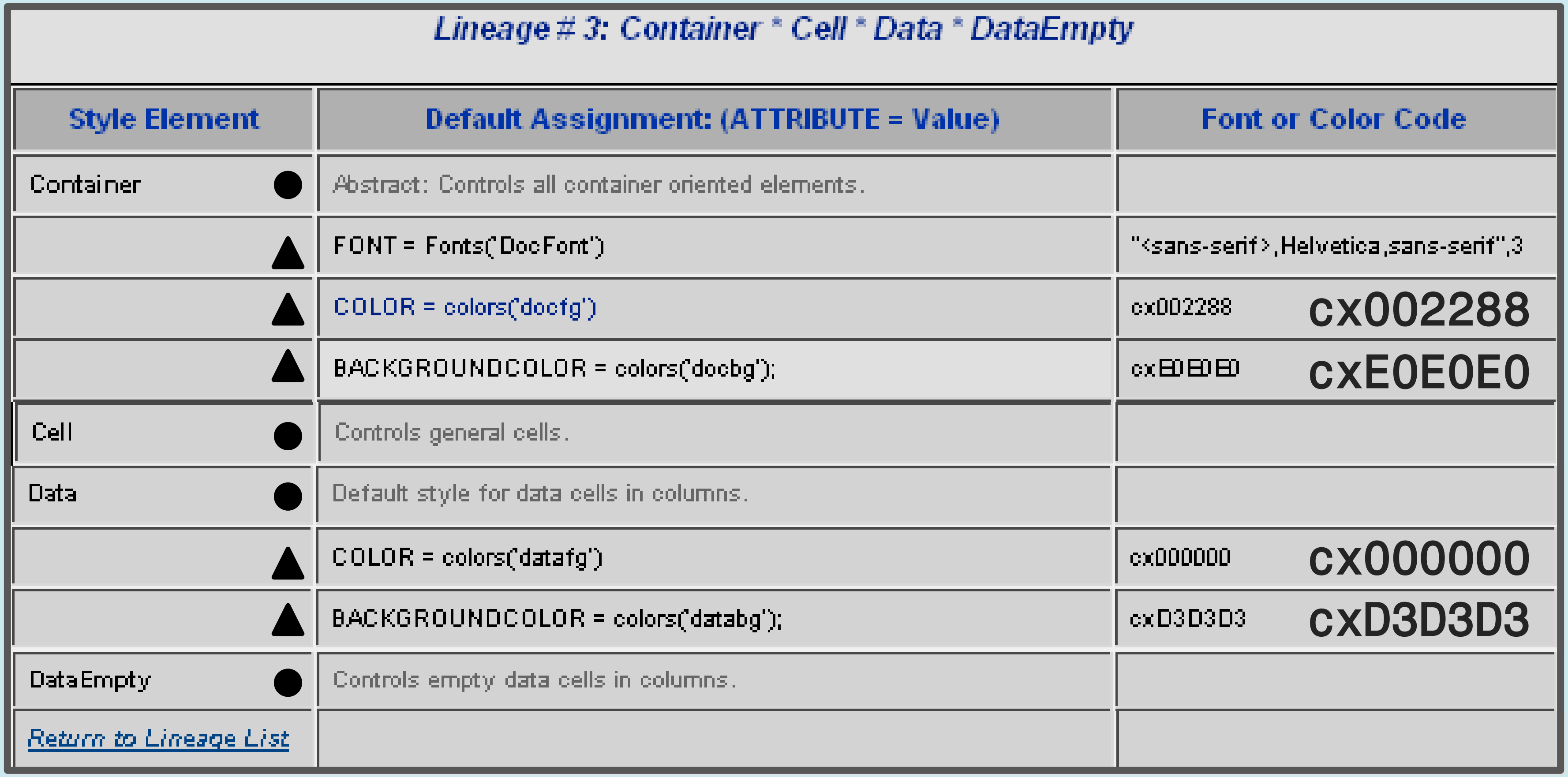
```
proc template;
define style Base.Template.Style;
...
style Container
 "Controls all container oriented elements." /
  abstract =| on;
```

```
proc template;
define style Styles.Default;
...
class Container /
 font = Fonts('DocFont')
 color = colors('docfg')
 backgroundcolor = colors('docbg');
```

Connecting Lineage to Attribute via Drill-Down



Lineage#	Element#1	Element#2	Element#3	Element#4	Element#5
3	Container	Cell	Data	DataEmpty	



*Lineage # 3: Container * Cell * Data * DataEmpty*

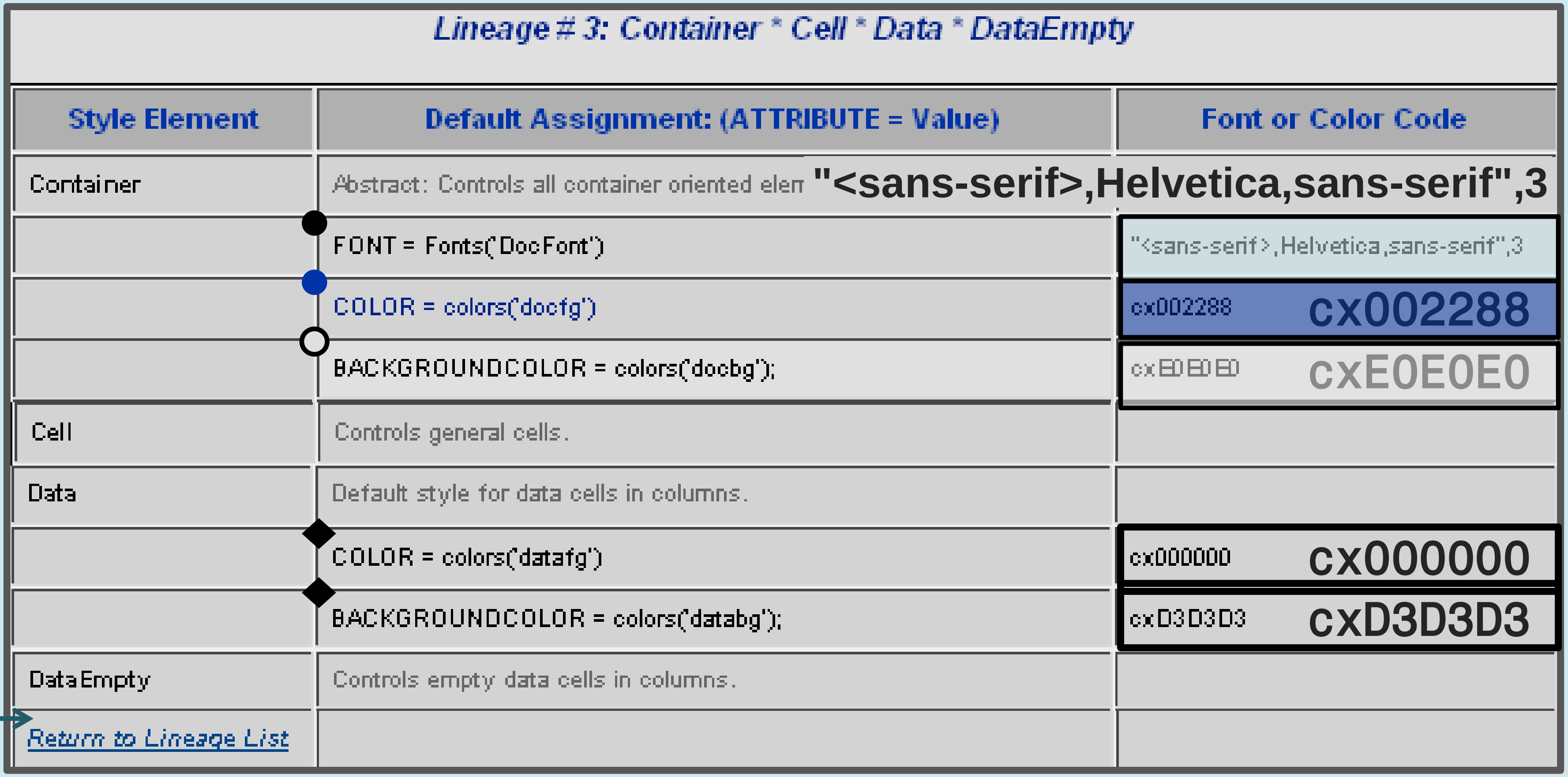
Style Element	Default Assignment: (ATTRIBUTE = Value)	Font or Color Code
Container ●	Abstract: Controls all container oriented elements.	
▲	FONT = Fonts('DocFont')	"<sans-serif>,Helvetica,sans-serif",3
▲	COLOR = colors('docfg')	cx002288 **cx002288**
▲	BACKGROUNDCOLOR = colors('docbg');	cxE0E0E0 **cxE0E0E0**
Cell ●	Controls general cells.	
Data ●	Default style for data cells in columns.	
▲	COLOR = colors('datafg')	cx000000 **cx000000**
▲	BACKGROUNDCOLOR = colors('databg');	cxD3D3D3 **cxD3D3D3**
DataEmpty ●	Controls empty data cells in columns.	
Return to Lineage List		

From: ● Base.Template.Style ▲ Styles.Default Template

Connecting Lineage to Attribute via Drill-Down

Color and Font Attributes are Enhanced:

(Settings are listed in the third column and applied in the second column)

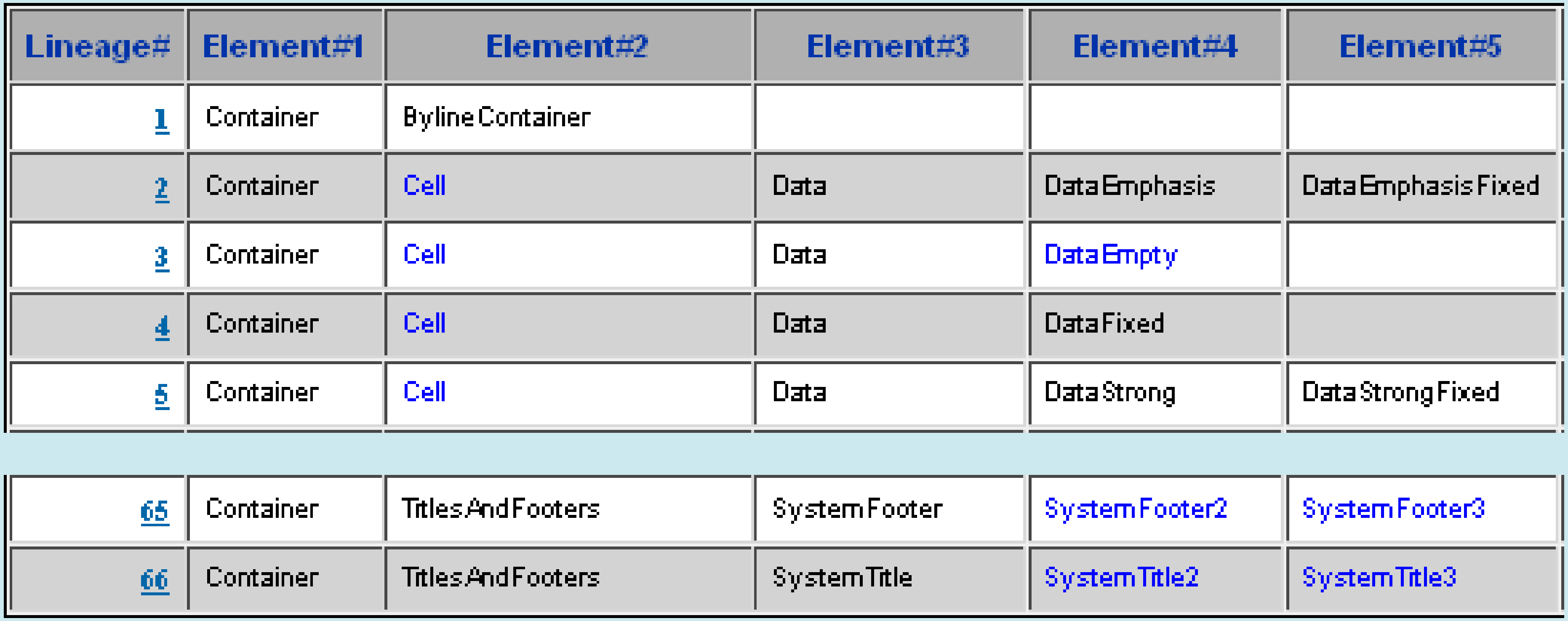


*Lineage # 3: Container * Cell * Data * DataEmpty*

Style Element	Default Assignment: (ATTRIBUTE = Value)	Font or Color Code
Container	Abstract: Controls all container oriented elem	"<sans-serif>,Helvetica,sans-serif",3
	● FONT = Fonts('DocFont')	"<sans-serif>,Helvetica,sans-serif",3
	● COLOR = colors('docfg')	cx002288 cx002288
	○ BACKGROUNDCOLOR = colors('docbg');	cxE0E0E0 cxE0E0E0
Cell	Controls general cells.	
Data	Default style for data cells in columns.	
	◆ COLOR = colors('datafg')	cx000000 cx000000
	◆ BACKGROUNDCOLOR = colors('databg');	cxD3D3D3 cxD3D3D3
DataEmpty	Controls empty data cells in columns.	
Return to Lineage List		

Connecting Lineage to Attribute via Drill-Down

Returns you to the Home Page



Lineage#	Element#1	Element#2	Element#3	Element#4	Element#5
1	Container	Byline Container			
2	Container	Cell	Data	Data Emphasis	Data Emphasis Fixed
3	Container	Cell	Data	Data Empty	
4	Container	Cell	Data	Data Fixed	
5	Container	Cell	Data	Data Strong	Data Strong Fixed
65	Container	Titles And Footers	System Footer	System Footer2	System Footer3
66	Container	Titles And Footers	System Title	System Title2	System Title3

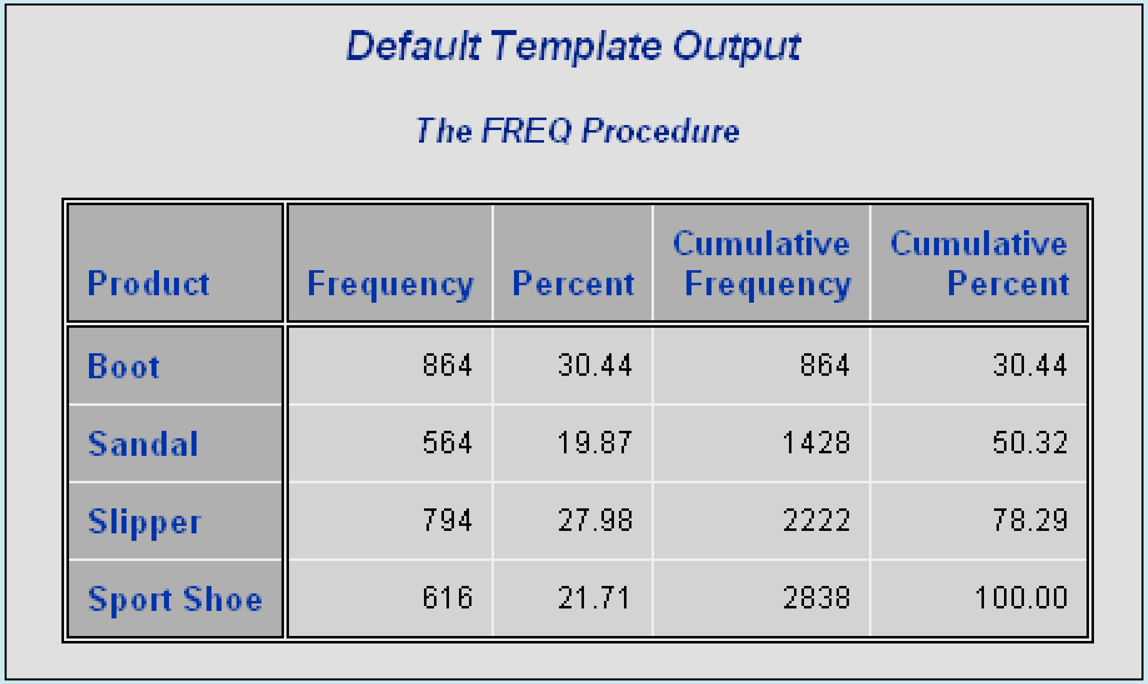
Use the Lineage Tracer to Define new Styles in ODS

Look at Default Output and where STYLE fits in to ODS

```
ods html path="&htmPath" (url=none)
  file='default.html' style=STYLES.DEFAULT;
  proc freq data=styleApp.shoes2;
    weight nstores; tables product;
  run;
ods _all_ close;
```

PROC TEMPLATE;
Define Style ...;

New Style Name here



Default Template Output

The FREQ Procedure

Product	Frequency	Percent	Cumulative Frequency	Cumulative Percent
Boot	864	30.44	864	30.44
Sandal	564	19.87	1428	50.32
Slipper	794	27.98	2222	78.29
Sport Shoe	616	21.71	2838	100.00

see Zender
Tiptoe Through the Templates

Using the CLASS Statement in New Style Definitions

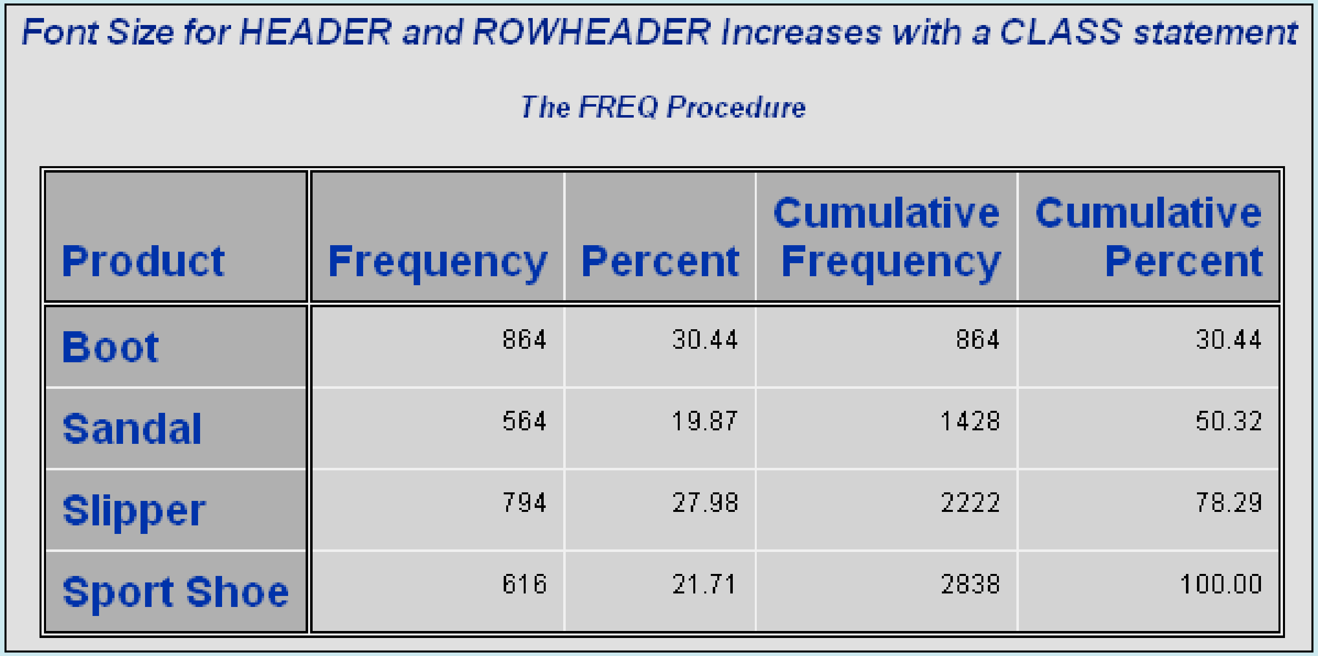
How Does **CLASS** Work?

- Inheritance is fixed in BASE.TEMPLATE.STYLE and visible in the home page of the Lineage Tracer.
- With fixed inheritance, **FROM** is not needed nor is it included in the syntax.
- However the following statements are equivalent:
  - **CLASS** Header
  - **STYLE** Header **FROM** Header
  - **STYLE** Header **FROM** _self_

Using the CLASS Statement in New Style Definitions

Change a Single Attribute in HEADER

```
proc template;
  define style styles.ChangeHeader;
  parent=styles.default;
  class header /
    font=(arial, 14pt, bold);
  end;
run;
```



Font Size for HEADER and ROWHEADER Increases with a CLASS statement

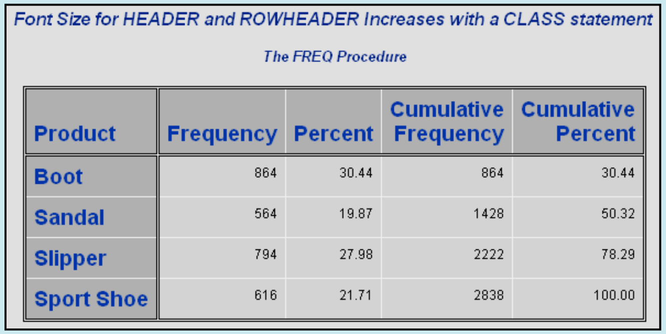
The FREQ Procedure

Product	Frequency	Percent	Cumulative Frequency	Cumulative Percent
Boot	864	30.44	864	30.44
Sandal	564	19.87	1428	50.32
Slipper	794	27.98	2222	78.29
Sport Shoe	616	21.71	2838	100.00

Using the CLASS Statement in New Style Definitions

Lineage Remains Fixed when CLASS= is used

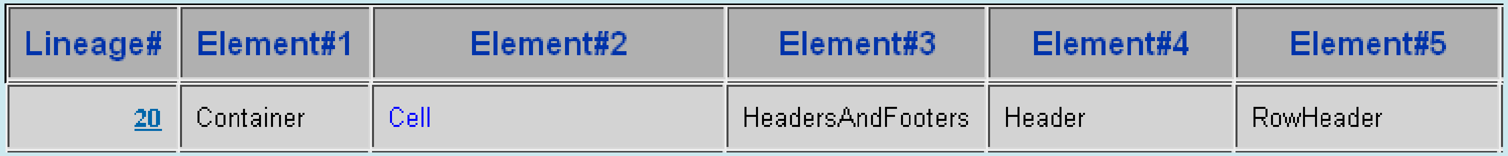
```
proc template;
define style ChangeHeader;
 class header /
  font=(arial, 14pt, bold); ...;
```



Font Size for HEADER and ROWHEADER Increases with a CLASS statement

The FREQ Procedure

Product	Frequency	Percent	Cumulative Frequency	Cumulative Percent
Boot	864	30.44	864	30.44
Sandal	564	19.87	1428	50.32
Slipper	794	27.98	2222	78.29
Sport Shoe	616	21.71	2838	100.00



Lineage#	Element#1	Element#2	Element#3	Element#4	Element#5
20	Container	Cell	HeadersAndFooters	Header	RowHeader

CLASS *Header*

$$\begin{tikzcd}\text{Container} \arrow[r] & \text{Cell} \arrow[r] & \text{HeadersAndFooters} \arrow[r] & \boxed{\text{Header}} \arrow[r] & \boxed{\text{RowHeader}}\end{tikzcd}$$

STYLE *Header from Header*

$$\begin{tikzcd}\text{Container} \arrow[r] & \text{Cell} \arrow[r] & \text{HeadersAndFooters} \arrow[r] & \text{Header} \arrow[r] & \boxed{\text{Header}} \arrow[r] & \boxed{\text{RowHeader}}\end{tikzcd}$$

Using the CLASS Statement in New Style Definitions

Inheritance at the **Attribute** Level: ***Parent*** and ***Child***

- If there are like-named style attributes in the two style elements, then the style attributes from the **child** style element are used.
- The child style element absorbs the style attributes from the **parent** style element.

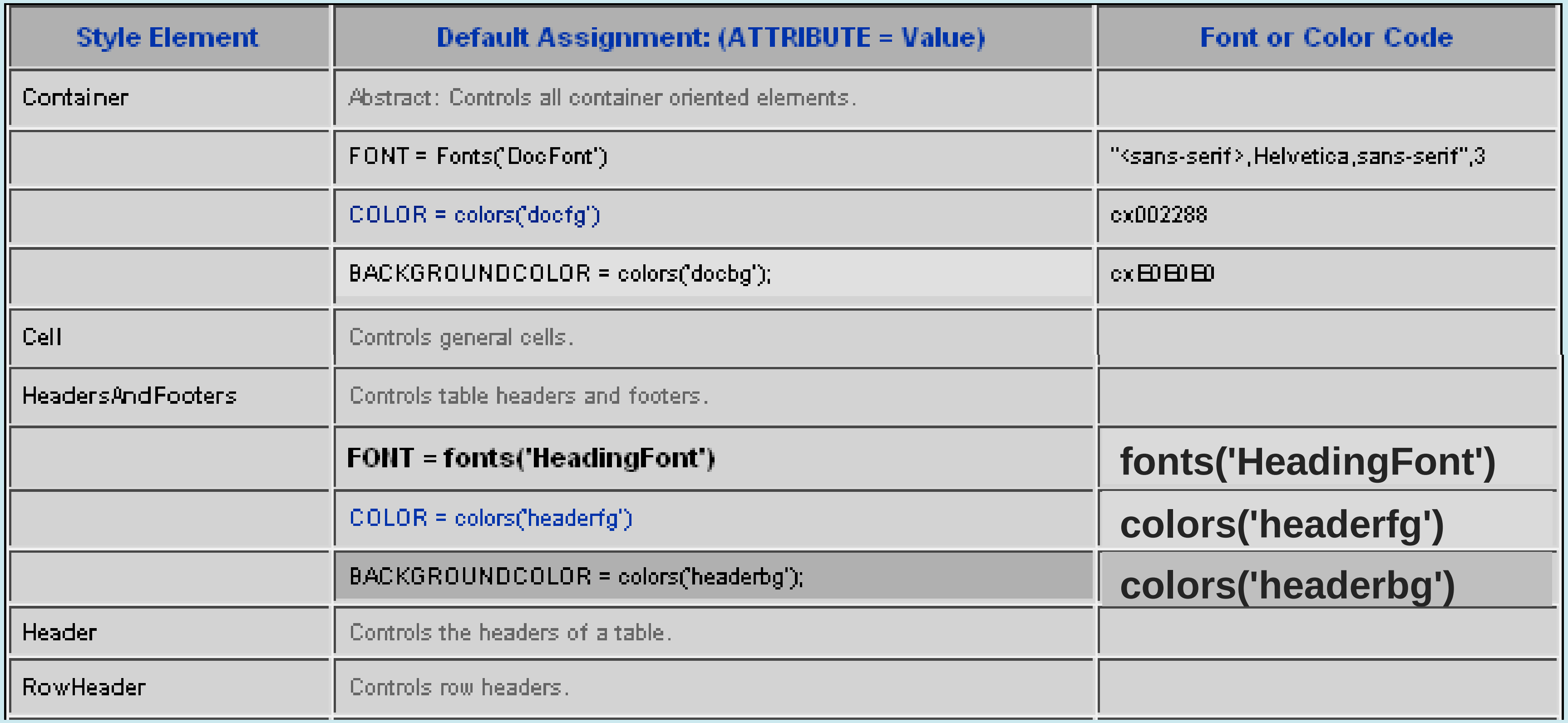
```
proc template;
define style MyStyle;
 parent=styles.default;
 style header from header/
  font=(arial, 14pt, bold);
end;
run;
```

○ **Parent** ***from header***	● **Child** **style header**
FONT = fonts('HeadingFont')	**Font=(arial, 14pt, bold)**
COLOR = colors('headerfg')	**COLOR = colors('headerfg')**
BACKGROUNDCOLOR = colors('headerbg');	**BACKGROUNDCOLOR = colors('headerbg');**

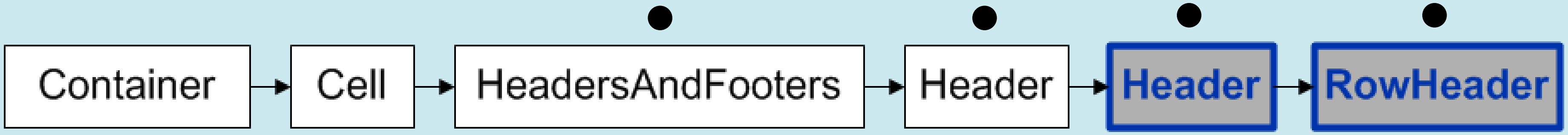
See ODS User's Guide 9.2, p. 521

Using the CLASS Statement in New Style Definitions

Change a Single Attribute in HEADER: `font=(arial, 14pt, bold)`



Style Element	Default Assignment: (ATTRIBUTE = Value)	Font or Color Code
Container	Abstract: Controls all container oriented elements.	
	FONT = Fonts('DocFont')	"<sans-serif>,Helvetica,sans-serif",3
	COLOR = colors('docfg')	cx002288
	BACKGROUNDCOLOR = colors('docbg');	cxE0E0E0
Cell	Controls general cells.	
● HeadersAndFooters	Controls table headers and footers.	
	FONT = fonts('HeadingFont')	**fonts('HeadingFont')**
	COLOR = colors('headerfg')	**colors('headerfg')**
	BACKGROUNDCOLOR = colors('headerbg');	**colors('headerbg')**
● Header	Controls the headers of a table.	
RowHeader	Controls row headers.	

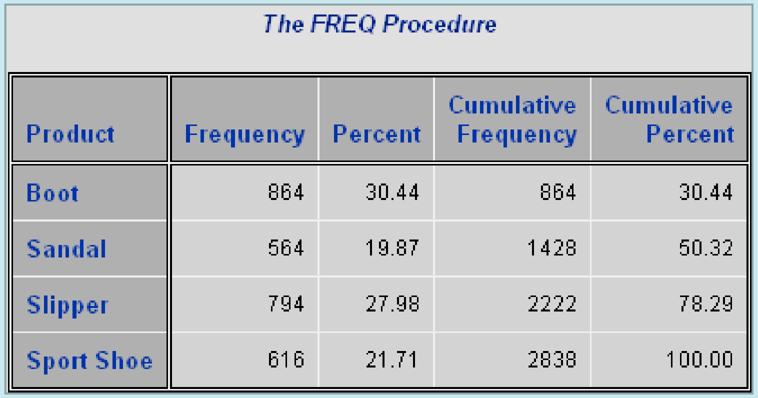


font=(arial, 14pt, bold)

Using the CLASS Statement in New Style Definitions

Change a Single Attribute in HEADER

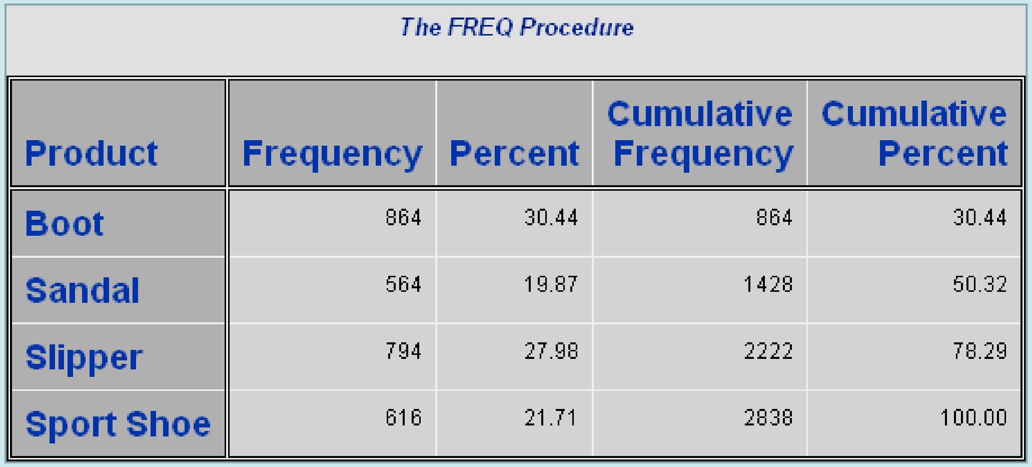
BEFORE
"<sans-serif>,Helvetica,sans-serif",4,bold



The FREQ Procedure

Product	Frequency	Percent	Cumulative Frequency	Cumulative Percent
Boot	864	30.44	864	30.44
Sandal	564	19.87	1428	50.32
Slipper	794	27.98	2222	78.29
Sport Shoe	616	21.71	2838	100.00

AFTER
font=(arial, 14pt, bold)



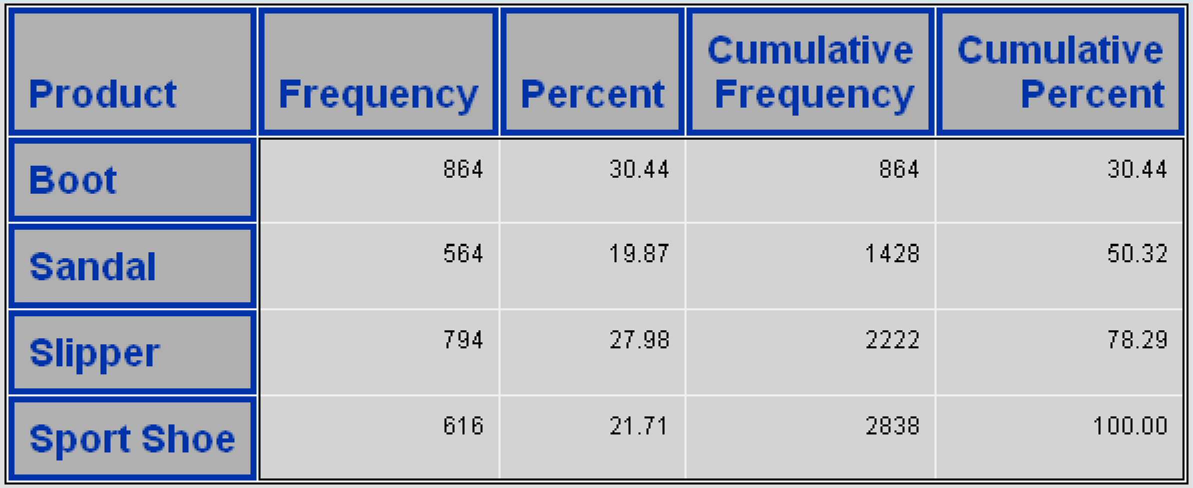
The FREQ Procedure

Product	Frequency	Percent	Cumulative Frequency	Cumulative Percent
Boot	864	30.44	864	30.44
Sandal	564	19.87	1428	50.32
Slipper	794	27.98	2222	78.29
Sport Shoe	616	21.71	2838	100.00

Using the CLASS Statement in New Style Definitions

Change non-lineage Attributes in HEADER?

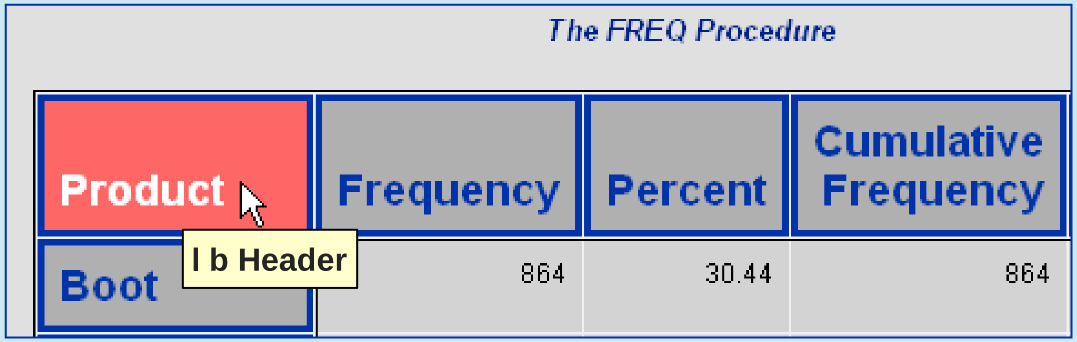
```
proc template;
 define style styles.NonLineageAttrs;
 parent=styles.default;
 class header /
  font=(arial, 14pt, bold)
  bordercolor=colors('headerfg')
  borderwidth=3;
end;
run;
```



Product	Frequency	Percent	Cumulative Frequency	Cumulative Percent
Boot	864	30.44	864	30.44
Sandal	564	19.87	1428	50.32
Slipper	794	27.98	2222	78.29
Sport Shoe	616	21.71	2838	100.00

Using the CLASS Statement in New Style Definitions

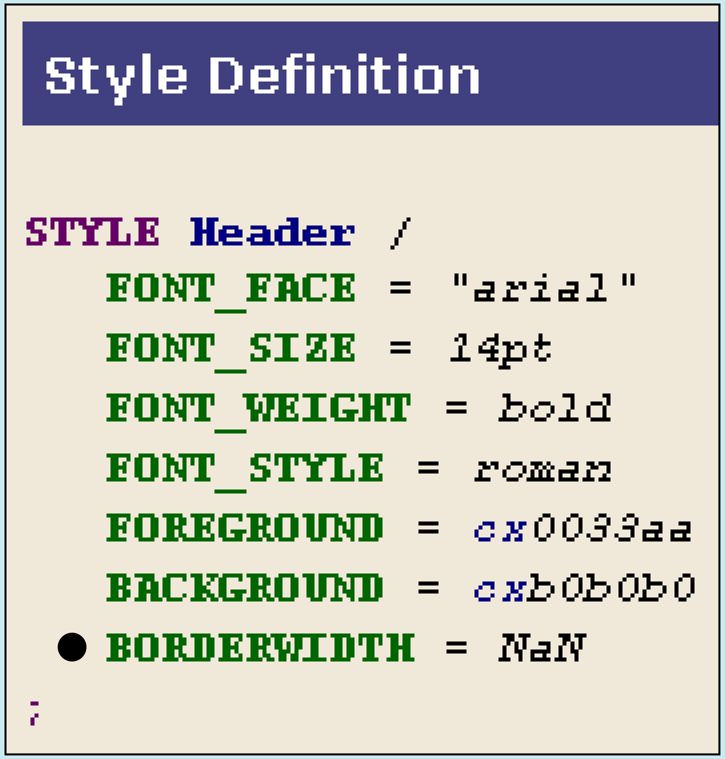
Change non-lineage Attributes in HEADER?



The FREQ Procedure

Product	Frequency	Percent	Cumulative Frequency
Boot	864	30.44	864

No



Style Definition

```
STYLE Header /
    FONT_FACE = "arial"
    FONT_SIZE = 14pt
    FONT_WEIGHT = bold
    FONT_STYLE = roman
    FOREGROUND = cx0033aa
    BACKGROUND = cxb0b0b0
  ● BORDERWIDTH = NaN
;
```

Will STYLE_POPUP list the border settings?

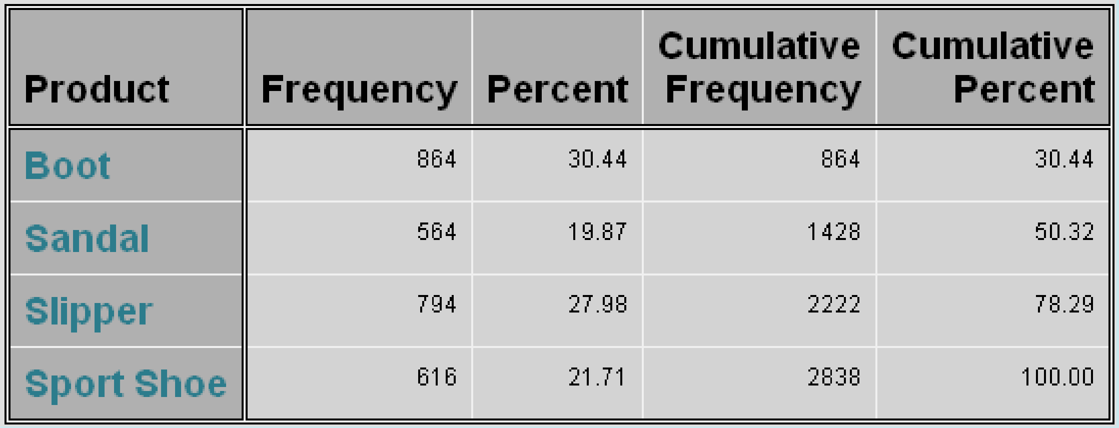
Borderwidth (3) is coded *NaN*
Bordercolor is not listed

To find out how the STYLE_POPUP tagset works, see Haworth, Zender, Berlew *Output Delivery System: The Basics and Beyond*

Using the CLASS Statement in New Style Definitions

Change Two Style Elements in the **Same** Lineage

```
proc template;
 define style styles.TwoClassesA;
 parent=styles.default;
 class header /
  font=(arial,14pt,bold)
  color=black;
 class rowheader /
  color=CX2C7C8C;
 end;
run;
```



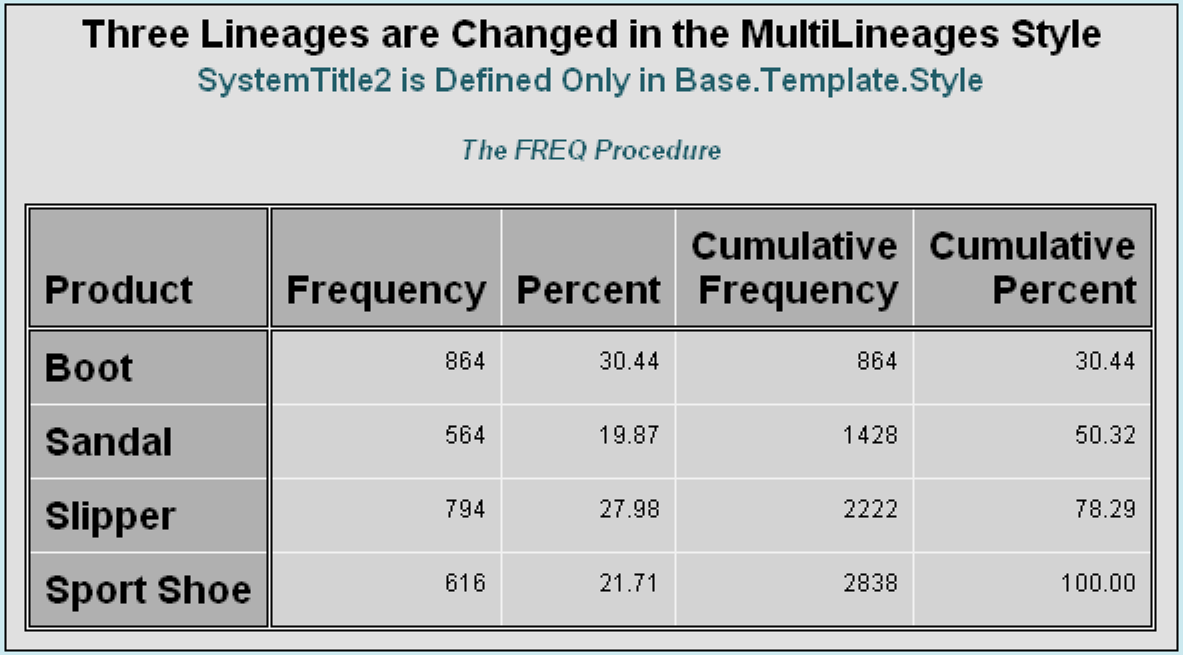
Product	Frequency	Percent	Cumulative Frequency	Cumulative Percent
Boot	864	30.44	864	30.44
Sandal	564	19.87	1428	50.32
Slipper	794	27.98	2222	78.29
Sport Shoe	616	21.71	2838	100.00

$$\begin{tikzcd} \text{Container} \arrow[r] & \text{Cell} \arrow[r] & \text{HeadersAndFooters} \arrow[r] & \text{Header} \arrow[r] & \text{Header} \arrow[r] & \text{RowHeader} \arrow[r] & \text{RowHeader} \end{tikzcd}$$

Using the CLASS Statement in New Style Definitions

Change Style Elements in **Different** Lineages

```
proc template;
  define style styles.MultiLineages;
  parent=styles.default;
  class Header/
    font=(arial, 14pt, bold)
    color=black;
  class systemTitle /
    color=black
    fontStyle=roman
    fontsize=14pt;
  class systemTitle2 /
    color=CX215D69
    fontSize=12pt;
  class procTitle /
    color=CX215D69;
 end;
run;
```



Three Lineages are Changed in the MultiLineages Style

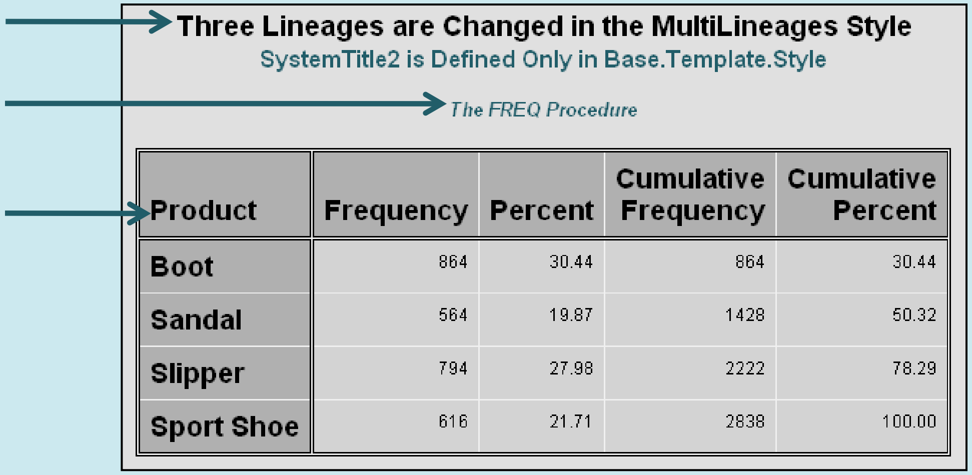
SystemTitle2 is Defined Only in Base.Template.Style

The FREQ Procedure

Product	Frequency	Percent	Cumulative Frequency	Cumulative Percent
Boot	864	30.44	864	30.44
Sandal	564	19.87	1428	50.32
Slipper	794	27.98	2222	78.29
Sport Shoe	616	21.71	2838	100.00

Using the CLASS Statement in New Style Definitions

Change Style Elements in **Different** Lineages


Three Lineages are Changed in the MultiLineages Style
SystemTitle2 is Defined Only in Base.Template.Style

The FREQ Procedure

Product	Frequency	Percent	Cumulative Frequency	Cumulative Percent
Boot	864	30.44	864	30.44
Sandal	564	19.87	1428	50.32
Slipper	794	27.98	2222	78.29
Sport Shoe	616	21.71	2838	100.00

From Lineage #20

$$\begin{tikzcd}\boxed{\text{Container}} \arrow[r] & \boxed{\text{Cell}} \arrow[r] & \boxed{\text{HeadersAndFooters}} \arrow[r] & \boxed{\text{Header}} \arrow[r] & \boxed{\textbf{Header}} \arrow[r] & \boxed{\textbf{RowHeader}}\end{tikzcd}$$

From Lineage #66

$$\begin{tikzcd}\boxed{\text{Container}} \arrow[r] & \boxed{\text{TitlesAndFooters}} \arrow[r] & \boxed{\text{SystemTitle}} \arrow[r] & \boxed{\textbf{SystemTitle}} \arrow[r] & \boxed{\textbf{SystemTitle2}}\end{tikzcd}$$

From Lineage #64

$$\begin{tikzcd}\boxed{\text{Container}} \arrow[r] & \boxed{\text{TitlesAndFooters}} \arrow[r] & \boxed{\text{ProcTitle}} \arrow[r] & \boxed{\textbf{ProcTitle}}\end{tikzcd}$$

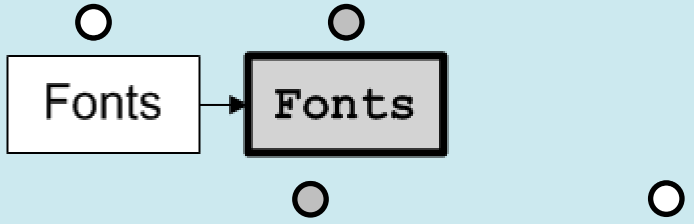
Using the CLASS Statement in New Style Definitions

Change the FONTS Style Element

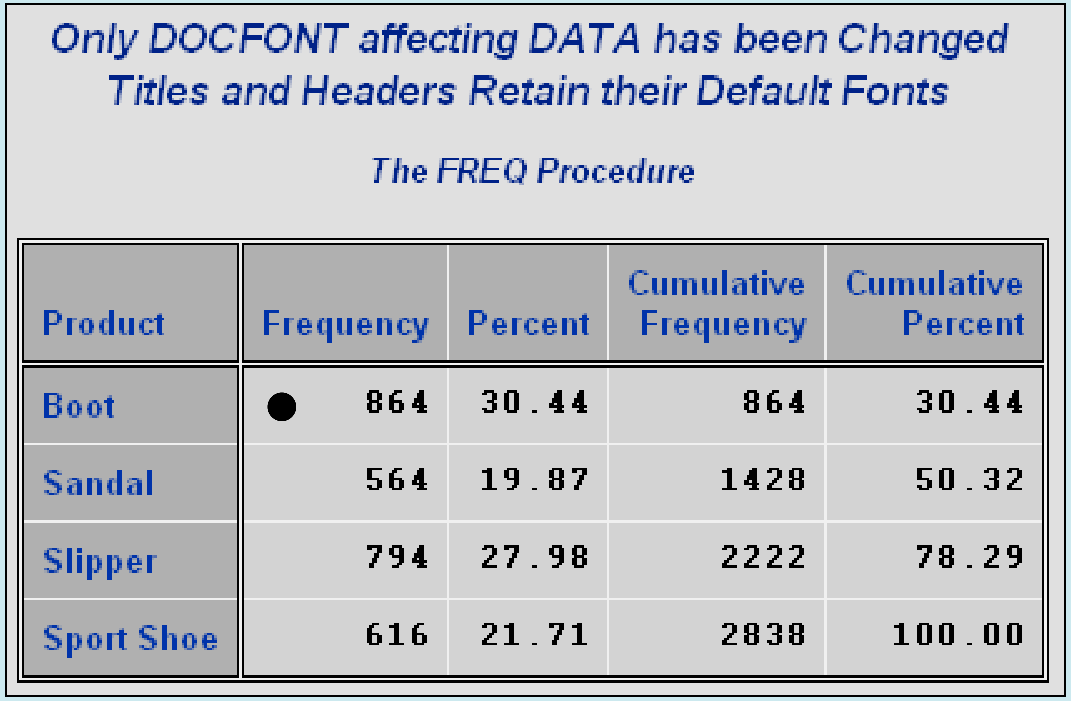
```
proc template;
 define style styles.Font1;
 parent=styles.default;
 CLASS fonts /
 'docFont'=("Courier",3,Bold);
 end;
run;
```

Not a CONTAINER lineage

Inheritance works the same:



STYLE=FONTS FROM FONTS



Only DOCFONT affecting DATA has been Changed
Titles and Headers Retain their Default Fonts

The FREQ Procedure

Product	Frequency	Percent	Cumulative Frequency	Cumulative Percent
Boot	● 864	30.44	864	30.44
Sandal	564	19.87	1428	50.32
Slipper	794	27.98	2222	78.29
Sport Shoe	616	21.71	2838	100.00

Using STYLE (no FROM) for New Style Definitions

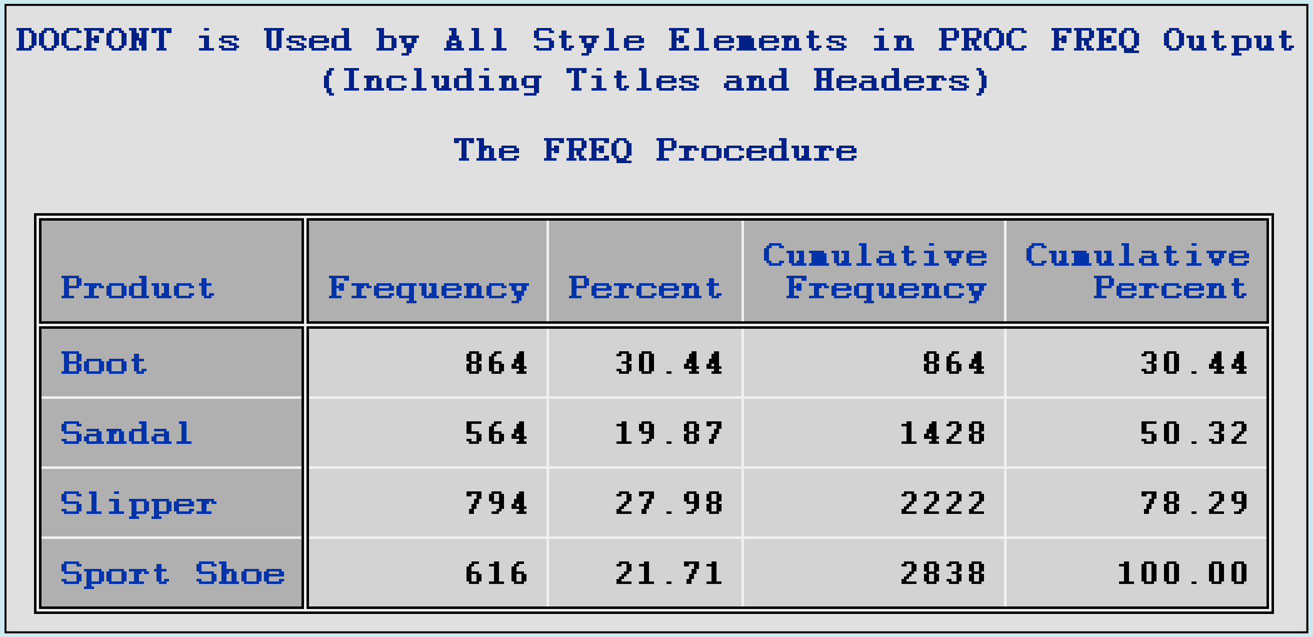
How does **STYLE** differ from **CLASS**?

- No inheritance from **ancestor** or **self**
- Equivalent to REPLACE in 9.1.3 SAS

Using the STYLE Statement in New Style Definitions

Change the FONTS Style Element

```
proc template;
 define style styles.Font2;
 parent=styles.default;
 STYLE fonts /
 'docFont'=("Courier",3,Bold);
 end;
run;
```



DOCFONT is Used by All Style Elements in PROC FREQ Output
(Including Titles and Headers)

The FREQ Procedure

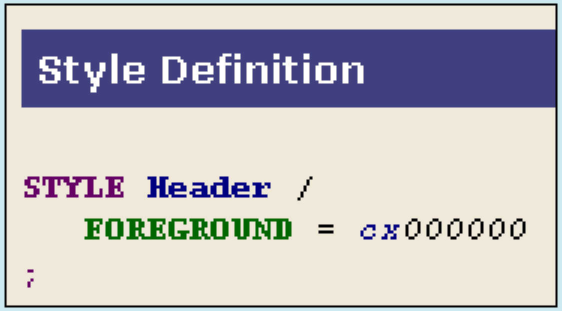
Product	Frequency	Percent	Cumulative Frequency	Cumulative Percent
Boot	864	30.44	864	30.44
Sandal	564	19.87	1428	50.32
Slipper	794	27.98	2222	78.29
Sport Shoe	616	21.71	2838	100.00

Starts a new lineage with just one style element

Using the STYLE Statement in New Style Definitions

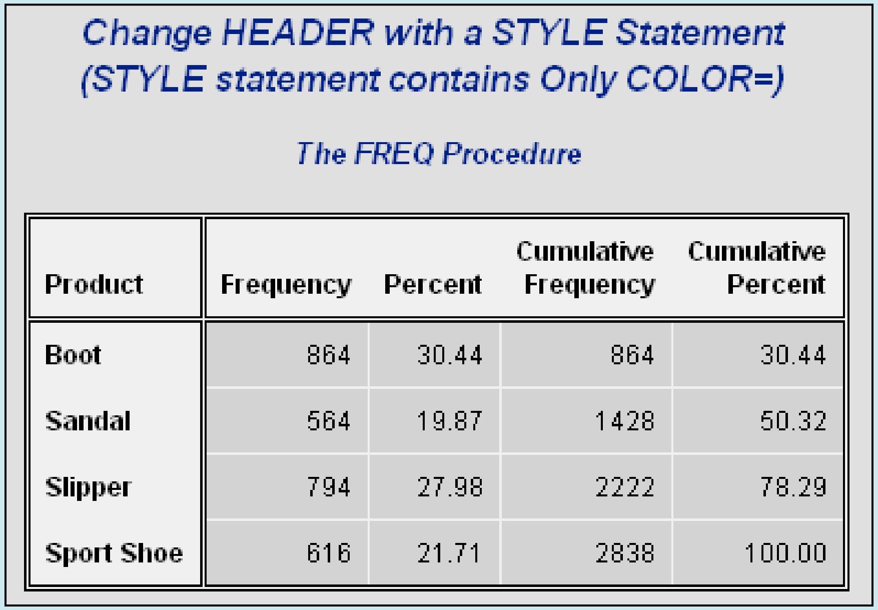
Change the HEADER Style Element

```
proc template;
 define style styles.HdrStyle;
  parent=styles.default;
 *style Header /
   color=CX000000;
 end;
run;
```



Style Definition

```
STYLE Header /
   FOREGROUND = cx000000
;
```



Change HEADER with a STYLE Statement
(STYLE statement contains Only COLOR=)

The FREQ Procedure

Product	Frequency	Percent	Cumulative Frequency	Cumulative Percent
Boot	864	30.44	864	30.44
Sandal	564	19.87	1428	50.32
Slipper	794	27.98	2222	78.29
Sport Shoe	616	21.71	2838	100.00

STYLE HEADER /
Defines a Lineage

$$\begin{tikzcd}\boxed{\text{Header}} \arrow[r] & \boxed{\text{RowHeader}}\end{tikzcd}$$

Where do the values for BACKGROUNDCOLOR and FONT come from?

Using STYLE … FROM in New Style Definitions

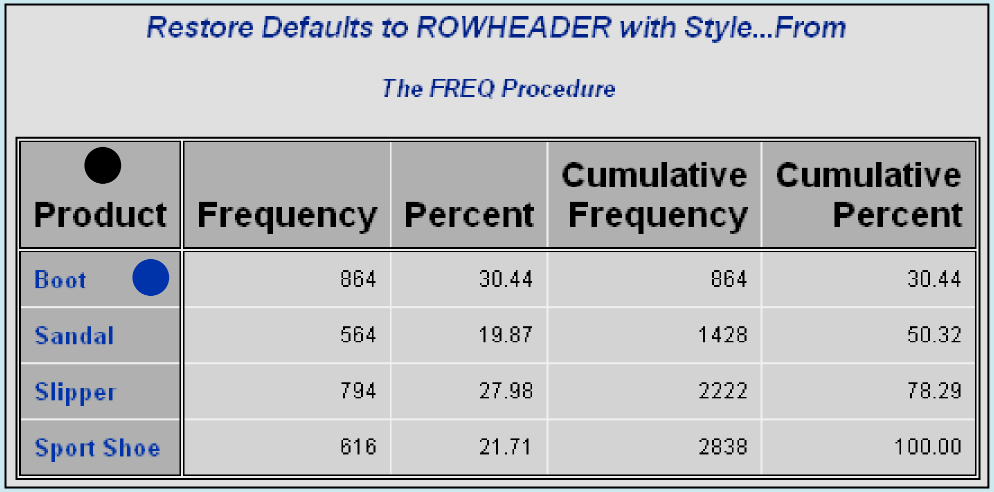
How STYLE… FROM Differs From CLASS and STYLE:

- In **CLASS,** inheritance is fixed in **Base.Template.Style** and expressed in the Home Page of the **Lineage Tracer**.
- In **STYLE,** there is no inheritance from any **ancestor** or **self.** Instead a new lineage is defined.
- **STYLE … FROM** is the most complicated system of inheritance in ODS. Inheritance is fluid, since FROM can reference an ***early ancestor***, ***descenden***t or ***cousin***. It is the "Aunt Suzie" of ODS.

Using STYLE … FROM in New Style Definitions

Inherit from a **Grandparent** to Preserve Default Settings

```
proc template;
 define style styles.DefRowHdr;
 parent=styles.default;
 class header /
   font=(arial, 14pt, bold)
   color=black;
 style rowHeader FROM headersAndFooters;
 end;
run;
```



Restore Defaults to ROWHEADER with Style...From

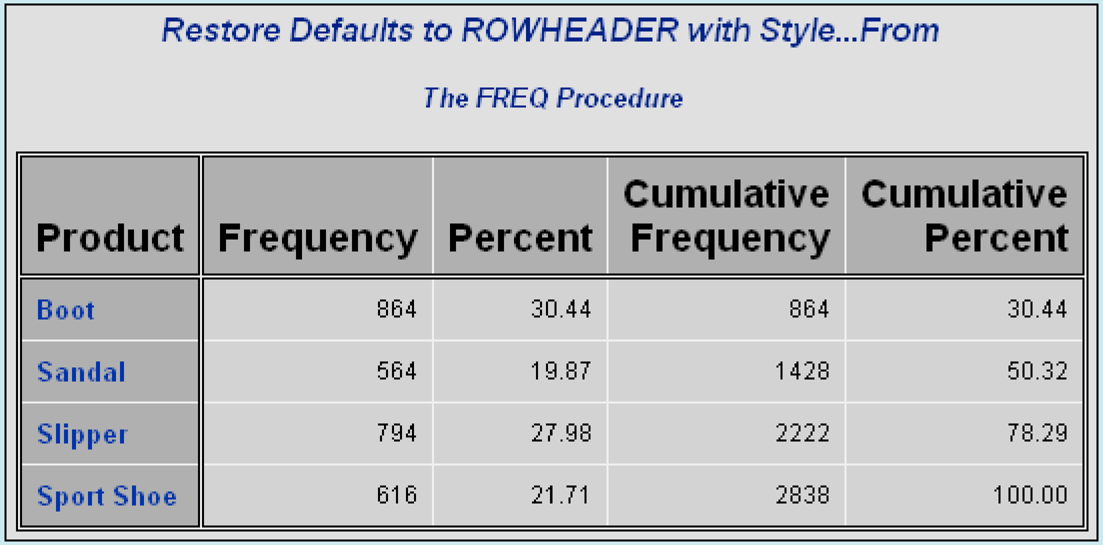
The FREQ Procedure

● Product	Frequency	Percent	Cumulative Frequency	Cumulative Percent
Boot ●	864	30.44	864	30.44
Sandal	564	19.87	1428	50.32
Slipper	794	27.98	2222	78.29
Sport Shoe	616	21.71	2838	100.00

Adapted from
SAS® Style Templates: Always in Fashion
by Cynthia Zender

Using STYLE … FROM in New Style Definitions

Inherit from a **Grandparent** Style Element

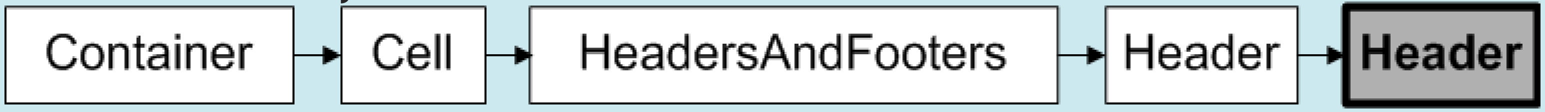


Restore Defaults to ROWHEADER with Style...From

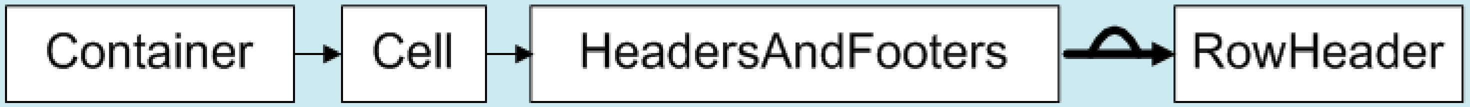
The FREQ Procedure

Product	Frequency	Percent	Cumulative Frequency	Cumulative Percent
Boot	864	30.44	864	30.44
Sandal	564	19.87	1428	50.32
Slipper	794	27.98	2222	78.29
Sport Shoe	616	21.71	2838	100.00

```
class header /
 font=(arial, 14pt, bold)
 color=black;
```



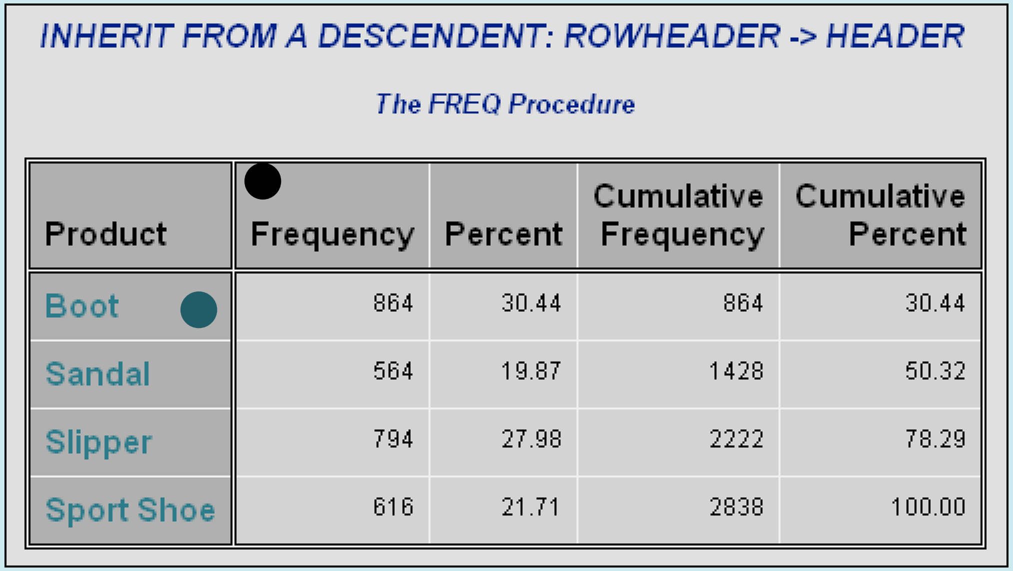
```
style rowHeader FROM headersAndFooters;
```



Using STYLE … FROM in New Style Definitions

Inherit from a **Child** Style Element

```
proc template;
  define style styles.rowHdr2Hdr;
  parent=styles.default;
 style rowHeader from headersAndFooters /
    font=(arial,12pt,bold)
    color=CX2C7C8C;
 style header from rowHeader /
    color=black;
end;
run;
```



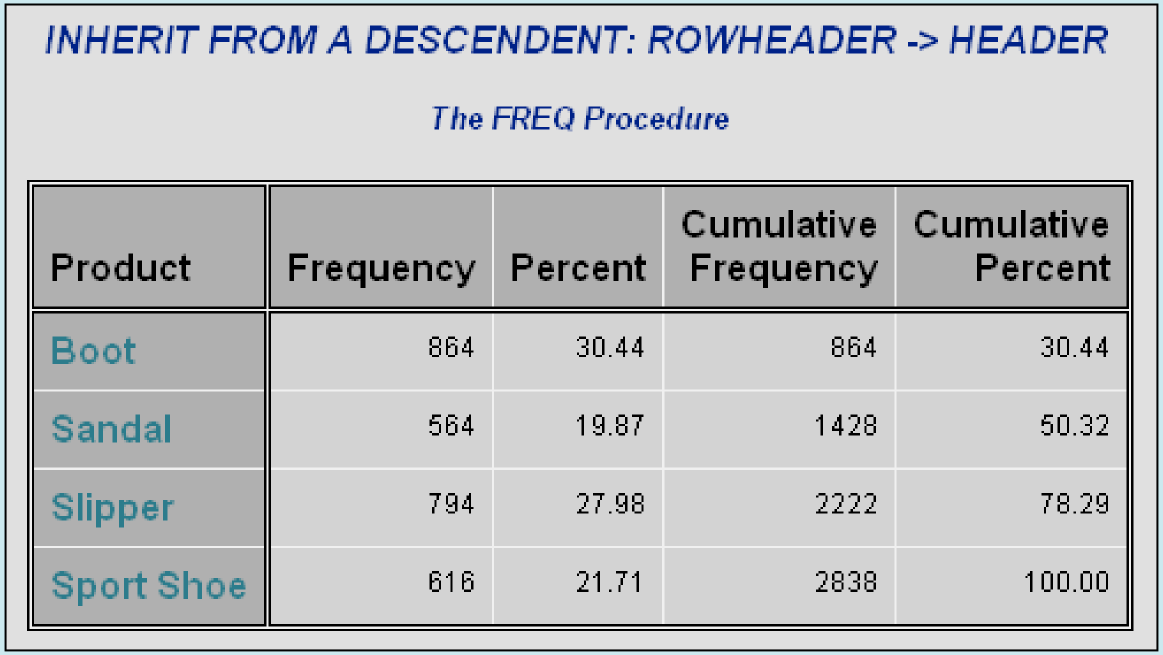
INHERIT FROM A DESCENDENT: ROWHEADER -> HEADER

The FREQ Procedure

Product	● Frequency	Percent	Cumulative Frequency	Cumulative Percent
Boot ●	864	30.44	864	30.44
Sandal	564	19.87	1428	50.32
Slipper	794	27.98	2222	78.29
Sport Shoe	616	21.71	2838	100.00

Using STYLE … FROM in New Style Definitions

Inherit from a **Child** Style Element

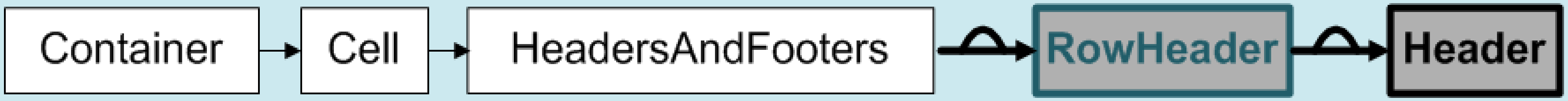


INHERIT FROM A DESCENDENT: ROWHEADER -> HEADER

The FREQ Procedure

Product	Frequency	Percent	Cumulative Frequency	Cumulative Percent
Boot	864	30.44	864	30.44
Sandal	564	19.87	1428	50.32
Slipper	794	27.98	2222	78.29
Sport Shoe	616	21.71	2838	100.00

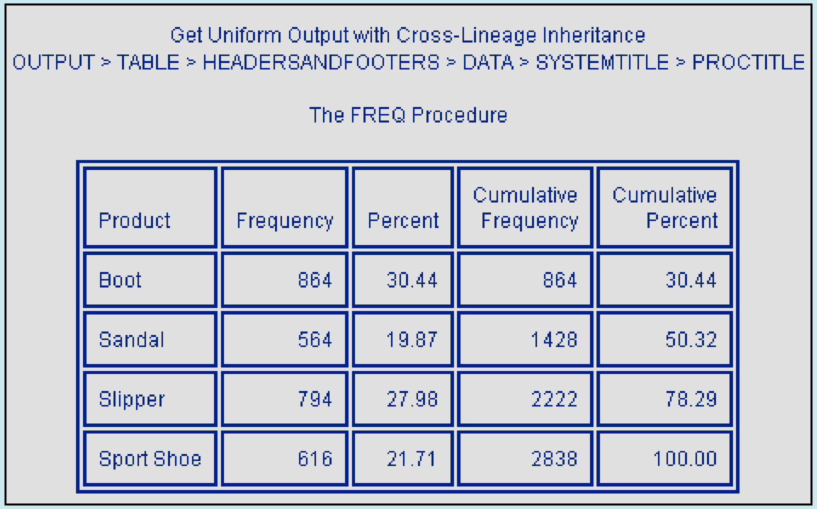
```
Style rowHeader from headersAndFooters/ ...;
style header FROM rowHeader / ...;
```



Using STYLE … FROM in New Style Definitions

Cross-Lineage Inheritance

```
proc template;
 define style styles.crossLin;
 parent=styles.default;
 style table from output/
  backgroundcolor=colors('docbg')
  bordercolor=colors('docfg')
  borderwidth=2  borderspacing=2;
 style headersAndFooters from table;
 style data from headersAndFooters;
 style systemTitle from data /
  padding=0  borderwidth=0  borderspacing=0;
 style procTitle from systemTitle;
 end;
run;
```



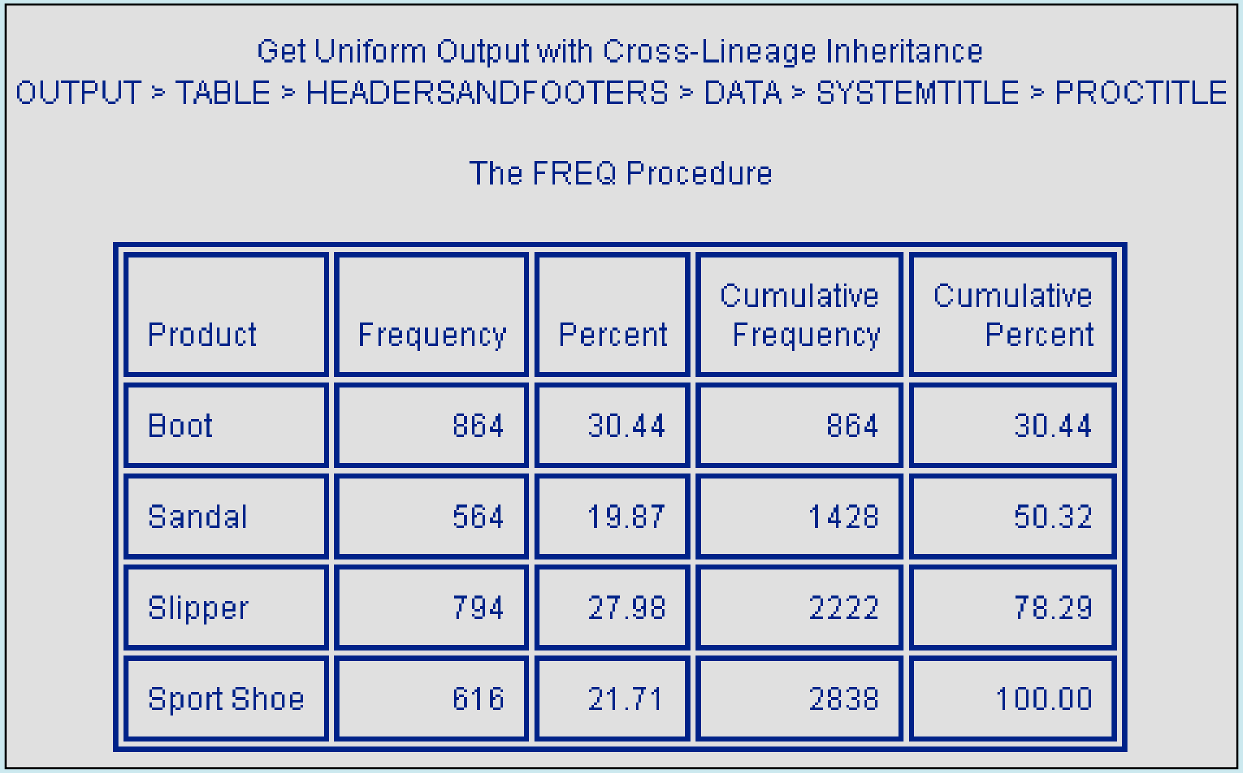
Get Uniform Output with Cross-Lineage Inheritance
OUTPUT > TABLE > HEADERSANDFOOTERS > DATA > SYSTEMTITLE > PROCTITLE

The FREQ Procedure

Product	Frequency	Percent	Cumulative Frequency	Cumulative Percent
Boot	864	30.44	864	30.44
Sandal	564	19.87	1428	50.32
Slipper	794	27.98	2222	78.29
Sport Shoe	616	21.71	2838	100.00

Using STYLE … FROM in New Style Definitions

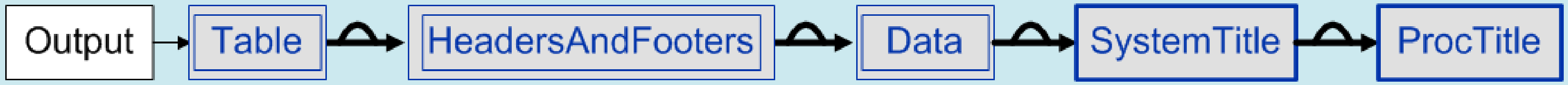
Cross-Lineage Inheritance
Where Inheritance is really s*t*r*e*t*c*h*e*d



Get Uniform Output with Cross-Lineage Inheritance
OUTPUT > TABLE > HEADERSANDFOOTERS > DATA > SYSTEMTITLE > PROCTITLE

The FREQ Procedure

Product	Frequency	Percent	Cumulative Frequency	Cumulative Percent
Boot	864	30.44	864	30.44
Sandal	564	19.87	1428	50.32
Slipper	794	27.98	2222	78.29
Sport Shoe	616	21.71	2838	100.00



Lineage #	53	53	6	2	66	64

Map new Styles to SAS Procedures

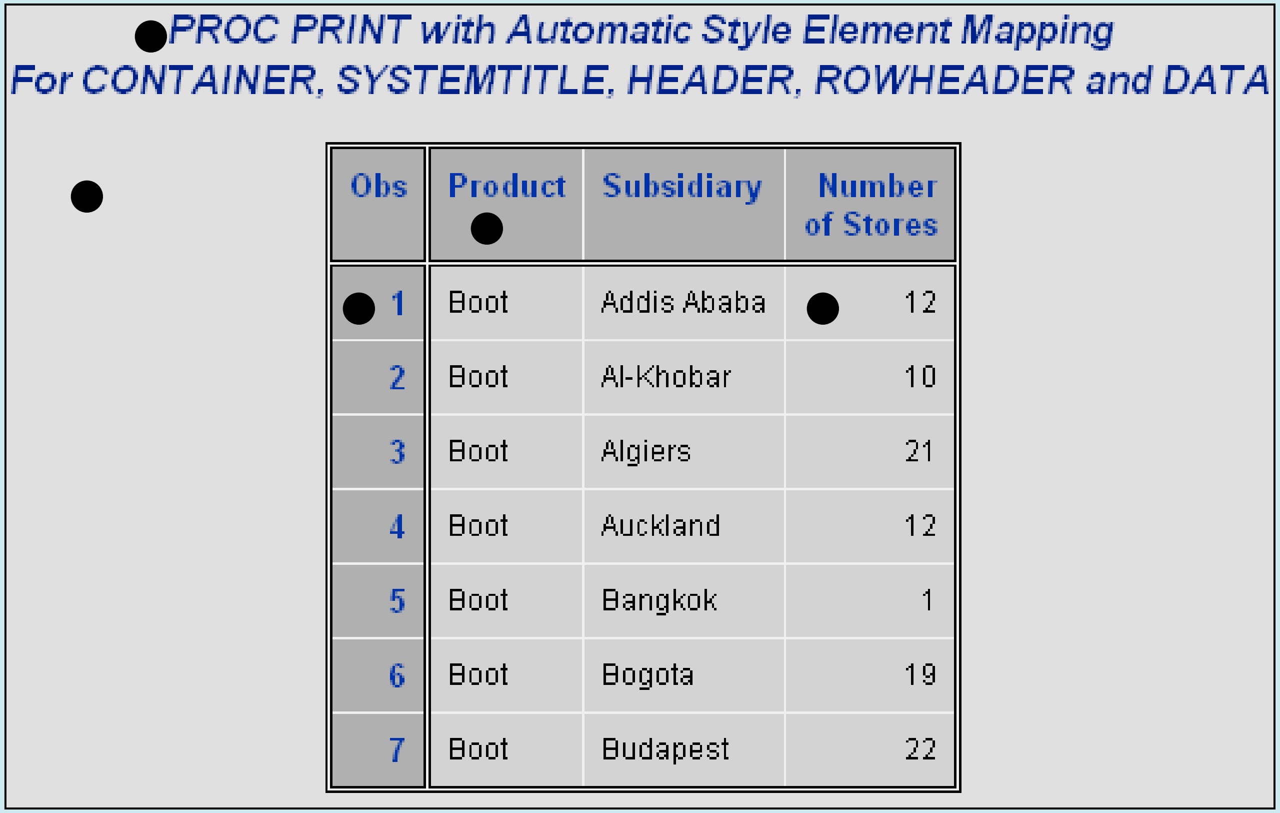
- Once styles are defined they can be mapped to SAS PROCs that make up the ***fixings*** of the ODS ***sandwich***
- Automatic mappings are **PROC-specific**
- Mappings use style element **Names** from base.template.style
- Automatic mapping has been used in every **PROC FREQ** example up to this point.
- Extra steps are required for mapping **user-named *style elements*** to output regions.

Map new Styles to SAS Procedures

PROC PRINT: **No Options**

```
ods html ... style=STYLES.default;
 proc print data=styleApp.shoes2(obs=7) label;
 run;
ods html close;
```

Container
SystemTitle
Header
RowHeader
Data



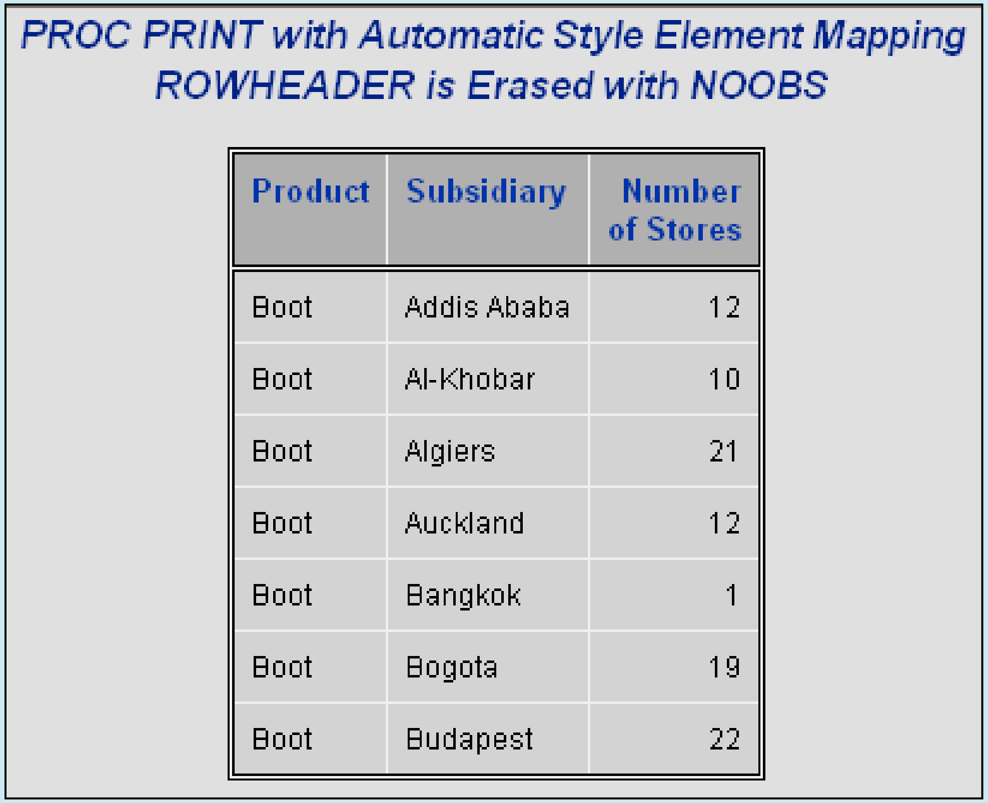
PROC PRINT with Automatic Style Element Mapping
For CONTAINER, SYSTEMTITLE, HEADER, ROWHEADER and DATA

Obs	Product	Subsidiary	Number of Stores
1	Boot	Addis Ababa	12
2	Boot	Al-Khobar	10
3	Boot	Algiers	21
4	Boot	Auckland	12
5	Boot	Bangkok	1
6	Boot	Bogota	19
7	Boot	Budapest	22

Map new Styles to SAS Procedures

PROC PRINT: **noobs**

```
ods html ... style=STYLES.default;
 proc print data=styleApp.shoes2(obs=7) noobs label;
 run;
ods html close;
```



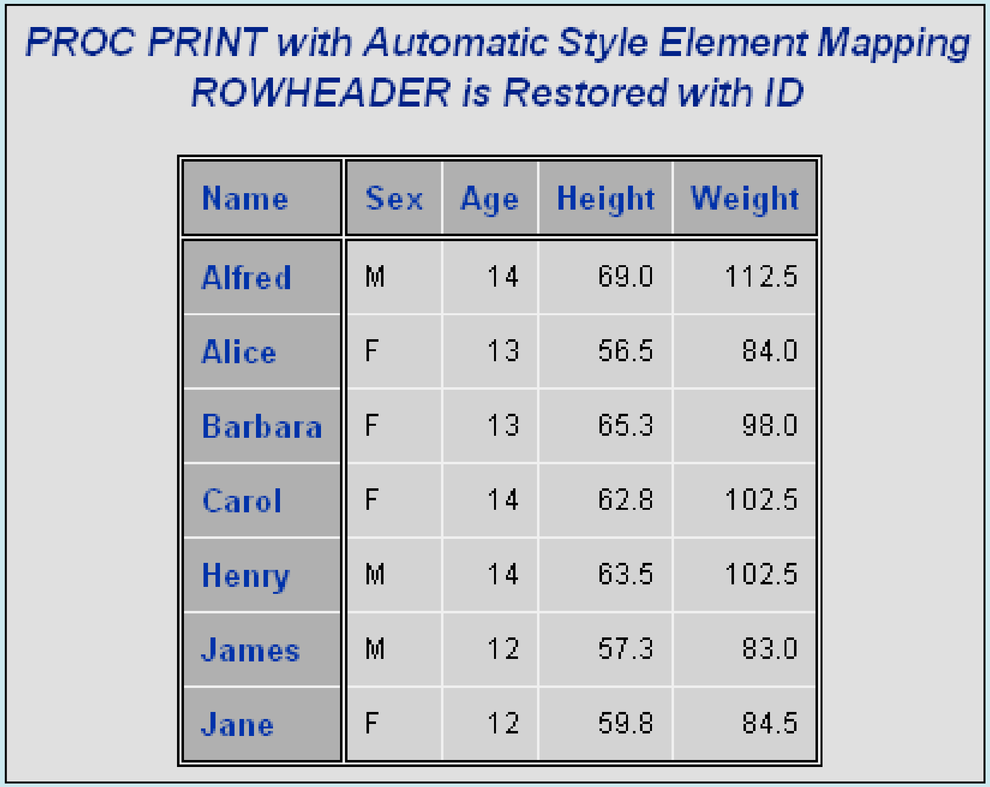
PROC PRINT with Automatic Style Element Mapping
ROWHEADER is Erased with NOOBS

Product	Subsidiary	Number of Stores
Boot	Addis Ababa	12
Boot	Al-Khobar	10
Boot	Algiers	21
Boot	Auckland	12
Boot	Bangkok	1
Boot	Bogota	19
Boot	Budapest	22

Map new Styles to SAS Procedures

PROC PRINT: **ID**

```
ods html ... style=STYLES.default;
 proc print data=sashelp.class(obs=7) noobs label;
   ID Name;
 run;
ods html close;
```



PROC PRINT with Automatic Style Element Mapping
ROWHEADER is Restored with ID

Name	Sex	Age	Height	Weight
Alfred	M	14	69.0	112.5
Alice	F	13	56.5	84.0
Barbara	F	13	65.3	98.0
Carol	F	14	62.8	102.5
Henry	M	14	63.5	102.5
James	M	12	57.3	83.0
Jane	F	12	59.8	84.5

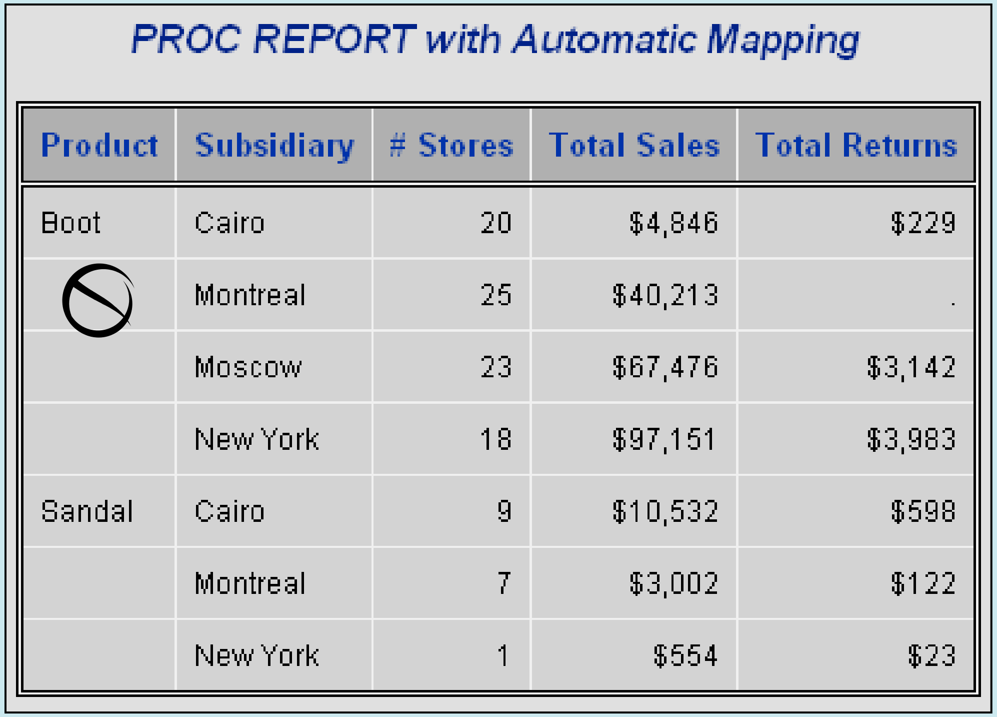
See also Delwiche and Slaughter
The Little SAS® Book: Fourth Edition

Map new Styles to SAS Procedures

PROC REPORT: **Automatic Mapping**

```
ods ... style=STYLES.Default;
 proc report data=shoesMiss nowindows;
  column Product Subsidiary nstores Sales Returns;
  Define Product     / ...;
  Define Subsidiary / ...;
  Define nStores     / ...;
  Define Sales       / ...;
  Define Returns     / ...;
 run;
ods HTML close;
```

RowHeader



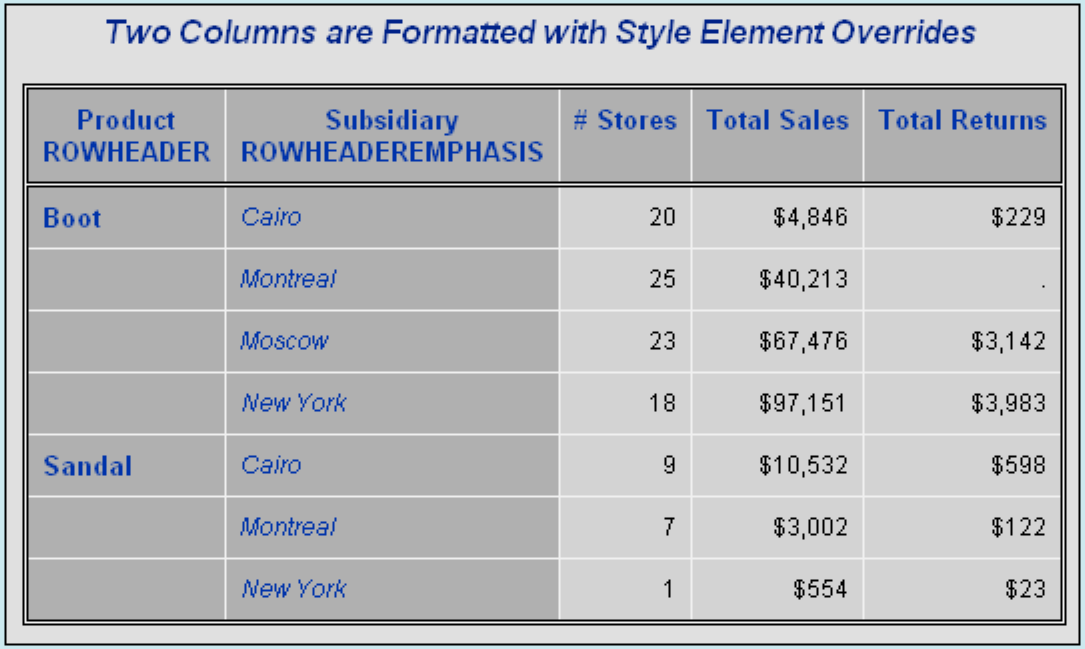
PROC REPORT with Automatic Mapping

Product	Subsidiary	# Stores	Total Sales	Total Returns
Boot	Cairo	20	$4,846	$229
⦸	Montreal	25	$40,213	.
	Moscow	23	$67,476	$3,142
	New York	18	$97,151	$3,983
Sandal	Cairo	9	$10,532	$598
	Montreal	7	$3,002	$122
	New York	1	$554	$23

Map new Styles to SAS Procedures

PROC REPORT: **Style Element Overrides**

```
ods ... style=STYLES.Default;
proc report data=shoesMiss nowindows;
 column Product Subsidiary nstores Sales Returns;
 Define Product /
   STYLE(COLUMN)=
    ROWHEADER;
 Define Subsidiary /
   STYLE(COLUMN)=
    ROWHEADEREMPHASIS;
 ...;
run;
ods HTML close;
```



Two Columns are Formatted with Style Element Overrides

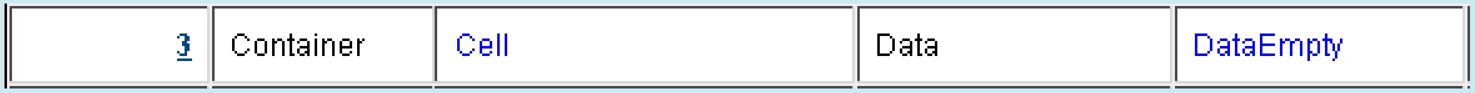
Product ROWHEADER	Subsidiary ROWHEADEREMPHASIS	# Stores	Total Sales	Total Returns
Boot	*Cairo*	20	$4,846	$229
	Montreal	25	$40,213	.
	Moscow	23	$67,476	$3,142
	New York	18	$97,151	$3,983
Sandal	*Cairo*	9	$10,532	$598
	Montreal	7	$3,002	$122
	New York	1	$554	$23

See Delgobbo
Traffic Lighting Your Multi-Sheet Microsoft Excel Workbooks the Easy Way with SAS®

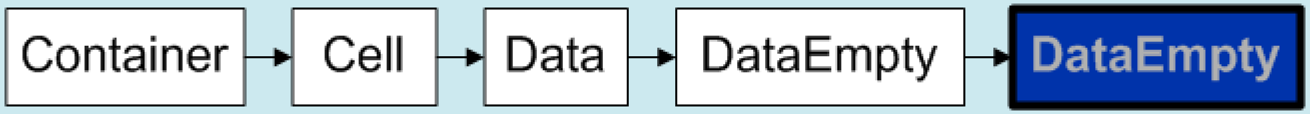
Map new Styles to SAS Procedures

PROC REPORT: *User-Named* Style Element Overrides

```
proc template;
 define style styles.AbbrOvRide;
   style Hdr from Header;  ...
   style RowHdrEmphFx from RowHeaderEmphasisFixed;
   class dataEmpty /
     backgroundcolor=colors('headerfg')
     color=colors('headerbg')
     ...;
 end;
run;
```



3	Container	Cell	Data	DataEmpty



$$\begin{tikzcd}\text{Container} \arrow[r] & \text{Cell} \arrow[r] & \text{Data} \arrow[r] & \text{DataEmpty} \arrow[r] & \boxed{\text{DataEmpty}}\end{tikzcd}$$

Map new Styles to SAS Procedures

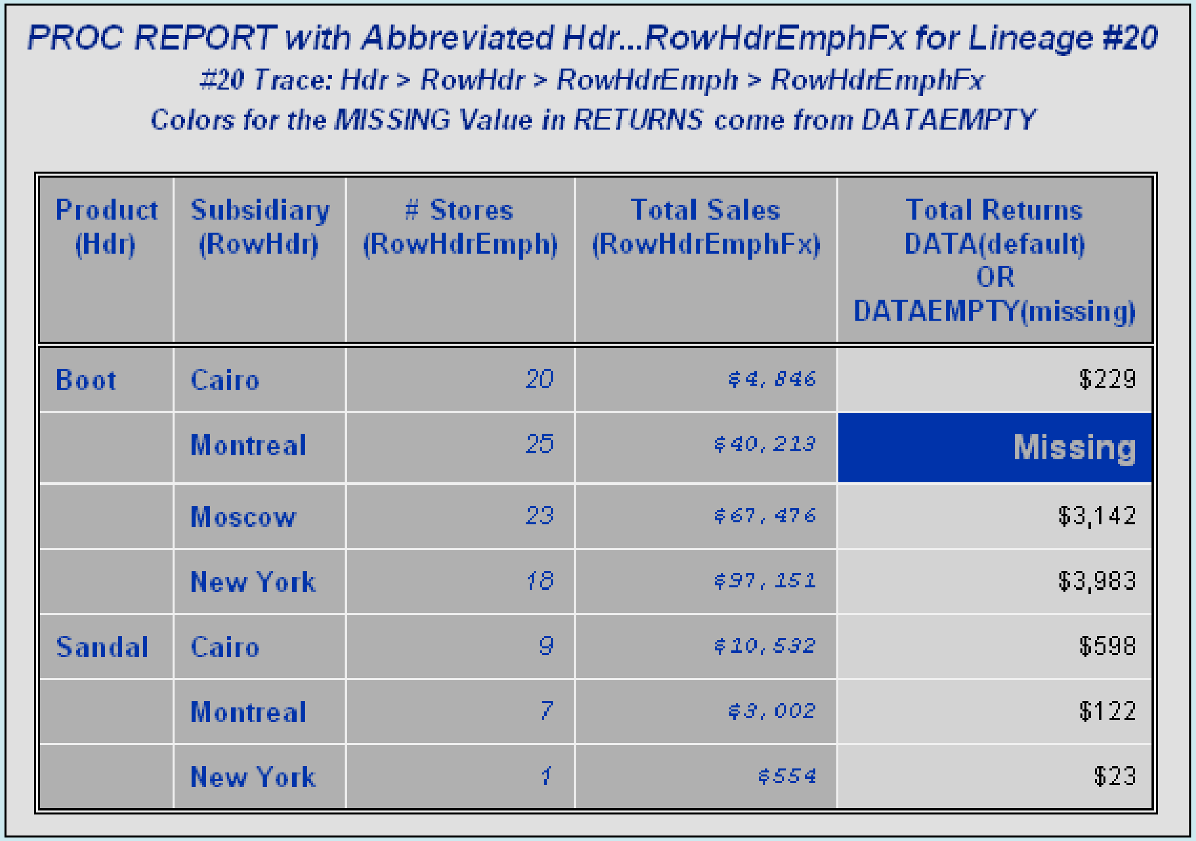
PROC REPORT ***User-Named* Style Element Overrides**

```
ods html ... style = AbbrOvride;
 proc report ...;
 column Product Subsidiary nstores Sales Returns;
 Define Product / STYLE(COLUMN) = Hdr;
 ...
 Define Sales   / STYLE(COLUMN) = RowHdrEmphFx;
*---------------------------------------------------------;
 Define Returns / ...;

 compute Returns;
  if Returns eq . then
  call define ('_c5_', "style", "style = DataEmpty");
 endcomp;
run;
ods HTML close;
```

Map new Styles to SAS Procedures

PROC REPORT *User-Named* Style Element Overrides



PROC REPORT with Abbreviated Hdr...RowHdrEmphFx for Lineage #20

#20 Trace: Hdr > RowHdr > RowHdrEmph > RowHdrEmphFx

Colors for the *MISSING* Value in *RETURNS* come from *DATAEMPTY*

Product (Hdr)	Subsidiary (RowHdr)	# Stores (RowHdrEmph)	Total Sales (RowHdrEmphFx)	Total Returns DATA(default) OR DATAEMPTY(missing)
Boot	Cairo	20	$4,846	$229
	Montreal	25	$40,213	Missing
	Moscow	23	$67,476	$3,142
	New York	18	$97,151	$3,983
Sandal	Cairo	9	$10,532	$598
	Montreal	7	$3,002	$122
	New York	1	$554	$23

20	Container	Cell	HeadersAndFooters	Header	RowHeader	RowHeaderEmphasis	RowHeaderEmphasisFixed

$$\begin{tikzcd}
20 & \boxed{\text{Container}} \arrow[r] & \boxed{\text{Cell}} \arrow[r] & \boxed{\text{HeadersAndFooters}} \arrow[r] & \boxed{\text{Hdr}} \arrow[r] & \boxed{\text{RowHdr}} \arrow[r] & \boxed{\text{RowHdrEmph}} \arrow[r] & \boxed{\text{RowHdrEmphFx}} \\
 & & & & ● & ● & ● & ●
\end{tikzcd}$$

Summary and Conclusions

Prove that Lineage plays a Central Role in ODS Inheritance:

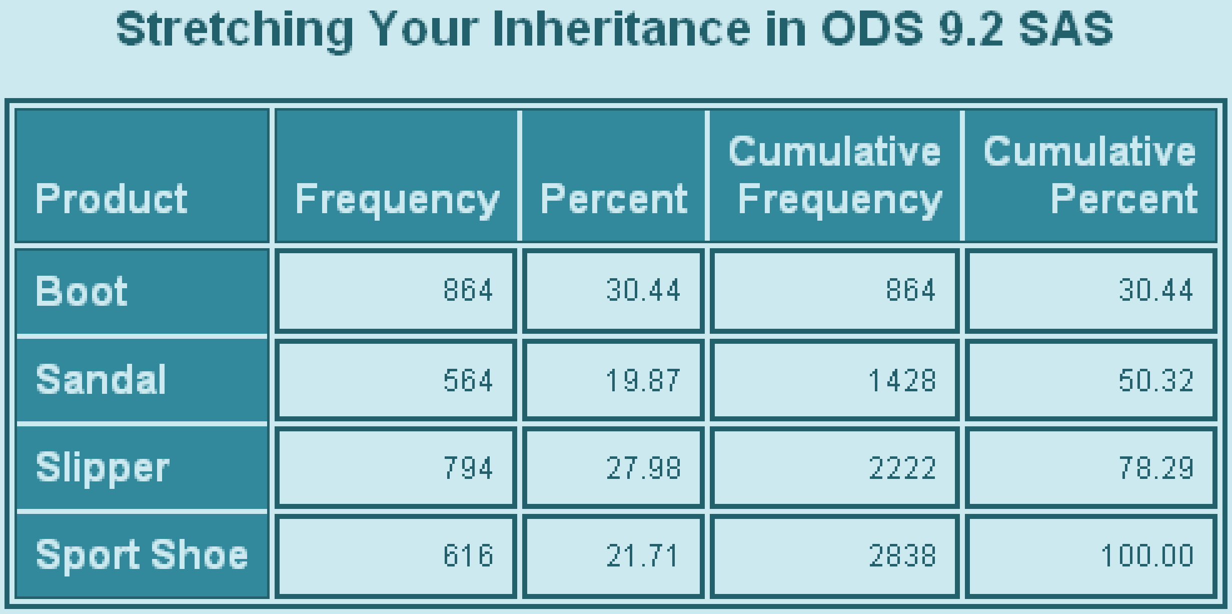
- When **terms** are **defined**
- When the **lineage tracer** is described
- When **predefined lineages** from the lineage tracer are used in **CLASS statements** to define new styles in ODS
- When **new lineages** are defined with **STYLE** and **STYLE ... FROM** statements in new template definitions

Also show how STYLE attribute settings are mapped to SAS PROC output

What Else is Available

- The ZIP file contains the following HTML files:
  - **Container92Lineages** aka the **lineage tracer**
  - **AttributeDescriptor92** for attributes, the "what" of inheritance
  - **Normal92Lineages** displays a single page of lineage definitions for the NORMAL style template in sashelp.Tmplmst
  - **Style92TemplateLineagesHighlighted** describes inheritance at the style template level for 53 styles in Sashelp.Tmplmst
- See the paper to find out what else is in the ZIP file
- The presentation plus an updated version of the paper will be posted on http://www.screencast.com/users/PerryWatts

Contact Information



Stretching Your Inheritance in ODS 9.2 SAS

Product	Frequency	Percent	Cumulative Frequency	Cumulative Percent
Boot	864	30.44	864	30.44
Sandal	564	19.87	1428	50.32
Slipper	794	27.98	2222	78.29
Sport Shoe	616	21.71	2838	100.00

perryWatts@comcast.net