Db2 Web Query for i Report Styling

Contents

Preface

This content describes styling and the different types of content and development methods you can use in Db2 Web Query to customize your reports and business intelligence dashboards.

How This Manual Is Organized

This manual includes the following chapters:

	Chapter/Appendix	Contents
1	Introducing Styling	Provides an introduction to styling and the different types of content and development methods.
2	Customizing Reports and Dashboards With Images and Style Sheets	Describes how to customize your reports and business intelligence dashboards with an attractive image or company logo.
3	Introducing Formatting and Style Sheets	Provides an introduction to formatting and style sheets.
4	Controlling Report Formatting	Describes how you can control how reports are formatted including generating internal cascading style sheets, conditional formatting, setting measurement units, and controlling the display of empty reports.
5	Identifying a Report Component in a Web Query StyleSheet	Describes how to identify a report component using StyleSheet declarations.
6	Using an External Cascading Style Sheet	Describes how you can increase your formatting options by using external cascading style sheets.
7	Laying Out the Report Page	Describes basic report page layout including page size, orientation, page numbers, margins, images, grids, borders, page-breaks, and mailing labels.
8	Using Headings, Footings, Titles, and Labels	Describes how to add and format headings, footings, titles, and labels to add context to your report.
9	Formatting Report Data	Describes formatting and positioning text in a report including font style, size, and color.

	Chapter/Appendix	Contents
10	Choosing a Display Format	Describes the display formats available for viewing reports on the screen, including PDF, XLSX, and PPTX.
11	Linking a Report to Other Resources	Describes how to use StyleSheets to link reports to other reports, URLs, and JavaScript functions.

Documentation Conventions

The following table lists and describes the conventions that apply in this manual.

Convention	Description
`THIS TYPEFACE` or `this typeface`	Denotes syntax that you must enter exactly as shown.
this typeface	Represents a placeholder (or variable), a cross-reference, or an important term.
underscore	Indicates a default setting.
this typeface	Highlights a file name or command. It may also indicate a button, menu item, or dialog box option you can click or select.
Key + Key	Indicates keys that you must press simultaneously.
`{ }`	Indicates two or three choices; type one of them, not the braces.
`[ ]`	Indicates a group of optional parameters. None is required, but you may select one of them. Type only the parameter in the brackets, not the brackets.
`|`	Separates mutually exclusive choices in syntax. Type one of them, not the symbol.
`...`	Indicates that you can enter a parameter multiple times. Type only the parameter, not the ellipsis points (...).

Convention	Description
. . .	Indicates that there are (or could be) intervening or additional commands.

Chapter 1

Introducing Styling

There are many ways to style and format content using the various tools within Db2 Web Query. Content can be a report, chart, document, dashboard, or an HTML page. Using the report styling features within the development tools is always recommended. However, if the GUI does not contain the feature for the exact styling you desire, you can use a style sheet, which is covered extensively starting in Chapter 3, *Introducing Formatting and Style Sheets* on page 27.

In this chapter:

Aspects of Report Styling

This document will cover all aspects of report styling and will answer some of the following commonly asked questions:

❑ What is the best practice for uploading style sheets and uploading images?

❑ Where do I store images that I may want to use in my content?

❑ What are the best practices for location of images (for example, grouping the images separately so as not to confuse them with reports and schedules in the tree)?

❑ What are the best practices for globally shared style sheets and images?

❑ What folder roles and permissions are needed for the tasks of uploading, embedding into a report, or running a report, with a style sheet or image included (Dev, DBA, Web Query Administrator, and so on)?

InfoAssist is the content development tool included in all Web Query editions. The Developer Workbench is an optional feature that provides an HTML canvas used to create HTML pages.

There are many report formatting and styling features within each of these respective tools.

The following table summarizes the different types of content and whether or not images, for example, a corporate logo, can be added to it.

Type of Content	**Add Image**	**Source Location of Image**
InfoAssist Report	Yes, using style sheet syntax.	Repository folder
InfoAssist Chart	No	N/A
InfoAssist Document	Yes	Reporting Server application folder. (The Choose Image dialog box only sees the Reporting Server). The image (*.jpg or *.gif) must reside in the Reporting Server APPPATH).
InfoAssist Dashboard (Active Document)	Yes	Reporting Server application folder
Developer Workbench HTML Page	Yes	Repository folder or Reporting Server application folder

Chapter 2

Customizing Reports and Dashboards With Images and Style Sheets

Web Query is a BI analytics tool, rich in features for visually representing your critical business data. You can use images to further exploit the presentation of that data for all your business needs, executive dashboards, business updates, and even marketing material.

In Db2 Web Query, if you want to customize your reports and business intelligence dashboards with an attractive image or company logo, there are several ways to accomplish this. You can customize a single report, document, or HTML dashboard, or create a default style sheet for use as a company-wide theme. The following content explains the options for adding an image to your content.

In this chapter:

Uploading an Image

The first step when incorporating a logo into a report is to upload the image into Web Query.

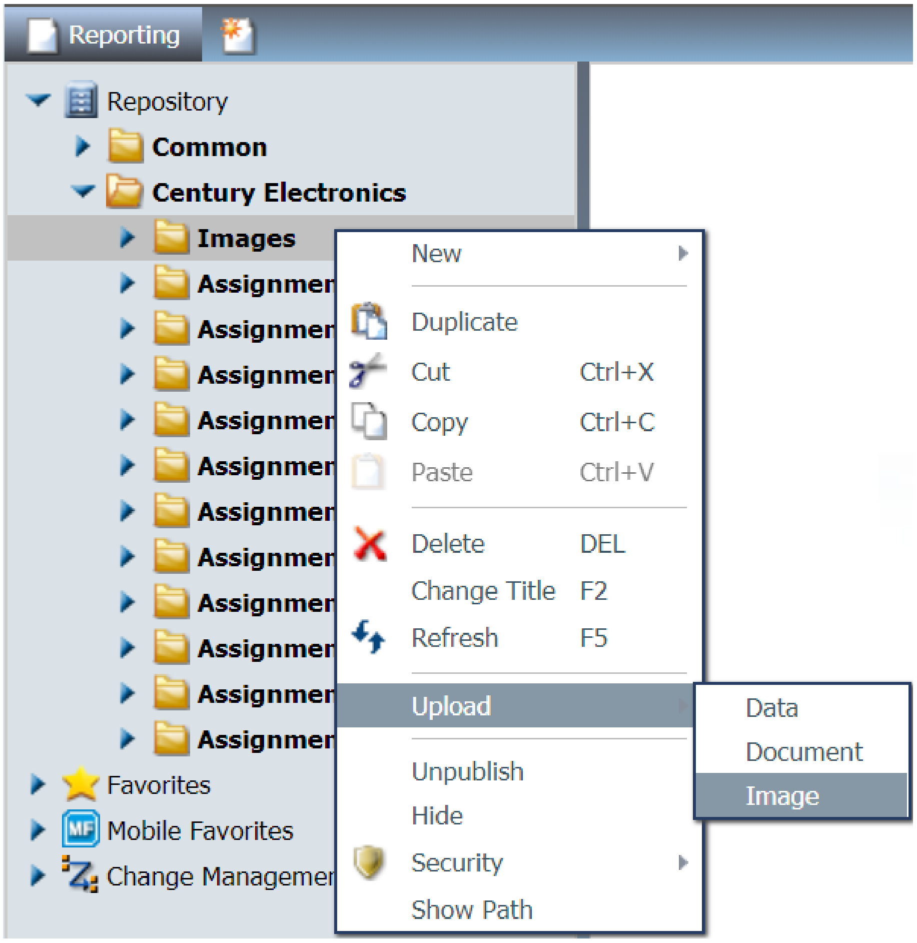
❑ For reports or HTML dashboards, the image can reside in either a repository folder or a Reporting Server application folder.

❑ For documents or reports scheduled through Report Broker, the image must reside in a Reporting Server application folder.

Procedure: How to Upload an Image to a Repository Folder

1. Sign in to Web Query as a developer or an administrator.

2. Right-click the folder in the BI Portal, select *Upload*, and then select *Image*, as shown in the following image.



3. Click *Browse* to select the image to upload.

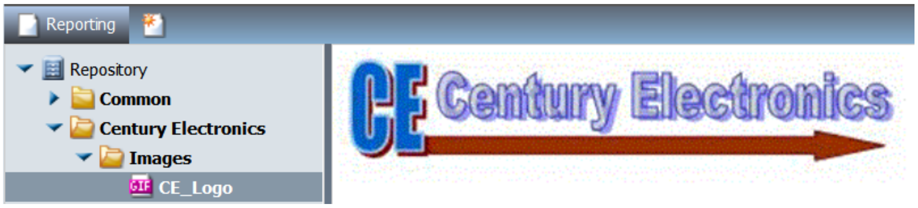
   The File Upload dialog box prompts you for the location of the image and stores the image in the repository, making it visible in the portal tree. A best practice is to organize images under a subfolder to avoid confusing them with reports and schedules. Both .gif and .jpg formats are supported.

4. Once you have selected a file, click *Open*.

5. Click *Upload*.

   The image now appears in the selected folder.

The object type for an image is easily identified by an icon. The following image shows the icon for the Web Query sample business, Century Electronics.



Uploading an Image to a Reporting Server Application Folder

There are two ways to upload an image to a Reporting Server application folder.

- ❑ A Web Query administrator can upload the image to the repository folder and then copy it to the Reporting Server application folder all within the BI Portal. For more information, see *How to Copy an Image From the Repository Folder to a Reporting Server Application Folder* on page 17.

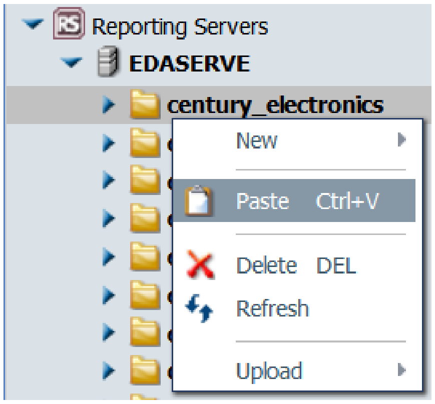
- ❑ A Web Query administrator or a user who is a member of a folder-DBA group can upload an image directly to the Reporting Server application folder. For more information, see *How to Upload an Image Directly to the Reporting Server Application Folder* on page 18.

Procedure: How to Copy an Image From the Repository Folder to a Reporting Server Application Folder

1. Upload an image to a repository folder, as described in *How to Upload an Image to a Repository Folder* on page 15.

2. Right-click the image in the portal tree and select *Copy*.

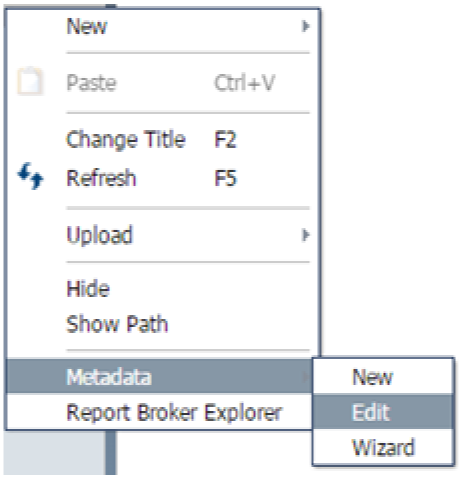
3. Expand *Reporting Servers*, right-click the folder where you want to paste it, and select *Paste*, as shown in the following image.



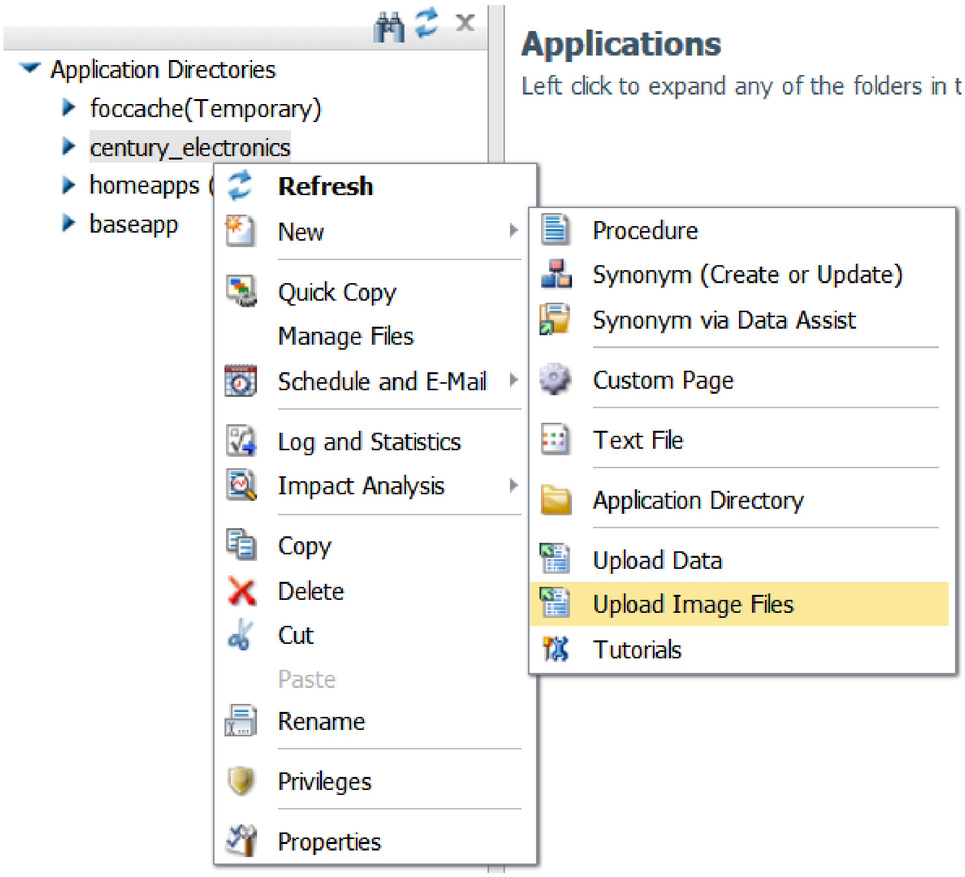
Note: If you want an image to be globally accessible to all report authors, independent of the folders to which they have access, place the image in the Common top-level folder and the baseapp Reporting Server folder. Keep in mind that the concept of a private or public object exists for images also. For example, you need to publish the image (right-click option) if you want others to be able to use it.

Procedure: How to Upload an Image Directly to the Reporting Server Application Folder

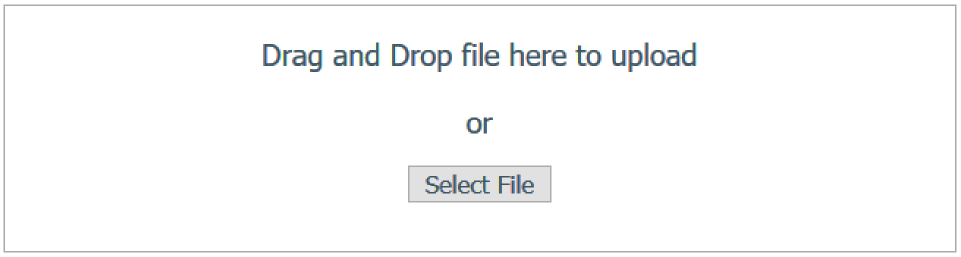
1. Right-click the folder, select *Metadata*, and then select *Edit*, as shown in the following image.



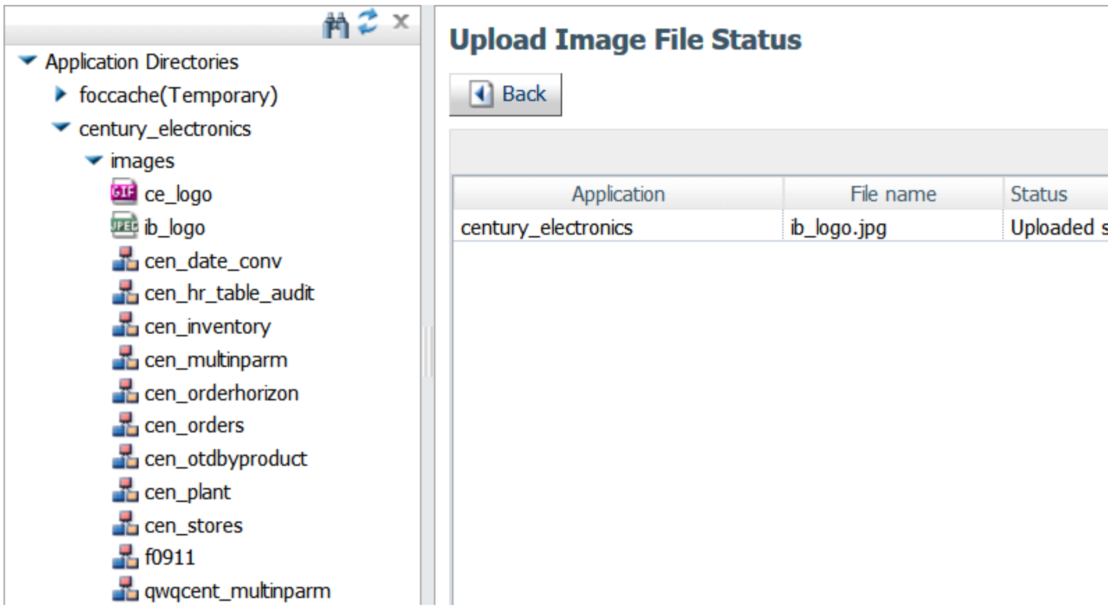
2. Right-click the application folder, select *New*, and then select *Upload Image Files*, as shown in the following image.



3. Click *Select File* or Drag and drop the image you want to upload, using the Upload dialog box, as shown in the following image.



Once the image is uploaded, you will see the image in the application folder, as shown in the following image.



Application	File name	Status
century_electronics	ib_logo.jpg	Uploaded s

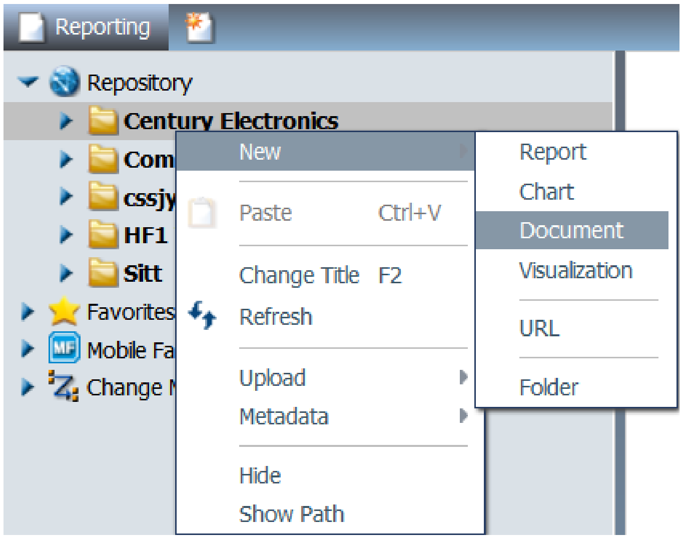
Once uploaded, the image is ready to go! Now, let us look at how you can insert the image into an InfoAssist Compound Document or an HTML dashboard (with Developer Workbench), or edit it into an individual report or custom style sheet. Note that editing a procedure (fex), which is the term for a report, is generally not recommended. However, this edit is fairly simple with minimal risk.

Inserting an Image Into an InfoAssist Document or HTML Dashboard

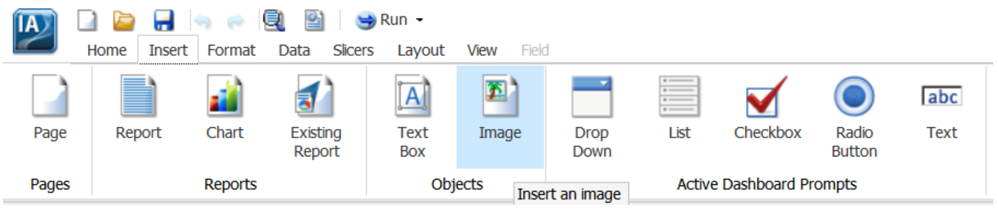
The easiest and preferred method for adding an image to a single report or chart is to create an InfoAssist document.

Procedure: How to Insert an Image Into an InfoAssist Document

1. From the BI portal, edit an existing document or create a new document. To create a new document, select a folder, select *New*, and then select *Document*, as shown in the following image.

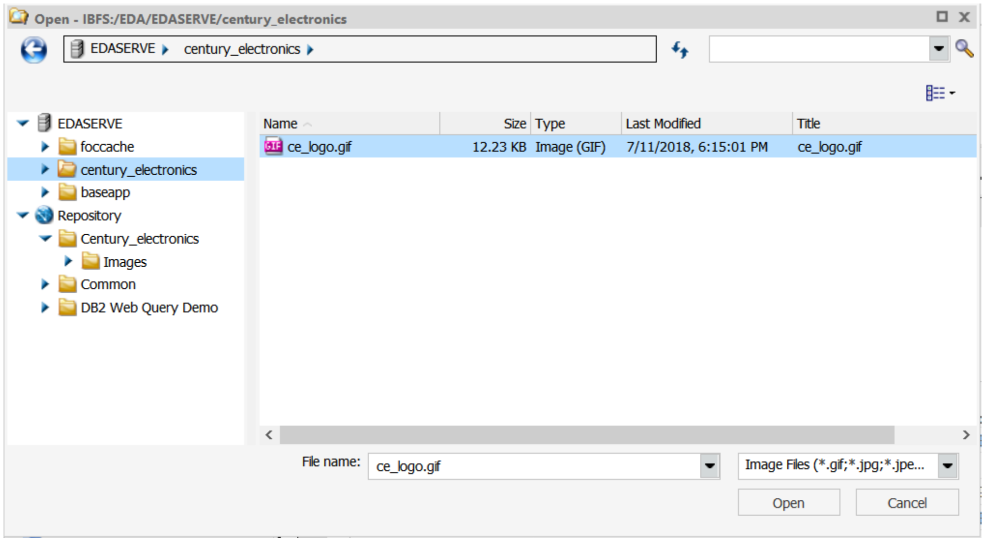


2. To add an image, select the *Insert* tab on the ribbon and then click the *Image* icon, as shown in the following image.



A dialog box prompts you for the image you want to use in the document.

3. Navigate to the image and select it, as shown in the following image.



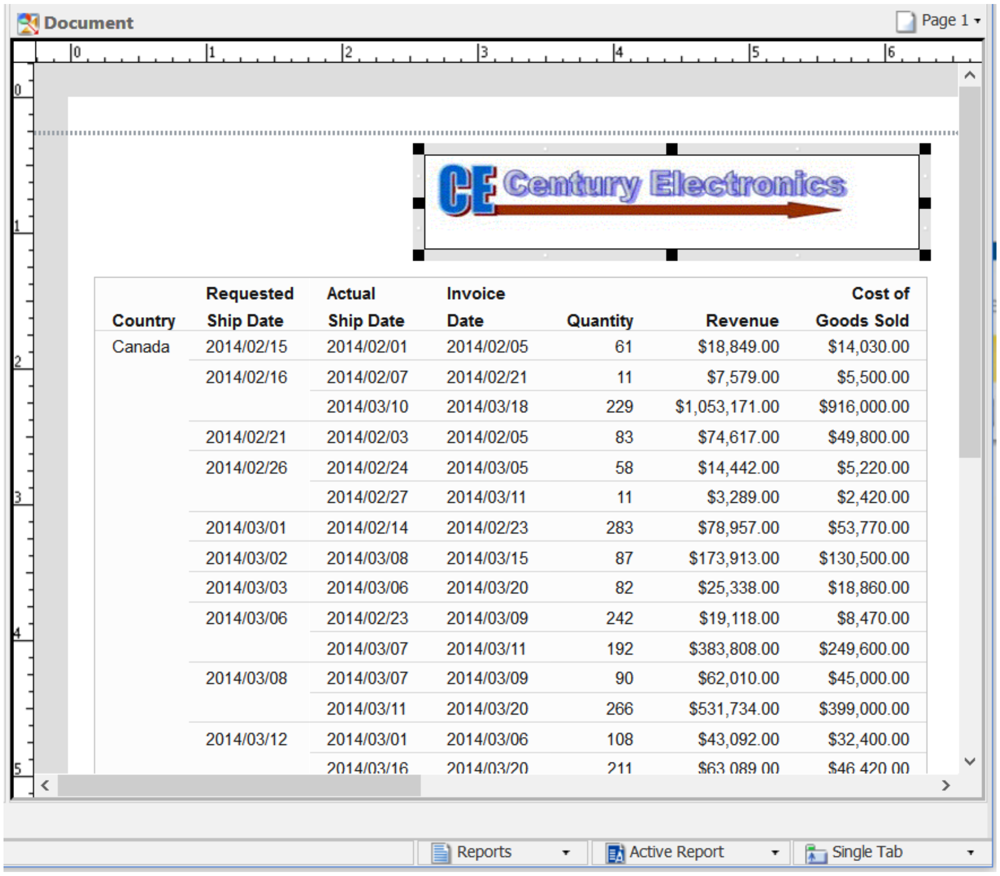
Name	Size	Type	Last Modified	Title
ce_logo.gif	12.23 KB	Image (GIF)	7/11/2018, 6:15:01 PM	ce_logo.gif

File name: ce_logo.gif

Image Files (*.gif;*.jpg;*.jpe...

4. Click *OK*.

The image appears on a ruled canvas, and you can easily resize and reposition it by dragging it with your mouse. The following image shows the image being centered as a letterhead above the report.



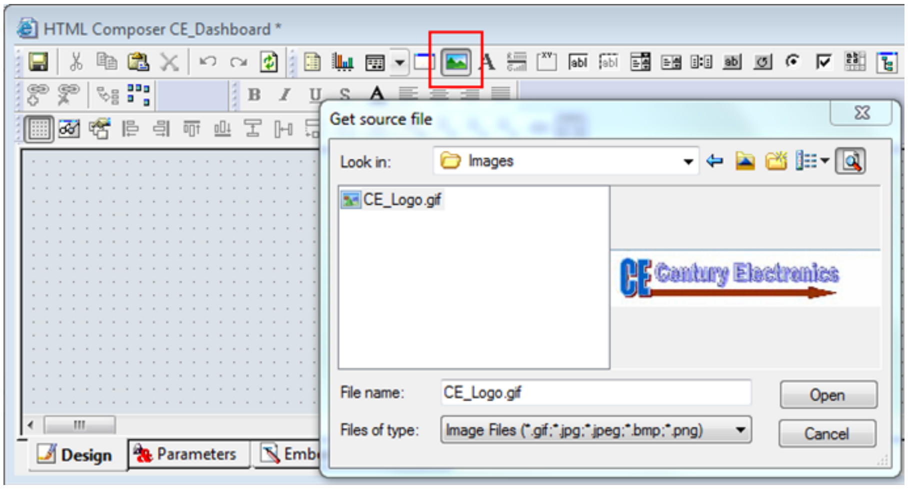
Country	Requested Ship Date	Actual Ship Date	Invoice Date	Quantity	Revenue	Cost of Goods Sold
Canada	2014/02/15	2014/02/01	2014/02/05	61	$18,849.00	$14,030.00
	2014/02/16	2014/02/07	2014/02/21	11	$7,579.00	$5,500.00
		2014/03/10	2014/03/18	229	$1,053,171.00	$916,000.00
	2014/02/21	2014/02/03	2014/02/05	83	$74,617.00	$49,800.00
	2014/02/26	2014/02/24	2014/03/05	58	$14,442.00	$5,220.00
		2014/02/27	2014/03/11	11	$3,289.00	$2,420.00
	2014/03/01	2014/02/14	2014/02/23	283	$78,957.00	$53,770.00
	2014/03/02	2014/03/08	2014/03/15	87	$173,913.00	$130,500.00
	2014/03/03	2014/03/06	2014/03/20	82	$25,338.00	$18,860.00
	2014/03/06	2014/02/23	2014/03/09	242	$19,118.00	$8,470.00
		2014/03/07	2014/03/11	192	$383,808.00	$249,600.00
	2014/03/08	2014/03/07	2014/03/09	90	$62,010.00	$45,000.00
		2014/03/11	2014/03/20	266	$531,734.00	$399,000.00
	2014/03/12	2014/03/01	2014/03/06	108	$43,092.00	$32,400.00
		2014/03/16	2014/03/20	211	$63,089.00	$46,420.00

Procedure: How to Insert an Image in an HTML Dashboard

The InfoAssist tool supports both documents and dashboards. Use a similar method to insert an image into an InfoAssist dashboard.

If you prefer to create dashboards using the HTML canvas, this is also an option available to you.

1. To add an image to an HTML dashboard, right-click the HTML document and select *Edit in Composer*. You can then click the *Image* icon on the ribbon, as shown in the following image.



2. Locate and open the image you want, and then position it onto your dashboard. The HTML canvas additionally supports .jpeg, .bmp, and .png formats.

Inserting an Image Directly Into a Report

An image can be directly inserted into a report using the text editor from a Web Query administrator user ID. Note again that you need to be careful when editing a procedure (fex) with a text editor. You should make a copy of your report first before making edits.

Procedure: How to Insert an Image Directly Into a Report

1. To edit the report, right-click the report, select *Edit With...*, and then select *Text Editor*. You can do this from either the BI Portal or Developer Workbench.

2. Scroll down in the report source, referred to as a *fex*, and update the styling section to add the location (path) of the image. The following is example syntax to insert an image into a fex.

```
ON TABLE SET STYLESHEET *

INCLUDE=IFBS:/FILE/IBI_HTML_DIR/javascript/intl/EN/
   ENIADefault_combine.sty ,$
```

```
TYPE=REPORT, TITLETEXT=&WF_TITLE.QUOTEDSTRING,
   SUMMARY=&WF_SUMMARY.QUOTEDSTRING, SQUEEZE=OFF, $

TYPE=REPORT, IMAGE=CE_Logo.gif, POSITION=(0.5000 0.25),
   SIZE=(3.0625 0.50), $
```

To find the location of an image, right-click the image in the portal, and select *Show Path*. If the image is stored in baseapp or another application path available to the Reporting Server, then you only need to specify the name of the image, for example, CE_Logo.gif.

An image is inserted at original size, by default. Size and position parameters are therefore optional. The syntax is SIZE=(*height width*) and POSITION=(*x y*), where *x* and *y* are distances from the margins. Measurements are in terms of units defined in the report.

Note: This method of directly inserting an image into a report is included here for completeness, but the recommended method is to create an InfoAssist document. Editing is an option for reports, not charts. Be aware that if you reopen the report with InfoAssist and save it, the image may be lost and you will have to re-insert it.

Inserting an Image Into a Custom Style Sheet

To standardize your logo across multiple reports or multiple developers, or to create a theme for your company, you can simplify the task with a style sheet. Style sheets improve the appearance of your report through use of images, color, and other detail.

It is easier to start with one of the existing style sheets included with Db2 Web Query, rather than try to create one from scratch. Style sheets packaged with the Web Query product have a file extension of .sty and are located in the /qibm/proddata/qwebqry/base80/ibi_html/javaassist/intl/EN/combine_templates IFS directory. Take a look at some of the style sheets to see if one is close to what you are looking for by applying them to an existing report in InfoAssist. Do not modify an included style sheet or copy a style sheet into the Web Query proddata directory mentioned above, because it will be overlaid when a PTF is applied. Instead, copy the style sheet to your PC and modify it there.

Procedure: How to Insert an Image Into a Custom Style Sheet and Apply it to a Report

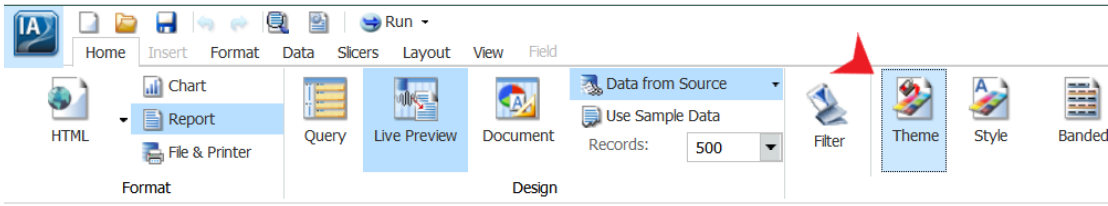
1. To include an image in a style sheet, simply add the same IMAGE statement as described in *How to Insert an Image Directly Into a Report* on page 24. This edit should be made to a style sheet on your local PC.

   Tip: Create an InfoAssist document to initially size and position the image, and then copy the IMAGE statement from that procedure (fex) into the style sheet.

2. To upload a style sheet, right-click a folder in the BI portal, select *Upload*, and then select *Document*. Click the *Browse* button and locate the style sheet from step 1.

Note: The same considerations for organizing, accessing, and publishing an image are applicable to a style sheet.

3. To apply a style sheet to an InfoAssist report or dashboard, click the *Theme* icon on the Home tab of the InfoAssist ribbon, as shown in the following image. You can choose one of the many style sheets provided with the Web Query product, or navigate the tree to find and select the one you created with your image.



4. Save the report.

Style sheets enhance the layout or presentation of a page. There is a distinction between document formats and an HTML report. Styling can be different between them because of positioned versus table-based approaches. There may be a need to create separate style sheets if you are not satisfied with the way your image renders in each format.

Chapter 3

Introducing Formatting and Style Sheets

To create an effective report, you need to account for:

- **Content.** The data in your report.
- **Formatting.** How to present that content to a reader in a way that achieves maximum impact.

There are many formatting options that you can use to give your report a professional appearance and affect how people read and interpret it. For example, you can control:

- The appearance of the data, which you can change to emphasize important values.
- The headings, footings, and other text with which you "frame" the data to give it context.
- The layout of the report on the page or screen, which you can adjust for different display environments and for audiences with different vision needs.

The following topics provide an overview of formatting and style sheets.

In this chapter:

What Kinds of Formatting Can I Do?

There are many kinds of formatting that you can apply to a report:

- **The appearance of the report data,** such as its font, size, style (italic, bold, or underlined), color (of the foreground and the background), position, and justification. You can also draw boxes or lines around data. You can use these properties to emphasize critical values and to draw attention to important data relationships.

  You can also select which character to use to mark decimal position, using either a period (.) or a comma (,), to match the convention of the country in which the report will be read. You can even choose which character to use to represent a null value and missing data.

- **Providing context for data by "framing" it**, for example, with headings, footings, subtotals, recaps, and customized column titles. You can include fields and images within headings and footings. As with data, you can specify a heading, footing, subtotal, subhead, subfoot, recap, and column title font, size, style, color, position, and justification, as well as enclose it within boxes or lines. You can use these framing devices to explain the context of the data and to engage the interest of the reader.

- **Laying out the report** on the screen or printed page. You can choose the report margins, where to place headings and footings, where to place background images (watermarks), and how to arrange the report columns (adjusting the space around and between columns, adjusting column width and column order, and even stacking one column above another to reduce report width). You can visually distinguish between different columns, rows, or sort groups using color and lines. If you wish, you can draw borders around parts of a report or around the entire report.

  You can lay out the report to optimize it for different display environments such as screens of different sizes and resolutions, and printed pages of different sizes. You can create multiple report panes on a single page to print labels. You can even combine several reports into a single file to display or print them as a group.

- **Conditionally formatting** a report based on the report data. You specify a condition that, at run time, is automatically evaluated for each instance of the report component you specify, such as each value of a sort column. The formatting option is applied to each instance of the report component for which the condition is true. For example, in a sales report, you can draw attention to sales staff who exceeded quota by making their names bold and using a different color.

- **Choosing a display format, such as HTML (the default),** PDF (Adobe Acrobat Portable Document Format), Excel, or PostScript, to suit the viewing and processing needs of the readers.

What is a Web Query StyleSheet?

A Web Query StyleSheet (referred to as a *StyleSheet* in this chapter) enables you to format and produce attractive reports that highlight key information. With StyleSheets, you can specify various characteristics of your report and format report components individually.

You can use a StyleSheet to:

- Format report components individually.
- Incorporate graphical elements.
- Define dynamic hyperlinks.
- Format data that meets specified conditions.

You can also use external cascading style sheets and you can enable internal cascading style sheets for HTML reports. For details, see *Using an External Cascading Style Sheet* on page 101 and *Controlling Report Formatting* on page 41.

Unless otherwise noted, all StyleSheet references in this chapter refer to Web Query StyleSheets.

How to Choose a Type of Style Sheet

You can choose between two types of style sheets to format a report:

- **Web Query StyleSheets**, the native Web Query style sheet language. These provide you with the flexibility to format reports in many display formats, including HTML, PDF, and Excel.

  If you are generating a report in HTML format, you can boost its performance, and increase the number of formatting options available to it, by having the StyleSheet dynamically generate an *internal* cascading style sheet (CSS). CSS is the standard style sheet language designed for HTML documents. The *internal* CSS generated by Web Query is internal to the report output, instead of being saved as a separate file. For more information about generating an internal cascading style sheet, see *Generating an Internal Cascading Style Sheet for HTML Reports* on page 41.

- **External cascading style sheets**, the standard style sheet language designed for HTML documents. You can apply an *external* cascading style sheet to any Web Query report in HTML format. An *external* cascading style sheet is one that is saved as a separate file, and so is *external* to the document.

How do you choose between the two types of style sheets? Consider choosing:

- ❏ **A Web Query StyleSheet** if:

  You want to display a report in different display formats, such as PDF and Excel. StyleSheets are supported for many kinds of display formats, but cascading style sheets work for reports in HTML format only.

- ❏ **An external cascading style sheet** for any of the following reasons:

  Your enterprise already uses cascading style sheets to format HTML documents, and it wants reports to conform to these same presentation guidelines.

  You want to apply the same formatting to other kinds of HTML documents in your enterprise.

Creating and Applying a StyleSheet File

You can create a StyleSheet as a separate file and apply it to as many reports as you wish. A StyleSheet file contains only declarations and optional comments. For information about StyleSheet declarations, see *General Web Query StyleSheet Syntax* on page 33.

As an alternative to creating a new StyleSheet file, you can use one of the sample StyleSheet files provided with Web Query as a template.

The following syntax is generated by InfoAssist when adding a StyleSheet to a report, chart, or document.

```
ON TABLE SET STYLE[SHEET] *
INCLUDE=IBFS:/stylesheet_location/custom_stylesheet.sty,$
```

where:

`SHEET`

Can be omitted to make the command shorter, and has no effect on its behavior.

`stylesheet_location`

Is the location of the StyleSheet file.

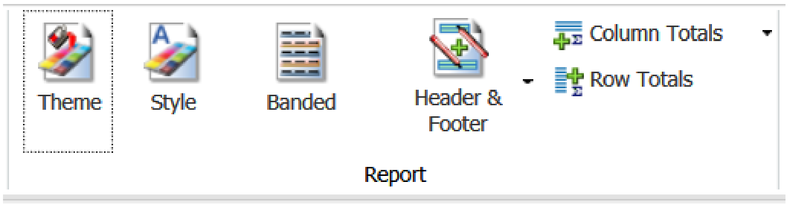
`custom_stylesheet`

Is the name of the custom StyleSheet file.

Procedure: How to Apply a StyleSheet File to a Report

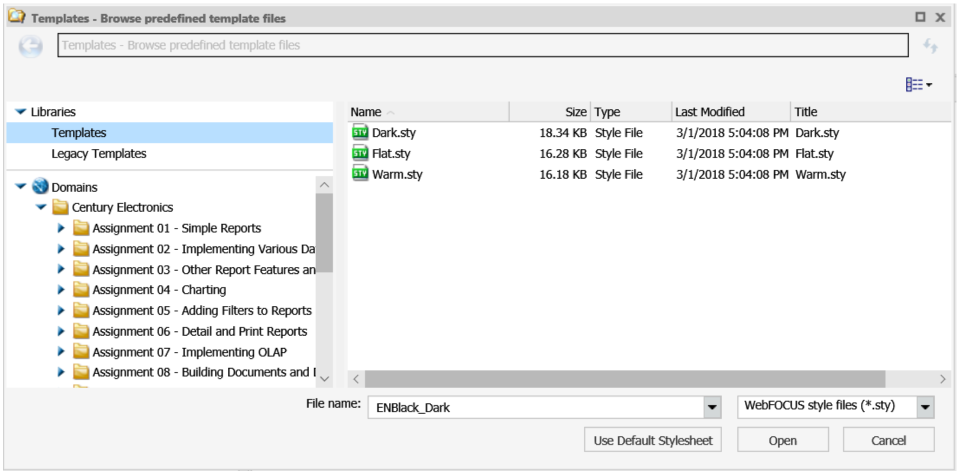
InfoAssist provides an option to incorporate a style sheet into a report.

1. On the *Home* tab, in the *Report* group, click *Theme*, as shown in the following image.



The Templates dialog box opens, where you can select a StyleSheet.

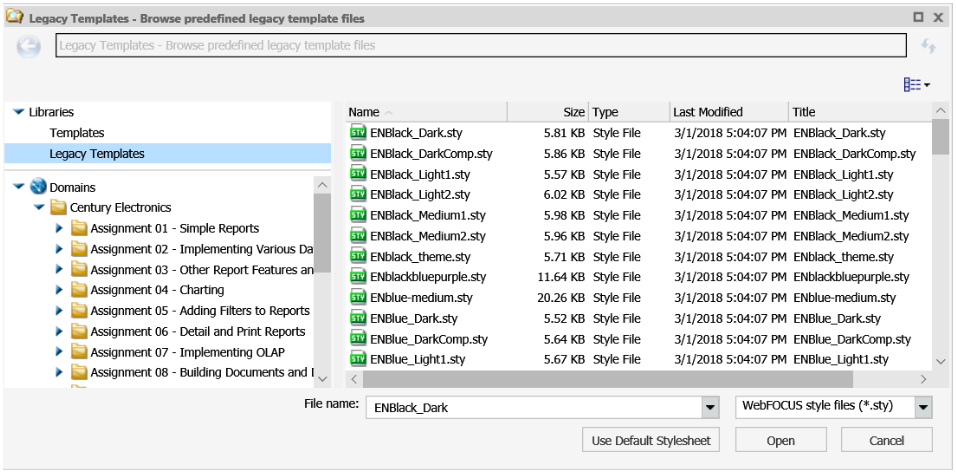
2. Select *Templates*, under Libraries, to display a list of current templates provided with Web Query, as shown in the following image.


Templates - Browse predefined template files
Templates - Browse predefined template files
Libraries
Templates
Legacy Templates
Domains
Century Electronics
Assignment 01 - Simple Reports
Assignment 02 - Implementing Various Da
Assignment 03 - Other Report Features an
Assignment 04 - Charting
Assignment 05 - Adding Filters to Reports
Assignment 06 - Detail and Print Reports
Assignment 07 - Implementing OLAP
Assignment 08 - Building Documents and I
File name:
WebFOCUS style files (*.sty)
Use Default Stylesheet
Open
Cancel

Name	Size	Type	Last Modified	Title
Dark.sty	18.34 KB	Style File	3/1/2018 5:04:08 PM	Dark.sty
Flat.sty	16.28 KB	Style File	3/1/2018 5:04:08 PM	Flat.sty
Warm.sty	16.18 KB	Style File	3/1/2018 5:04:08 PM	Warm.sty

File name: ENBlack_Dark

3. Select *Legacy Templates*, under Libraries, to display a list of templates provided in earlier releases of Web Query, as shown in the following image.

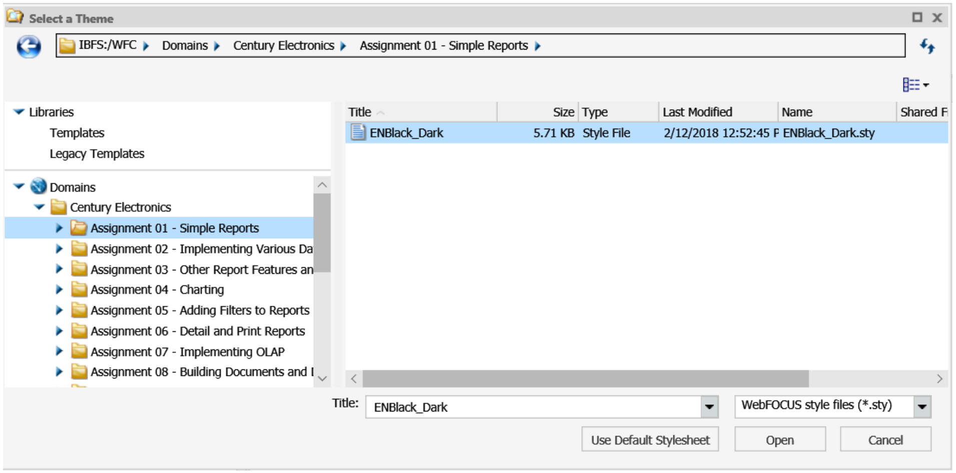


Legacy Templates - Browse predefined legacy template files

Name	Size	Type	Last Modified	Title
ENBlack_Dark.sty	5.81 KB	Style File	3/1/2018 5:04:07 PM	ENBlack_Dark.sty
ENBlack_DarkComp.sty	5.86 KB	Style File	3/1/2018 5:04:07 PM	ENBlack_DarkComp.sty
ENBlack_Light1.sty	5.57 KB	Style File	3/1/2018 5:04:07 PM	ENBlack_Light1.sty
ENBlack_Light2.sty	6.02 KB	Style File	3/1/2018 5:04:07 PM	ENBlack_Light2.sty
ENBlack_Medium1.sty	5.98 KB	Style File	3/1/2018 5:04:07 PM	ENBlack_Medium1.sty
ENBlack_Medium2.sty	5.96 KB	Style File	3/1/2018 5:04:07 PM	ENBlack_Medium2.sty
ENblack_theme.sty	5.71 KB	Style File	3/1/2018 5:04:07 PM	ENblack_theme.sty
ENblackbluepurple.sty	11.64 KB	Style File	3/1/2018 5:04:07 PM	ENblackbluepurple.sty
ENblue-medium.sty	20.26 KB	Style File	3/1/2018 5:04:07 PM	ENblue-medium.sty
ENBlue_Dark.sty	5.52 KB	Style File	3/1/2018 5:04:07 PM	ENBlue_Dark.sty
ENBlue_DarkComp.sty	5.64 KB	Style File	3/1/2018 5:04:07 PM	ENBlue_DarkComp.sty
ENBlue_Light1.sty	5.67 KB	Style File	3/1/2018 5:04:07 PM	ENBlue_Light1.sty

Libraries
Templates
Legacy Templates
Domains
Century Electronics
Assignment 01 - Simple Reports
Assignment 02 - Implementing Various Da
Assignment 03 - Other Report Features an
Assignment 04 - Charting
Assignment 05 - Adding Filters to Reports
Assignment 06 - Detail and Print Reports
Assignment 07 - Implementing OLAP
Assignment 08 - Building Documents and I
File name: ENBlack_Dark
WebFOCUS style files (*.sty)
Use Default Stylesheet
Open
Cancel

4. If a customized style sheet has been uploaded to a repository folder, you can apply it by selecting the folder that contains the style sheet, as shown in the following image.



Select a Theme

IBFS:/WFC ▸ Domains ▸ Century Electronics ▸ Assignment 01 - Simple Reports ▸

Title	Size	Type	Last Modified	Name	Shared F
ENBlack_Dark	5.71 KB	Style File	2/12/2018 12:52:45 F	ENBlack_Dark.sty	

Libraries
Templates
Legacy Templates
Domains
Century Electronics
Assignment 01 - Simple Reports
Assignment 02 - Implementing Various Da
Assignment 03 - Other Report Features an
Assignment 04 - Charting
Assignment 05 - Adding Filters to Reports
Assignment 06 - Detail and Print Reports
Assignment 07 - Implementing OLAP
Assignment 08 - Building Documents and I
Title: ENBlack_Dark
WebFOCUS style files (*.sty)
Use Default Stylesheet
Open
Cancel

General Web Query StyleSheet Syntax

A StyleSheet consists of declarations that identify the report components you wish to format and the formatting you wish to apply. A declaration usually begins with the TYPE attribute and is followed by the *attribute=value* pairs you assign to the report component. You can also include comments that provide context for your StyleSheet. Comments do not affect StyleSheet behavior. For details, see *Adding a Comment to a Web Query StyleSheet* on page 35.

For information about identifying a report component, see *Identifying a Report Component in a Web Query StyleSheet* on page 57.

Syntax: How to Specify a Web Query StyleSheet Declaration

Each StyleSheet declaration specifies a series of attributes in the form

```
attribute=value, [attribute=value, ...] $
```

where:

attribute

Is the attribute you are specifying, such as TYPE, COLUMN, COLOR, or FONT.

value

Is the value you assign to the attribute.

Example: Sample Web Query StyleSheet

The following is a request that includes a sample StyleSheet.

Report Request

```
TABLE FILE CENTORD
HEADING
" "
"C e n t u r y  C o r p o r a t i o n"
" "
"Order Revenue - 2000 Q3"
" "
"page <TABPAGENO"
" "
SUM ORDER_DATE/MtDY ORDER_NUM LINEPRICE AS 'Order,Total:'
BY LOWEST 9 ORDER_DATE NOPRINT
WHERE (ORDER_DATE GE '2000/10/01') AND (ORDER_DATE LE '2000/12/31');
ON TABLE SET STYLE *
INCLUDE=IBFS:/stylesheet_location/custom_stylesheet.sty,$
END
```

***custom_stylesheet*.sty**

```
TYPE=REPORT, GRID=OFF, UNITS=INCHES, $
TYPE=DATA, FONT='Times', $
TYPE=DATA, BACKCOLOR=Aqua, COLOR=Navy,
   WHEN=LinePrice GT 500000, $
TYPE=DATA, COLUMN=LinePrice, BACKCOLOR=Aqua, COLOR=Navy, STYLE=Bold,
   WHEN=LinePrice GT 500000, $
TYPE=TITLE, FONT='Helvetica', $
TYPE=HEADING, FONT='Helvetica', STYLE=Bold, SIZE=14, JUSTIFY=Center,
   COLOR=White, BACKCOLOR=Dark Turquoise, $
TYPE=HEADING, LINE=6, BACKCOLOR=White, COLOR=Dark Turquoise, $
TYPE=HEADING, LINE=7, BACKCOLOR=White, $
```

Improving Web Query StyleSheet Readability

There are many ways to structure your StyleSheet declarations in order to make the StyleSheet easy to read. You can do any one, or a combination, of the following:

- Begin a declaration in any column using blank spaces or tabs.
- Include blank lines between declarations.
- Create declarations in all uppercase, all lowercase, or mixed-case.
- Use more than one declaration to format a single report component.
- Include blank spaces or tabs in between the attribute, equal sign (=), value, comma (,), and dollar sign ($).
- Split a single declaration across a line. The declaration will continue to be processed until the terminating dollar sign. For example, you can split a declaration like this:

  ```
  TYPE=HEADING, FONT=ARIAL,
  SIZE=14, STYLE=BOLD, $
  ```

- Split an *attribute*=*value* pair across a line. Use the backslash (\) character as continuation syntax at the end of the first line if you are splitting an attribute or value in a declaration across a line. For example:

  ```
  TYPE=TITLE, COLUMN=PRODUCT, STY\
  LE=BOLD+ITALIC, COLOR=BLUE, $
  ```

Adding a Comment to a Web Query StyleSheet

You can add comments to a StyleSheet to give context to a declaration. Comments do not affect StyleSheet behavior.

You can add a comment:

- **On a declaration line.** Add the desired text after the dollar sign ($). For example:

  ```
  TYPE=HEADING, STYLE=BOLD, COLOR=BLUE, SIZE=14, $ Sample comment
  ```

- **On its own line.** Begin the line with either a dollar sign ($), or a hyphen and an asterisk (-*), followed by the desired text. For example:

  ```
  -* This is a sample comment
  $ This is another sample comment
  ```

Note: You can add comments anywhere in your request, not only in StyleSheets.

Web Query StyleSheet Attribute Inheritance

Each report component inherits StyleSheet attributes from its parent component. You can override an inherited attribute by explicitly specifying the same attribute with a different value in the declaration for the child component. Since each component inherits automatically, you need to specify only those attributes that differ from, or that augment, the inherited attributes of a component.

Inheritance enables you to define common formatting in a single declaration, and to apply it automatically to all child components, except for those components for which you specify different attribute values to override the inherited values. You benefit from less coding and a more concise StyleSheet.

For example, you could specify that all report titles should be blue and bold:

```
TYPE=TITLE, COLOR=BLUE, STYLE=BOLD, $
```

Each column title will inherit this formatting, appearing in blue and bold, by default. However, you can choose to format one column differently, allowing it to inherit the blue color, but specifying that it override the bold style and that it add a yellow background color:

```
TYPE=TITLE, COLUMN=PRODUCT, STYLE=-BOLD, BACKCOLOR=YELLOW, $
```

Reference: Web Query StyleSheet Inheritance Hierarchy

Report components inherit StyleSheet attributes according to a hierarchy. The root of the hierarchy is the entire report, specified in a StyleSheet declaration by TYPE=REPORT. Declarations that omit TYPE default to TYPE=REPORT and are also applied to the entire report. Attributes that are unspecified for the entire report default to values that are determined according to the display format of the report, such as HTML or PDF.

Each report component inherits from its parent component. Component X is a parent of component Y if X is specified by a subset of all the TYPE attributes that specify Y, and if those shared type attributes have the same values. For example:

- A component specified by TYPE=*x*, *subtype*=*y*, *elementtype*=*z* is a child of the component specified by TYPE=*x*, *subtype*=*y* and inherits attributes from it.
- The component specified by TYPE=*x*, *subtype*=*y* is a child of the component specified by TYPE=*x*, and inherits from it.
- The component specified by TYPE=*x*, where *x* is any value other than REPORT, is a child of the entire report (TYPE=REPORT) and inherits from it.

When you use an external cascading style sheet (CSS), a report component inherits formatting from parent HTML elements, not from a parent report component. For more information, see *Inheritance and External Cascading Style Sheets* on page 119.

Example: Augmenting Inherited Web Query StyleSheet Attributes

The following illustrates how to augment inherited StyleSheet attributes.

The page heading in this report has two lines. The first StyleSheet declaration identifies the report component HEADING to be formatted in bold and have 12-point font size. This will format both lines of the heading with these styles.

To augment the format for the second line of the heading, a second declaration has been added that specifies the heading line number and the additional style characteristic. In this case, we have added the declaration TYPE=HEADING, LINE=2, STYLE=ITALIC. The second line of the heading will inherit the bold style and 12-point font size from the first HEADING declaration, and will also receive the italic style defined in the second declaration.

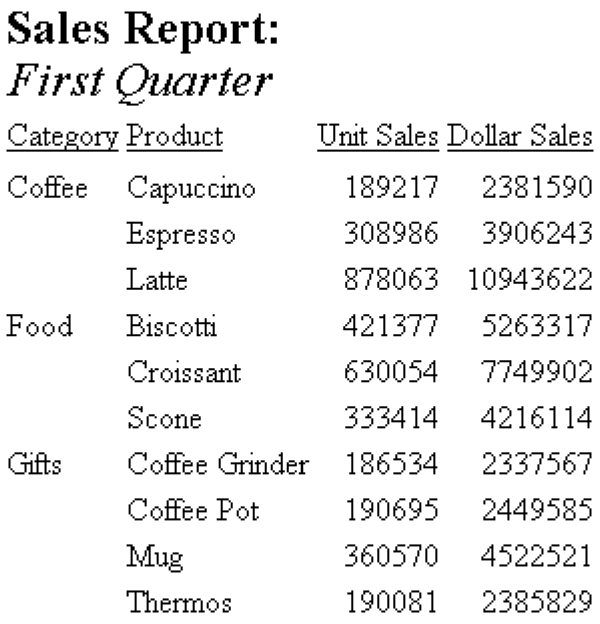
Report Request

```
TABLE FILE GGSALES
SUM UNITS DOLLARS
BY CATEGORY BY PRODUCT
HEADING
"Sales Report:"
"First Quarter"
ON TABLE SET STYLE *
INCLUDE=IBFS:/stylesheet_location/custom_stylesheet.sty,$
END
```

***custom_stylesheet*.sty**

```
TYPE=REPORT, GRID=OFF, $
TYPE=HEADING, STYLE=BOLD, SIZE=12, $
TYPE=HEADING, LINE=2, STYLE=ITALIC, $
```

The output is:



Sales Report:
First Quarter

Category	Product	Unit Sales	Dollar Sales
Coffee	Capuccino	189217	2381590
	Espresso	308986	3906243
	Latte	878063	10943622
Food	Biscotti	421377	5263317
	Croissant	630054	7749902
	Scone	333414	4216114
Gifts	Coffee Grinder	186534	2337567
	Coffee Pot	190695	2449585
	Mug	360570	4522521
	Thermos	190081	2385829

Example: Overriding Inherited Web Query StyleSheet Attributes

The following illustrates how to override StyleSheet inheritance.

Report Request

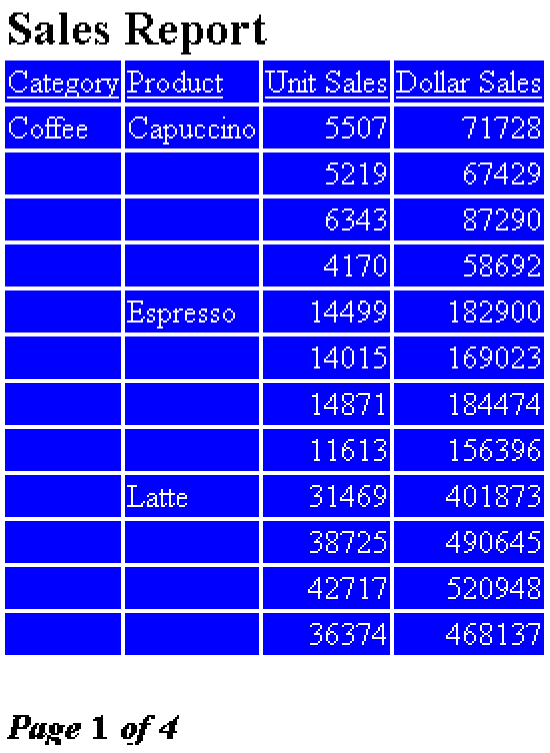
```
TABLE FILE GGSALES
HEADING
"Sales Report"
SUM UNITS DOLLARS
BY CATEGORY BY PRODUCT BY DATE NOPRINT
WHERE DATE GE 19960101 AND DATE LE 19960401
FOOTING
"Page <TABPAGENO of <TABLASTPAGE"
ON TABLE SET STYLE *
INCLUDE=IBFS:/stylesheet_location/custom_stylesheet.sty,$
END
```

***custom_stylesheet*.sty**

```
TYPE=REPORT, BACKCOLOR=BLUE, COLOR=WHITE, $
TYPE=HEADING, BACKCOLOR=WHITE, COLOR=BLACK, STYLE=BOLD, SIZE=12, $
TYPE=FOOTING, SIZE=11, STYLE=BOLD+ITALIC, BACKCOLOR=WHITE, COLOR=BLACK, $
TYPE=FOOTING, OBJECT=FIELD, ITEM=1, STYLE=-ITALIC, $
```

- The TYPE=REPORT declaration formats the entire report (all components) to appear with a blue background and white font.
- The TYPE=HEADING declaration overrides the inherited format for the page heading (defined in the TYPE=REPORT declaration) by specifying the background color as white and the font as black.
- The first TYPE=FOOTING declaration formats the page footing as font size 11, with a bold and italic style, and overrides the report color by specifying BACKCOLOR=WHITE and COLOR=BLACK.
- Since the <TABPAGENO system variable is part of the page footing, the second TYPE=FOOTING declaration inherits all of the formatting specified in the first TYPE=FOOTING declaration. This declaration overrides the inherited format for the page footing by specifying OBJECT=FIELD, ITEM=1, and removing the italic style (STYLE=-ITALIC). Note that ITEM=1 needs to be specified since there are two embedded fields in the footing.

The output is:



Sales Report

Category	Product	Unit Sales	Dollar Sales
Coffee	Capuccino	5507	71728
		5219	67429
		6343	87290
		4170	58692
	Espresso	14499	182900
		14015	169023
		14871	184474
		11613	156396
	Latte	31469	401873
		38725	490645
		42717	520948
		36374	468137

Page* 1 *of 4

Chapter 4

Controlling Report Formatting

When you format a report, you can control *how* the formatting is applied. You can:

- **Generate an internal cascading style sheet for HTML reports.** This improves performance and provides more formatting options.
- **Apply formatting conditionally.** You can make the appearance of a report conditional based on the report values. You can also make its links to other reports and Internet resources conditional. For example, in a monthly order report, you can specify that all unpaid orders be displayed in red and be linked to a recent payment history report of a customer.
- **Select a scale for specifying measurements,** such as the size of a top margin in a report. You can select an inch, centimeter, or point as the unit of measure.
- **Control the display of empty reports.** If a report request returns no records, you can choose to display the report without data, or to display a message stating there is no output.

In this chapter:

Generating an Internal Cascading Style Sheet for HTML Reports

When you create a report in HTML format, code is generated that specifies how the report is formatted. An internal cascading style sheet is generated as part of this HTML code. This will:

- **Improve performance** by significantly reducing the size of the HTML file, decreasing transmission bandwidth, and displaying large reports more quickly.
- **Provide more formatting options** for your HTML report. Some Web Query StyleSheet attributes are supported for HTML display format only in reports that generate an internal cascading style sheet.

Internal cascading style sheets enable HTML support for the UNITS, BOTTOMMARGIN, TOPMARGIN, LEFTMARGIN, RIGHTMARGIN, SIZE, POSITION, WRAP, and PAGECOLOR attributes. It also enables you to add and remove underlines from most report components and specify the starting position and size of an image. For details on the UNITS attribute, see *Selecting a Unit of Measurement* on page 42. For more information on all other attributes, see *Laying Out the Report Page* on page 123.

You can apply an external cascading style sheet in the same report request. If any formatting instructions conflict, the internal cascading style sheet overrides the external cascading style sheet.

Note: In most cases, you should not specify native Web Query StyleSheet attributes and external CSS classes in the same report or style sheet. For more information, see *Using an External Cascading Style Sheet* on page 101.

Selecting a Unit of Measurement

InfoAssist contains a Layout tab that allows a developer to define margins, unit of measurement, page size, and other attributes that are covered in this chapter. It will generate the StyleSheet code for you. You can select the unit of measurement for page margins and column width for HTML reports that generate an internal cascading style sheet, as well as PDF reports. In addition, you can also select the unit of measurement for column position in PDF reports. You can select inches, centimeters, or points as the unit of measure.

If you change the unit of measure, all existing measurements are automatically converted to the new scale. For example, if the unit of measure is inches and the top margin for the report is set to 1, and you later change the unit of measure to centimeters, the size of the top margin is automatically converted to 1 centimeter.

Syntax: How to Set the Unit of Measurement

To set a unit of measurement in a StyleSheet, add the following attribute:

```
UNITS=units
```

where:

units

Is the unit of measure. Values can be:

- **INCHES,** that specifies the unit of measure as inches. This is the default value.
- **CM,** that specifies the unit of measure as centimeters.

- **PTS,** that specifies the unit of measure as points. Points is a common measurement scale for typefaces.

Conditionally Formatting, Displaying, and Linking in a StyleSheet

You can conditionally format report components, display a graphic, and include links in your report based on the values in your report. Using conditional styling, you can:

- Draw attention to particular items in the report.
- Emphasize differences between significant values.
- Customize the resources to which an end user navigates from different parts of the report.

To conditionally format reports, add the WHEN attribute to a StyleSheet declaration. The WHEN attribute specifies a condition that is evaluated for each instance of a report component (that is, for each cell of a tabular report column, each element in a chart, or each free-form report page). The StyleSheet declaration is applied to each instance that satisfies the condition, and is ignored by each instance that fails to satisfy the condition.

You can also apply sequential conditional formatting.

Note: The variables TABPAGENO and TABLASTPAGE cannot be used to define styling with conditional styling (WHEN).

Applying Sequential Conditional Formatting

Within InfoAssist, fields on a report can have Traffic Light conditions applied to them. This automatically generates the StyleSheet syntax covered in this topic. You can apply sequential conditional logic to a report component by creating a series of declarations, each with a different condition. This is the StyleSheet equivalent of a sequence of nested IF-THEN-ELSE statements. When several conditional declarations specify the same report component (for example, the same column) *and* evaluate the same field in the condition, they are processed together as a group. For each instance of the report component (for example, for each cell of a column):

1. The conditional declarations in the *group* are evaluated, in the order in which they are found in the StyleSheet, until one of the conditions is satisfied. That declaration is then applied to that instance of the report component. The other conditional declarations in the *group*, and any non-conditional declarations that specify the same report component and the same attributes, are ignored for that instance.
2. If, however, none of the conditional declarations have been satisfied for that instance, then the first unconditional declaration for that report component that specifies the same attribute or attributes is applied to that instance.

3. Any unconditional declarations for that report component that specify other attributes (that is, attributes that have not already been applied to the instance in steps 1 or 2) are now applied to the instance.
4. The entire process is repeated for the next instance of the report component (for example, for the next cell of the column).

Syntax: How to Conditionally Format, Display, or Link in a StyleSheet

```
TYPE=type, [subtype,] attributes, WHEN=field1 operator {field2|value},$
```

where:

type

Is the value of the TYPE attribute. You can specify any report component. For details, see *Identifying a Report Component in a Web Query StyleSheet* on page 57.

subtype

Are any additional attributes, such as COLUMN, ACROSS, or ITEM, that are needed to identify the report component to which you are applying the declaration.

attributes

Are the attributes in the StyleSheet declaration that are made conditional by the WHEN attribute. They can include most formatting, graphic images, and hyperlink attributes.

field1, field2

Identifies the report fields that are being compared. Each one can be:

- The name of a display field or vertical or horizontal sort field in a graph or tabular report. ACROSS values can be used as part of the conditional expressions used to define styling attributes for each cell in the table.
- A column reference in a chart or tabular report.
- The name of an embedded field in the heading or footing of a free-form report.

If you wish to use a field that you do not want to display in the report, you can specify the field in the report request, and use the NOPRINT option to prevent the field from being displayed (for example, PRINT *fieldname* NOPRINT).

To apply a prefix operator to a field in a report:

- Use the same prefix operator in the WHEN attribute. You must refer to the field by name in the WHEN attribute (for example, WHEN=AVE.PRICE GT 300).

- You cannot use compound Boolean expressions with the WHEN attribute.

The field cannot be a packed (P) numeric field.

operator

Defines how the condition is satisfied. You can use these relational operators:

`EQ` where the condition is satisfied if the values on the left and right are equal. If the values being compared are alphanumeric, their case (uppercase, lowercase, or mixed-case) must match.

`NE` where the condition is satisfied if the values on the left and right are not equal.

`LT` where the condition is satisfied if the value on the left is less than the value on the right.

`LE` where the condition is satisfied if the value on the left is less than or equal to the value on the right.

`GT` where the condition is satisfied if the value on the left is greater than the value on the right.

`GE` where the condition is satisfied if the value on the left is greater than or equal to the value on the right.

value

Is a constant, such as a number, character string, or date. You must enclose non-numeric constants, such as character strings and dates, in single quotation marks (').

Although you cannot use functions or operators here to specify the value, you can define a temporary field (COMPUTE or DEFINE) using functions and operators, use the temporary field in the report, and specify it here instead of a constant.

Example: Using Sequential Conditional Formatting

The following example illustrates how to apply sequential conditional formatting to a report. This report uses sequential conditional logic to format each row, based on its order total (LINEPRICE).

- The first TYPE=DATA conditional declaration formats any rows whose order total is greater than 500,000.
- The second TYPE=DATA conditional declaration formats any rows whose order total is greater than 400,000 and less than or equal to 500,000. This is because rows with an order total greater than 500,000 would have already been formatted by the first conditional declaration.

- The third TYPE=DATA conditional declaration formats any rows whose order total is greater than 100,000 and less than or equal to 400,000. This is because rows with an order total greater than 400,000 would have already been formatted by one of the first two conditional declarations.
- The unconditional declaration, which does not include a WHEN clause, following the conditional declarations specifies:
  - Background color, which is also specified by the conditional declarations. It applies background color (silver) to any rows whose order total is less than or equal to 100,000, since those rows have not already been formatted by the conditional declarations.
  - Font, which is *not* specified by the conditional declarations. It applies font (Arial) to all data rows.

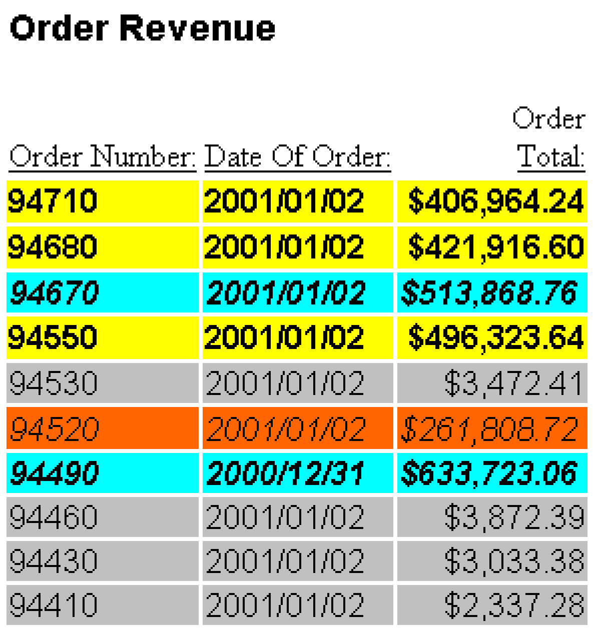
Report Request

```
TABLE FILE CENTORD
HEADING
"Order Revenue"
" "
SUM ORDER_DATE LINEPRICE AS 'Order,Total:'
BY HIGHEST 10 ORDER_NUM
ON TABLE SET STYLE *
INCLUDE=IBFS:/stylesheet_location/custom_stylesheet.sty,$
END
```

***custom_stylesheet*.sty**

```
TYPE=REPORT, GRID=OFF, $
TYPE=DATA, BACKCOLOR=AQUA, STYLE=BOLD+ITALIC, WHEN=LINEPRICE GT 500000, $
TYPE=DATA, BACKCOLOR=YELLOW, STYLE=BOLD, WHEN=LINEPRICE GT 400000, $
TYPE=DATA, BACKCOLOR=ORANGE, STYLE=ITALIC, WHEN=LINEPRICE GT 100000, $
TYPE=DATA, BACKCOLOR=SILVER, FONT='Arial', $
TYPE=HEADING, FONT='Arial', STYLE=BOLD, SIZE=11, $
```

The output is:



Order Revenue

Order Number:	Date Of Order:	Order Total:
94710	2001/01/02	$406,964.24
94680	2001/01/02	$421,916.60
94670	2001/01/02	$513,868.76
94550	2001/01/02	$496,323.64
94530	2001/01/02	$3,472.41
94520	2001/01/02	$261,808.72
94490	2000/12/31	$633,723.06
94460	2001/01/02	$3,872.39
94430	2001/01/02	$3,033.38
94410	2001/01/02	$2,337.28

Example: Applying Basic Conditional Formatting

The following example illustrates how to apply conditional formatting to a report. The conditional formatting draws attention to orders that total more than 200,000.

Notice that because a particular column is not specified in the declaration, the formatting is applied to the entire data row.

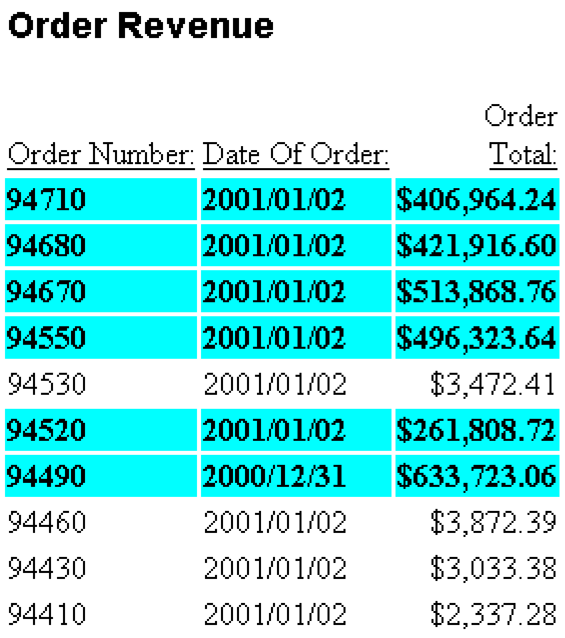
Report Request

```
TABLE FILE CENTORD
HEADING
"Order Revenue"
" "
SUM ORDER_DATE LINEPRICE AS 'Order,Total:'
BY HIGHEST 10 ORDER_NUM
ON TABLE SET STYLE *
INCLUDE=IBFS:/stylesheet_location/custom_stylesheet.sty,$
END
```

***custom_stylesheet*.sty**

```
TYPE=REPORT, GRID=OFF, $
TYPE=DATA, BACKCOLOR=AQUA, STYLE=BOLD, WHEN=LINEPRICE GT 200000, $
TYPE=HEADING, FONT='Arial', STYLE=BOLD, SIZE=11, $
```

The output is:



Order Revenue

Order Number:	Date Of Order:	Order Total:
94710	**2001/01/02**	**$406,964.24**
94680	**2001/01/02**	**$421,916.60**
94670	**2001/01/02**	**$513,868.76**
94550	**2001/01/02**	**$496,323.64**
94530	2001/01/02	$3,472.41
94520	**2001/01/02**	**$261,808.72**
94490	**2000/12/31**	**$633,723.06**
94460	2001/01/02	$3,872.39
94430	2001/01/02	$3,033.38
94410	2001/01/02	$2,337.28

Example: Applying Conditional Formatting to a Column

The following example illustrates how you can use conditional formatting to draw attention to columns that are not specified in the condition. The WHEN condition states that the order number for orders exceeding 200,000 should display in boldface with an aqua background.

Notice that the column that is evaluated in the WHEN condition (LINEPRICE) is different from the column that is formatted (ORDER_NUM). They do not need to be the same.

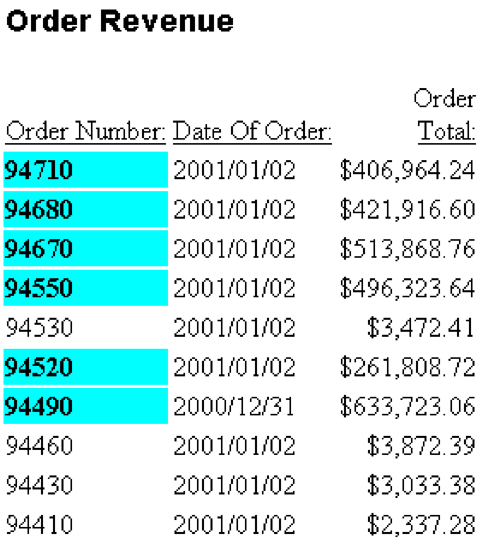
Report Request

```
TABLE FILE CENTORD
HEADING
"Order Revenue"
" "
SUM ORDER_DATE LINEPRICE AS 'Order,Total:'
BY HIGHEST 10 ORDER_NUM
ON TABLE SET STYLE *
INCLUDE=IBFS:/stylesheet_location/custom_stylesheet.sty,$
END
```

***custom_stylesheet*.sty**

```
TYPE=REPORT, GRID=OFF, $
TYPE=DATA, COLUMN=ORDER_NUM, BACKCOLOR=AQUA, STYLE=BOLD,
   WHEN=LINEPRICE GT 200000, $
TYPE=HEADING, FONT='Arial', STYLE=BOLD, SIZE=11, $
```

The output is:



Order Revenue

Order Number:	Date Of Order:	Order Total:
94710	2001/01/02	$406,964.24
94680	2001/01/02	$421,916.60
94670	2001/01/02	$513,868.76
94550	2001/01/02	$496,323.64
94530	2001/01/02	$3,472.41
94520	2001/01/02	$261,808.72
94490	2000/12/31	$633,723.06
94460	2001/01/02	$3,872.39
94430	2001/01/02	$3,033.38
94410	2001/01/02	$2,337.28

Example: Conditionally Styling an ACROSS Value

In the following example, the ACROSS values are used in conditional styling to set a unique backcolor for all ACROSS columns in the Category Coffee, and additional font styling for the Espresso ACROSS column.

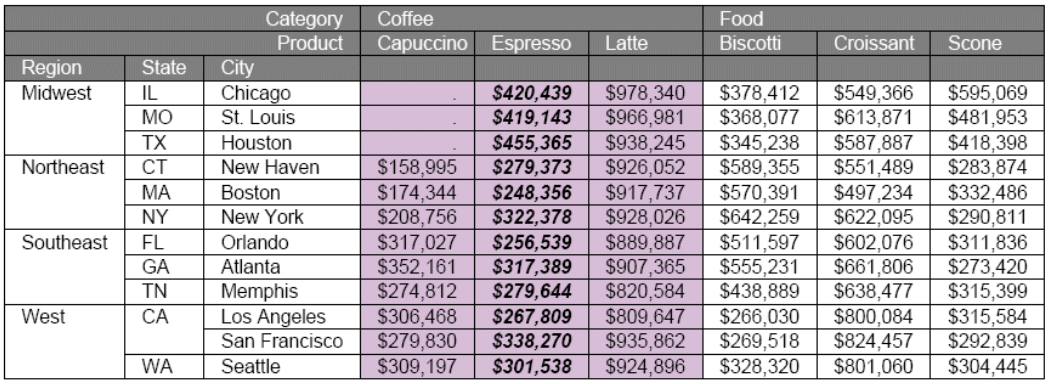
Report Request

```
SET ACROSSTITLE=SIDE
TABLE FILE GGSALES
SUM DOLLARS/I8M AS ''
BY REGION
BY ST
BY CITY
ACROSS CATEGORY
ACROSS PRODUCT
WHERE CATEGORY EQ 'Coffee' OR 'Food';
ON TABLE NOTOTAL
ON TABLE PCHOLD FORMAT PDF
ON TABLE SET STYLE *
INCLUDE=IBFS:/stylesheet_location/custom_stylesheet.sty,$
END
```

***custom_stylesheet*.sty**

```
TYPE=REPORT, SQUEEZE=ON, UNITS=IN, ORIENTATION=PORTRAIT, $
TYPE=REPORT, FONT='ARIAL', SIZE=10, BORDER=LIGHT, $
TYPE=ACROSSTITLE, COLOR=WHITE, BACKCOLOR=GREY, $
TYPE=ACROSSVALUE ,COLOR=WHITE, BACKCOLOR=GREY, $
TYPE=TITLE, COLOR=WHITE, BACKCOLOR=GREY, $
TYPE=DATA, ACROSSCOLUMN=DOLLARS, BACKCOLOR=THISTLE,
   WHEN=CATEGORY EQ 'Coffee', $
TYPE=DATA, ACROSSCOLUMN=DOLLARS, STYLE=BOLD+ITALIC, WHEN=A2 EQ 'Espresso', $
```

The output is:



		Category	Coffee			Food		
		Product	Capuccino	Espresso	Latte	Biscotti	Croissant	Scone
Region	State	City						
Midwest	IL	Chicago	.	***$420,439***	$978,340	$378,412	$549,366	$595,069
	MO	St. Louis	.	***$419,143***	$966,981	$368,077	$613,871	$481,953
	TX	Houston	.	***$455,365***	$938,245	$345,238	$587,887	$418,398
Northeast	CT	New Haven	$158,995	***$279,373***	$926,052	$589,355	$551,489	$283,874
	MA	Boston	$174,344	***$248,356***	$917,737	$570,391	$497,234	$332,486
	NY	New York	$208,756	***$322,378***	$928,026	$642,259	$622,095	$290,811
Southeast	FL	Orlando	$317,027	***$256,539***	$889,887	$511,597	$602,076	$311,836
	GA	Atlanta	$352,161	***$317,389***	$907,365	$555,231	$661,806	$273,420
	TN	Memphis	$274,812	***$279,644***	$820,584	$438,889	$638,477	$315,399
West	CA	Los Angeles	$306,468	***$267,809***	$809,647	$266,030	$800,084	$315,584
		San Francisco	$279,830	***$338,270***	$935,862	$269,518	$824,457	$292,839
	WA	Seattle	$309,197	***$301,538***	$924,896	$328,320	$801,060	$304,445

Example: Conditionally Formatting a Data Visualization Bar Graph

The following example illustrates how to apply conditional formatting to a data visualization bar graph. This report request incorporates a data visualization bar chart to graphically represent the data in the LINEPRICE column. It uses conditional formatting to draw attention to orders that total more than 200,000. It conditionally applies that formatting to both the data columns (TYPE=DATA) and to the bar graph (GRAPHTYPE=DATA).

Note that data visualization is only supported for HTML reports.

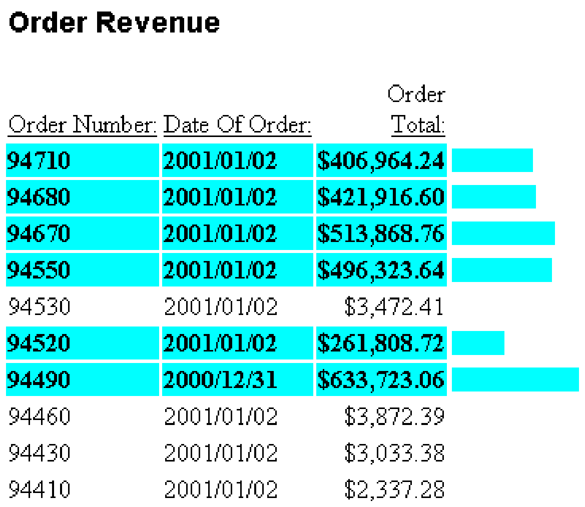
Report Request

```
TABLE FILE CENTORD
HEADING
"Order Revenue"
" "
SUM ORDER_DATE LINEPRICE AS 'Order,Total:'
BY HIGHEST 10 ORDER_NUM
ON TABLE SET STYLE *
INCLUDE=IBFS:/stylesheet_location/custom_stylesheet.sty,$
END
```

***custom_stylesheet*.sty**

```
TYPE=REPORT, GRID=OFF, $
TYPE=DATA, BACKCOLOR=AQUA, STYLE=BOLD, WHEN=LINEPRICE GT 200000, $
GRAPHTYPE=DATA, COLUMN=LINEPRICE, $
GRAPHTYPE=DATA, GRAPHCOLOR=AQUA, WHEN=LINEPRICE GT 200000, $
TYPE=HEADING, FONT='Arial', STYLE=BOLD, SIZE=11, $
```

The output is:



Order Revenue

Order Number:	Date Of Order:	Order Total:
94710	**2001/01/02**	**$406,964.24**
94680	**2001/01/02**	**$421,916.60**
94670	**2001/01/02**	**$513,868.76**
94550	**2001/01/02**	**$496,323.64**
94530	2001/01/02	$3,472.41
94520	**2001/01/02**	**$261,808.72**
94490	**2000/12/31**	**$633,723.06**
94460	2001/01/02	$3,872.39
94430	2001/01/02	$3,033.38
94410	2001/01/02	$2,337.28

Example: Applying Conditional Formatting Based on Hidden (NOPRINT) Field Values

The following example illustrates how to apply conditional formatting based on the values of a hidden (NOPRINT) field. This report uses conditional formatting to draw attention to those employees who have resigned.

Notice that the WHEN attribute condition evaluates a field (STATUS) that is hidden in the report. Although the field that is evaluated in the condition must be included in the report request, you can prevent it from displaying in the report by using the NOPRINT option, as shown in the following request.

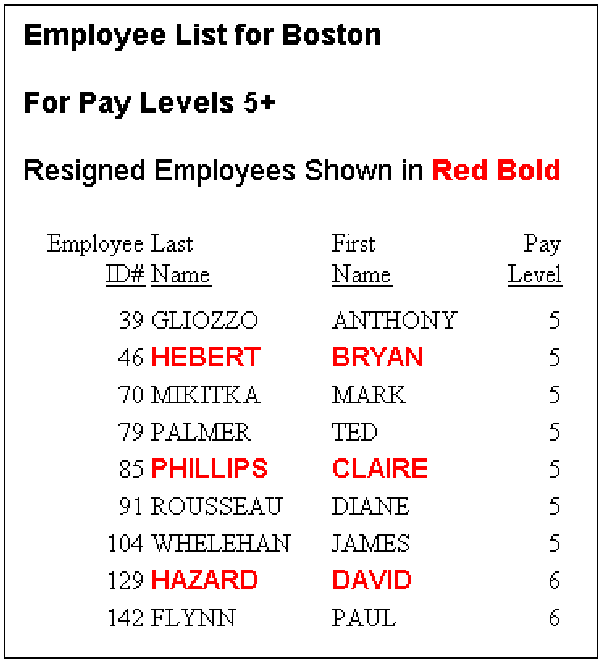
Report Request

```
TABLE FILE CENTHR
HEADING
"Employee List for Boston"
" "
"For Pay Levels 5+"
" "
"Resigned Employees Shown in <0>Red Bold"
" "
PRINT LNAME FNAME PAYSCALE STATUS NOPRINT
BY ID_NUM
WHERE PLANT EQ 'BOS' AND PAYSCALE GE 5
ON TABLE SET STYLE *
INCLUDE=IBFS:/stylesheet_location/custom_stylesheet.sty,$
END
```

***custom_stylesheet*.sty**

```
TYPE=REPORT, GRID=OFF, $
TYPE=DATA, COLUMN=LNAME, COLOR=RED, FONT='Arial', STYLE=BOLD,
   WHEN=STATUS EQ 'RESIGNED', $
TYPE=DATA, COLUMN=FNAME, COLOR=RED, FONT='Arial', STYLE=BOLD,
   WHEN=STATUS EQ 'RESIGNED', $
TYPE=HEADING, FONT='Arial', STYLE=BOLD, SIZE=11, $
TYPE=HEADING, LINE=5, STYLE=-BOLD, $
TYPE=HEADING, LINE=5, ITEM=2, STYLE=BOLD, COLOR=RED, $
```

The output is:



```
Employee List for Boston

For Pay Levels 5+

Resigned Employees Shown in Red Bold

   Employee Last              First           Pay
        ID# Name              Name          Level

         39 GLIOZZO           ANTHONY           5
         46 HEBERT            BRYAN             5
         70 MIKITKA           MARK              5
         79 PALMER            TED               5
         85 PHILLIPS          CLAIRE            5
         91 ROUSSEAU          DIANE             5
        104 WHELEHAN          JAMES             5
        129 HAZARD            DAVID             6
        142 FLYNN             PAUL              6
```

Example: **Applying Conditional Formatting to a Sort Group**

The following example illustrates how to apply conditional formatting to a sort group. This report uses conditional formatting to draw attention to those employees who have resigned.

Notice that one conditional declaration can apply formatting to all the sort group rows. You can accomplish this by evaluating the sort field (STATUS) in the WHEN attribute condition.

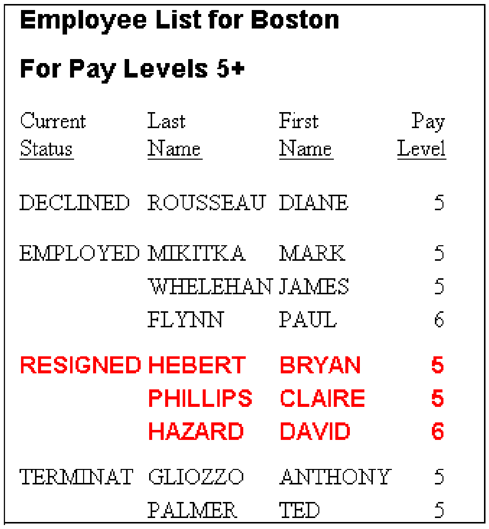
Report Request

```
TABLE FILE CENTHR
HEADING
"Employee List for Boston"
" "
"For Pay Levels 5+"
" "
PRINT LNAME FNAME PAYSCALE
BY STATUS SKIP-LINE
WHERE PLANT EQ 'BOS' AND PAYSCALE GE 5
ON TABLE SET STYLE *
INCLUDE=IBFS:/stylesheet_location/custom_stylesheet.sty,$
END
```

***custom_stylesheet*.sty**

```
TYPE=REPORT, GRID=OFF, $
TYPE=DATA, COLOR=RED, FONT='Arial', STYLE=BOLD,
   WHEN=STATUS EQ 'RESIGNED', $
TYPE=HEADING, FONT='Arial', STYLE=BOLD, SIZE=11, $
```

The output is:



```
Employee List for Boston

For Pay Levels 5+

Current   Last      First        Pay
Status    Name      Name       Level

DECLINED  ROUSSEAU  DIANE          5

EMPLOYED  MIKITKA   MARK           5
          WHELEHAN  JAMES          5
          FLYNN     PAUL           6

RESIGNED  HEBERT    BRYAN          5
          PHILLIPS  CLAIRE         5
          HAZARD    DAVID          6

TERMINAT  GLIOZZO   ANTHONY        5
          PALMER    TED            5
```

In order to apply the same conditional formatting to only two columns, instead of all the columns, the following version of the report request uses two declarations, each specifying a different column (LNAME and FNAME).

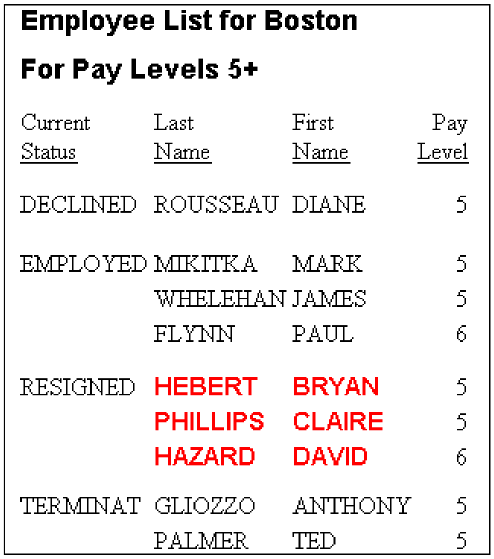
Report Request

```
TABLE FILE CENTHR
HEADING
"Employee List for Boston"
" "
"Pay Levels 5+"
" "
PRINT LNAME FNAME PAYSCALE
BY STATUS SKIP-LINE
WHERE PLANT EQ 'BOS' AND PAYSCALE GE 5
ON TABLE SET STYLE *
INCLUDE=IBFS:/stylesheet_location/custom_stylesheet.sty,$
END
```

***custom_stylesheet*.sty**

```
TYPE=REPORT, GRID=OFF, $
TYPE=DATA, COLUMN=LNAME, COLOR=RED, FONT='Arial', STYLE=BOLD,
   WHEN=STATUS EQ 'RESIGNED', $
TYPE=DATA, COLUMN=FNAME, COLOR=RED, FONT='Arial', STYLE=BOLD,
   WHEN=STATUS EQ 'RESIGNED', $
TYPE=HEADING, FONT='Arial', STYLE=BOLD, SIZE=11, $
```

The output is:



```
Employee List for Boston

For Pay Levels 5+

Current    Last       First         Pay
Status     Name       Name        Level

DECLINED   ROUSSEAU   DIANE           5

EMPLOYED   MIKITKA    MARK            5
           WHELEHAN   JAMES           5
           FLYNN      PAUL            6

RESIGNED   HEBERT     BRYAN           5
           PHILLIPS   CLAIRE          5
           HAZARD     DAVID           6

TERMINAT   GLIOZZO    ANTHONY         5
           PALMER     TED             5
```

Chapter 5

Identifying a Report Component in a Web Query StyleSheet

A report consists of several types of components, each of which you can identify in a StyleSheet in order to format it.

Note: InfoAssist has built-in styling options that generate some of the StyleSheet syntax described in this chapter.

Report components are subject to StyleSheet inheritance. For details, see *Introducing Formatting and Style Sheets* on page 27.

Unless otherwise noted, all StyleSheet references in this chapter refer to Web Query StyleSheets.

In this chapter:

Identifying an Entire Report, Column, or Row

You can apply formatting to an:

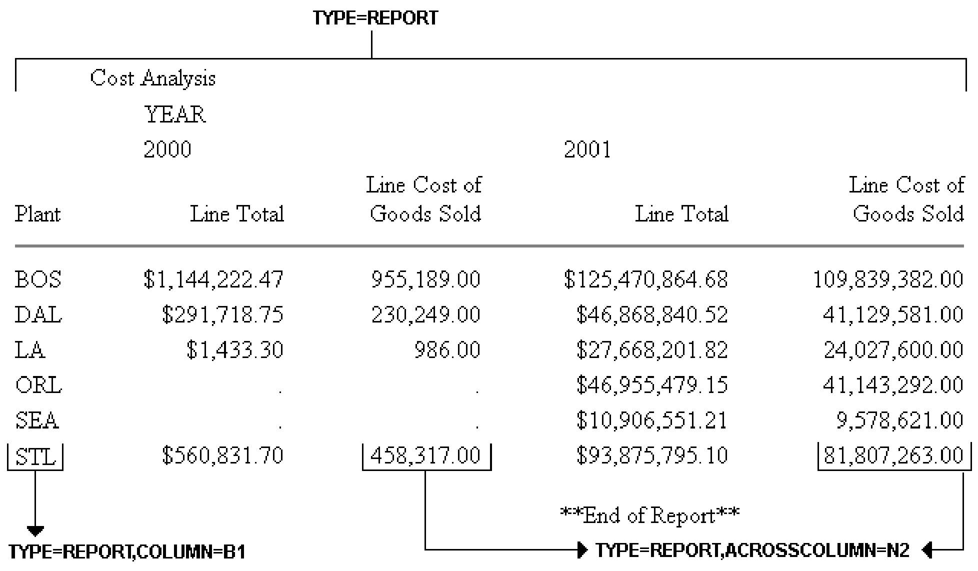
- ❑ **Entire report**. For more information, see *How to Identify an Entire Report* on page 58.
- ❑ **Entire column** within a report, both its title and data (including ROW-TOTAL columns). For more information, see *How to Identify an Entire Column* on page 60.
- ❑ **Entire row** within a report, or a total or subtotal row, comprising the data of the row. For more information, *How to Identify an Entire Total or Subtotal Row* on page 63.

  You can also identify an entire horizontal sort (ACROSS) title or value row in a StyleSheet, although each of these rows contains only a single kind of information. For details, see *How to Identify a Column Title* on page 81.

The following example illustrates where the REPORT component and the COLUMN and ACROSSCOLUMN attributes appear in a report, and which TYPE values you use to identify them. Although in this example the value for COLUMN is PLANT and the value for ACROSSCOLUMN is YEAR, these are not the only values you can use to identify these components.

Report Request

```
TABLE FILE CENTORD
SUM LINEPRICE LINE_COGS AS 'Line Cost of,Goods Sold'
BY PLANT AS 'Plant'
ACROSS YEAR
WHERE YEAR EQ 2000 or 2001
HEADING
"Cost Analysis"
FOOTING CENTER
"**End of Report**"
END
```



```
      Cost Analysis
          YEAR
          2000                                   2001
                                Line Cost of                          Line Cost of
Plant          Line Total         Goods Sold          Line Total        Goods Sold
------------------------------------------------------------------------------------
BOS         $1,144,222.47         955,189.00     $125,470,864.68    109,839,382.00
DAL           $291,718.75         230,249.00      $46,868,840.52     41,129,581.00
LA              $1,433.30             986.00      $27,668,201.82     24,027,600.00
ORL                     .                  .      $46,955,479.15     41,143,292.00
SEA                     .                  .      $10,906,551.21      9,578,621.00
STL           $560,831.70         458,317.00      $93,875,795.10     81,807,263.00

                                          **End of Report**
```

Note: Since this request simply illustrates where the components appear in a report, it omits a StyleSheet.

Syntax:

How to Identify an Entire Report

To identify an entire report in a StyleSheet, use this attribute and value:

```
TYPE=REPORT
```

Example: Identifying an Entire Report

The following example illustrates how to identify and apply formatting to an entire report.

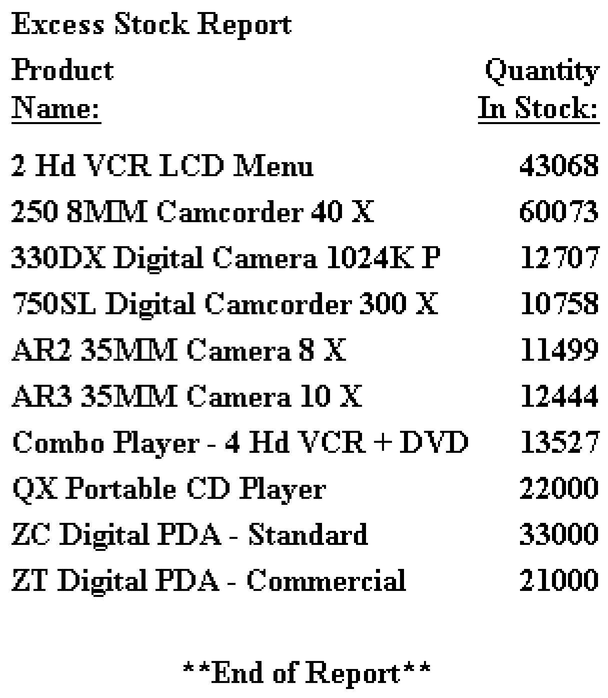
Report Request

```
TABLE FILE CENTINV
HEADING
"Excess Stock Report"
SUM QTY_IN_STOCK
BY PRODNAME
WHERE QTY_IN_STOCK GT 10000
FOOTING CENTER
"**End of Report**"
ON TABLE SET STYLE *
INCLUDE=IBFS:/stylesheet_location/custom_stylesheet.sty,$
END
```

***custom_stylesheet*.sty**

```
TYPE=REPORT, GRID=OFF, STYLE=BOLD, $
```

The output is:



```
Excess Stock Report
Product                              Quantity
Name:                                In Stock:

2 Hd VCR LCD Menu                       43068
250 8MM Camcorder 40 X                  60073
330DX Digital Camera 1024K P            12707
750SL Digital Camcorder 300 X           10758
AR2 35MM Camera 8 X                     11499
AR3 35MM Camera 10 X                    12444
Combo Player - 4 Hd VCR + DVD           13527
QX Portable CD Player                   22000
ZC Digital PDA - Standard               33000
ZT Digital PDA - Commercial             21000

              **End of Report**
```

Syntax: How to Identify an Entire Column

```
TYPE=REPORT, coltype=column
```

where:

`coltype`

Specifies the type of column. It can be:

- ❏ **COLUMN,** which specifies a sort column (generated by BY), a display column (generated by PRINT, LIST, SUM, or COUNT), a computed column (generated by COMPUTE), or a column of row totals (generated by ROW-TOTAL).
- ❏ **ACROSSCOLUMN,** which specifies every instance of a column that is repeated across a horizontal sort (ACROSS) row. This also applies the formatting to the horizontal sort (ACROSS) values that appear above the column titles.

`column`

Specifies one or more columns.

Options for identifying columns in a StyleSheet are:

Identifier	**Description**
`field`	Identifies a column by its field name. When a field occurs more than once, use field(*n*) to select a particular occurrence or field(*) to select all occurrences of the field.
`ROWTOTAL`	Identifies a column of row totals generated using ROW-TOTAL. When used with ACROSS and multiple display commands, ROWTOTAL generates multiple total columns. Use ROWTOTAL(*n*) to select a particular total column. Use ROWTOTAL(*field*) to select the row total column for a particular field. Use ROWTOTAL(*) to select all row total columns in the report.

Example: Identifying an Entire Column

The following example illustrates how to identify an entire column, which consists of the column data and the column title, in a report.

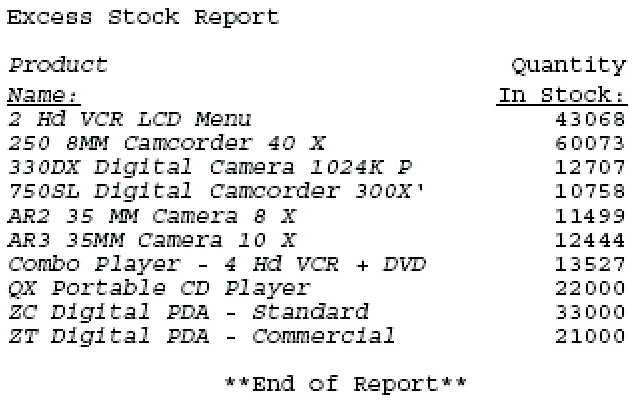
Report Request

```
TABLE FILE CENTINV
HEADING
"Excess Stock Report"
SUM QTY_IN_STOCK
BY PRODNAME
WHERE QTY_IN_STOCK GT 10000
FOOTING CENTER
"**End of Report**"
ON TABLE SET STYLE *
INCLUDE=IBFS:/stylesheet_location/custom_stylesheet.sty,$
END
```

***custom_stylesheet*.sty**

```
TYPE=REPORT, GRID=OFF, $
TYPE=REPORT, COLUMN=PRODNAME, STYLE=ITALIC, $
```

The output is:



```
Excess Stock Report

Product                              Quantity
Name:                               In Stock:
2 Hd VCR LCD Menu                       43068
250 8MM Camcorder 40 X                  60073
330DX Digital Camera 1024K P            12707
750SL Digital Camcorder 300X'           10758
AR2 35 MM Camera 8 X                    11499
AR3 35MM Camera 10 X                    12444
Combo Player - 4 Hd VCR + DVD           13527
QX Portable CD Player                   22000
ZC Digital PDA - Standard               33000
ZT Digital PDA - Commercial             21000

                 **End of Report**
```

Example: Identifying an Entire Horizontal (ACROSS) Column

The following example illustrates how to identify a horizontal (ACROSS) column. When you identify and format an ACROSSCOLUMN, all data values, the column title, and any horizontal sort (ACROSS) values associated with the field are formatted for every instance of the column in the report output.

Note: To produce the same results you can alternatively use the values P1 and P2, respectively, for the ACROSSCOLUMN attribute.

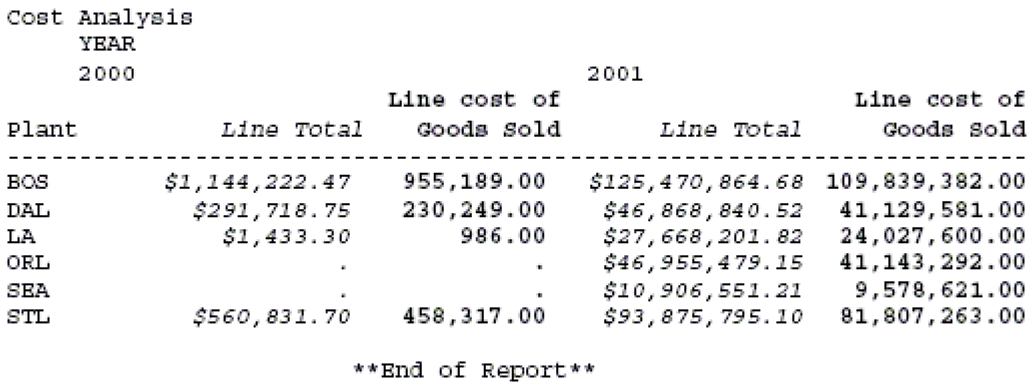
Report Request

```
TABLE FILE CENTORD
SUM LINEPRICE LINE_COGS AS 'Line Cost of,Goods Sold'
BY PLANT AS 'Plant'
ACROSS YEAR
WHERE YEAR EQ 2000 OR 2001
HEADING
"Cost Analysis"
FOOTING CENTER
"**End of Report**"
ON TABLE SET STYLE *
INCLUDE=IBFS:/stylesheet_location/custom_stylesheet.sty,$
END
```

***custom_stylesheet*.sty**

```
TYPE=REPORT, GRID=OFF, $
TYPE=REPORT, ACROSSCOLUMN=LINEPRICE, STYLE=ITALIC, $
TYPE=REPORT, ACROSSCOLUMN=LINE_COGS, STYLE=BOLD, $
```

The output is:



```
Cost Analysis
     YEAR
     2000                                    2001
                             Line cost of                          Line cost of
Plant           Line Total    Goods Sold          Line Total        Goods Sold
--------------------------------------------------------------------------------
BOS          $1,144,222.47    955,189.00     $125,470,864.68  109,839,382.00
DAL            $291,718.75    230,249.00      $46,868,840.52   41,129,581.00
LA               $1,433.30        986.00      $27,668,201.82   24,027,600.00
ORL                      .             .      $46,955,479.15   41,143,292.00
SEA                      .             .      $10,906,551.21    9,578,621.00
STL            $560,831.70    458,317.00      $93,875,795.10   81,807,263.00

                          **End of Report**
```

Syntax: How to Identify an Entire Total or Subtotal Row

```
TYPE=type, [BY=sortcolumn]
```

where:

type

Identifies a subtotal or total. Select from:

GRANDTOTAL, which is a grand total (generated by COLUMN-TOTAL, SUBTOTAL, SUB-TOTAL, RECOMPUTE, or SUMMARIZE).

SUBTOTAL, which is a subtotal (generated by SUBTOTAL, SUB-TOTAL, RECOMPUTE, or SUMMARIZE).

RECAP, which is a subtotal calculation (generated by ON *sortfield* RECAP or ON *sortfield* COMPUTE).

BY

When there are several subtotal commands, each associated with a different vertical sort (BY) column, this enables you to identify which of the subtotal commands you wish to format.

sortcolumn

Specifies the vertical sort (BY) column associated with one of the several subtotal in the report commands. Use the field name to identify the sort column.

Example: Identifying an Entire Total Row

The following example illustrates how to identify an entire COLUMN-TOTAL row in a StyleSheet.

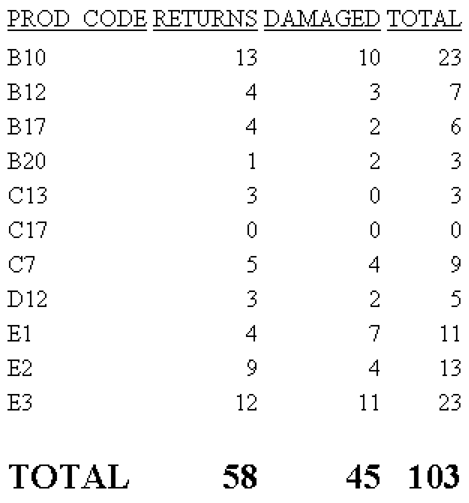
Report Request

```
TABLE FILE SALES
SUM RETURNS DAMAGED AND ROW-TOTAL AND COLUMN-TOTAL
BY PROD_CODE
ON TABLE SET STYLE *
INCLUDE=IBFS:/stylesheet_location/custom_stylesheet.sty,$
END
```

***custom_stylesheet*.sty**

```
TYPE=REPORT, GRID=OFF, $
TYPE=GRANDTOTAL, STYLE=BOLD, SIZE=12, $
```

The output is:



```
PROD_CODE RETURNS DAMAGED TOTAL
B10            13      10    23
B12             4       3     7
B17             4       2     6
B20             1       2     3
C13             3       0     3
C17             0       0     0
C7              5       4     9
D12             3       2     5
E1              4       7    11
E2              9       4    13
E3             12      11    23

TOTAL          58      45   103
```

Example: **Identifying a Row Total**

The following example illustrates how to identify a row total. Note that if you want to format an instance of row-total, you can add a WHEN statement to your StyleSheet. For details, see *Controlling Report Formatting* on page 41.

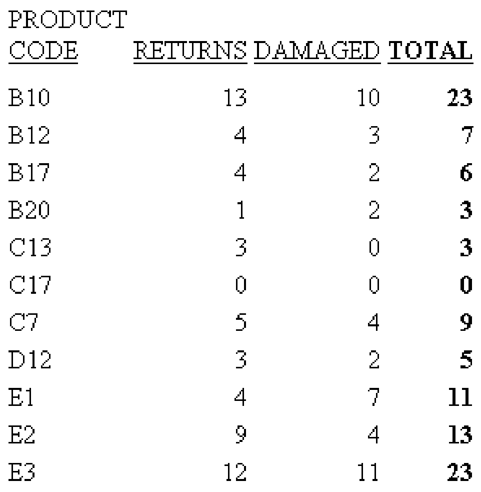
Report Request

```
TABLE FILE SALES
SUM RETURNS DAMAGED AND ROW-TOTAL
BY PROD_CODE AS 'PRODUCT,CODE'
ON TABLE SET STYLE *
INCLUDE=IBFS:/stylesheet_location/custom_stylesheet.sty,$
END
```

***custom_stylesheet*.sty**

```
TYPE=REPORT, GRID=OFF, $
TYPE=REPORT, COLUMN=ROWTOTAL, STYLE=BOLD, $
```

The output is:



```
PRODUCT
CODE        RETURNS DAMAGED TOTAL

B10              13      10    23
B12               4       3     7
B17               4       2     6
B20               1       2     3
C13               3       0     3
C17               0       0     0
C7                5       4     9
D12               3       2     5
E1                4       7    11
E2                9       4    13
E3               12      11    23
```

Identifying Tags for SUBTOTAL and GRANDTOTAL Lines

The *tag* is the text that is displayed in the left-most portion of each SUBTOTAL and GRANDTOTAL row in a report. The tag is used to identify the type of data represented within this row. The text used to generate this tag can be customized by adding an AS name to the SUBTOTAL syntax.

You can define styling for the subtotal and grand total tag separately from the rest of the row. Text attributes available for the tag, including font, color, size, and style, can be used to differentiate and highlight the tags. Additionally, styling can be applied that turns tags into drill-down links.

Styling is supported for text attributes only. Cell or column features, such as borders, background color, or justification are not supported.

This feature is available for PDF, DHTML, HTML, AHTML, PPTX, XLSX, and EXL2K formats.

Syntax: How to Style Subtotal and Grand Total Tags

```
TYPE={SUBTOTAL|GRANDTOTAL}, OBJECT=TAG,
   [FONT=font], [SIZE=size], [STYLE=style],
[COLOR={color|RGB({rgb|#hexcolor})],
   [drilltype=drillparms], $
```

where:

font

Is the name of the font.

size

Is the point size of the font.

style

Is the font style, for example bold, italic, or bold+italic.

color

Is a color name.

rgb

Specifies the font color using a mixture of red, green, and blue.

(r g b) is the desired intensity of red, green, and blue, respectively. The values are on a scale of 0 to 255, where 0 is the least intense and 255 is the most intense. Note that using the three color components in equal intensities results in shades of gray.

#hexcolor

Is the hexadecimal value for the color. For example, FF0000 is the hexadecimal value for red. The hexadecimal digits can be in uppercase or lowercase and must be preceded by a pound sign (#).

drilltype

Is any valid drill-down attribute, for example, URL= or FOCEXEC=.

drillparms

Are valid attribute values for the type of drill down.

Example: Styling SUBTOTAL and GRANDTOTAL Tags

The following request against the GGSALES data source generates subtotal and grand total rows. The tags for the subtotal rows are in italics and are white. The tag for the grand total row has a drill-down link to a URL.

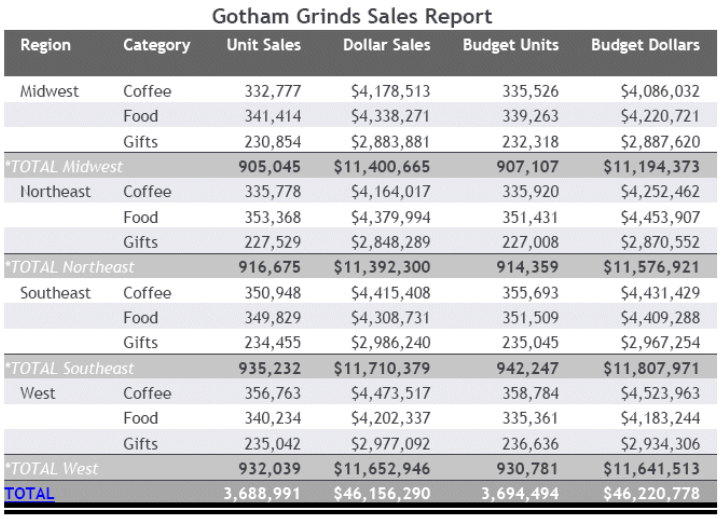
Report Request

```
TABLE FILE GGSALES
SUM UNITS/D8C DOLLARS/D12CM BUDUNIT/D8C BUDDOLLARS/D12CM
BY REGION
BY CATEGORY
ON REGION SUBTOTAL
HEADING
"Gotham Grinds Sales Report"
ON TABLE PCHOLD FORMAT PDF
ON TABLE SET STYLE *
INCLUDE=IBFS:/stylesheet_location/custom_stylesheet.sty,$
END
```

***custom_stylesheet*.sty**

```
TYPE=SUBTOTAL, OBJECT=TAG,STYLE=ITALIC,COLOR=WHITE, $
TYPE=GRANDTOTAL, BACKCOLOR='LIGHT GREY', $
TYPE=GRANDTOTAL, OBJECT=TAG,URL='http://www.ibm.com', $
```

The output is:



Gotham Grinds Sales Report

Region	Category	Unit Sales	Dollar Sales	Budget Units	Budget Dollars
Midwest	Coffee	332,777	$4,178,513	335,526	$4,086,032
	Food	341,414	$4,338,271	339,263	$4,220,721
	Gifts	230,854	$2,883,881	232,318	$2,887,620
**TOTAL Midwest*		**905,045**	**$11,400,665**	**907,107**	**$11,194,373**
Northeast	Coffee	335,778	$4,164,017	335,920	$4,252,462
	Food	353,368	$4,379,994	351,431	$4,453,907
	Gifts	227,529	$2,848,289	227,008	$2,870,552
**TOTAL Northeast*		**916,675**	**$11,392,300**	**914,359**	**$11,576,921**
Southeast	Coffee	350,948	$4,415,408	355,693	$4,431,429
	Food	349,829	$4,308,731	351,509	$4,409,288
	Gifts	234,455	$2,986,240	235,045	$2,967,254
**TOTAL Southeast*		**935,232**	**$11,710,379**	**942,247**	**$11,807,971**
West	Coffee	356,763	$4,473,517	358,784	$4,523,963
	Food	340,234	$4,202,337	335,361	$4,183,244
	Gifts	235,042	$2,977,092	236,636	$2,934,306
**TOTAL West*		**932,039**	**$11,652,946**	**930,781**	**$11,641,513**
TOTAL		**3,688,991**	**$46,156,290**	**3,694,494**	**$46,220,778**

Identifying Data

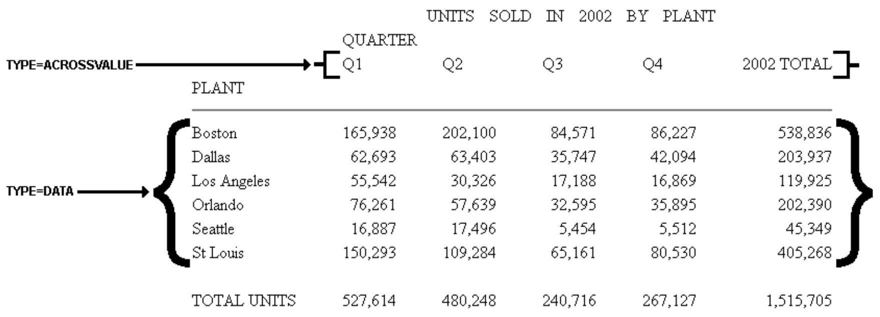
You can identify and format many categories of data in a report, including:

- **All report data**. For more information, see *How to Identify All Data* on page 69.
- **Columns of data**, including sort columns and display columns. For more information, see *How to Identify a Column of Data* on page 70.
- **Sort rows** (that is, ACROSS field values). For more information, see *How to Identify a Row of Horizontal Sort (ACROSS) Data* on page 71.
- **Totals and subtotals**. For more information, see *Identifying Totals and Subtotals* on page 74.

The following example illustrates where the DATA and ACROSSVALUE components appear in a report, and which TYPE values you use to identify them.

Report Request

```
TABLE FILE CENTORD
HEADING CENTER
"UNITS SOLD IN 2002 BY PLANT"
SUM QUANTITY AND ROW-TOTAL AS '2002 TOTAL'
ACROSS QUARTER
BY PLANTLNG AS 'PLANT'
WHERE YEAR EQ 2002
ON TABLE COLUMN-TOTAL AS 'TOTAL UNITS'
ON TABLE SET STYLE *
INCLUDE=IBFS:/stylesheet_location/custom_stylesheet.sty,$
TYPE=REPORT, GRID=OFF, $
END
```



```
                      UNITS  SOLD  IN  2002  BY  PLANT
              QUARTER
              Q1         Q2         Q3         Q4         2002 TOTAL
PLANT
Boston        165,938    202,100    84,571     86,227     538,836
Dallas         62,693     63,403    35,747     42,094     203,937
Los Angeles    55,542     30,326    17,188     16,869     119,925
Orlando        76,261     57,639    32,595     35,895     202,390
Seattle        16,887     17,496     5,454      5,512      45,349
St Louis      150,293    109,284    65,161     80,530     405,268

TOTAL UNITS   527,614    480,248   240,716    267,127   1,515,705
```

Note: Since this request simply illustrates where the components appear in a report, it omits a StyleSheet.

Syntax: How to Identify All Data

To identify all report data in a StyleSheet (except totals, grand totals, subtotals, and horizontal sort (ACROSS) values, which need to be identified separately) use this attribute and value:

```
TYPE=DATA
```

Example: Identifying All Data in a Report

The following example illustrates how to identify all of the data in a report.

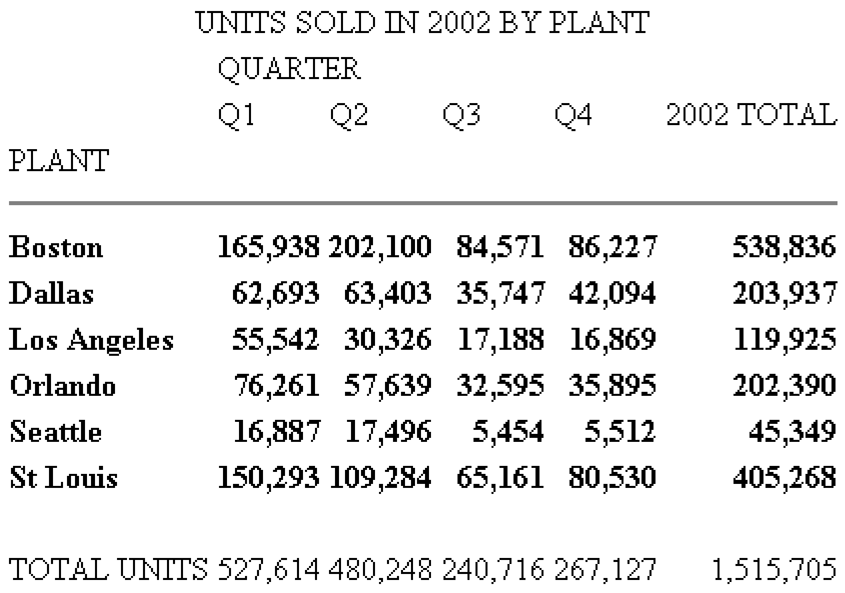
Report Request

```
TABLE FILE CENTORD
HEADING CENTER
"UNITS SOLD IN 2002 BY PLANT"
SUM QUANTITY AND ROW-TOTAL AS '2002 TOTAL'
ACROSS QUARTER
BY PLANTLNG AS 'PLANT'
WHERE YEAR EQ 2002
ON TABLE COLUMN-TOTAL AS 'TOTAL UNITS'
ON TABLE SET STYLE *
INCLUDE=IBFS:/stylesheet_location/custom_stylesheet.sty,$
END
```

***custom_stylesheet*.sty**

```
TYPE=REPORT, GRID=OFF, $
TYPE=DATA, STYLE=BOLD, $
```

The output is:



```
                 UNITS SOLD IN 2002 BY PLANT
                  QUARTER
                  Q1       Q2       Q3       Q4       2002 TOTAL
PLANT

Boston          165,938  202,100   84,571   86,227      538,836
Dallas           62,693   63,403   35,747   42,094      203,937
Los Angeles      55,542   30,326   17,188   16,869      119,925
Orlando          76,261   57,639   32,595   35,895      202,390
Seattle          16,887   17,496    5,454    5,512       45,349
St Louis        150,293  109,284   65,161   80,530      405,268

TOTAL UNITS     527,614  480,248  240,716  267,127    1,515,705
```

Syntax: **How to Identify a Column of Data**

```
TYPE=DATA, COLUMN=column
```

where:

`column`

Specifies one or more columns that you wish to format.

Example: **Identifying a Column of Data**

The following example illustrates how to identify a column of data.

Note that when identifying a column using N*n*, NOPRINT columns are counted. Even though the Product Name field is the first column in this report, it is identified with N2 because of the NOPRINT column.

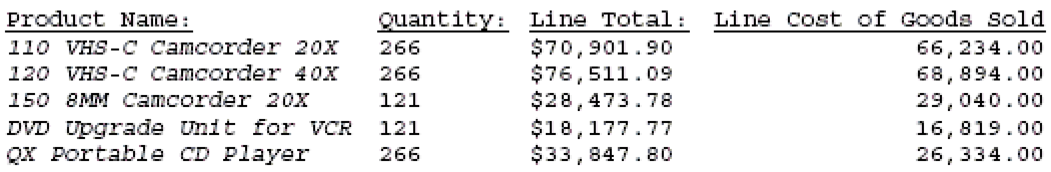
Report Request

```
TABLE FILE CENTORD
PRINT QUANTITY LINEPRICE LINE_COGS
BY ORDER_NUM NOPRINT
BY PRODNAME
WHERE ORDER_NUM EQ '48045'
ON TABLE SET PAGE-NUM OFF
ON TABLE SET STYLE *
INCLUDE=IBFS:/stylesheet_location/custom_stylesheet.sty,$
END
```

***custom_stylesheet*.sty**

```
TYPE=REPORT, GRID=OFF, $
TYPE=DATA, COLUMN=N2, STYLE=ITALIC, $
```

The output is:



```
Product Name:              Quantity:  Line Total:  Line Cost of Goods Sold
110 VHS-C Camcorder 20X    266        $70,901.90                 66,234.00
120 VHS-C Camcorder 40X    266        $76,511.09                 68,894.00
150 8MM Camcorder 20X      121        $28,473.78                 29,040.00
DVD Upgrade Unit for VCR   121        $18,177.77                 16,819.00
QX Portable CD Player      266        $33,847.80                 26,334.00
```

Syntax: How to Identify a Row of Horizontal Sort (ACROSS) Data

```
TYPE=ACROSSVALUE, [ACROSS=fieldname]
```

where:

`ACROSS`

If you have a request with multiple ACROSS fields, you can identify each field using the ACROSS identifier. You only need to include the ACROSS identifier if you have multiple ACROSS fields in your request.

`fieldname`

Specifies a horizontal sort row by its field name.

Example: Identifying a Row of Horizontal Sort (ACROSS) Data

The following example illustrates how to identify a row of horizontal sort data.

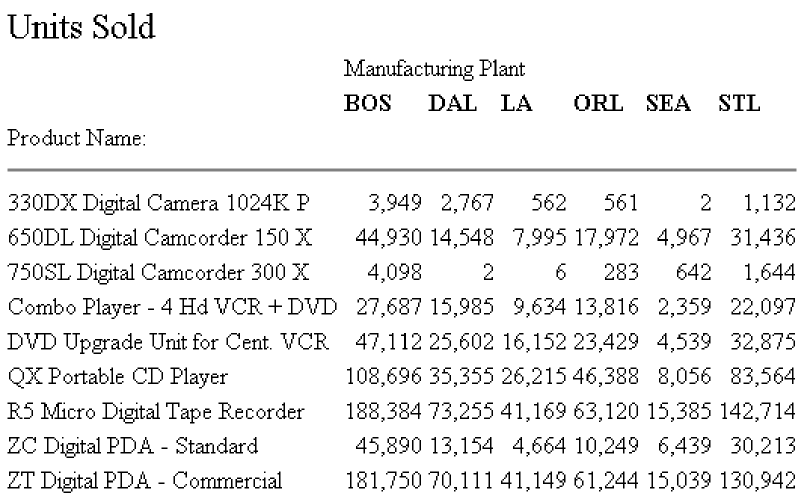
Report Request

```
TABLE FILE CENTORD
HEADING
"Units Sold"
SUM QUANTITY
BY PRODNAME
ACROSS PLANT AS 'Manufacting Plant'
WHERE PRODTYPE EQ 'Digital'
ON TABLE SET STYLE *
INCLUDE=IBFS:/stylesheet_location/custom_stylesheet.sty,$
END
```

***custom_stylesheet*.sty**

```
TYPE=REPORT, GRID=OFF, $
TYPE=HEADING, SIZE=12, $
TYPE=ACROSSVALUE, ACROSS=PLANT, STYLE=BOLD, $
```

The output is:



Units Sold

	Manufacturing Plant					
	BOS	**DAL**	**LA**	**ORL**	**SEA**	**STL**
Product Name:						
330DX Digital Camera 1024K P	3,949	2,767	562	561	2	1,132
650DL Digital Camcorder 150 X	44,930	14,548	7,995	17,972	4,967	31,436
750SL Digital Camcorder 300 X	4,098	2	6	283	642	1,644
Combo Player - 4 Hd VCR + DVD	27,687	15,985	9,634	13,816	2,359	22,097
DVD Upgrade Unit for Cent. VCR	47,112	25,602	16,152	23,429	4,539	32,875
QX Portable CD Player	108,696	35,355	26,215	46,388	8,056	83,564
R5 Micro Digital Tape Recorder	188,384	73,255	41,169	63,120	15,385	142,714
ZC Digital PDA - Standard	45,890	13,154	4,664	10,249	6,439	30,213
ZT Digital PDA - Commercial	181,750	70,111	41,149	61,244	15,039	130,942

Example: Identifying Row Totals (ACROSS-TOTAL) for Horizontal Sort Data

The following example illustrates how to identify a row total (ACROSS-TOTAL) for horizontal sort (ACROSS) data using the ACROSSVALUE component and a numeric column reference (N*n*).

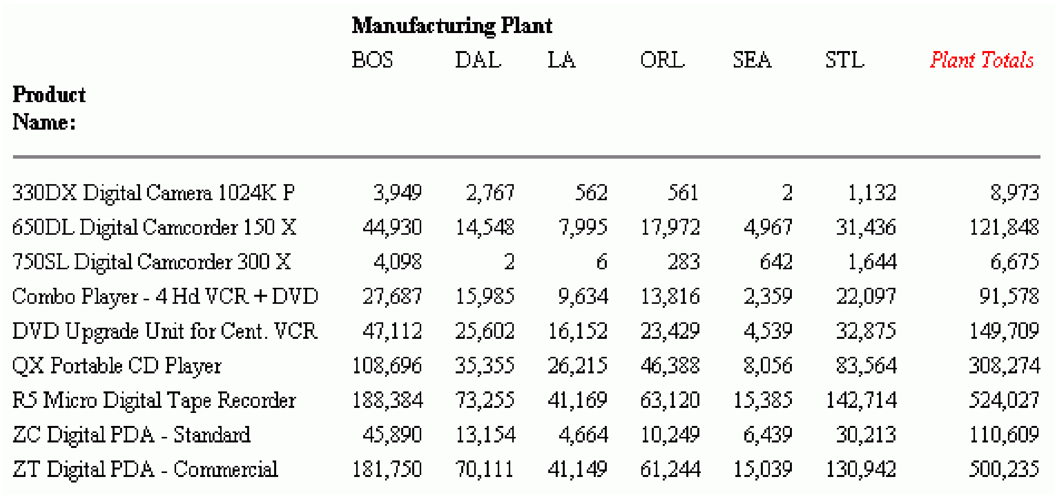
Report Request

```
TABLE FILE CENTORD
SUM QUANTITY
BY PRODNAME
ACROSS PLANT AS 'Manufacturing Plant'
ACROSS-TOTAL AS 'Plant Totals'
WHERE PRODTYPE EQ 'Digital'
ON TABLE SET STYLE *
INCLUDE=IBFS:/stylesheet_location/custom_stylesheet.sty,$
END
```

***custom_stylesheet*.sty**

```
TYPE=REPORT, GRID=OFF, $
TYPE=ACROSSVALUE, COLUMN=N8, STYLE=ITALIC, COLOR='RED', $
```

The following image shows the output with the ACROSS-TOTAL value, Plant Totals, styled in red italics.



	Manufacturing Plant						
	BOS	DAL	LA	ORL	SEA	STL	*Plant Totals*
Product Name:							
330DX Digital Camera 1024K P	3,949	2,767	562	561	2	1,132	8,973
650DL Digital Camcorder 150 X	44,930	14,548	7,995	17,972	4,967	31,436	121,848
750SL Digital Camcorder 300 X	4,098	2	6	283	642	1,644	6,675
Combo Player - 4 Hd VCR + DVD	27,687	15,985	9,634	13,816	2,359	22,097	91,578
DVD Upgrade Unit for Cent. VCR	47,112	25,602	16,152	23,429	4,539	32,875	149,709
QX Portable CD Player	108,696	35,355	26,215	46,388	8,056	83,564	308,274
R5 Micro Digital Tape Recorder	188,384	73,255	41,169	63,120	15,385	142,714	524,027
ZC Digital PDA - Standard	45,890	13,154	4,664	10,249	6,439	30,213	110,609
ZT Digital PDA - Commercial	181,750	70,111	41,149	61,244	15,039	130,942	500,235

Identifying Totals and Subtotals

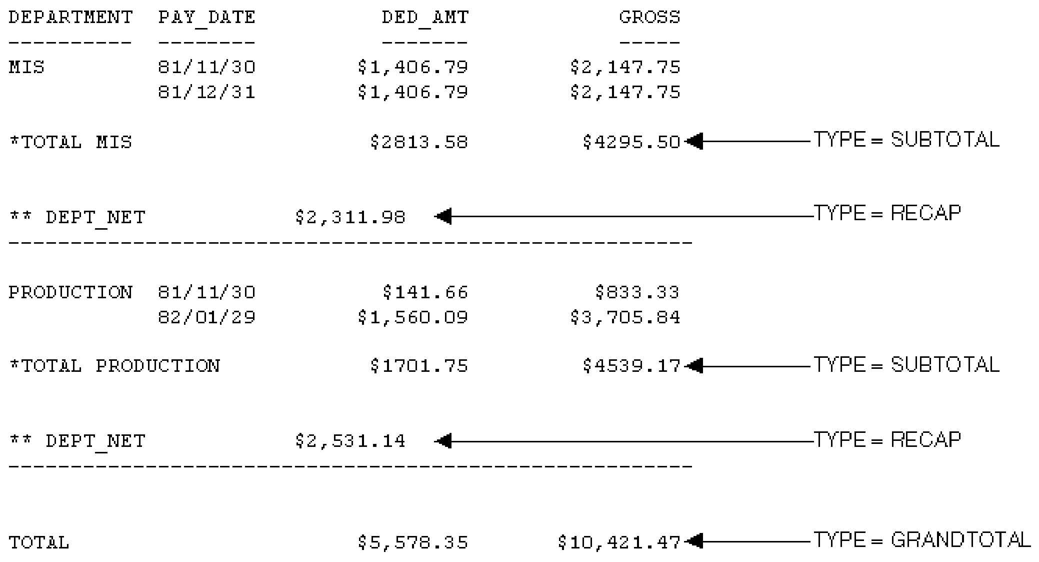
Within a StyleSheet, you can identify the grand totals, subtotals, subtotal calculations (generated by ON *sortfield* RECAP or ON *sortfield* COMPUTE), column totals, and row totals of a report in order to format them. For details on identifying row totals, see *Identifying an Entire Report, Column, or Row* on page 57.

The following example illustrates where these components are in a report, and which TYPE values you use to identify them.

Report Request

```
TABLE FILE EMPLOYEE
SUM DED_AMT AND GROSS
BY DEPARTMENT BY PAY_DATE
ON DEPARTMENT RECAP DEPT_NET/D8.2M = GROSS-DED_AMT;
WHEN PAY_DATE GT 820101
ON DEPARTMENT SUBTOTAL
END
```

The following image shows each component:



```
DEPARTMENT  PAY_DATE          DED_AMT            GROSS
----------  --------          -------            -----
MIS         81/11/30        $1,406.79        $2,147.75
            81/12/31        $1,406.79        $2,147.75

*TOTAL MIS                   $2813.58         $4295.50

** DEPT_NET           $2,311.98
---------------------------------------------------------

PRODUCTION  81/11/30          $141.66          $833.33
            82/01/29        $1,560.09        $3,705.84

*TOTAL PRODUCTION            $1701.75         $4539.17

** DEPT_NET           $2,531.14
---------------------------------------------------------

TOTAL                       $5,578.35       $10,421.47
```

Note: Since this request simply illustrates how to identify different types of totals and subtotals, it omits a StyleSheet.

Syntax: How to Identify a Grand Total, Subtotal, or Subtotal Calculation

```
TYPE=type, [BY=sortfield] [coltype=column]
```

where:

type

Identifies a subtotal or total. Select from:

GRANDTOTAL, which is a grand total (generated by COLUMN-TOTAL, SUBTOTAL, SUB-TOTAL, RECOMPUTE, or SUMMARIZE).

SUBTOTAL, which is a subtotal (generated by SUBTOTAL, SUB-TOTAL, RECOMPUTE, or SUMMARIZE).

RECAP, which is a subtotal calculation (generated by ON *sortfield* RECAP or ON *sortfield* COMPUTE).

BY

If you have a request with multiple BY fields, and two or more have subtotal commands associated with them, you can identify each field using the BY identifier. This is helpful when you want to format each subtotal differently or when you want to format only one subtotal.

You only need to include the BY identifier if you have multiple BY fields in your request.

sortfield

Specifies the BY field associated with several subtotal commands of a report. Use the fieldname for the value (BY=*fieldname*).

coltype

Identifies a specific column to apply formatting. When you include the COLUMN or ACROSSCOLUMN identifier in your declaration, only the subtotal *values* receive the formatting, the labeling text will not. Values can be:

COLUMN, which is a display column (generated by PRINT, LIST, SUM, or COUNT) or a computed column (generated by COMPUTE).

ACROSSCOLUMN, where every instance of a display or computed column is repeated across a horizontal sort (ACROSS) row.

If there are several columns being totaled or subtotaled by one command, and you do not specify a column in the StyleSheet, the formatting will be applied to the totals or subtotals for *all* of the columns. It will also be applied to the labeling text for the total and subtotal values.

column

Specifies the column whose totals or subtotals you wish to format. For a list of values, see *How to Identify an Entire Column* on page 60.

Example: Identifying a Grand Total

The following example illustrates how to identify a grand total in a report request. In this example, we only want to format the grand total value for the LINE_COGS field, so the COLUMN attribute is included in the StyleSheet declaration. The grand total in this request is generated by COLUMN-TOTAL.

Note: To style the entire grand total row, remove the COLUMN attribute from the StyleSheet declaration.

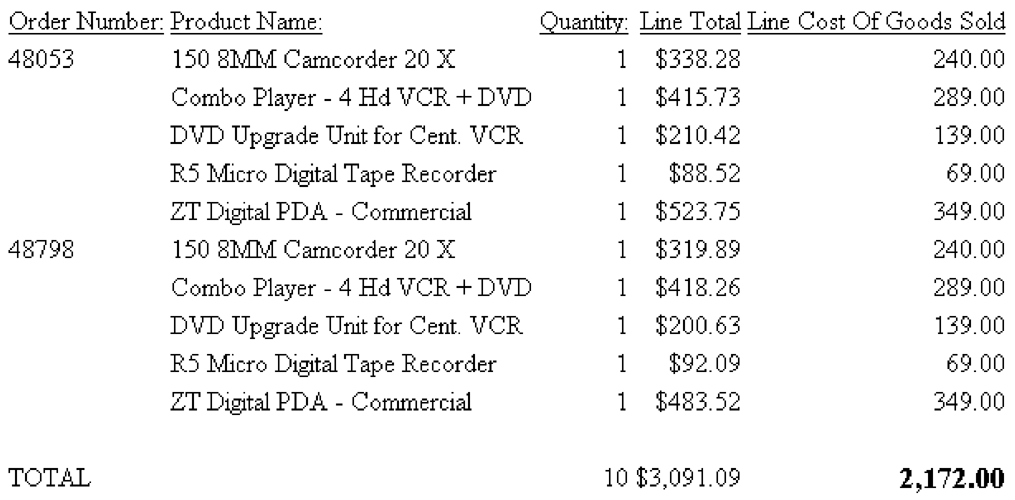
Report Request

```
TABLE FILE CENTORD
SUM QUANTITY LINEPRICE LINE_COGS AND COLUMN-TOTAL
BY ORDER_NUM BY PRODNAME
WHERE ORDER_NUM EQ '48053' OR '48798'
ON TABLE SET STYLE *
INCLUDE=IBFS:/stylesheet_location/custom_stylesheet.sty,$
END
```

***custom_stylesheet*.sty**

```
TYPE=REPORT, GRID=OFF, $
TYPE=GRANDTOTAL, COLUMN=LINE_COGS, STYLE=BOLD, SIZE=11, $
```

The output is:



```
Order Number: Product Name:                  Quantity: Line Total Line Cost Of Goods Sold
48053         150 8MM Camcorder 20 X                 1    $338.28                  240.00
              Combo Player - 4 Hd VCR + DVD          1    $415.73                  289.00
              DVD Upgrade Unit for Cent. VCR         1    $210.42                  139.00
              R5 Micro Digital Tape Recorder         1     $88.52                   69.00
              ZT Digital PDA - Commercial            1    $523.75                  349.00
48798         150 8MM Camcorder 20 X                 1    $319.89                  240.00
              Combo Player - 4 Hd VCR + DVD          1    $418.26                  289.00
              DVD Upgrade Unit for Cent. VCR         1    $200.63                  139.00
              R5 Micro Digital Tape Recorder         1     $92.09                   69.00
              ZT Digital PDA - Commercial            1    $483.52                  349.00

TOTAL                                               10  $3,091.09                **2,172.00**
```

Example: **Identifying Subtotals**

The following example illustrates how to identify subtotals in a report request. In this example, only subtotal values in the QUANTITY and LINE_COGS fields are formatted, so the COLUMN attribute is included in the StyleSheet declarations.

Also, since there are two SUBTOTAL commands associated with two of the three BY fields (PLANT and ORDER_NUM), the BY attribute is also included in each declaration to ensure the formatting is applied to the correct value.

Note: To style an entire subtotal row, remove the COLUMN and BY attributes from the StyleSheet declaration.

Report Request

```
TABLE FILE CENTORD
SUM QUANTITY LINEPRICE LINE_COGS AS 'Line Cost of, Goods Sold'
BY PLANT
BY ORDER_NUM
BY PRODNAME
ON PLANT SUBTOTAL
ON ORDER_NUM SUBTOTAL
WHERE ORDER_NUM EQ '35774' OR '48041'
WHERE PLANT EQ 'BOS'
ON TABLE SET STYLE *
INCLUDE=IBFS:/stylesheet_location/custom_stylesheet.sty,$
END
```

***custom_stylesheet*.sty**

```
TYPE=REPORT, GRID=OFF, $
TYPE=SUBTOTAL, BY=PLANT, COLUMN=LINE_COGS, STYLE=BOLD+ITALIC, COLOR=BLUE, $
TYPE=SUBTOTAL, BY=ORDER_NUM, COLUMN=QUANTITY, STYLE=BOLD, SIZE=11, $
```

The output is:

```
                                                                                  Line Cost of
Manufacturing Plant Order Number: Product Name:                 Quantity:   Line Total  Goods Sold
BOS                 35774         110 VHS-C Camcorder 20 X            146   $38,817.56   36,354.00
                                  120 VHS-C Camcorder 40 X            146   $40,426.90   37,814.00
                                  2 Hd VCR LCD Menu                   279   $39,693.32   35,991.00
                                  650DL Digital Camcorder 150 X       146  $116,907.15  103,660.00

*TOTAL ORDER_NUM 35774                                                717  $235,844.93  213,819.00

                    48041         150 8MM Camcorder 20 X                1      $307.43      240.00
                                  Combo Player - 4 Hd VCR + DVD         1      $398.95      289.00
                                  DVD Upgrade Unit for Cent. VCR        1      $194.40      139.00
                                  R5 Micro Digital Tape Recorder        1       $93.91       69.00
                                  ZT Digital PDA - Commercial           1      $528.38      349.00

*TOTAL ORDER_NUM 48041                                                  5    $1,523.07    1,086.00
*TOTAL PLANT BOS                                                      722  $237,368.00  214,905.00


TOTAL                                                                 722  $237,368.00  214,905.00
```

Example: Identifying a Subtotal Calculation (RECAP/COMPUTE)

The following example illustrates how to identify a subtotal calculation created with a RECAP or COMPUTE phrase. In this example, the subtotal calculation is generated with ON PLANT RECAP QTY/F6=QUANTITY.

Note: If there is more than one RECAP or COMPUTE field in your request, you can distinguish them by adding BY=*fieldname* to the StyleSheet declaration.

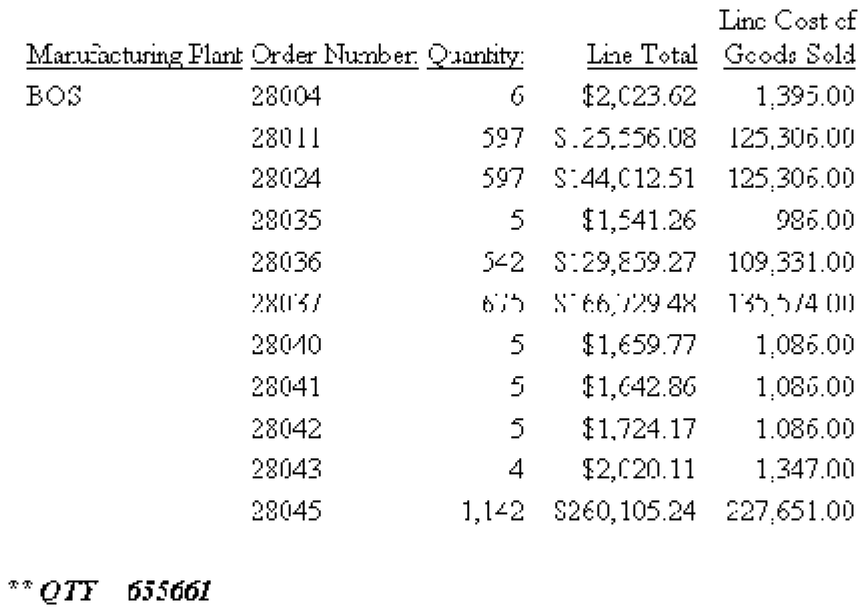
Report Request

```
TABLE FILE CENTORD
SUM QUANTITY LINEPRICE LINE_COGS AS 'Line Cost of, Goods Sold'
BY PLANT BY ORDER_NUM
ON PLANT RECAP QTY/F6=QUANTITY;
WHERE PLANT EQ 'BOS'
WHERE ORDER_NUM LT '56098'
ON TABLE SET STYLE *
INCLUDE=IBFS:/stylesheet_location/custom_stylesheet.sty,$
END
```

***custom_stylesheet*.sty**

```
TYPE=REPORT, GRID=OFF, $
TYPE=RECAP, STYLE=BOLD+ITALIC, $
```

The output is:



Manufacturing Plant	Order Number	Quantity:	Line Total	Line Cost of Goods Sold
BOS	28004	6	$2,023.62	1,395.00
	28011	597	$125,556.08	125,306.00
	28024	597	$144,012.51	125,306.00
	28035	5	$1,541.26	986.00
	28036	542	$129,859.27	109,331.00
	28037	675	$166,729.48	135,574.00
	28040	5	$1,659.77	1,086.00
	28041	5	$1,642.86	1,086.00
	28042	5	$1,724.17	1,086.00
	28043	4	$2,020.11	1,347.00
	28045	1,142	$260,105.24	227,651.00

***** QTY   655661***

Identifying a Heading, Footing, or Title

A report data is framed by headings, footings, and titles. These provide context for the data. You can identify and format many categories of headings, footings, and titles in a report, including:

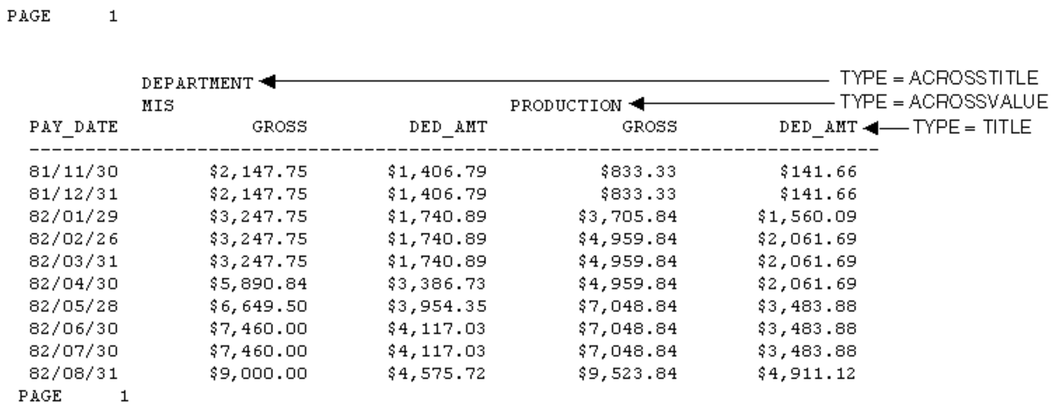
- Report, page, and sort headings.
- Report, page, and sort footings.
- Column titles.
- Horizontal sort (ACROSS) titles and values.

Identifying a Column or Row Title

Within a StyleSheet you can identify column titles and horizontal sort (ACROSS) values of a report in order to format them. The following example illustrates where column titles and horizontal sort values are in a report, and which TYPE values you use to identify them.

Report Request

```
TABLE FILE EMPLOYEE
SUM GROSS AND DED_AMT
ACROSS DEPARTMENT BY PAY_DATE
END
```



```
PAGE     1

            DEPARTMENT
            MIS                                  PRODUCTION
  PAY_DATE            GROSS           DED_AMT             GROSS          DED_AMT
  ---------------------------------------------------------------------------
  81/11/30        $2,147.75         $1,406.79           $833.33          $141.66
  81/12/31        $2,147.75         $1,406.79           $833.33          $141.66
  82/01/29        $3,247.75         $1,740.89         $3,705.84        $1,560.09
  82/02/26        $3,247.75         $1,740.89         $4,959.84        $2,061.69
  82/03/31        $3,247.75         $1,740.89         $4,959.84        $2,061.69
  82/04/30        $5,890.84         $3,386.73         $4,959.84        $2,061.69
  82/05/28        $6,649.50         $3,954.35         $7,048.84        $3,483.88
  82/06/30        $7,460.00         $4,117.03         $7,048.84        $3,483.88
  82/07/30        $7,460.00         $4,117.03         $7,048.84        $3,483.88
  82/08/31        $9,000.00         $4,575.72         $9,523.84        $4,911.12
 PAGE     1
```

Note: Since this request simply illustrates how to identify column titles and horizontal sort values in a report, it omits a StyleSheet.

Syntax: How to Identify a Column Title

```
TYPE=TITLE, [COLUMN=column]
```

where:

`COLUMN`

Is used to specify one or more column titles. If you omit this attribute and value, the formatting will be applied to all of the report column titles.

`column`

Specifies the column whose title you wish to format. For column values, see *How to Identify an Entire Column* on page 60.

Syntax: How to Identify a Horizontal Sort Title or Value

```
TYPE={ACROSSTITLE|ACROSSVALUE}, [ACROSS=column]
```

where:

`ACROSSTITLE`

Specifies a horizontal sort (ACROSS) title.

`ACROSSVALUE`

Specifies a horizontal sort (ACROSS) value.

Although horizontal sort values are not technically titles, they often function as titles that categorize the column titles appearing beneath them.

`ACROSS`

Is used to specify titles or values for a specific horizontal sort field. If you omit this attribute and value, the formatting will be applied to the titles or values of the horizontal sort fields of all reports.

`column`

Specifies the horizontal sort (ACROSS) field whose title or values you wish to format. For values you can assign to this attribute, see *How to Identify a Row of Horizontal Sort (ACROSS) Data* on page 71.

Example: Identifying Column Titles and Horizontal Sort (ACROSS) Values

The following example illustrates how to identify vertical sort titles, horizontal sort titles, and horizontal sort values. The vertical sort titles (TYPE=TITLE) are Manufacturing Plant, Quantity Sold, and Product Cost, the horizontal sort title (TYPE=ACROSSTITLE) is Year, and the horizontal sort values (TYPE=ACROSSVALUE) are 2001, 2002, and TOTAL.

The following example also demonstrates how to assign drill-down values to the individual ACROSS values of 2001 and 2002, and not the ROW-TOTAL value of TOTAL.

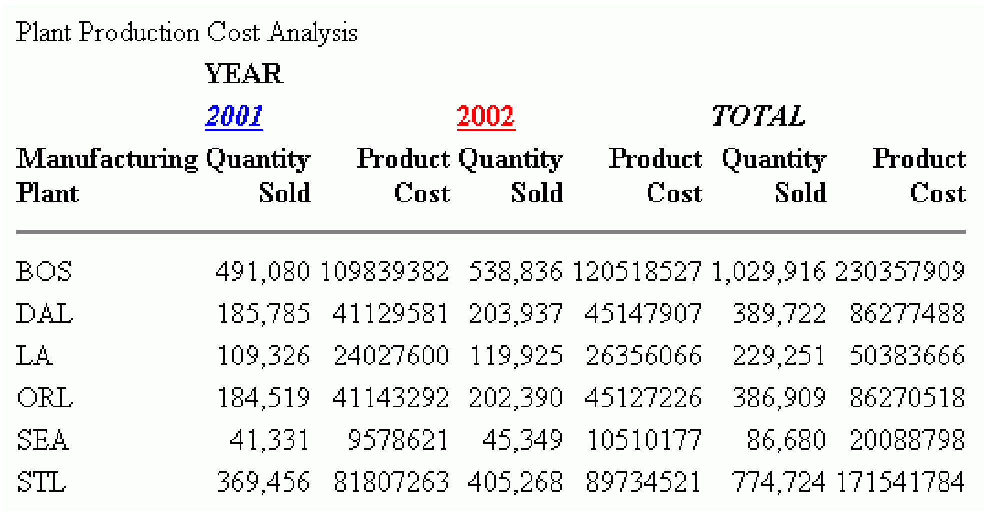
Report Request

```
TABLE FILE CENTORD
SUM QUANTITY AS 'Quantity,Sold' LINE_COGS/I9 AS 'Product,Cost'
BY PLANT
ACROSS YEAR
WHERE YEAR EQ '2001' OR '2002'
HEADING
"Plant Production Cost Analysis"
ON TABLE ROW-TOTAL AS 'TOTAL'
ON TABLE SET STYLE *
INCLUDE=IBFS:/stylesheet_location/custom_stylesheet.sty,$
END
```

***custom_stylesheet*.sty**

```
TYPE=REPORT, GRID=OFF, $
TYPE=TITLE, STYLE=BOLD, $
TYPE=ACROSSTITLE, STYLE=BOLD, $
TYPE=ACROSSVALUE, STYLE=BOLD+ITALIC, COLOR=BLUE, FOCEXEC=DETAILS, $
TYPE=ACROSSVALUE, COLUMN=N4, STYLE=BOLD, COLOR=RED, $
TYPE=ACROSSVALUE, COLUMN=ROWTOTAL(1), COLOR='BLACK',FOCEXEC=NONE, $
```

The following image shows the report output.



Plant Production Cost Analysis

	YEAR					
	2001		**2002**		***TOTAL***	
Manufacturing Plant	**Quantity Sold**	**Product Cost**	**Quantity Sold**	**Product Cost**	**Quantity Sold**	**Product Cost**
BOS	491,080	109839382	538,836	120518527	1,029,916	230357909
DAL	185,785	41129581	203,937	45147907	389,722	86277488
LA	109,326	24027600	119,925	26356066	229,251	50383666
ORL	184,519	41143292	202,390	45127226	386,909	86270518
SEA	41,331	9578621	45,349	10510177	86,680	20088798
STL	369,456	81807263	405,268	89734521	774,724	171541784

Identifying a Heading or Footing

Within a StyleSheet you can identify report headings and footings, and the individual lines, text strings, and fields within them, in order to format them.

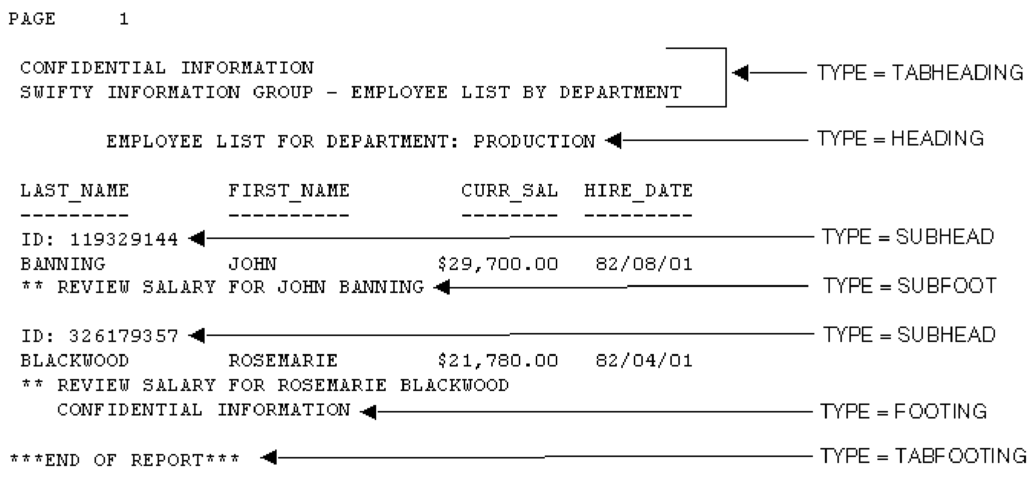
A report request can have more than one page heading or footing. For each heading or footing, a WHEN clause against the data being retrieved can determine whether the heading or footing displays on the report output. The CONDITION StyleSheet attribute enables you to identify a specific WHEN clause so that you can style each heading or footing separately.

The following examples illustrate where a report heading (TABHEADING), a page heading (HEADING), a sort heading (SUBHEAD), a sort footing (SUBFOOT), and a report footing (TABFOOTING) are in a report, and which TYPE values you use to identify them.

Report Request

```
TABLE FILE EMPLOYEE
PRINT CURR_SAL HIRE_DATE
BY LAST_NAME
BY FIRST_NAME
ON TABLE SUBHEAD
"CONFIDENTIAL INFORMATION"
"SWIFTY INFORMATION GROUP - EMPLOYEE LIST BY DEPARTMENT"
HEADING CENTER
"</1>EMPLOYEE LIST FOR DEPARTMENT: <DEPARTMENT"
ON LAST_NAME SUBHEAD
"ID: <EMP_ID"
ON LAST_NAME SUBFOOT
"** REVIEW SALARY FOR <FIRST_NAME <LAST_NAME"
FOOTING
"CONFIDENTIAL INFORMATION"
ON TABLE SUBFOOT
"</1>***END OF REPORT***"
END
```

The output is:



```
PAGE      1

 CONFIDENTIAL INFORMATION
 SWIFTY INFORMATION GROUP - EMPLOYEE LIST BY DEPARTMENT

        EMPLOYEE LIST FOR DEPARTMENT: PRODUCTION

 LAST_NAME         FIRST_NAME          CURR_SAL   HIRE_DATE
 ---------         ----------          --------   ---------
 ID: 119329144
 BANNING           JOHN              $29,700.00    82/08/01
 ** REVIEW SALARY FOR JOHN BANNING

 ID: 326179357
 BLACKWOOD         ROSEMARIE         $21,780.00    82/04/01
 ** REVIEW SALARY FOR ROSEMARIE BLACKWOOD
    CONFIDENTIAL INFORMATION

***END OF REPORT***
```

Note: Since this request simply illustrates how to identify different types of headings and footings, it omits a StyleSheet.

Syntax: How to Identify a Heading or Footing

```
TYPE=headfoot, [BY=sortcolumn]
```

where:

headfoot

Identifies a heading or footing. Select from:

- **TABHEADING,** which is a report heading. This appears once at the beginning of the report and is generated by ON TABLE SUBHEAD.
- **TABFOOTING**, which is a report footing. This appears once at the end of the report and is generated by ON TABLE SUBFOOT.
- **HEADING,** which is a page heading. This appears at the top of every report page and is generated by HEADING.
- **FOOTING,** which is a page footing. This appears at the bottom of every report page and is generated by FOOTING.
- **SUBHEAD,** which is a sort heading. This appears at the beginning of a vertical (BY) sort group (generated by ON *sortfield* SUBHEAD).
- **SUBFOOT,** which is a sort footing. This appears at the end of a vertical (BY) sort group (generated by ON *sortfield* SUBFOOT).

BY

When there are several sort headings or sort footings, each associated with a different vertical sort (BY) column, this enables you to identify which sort heading or sort footing you wish to format.

If there are several sort headings or sort footings associated with different vertical sort (BY) columns, and you omit this attribute and value, the formatting will be applied to all of the sort headings or footings.

sortcolumn

Specifies the vertical sort (BY) column associated with one of the report sort headings or sort footings.

Example: Identifying a Page Heading and a Report Footing

The following example illustrates how to identify a page heading, which appears at the top of every report page, and a report footing, which appears only on the last page of the report.

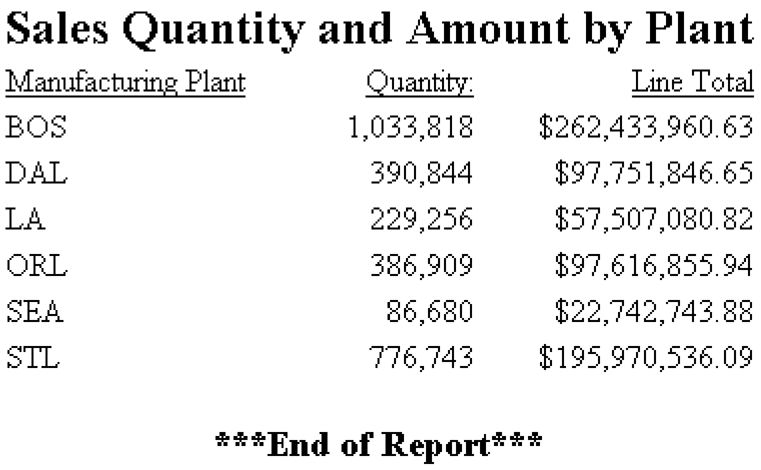
Report Request

```
TABLE FILE CENTORD
HEADING
"Sales Quantity and Amount by Plant"
SUM QUANTITY LINEPRICE
BY PLANT
ON TABLE SUBFOOT
" "
"***End of Report***"
ON TABLE SET STYLE *
INCLUDE=IBFS:/stylesheet_location/custom_stylesheet.sty,$
END
```

***custom_stylesheet*.sty**

```
TYPE=REPORT, GRID=OFF, $
TYPE=HEADING, FONT=TIMES, SIZE=12, STYLE=BOLD, $
TYPE=TABFOOTING, JUSTIFY=CENTER, STYLE=BOLD, SIZE=11, $
```

The output is:



```
Sales Quantity and Amount by Plant
Manufacturing Plant        Quantity:        Line Total
BOS                        1,033,818   $262,433,960.63
DAL                          390,844    $97,751,846.65
LA                           229,256    $57,507,080.82
ORL                          386,909    $97,616,855.94
SEA                           86,680    $22,742,743.88
STL                          776,743   $195,970,536.09

              ***End of Report***
```

Syntax: How to Identify an Individual Line in a Heading or Footing

```
TYPE=type, LINE=line_#
```

where:

type

Identifies a type of heading or footing. Select from HEADING, FOOTING, TABHEADING, TABFOOTING, SUBHEAD, or SUBFOOT. For details, see *How to Identify a Heading or Footing* on page 86.

line_#

Identifies a line by its position in the heading or footing.

Example: Identifying an Individual Line in a Heading

The following example illustrates how to format individual lines in a heading. Heading line 1 (Sales Quantity Analysis) is formatted in bold, point-size 11. Heading line 2 (**Confidential**) is formatted in bold and red.

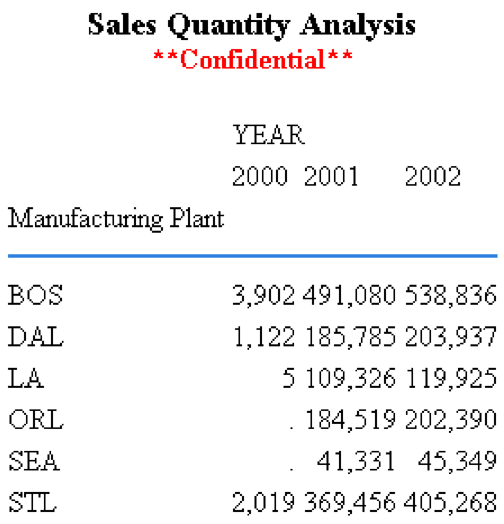
Report Request

```
TABLE FILE CENTORD
HEADING
"Sales Quantity Analysis"
"**Confidential**"
" "
SUM QUANTITY
ACROSS YEAR
BY PLANT
ON TABLE SET STYLE *
INCLUDE=IBFS:/stylesheet_location/custom_stylesheet.sty,$
END
```

***custom_stylesheet*.sty**

```
TYPE=REPORT, GRID=OFF, $
TYPE=HEADING, LINE=1, SIZE=11, STYLE=BOLD, $
TYPE=HEADING, LINE=2, COLOR=RED, STYLE=BOLD, $
TYPE=HEADING, JUSTIFY=CENTER, $
```

The output is:



```
        Sales Quantity Analysis
            **Confidential**

                        YEAR
                        2000  2001    2002
Manufacturing Plant

BOS                     3,902 491,080 538,836
DAL                     1,122 185,785 203,937
LA                          5 109,326 119,925
ORL                         . 184,519 202,390
SEA                         .  41,331  45,349
STL                     2,019 369,456 405,268
```

Syntax: How to Identify a Text String in a Heading or Footing

```
TYPE=type, [LINE=line_#], [OBJECT=TEXT], ITEM=item_#
```

where:

type

Identifies a type of heading or footing. Select from HEADING, FOOTING, TABHEADING, TABFOOTING, SUBHEAD, or SUBFOOT. For details, see *Identifying a Heading or Footing* on page 84.

line_#

Identifies a line by its position in the heading or footing. You only need to include the LINE attribute if you have a multi-line heading or footing.

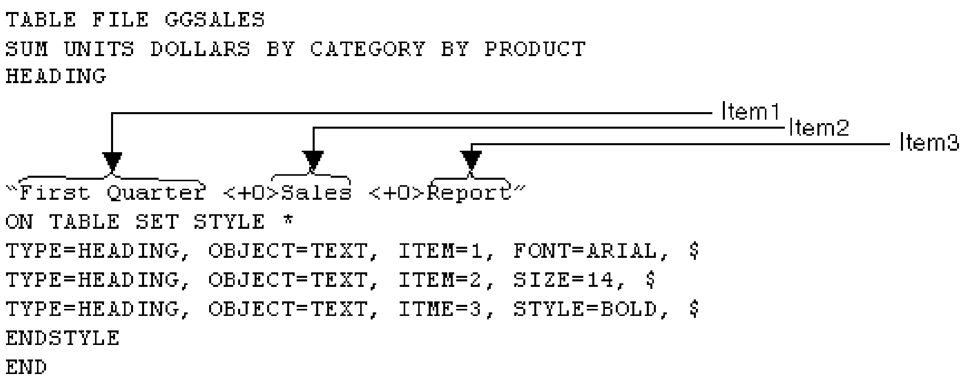
TEXT

Formats only text strings. It is not necessary to use OBJECT=TEXT in your declaration, unless you are styling both text strings and embedded fields in the same heading or footing.

item_#

Identifies an item by its position in a line.

If you need to apply formatting to several parts of a continuous text string that displays on one line, you can break the header or footer into multiple parts using spot markers. Place the spot marker after the text string you wish to specify. The <+0> spot marker will not add any additional spaces to your heading or footing. When using spot markers, text is divided as follows:



```
TABLE FILE GGSALES
SUM UNITS DOLLARS BY CATEGORY BY PRODUCT
HEADING
"First Quarter <+0>Sales <+0>Report"
ON TABLE SET STYLE *
TYPE=HEADING, OBJECT=TEXT, ITEM=1, FONT=ARIAL, $
TYPE=HEADING, OBJECT=TEXT, ITEM=2, SIZE=14, $
TYPE=HEADING, OBJECT=TEXT, ITME=3, STYLE=BOLD, $
ENDSTYLE
END
```

Note: When a closing spot marker is immediately followed by an opening spot marker (><), a single space text item will be placed between the two spot markers (> <). This must be considered when applying formatting.

The position value also depends on whether you are using the OBJECT attribute or not. If you are using:

- OBJECT=TEXT, count only text strings from left to right.
- No OBJECT, count text strings and embedded field values from left to right.

Example: Identifying a Text String in a Heading Using Spot Markers

The following example illustrates how to apply different formats to text strings in a heading using spot markers. The spot markers used in this example are <+0> since they do not add any spaces.

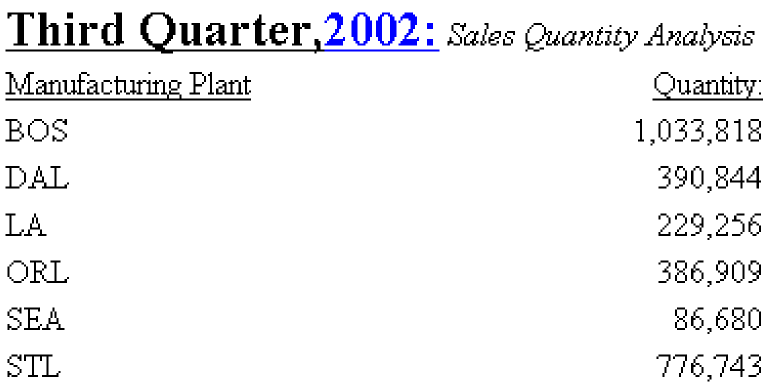
Report Request

```
TABLE FILE CENTORD
HEADING
"Third Quarter,<+0>2002:<+0> Sales Quantity Analysis"
SUM QUANTITY  BY PLANT
ON TABLE SET STYLE *
INCLUDE=IBFS:/stylesheet_location/custom_stylesheet.sty,$
END
```

***custom_stylesheet*.sty**

```
TYPE=REPORT, GRID=OFF, $
TYPE=HEADING, OBJECT=TEXT, ITEM=1, STYLE=BOLD+UNDERLINE, SIZE=12, $
TYPE=HEADING, OBJECT=TEXT, ITEM=2, COLOR=BLUE, SIZE=12,
   STYLE=BOLD+UNDERLINE, $
TYPE=HEADING, OBJECT=TEXT, ITEM=3, STYLE=ITALIC, $
```

The output is:



```
Third Quarter,2002: Sales Quantity Analysis
Manufacturing Plant                       Quantity:
BOS                                       1,033,818
DAL                                         390,844
LA                                          229,256
ORL                                         386,909
SEA                                          86,680
STL                                         776,743
```

Syntax: **How to Identify an Embedded Field in a Heading or Footing**

```
TYPE=type, [LINE=line_#], OBJECT=FIELD, [ITEM=item #]
```

where:

type

Identifies a type of heading or footing. Select from HEADING, FOOTING, TABHEADING, TABFOOTING, SUBHEAD, or SUBFOOT. For details, see *Identifying a Heading or Footing* on page 84.

line_#

Identifies a line by its position in the heading or footing. You only need to include the LINE attribute if you have a multi-line heading or footing.

item_#

Identifies an item by its position in a line.

If you have more than one embedded field in a heading or footing, you must specify the field you wish to format by giving the item number. Count items from left to right. Do not include text fields in the count. You do not need to specify the item number if there is only one embedded field in the heading or footing.

Note: BORDER options are not supported on specific ITEMS in a HEADING, FOOTING, SUBHEAD, SUBFOOT.

Example: Identifying Embedded Fields in a Heading

The following example illustrates how to format an embedded field in a heading. Notice that the item number is not specified in the StyleSheet declaration since there is only one embedded field in the heading.

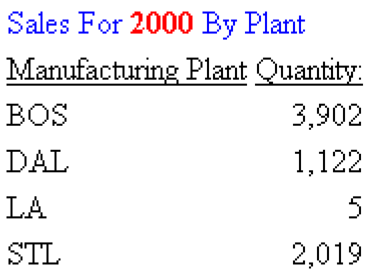
Report Request

```
TABLE FILE CENTORD
HEADING
"Sales For <YEAR By Plant"
SUM QUANTITY  BY PLANT
WHERE YEAR EQ 2000
ON TABLE SET STYLE *
INCLUDE=IBFS:/stylesheet_location/custom_stylesheet.sty,$
END
```

***custom_stylesheet*.sty**

```
TYPE=REPORT, GRID=OFF, $
TYPE=HEADING, OBJECT=TEXT, COLOR=BLUE, $
TYPE=HEADING, OBJECT=FIELD, COLOR=RED, STYLE=BOLD, $
```

The output is:



```
Sales For 2000 By Plant
Manufacturing Plant Quantity
BOS                    3,902
DAL                    1,122
LA                         5
STL                    2,019
```

Syntax: How to Identify a Specific Heading or Footing Based on a WHEN Clause

```
TYPE={HEADING|FOOTING}, CONDITION = n, ... , $
```

where:

n

Is the number of the WHEN condition in the heading or footing from top to bottom. If not specified, formatting applies to all headings and footings.

Example: Styling Multiple Headings With WHEN

The following example illustrates how to display a page for each employee with salary and job code information for that employee. The first WHEN condition applies if the employee is female. The second WHEN condition applies if the employee is male. The third WHEN condition applies if the department is MIS. The fourth WHEN condition applies if the department is PRODUCTION. The StyleSheet declarations include styling elements for the second and third conditions.

Report Request

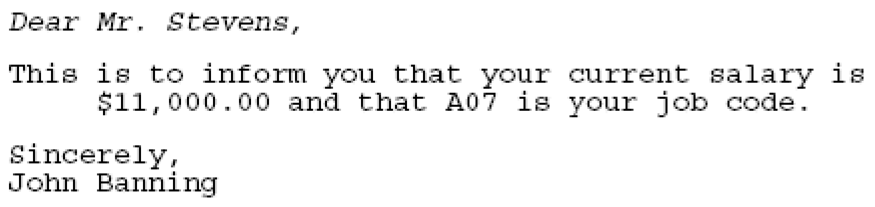
```
DEFINE FILE EMPLOYEE
GENDER/A1 = DECODE FIRST_NAME(ALFRED 'M' RICHARD 'M' JOHN 'M'
  ANTHONY 'M' ROGER 'M' MARY 'F' DIANE 'F' JOAN 'F' ROSEMARIE 'F'
  BARBARA 'F');
MIXEDNAME/A15 = LCWORD(15, LAST_NAME, MIXEDNAME);
NAME/A16 = MIXEDNAME||',';
END

TABLE FILE EMPLOYEE
PRINT LAST_NAME NOPRINT GENDER NOPRINT NAME NOPRINT
HEADING
"Dear Ms. <NAME"
   WHEN GENDER EQ 'F';
HEADING
"Dear Mr. <NAME>"
   WHEN GENDER EQ 'M';
HEADING
" "
HEADING
"This is to inform you that your current salary is "
"<CURR_SAL and <CURR_JOBCODE>is your job code."
" "
"Sincerely,"
HEADING
"Barbara Cross "
  WHEN DEPARTMENT EQ 'MIS';
HEADING
"John Banning   "
  WHEN DEPARTMENT EQ 'PRODUCTION' ;
WHERE LAST_NAME NE 'BANNING' OR 'CROSS'
BY EMP_ID NOPRINT PAGE-BREAK
ON TABLE PCHOLD FORMAT PDF
ON TABLE SET STYLE *
INCLUDE=IBFS:/stylesheet_location/custom_stylesheet.sty,$
END
```

***custom_stylesheet*.sty**

```
TYPE=HEADING, CONDITION=2, STYLE=ITALIC, $
TYPE=HEADING, CONDITION=3, STYLE=BOLD, $
```

In the StyleSheet for the request, heading lines displayed because of the first condition are in italics and heading lines displayed because of the third condition are in boldface.

The first page of output is for a male employee, so the greeting line is in italics:

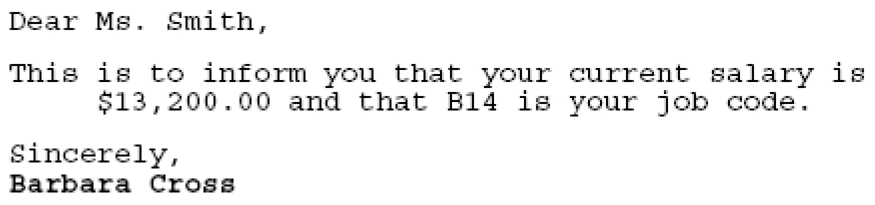


Dear Mr. Stevens,

```
This is to inform you that your current salary is
     $11,000.00 and that A07 is your job code.
```

Sincerely,
John Banning

The second page of output is for an employee in the MIS department, so the signature line is in boldface:



Dear Ms. Smith,

```
This is to inform you that your current salary is
     $13,200.00 and that B14 is your job code.
```

Sincerely,
Barbara Cross

Identifying a Page Number, Underline, or Skipped Line

In a report, you can identify and format page numbers, underlines, and skipped lines using the PAGENUM, SKIPLINE, and UNDERLINE attributes.

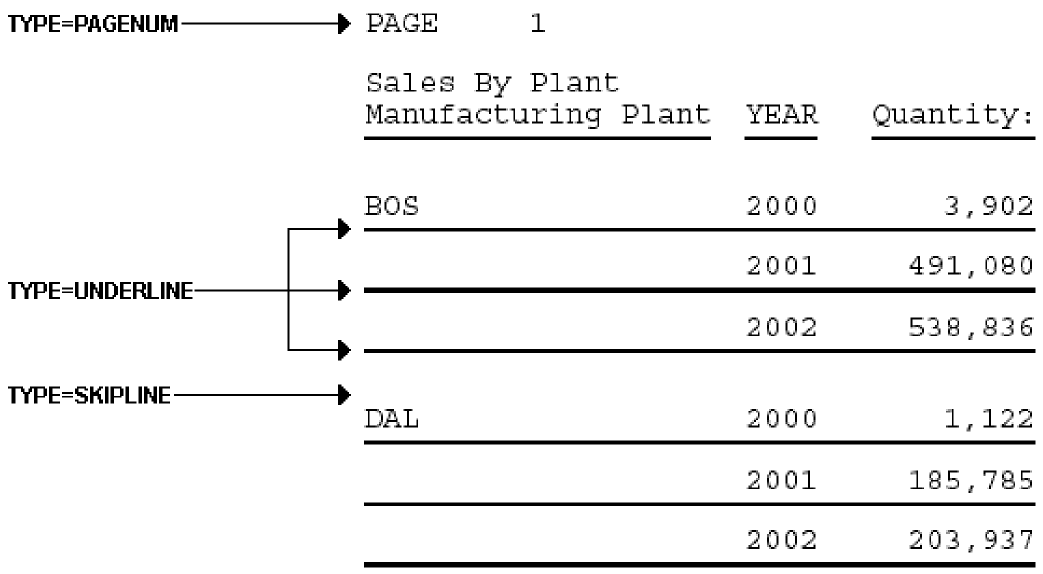
Note that although you can insert skipped lines and underlines in an HTML report, formatting is not supported.

The following examples illustrate where the PAGENUM, UNDERLINE, and SKIPLINE components appear in a report, and which TYPE values you use to identify them.

Report Request

```
TABLE FILE CENTORD
HEADING
"Sales By Plant"
SUM QUANTITY
BY PLANT BY YEAR
WHERE PLANT EQ 'BOS' OR 'DAL'
ON YEAR UNDER-LINE
ON PLANT SKIP-LINE
ON TABLE PCHOLD FORMAT PDF
END
```

The output is:



```
PAGE     1

Sales By Plant
Manufacturing Plant  YEAR    Quantity:


BOS                  2000        3,902

                     2001      491,080

                     2002      538,836


DAL                  2000        1,122

                     2001      185,785

                     2002      203,937
```

Note: Since this request simply illustrates where the components appear in a report, it omits a StyleSheet.

Syntax: How to Identify a Page Number, Underline, or Skipped Line

```
TYPE=type
```

where:

type

Identifies the report component. Select from:

`PAGENUM`, which identifies page numbers.

`SKIPLINE`, which denotes skipped lines generated by ON *field* SKIP-LINE. This is not supported for reports in HTML format.

`UNDERLINE`, which identifies underlines generated by ON *field* UNDER-LINE. This is not supported for reports in HTML format.

Example: Identifying Underlines and Page Numbers

The following example illustrates how to identify underlines and page numbers in a report request.

Note that this report is formatted in PDF since formatting is not supported for underlines in an HTML report.

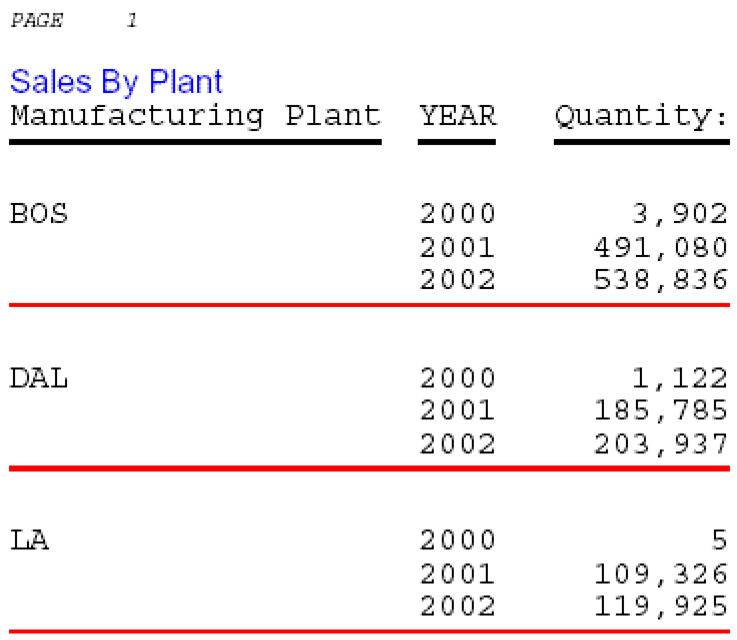
Report Request

```
TABLE FILE CENTORD
HEADING
"Sales By Plant"
SUM QUANTITY
BY PLANT BY YEAR
WHERE PLANT EQ 'BOS' OR 'DAL' OR 'LA'
ON PLANT UNDER-LINE SKIP-LINE
ON TABLE PCHOLD FORMAT PDF
ON TABLE SET STYLE *
INCLUDE=IBFS:/stylesheet_location/custom_stylesheet.sty,$
END
```

***custom_stylesheet*.sty**

```
TYPE=REPORT, GRID=OFF, $
TYPE=HEADING, OBJECT=TEXT, COLOR=BLUE, FONT=ARIAL, $
TYPE=PAGENUM, STYLE=ITALIC, SIZE=8, $
TYPE=UNDERLINE, COLOR=RED, $
```

The output is:



```
PAGE     1

Sales By Plant
Manufacturing Plant  YEAR    Quantity:


BOS                  2000        3,902
                     2001      491,080
                     2002      538,836


DAL                  2000        1,122
                     2001      185,785
                     2002      203,937


LA                   2000            5
                     2001      109,326
                     2002      119,925
```

Example: Identifying Skipped Lines

The following example illustrates how to identify skipped lines in a report.

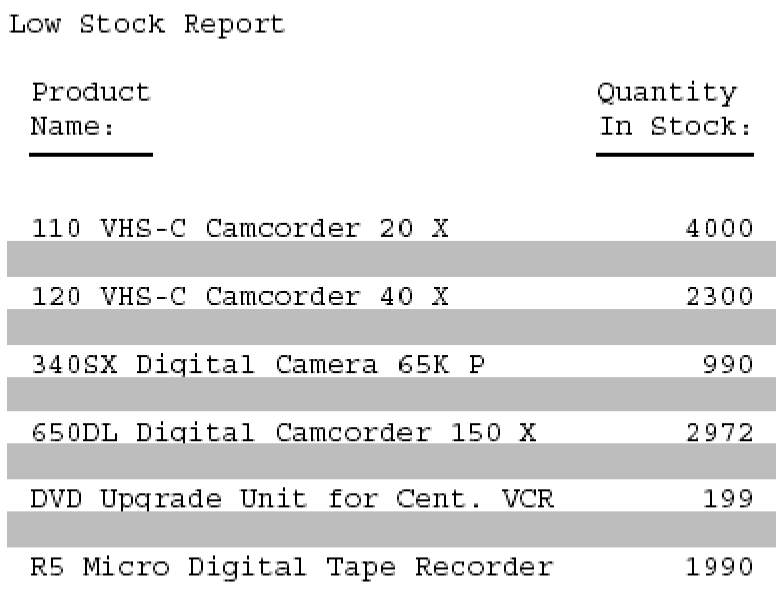
Report Request

```
TABLE FILE CENTINV
HEADING
"Low Stock Report"
" "
SUM QTY_IN_STOCK
WHERE QTY_IN_STOCK LT 5000
BY PRODNAME
ON PRODNAME SKIP-LINE
ON TABLE PCHOLD FORMAT PDF
ON TABLE SET STYLE *
INCLUDE=IBFS:/stylesheet_location/custom_stylesheet.sty,$
END
```

***custom_stylesheet*.sty**

```
TYPE=SKIPLINE, BACKCOLOR=SILVER, $
```

The output is:



```
Low Stock Report

 Product                                Quantity
 Name:                                   In Stock:

 110 VHS-C Camcorder 20 X                     4000

 120 VHS-C Camcorder 40 X                     2300

 340SX Digital Camera 65K P                    990

 650DL Digital Camcorder 150 X                2972

 DVD Upgrade Unit for Cent. VCR                199

 R5 Micro Digital Tape Recorder               1990
```

Chapter 6

Using an External Cascading Style Sheet

Cascading style sheets (CSS) provide a standard way of formatting HTML documents. To format Web Query HTML report output using an external CSS, simply link the CSS to the report (using the Web Query StyleSheet CSSURL attribute) and, optionally, apply the CSS classes to specific report components (using the CLASS attribute).

In this chapter:

What Is a Cascading Style Sheet?

Cascading style sheets (CSS) are an extension to HTML that allow you to specify formatting for an HTML document. You can use two kinds of cascading style sheets with Web Query:

- **An internal cascading style sheet,** which is stored internally in the HTML document that it formats. For information about generating and using an internal CSS for a Web Query report, see *Generating an Internal Cascading Style Sheet for HTML Reports* on page 41.
- **An external cascading style sheet,** which is stored in a separate file that can be shared by multiple documents. The external CSS file can reside on any web server accessible to the browser. You specify its location using the CSSURL Web Query StyleSheet attribute or (in special cases) the LINK element.

You can define classes in a cascading style sheet, and format a report component by assigning one of these CSS classes to it. Classes are described in *What Are Cascading Style Sheet Rules and Classes?* on page 102.

Cascading style sheets are called *cascading* because several different style sheets can be active for a single document at the same time. For example, one style sheet may be associated with the document itself, another style sheet may be linked to the first one, and yet another may be associated with the web browser on which the document is being displayed. When multiple style sheets are in effect, they are applied to the document in a pre-determined sequence set by the browser. Their formatting instructions can be thought of as cascading from one style sheet to the next.

The benefits of using an external cascading style sheet to format a report are described in *Why Use an External Cascading Style Sheet?* on page 103.

You will find external cascading style sheets relevant if you:

- ❑ **Develop reports,** since you now have an improved way of formatting those reports.
- ❑ **Are responsible for presentation guidelines for web documents,** since you will now be able to apply your existing cascading style sheets to reports.

For information about *internal* cascading style sheets, see *Generating an Internal Cascading Style Sheet for HTML Reports* on page 41.

Need more information about CSS? This documentation assumes that you have a working knowledge of cascading style sheets. Teaching about CSS is beyond the scope of this documentation, but many sources of information are available to you. A useful place to begin online is the World Wide Web Consortium cascading style sheets home page (*http://www.w3.org/Style/CSS/*).

What Are Cascading Style Sheet Rules and Classes?

A cascading style sheet (CSS) defines formatting in statements called *rules*. For example, this is a simple rule that makes the background color of the body of an HTML page yellow:

```
BODY {background: yellow}
```

Each rule has a *selector* (BODY in this example) and a *declaration* (background: yellow). A declaration has a property (background) and a value assigned to the property (yellow).

A declaration defines formatting, and a selector determines to what the formatting will be applied. A selector can be any HTML element. A selector can also be a *class*. You can define a class simply by creating a rule for it. By creating rules for classes of an element, you can define different formatting for the same element.

For example, you may wish to display text in a different color depending on whether it is in a sort column, aggregate column, or detail column in a report. To accomplish this you could create three classes of the BODY element, sortColumn, aggregateColumn, and detailColumn:

```
BODY.sortColumn {color: blue}
BODY.aggregateColumn {color: green}
BODY.detailColumn {color: black}
```

You can also define a generic class, that is, a class that is not limited to a single element. For example:

```
.pageFooting {font-weight: bolder}
```

You can use a generic class to specify formatting for any Web Query report component.

Using external cascading style sheet rules and classes to format a report is described in *Formatting a Report With an External Cascading Style Sheet* on page 104.

Why Use an External Cascading Style Sheet?

If you already use Web Query StyleSheets to format reports, you can realize these additional benefits by combining them with external cascading style sheets:

- ❑ **Increased formatting options.** Almost any formatting that you can specify in a cascading style sheet, you can apply to a report. This enables you to take advantage of formatting options that are unavailable using native Web Query StyleSheet attributes. For example, you can use browser-based measurements so that the person who views the report can control the size of fonts, margins, and other elements whose size has been specified in the CSS in terms of *em*, a unit of relative measurement. You can also use CSS to control line height and letter spacing, and in general can use CSS to exercise more control over positioning items in a report.

- ❑ **Improved performance.** Cascading style sheets enable Web Query to generate more concise HTML output. This reduces the bandwidth used by the network to return a report to a browser, and displays the report faster.

- ❑ **Reduced effort.** Enterprises that already use cascading style sheets can now also apply them to Web Query reports, avoiding duplication of effort to specify and maintain formatting instructions.

- ❑ **Easier standards conformance.** You can ensure that reports conform to your enterprise formatting guidelines, because now formatting instructions for all your web documents can be specified in one set of cascading style sheets (instead of replicating some of them in Web Query StyleSheets).

Formatting a Report With an External Cascading Style Sheet

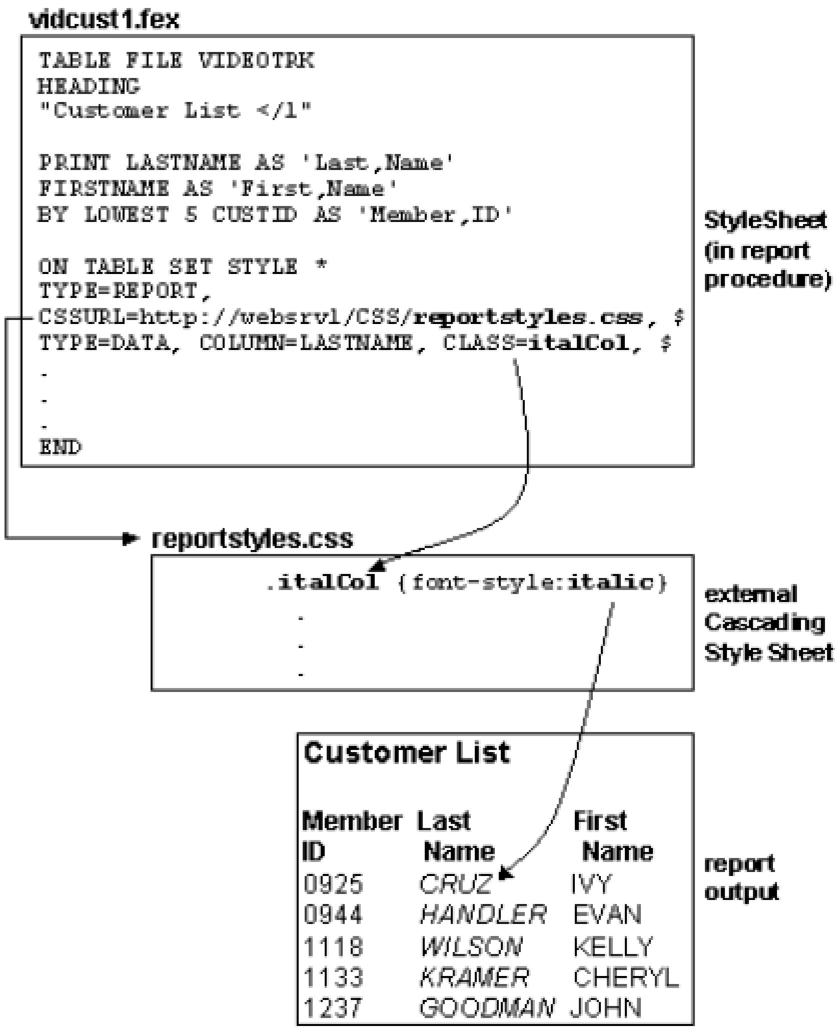
There are three items required to format a report with an external cascading style sheet (CSS):

- ❑ **An external cascading style sheet** that specifies the formatting to be applied to the report. For more information, see *Working With an External Cascading Style Sheet* on page 110.
- ❑ **A Web Query StyleSheet** in which you apply external CSS formatting to the components of your report. However, you do not need a Web Query StyleSheet when you apply formatting to the *entire* report. For more information, see *Applying External Cascading Style Sheet Formatting* on page 113.

  Although you can also use a Web Query StyleSheet to specify additional formatting outside of the external CSS, this is subject to restrictions. For more information, see *Combining an External CSS With Other Formatting Methods* on page 115.
- ❑ **A link to the external cascading style sheet** from the report. For more information, see *Linking to an External Cascading Style Sheet* on page 117.

To find out how to use these three items to format a report, see *How to Format a Report Using an External Cascading Style Sheet* on page 106.

For an example that demonstrates how these items work together, see *Linking to the ReportStyles External Cascading Style Sheet* on page 107 and the following image:


vidcust1.fex
StyleSheet (in report procedure)
reportstyles.css
external Cascading Style Sheet
report output

```
TABLE FILE VIDEOTRK
HEADING
"Customer List </1"

PRINT LASTNAME AS 'Last,Name'
FIRSTNAME AS 'First,Name'
BY LOWEST 5 CUSTID AS 'Member,ID'

ON TABLE SET STYLE *
TYPE=REPORT,
CSSURL=http://websrv1/CSS/reportstyles.css, $
TYPE=DATA, COLUMN=LASTNAME, CLASS=italCol, $
.
.
.
END
```

```
.italCol {font-style:italic}
.
.
.
```

```
Customer List

Member  Last      First
ID      Name      Name
0925    CRUZ      IVY
0944    HANDLER   EVAN
1118    WILSON    KELLY
1133    KRAMER    CHERYL
1237    GOODMAN   JOHN
```

Procedure: How to Format a Report Using an External Cascading Style Sheet

To format a report using an external cascading style sheet (CSS):

1. **Specify the report formatting in the CSS.** Determine which cascading style sheet you will use, which of its rules specify default formatting for your report, and which of its classes are suitable for you to apply to the report components. Alternatively, if you are creating a new CSS, or extending an existing one, define new rules to specify formatting for your report. To specify formatting for:

   ❑ **A report component,** you can use a rule for any generic class (that is, any class not declared for an element). For an example, see *A CSS Rule for the ColumnTitle Class* on page 106.

   ❑ **The entire report,** use a rule for the BODY or TD elements (not a rule for a class of these elements), and skip step 2. This is an effective way of specifying the default formatting of a report, and generates more efficient report output than does applying a CSS class to the entire report. For an example, see *A CSS Rule for the TD Element* on page 107.

2. **Assign classes to report components.** In a Web Query StyleSheet, assign a cascading style sheet class to each report component that you want to format. Specify the class using the CLASS attribute. You can assign each component a different class, and you can assign the same class to multiple components.

   For an example, see *Applying a CSS Class to ACROSS Values in a Report* on page 107. For more information, see *Applying External Cascading Style Sheet Formatting* on page 113.

3. **Link to the CSS.** Link to the external cascading style sheet by assigning the CSS file URL to the CSSURL Web Query StyleSheet attribute. For more information, see *Using the CSSURL Attribute* on page 117.

Reference: A CSS Rule for the ColumnTitle Class

The following cascading style sheet (CSS) rule declares the ColumnTitle generic class (that is, a class not tied to an element):

```
.ColumnTitle {font-family:helvetica; font-weight:bold; color:blue;}
```

Reference: A CSS Rule for the TD Element

The following cascading style sheet (CSS) rule for the TD element specifies the element font family:

```
TD {font-family:helvetica}
```

Because this rule is for the TD element, its formatting is applied to an entire report, not just a component of the report.

Reference: Applying a CSS Class to ACROSS Values in a Report

The following Web Query StyleSheet declaration formats ACROSS values by applying the formatting specified for the ColumnTitle class:

```
TYPE=AcrossValue, CLASS=ColumnTitle, $
```

Reference: Linking to the ReportStyles External Cascading Style Sheet

The following Web Query StyleSheet declaration links to the ReportStyles external cascading style sheet:

```
TYPE=REPORT, CSSURL=http://webserv1/css/reportstyles.css
```

Example: Formatting a Report Using an External CSS

The following request displays the products currently offered by Gotham Grinds, and is formatted using an external cascading style sheet (CSS). The report is formatted so that:

- ❏ Its default font family is Arial.
- ❏ The report heading overrides the default with a font family of Times New Roman. The heading is also in a larger font and center justified.
- ❏ All column titles are in a bolder font and have a light blue background.
- ❏ When a product unit price is less than $27, the report displays the product row in green italics.

The report request, StyleSheet declarations, and the external cascading style sheet, named report01.css, are shown below.

Report Request

```
    TABLE FILE GGPRODS
    HEADING
    "</1 Current Products</1"
    PRINT PRODUCT_DESCRIPTION UNIT_PRICE
    BY PRODUCT_ID

1.  ON TABLE SET STYLE *
    INCLUDE=IBFS:/stylesheet_location/custom_stylesheet.sty,$
    END
```

***custom_stylesheet*.sty**

```
2.  TYPE=REPORT, CSSURL=http://websrv2/css/report01.css, $
3.  TYPE=HEADING, CLASS=headText, $
4.  TYPE=TITLE, CLASS=reportTitles, $
5.  TYPE=DATA, CLASS=lowCost, WHEN=UNIT_PRICE LT 27, $
```

report01.css

```
6.  BODY  {font-family:Arial, Sans-Serif}
7.  TABLE {border:0}
7.  TD    {border:0}
8.  .reportTitles {font-weight:bolder; background:lightblue;}
9.  .lowCost {color:green; font-style:italic;}
10. .headText {font-family:Times New Roman, Serif; font-size:larger;
               text-align:center}
```

1. Begin the Web Query StyleSheet.
2. Link to the external cascading style sheet, report01.css.
3. Format the report heading using the cascading style sheet rule for the headText class.
4. Format the report column titles using the CSS rule for the reportTitles class.
5. For each report row for which the product unit cost is less than $27, format that row using the CSS rule for the lowCost class.
6. This CSS rule for the BODY element specifies the font family Arial and, if Arial is unavailable, the generic font family Sans Serif.

   Because this is a rule for BODY, it is applied to the entire report. All text in the report will default to Arial. You can override this for a particular report component by applying a rule for a generic class to that component, as is done in this procedure with the rule for the headText class (see line 10).
7. These CSS rules for the TABLE and TD elements remove the report default grid.

8. This CSS rule for the generic class reportTitles specifies a bolder relative font weight and a light blue background color.

   The Web Query StyleSheet applies this to the report column titles (see line 4).

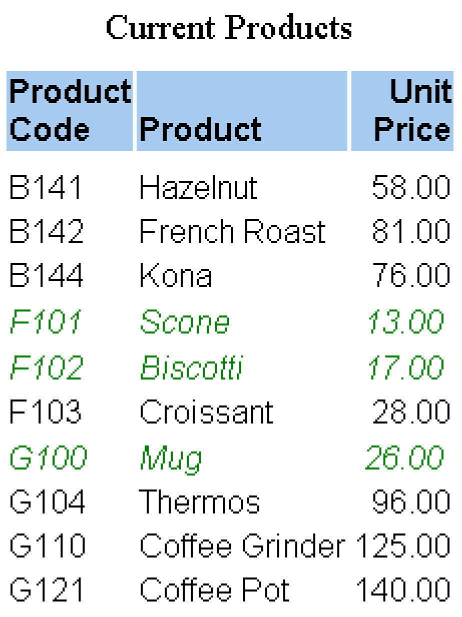
9. This CSS rule for the generic class lowCost specifies the text color green and the font style italic.

   The Web Query StyleSheet applies this rule conditionally to report rows for which the product unit cost is less than $27 (see line 5).

10. The CSS rule for the generic class headText specifies the font family Times New Roman and, if Times New Roman is unavailable, the generic font family Serif. It also specifies a larger relative font size and center justification.

    The Web Query StyleSheet applies this rule to the report heading. It overrides the default font family specified in the rule for the BODY element (see line 6).

The output is:



Current Products

Product Code	Product	Unit Price
B141	Hazelnut	58.00
B142	French Roast	81.00
B144	Kona	76.00
F101	*Scone*	*13.00*
F102	*Biscotti*	*17.00*
F103	Croissant	28.00
G100	*Mug*	*26.00*
G104	Thermos	96.00
G110	Coffee Grinder	125.00
G121	Coffee Pot	140.00

Working With an External Cascading Style Sheet

When you work with an external cascading style sheet (CSS) to specify report formatting, you need to know about:

- ❑ Choosing an existing or new external CSS. For more information, see *Choosing an External Cascading Style Sheet* on page 110.
- ❑ Where an external CSS can reside. For more information, see *External Cascading Style Sheet Location* on page 111.
- ❑ How you can apply multiple cascading style sheets to one report. For more information, see *Using Several External Cascading Style Sheets* on page 111.
- ❑ Editing an external CSS. For more information, see *Editing an External Cascading Style Sheet* on page 111.
- ❑ Using CSS rules and classes to specify report formatting. For more information, see *Choosing a Cascading Style Sheet Rule* on page 111.
- ❑ Suggestions for naming cascading style sheet classes. For more information, see *Naming a Cascading Style Sheet Class* on page 112.
- ❑ Combining other formatting methods, such as Web Query StyleSheets, with an external CSS. For more information, see *Combining an External CSS With Other Formatting Methods* on page 115.

Choosing an External Cascading Style Sheet

To format a report using an external cascading style sheet (CSS), you can choose to:

- ❑ **Apply an existing CSS with no changes.** The external cascading style sheet can be one that you use for other documents, and can contain all kinds of rules, not only rules that format reports. For example, the CSS could include rules to format other elements in the webpages used by your Web Query applications, as well as rules for other kinds of webpages. This enables you to use one cascading style sheet to format all of your web documents.
- ❑ **Edit an existing CSS to add or modify rules.** For example, you might edit a cascading style sheet to add new generic classes to format report components.
- ❑ **Create a new CSS.** You can create a new cascading style sheet to format your reports. See the recommendations in *Naming a Cascading Style Sheet Class* on page 112 about naming classes.

External Cascading Style Sheet Location

An external cascading style sheet (CSS) can reside on any web server platform. However, if the CSSURL StyleSheet attribute specifies a relative URL, the cascading style sheet must reside on the web server used by Web Query.

Using Several External Cascading Style Sheets

Although each report procedure can link to only one external cascading style sheet (CSS), you can use several cascading style sheets to format a report by linking to one CSS that then imports several others. For information about importing one CSS into another, see your third-party CSS documentation.

Editing an External Cascading Style Sheet

You can edit an external cascading style sheet (CSS) using a text editor or a third-party web development tool.

If the formatting of a report is specified entirely using a cascading style sheet, and you edit that CSS, the next time someone displays the report it will reflect the changes to the CSS without the report having to be rerun.

However, if the report does not reflect the changes, it may be because the web browser is continuing to use the old version of the CSS that it had stored in cache. The user displaying the report may need to reload the CSS file from the web server by clicking the browser Refresh button to ensure that the browser uses the most current version of the CSS to format the report.

Choosing a Cascading Style Sheet Rule

You can format different parts of a report using different types of rules.

To format:	Use a rule for:
The entire report	BODY or TD
Any report component	A generic class (that is, one declared without an element)

To choose between using a rule for BODY or for TD, note that a rule for:

- ❑ **BODY** will specify default formatting for the entire webpage in which the report appears, including the report itself. (Note that this relies upon CSS inheritance which, like all CSS behavior, is implemented by the web browser of each user and is browser-dependent.)
- ❑ **TD** will specify default formatting only for the report, and for any other table cells that you may have on the page. TD is the table data (that is, table cell) element. Web Query generates most HTML report output as an HTML table, placing each report item in a separate cell. This enables a rule for TD to format the entire report.

When you use a rule for a class to format a report component, you must assign the class to the component in a Web Query StyleSheet using the CLASS attribute, as described in *How to Use the CLASS Attribute to Apply CSS Formatting* on page 114.

If you wish to apply several CSS properties to a single report component, it is recommended that you declare them in a single class. This generates more efficient output than does declaring one property per class.

The owner of each cascading style sheet should consider making available a list of all the classes in that CSS that can be used to format reports, so that everyone who develops reports knows from which classes they may choose.

For an example of a rule for:

- ❑ A generic class to format a report component, see *A CSS Rule for the ColumnTitle Class* on page 106.
- ❑ The TD element to format an entire report, see *A CSS Rule for the TD Element* on page 107.

Naming a Cascading Style Sheet Class

When you provide a name for a new class, note that class names are case-sensitive (although some web browsers may not enforce case sensitivity).

When you create a new class, we recommend naming it after the function, not the appearance, of the report component to which you will be applying it. This ensures that the name remains meaningful even if you later change the appearance of the report component. For example, if you want all report titles to be red, the class you declare to format titles might be named Title, but not Red.

Applying External Cascading Style Sheet Formatting

You can apply external cascading style sheet (CSS) formatting to:

❏ **A report component** (for example, to make a column italic). Assign a cascading style sheet class to the report component using the Web Query StyleSheet CLASS attribute. For information about the CLASS attribute, see *How to Use the CLASS Attribute to Apply CSS Formatting* on page 114. For information about specifying different types of report components, see *Identifying a Report Component in a Web Query StyleSheet* on page 57.

When formatting a tabular or free-form report, you can format any report component by assigning a CSS class.

❏ **An entire report** (for example, to make the entire report italic). You specify the formatting in the external CSS in a rule for the BODY or TD elements. You must also link the report to the CSS. You do not need a rule for a class of an element, and you do not need a Web Query StyleSheet declaration. For an example, see *A CSS Rule for the TD Element* on page 107.

It is recommended that when you use an external cascading style sheet to format a report, you do not also use a Web Query StyleSheet to specify the report formatting, unless you also generate an internal cascading style sheet. For more information, see *Combining an External CSS With Other Formatting Methods* on page 115.

Syntax: How to Use the CLASS Attribute to Apply CSS Formatting

To apply an external cascading style sheet (CSS) class to a report component, use the following syntax in a Web Query StyleSheet declaration

```
TYPE = type, [subtype,] CLASS = classname, [when,] [link,] $
```

where:

type

Identifies the report component to which you are applying the class formatting. For tabular and free-form reports, it can be any component, as described in *Identifying a Report Component in a Web Query StyleSheet* on page 57.

Each report component can be formatted by one class. If you specify several classes for a report component:

1. The classes that are in declarations with conditional formatting are evaluated first. For each cell in the report component, the first class whose condition is satisfied by the cell row is assigned to the cell.
2. If none of the conditions is satisfied, or if there are no conditional declarations, the class in the first unconditional declaration is assigned to the report component. All following declarations for that component are ignored.

subtype

Is an optional attribute and value needed to completely specify some kinds of report components. For example, COLUMN is needed to specify a particular report column.

classname

Is the name of the cascading style sheet class whose formatting you are applying to the report component. You can assign the same class to multiple report components.

Class names can be up to 23 characters and are case-sensitive. You must use the same case found in the class rule in the external cascading style sheet. Note, however, that some web browsers may not enforce case sensitivity.

when

Is an optional WHEN attribute and value. Supply this if you want to apply the formatting conditionally.

link

Is an optional FOCEXEC, URL, or JAVASCRIPT attribute and value. Supply this if you want to link the report component to another resource, such as a report to which the user can drill down.

Combining an External CSS With Other Formatting Methods

When you use an external cascading style sheet (CSS) to format a report, you can use other formatting methods at the same time. Some of these other methods are subject to restrictions. The other methods that you can use with an external CSS are:

- **Web Query StyleSheets.** An effective way of combining an external CSS with a Web Query StyleSheet is to link to an external CSS to provide default formatting, and use a Web Query StyleSheet to override those defaults for individual report components.

  Do not attempt to format the same property of the same report component using both an external CSS class (through the CLASS attribute) and a Web Query StyleSheet attribute, since the two formatting instructions could conflict with each other.

  For complete instructions about using an external CSS with a Web Query StyleSheet, see *Combining an External CSS With a Web Query StyleSheet* on page 116.

- **Reporting language instructions.** You can use reporting language formatting instructions, such as HEADING CENTER, PAGE-BREAK, and spot markers (for example, </3). However, you should not apply both a reporting language instruction, and an external cascading style sheet rule, to perform the same formatting on the same report component, because they might conflict with each other.

  For example, you should not specify *both* of the following for the same report:

  - HEADING CENTER in the report request.

  - Text-align in an external CSS, applied to the report page heading.

  Both of these will attempt to align the report page heading.

Combining an External CSS With a Web Query StyleSheet

When you use an external cascading style sheet (CSS) to format a report, you can use a Web Query StyleSheet at the same time.

An effective way of doing this is to link to an external CSS to provide default formatting, and use a Web Query StyleSheet to override those defaults for individual report components. The cascading style sheet BODY or TD rule will provide the default formatting for the report. If you wish, you can override the defaults for individual report components using native Web Query StyleSheet attributes. This enables you to conform to the formatting standards of your organization as they are implemented in a CSS, while allowing you to customize those standards for Web Query reports using Web Query StyleSheet attributes. For information about using a BODY or TD rule for default formatting, see *Choosing a Cascading Style Sheet Rule* on page 111. For an example, see *Inheritance and External Cascading Style Sheets* on page 119.

You cannot double-format. You should not attempt to format the same property of the same report component using both an external CSS class (the CLASS attribute) and a Web Query StyleSheet attribute, since the class and the StyleSheet attribute could conflict with each other.

For example, you should not include the following declarations in the same StyleSheet because they would both try to assign a color to the Country column:

```
TYPE=DATA, COLUMN=Country, COLOR=Orange, $
TYPE=DATA, CLASS=TextColor, $
```

You can specify classes and Web Query StyleSheet attributes that format different properties of the same report component, and that format different report components. For example, the following declarations are acceptable in the same StyleSheet:

```
1. TYPE=HEADING, COLOR=Green, $
1. TYPE=HEADING, CLASS=HeadingFontSize, $
2. TYPE=DATA, COLUMN=COUNTRY, BACKCOLOR=Yellow, $
2. TYPE=DATA, COLUMN=CAR, CLASS=DataBackgroundColor, $
3. TYPE=DATA, COLUMN=MODEL, FOCEXEC=NewSales(CarGroup=Car), $
```

1. These two declarations are compatible because they format different properties (color and font size).
2. These two declarations are compatible because they format different report components (the Country column and the Car column).
3. This declaration will be compatible with all CSS classes, since it does not format a report component, but instead defines a hyperlink.

Linking to an External Cascading Style Sheet

To format a report using an external cascading style sheet (CSS), you must link the cascading style sheet to the report.

Using the CSSURL Attribute

You can link an external cascading style sheet (CSS) to a report using the CSSURL Web Query StyleSheet attribute.

Using CSSURL as a StyleSheet attribute enables you to specify:

- **A longer URL**, since the maximum URL length is 255 characters in the attribute, compared with 69 characters in the parameter.
- **All formatting information in one place**, since you can specify the link to the external CSS *and* the references to CSS classes within the Web Query StyleSheet. This makes it easier for you to maintain your formatting logic.

For more information about the CSS attribute, see *How to Use the CSSURL Attribute to Link to an External CSS* on page 117.

Syntax: How to Use the CSSURL Attribute to Link to an External CSS

To link an external cascading style sheet (CSS) to a report using a Web Query StyleSheet attribute, use the following syntax

```
[TYPE=REPORT,] CSSURL=css_url, $
```

where:

`TYPE=REPORT`

Specifies that this attribute is being applied to the entire report. If it is omitted, the StyleSheet declaration defaults to it.

`css_url`

Is the URL of the external cascading style sheet. If the external CSS resides on a web server platform that is case-sensitive, you must specify it using the correct case.

The URL can be up to 255 characters. If your external cascading style sheet URL exceeds this limit, you can shorten the URL by defining an alias (also known as a virtual directory) on the web server to represent part of the path.

You can specify an absolute or relative URL. If it is relative, the external CSS must reside on the web server used by Web Query.

Example: Linking to an External Cascading Style Sheet Using the CSSURL Attribute

The following example illustrates how to display the products currently offered by Gotham Grinds. It is formatted using an external cascading style sheet (CSS), and links to the CSS using the CSSURL attribute in the Web Query StyleSheet:

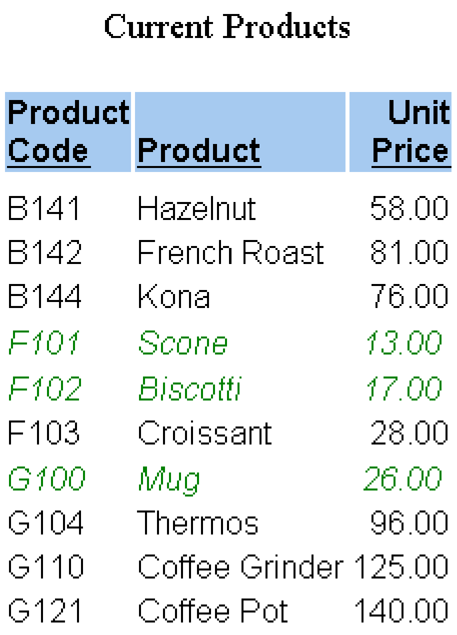
Report Request

```
TABLE FILE GGPRODS
HEADING
"</1 Current Products</1"
PRINT PRODUCT_DESCRIPTION UNIT_PRICE
BY PRODUCT_ID
ON TABLE SET STYLE *
INCLUDE=IBFS:/stylesheet_location/custom_stylesheet.sty,$
END
```

***custom_stylesheet*.sty**

```
TYPE=REPORT, CSSURL=http://websrv2/css/report01.css, $
TYPE=HEADING, CLASS=headText, $
TYPE=TITLE, CLASS=reportTitles, $
TYPE=DATA, CLASS=lowCost, WHEN=UNIT_PRICE LT 27, $
```

The output is:



Current Products

Product Code	Product	Unit Price
B141	Hazelnut	58.00
B142	French Roast	81.00
B144	Kona	76.00
F101	*Scone*	*13.00*
F102	*Biscotti*	*17.00*
F103	Croissant	28.00
G100	*Mug*	*26.00*
G104	Thermos	96.00
G110	Coffee Grinder	125.00
G121	Coffee Pot	140.00

Inheritance and External Cascading Style Sheets

In a report that is formatted using an external cascading style sheet (CSS), a report component inherits formatting from the TD element and from all elements that TD nests within, such as BODY. Note that inheritance, like all CSS behavior, is implemented by the web browser of each user and is browser-dependent.

This differs from a report that is formatted using a Web Query StyleSheet, in which a report component inherits formatting from a higher-level component. When you format a report using external cascading style sheet classes, a class assigned to a report component *does not* inherit formatting from a class that has been assigned to a higher-level component.

Example: A Report Column Inheriting Formatting From the TD Element

This report displays a list of the vendors that supply products to Gotham Grinds. Its formatting instructions specify that:

- ❑ The entire report has an orange default background color. This is specified in a rule for the TD element.
- ❑ The report data is displayed in an italic Arial font. The report data inherits the orange background color from the rule for TD.
- ❑ The report PRODUCT_ID data has a yellow background color, overriding the default specified in the rule for TD.

  If the formatting of the report had been specified in a Web Query StyleSheet, instead of in an external CSS, PRODUCT_ID would inherit the italic Arial font from its parent report component (that is, from the report data). Instead, because its formatting is specified in an external CSS, PRODUCT_ID inherits formatting from the rule for the TD element, not from a higher-level report component, and so it does not inherit the italic Arial font.

The report request, StyleSheet declarations, and external cascading style sheet, named report02.css, are shown below.

Report Request

```
TABLE FILE GGPRODS
PRINT PRODUCT_DESCRIPTION VENDOR_NAME
BY PRODUCT_ID
ON TABLE SET STYLE *
INCLUDE=IBFS:/stylesheet_location/custom_stylesheet.sty,$
END
```

***custom_stylesheet*.sty**

```
1. TYPE=REPORT, CSSURL=http://websrv2/css/report02.css, $
2. TYPE=DATA, CLASS=Data, $
3. TYPE=DATA, COLUMN=PRODUCT_ID, CLASS=Sort, $
```

report02.css

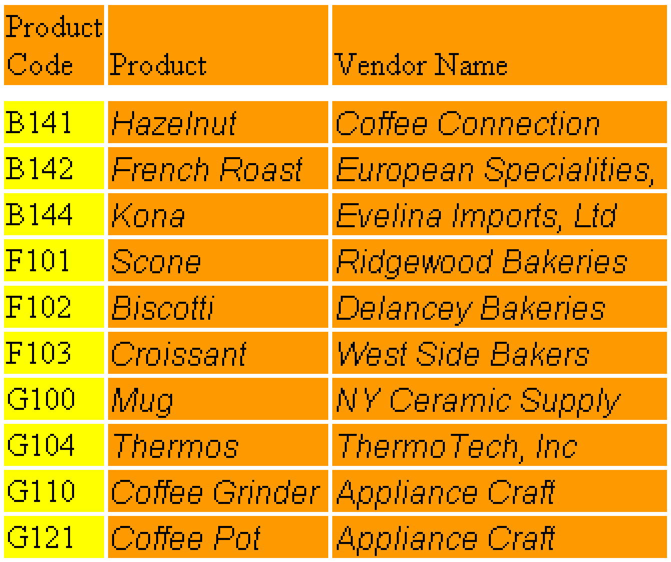
```
4. TD    {background:orange; border:0}
5. TABLE {border:0}
6. .Data {font-style:italic; font-family:Arial}
7. .Sort {background:yellow}
```

1. Set CSSURL to link to the external cascading style sheet, report02.css.
2. Format the report data using the CSS rule for the Data class.
3. Format the report PRODUCT_ID data using the CSS rule for the Sort class. This overrides the declaration for report data in general in line 2.
4. This CSS rule for the TD element specifies an orange background. Because it is a rule for TD, it is applied to the entire report. You can override this for a particular report component by applying a rule for a generic class to that component, as is done in this procedure with the rule for the Sort class (see line 7).
5. These CSS rules for the TD and TABLE elements remove the default grid for the report.
6. This CSS rule for the generic class Data specifies an Arial font family and an italic font style. The Web Query StyleSheet applies this to the report data (see line 2).

   This rule inherits background color from the rule for the TD element (line 4).
7. This CSS rule for the generic class Sort specifies a yellow background. The Web Query StyleSheet applies this rule to data for PRODUCT_ID (see line 3).

   This rule overrides the default background color specified in line 4.

The output is:



Product Code	Product	Vendor Name
B141	Hazelnut	Coffee Connection
B142	French Roast	European Specialities,
B144	Kona	Evelina Imports, Ltd
F101	Scone	Ridgewood Bakeries
F102	Biscotti	Delancey Bakeries
F103	Croissant	West Side Bakers
G100	Mug	NY Ceramic Supply
G104	Thermos	ThermoTech, Inc
G110	Coffee Grinder	Appliance Craft
G121	Coffee Pot	Appliance Craft

Requirements for Using an External Cascading Style Sheet

When you use an external cascading style sheet (CSS) to format a report, be aware of the following requirements:

- **Generate HTML report output.** You can use an external cascading style sheet to format any report that you generate as HTML, whether you save the report output in a file or send it directly to a web browser. You cannot use an external CSS for a report generated in a different format, such as PDF or Excel.

- If you are not generating an internal cascading style sheet, do not specify external CSS classes (CLASS=) and native Web Query StyleSheet attributes in the same Web Query StyleSheet (other than the exceptions noted in the next paragraph). Doing so could create formatting conflicts.

  Exceptions. Even when specifying external CSS classes, you should use native Web Query StyleSheet attributes to:

  - Create hyperlinks (using the FOCEXEC, JAVASCRIPT, and URL attributes). However, if you wish to format a hyperlink, you should do so using the cascading style sheet.

  - Make a Web Query StyleSheet declaration conditional (using the WHEN attribute).

  - Embed an image (using the IMAGE attribute). However, if you wish to format the image (for example, to position it), you should do so using the cascading style sheet.

For more information, see *Combining an External CSS With Other Formatting Methods* on page 115.

- ❑ **Do not specify the same formatting using the reporting language and CSS.** You can use reporting language formatting instructions, such as HEADING CENTER, PAGE-BREAK, and spot markers (for example, </3). However, you should not apply both a TABLE language instruction, and an external cascading style sheet rule, to perform the same formatting on the same report component. For more information, see *Combining an External CSS With Other Formatting Methods* on page 115.

- ❑ **Use a cascading style sheet-enabled web browser.** Each user who wishes to display a report formatted using a cascading style sheet must have a web browser that supports CSS.

  Note that how a cascading style sheet rule formats your report is determined entirely by the support of your web browser and implementation of cascading style sheets, not by and CSS. Some web browsers may not fully support the latest CSS version, or may implement a CSS feature in different ways.

- ❑ **Do not override the cascading style sheet specified for the report.** If a browser has been customized to ignore cascading style sheets or to employ the personal cascading style sheet of the user, and the user wishes to view reports as they were intended to be seen (with the specified cascading style sheet), the user must reset the browser to accept the cascading style sheet of each document.

  For instructions about checking or changing a browser setting, see the browser Help. For information about how conflicts between CSS rules are resolved (for example, between a rule specified in a CSS document and a rule specified in the reader web browser CSS), see your third-party CSS documentation.

Chapter 7

Laying Out the Report Page

You can customize page layout using StyleSheet attributes. The page layout attributes available to you, like all other attributes, depend on the display format. For instance, some attributes support print-oriented display formats, such as PDF, and they do not apply to HTML reports displayed in a browser.

A report page has default layout characteristics. However, you can change any of them to customize the layout. You can control the appearance of a report, including column arrangement on a page, page numbering, page breaks, use of grids and images, and much more.

Note: InfoAssist has built-in styling options that generate some of the StyleSheet syntax described in this chapter.

In this chapter:

Selecting Page Size, Orientation, and Color

You can select the page size, page orientation (portrait or landscape), and page color for your report. The default page size is letter (8.5 x 11 inches), but you can select from many other sizes, including legal and envelopes.

Reference: Page Size, Orientation, and Color Attributes

Attribute	Description	Applies to
`PAGESIZE`	Sets the page size.	In the Development environment: ❏ PDF ❏ PPTX ❏ PPT
`ORIENTATION`	Sets the page orientation.	In the Development environment: ❏ PDF ❏ XLSX ❏ EXL2K ❏ PPTX ❏ PPT
`PAGECOLOR`	Sets the page color.	HTML report with an internal cascading style sheet

Syntax: How to Set Page Size

This syntax applies to a PDF, PPT, or PPTX report .

```
[TYPE=REPORT,] PAGESIZE={size|LETTER}, $
```

where:

`TYPE=REPORT`

Applies the page size to the entire report. Not required, as it is the default value.

`size`

Is the page size. If printing a report, the value should match the size of the paper. Otherwise, the report may be cropped or printed with extra blank space.

Valid values are:

Value	Description
`LETTER`	8.5 x 11 inches. LETTER is the default value.
`LEGAL`	8.5 x 14 inches.
`TABLOID`	11 x 17 inches.
`LEDGER`	17 x 11 inches.
`C`	17 x 22 inches.
`D`	22 x 34 inches.
`E`	34 x 44 inches.
`STATEMENT`	5.5 x 8.5 inches.
`EXECUTIVE`	7.5 x 10.5 inches.
`FOLIO`	8.5 x 13 inches.
`10x14`	10 x 14 inches.
`A3`	297 x 420 millimeters.
`A4`	210 x 297 millimeters.
`A5`	148 x 210 millimeters.
`B4`	250 x 354 millimeters.
`B5`	182 x 257 millimeters.
`QUARTO`	215 x 275 millimeters.
`ENVELOPE-9`	3.875 x 8.875 inches.

Value	Description
ENVELOPE-10	4.125 x 9.5 inches.
ENVELOPE-11	4.5 x 10.375 inches.
ENVELOPE-12	4.5 x 11 inches.
ENVELOPE-14	5 x 11.5 inches.
ENVELOPE-MONARCH	3.875 x 7.5 inches.
ENVELOPE-PERSONAL	3.625 x 6.5 inches.
ENVELOPE-DL	110 x 220 millimeters.
ENVELOPE-C3	324 x 458 millimeters.
ENVELOPE-C4	229 x 324 millimeters.
ENVELOPE-C5	162 x 229 millimeters.
ENVELOPE-C6	114 x 162 millimeters.
ENVELOPE-C65	114 x 229 millimeters.
ENVELOPE-B4	250 x 353 millimeters.
ENVELOPE-B5	176 x 250 millimeters.
ENVELOPE-B6	176 x 125 millimeters.
ENVELOPE-ITALY	110 x 230 millimeters.
US-STANDARD-FANFOLD	14.875 x 11 inches.
GERMAN-STANDARD-FANFOLD	8.5 x 12 inches.
GERMAN-LEGAL-FANFOLD	8.5 x 13 inches.

Syntax: How to Set Page Orientation

The following syntax applies to a PDF, XLSX, or EXL2K report.

```
[TYPE=REPORT,] ORIENTATION={PORTRAIT|LANDSCAPE}, $
```

where:

`TYPE=REPORT`

Applies the page orientation to the entire report. Not required, as it is the default.

`PORTRAIT`

Displays the report across the narrower dimension of a vertical page, producing a page that is longer than it is wide. PORTRAIT is the default value.

`LANDSCAPE`

Displays the report across the wider dimension of a horizontal page, producing a page that is wider than it is long.

Example: Setting Page Orientation

The following example illustrates how to set the page orientation of a PDF report to landscape.

Report Request

```
TABLE FILE CENTQA
SUM CNT.PROBNUM AS 'Total Number, of Problems'
SUM CNT.PROBNUM AS 'Problems From, Each Plant' BY PLANT
SUM CNT.PROBNUM AS 'Problem by Product' BY PLANT BY PRODNAME
ON PLANT PAGE-BREAK
HEADING CENTER
"QA Report for Company, Plant, and Product"
" "
ON TABLE COLUMN-TOTAL
ON TABLE PCHOLD FORMAT PDF
ON TABLE SET STYLE *
INCLUDE=IBFS:/stylesheet_location/custom_stylesheet.sty,$
END
```

***custom_stylesheet*.sty**

```
TYPE=REPORT, ORIENTATION=LANDSCAPE, $
```

Syntax: How to Set Page Color

The following syntax applies to an HTML report with an internal cascading style sheet.

```
[TYPE=REPORT,] ... PAGECOLOR=color, ... , $
```

where:

`TYPE=REPORT`

The TYPE specification is optional with this feature. If omitted, TYPE defaults to REPORT.

`color`

Is a supported color. For a list of values, see *Formatting Report Data* on page 279.

Example: Setting Page Color

The following example illustrates how to set the page color of an HTML report with an internal cascading style sheet to silver.

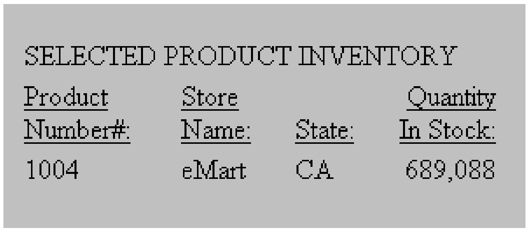
Report Request

```
TABLE FILE CENTORD
ON TABLE SUBHEAD
"SELECTED PRODUCT INVENTORY"
SUM QTY_IN_STOCK/D12 BY PROD_NUM BY SNAME BY STATE
WHERE PROD_NUM EQ '1004'
WHERE SNAME EQ 'eMart'
WHERE STATE EQ 'CA'
ON TABLE SET STYLE *
INCLUDE=IBFS:/stylesheet_location/custom_stylesheet.sty,$
END
```

***custom_stylesheet*.sty**

```
TYPE=REPORT, PAGECOLOR=SILVER, GRID=OFF, $
```

The output is:



```
SELECTED PRODUCT INVENTORY
Product     Store              Quantity
Number#:    Name:     State:   In Stock:
1004        eMart     CA        689,088
```

Setting Page Margins

You can set the page margins for your report. This includes the top, bottom, left, and right margins. You can also change the default unit of measurement (inches) to either centimeters or points. The unit of measurement applies to page margins, column width, and column position.

Reference: Page Margin Attributes

Attribute	Description	Applies to
`UNITS`	Sets the unit of measurement. Used when specifying margin size or other page characteristics. If you change the current unit of measurement, the new value is applied to all instances in which unit of measurement is used.	PDF HTML report with an internal cascading style sheet

`TOPMARGIN`
`BOTTOMMARGIN`
`LEFTMARGIN`
`RIGHTMARGIN`
	Sets the size of the top, bottom, left, and right margin.	PDF
		
		HTML report with an internal cascading style sheet

Syntax: How to Set the Unit of Measurement

The following syntax applies to a PDF or HTML report with an internal cascading style sheet.

```
UNITS=units
```

where:

`units`

Is the unit of measure. Values can be:

- ❏ **INCHES,** which specifies the unit of measure as inches. This is the default value.
- ❏ **CM,** which specifies the unit of measure as centimeters.
- ❏ **PTS,** which specifies the unit of measure as points. Points is a common measurement scale for typefaces.

Syntax: How to Set Margin Size

The following syntax applies to a PDF or HTML report with an internal cascading style sheet.

```
[TYPE=REPORT,] [TOPMARGIN={value|.25},] [BOTTOMMARGIN={value|.25},]
   [LEFTMARGIN={value|.25},] [RIGHTMARGIN={value|.25},] $
```

where:

`TYPE=REPORT`

Applies the margin size to the entire report. Not required, as it is the default.

`TOPMARGIN`

Sets the top boundary of the report content.

`BOTTOMMARGIN`

Sets the bottom boundary of the report content.

`LEFTMARGIN`

Sets the left boundary of the report content.

`RIGHTMARGIN`

Sets the right boundary of the report content.

`value`

Is the size of the specified margin. The report content displays inside the margin. If printing a report, specify a value compatible with the print area of the printer. For example, if the print area has 0.25 inch margins all around, set the margins to 0.25 inches, or larger.

The default value for all margins is 0.25 inches.

Example: Setting the Left Margin

The following example illustrates how to set the left margin of an HTML report with an internal cascading style sheet to one inch.

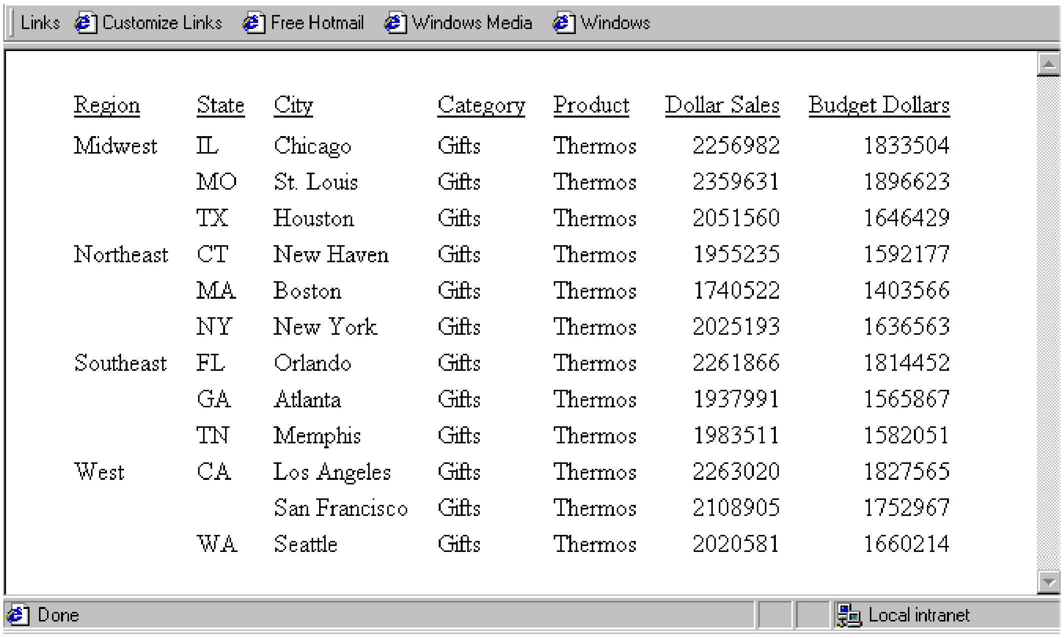
Report Request

```
TABLE FILE GGSALES
SUM CATEGORY PRODUCT DOLLARS BUDDOLLARS
BY REGION BY ST BY CITY
WHERE DOLLARS GT BUDDOLLARS
ON TABLE SET STYLE *
INCLUDE=IBFS:/stylesheet_location/custom_stylesheet.sty,$
END
```

***custom_stylesheet*.sty**

```
TYPE=REPORT, GRID=OFF, LEFTMARGIN=1, $
```

The output is:



Region	State	City	Category	Product	Dollar Sales	Budget Dollars
Midwest	IL	Chicago	Gifts	Thermos	2256982	1833504
	MO	St. Louis	Gifts	Thermos	2359631	1896623
	TX	Houston	Gifts	Thermos	2051560	1646429
Northeast	CT	New Haven	Gifts	Thermos	1955235	1592177
	MA	Boston	Gifts	Thermos	1740522	1403566
	NY	New York	Gifts	Thermos	2025193	1636563
Southeast	FL	Orlando	Gifts	Thermos	2261866	1814452
	GA	Atlanta	Gifts	Thermos	1937991	1565867
	TN	Memphis	Gifts	Thermos	1983511	1582051
West	CA	Los Angeles	Gifts	Thermos	2263020	1827565
		San Francisco	Gifts	Thermos	2108905	1752967
	WA	Seattle	Gifts	Thermos	2020581	1660214

Positioning a Report Component

A StyleSheet enables you to specify an absolute or relative starting position for a column, heading, footing, or element in a heading or footing. You can also add blank space around a report component.

This topic addresses column positioning using the POSITION StyleSheet attribute.

For details on positioning a heading or footing, or an element in a heading or footing, see *Using Headings, Footings, Titles, and Labels* on page 231.

Reference: Positioning Attributes

Attribute	Description	Applies to
`POSITION`	Sets the absolute or relative starting position of a column. An absolute position is the distance from the left margin of the printed paper. A relative position is the distance from the default position. After the first column, the default position is the end of the preceding column.	PDF
`TOPGAP` `BOTTOMGAP`	Adds blank space to the top or bottom of a report line.	PDF
`LEFTGAP` `RIGHTGAP`	Adds blank space to the left or right of a report column.	PDF

Syntax: How to Specify the Starting Position of a Column

The following syntax applies to a PDF report.

```
TYPE=REPORT, COLUMN=identifier, POSITION={+|-}position, $
```

where:

`identifier`

Selects a single column and collectively positions the column title, data, and totals if applicable. For valid values, see *Identifying a Report Component in a Web Query StyleSheet* on page 57.

`+`

Starts the column at the specified distance to the right of the default starting position.

By default, text items and alphanumeric fields are left-justified in a column, and numeric fields are right-justified in a column.

-

Starts the column at the specified distance to the left of the default starting position.

It is possible to create a report in which columns overlap. If this occurs, simply adjust the values.

position

Is the desired distance, in the unit of measurement specified with the UNITS attribute.

Example: Specifying an Absolute Starting Position for a Column

The following example illustrates how to position a column in a printed report. It is specified in the request that the PRODUCT_DESCRIPTION field display three inches from the left margin of the PDF report.

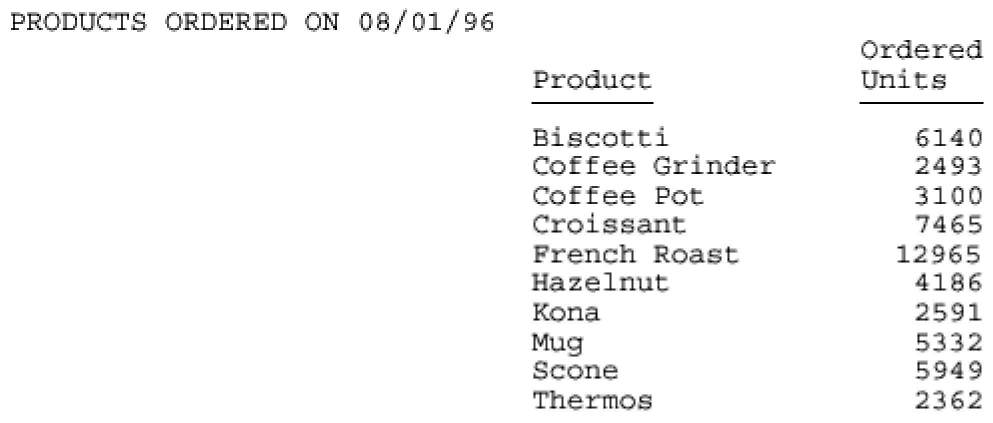
Report Request

```
TABLE FILE GGORDER
"PRODUCTS ORDERED ON 08/01/96"
SUM QUANTITY BY PRODUCT_DESCRIPTION
WHERE ORDER_DATE EQ '080196'
ON TABLE PCHOLD FORMAT PDF
ON TABLE SET STYLE *
INCLUDE=IBFS:/stylesheet_location/custom_stylesheet.sty,$
END
```

***custom_stylesheet*.sty**

```
TYPE=REPORT, COLUMN=PRODUCT_DESCRIPTION, POSITION=3, $
```

The output is:



```
PRODUCTS ORDERED ON 08/01/96
                                                   Ordered
                               Product             Units
                               -------             -------

                               Biscotti               6140
                               Coffee Grinder         2493
                               Coffee Pot             3100
                               Croissant              7465
                               French Roast          12965
                               Hazelnut               4186
                               Kona                   2591
                               Mug                    5332
                               Scone                  5949
                               Thermos                2362
```

Example: Specifying a Relative Starting Position for a Column

The following example illustrates how to position the column title and data for the QUANTITY field two inches from the default position, in this case, two inches from the end of the preceding column.

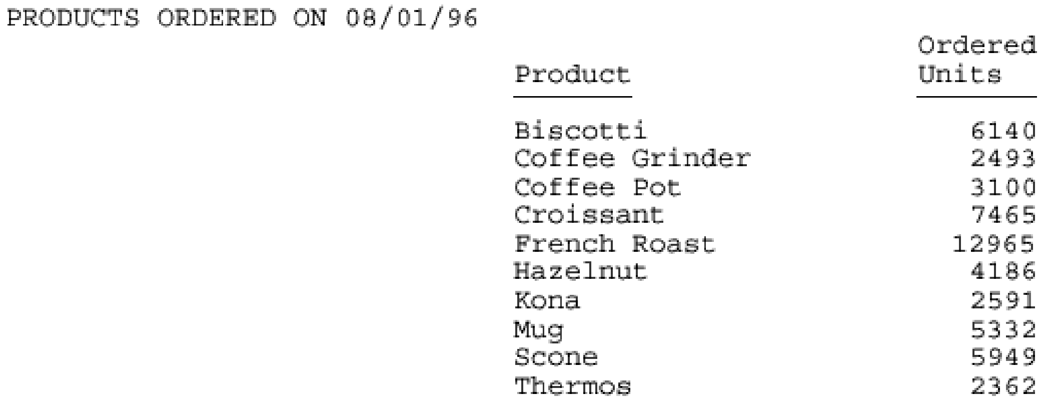
Report Request

```
TABLE FILE GGORDER
"PRODUCTS ORDERED ON 08/01/96"
SUM QUANTITY BY PRODUCT_DESCRIPTION
WHERE ORDER_DATE EQ '080196'
ON TABLE PCHOLD FORMAT PDF
ON TABLE SET STYLE *
INCLUDE=IBFS:/stylesheet_location/custom_stylesheet.sty,$
END
```

***custom_stylesheet*.sty**

```
TYPE=REPORT, COLUMN=PRODUCT_DESCRIPTION, POSITION=3, $
TYPE=REPORT, COLUMN=QUANTITY, POSITION=+2, $
```

QUANTITY, titled Ordered Units in the report, is relatively positioned to Product:



```
PRODUCTS ORDERED ON 08/01/96
                                                   Ordered
                               Product             Units
                               -------             -------

                               Biscotti               6140
                               Coffee Grinder         2493
                               Coffee Pot             3100
                               Croissant              7465
                               French Roast          12965
                               Hazelnut               4186
                               Kona                   2591
                               Mug                    5332
                               Scone                  5949
                               Thermos                2362
```

Syntax: How to Add Blank Space Around a Report Component

The following syntax applies to a PDF report.

```
TYPE=REPORT, {TOPGAP|BOTTOMGAP}=gap, $
TYPE=type, [COLUMN=identifier,|ACROSSCOLUMN=acrosscolumn,]
   {LEFTGAP|RIGHTGAP}=gap, $
TYPE=type, [COLUMN=identifier,|ACROSSCOLUMN=acrosscolumn,]
   {LEFTGAP|RIGHTGAP}=gap, $
```

where:

`TOPGAP`

Indicates how much space to add above the report line.

`BOTTOMGAP`

Indicates how much space to add below the report line.

`gap`

Is the amount of blank space, in the unit of measurement specified with the UNITS attribute.

In the absence of grids or background color, the default value is 0. For RIGHTGAP, the default value is proportional to the size of the text font.

In the presence of grids or background color, the default value increases to provide space between the grid and the text or to extend the color beyond the text.

The gaps must be the same within a single column or row. That is, you cannot specify different left or right gaps for individual cells in the same column, or different top and bottom gaps for individual cells in the same row.

`type`

Identifies the report component. For valid values, see *Identifying a Report Component in a Web Query StyleSheet* on page 57.

`identifier`

Selects one or more columns using the COLUMN attribute described in *Identifying a Report Component in a Web Query StyleSheet* on page 57.

`acrosscolumn`

Selects the same column under every occurrence of an ACROSS sort field using the ACROSSCOLUMN attribute described in *Identifying a Report Component in a Web Query StyleSheet* on page 57.

`LEFTGAP`

Indicates how much space to add to the left of a report column.

`RIGHTGAP`

Indicates how much space to add to the right of a report column.

Note: For TOPGAP, BOTTOMGAP, LEFTGAP, and RIGHTGAP, you must specify a value of at least 0.013889 (the decimal size of a point in inches). If you specify a value less than this, Web Query will round down to the nearest point, which is zero (0).

Example: Adding Blank Space Above Data Values

The following example illustrates how to add one-tenth of an inch of blank space above every data value in a PDF report.

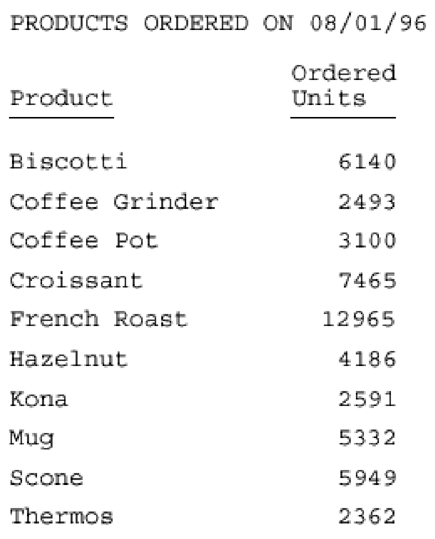
Report Request

```
TABLE FILE GGORDER
"PRODUCTS ORDERED ON 08/01/96"
" "
SUM QUANTITY BY PRODUCT_DESCRIPTION
WHERE ORDER_DATE EQ '080196'
ON TABLE PCHOLD FORMAT PDF
ON TABLE SET STYLE *
INCLUDE=IBFS:/stylesheet_location/custom_stylesheet.sty,$
END
```

***custom_stylesheet*.sty**

```
TYPE=DATA, TOPGAP=0.1, $
```

The data is spaced for readability:



```
PRODUCTS ORDERED ON 08/01/96

                    Ordered
Product             Units
-------             -------

Biscotti               6140

Coffee Grinder         2493

Coffee Pot             3100

Croissant              7465

French Roast          12965

Hazelnut               4186

Kona                   2591

Mug                    5332

Scone                  5949

Thermos                2362
```

Example: Adding Blank Space to the Left of a Column

The following example illustrates how to add blank space to the left of a report component. In this example, 1.5 inches of blank space are inserted to the left of the Product Category column.

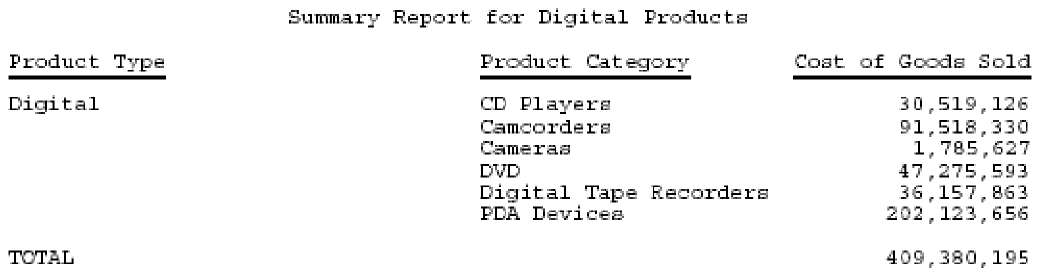
Report Request

```
TABLE FILE CENTORD
HEADING CENTER
"Summary Report for Digital Products"
" "
SUM LINE_COGS/D12 AS 'Cost of Goods Sold'
BY PRODTYPE AS 'Product Type'
BY PRODCAT AS 'Product Category'
WHERE PRODTYPE EQ 'Digital';
ON TABLE COLUMN-TOTAL/D12
ON TABLE PCHOLD FORMAT PDF
ON TABLE SET STYLE *
INCLUDE=IBFS:/stylesheet_location/custom_stylesheet.sty,$
END
```

***custom_stylesheet*.sty**

```
TYPE=REPORT, COLUMN=PRODCAT, LEFTGAP=1.5, $
```

The output is:



```
                       Summary Report for Digital Products

Product Type                          Product Category          Cost of Goods Sold

Digital                               CD Players                        30,519,126
                                      Camcorders                        91,518,330
                                      Cameras                            1,785,627
                                      DVD                               47,275,593
                                      Digital Tape Recorders            36,157,863
                                      PDA Devices                      202,123,656

TOTAL                                                                  409,380,195
```

Arranging Columns on a Page

How easily a user locates data depends on the arrangement of columns on a page. Using StyleSheet attributes, you can:

- ❑ Determine column width.
- ❑ Change the order of vertical sort (BY) columns.

Reference: Column Arrangement Features

Feature	Description	Applies to
`SQUEEZE`	Sets the column width.	HTML PDF
`SET SPACES`	Sets number of spaces between columns.	HTML
`SEQUENCE`	Sets the column order.	PDF HTML EXL2K **Note:** Does not work with XLSX and EXL2K FORMULA.
`FOLD-LINE`	Reduces report width by stacking columns.	PDF
`OVER`	Stacks columns by placing them over one another.	PDF HTML
`IN {n|+n}`	Sets absolute or relative starting position of a column.	PDF HTML

Determining Column Width

The value of the SQUEEZE attribute in a StyleSheet determines column width in a report.

When SQUEEZE is set to ON (the default), StyleSheet column width is ignored. Column width is determined using your browser default settings.

When using SQUEEZE, it may affect the way headings, footings, and column titles display in your report. For details, see *Using Headings, Footings, Titles, and Labels* on page 231.

Syntax: How to Determine Column Width (HTML)

The following syntax applies to an HTML report. For the syntax for a PDF report, see *How to Determine Column Width (PDF)* on page 141.

```
[TYPE=REPORT,] SQUEEZE={ON|OFF}, $
```

where:

`TYPE=REPORT`

Applies the column width to the entire report. Not required, as it is the default.

`ON`

Determines column width based on the widest data value or column title, whichever is greater. ON is the default value.

For HTML reports, the web browser shrinks the column width to the shortest column title or field value.

`OFF`

Determines column width based on the field format in the Master File. Blank spaces pad the column width up to the length of the column title or field format, whichever is greater.

Note: SQUEEZE is not supported for columns created with the OVER phrase.

Example: Using Default Column Width (HTML)

The following example uses SQUEEZE=ON (the default) for an HTML report. Column width is based on the wider of the data value or column title.

Report Request

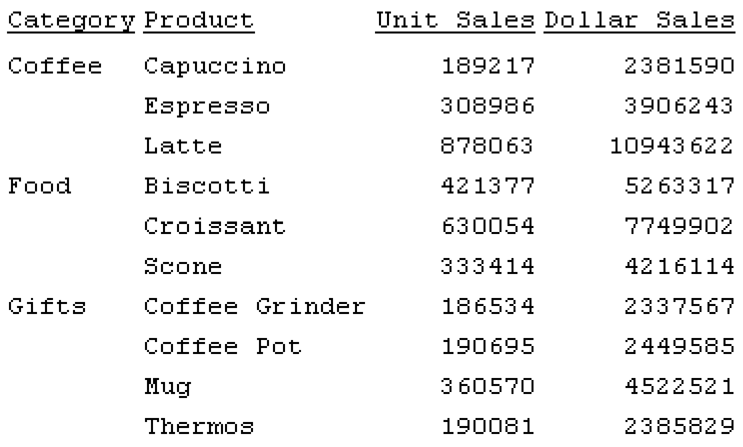
```
TABLE FILE GGSALES
SUM UNITS DOLLARS
BY CATEGORY BY PRODUCT
ON TABLE SET STYLE *
INCLUDE=IBFS:/stylesheet_location/custom_stylesheet.sty,$
END
```

***custom_stylesheet*.sty**

```
TYPE=REPORT, GRID=OFF, FONT=COURIER, $
```

For Category, Unit Sales, and Dollar Sales, the column title is wider than the corresponding data values. For Product, the wider data values determine column width.

The output is:



```
Category Product         Unit Sales Dollar Sales

Coffee   Capuccino           189217      2381590
         Espresso            308986      3906243
         Latte               878063     10943622
Food     Biscotti            421377      5263317
         Croissant           630054      7749902
         Scone               333414      4216114
Gifts    Coffee Grinder      186534      2337567
         Coffee Pot          190695      2449585
         Mug                 360570      4522521
         Thermos             190081      2385829
```

Example: **Using Column Width Based on Field Format in Master File (HTML)**

The following example sets SQUEEZE to OFF for an HTML report. Column width is based on the field format in the Master File.

Report Request

```
TABLE FILE GGSALES
SUM UNITS DOLLARS
BY CATEGORY BY PRODUCT
ON TABLE SET STYLE *
INCLUDE=IBFS:/stylesheet_location/custom_stylesheet.sty,$
END
```

***custom_stylesheet*.sty**

```
TYPE=REPORT, GRID=OFF, SQUEEZE=OFF, FONT=COURIER, $
```

Blank spaces pad the column width up to the length of the field format for Category (A11) and Product (A16).

The output is:

```
Category    Product             Unit Sales Dollar Sales

Coffee      Capuccino              189217      2381590
            Espresso               308986      3906243
            Latte                  878063     10943622
Food        Biscotti               421377      5263317
            Croissant              630054      7749902
            Scone                  333414      4216114
Gifts       Coffee Grinder         186534      2337567
            Coffee Pot             190695      2449585
            Mug                    360570      4522521
            Thermos                190081      2385829
```

Syntax: How to Determine Column Width (PDF)

The follolwing syntax applies to a PDF report. For the syntax for an HTML report, see *How to Determine Column Width (HTML)* on page 139.

```
[TYPE=REPORT,] COLUMN=column, SQUEEZE={ON|OFF|width}, $
```

where:

`TYPE=REPORT`

Applies the column width to the entire report. Not required, as it is the default.

`column`

Selects a column using the COLUMN attribute described in *Identifying a Report Component in a Web Query StyleSheet* on page 57. If you omit a column, the value for SQUEEZE applies to all columns in a report.

`ON`

Determines column width based on the widest data value or column title, whichever is greater.

`OFF`

Determines column width based on the field format in the Master File. Blank spaces pad the column width up to the length of the column title or field format, whichever is greater. OFF is the default value.

`width`

Is a measurement for the column width, specified with the UNITS attribute.

If the widest data value exceeds the specified measurement:

And the field is...	The following displays...
Alphanumeric	As much of the value as will fit in the specified width, followed by an exclamation mark (!) to indicate truncation.
Numeric	Asterisks (*) in place of the field value.

Note: SQUEEZE is not supported for columns created with the OVER phrase.

Example: Determining Column Width (PDF)

The following example uses SQUEEZE=2.5 to increase the default column width of the PRODUCT field in a PDF report. Note that this feature is used primarily for printed reports. Depending on your screen resolution, the column width may look different than how it will print.

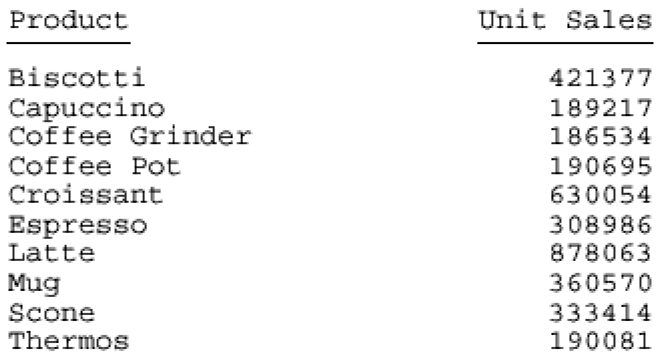
Report Request

```
TABLE FILE GGSALES
SUM UNITS
BY PRODUCT
ON TABLE PCHOLD FORMAT PDF
ON TABLE SET STYLE *
INCLUDE=IBFS:/stylesheet_location/custom_stylesheet.sty,$
END
```

***custom_stylesheet*.sty**

```
TYPE=REPORT, COLUMN=PRODUCT, SQUEEZE=2.5, $
```

The output is:



```
Product                          Unit Sales

Biscotti                             421377
Capuccino                            189217
Coffee Grinder                       186534
Coffee Pot                           190695
Croissant                            630054
Espresso                             308986
Latte                                878063
Mug                                  360570
Scone                                333414
Thermos                              190081
```

Changing Column Order

You can change the order in which vertical sort (BY) columns are displayed in a report. This feature does not apply to horizontal sort (ACROSS) rows or stacked (OVER) columns.

Syntax: How to Change Column Order

This syntax applies to PDF, HTML, XLSX, and EXL2K reports. XLSX FORMULA and EXL2K FORMULA formats are not supported.

```
[TYPE=REPORT,] COLUMN=column, SEQUENCE=sequence, $
```

where:

`TYPE=REPORT`

Applies the column order to the entire report. Not required, as it is the default.

`column`

Selects a column using the COLUMN attribute described in *Identifying a Report Component in a Web Query StyleSheet* on page 57.

`sequence`

Is a number that represents the order of the selected column.

Numbers need not be in sequential order or in increments of one. The order of the columns is from lowest to highest. NOPRINT columns are not included.

Example: Changing Column Order

The following example rearranges the order in which columns normally appear in the report, that is, with SNAME first, PRODCAT second, and LINEPRICE third.

Report Request

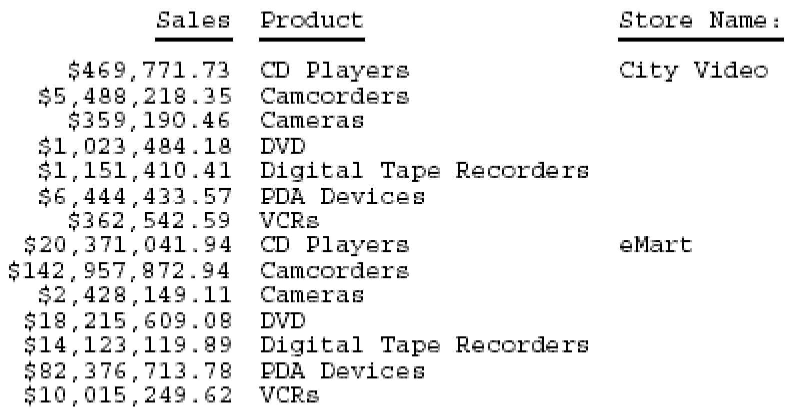
```
TABLE FILE CENTORD
SUM LINEPRICE AS 'Sales'
BY SNAME BY PRODCAT AS 'Product'
WHERE SNAME EQ 'eMart' OR 'City Video'
ON TABLE PCHOLD FORMAT PDF
ON TABLE SET STYLE *
INCLUDE=IBFS:/stylesheet_location/custom_stylesheet.sty,$
END
```

***custom_stylesheet*.sty**

```
TYPE=REPORT, COLUMN=SNAME, SEQUENCE=3, $
TYPE=REPORT, COLUMN=PRODCAT, SEQUENCE=2, $
TYPE=REPORT, COLUMN=LINEPRICE, SEQUENCE=1, $
```

LINEPRICE (Sales) is now the first column, PRODCAT (Product) is the second column (as it was by default), and SNAME (Store Name) is the third column.

The output is:

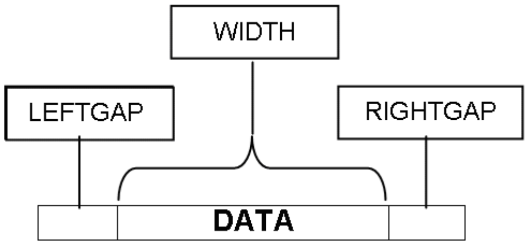


```
          Sales  Product                  Store Name:

    $469,771.73  CD Players               City Video
  $5,488,218.35  Camcorders
    $359,190.46  Cameras
  $1,023,484.18  DVD
  $1,151,410.41  Digital Tape Recorders
  $6,444,433.57  PDA Devices
    $362,542.59  VCRs
 $20,371,041.94  CD Players               eMart
$142,957,872.94  Camcorders
  $2,428,149.11  Cameras
 $18,215,609.08  DVD
 $14,123,119.89  Digital Tape Recorders
 $82,376,713.78  PDA Devices
 $10,015,249.62  VCRs
```

Alignment of Fields in Reports Using OVER in PDF Report Output

When columns are placed on report output, they are separated by gaps. You can control the size of the gaps between columns with the LEFTGAP and RIGHTGAP StyleSheet attributes.

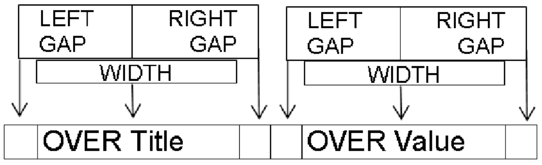
By default, the gaps between columns are placed outside of the boundaries reserved for the fields on the report output. Therefore, the width or squeeze value defined for a field defines the size of the text area for the data value. It does not count the width of the gaps between columns. The bounding box used to define borders and background color is determined based on the data width plus the left gap plus the right gap.



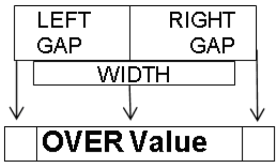
Gaps external to the column boundaries must be accounted for when you try to align fields in reports that use the OVER phrase.

This feature is designed to support the development of multi-row reports using blank AS names (column titles). Unless otherwise noted, these features work with non-blank titles, but they have not been designed to support alignment with non-blank column titles.

By default, column titles are placed to the left of the field values in a report using OVER. The OVER Title and the OVER Value each are measured by the combination of three parameters, LEFTGAP, WIDTH, and RIGHTGAP:



With OVER and blank AS names, each data value becomes a data cell that can be used to construct rows and columns within the data lines of the report. In order to align data values on a lower line with the columns above them, you must calculate widths for the lower-level columns that take into account the widths of the data above them plus the widths of all of the left gaps and right gaps in between.



It can be complex to calculate how to size each column when aligning data and headings in reports using OVER. Each calculation of the column size must additionally account for the external left and right gap, and these gaps are cumulative as the number of columns on a given row increases.

Using the GAPINTERNAL=ON StyleSheet attribute, you can have the gaps placed within the column boundaries for PDF report output. This feature makes it much easier to align fields and headings in reports that use the OVER phrase to create multiple lines.

Note: OVER is supported with SQUEEZE.

Syntax: How to Control GAP Placement on Reports

```
TYPE=REPORT, GAPINTERNAL={OFF|ON}
```

where:

OFF

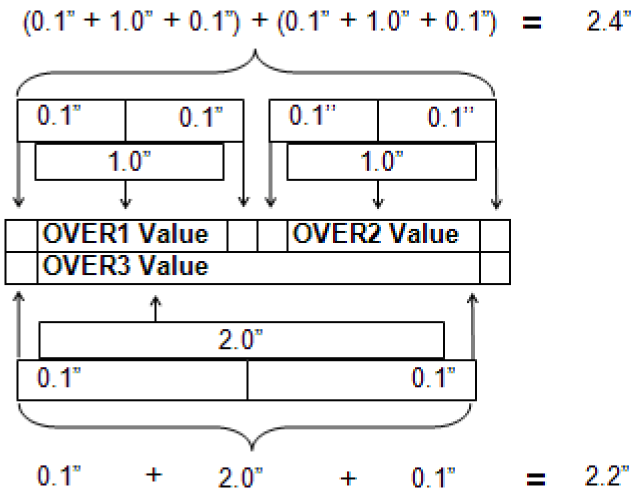
Places the left and right gaps outside the defined field width. OFF is the default value.

ON

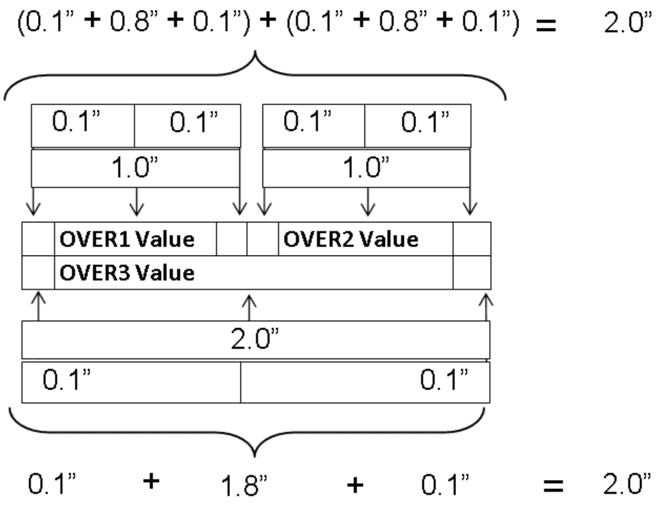
Places the left and right gaps internal to the defined field width.

Example: Comparing External Gaps With Internal Gaps

With GAPINTERNAL=OFF, you must account for the accumulation of left and right gaps, as well as the field widths when defining widths of stacked columns.



With GAPINTERNAL=ON, the defined WIDTH represents the entire space used by the given data cell or column. This takes the cumulative effect out as the OVER values proceed across a row.



Example: Using GAPINTERNAL in a Report

The following request against the GGSALES data source places the PRODUCT field over the UNITS and DOLLARS fields and sets GAPINTERNAL to OFF.

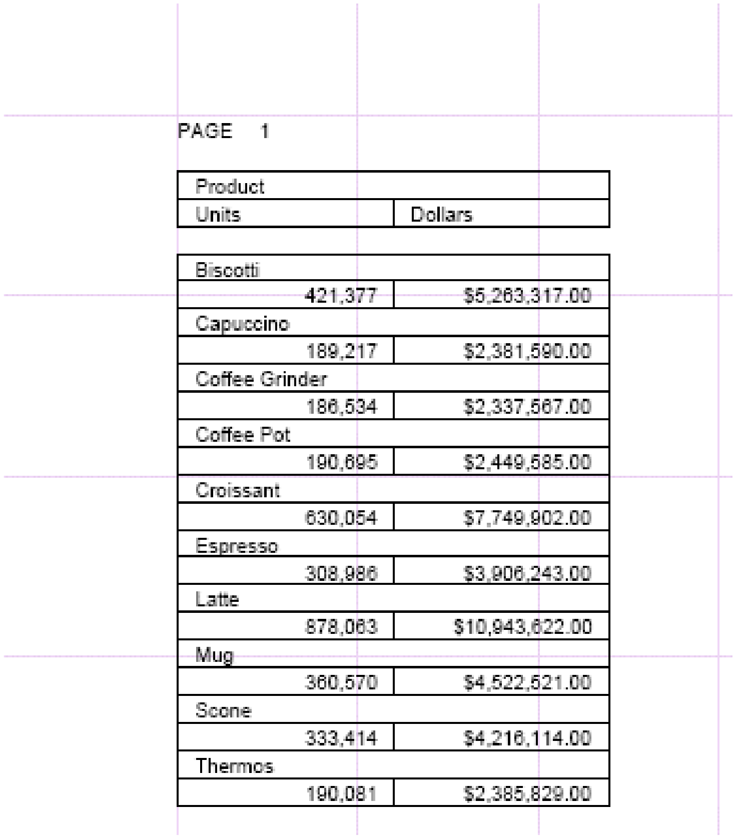
Report Request

```
SET LAYOUTGRID=ON
TABLE FILE GGSALES
"Product<+0>"
"Units<+0>Dollars"
SUM
PRODUCT AS ''
OVER
UNITS/D8C AS '' DOLLARS/D12.2CM AS ''
BY PRODUCT NOPRINT
ON TABLE PCHOLD FORMAT PDF
ON TABLE SET STYLE *
INCLUDE=IBFS:/stylesheet_location/custom_stylesheet.sty,$
END
```

***custom_stylesheet*.sty**

```
TYPE=REPORT, SQUEEZE=ON, BORDER=ON, FONT=ARIAL, SIZE=8, LEFTMARGIN=1,
   TOPMARGIN=1, LEFTGAP=.1, RIGHTGAP=.1, GAPINTERNAL=OFF, $
TYPE=HEADING, BORDERALL=ON, $
TYPE=HEADING, LINE=1, ITEM=1, POSITION=PRODUCT, $
TYPE=HEADING, LINE=2, ITEM=1, POSITION=UNITS, $
TYPE=HEADING, LINE=2, ITEM=2, POSITION=DOLLARS, $
TYPE=REPORT, COLUMN=PRODUCT(2), SQUEEZE=2, $
TYPE=REPORT, COLUMN=UNITS, SQUEEZE=1, $
TYPE=REPORT, COLUMN=DOLLARS, SQUEEZE=1, $
```

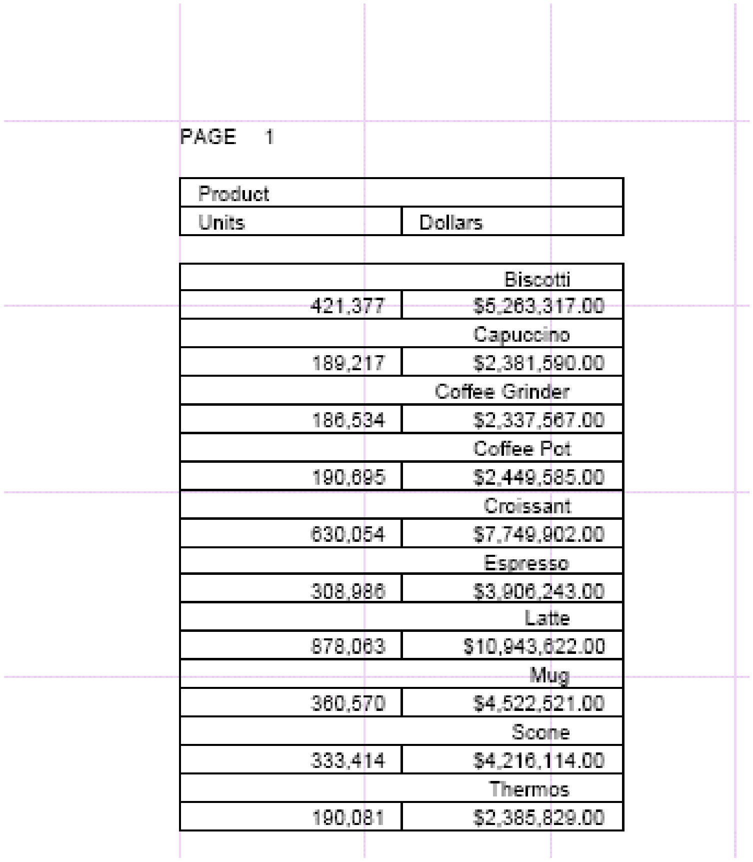
The widths specified for UNITS and DOLLARS are one inch each, while the PRODUCT field is specified to be two inches. With GAPINTERNAL=OFF, the LAYOUTGRID shows that the widths used to place the columns are greater than the widths specified in the request. The additional space presented by the external leftgap and rightgap accounts for this effect.



PAGE 1

Product	
Units	Dollars
Biscotti	
421,377	$5,263,317.00
Capuccino	
189,217	$2,381,590.00
Coffee Grinder	
186,534	$2,337,567.00
Coffee Pot	
190,695	$2,449,585.00
Croissant	
630,054	$7,749,902.00
Espresso	
308,986	$3,906,243.00
Latte	
878,063	$10,943,622.00
Mug	
360,570	$4,522,521.00
Scone	
333,414	$4,216,114.00
Thermos	
190,081	$2,385,829.00

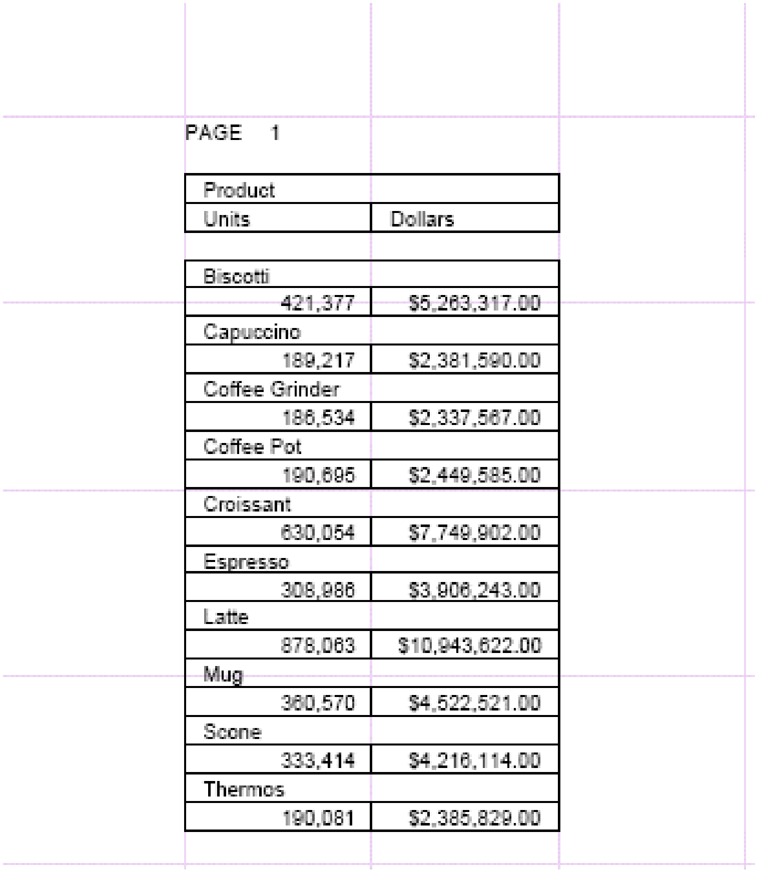
The heading borders are aligned on the right of the report because of the SQUEEZE=ON attribute in the StyleSheet. Extra space was added to the report to align the headings. If you change the StyleSheet declaration for the PRODUCTS field to JUSTIFY=RIGHT, you can see that the extra space prevents the product value from aligning with the dollar value.



PAGE 1

Product	
Units	Dollars
	Biscotti
421,377	$5,263,317.00
	Capuccino
189,217	$2,381,590.00
	Coffee Grinder
186,534	$2,337,567.00
	Coffee Pot
190,695	$2,449,585.00
	Croissant
630,054	$7,749,902.00
	Espresso
308,986	$3,906,243.00
	Latte
878,063	$10,943,622.00
	Mug
360,570	$4,522,521.00
	Scone
333,414	$4,216,114.00
	Thermos
190,081	$2,385,829.00

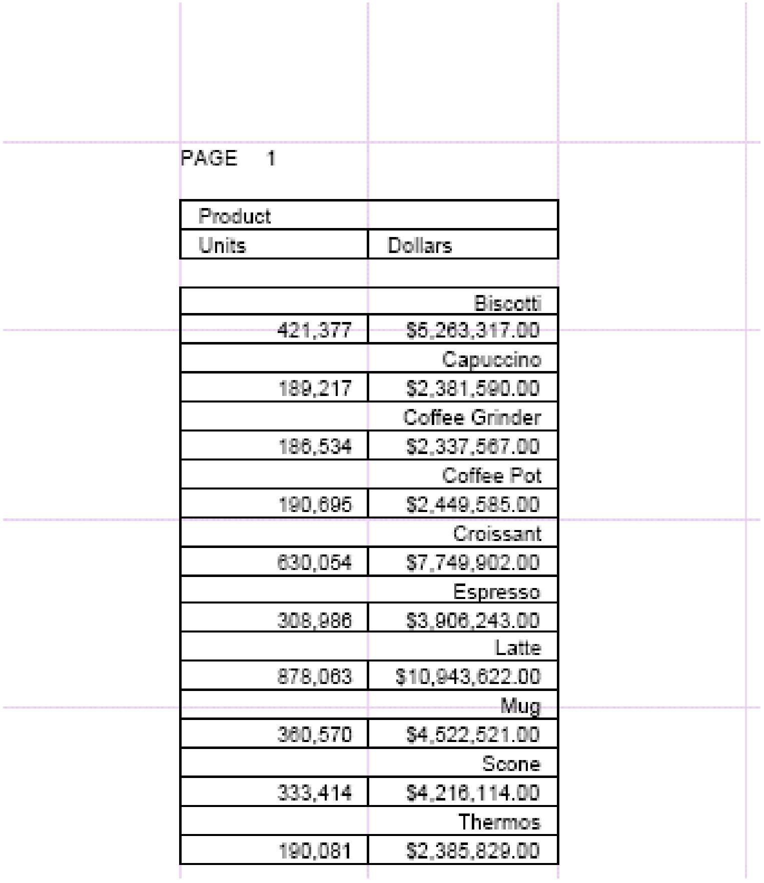
Changing the StyleSheet declaration to GAPINTERNAL=ON causes the specified widths to be used because the gaps are internal and are included in the specified values.



PAGE 1

Product	
Units	Dollars
Biscotti	
421,377	$5,263,317.00
Capuccino	
189,217	$2,381,590.00
Coffee Grinder	
186,534	$2,337,567.00
Coffee Pot	
190,695	$2,449,585.00
Croissant	
630,054	$7,749,902.00
Espresso	
308,986	$3,906,243.00
Latte	
878,063	$10,943,622.00
Mug	
360,570	$4,522,521.00
Scone	
333,414	$4,216,114.00
Thermos	
190,081	$2,385,829.00

The following output demonstrates that the values align properly even if the PRODUCT values are defined with JUSTIFY=RIGHT.



PAGE 1

Product	
Units	Dollars
	Biscotti
421,377	$5,263,317.00
	Capuccino
189,217	$2,381,590.00
	Coffee Grinder
186,534	$2,337,567.00
	Coffee Pot
190,695	$2,449,585.00
	Croissant
630,054	$7,749,902.00
	Espresso
308,986	$3,906,243.00
	Latte
878,063	$10,943,622.00
	Mug
360,570	$4,522,521.00
	Scone
333,414	$4,216,114.00
	Thermos
190,081	$2,385,829.00

Adding Grids and Borders

By default, an HTML report contains horizontal and vertical grid lines. You can remove the grid lines or adjust their use on a horizontal (BY) sort field. Grid characteristics apply to an entire HTML report, not to individual components of a report.

You can emphasize headings, footings, and column titles in a report by adding borders and grid lines around them.

Borders. In a PDF, HTML, DHTML, XLSX, EXL2K, PPTX, or PPT report, you can use BORDER attributes in a StyleSheet to specify the weight, style, and color of border lines. If you wish, you can specify formatting variations for the top, bottom, left, and right borders.

The BORDERALL StyleSheet attribute supports a heading or footing grid with borders around each individual heading or footing cell in PDF report output. Using this attribute along with BORDER attributes for individual objects in a heading or footing enables you to create borders around individual items.

Currently, with SQUEEZE=ON, the right margin border for subheadings and subfootings is defined based on the maximum width of all heading, footing, subheading, and subfooting lines. The length of subheading and subfooting lines is tied to the lengths of the page heading and page footing, not to the size of the data columns in the body of the report. You can use the ALIGN-BORDERS=BODY attribute in a StyleSheet to align the subheadings and subfootings with the report body on PDF report output, instead of the other heading elements.

Grids. In an HTML report, you can use the GRID attribute in a StyleSheet to turn grid lines on and off for the entire report. When used in conjunction with internal cascading style sheets, GRID produces a thin grid line rather than a thick double line (the HTML default). In PDF reports, you can use the HGRID and VGRID attributes to add horizontal or vertical grid lines and adjust their density.

Reference: Grid Display Attributes

Attribute	Description	Applies to
`GRID`	Controls grid display.	HTML
`HGRID`	Controls horizontal grid display and grid line density.	PDF
`VGRID`	Controls vertical grid display and grid line density.	PDF

Note: When viewing PDF reports with the Adobe Reader, GRID lines may appear thinner than specified if the view is not set to 100%. For example, if you view the document at the 50% setting, some GRID lines may be thinner than others.

Syntax: How to Control Grid Display in HTML Reports

```
[TYPE=REPORT,] GRID=option, $
```

where:

`TYPE=REPORT`

Applies the grid to the entire report. Not required, as it is the default.

`option`

Is one of the following:

<u>`ON`</u> applies a grid to a report. Does not apply grid lines to cells underneath a BY field value until the value changes. Column titles are not underlined. ON is the default value.

`OFF` disables the default grid. Column titles are underlined. You can include blank lines and underlines. You cannot wrap cell data. With this setting, a report may be harder to read.

`FILL` applies grid lines to all cells of a report. Column titles are not underlined.

Syntax: How to Add and Format Borders

To request a uniform border, use this syntax:

```
TYPE=type, BORDER=option, [BORDER-STYLE=line_style,]
   [BORDER-COLOR={color|RGB(r g b)},] $
```

To specify different characteristics for the top, bottom, left, and/or right borders, use this syntax:

```
TYPE=type, BORDER-position=option,
   [BORDER[-position]-STYLE=line_style,]
   [BORDER[-position]-COLOR={color|RGB(r g b)},] $
```

where:

`type`

Identifies the report component to which borders are applied. See *Identifying a Report Component in a Web Query StyleSheet* on page 57 for valid values.

option

Can be one of the following values:

ON turns borders on. ON generates the same line as MEDIUM.

Note: The MEDIUM line setting ensures consistency with lines created with GRID attributes.

OFF turns borders off. OFF is the default value.

LIGHT specifies a thin line.

MEDIUM identifies a medium line. ON sets the line to MEDIUM.

HEAVY identifies a thick line.

width specifies the line width in points, where 72 pts=1 inch. Note that this option is not supported with XLSX or EXL2K, which do not have an option for specifying a number to precisely set the border width (thickness) in points.

Tip: Line width specified in points is displayed differently in HTML and PDF output. For uniform appearance, regardless of display format, use LIGHT, MEDIUM, or HEAVY.

position

Specifies which border line to format. Valid values are: TOP, BOTTOM, LEFT, RIGHT.

You can specify a position qualifier for any of the BORDER attributes. This enables you to format line width, line style, and line color individually, for any side of the border.

line_style

Sets the style of the border line. Web Query StyleSheets support all of the standard cascading style sheet line styles. Several 3-dimensional styles are available only in HTML, as noted by asterisks. Valid values are:

Style	Description
NONE	No border is drawn.
SOLID	Solid line.
DOTTED	Dotted line.
DASHED	Dashed line.

Style	Description
DOUBLE	Double line.
GROOVE*	3D groove. Not supported with XLSX or EXL2K, which do not have an option for specifying this type of border.
RIDGE*	3D ridge. Not supported with XLSX or EXL2K, which do not have an option for specifying this type of border.
INSET*	3D inset.
OUTSET*	3D outset.

color

Is one of the preset color values. The default value is BLACK.

If the display or output device does not support colors, it substitutes shades of gray. For a complete list of available color values, see *Formatting Report Data* on page 279.

RGB

Specifies the font color using a mixture of red, green, and blue.

(*r g b*)

Is the desired intensity of red, green, and blue, respectively. The values are on a scale of 0 to 255, where 0 is the least intense and 255 is the most intense. Using the three color components in equal intensities results in shades of gray.

Example: Inserting and Formatting a Border

The following example illustrates how to generate an HTML report with a heavy red dotted line around the entire report heading.

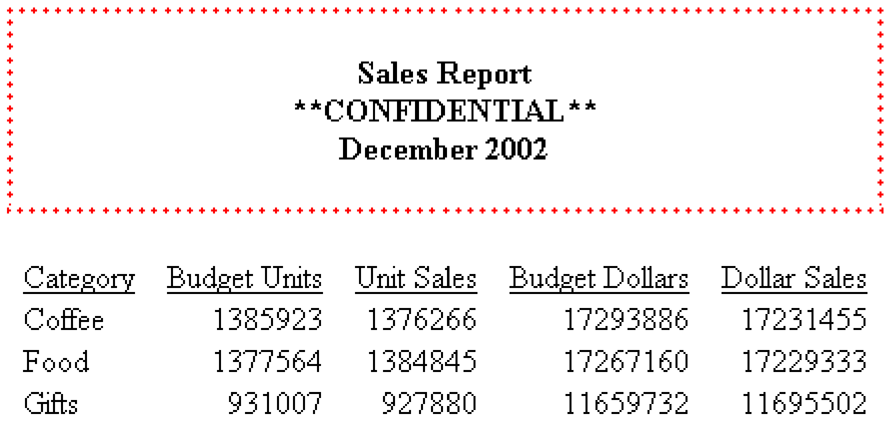
Report Request

```
TABLE FILE GGSALES
SUM BUDUNITS UNITS BUDDOLLARS DOLLARS
BY CATEGORY
ON TABLE SUBHEAD
"</1 Sales Report"
"**CONFIDENTIAL**"
"December 2002 </1"
ON TABLE PCHOLD FORMAT PDF
ON TABLE SET STYLE *
INCLUDE=IBFS:/stylesheet_location/custom_stylesheet.sty,$
END
```

***custom_stylesheet*.sty**

```
TYPE=REPORT, GRID=OFF, $
TYPE=TABHEADING, STYLE=BOLD, JUSTIFY=CENTER, BORDER=HEAVY,
   BORDER-COLOR=RED, BORDER-STYLE=DOTTED, $
```

The output is:



```
                    Sales Report
                  **CONFIDENTIAL**
                   December 2002

Category  Budget Units  Unit Sales  Budget Dollars  Dollar Sales
Coffee         1385923     1376266        17293886      17231455
Food           1377564     1384845        17267160      17229333
Gifts           931007      927880        11659732      11695502
```

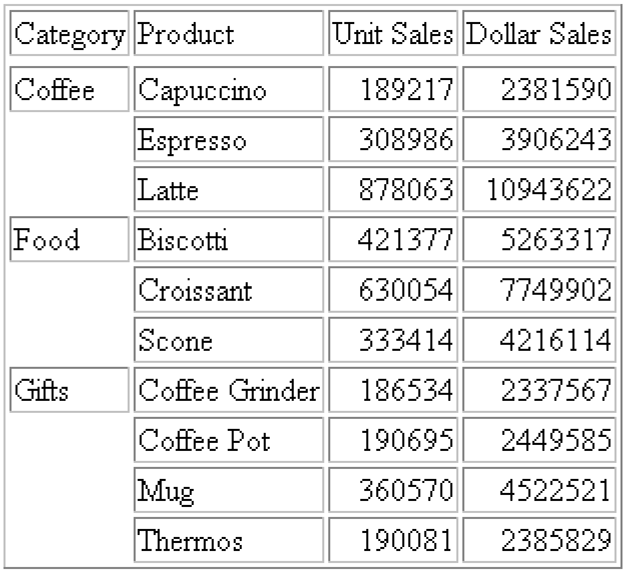
Tip: You can use the same BORDER syntax to generate this output in a PDF report.

Example: Displaying the Default Grid on an HTML Report

The following example uses the default setting GRID=ON.

```
TABLE FILE GGSALES
SUM UNITS DOLLARS
BY CATEGORY BY PRODUCT
END
```

The cells underneath the sort field CATEGORY do not have grid lines until the value changes (for example, from Coffee to Food).



Category	Product	Unit Sales	Dollar Sales
Coffee	Capuccino	189217	2381590
	Espresso	308986	3906243
	Latte	878063	10943622
Food	Biscotti	421377	5263317
	Croissant	630054	7749902
	Scone	333414	4216114
Gifts	Coffee Grinder	186534	2337567
	Coffee Pot	190695	2449585
	Mug	360570	4522521
	Thermos	190081	2385829

Example: Applying Grid Lines to All Cells of an HTML Report

The following example uses GRID=FILL to apply grid lines to all cells, including those underneath the sort field CATEGORY.

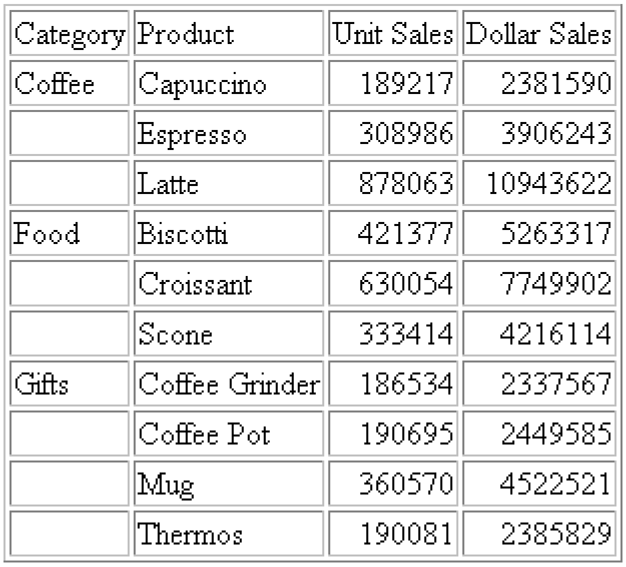
Report Request

```
TABLE FILE GGSALES
SUM UNITS DOLLARS
BY CATEGORY BY PRODUCT
ON TABLE SET STYLE *
INCLUDE=IBFS:/stylesheet_location/custom_stylesheet.sty,$
END
```

***custom_stylesheet*.sty**

```
TYPE=REPORT, GRID=FILL, $
```

All cells have grid lines:



Category	Product	Unit Sales	Dollar Sales
Coffee	Capuccino	189217	2381590
	Espresso	308986	3906243
	Latte	878063	10943622
Food	Biscotti	421377	5263317
	Croissant	630054	7749902
	Scone	333414	4216114
Gifts	Coffee Grinder	186534	2337567
	Coffee Pot	190695	2449585
	Mug	360570	4522521
	Thermos	190081	2385829

Example:

Removing a Grid From an HTML Report

The following example uses GRID=OFF to remove the default grid from a report.

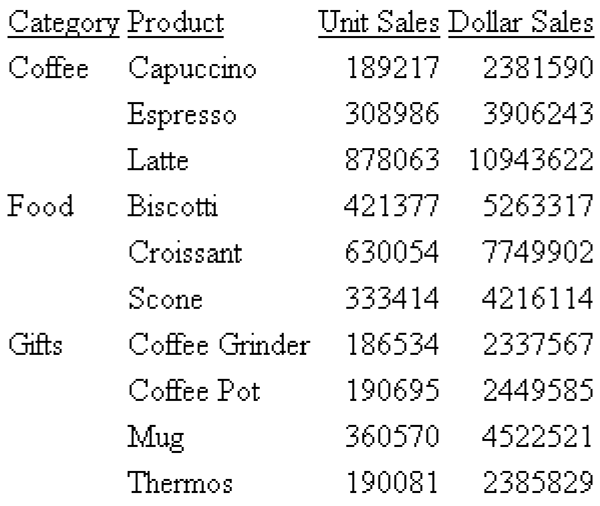
Report Request

```
TABLE FILE GGSALES
SUM UNITS DOLLARS
BY CATEGORY BY PRODUCT
ON TABLE SET STYLE *
INCLUDE=IBFS:/stylesheet_location/custom_stylesheet.sty,$
END
```

***custom_stylesheet*.sty**

```
TYPE=REPORT, GRID=OFF, $
```

Column titles are underlined:



Category	Product	Unit Sales	Dollar Sales
Coffee	Capuccino	189217	2381590
	Espresso	308986	3906243
	Latte	878063	10943622
Food	Biscotti	421377	5263317
	Croissant	630054	7749902
	Scone	333414	4216114
Gifts	Coffee Grinder	186534	2337567
	Coffee Pot	190695	2449585
	Mug	360570	4522521
	Thermos	190081	2385829

Syntax: How to Insert Inner and Outer Borders Within Headings or Footings in PDF Reports

BORDERALL is the quickest way to add borders to the entire heading grid. Individual borders can be removed by explicitly turning the border off in individual items using BORDER, BORDER-LEFT, BORDER-RIGHT, BORDER-TOP, and BORDER-BOTTOM. For a given item that is bordered by BORDERALL, BORDER-LEFT=OFF presents the item with no left border, but the defined border style is retained for top, bottom, and right borders.

Three levels of borders for headings and footings are supported:

1. Individual cell borders.

   BORDER-LEFT, BORDER-RIGHT, BORDER-TOP, and BORDER-BOTTOM can be used to set the individual components of the external border of the heading or a selected item or cell.

2. All outer borders.

   BORDER= is used to set the external borders within a heading or footing.

3. All outer and internal borders.

   BORDERALL is used to apply border characteristics to both the internal and external borders of the selected heading or footing.

Note: BORDERALL applies to the entire heading or footing element. It cannot be used for individual lines or items within a heading or footing element.

To turn on all external and internal borders (a border grid):

```
TYPE=headfoot, BORDERALL=option, [BORDER-STYLE=line_style,]
   [BORDER-COLOR={color|RGB(rgb)},] $
```

where:

headfoot

Is the type of heading or footing. Valid values are TABHEADING, TABFOOTING, HEADING, FOOTING, SUBHEAD, and SUBFOOT.

Note: BORDERALL applies to the entire heading or footing element. It cannot be used for individual lines or items within a heading or footing element.

To request a uniform border, use this syntax:

```
TYPE=headfoot, BORDER=option
```

To specify different characteristics for the top, bottom, left, and/or right borders, use this syntax:

```
TYPE=headfoot, BORDER-position=option,
   [BORDER[-position]-STYLE=line_style,]
   [BORDER[-position]-COLOR={color|RGB(rgb)},] $
```

where:

headfoot

Identifies the heading, footing, subheading, or subfooting to which borders are applied.

option

Can be one of the following values:

`ON` turns borders on. ON generates the same line as MEDIUM.

Note: The MEDIUM line setting ensures consistency with lines created with GRID attributes.

`OFF` turns borders off. OFF is the default value.

`LIGHT` specifies a thin line.

`MEDIUM` identifies a medium line. ON sets the line to MEDIUM.

`HEAVY` identifies a thick line.

width specifies the line width in points, where 72 pts=1 inch. Note that this option is not supported with XLSX or EXL2K, which do not have an option for specifying a number to precisely set the border width (thickness) in points.

Tip: Line width specified in points is displayed differently in HTML and PDF output. For uniform appearance, regardless of display format, use LIGHT, MEDIUM, or HEAVY.

position

Specifies which border line to format. Valid values are: TOP, BOTTOM, LEFT, RIGHT.

You can specify a position qualifier for any of the BORDER attributes. This enables you to format line width, line style, and line color individually, for any side of the border.

line_style

Sets the style of the border line. Web Query StyleSheets support all of the standard cascading style sheet line styles. Several three-dimensional styles are available only in HTML, as noted by asterisks. Valid values are:

Style	Description
NONE	No border is drawn.
SOLID	Solid line.
DOTTED	Dotted line.
DASHED	Dashed line.
DOUBLE	Double line.
GROOVE*	3D groove. (Not supported with Excel 2003, which has no option for specifying this type of border.)
RIDGE*	3D ridge. (Not supported with Excel 2003, which has no option for specifying this type of border.)
INSET*	3D inset.
OUTSET*	3D outset.

Note: All line types supported for PDF can be used for individual internal borders with HEADALIGN=BODY.

color

Is one of the preset color values. The default value is BLACK.

If the display or output device does not support colors, it substitutes shades of gray. For a complete list of available color values, see the *Color Values in a Report* on page 284.

RGB

Specifies the font color using a mixture of red, green, and blue.

(*rgb*)

Is the desired intensity of red, green, and blue, respectively. The values are on a scale of 0 to 255, where 0 is the least intense and 255 is the most intense. Using the three color components in equal intensities results in shades of gray.

Example: Controlling Borders Within Heading and Footing Elements in PDF Report Output

The following request against the EMPLOYEE data source has a page heading, a subheading, a subfooting, and a report footing.

- The REPORT component has BORDER=ON, so the page heading has an external border.
- The subheading has BORDERALL=ON and HEADALIGN=BODY, so the subheading grid aligns with the body grid, and each item within the subhead is presented as fully bordered individual cells.
- The StyleSheet aligns the subfooting elements with the body of the report, and has the salary subtotal on the second line aligned and justified with the CURR_SAL column.
- The table footing has a border around the entire footing because the REPORT component specifies BORDER=ON. The grand total is aligned and justified with the CURR_SAL column on the report.

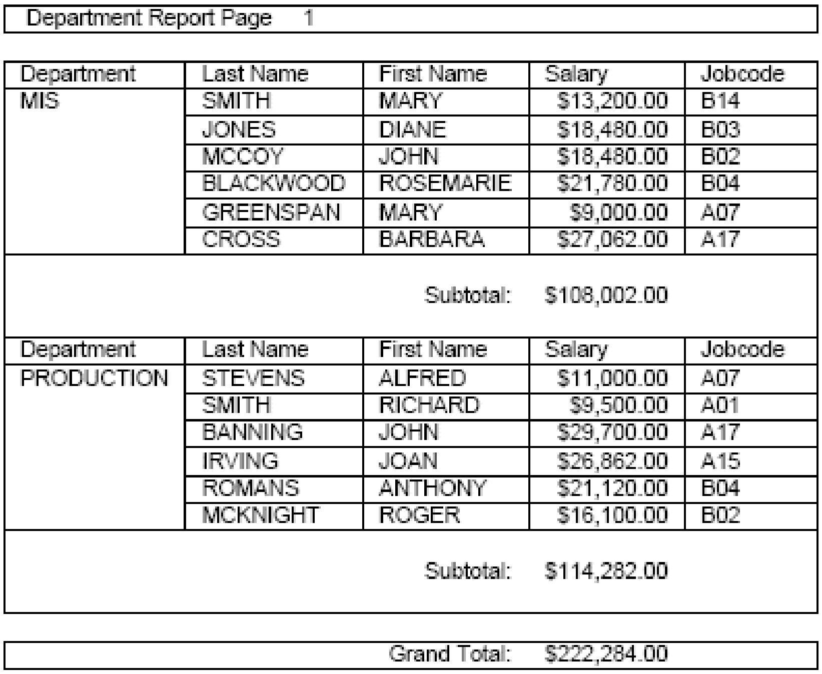
Report Request

```
TABLE FILE EMPLOYEE
HEADING
" Department Report Page <TABPAGENO "
PRINT LAST_NAME AS ''
FIRST_NAME AS ''
CURR_SAL AS ''
CURR_JOBCODE AS ''
BY DEPARTMENT AS ''
WHERE CURR_SAL NE 0.0
ON TABLE PCHOLD FORMAT PDF
ON DEPARTMENT SUBFOOT
" "
"Subtotal:<ST.CURR_SAL"
" "
ON DEPARTMENT SUBHEAD
"Department <+0>Last Name <+0>First Name <+0>Salary<+0>Jobcode <+0>"
ON TABLE SUBFOOT
"Grand Total:<ST.CURR_SAL"
ON TABLE SET STYLE *
INCLUDE=IBFS:/stylesheet_location/custom_stylesheet.sty,$
END
```

***custom_stylesheet*.sty**

```
TYPE=REPORT, FONT=ARIAL, BORDER=ON, SQUEEZE=ON, $
TYPE=REPORT, COLUMN=CURR_JOBCODE,SQUEEZE=.75, $
TYPE=SUBHEAD, HEADALIGN=BODY, BORDERALL=ON, $
TYPE=SUBFOOT, HEADALIGN=BODY, $
TYPE=SUBFOOT, LINE=2, ITEM=1, COLSPAN=3, JUSTIFY=RIGHT, $
TYPE=SUBFOOT, LINE=2, ITEM=2, JUSTIFY=RIGHT, $
TYPE=TABFOOTING, HEADALIGN=BODY, $
TYPE=TABFOOTING, ITEM=1, COLSPAN=3, JUSTIFY=RIGHT, $
TYPE=TABFOOTING, ITEM=2, JUSTIFY=RIGHT, $
```

The output is:



Department Report Page 1

Department	Last Name	First Name	Salary	Jobcode
MIS	SMITH	MARY	$13,200.00	B14
	JONES	DIANE	$18,480.00	B03
	MCCOY	JOHN	$18,480.00	B02
	BLACKWOOD	ROSEMARIE	$21,780.00	B04
	GREENSPAN	MARY	$9,000.00	A07
	CROSS	BARBARA	$27,062.00	A17
		Subtotal:	$108,002.00	
Department	**Last Name**	**First Name**	**Salary**	**Jobcode**
PRODUCTION	STEVENS	ALFRED	$11,000.00	A07
	SMITH	RICHARD	$9,500.00	A01
	BANNING	JOHN	$29,700.00	A17
	IRVING	JOAN	$26,862.00	A15
	ROMANS	ANTHONY	$21,120.00	B04
	MCKNIGHT	ROGER	$16,100.00	B02
		Subtotal:	$114,282.00	
		Grand Total:	$222,284.00	

Syntax:

How to Align Subheading and Subfooting Margins With the Report Body

Currently, with SQUEEZE=ON, the right margin border for subheadings and subfootings is defined based on the maximum width of all heading, footing, subheading, and subfooting lines. The length of subheading and subfooting lines is tied to the lengths of the page heading and page footing, not to the size of the data columns in the body of the report.

You can use the ALIGN-BORDERS=BODY attribute in a StyleSheet to align the subheadings and subfootings with the report body on PDF report output, instead of the other heading elements.

You can align subheading and subfooting margins with the report body either by adding the ALIGN-BORDERS=BODY attribute to the StyleSheet declaration for the REPORT component, or placing it in its own declaration without a TYPE attribute.

```
[TYPE=REPORT,] ALIGN-BORDERS={OFF|BODY}, $
```

where:

`OFF`

Does not align the right margin of subheadings and subfootings with the report body.

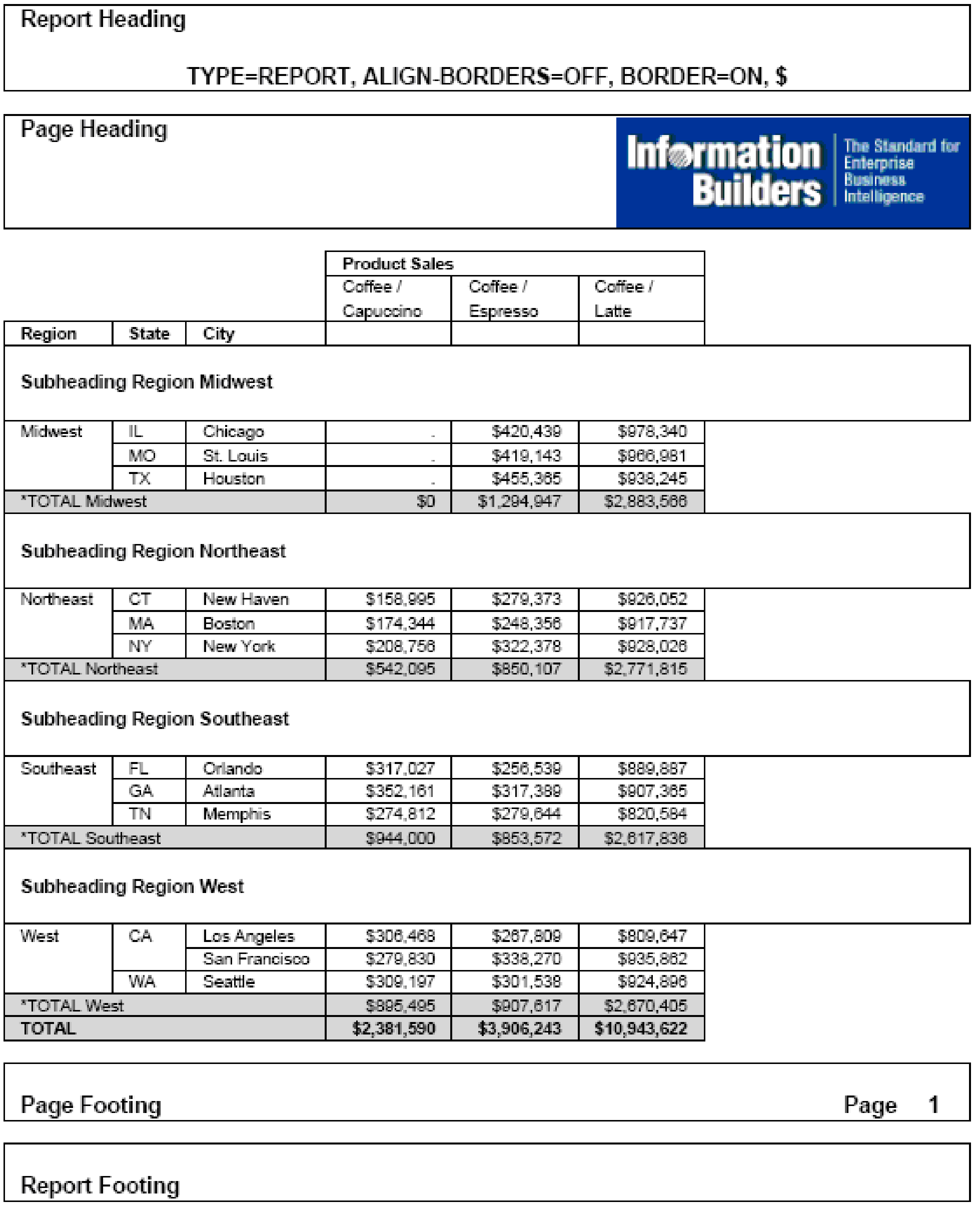
`BODY`

Specifies that the width of subheading and subfooting lines is independent of heading, footing, tabheading, and tabfooting lines, and that the right border of the report body will be aligned by either extending subheading and subfooting lines (if they are narrower than the data columns) or extending the data columns (if the data columns are narrower than the maximum width of subheadings and subfootings).

Reference: Considerations for Aligning Subheading and Subfooting Margins With the Report Body

Without the ALIGN-BORDERS=BODY attribute, the width of the subheading and subfooting lines is determined by the largest width of all of the headings and footings (report, page, subheadings, and subfootings).

The following image illustrates report output without the ALIGN-BORDERS=BODY attribute.



Report Heading

TYPE=REPORT, ALIGN-BORDERS=OFF, BORDER=ON, $

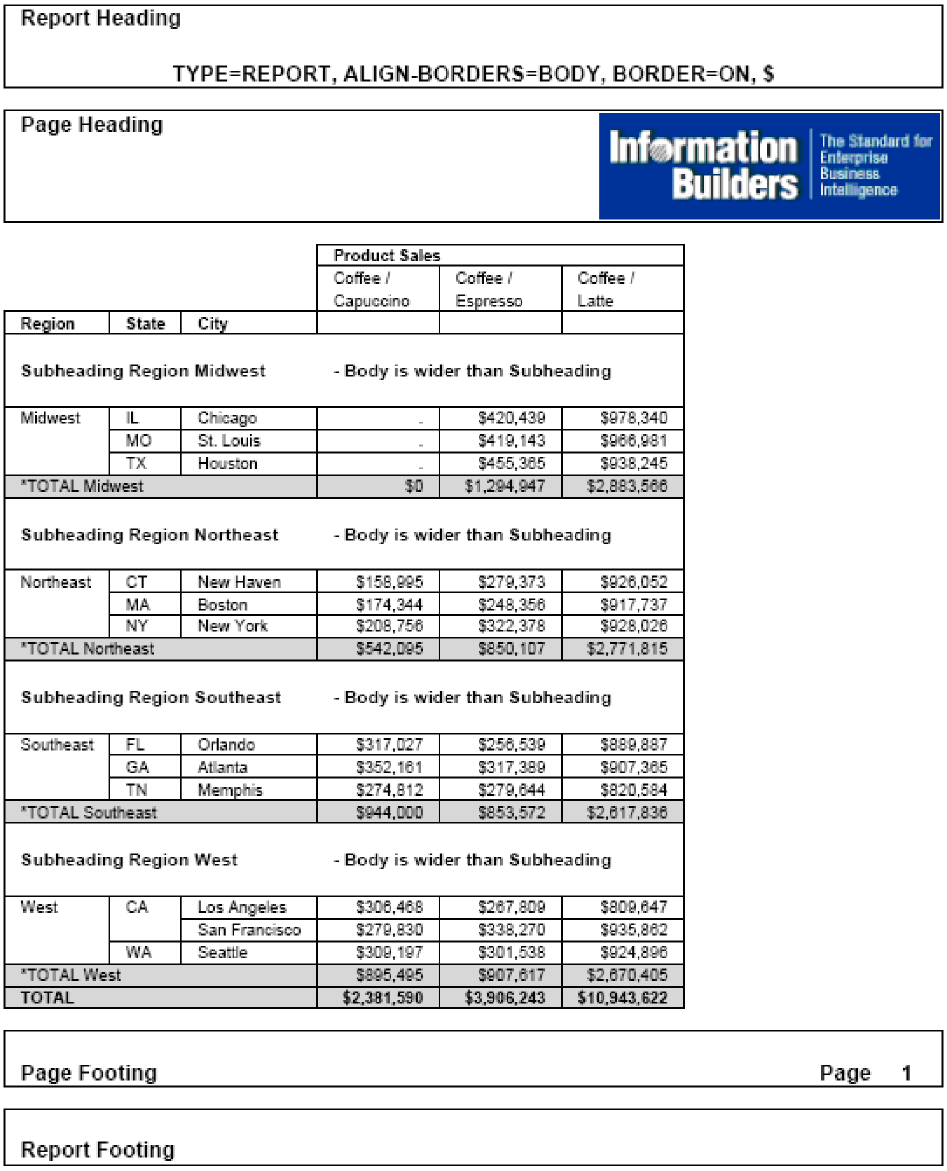
Page Heading

			Product Sales		
			Coffee / Capuccino	Coffee / Espresso	Coffee / Latte
Region	**State**	**City**			
Subheading Region Midwest					
Midwest	IL	Chicago	.	$420,439	$978,340
	MO	St. Louis	.	$419,143	$966,981
	TX	Houston	.	$455,365	$938,245
*TOTAL Midwest			$0	$1,294,947	$2,883,566
Subheading Region Northeast					
Northeast	CT	New Haven	$158,995	$279,373	$926,052
	MA	Boston	$174,344	$248,356	$917,737
	NY	New York	$208,756	$322,378	$928,026
*TOTAL Northeast			$542,095	$850,107	$2,771,815
Subheading Region Southeast					
Southeast	FL	Orlando	$317,027	$256,539	$889,887
	GA	Atlanta	$352,161	$317,389	$907,365
	TN	Memphis	$274,812	$279,644	$820,584
*TOTAL Southeast			$944,000	$853,572	$2,617,836
Subheading Region West					
West	CA	Los Angeles	$306,468	$267,809	$809,647
		San Francisco	$279,830	$338,270	$935,862
	WA	Seattle	$309,197	$301,538	$924,896
*TOTAL West			$895,495	$907,617	$2,670,405
TOTAL			**$2,381,590**	**$3,906,243**	**$10,943,622**

Page Footing Page 1

Report Footing

When the body lines are wider than the subheading and subfooting lines, the border and backcolor of the subheading and subfooting lines are expanded to match the width of the data lines, as shown on the following report output.



Report Heading

TYPE=REPORT, ALIGN-BORDERS=BODY, BORDER=ON, $

Page Heading

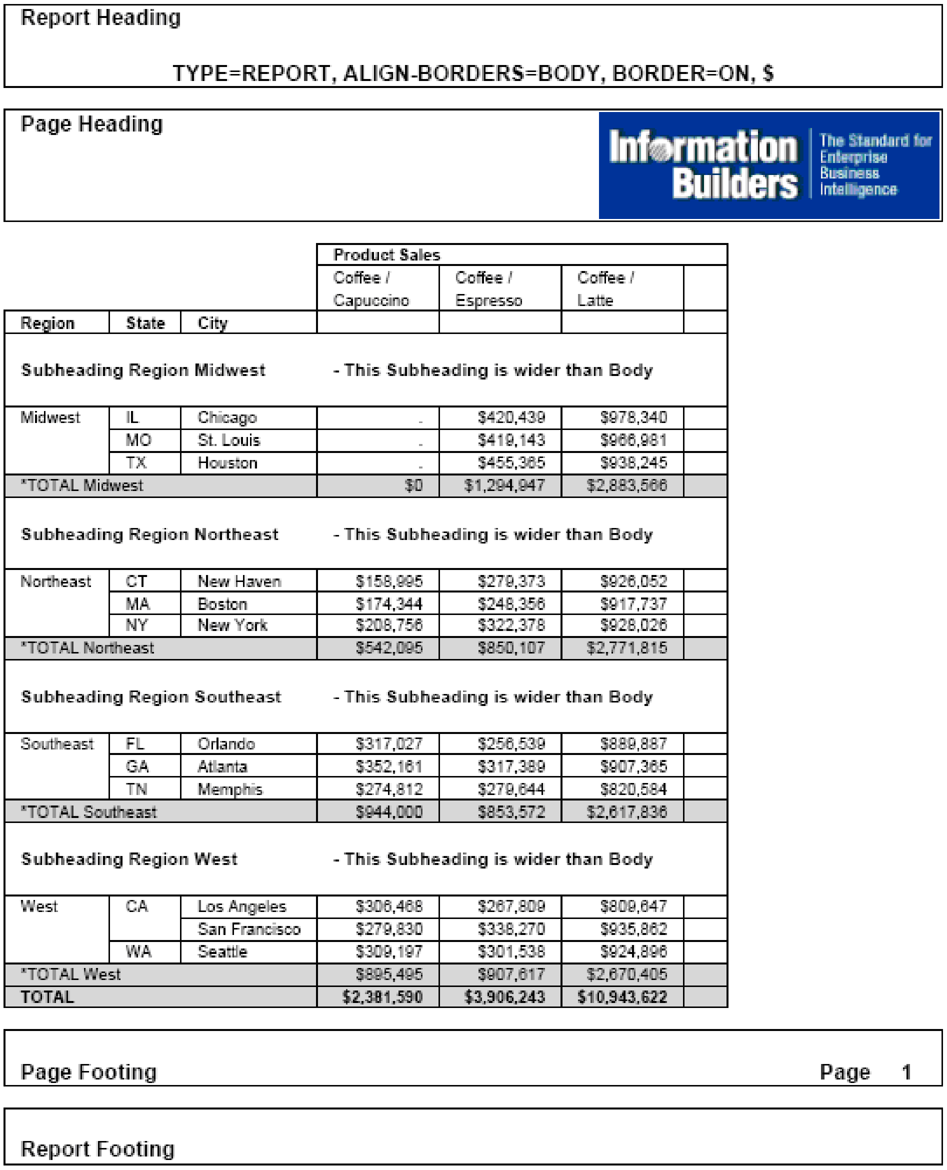
Information Builders The Standard for Enterprise Business Intelligence

			Product Sales		
			Coffee / Capuccino	Coffee / Espresso	Coffee / Latte
Region	State	City			
Subheading Region Midwest			- Body is wider than Subheading		
Midwest	IL	Chicago	.	$420,439	$978,340
	MO	St. Louis	.	$419,143	$966,981
	TX	Houston	.	$455,365	$938,245
*TOTAL Midwest			$0	$1,294,947	$2,883,566
Subheading Region Northeast			- Body is wider than Subheading		
Northeast	CT	New Haven	$158,995	$279,373	$926,052
	MA	Boston	$174,344	$248,356	$917,737
	NY	New York	$208,756	$322,378	$928,026
*TOTAL Northeast			$542,095	$850,107	$2,771,815
Subheading Region Southeast			- Body is wider than Subheading		
Southeast	FL	Orlando	$317,027	$256,539	$889,887
	GA	Atlanta	$352,161	$317,389	$907,365
	TN	Memphis	$274,812	$279,644	$820,584
*TOTAL Southeast			$944,000	$853,572	$2,617,836
Subheading Region West			- Body is wider than Subheading		
West	CA	Los Angeles	$306,468	$267,809	$809,647
		San Francisco	$279,830	$338,270	$935,862
	WA	Seattle	$309,197	$301,538	$924,896
*TOTAL West			$895,495	$907,617	$2,670,405
TOTAL			$2,381,590	$3,906,243	$10,943,622

Page Footing Page 1

Report Footing

If the subheading and subfooting lines are longer than the body lines, an additional filler cell is added to each data line to allow the defined borders and backcolor to fill the width defined by the subheading and subfooting lines, as shown on the following report output.



Report Heading

TYPE=REPORT, ALIGN-BORDERS=BODY, BORDER=ON, $

Page Heading

			Product Sales			
			Coffee / Capuccino	Coffee / Espresso	Coffee / Latte	
Region	**State**	**City**				
Subheading Region Midwest			**- This Subheading is wider than Body**			
Midwest	IL	Chicago	.	$420,439	$978,340	
	MO	St. Louis	.	$419,143	$966,981	
	TX	Houston	.	$455,365	$938,245	
*TOTAL Midwest			$0	$1,294,947	$2,883,566	
Subheading Region Northeast			**- This Subheading is wider than Body**			
Northeast	CT	New Haven	$158,995	$279,373	$926,052	
	MA	Boston	$174,344	$248,356	$917,737	
	NY	New York	$208,756	$322,378	$928,026	
*TOTAL Northeast			$542,095	$850,107	$2,771,815	
Subheading Region Southeast			**- This Subheading is wider than Body**			
Southeast	FL	Orlando	$317,027	$256,539	$889,887	
	GA	Atlanta	$352,161	$317,389	$907,365	
	TN	Memphis	$274,812	$279,644	$820,584	
*TOTAL Southeast			$944,000	$853,572	$2,617,836	
Subheading Region West			**- This Subheading is wider than Body**			
West	CA	Los Angeles	$306,468	$267,809	$809,647	
		San Francisco	$279,830	$338,270	$935,862	
	WA	Seattle	$309,197	$301,538	$924,896	
*TOTAL West			$895,495	$907,617	$2,670,405	
TOTAL			**$2,381,590**	**$3,906,243**	**$10,943,622**	

Page Footing Page 1

Report Footing

ALIGN-BORDERS=BODY has been designed to work on:

- ❏ Single panel reports (reports that do not panel horizontally).
- ❏ Paneled reports where HEADPANEL has been turned on for all of the subheadings and subfootings defined in the report.

  Setting HEADPANEL ON causes the headings and footings from the first page of a Paneled report to replicate on the subsequent panels. If HEADPANEL is not used, content can be placed in the Paneled headings by explicitly positioning items within the headings using the StyleSheet attribute POSITION. In these situations, ALIGN-BORDERS=BODY is ignored.

  Therefore, if HEADPANEL is turned on at the REPORT level and not explicitly turned off for any of the individual subheadings or subfootings, or if it is explicitly turned on for all subheadings and subfootings, ALIGN-BORDERS=BODY will align the borders of all subheadings and subfootings to the data. Otherwise, the borders will continue to exhibit the default behavior of aligning with the page headings and footings.

Example: Aligning Subheading and Subfooting Margins in a Single Panel PDF Report

The following request against the GGSALES data source has a report heading, report footing, page heading, page footing, and a subheading for each region. The margins of the subheadings and subfootings are not aligned (ALIGN-BORDERS=OFF, $).

Report Request

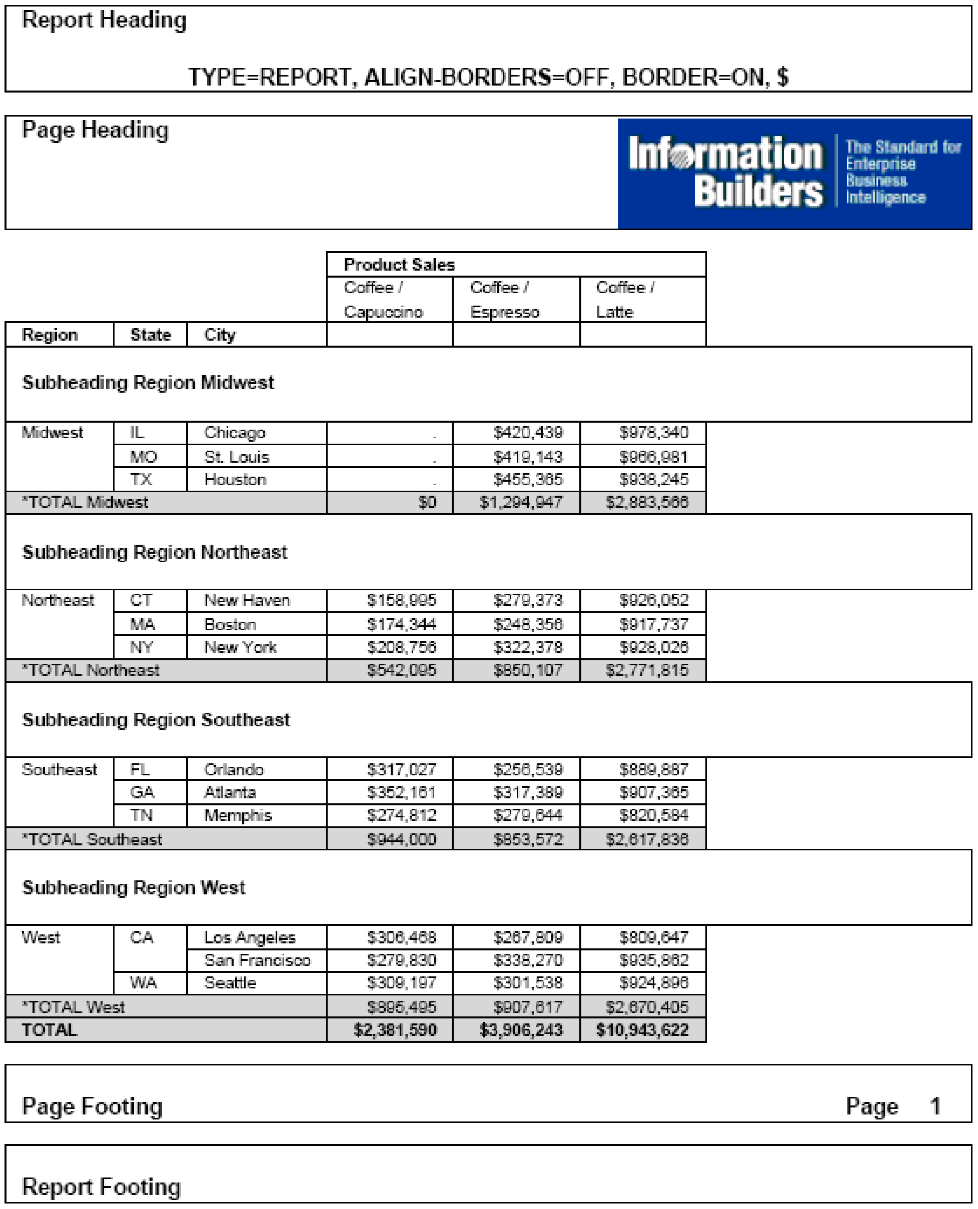
```
DEFINE FILE GGSALES
SHOWCATPROD/A30 = CATEGORY || (' / ' | PRODUCT);
END
TABLE FILE GGSALES
SUM
     DOLLARS/I8M AS ''
BY REGION
BY ST
BY CITY
ACROSS SHOWCATPROD AS 'Product Sales'
```

```
ON REGION SUBHEAD
" "
"Subheading <+0>Region <REGION<+0> "
" "
ON REGION SUBTOTAL AS '*TOTAL'
ON TABLE SUBHEAD
"Report Heading"
" "
"TYPE=REPORT, ALIGN-BORDERS=OFF, BORDER=ON, $"
HEADING
"Page Heading "
" "
" "
" "
FOOTING
" "
"Page Footing<+0>Page <TABPAGENO "
ON TABLE SUBFOOT
" "
"Report Footing"
WHERE CATEGORY EQ 'Coffee';
ON TABLE PCHOLD FORMAT PDF
ON TABLE SET STYLE *
INCLUDE=IBFS:/stylesheet_location/custom_stylesheet.sty,$
END
```

***custom_stylesheet*.sty**

```
TYPE=REPORT, FONT='ARIAL', SIZE=9, LEFTMARGIN=.75, RIGHTMARGIN=.5,
   TOPMARGIN=.1, BOTTOMMARGIN=.1, ALIGN-BORDERS=OFF, BORDER=ON,
   SQUEEZE=ON, $
TYPE=TITLE, STYLE=BOLD, $
TYPE=TABHEADING, SIZE=12, STYLE=BOLD, $
TYPE=TABHEADING, LINE=3, JUSTIFY=CENTER, $
TYPE=TABFOOTING, SIZE=12, STYLE=BOLD, $
TYPE=HEADING, SIZE=12, STYLE=BOLD, $
TYPE=HEADING, IMAGE=smplogo1.gif, POSITION=(+4.6000000 +0.03000000),
   JUSTIFY=RIGHT, $
TYPE=FOOTING, SIZE=12, STYLE=BOLD, $
TYPE=FOOTING, LINE=2, ITEM=2, OBJECT=TEXT, POSITION=6.3, SIZE=12,
   STYLE=BOLD, $
TYPE=SUBHEAD, SIZE=10, STYLE=BOLD, $
TYPE=SUBHEAD, LINE=2, ITEM=3, OBJECT=TEXT, POSITION=2.5, $
TYPE=SUBFOOT, SIZE=10, STYLE=BOLD, $
TYPE=SUBTOTAL, BACKCOLOR=RGB(210 210 210), $
TYPE=ACROSSVALUE, SIZE=9, WRAP=ON, $
TYPE=ACROSSTITLE, STYLE=BOLD, $
TYPE=GRANDTOTAL, BACKCOLOR=RGB(210 210 210), STYLE=BOLD, $
```

The output shows that the subheading margins align with the heading, not with the report body.



Report Heading

TYPE=REPORT, ALIGN-BORDERS=OFF, BORDER=ON, $

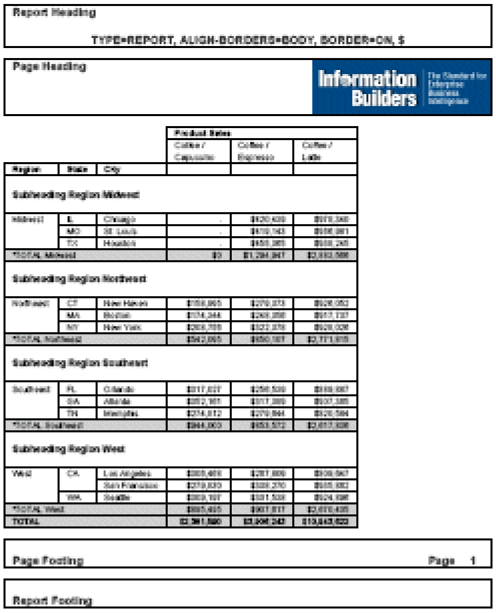
Page Heading

			Product Sales		
			Coffee / Capuccino	Coffee / Espresso	Coffee / Latte
Region	**State**	**City**			
Subheading Region Midwest					
Midwest	IL	Chicago	.	$420,439	$978,340
	MO	St. Louis	.	$419,143	$966,981
	TX	Houston	.	$455,365	$938,245
*TOTAL Midwest			$0	$1,294,947	$2,883,566
Subheading Region Northeast					
Northeast	CT	New Haven	$158,995	$279,373	$926,052
	MA	Boston	$174,344	$248,356	$917,737
	NY	New York	$208,756	$322,378	$928,026
*TOTAL Northeast			$542,095	$850,107	$2,771,815
Subheading Region Southeast					
Southeast	FL	Orlando	$317,027	$256,539	$889,887
	GA	Atlanta	$352,161	$317,389	$907,365
	TN	Memphis	$274,812	$279,644	$820,584
*TOTAL Southeast			$944,000	$853,572	$2,617,836
Subheading Region West					
West	CA	Los Angeles	$306,468	$267,809	$809,647
		San Francisco	$279,830	$338,270	$935,862
	WA	Seattle	$309,197	$301,538	$924,896
*TOTAL West			$895,495	$907,617	$2,670,405
TOTAL			**$2,381,590**	**$3,906,243**	**$10,943,622**

Page Footing Page 1

Report Footing

Now change the ALIGN-BORDERS attribute to ALIGN-BORDERS=BODY and rerun the request. The subheadings now align with the report body, as shown in the following image.



Report Heading

TYPE=REPORT, ALIGN-BORDERS=BODY, BORDER=ON, $

Page Heading

			Product Sales		
			Coffee / Capuccino	Coffee / Espresso	Coffee / Latte
Region	State	City			
Subheading Region Midwest					
Midwest	IL	Chicago	.	$420,439	$878,340
	MO	St. Louis	.	$419,143	$856,981
	TX	Houston	.	$455,365	$858,245
*TOTAL Midwest			$0	$1,294,947	$2,593,566
Subheading Region Northeast					
Northeast	CT	New Haven	$158,855	$279,373	$926,052
	MA	Boston	$174,344	$268,258	$917,737
	NY	New York	$208,799	$322,379	$928,026
*TOTAL Northeast			$542,095	$850,107	$2,771,815
Subheading Region Southeast					
Southeast	FL	Orlando	$317,027	$256,539	$889,887
	GA	Atlanta	$352,161	$317,389	$907,385
	TN	Memphis	$274,812	$279,644	$820,584
*TOTAL Southeast			$944,000	$853,572	$2,617,836
Subheading Region West					
West	CA	Los Angeles	$305,469	$297,800	$809,567
		San Francisco	$279,830	$308,270	$805,082
	WA	Seattle	$303,107	$301,532	$824,296
*TOTAL West			$885,403	$907,611	$2,670,435
TOTAL			$2,381,590	$3,906,243	$10,943,622

Page Footing Page 1

Report Footing

Example: Aligning Subheading and Subfooting Margins in a Multi-Panel Report

The following example uses HEADPANEL=ON for all headings and footings. It also uses the ALIGN-BORDERS=BODY attribute.

Report Request

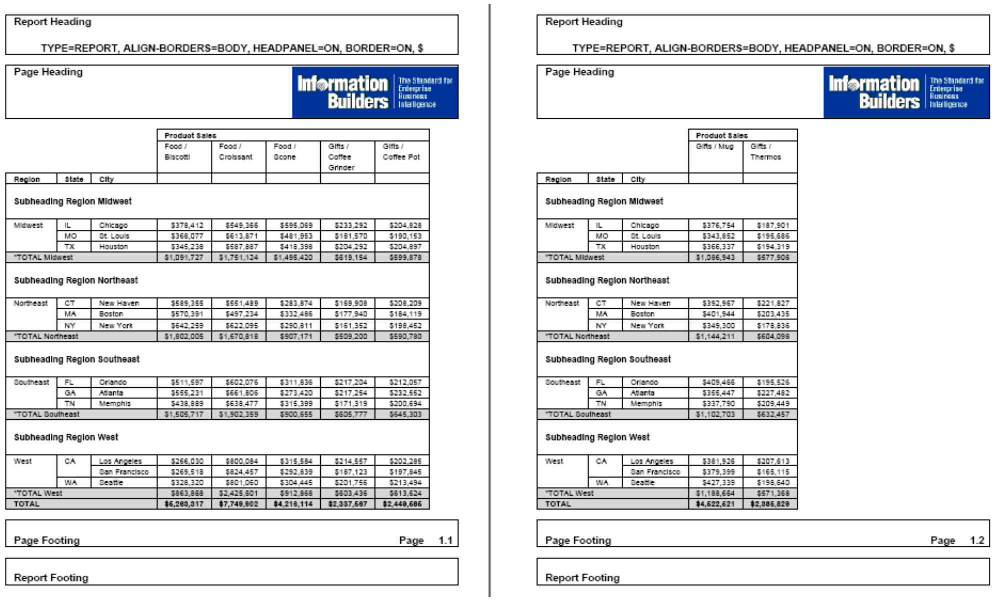
```
SET BYPANEL=ON
DEFINE FILE GGSALES
SHOWCATPROD/A30 = CATEGORY || (' / ' | PRODUCT);
END
TABLE FILE GGSALES
SUM
     DOLLARS/I8M AS ''
BY REGION
BY ST
BY CITY
ACROSS SHOWCATPROD AS 'Product Sales'
```

```
ON REGION SUBHEAD
" "
"Subheading <+0>Region <REGION<+0> "
" "
ON REGION SUBTOTAL AS '*TOTAL'
ON TABLE SUBHEAD
"Report Heading"
" "
"TYPE=REPORT, ALIGN-BORDERS=BODY, HEADPANEL=ON, BORDER=ON, $"
HEADING
"Page Heading "
" "
" "
" "
FOOTING
" "
"Page Footing<+0>Page <TABPAGENO "
ON TABLE SUBFOOT
" "
"Report Footing"
WHERE CATEGORY NE 'Coffee';
ON TABLE PCHOLD FORMAT PDF
ON TABLE SET STYLE *
INCLUDE=IBFS:/stylesheet_location/custom_stylesheet.sty,$
END
```

***custom_stylesheet*.sty**

```
TYPE=REPORT, UNITS=IN, SQUEEZE=ON, ORIENTATION=PORTRAIT, $
TYPE=REPORT, FONT='ARIAL', SIZE=9, LEFTMARGIN=.75, RIGHTMARGIN=.5,
   TOPMARGIN=.1, BOTTOMMARGIN=.1, HEADPANEL=ON, ALIGN-BORDERS=BODY,
BORDER=ON, $
TYPE=TITLE, STYLE=BOLD, $
TYPE=TABHEADING, SIZE=12, STYLE=BOLD, $
TYPE=TABHEADING, LINE=3, JUSTIFY=CENTER, $
TYPE=TABFOOTING, SIZE=12, STYLE=BOLD, $
TYPE=HEADING, SIZE=12, STYLE=BOLD, $
TYPE=HEADING, IMAGE=smplogo1.gif, POSITION=(+4.6000000 +0.03000000),
   JUSTIFY=RIGHT, $TYPE=FOOTING, SIZE=12, STYLE=BOLD, $
TYPE=FOOTING, LINE=2, ITEM=2, OBJECT=TEXT, POSITION=6.3,
   SIZE=12, STYLE=BOLD, $
TYPE=SUBHEAD, SIZE=10, STYLE=BOLD, $
TYPE=SUBHEAD, LINE=2, ITEM=3, OBJECT=TEXT, POSITION=2.5, $
TYPE=SUBFOOT, SIZE=10, STYLE=BOLD, $
TYPE=SUBTOTAL, BACKCOLOR=RGB(210 210 210), $
TYPE=ACROSSVALUE, SIZE=9, WRAP=ON, $
TYPE=ACROSSTITLE, STYLE=BOLD, $
TYPE=GRANDTOTAL, BACKCOLOR=RGB(210 210 210), STYLE=BOLD, $
```

The output shows that the subheadings are aligned with the data on each panel.



Report Heading

TYPE=REPORT, ALIGN-BORDERS=BODY, HEADPANEL=ON, BORDER=ON, $

Page Heading

			Product Sales				
			Food / Biscotti	Food / Croissant	Food / Scone	Gifts / Coffee Grinder	Gifts / Coffee Pot
Region	State	City					
Subheading Region Midwest							
Midwest	IL	Chicago	$378,412	$549,366	$595,069	$233,292	$204,828
	MO	St. Louis	$368,077	$613,871	$481,953	$181,570	$190,153
	TX	Houston	$345,238	$587,887	$418,398	$204,292	$204,897
*TOTAL Midwest			$1,091,727	$1,751,124	$1,495,420	$619,154	$599,878
Subheading Region Northeast							
Northeast	CT	New Haven	$589,355	$551,489	$283,874	$169,908	$208,209
	MA	Boston	$570,391	$497,234	$332,486	$177,940	$184,119
	NY	New York	$642,259	$622,095	$290,811	$161,352	$198,452
*TOTAL Northeast			$1,802,005	$1,670,818	$907,171	$509,200	$590,780
Subheading Region Southeast							
Southeast	FL	Orlando	$511,597	$602,076	$311,836	$217,204	$212,057
	GA	Atlanta	$555,231	$661,806	$273,420	$217,254	$232,552
	TN	Memphis	$438,889	$638,477	$315,399	$171,319	$200,694
*TOTAL Southeast			$1,505,717	$1,902,359	$900,655	$605,777	$645,303
Subheading Region West							
West	CA	Los Angeles	$266,030	$800,084	$315,584	$214,557	$202,285
		San Francisco	$269,518	$824,457	$292,839	$187,123	$197,845
	WA	Seattle	$328,320	$801,060	$304,445	$201,756	$213,494
*TOTAL West			$863,868	$2,425,601	$912,868	$603,436	$613,624
TOTAL			$6,263,317	$7,749,902	$4,216,114	$2,337,567	$2,449,686

Page Footing Page 1.1

Report Footing

Report Heading

TYPE=REPORT, ALIGN-BORDERS=BODY, HEADPANEL=ON, BORDER=ON, $

Page Heading

			Product Sales	
			Gifts / Mug	Gifts / Thermos
Region	State	City		
Subheading Region Midwest				
Midwest	IL	Chicago	$376,754	$187,901
	MO	St. Louis	$343,852	$195,686
	TX	Houston	$366,337	$194,319
*TOTAL Midwest			$1,086,943	$577,906
Subheading Region Northeast				
Northeast	CT	New Haven	$392,967	$221,827
	MA	Boston	$401,944	$203,435
	NY	New York	$349,300	$178,836
*TOTAL Northeast			$1,144,211	$604,098
Subheading Region Southeast				
Southeast	FL	Orlando	$409,466	$195,526
	GA	Atlanta	$355,447	$227,482
	TN	Memphis	$337,790	$209,449
*TOTAL Southeast			$1,102,703	$632,457
Subheading Region West				
West	CA	Los Angeles	$381,926	$207,613
		San Francisco	$379,399	$165,115
	WA	Seattle	$427,339	$198,640
*TOTAL West			$1,188,664	$571,368
TOTAL			$4,622,621	$2,385,829

Page Footing Page 1.2

Report Footing

Syntax: How to Add and Adjust Grid Lines (PDF)

The following syntax applies to a PDF report.

```
TYPE=type, {HGRID|VGRID}={ON|OFF|HEAVY}, $
```

where:

type

Identifies the report component to which grid lines are applied. For valid values, see *Identifying a Report Component in a Web Query StyleSheet* on page 57.

HGRID

Specifies horizontal grid lines.

VGRID

Specifies vertical grid lines.

ON

Applies light grid lines.

OFF

Suppresses grid lines. OFF is the default value.

`HEAVY`

Applies heavy grid lines.

Example: Applying Grid Lines to Report Data (PDF)

The following syntax applies to a PDF report. This request applies light, horizontal grid lines to report data.

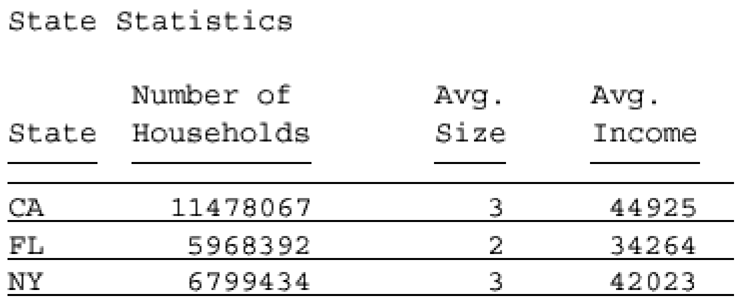
Report Request

```
TABLE FILE GGDEMOG
HEADING
"State Statistics"
" "
SUM HH AS 'Number of,Households' AVGHHSZ98 AS 'Avg.,Size'
MEDHHI98 AS 'Avg.,Income'
BY ST
WHERE ST EQ 'CA' OR 'FL' OR 'NY'
ON TABLE PCHOLD FORMAT PDF
ON TABLE SET STYLE *
INCLUDE=IBFS:/stylesheet_location/custom_stylesheet.sty,$
END
```

***custom_stylesheet*.sty**

```
TYPE=DATA, HGRID=ON, $
```

In the output, the lines make it easier to distinguish the data by state:



```
State Statistics

       Number of          Avg.     Avg.
State  Households         Size     Income

CA       11478067            3      44925
FL        5968392            2      34264
NY        6799434            3      42023
```

Defining Borders Around Boxes With PPTX and PDF Formats

In PPTX and PDF formats, the backcolor of a box may be defined independently of the border color.

Borders Without Backcolor

When the border color is defined and there is no backcolor, only the border or outline of the box is displayed in the border color specified, as shown in the example below.

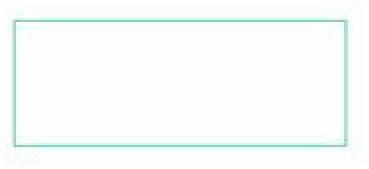
Report Request

```
TABLE FILE GGSALES
BY REGION NOPRINT
ON TABLE PCHOLD FORMAT PPTX
N TABLE SET STYLE *
INCLUDE=IBFS:/stylesheet_location/custom_stylesheet.sty,$
END
```

***custom_stylesheet*.sty**

```
TYPE=REPORT, OBJECT=BOX, POSITION=(1 1), DIMENSION=(2 1),
   BORDER-COLOR=GREEN, $
```

The output is:



Backcolor Without Borders

In PPTX format, when a backcolor is defined and there is no border (BORDER-STYLE=NONE), the box retains the color defined for the backcolor.

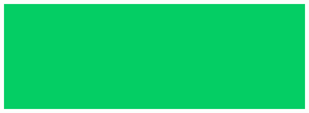
Report Request

```
TABLE FILE GGSALES
BY REGION NOPRINT
ON TABLE PCHOLD FORMAT PPTX
ON TABLE SET STYLE *
INCLUDE=IBFS:/stylesheet_location/custom_stylesheet.sty,$
END
```

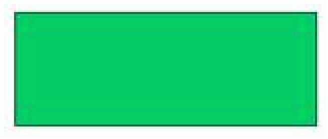
***custom_stylesheet*.sty**

```
TYPE=REPORT, OBJECT=BOX, POSITION=(1 1), DIMENSION=(2 1), BACKCOLOR=GREEN,
   BORDER-STYLE=NONE, $
```

The output is:



In PDF format, a gray outline appears around the backcolor, as shown in the following image.



Border Styles Supported

Border styles, except 3D border styles such as ridged, groove, inset, and outset, are supported in PPTX and PDF formats.

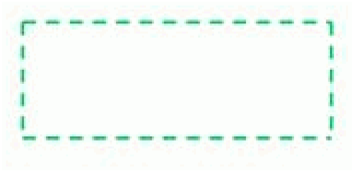
Report Request

```
TABLE FILE GGSALES
BY REGION NOPRINT
ON TABLE PCHOLD FORMAT PPTX
ON TABLE SET STYLE *
INCLUDE=IBFS:/stylesheet_location/custom_stylesheet.sty,$
END
```

***custom_stylesheet*.sty**

```
TYPE=REPORT, OBJECT=BOX, POSITION=(1 1), DIMENSION=(2 1), BORDER-
COLOR=GREEN,
   BORDER-STYLE=DASHED, $
```

The output is:



Note: When OBJECT=BOX and BORDER-STYLE=DOUBLE is used with FORMAT PPTX in StyleSheet syntax, a solid border, instead of a double border, is generated.

Displaying Superscripts On Data, Heading, and Footing Lines

Superscript characters are supported as a text style in text objects using HTML markup tags. The superscript markup tag is now supported in data columns, headings, and footings in HTML and PDF output formats. Superscript values can be defined within the data, added to virtual fields, or added to text strings displayed in headings and footings.

In order to activate the translation of the HTML markup tags, set MARKUP=ON in the StyleSheet for any report component that will display superscripts. Without this attribute, the markup tags will be treated as text, not tags.

Syntax: How to Display Superscripts on Report Data, Heading, and Footing Lines

If the tags are not within the data itself, create a field that contains the text to be used as a superscript. Also, turn markup tags on for the components that will display superscripts:

- ❑ In a DEFINE or COMPUTE command, define a field that contains the text to be displayed as a superscript.

  For a DEFINE FILE command, the syntax is:

  ```
  DEFINE FILE ...
  field/An = <sup>text</sup>;
  END
  ```

  For a COMPUTE command or a DEFINE in a Master File, the syntax is:

  ```
  {COMPUTE|DEFINE} field/An = <sup>text</sup>;
  ```

  where:

  n

  Is the length of the string defining the superscript, including the text to be used as the superscript and the opening and closing markup tags (<sup> and </sup>).

  text

  Is the text to be used as the superscript.

- ❑ In the StyleSheet, set MARKUP=ON for any report component that will display superscripts:

  ```
  TYPE=component, MARKUP=ON ... ,$
  ```

  where:

  `component`

  Is one of the following report components: DATA, HEADING, FOOTING, SUBHEAD, SUBFOOT, TABHEADING, TABFOOTING.

Example: Displaying Superscripts in Data and Footing Lines in PDF Output

The following request against the GGSALES data source defines two fields that will display as superscripts. SUP1 and SUP2 consist of the numbers 1 and 2, respectively. SUPCOPY consists of a copyright symbol. Note that the difference is the syntax as defined for a text value as opposed to a HEX value.

The COMPUTE command compares sales dollars to budgeted dollars. If the value calculated is less than a minimum defined, the superscript SUP1 is concatenated after the category name. If the value is greater, SUP2 is concatenated.

The superscript SUPCOPY is used to display the copyright symbol in the footing of the report.

The footing concatenates the superscript fields in front of their explanations.

In the StyleSheet, every component that will display a superscript has the attribute MARKUP=ON.

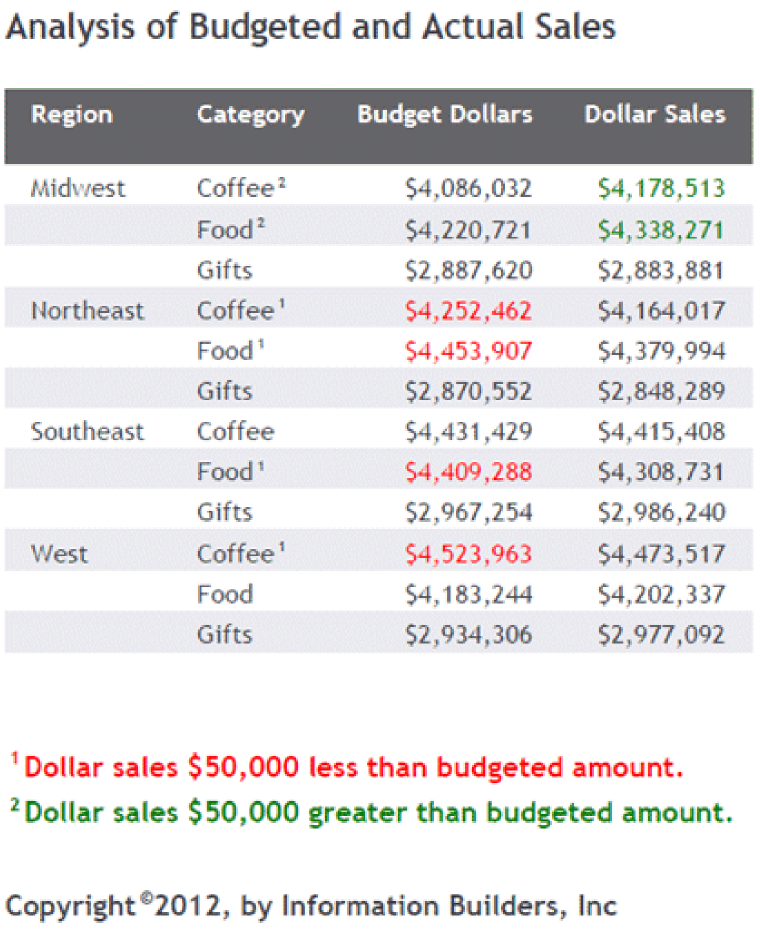
Report Request

```
DEFINE FILE GGSALES
SUP1/A12= '<SUP>1</SUP>';
SUP2/A15= '<SUP>2</SUP>';
SUPCOPY/A20= '<SUP>'||HEXBYT(169,'A2')||'</SUP>';
END
TABLE FILE GGSALES
SUM
COMPUTE PROFIT/D12CM=DOLLARS-BUDDOLLARS; NOPRINT
COMPUTE SHOWCAT/A100=IF PROFIT LE -50000 THEN CATEGORY || SUP1
       ELSE IF PROFIT GT 50000 THEN CATEGORY || SUP2
           ELSE CATEGORY; AS Category
BUDDOLLARS/D12CM
DOLLARS/D12CM
BY REGION
BY CATEGORY NOPRINT
HEADING
"Analysis of Budgeted and Actual Sales"
FOOTING
""
"<SUP1 Dollar sales $50,000 less than budgeted amount."
"<SUP2 Dollar sales $50,000 greater than budgeted amount."
""
"Copyright<SUPCOPY 2012, by Information Builders, Inc "
ON TABLE PCHOLD FORMAT PDF
ON TABLE SET STYLE *
INCLUDE=IBFS:/stylesheet_location/custom_stylesheet.sty,$
END
```

***custom_stylesheet*.sty**

```
TYPE=DATA,MARKUP=ON, $
TYPE=DATA,COLUMN=N5, COLOR=RED, WHEN=PROFIT LT -50000, $
TYPE=DATA,COLUMN=N6, COLOR=GREEN, WHEN=PROFIT GT 50000, $
TYPE=HEADING, JUSTIFY=LEFT, $
TYPE=FOOTING, MARKUP=ON, JUSTIFY=LEFT, $
TYPE=FOOTING, LINE=2,JUSTIFY=LEFT, COLOR=RED, $
TYPE=FOOTING, LINE=3,JUSTIFY=LEFT, COLOR=GREEN, $
```

The output is:



Analysis of Budgeted and Actual Sales

Region	Category	Budget Dollars	Dollar Sales
Midwest	Coffee[2]	$4,086,032	$4,178,513
	Food[2]	$4,220,721	$4,338,271
	Gifts	$2,887,620	$2,883,881
Northeast	Coffee[1]	$4,252,462	$4,164,017
	Food[1]	$4,453,907	$4,379,994
	Gifts	$2,870,552	$2,848,289
Southeast	Coffee	$4,431,429	$4,415,408
	Food[1]	$4,409,288	$4,308,731
	Gifts	$2,967,254	$2,986,240
West	Coffee[1]	$4,523,963	$4,473,517
	Food	$4,183,244	$4,202,337
	Gifts	$2,934,306	$2,977,092

[1] Dollar sales $50,000 less than budgeted amount.

[2] Dollar sales $50,000 greater than budgeted amount.

Copyright ©2012, by Information Builders, Inc

Adding Underlines and Skipped Lines

You can make a detailed tabular report easier to read by separating sections with blank lines or underlines.

Reference: Section Separation Features

Feature	Description	Applies to
`SKIP-LINE*`	Adds a blank line.	HTML (requires GRID=OFF) DHTML PDF XLSX EXL2K
`TYPE=SKIPLINE`	Formats a blank line.	DHTML PDF
`UNDER-LINE*`	Underlines a sort group.	HTML (requires GRID=OFF) DHTML PDF
`TYPE=UNDERLINE`	Formats an underline.	DHTML PDF
`STYLE={+|-}UNDERLINE*`	Adds an underline to a report component, or removes an underline from a report component other than a column title.	HTML DHTML PDF XLSX EXL2K

`STYLE={+|-}`
`EXTUNDERLINE*`
	Extends the underline to or removes the underline from the entire report column in a styled report.	DHTML
		PDF
		PPT
		PPTX

Feature	Description	Applies to
`BAR AS '{-|=}'*`	Selects a single or double underline in an FML report. For HTML, selects a light or heavy underline in an FML report.	HTML DHTML PDF XLSX EXL2K

* Not supported with border.

Syntax:

How to Format a Blank Line

```
TYPE=SKIPLINE, attribute=value, $
```

where:

attribute

Is a valid StyleSheet attribute.

value

Is the value of the attribute.

Note: This option is supported for PDF and HTML reports (when used in conjunction with internal cascading style sheets).

Example:

Adding Color to Blank Lines

In this request, blank lines are formatted to display as silver in the output.

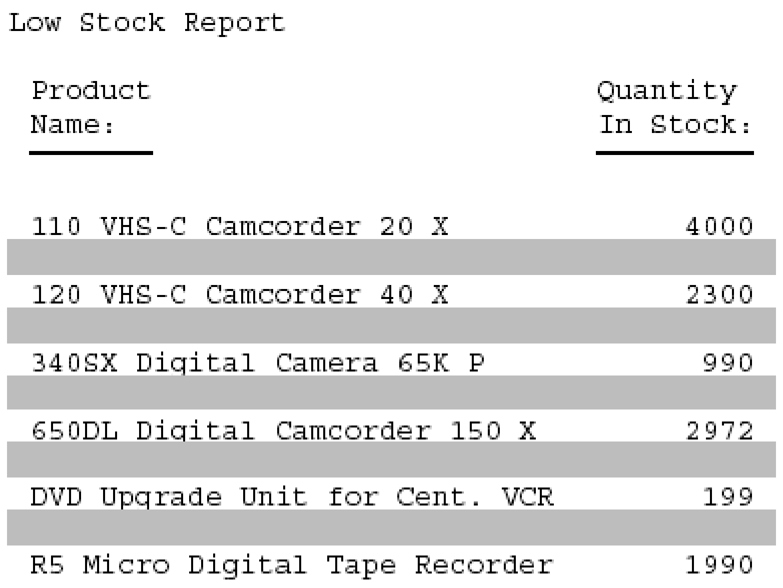
Report Request

```
TABLE FILE CENTINV
HEADING
"Low Stock Report"
" "
SUM QTY_IN_STOCK
WHERE QTY_IN_STOCK LT 5000
BY PRODNAME
ON PRODNAME SKIP-LINE
ON TABLE PCHOLD FORMAT PDF
ON TABLE SET STYLE *
INCLUDE=IBFS:/stylesheet_location/custom_stylesheet.sty,$
END
```

***custom_stylesheet*.sty**

```
TYPE=SKIPLINE, BACKCOLOR=SILVER, $
```

The report is:



```
Low Stock Report

 Product                                   Quantity
 Name:                                     In Stock:


 110 VHS-C Camcorder 20 X                       4000

 120 VHS-C Camcorder 40 X                       2300

 340SX Digital Camera 65K P                      990

 650DL Digital Camcorder 150 X                  2972

 DVD Upgrade Unit for Cent. VCR                  199

 R5 Micro Digital Tape Recorder                 1990
```

Syntax: **How to Format an Underline**

```
TYPE=UNDERLINE ... COLOR={color|RGB} (r g b), $
```

where:

`UNDERLINE`

Denotes underlines generated by ON *fieldname* UNDER-LINE.

`COLOR`

Specifies the color of the underline. If the display or output device does not support colors, it substitutes shades of gray. The default value is black.

`color`

Is one of the supported color values. For a list of supported values, see *Color Values in a Report* on page 284.

`RGB`

Specifies the text color using a mixture of red, green, and blue.

`(r g b)`

Is the desired intensity of red, green, and blue, respectively. The values are on a scale of 0 to 255, where 0 is the least intense and 255 is the most intense.

Note that using the three-color components in equal intensities results in shades of gray.

Note: This option is supported for PDF and HTML reports (when used in conjunction with internal cascading style sheets).

Example: Formatting a Sort Group Underline

The following example uses UNDERLINE to change the default color of an underline from black to red.

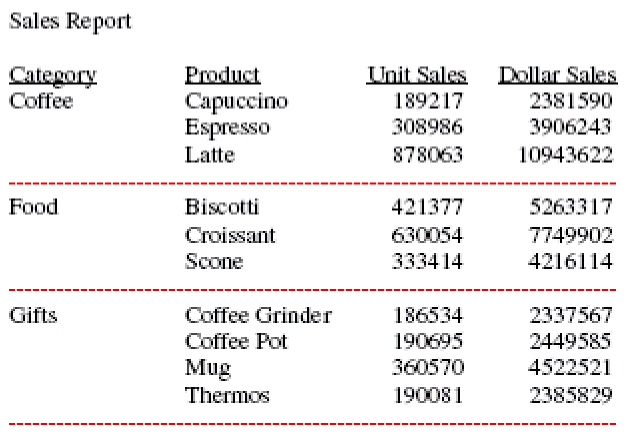
Report Request

```
TABLE FILE GGSALES
SUM UNITS DOLLARS
BY CATEGORY BY PRODUCT
HEADING
"Sales Report"
" "
ON CATEGORY UNDER-LINE
ON TABLE PCHOLD FORMAT PDF
ON TABLE SET STYLE *
INCLUDE=IBFS:/stylesheet_location/custom_stylesheet.sty,$
END
```

***custom_stylesheet*.sty**

```
TYPE=UNDERLINE, COLOR=RED, $
```

The result is a separation between sort group values. The output is:



```
Sales Report

Category     Product          Unit Sales   Dollar Sales
Coffee       Capuccino            189217        2381590
             Espresso             308986        3906243
             Latte                878063       10943622
-------------------------------------------------------
Food         Biscotti             421377        5263317
             Croissant            630054        7749902
             Scone                333414        4216114
-------------------------------------------------------
Gifts        Coffee Grinder       186534        2337567
             Coffee Pot           190695        2449585
             Mug                  360570        4522521
             Thermos              190081        2385829
-------------------------------------------------------
```

Syntax: How to Add or Remove a Report Component Underline

```
TYPE=type, [subtype,] STYLE=[+|-]UNDERLINE, $
```

where:

type

Is the report component. For valid values, see *Identifying a Report Component in a Web Query StyleSheet* on page 57.

subtype

Are additional attributes, such as COLUMN, ACROSS, or ITEM, needed to identify the report component. For valid values, see *Identifying a Report Component in a Web Query StyleSheet* on page 57.

+

Adds an underline to the inherited text style or specifies a combination of text styles (for example, STYLE=BOLD+UNDERLINE). This is the default value.

-

Removes an underline from an inherited text style.

Syntax: How to Remove an Underline From a Column Title

The following syntax applies to an HTML report with an internal cascading style sheet.

```
TYPE=TITLE, [COLUMN=column,] STYLE=-UNDERLINE, $
```

where:

column

Specifies a column. For valid values, see *Identifying a Report Component in a Web Query StyleSheet* on page 57.

Example: Adding Column Underlines and Removing Column Title Underlines

The following example illustrates how to add underlines to the values of the CATEGORY column and remove the default underlines from the column titles in an HTML report with an internal cascading style sheet.

Report Request

```
TABLE FILE MOVIES
PRINT TITLE DIRECTOR
BY CATEGORY
ON TABLE SET STYLE *
INCLUDE=IBFS:/stylesheet_location/custom_stylesheet.sty,$
END
```

***custom_stylesheet*.sty**

```
TYPE=REPORT, GRID=OFF, $
TYPE=REPORT, COLUMN=CATEGORY, STYLE=UNDERLINE, $
TYPE=TITLE, STYLE=-UNDERLINE, $
```

The partial output is:

```
CATEGORY  TITLE                                       DIRECTOR
ACTION    JAWS                                        SPIELBERG S.
          ROBOCOP                                     VERHOVEN P.
          TOTAL RECALL                                VERHOVEN P.
          TOP GUN                                     SCOTT T.
          RAMBO III                                   MCDONALD P.
CHILDREN  SMURFS, THE
          SHAGGY DOG, THE                             BARTON C.
          SCOOBY-DOO-A DOG IN THE RUFF
          ALICE IN WONDERLAND                         GEROMINI
          SESAME STREET-BEDTIME STORIES AND SONGS
          ROMPER ROOM-ASK MISS MOLLY
          SLEEPING BEAUTY                             DISNEY W.
          BAMBI                                       DISNEY W.
CLASSIC   EAST OF EDEN                                KAZAN E.
          CITIZEN KANE                                WELLES O.
          CYRANO DE BERGERAC                          GORDON M.
          MARTY                                       MANN D.
          MALTESE FALCON, THE                         HUSTON J.
          GONE WITH THE WIND                          FLEMING V.
```

Syntax: How to Extend an Underline to the Entire Report Column

By default, underlines for column titles on a report extend only from the beginning to the end of the column title text. You can extend the underline to the entire report column in styled report output using the EXTUNDERLINE option in your Web Query StyleSheet. EXTUNDERLINE is an option of the STYLE attribute for the TITLE report component. It is supported for formats DHTML, PDF, and PPTX.

```
TYPE=TITLE, [COLUMN=column,] STYLE=[+|-]EXTUNDERLINE, $
```

where:

column

Is any valid column name.

`+EXTUNDERLINE`

Adds the EXTUNDERLINE option to the inherited text style or specifies a combination of text styles, for example, STYLE=BOLD+UNDERLINE.

`-EXTUNDERLINE`

Removes the EXTUNDERLINE option from the inherited text style.

Reference: Usage Notes for the EXTUNDERLINE Attribute

- HTML format is not supported because the browser calculates the column width and renders the report.
- GRID=ON and EXTUNDERLINE are mutually exclusive since the GRID line spans the width of the column. GRID overrides any styling specified for the column title underline.

Example: Extending an Underline to the Entire Report Column

To underline entire columns, generate the output in a format that can be styled and use the EXTUNDERLINE option in the STYLE attribute for the TITLE component. For example, the following example creates DHTML output in which the column titles are in boldface and left-justified, and the underline is extended to the entire report column.

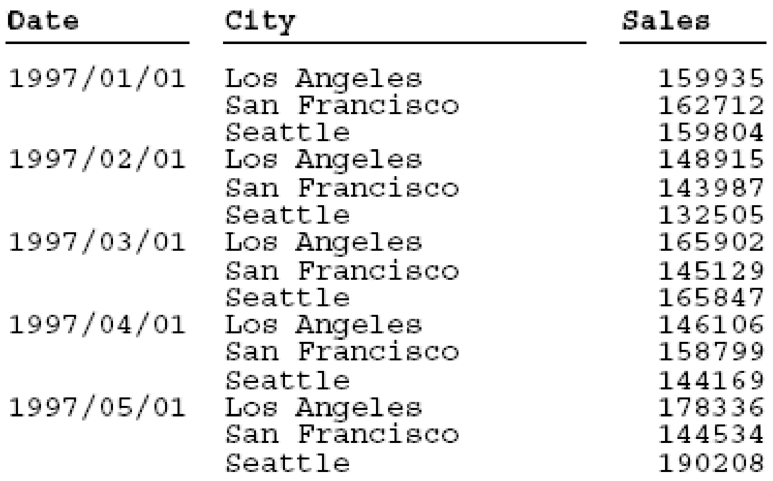
Report Request

```
DEFINE FILE GGSALES
YEAR/YY = DATE;
MONTH/M = DATE;
END
TABLE FILE GGSALES
SUM DOLLARS AS 'Sales'
BY DATE
BY CITY
WHERE YEAR EQ 1997
WHERE MONTH FROM 01 TO 05
WHERE CITY EQ 'Seattle' OR 'San Francisco' OR 'Los Angeles'
ON TABLE PCHOLD FORMAT DHTML
ON TABLE SET STYLE *
INCLUDE=IBFS:/stylesheet_location/custom_stylesheet.sty,$
END
```

***custom_stylesheet*.sty**

```
TYPE=TITLE, STYLE=BOLD+EXTUNDERLINE, JUSTIFY=LEFT, $
```

The output is:



```
Date          City                      Sales

1997/01/01    Los Angeles              159935
              San Francisco            162712
              Seattle                  159804
1997/02/01    Los Angeles              148915
              San Francisco            143987
              Seattle                  132505
1997/03/01    Los Angeles              165902
              San Francisco            145129
              Seattle                  165847
1997/04/01    Los Angeles              146106
              San Francisco            158799
              Seattle                  144169
1997/05/01    Los Angeles              178336
              San Francisco            144534
              Seattle                  190208
```

The following example makes the EXTUNDERLINE and JUSTIFY=LEFT options the default for the TITLE component, then makes the Date column title bold and removes the extended underline from that column.

Report Request

```
DEFINE FILE GGSALES
YEAR/YY = DATE;
MONTH/M = DATE;
END
TABLE FILE GGSALES
SUM DOLLARS AS 'Sales'
BY DATE
BY CITY
WHERE YEAR EQ 1997
WHERE MONTH FROM 01 TO 05
WHERE CITY EQ 'Seattle' OR 'San Francisco' OR 'Los Angeles'
ON TABLE PCHOLD FORMAT DHTML
ON TABLE SET STYLE *
INCLUDE=IBFS:/stylesheet_location/custom_stylesheet.sty,$
END
```

***custom_stylesheet*.sty**

```
TYPE=TITLE, STYLE=EXTUNDERLINE, JUSTIFY=LEFT, $
TYPE=TITLE, COLUMN=DATE, STYLE=-EXTUNDERLINE+BOLD, $
```

The output is:

```
Date        City                    Sales

1997/01/01  Los Angeles              159935
            San Francisco            162712
            Seattle                  159804
1997/02/01  Los Angeles              148915
            San Francisco            143987
            Seattle                  132505
1997/03/01  Los Angeles              165902
            San Francisco            145129
            Seattle                  165847
1997/04/01  Los Angeles              146106
            San Francisco            158799
            Seattle                  144169
1997/05/01  Los Angeles              178336
            San Francisco            144534
            Seattle                  190208
```

Adding an Image to a Report

With a StyleSheet you can add and position an image in a report. An image, such as a logo, gives corporate identity to a report, or provides visual appeal. You can add more than one image by creating multiple declarations.

You can also add an image as background to a report. A background image is tiled or repeated, covering the entire area on which the report displays. An image attached to an entire report, or an image in a heading or footing, can appear with a background image.

Images must exist in a file format your browser supports, such as GIF (Graphic Interchange Format) or JPEG (Joint Photographic Experts Group, .jpg extension).

Image support with WebFOCUS standard reporting formats

- GIF and JPG images are supported in DHTML, HTML, PDF, PPTX, XLSX, and PPT standard report formats. JPEG images are only supported with HTML standard report format. For other report formats, you can change the extension of the image name from .jpeg to .jpg, and the image will be displayed in the report output.
- PNG images are supported with DHTML, HTML, PPTX, and PDF standard report formats, while WebFOCUS generated SVG charts will display when inserted in HTML and PDF report formats.
- SVG images are supported only with HTML reports.

- Images are not supported in EXL2K standard report format.

Image support in Compound Report syntax

- GIF and JPG images are supported in DHTML, PDF, PPTX, and PPT Compound document syntax. JPEG images are not supported with any reporting format, but the images will work in these compound formats if the extension is changed from .jpeg to .jpg.
- PNG images are supported with DHTML, PPT, PPTX, and PDF Compound documents.
- SVG images are not supported with any WebFOCUS reporting format in Compound documents, while WebFOCUS generated SVG charts are supported only with PDF Compound documents.
- Images are not supported in EXL2K Compound documents.

For PDF, HTML, and DHTML output against data sources that support the Binary Large Object (BLOB) data type (Db2 and Microsoft SQL Server, using its BYTEA data type), an image can be stored in a BLOB field in the data source.

For more information on inserting images in a report, see *Customizing Reports and Dashboards With Images and Style Sheets* on page 15.

Note: For JPEG files, currently only the .jpg extension is supported. The .jpeg extension is not supported.

Reference: Image Attributes

Attribute	Description
IMAGE	Adds an image.
IMAGEALIGN	Positions an image. This applies only to HTML reports.
POSITION	Positions an image.
IMAGEBREAK	Controls generation of a line break after an image. This applies only to HTML reports without internal cascading style sheets.
SIZE	Sizes an image.

Attribute	Description
ALT	Supplies a description of an image for compliance with Section accessibility (Workforce Investment Act of 1998). ALT only applies to HTML reports. The description also displays as a pop-up description when your mouse or cursor hovers over the image in the report output.
PRESERVERATIO	ON specifies that the aspect ratio (ratio of height to width) of the image should be preserved when it is scaled to the specified SIZE. This avoids distorting the appearance of the image. The image is scaled to the largest size possible within the bounds specified by SIZE for which the aspect ratio can be maintained. Supported for images in PDF report output.
BACKIMAGE	Adds a background image.

Syntax: How to Add an Image to an HTML Report

The following syntax applies to an HTML report. For details on adding an image to a PDF or HTML report with an internal cascading style sheet, see *How to Add an Image to a PDF or HTML Report With an Internal Cascading Style Sheet* on page 204.

```
TYPE={REPORT|heading}, IMAGE={url|(column)} [,IMAGEALIGN=position]
   [,IMAGEBREAK={ON|OFF}] [,ALT='description'], $
```

where:

REPORT

Embeds an image in the body of a report. REPORT is the default value.

Note: The IMAGE=(column) option is not supported with TYPE=REPORT.

heading

Embeds an image in a heading or footing. Valid values are TABHEADING, TABFOOTING, HEADING, FOOTING, SUBHEAD, and SUBFOOT.

url

Is the URL for the image file. The image must exist in a separate file in a format that your browser supports, such as GIF or JPEG (.jpg). The file can be on your local web server, or on any server accessible from your network. For details, see *Specifying a URL* on page 196.

column

Is an alphanumeric field in a request (for example, a display field or a BY field) whose value is a URL that points to an image file. Specify a value using the COLUMN attribute described in *Identifying a Report Component in a Web Query StyleSheet* on page 57. Enclose *column* in parentheses.

This option enables you to add different images to a heading or footing, depending on the value of the field.

position

Is the position of the image.

Note: IMAGEALIGN is not supported. You can position images within a heading or footing by using the POSITION attribute to specify a position relative to the upper-left corner of the heading or footing. For more information about the POSITION attribute, see *How to Add an Image From a BLOB Field to a PDF, DHTML, or HTML Report* on page 209.

Valid values are:

TOP where the top right corner of the image aligns with heading or footing text. If the image is attached to the entire report, it appears on top of the report.

MIDDLE where the image appears in the middle of the heading or footing text. If the image is attached to the entire report, it appears in the middle of the report.

BOTTOM where the bottom right corner of the image aligns with heading or footing text. If the image is attached to the entire report, it appears at the bottom of the report.

LEFT where the image appears to the left of heading or footing text. If the image is attached to the entire report, it appears to the left of the report.

RIGHT where the image appears to the right of heading or footing text. If the image is attached to the entire report, it appears to the right of the report.

`IMAGEBREAK`

Controls generation of a line break after the image. Valid values are:

`ON` which generates a line break after the image so that an element following it (such as, report heading text) appears on the next line.

`OFF` which suppresses a line break after the image so that an element following it is on the same line. OFF is the default value.

`description`

Is a textual description of an image for compliance with Section 508 accessibility. Enclose the description in single quotation marks (').

The description also displays as a pop-up description when your mouse or cursor hovers over the image in the report output.

Reference: Specifying a URL

The following guidelines are the same for IMAGE=*url* and IMAGE=(*column*) syntax. In the latter case, they apply to a URL stored in a data source field.

Specify a URL by:

- ❏ Supplying an absolute or relative address that points to an image file, for example:

```
TYPE=TABHEADING, IMAGE=http://www.ibm.com/images/logo_wf3.gif, $
TYPE=TABHEADING, IMAGE=/webquery/ibi_html/ggdemo/gotham.gif, $
```

- ❏ Using the SET BASEURL parameter to establish a URL that is logically prefixed to all relative URLs in the request. With this feature, you can add an image by specifying just its file name in the IMAGE attribute. For example:

```
SET BASEURL=http://host:port/
.
.
.
TYPE=REPORT, IMAGE=gotham.gif, $
```

The following apply:

- ❏ A base URL must end with a slash (/).
- ❏ An absolute URL (which begins with http://) overrides a base URL.
- ❏ If the name of the image file does not contain an extension, .gif is used.

Example: Adding a GIF Image to an HTML Report Heading

The following example illustrates how to add the Gotham Grinds logo to a report heading. The logo is in a separate image file identified by a relative URL in the IMAGE attribute.

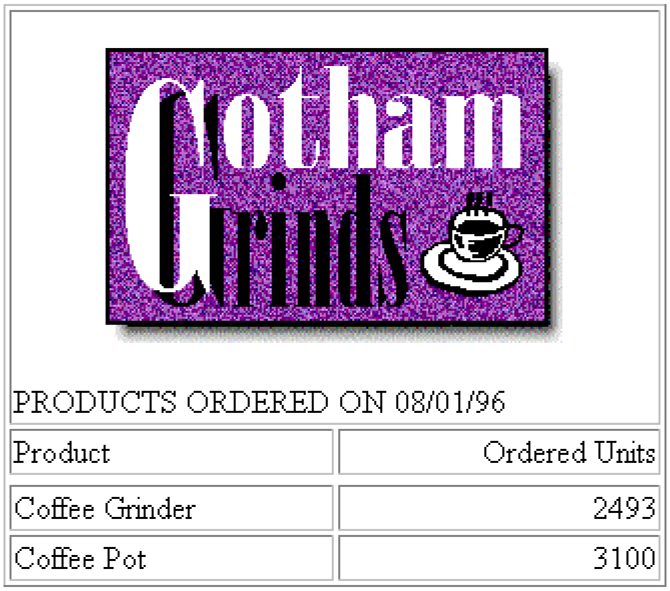
Report Request

```
TABLE FILE GGORDER
ON TABLE SUBHEAD
"PRODUCTS ORDERED ON 08/01/96"
SUM QUANTITY AS 'Ordered Units' BY PRODUCT
WHERE PRODUCT EQ 'Coffee Grinder' OR 'Coffee Pot'
WHERE ORDER_DATE EQ '08/01/96'
ON TABLE SET STYLE *
INCLUDE=IBFS:/stylesheet_location/custom_stylesheet.sty,$
END
```

***custom_stylesheet*.sty**

```
TYPE=TABHEADING,IMAGE=/webquery/ibi_html/GGDEMO/GOTHAM.GIF,
   IMAGEBREAK=ON, $
```

IMAGEBREAK, set to ON, generates a line break between the logo and the heading text.



PRODUCTS ORDERED ON 08/01/96

Product	Ordered Units
Coffee Grinder	2493
Coffee Pot	3100

Example: Creating a Report Heading With an Embedded JPEG Image

The following example illustrates how to add an embedded image to a report heading. The file is in a separate image file identified the IMAGE attribute.

Report Request

```
TABLE FILE EMPLOYEE
ON TABLE SUBHEAD
"Employee Salary Information and Courses"
" "
" "
" "
" "
" "
" "
" "
" "
" "
PRINT CURR_SAL BY COURSE_NAME
ON TABLE SET STYLE *
INCLUDE=IBFS:/stylesheet_location/custom_stylesheet.sty,$
END
```

***custom_stylesheet*.sty**

```
TYPE=REPORT, GRID=OFF, $
TYPE=TABHEADING, IMAGE=/webquery/ibi_html/IMAGES/Pencils.jpg,
   POSITION=(.5 .5), SIZE=(.5 .5), $
```

Note: The image used in this request is not distributed with Web Query.

The output is:

```
     EMPLOYEE SALARY INFORMATION AND SKILLS
```

```
COURSE  NAME                                  CURR  SAL
ADVANCED TECHNIQUES                          $18,480.00
BASIC REPORT PREP DP MGRS                    $27,062.00
BASIC REPORT PREP FOR PROG                   $13,200.00
                                             $18,480.00
BASIC REPORT PREP NON-PROG                   $21,780.00
BASIC RPT NON-DP MGRS                         $9,500.00
DECISION SUPPORT WORKSHOP                    $21,780.00
FILE DESC & MAINT NON-PROG                   $21,780.00
FILE DESCRPT & MAINT                         $11,000.00
                                             $13,200.00
                                             $18,480.00
                                             $16,100.00
FOCUS INTERNALS                              $18,480.00
HOST LANGUAGE INTERFACE                      $27,062.00
TIMESHARING WORKSHOP                         $21,780.00
WHAT'S NEW IN FOCUS                          $21,780.00
```

Example: Using a File Name in a Data Source Field in an HTML Report

The following example illustrates how to embed an image in a SUBHEAD, and use a different image for each value of the BY field on which the SUBHEAD occurs.

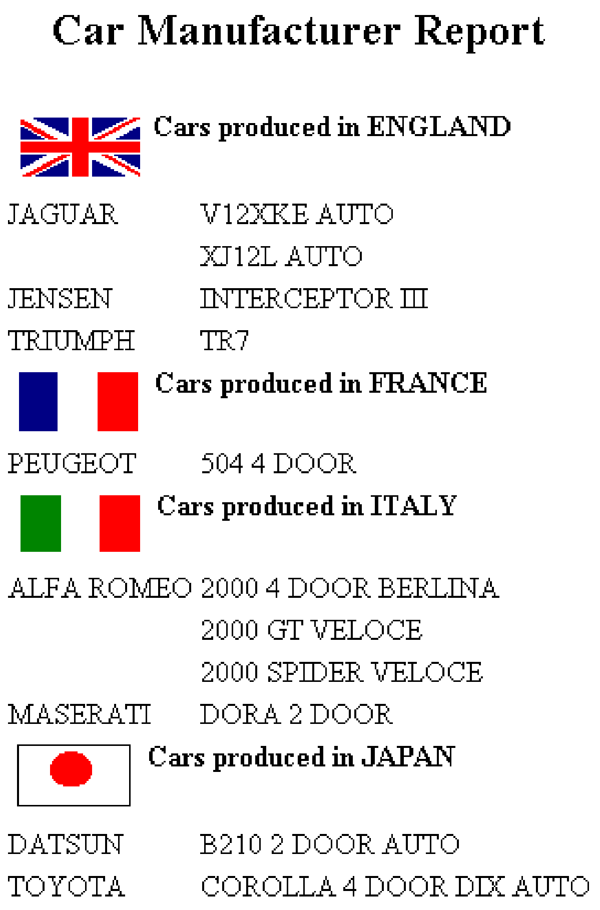
Report Request

```
DEFINE FILE CAR
FLAG/A12=
DECODE COUNTRY ( 'ENGLAND' 'uk' 'ITALY' 'italy'
   'FRANCE' 'france' 'JAPAN' 'japan' );
END

TABLE FILE CAR
PRINT FLAG NOPRINT AND MODEL AS '' BY COUNTRY NOPRINT AS '' BY CAR AS ''
WHERE COUNTRY EQ 'ENGLAND' OR 'FRANCE' OR 'ITALY' OR 'JAPAN'
ON COUNTRY SUBHEAD
"                      <+0>Cars produced in <ST.COUNTRY"
HEADING CENTER
"Car Manufacturer Report"
" "
ON TABLE SET STYLE *
INCLUDE=IBFS:/stylesheet_location/custom_stylesheet.sty,$
END
```

***custom_stylesheet*.sty**

```
TYPE=REPORT, GRID=OFF, $
TYPE=HEADING, SIZE=12, STYLE=BOLD, $
TYPE=SUBHEAD, IMAGE=(FLAG), $
TYPE=SUBHEAD, STYLE=BOLD, $
```

The output is:



```
Car Manufacturer Report

        Cars produced in ENGLAND

JAGUAR        V12XKE AUTO
              XJ12L AUTO
JENSEN        INTERCEPTOR III
TRIUMPH       TR7
        Cars produced in FRANCE

PEUGEOT       504 4 DOOR
        Cars produced in ITALY

ALFA ROMEO 2000 4 DOOR BERLINA
              2000 GT VELOCE
              2000 SPIDER VELOCE
MASERATI      DORA 2 DOOR
        Cars produced in JAPAN

DATSUN        B210 2 DOOR AUTO
TOYOTA        COROLLA 4 DOOR DIX AUTO
```

Example: Supplying an Image Description Using the ALT Attribute

The following example illustrates how to use the ALT attribute. The ALT attribute supplies a description of an image that screen readers can interpret to comply with Section 508 accessibility (Workforce Investment Act of 1998). The description also displays as a pop-up description when your mouse or cursor hovers over the image in the report output.

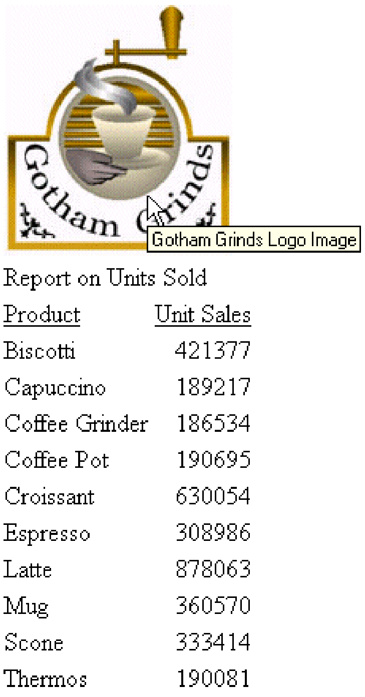
Report Request

```
TABLE FILE GGSALES
SUM UNITS BY PRODUCT
ON TABLE SUBHEAD
"Report on Units Sold"
ON TABLE SET STYLE *
INCLUDE=IBFS:/stylesheet_location/custom_stylesheet.sty,$
END
```

***custom_stylesheet*.sty**

```
TYPE=REPORT, GRID=OFF, $
TYPE=TABHEADING, IMAGE=gglogo, IMAGEBREAK=ON, POSITION=(.25 .25),
   SIZE=(.5 .5), ALT='Gotham Grinds Logo Image', $
```

The output is:



```
Report on Units Sold
Product        Unit Sales
Biscotti           421377
Capuccino          189217
Coffee Grinder     186534
Coffee Pot         190695
Croissant          630054
Espresso           308986
Latte              878063
Mug                360570
Scone              333414
Thermos            190081
```

Syntax:

How to Add a Background Image

The following syntax applies to an HTML report.

```
[TYPE=REPORT,] BACKIMAGE=url, $
```

where:

`TYPE=REPORT`

Applies the image to the entire report. Not required, as it is the default.

`url`

Is the URL of a GIF or JPEG file (.jpg). Specify a file on your local web server, or on a server accessible from your network.

The URL can be an absolute or relative address.

When specifying a GIF file, you can omit the file extension.

Example: Adding a Background Image

The following example illustrates how to add a background image to a report. The image file, CALM_BKG.GIF, resides in the relative address shown.

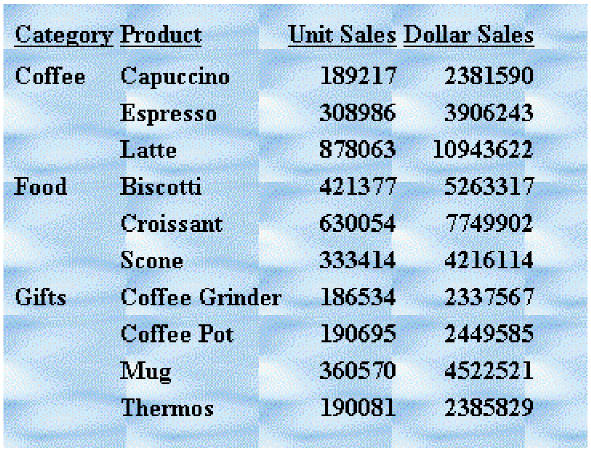
Report Request

```
TABLE FILE GGSALES
SUM UNITS DOLLARS
BY CATEGORY BY PRODUCT
ON TABLE SET STYLE *
INCLUDE=IBFS:/stylesheet_location/custom_stylesheet.sty,$
END
```

***custom_stylesheet*.sty**

```
TYPE=REPORT, STYLE=BOLD, GRID=OFF, $
TYPE=REPORT, BACKIMAGE=/webquery/ibi_html/TEMPLATE/CALM_BKG.GIF, $
```

The background is tiled across the report area.



Category	Product	Unit Sales	Dollar Sales
Coffee	Capuccino	189217	2381590
	Espresso	308986	3906243
	Latte	878063	10943622
Food	Biscotti	421377	5263317
	Croissant	630054	7749902
	Scone	333414	4216114
Gifts	Coffee Grinder	186534	2337567
	Coffee Pot	190695	2449585
	Mug	360570	4522521
	Thermos	190081	2385829

Syntax: How to Add an Image to a PDF or HTML Report With an Internal Cascading Style Sheet

The following syntax applies to a PDF or HTML report with an internal cascading style sheet. The image can be in a separate file.

A report with an Internal cascading style sheet is an HTML page with an HTML cascading style sheet (CSS) stored between the style tags within the HTML document.

```
TYPE={REPORT|heading}, IMAGE={url|file|(column)} [,BY=byfield]
   [,POSITION=([+|-]x [+|-]y )] [,SIZE=(w h)] , $
```

where:

REPORT

Embeds an image in the body of a report. The image appears in the background of the report. REPORT is the default value.

heading

Embeds an image in a heading or footing. Valid values are TABHEADING, TABFOOTING, FOOTING, HEADING, SUBHEAD, and SUBFOOT.

Provide sufficient blank space in the heading or footing so that the image does not overlap the heading or footing text. Also, you may want to place heading or footing text to the right of the image using spot markers or the POSITION attribute in the StyleSheet.

url

HTML report with an internal cascading style sheet:

Is the absolute or relative address for the image file. The image must exist in a separate file in a format that your browser supports, such as GIF or JPEG (.jpg). The file can be on your local web server, or on any server accessible from your network. For details, see *Specifying a URL* on page 196.

file

PDF report:

Is the name of the image file. It must reside on the Reporting Server in a directory named on EDAPATH or APPPATH. If the file is not on the search path, supply the full path name.

When specifying a GIF file, you can omit the file extension.

column

Is an alphanumeric field in the data source that contains the name of an image file. Use the COLUMN attribute described in *Identifying a Report Component in a Web Query StyleSheet* on page 57. Enclose *column* in parentheses ().

The field containing the file name or image must be a display field or BY field referenced in the request.

Note that the value of the field is interpreted exactly as if it were typed as the URL of the image in the StyleSheet. If you omit the suffix, .GIF is supplied by default. SET BASEURL can be useful for supplying the base URL of the images. If you do that, the value of the field does not have to include the complete URL.

This syntax is useful, for example, if you want to embed an image in a SUBHEAD, and you want a different image for each value of the BY field on which the SUBHEAD occurs.

byfield

Is the sort field that generated the subhead or subfoot.

POSITION

Is the starting position of the image.

+|-

Measures the horizontal or vertical distance from the upper-left corner of the report component in which the image is embedded.

x

Is the horizontal starting position of the image from the upper-left corner of the physical report page, expressed in the unit of measurement specified by the UNITS parameter.

Enclose the *x* and *y* values in parentheses (). Do not include a comma (,) between them.

y

Is the vertical starting position of the image from the upper-left corner of the physical report page, expressed in the unit of measurement specified by the UNITS parameter.

SIZE

Is the size of the image. By default, an image is added at its original size.

w

Is the width of the image, expressed in the unit of measurement specified by the UNITS parameter.

Enclose the *w* and *h* values in parentheses. Do not include a comma (,) between them.

h

Is the height of the image, expressed in the unit of measurement specified by the UNITS parameter.

Example: Adding a GIF Image to an HTML Report With an Internal Cascading Style Sheet

A URL locates the image file, GOTHAM.GIF, on a server named WEBSRVR1. The TYPE attribute adds the image to the report heading. POSITION places the image one-quarter inch horizontally and one-tenth inch vertically from the upper-left corner of the report page. The image is one inch wide and one inch high as specified by SIZE.

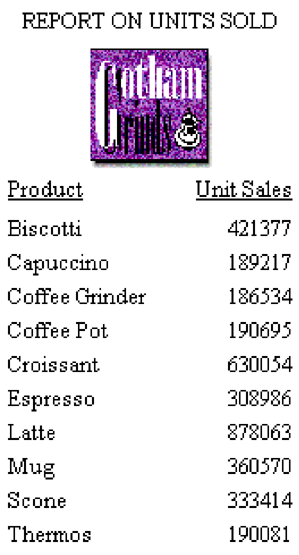
Report Request

```
TABLE FILE GGSALES
SUM UNITS BY PRODUCT
ON TABLE SUBHEAD
"REPORT ON UNITS SOLD"
" "
" "
" "
" "
" "
ON TABLE SET STYLE *
INCLUDE=IBFS:/stylesheet_location/custom_stylesheet.sty,$
END
```

***custom_stylesheet*.sty**

```
TYPE=REPORT, GRID=OFF, $
TYPE=TABHEADING,IMAGE=/webquery/ibi_html/GGDEMO/GOTHAM.GIF,
   POSITION=(.25 .10), SIZE=(1 1), $
```

The company logo is positioned and sized in the report heading.



REPORT ON UNITS SOLD

Product	Unit Sales
Biscotti	421377
Capuccino	189217
Coffee Grinder	186534
Coffee Pot	190695
Croissant	630054
Espresso	308986
Latte	878063
Mug	360570
Scone	333414
Thermos	190081

Example:

Adding a GIF Image to a PDF Report

The image file for this example is GOTHAM.GIF. The POSITION attribute places the image one-quarter inch horizontally and one-quarter vertically from the upper-left corner of the report page. The image is one-half inch wide and one-half inch high as specified by SIZE.

Report Request

```
TABLE FILE GGSALES
SUM UNITS BY PRODUCT
ON TABLE SUBHEAD
"Report on Units Sold"
" "
" "
" "
" "
" "
ON TABLE PCHOLD FORMAT PDF
ON TABLE SET STYLE *
INCLUDE=IBFS:/stylesheet_location/custom_stylesheet.sty,$
END
```

***custom_stylesheet*.sty**

```
TYPE=TABHEADING, IMAGE=GOTHAM.GIF, POSITION=(.25 .25), SIZE=(.5 .5), $
```

The report is:

```
Report on Units Sold


Product              Unit Sales

Biscotti                 421377
Capuccino                189217
Coffee Grinder           186534
Coffee Pot               190695
Croissant                630054
Espresso                 308986
Latte                    878063
Mug                      360570
Scone                    333414
Thermos                  190081
```

Syntax: How to Add an Image From a BLOB Field to a PDF, DHTML, or HTML Report

For PDF, HTML, and DHTML output against data sources that support the Binary Large Object (BLOB) data type (Db2 and Microsoft SQL Server, using its BYTEA data type), an image can be stored in a BLOB field in the data source.

Web Query StyleSheets used to produce report output in PDF, HTML, or DHTML format can access a BLOB field as an image source when an instance of the BLOB field contains an exact binary copy of a GIF or JPG image. HTML and DHTML reports also support PNG images. Images of different formats (GIF, JPG, PNG) can be mixed within the same BLOB field. Web Query can determine the format from the header of the image. The image can be inserted in report columns, headings, footings, subheadings, and subfootings.

The BLOB field must be referenced in a PRINT or LIST command in the request (aggregation is not supported).

With the following SET command, BLOB images will work for both HTML and DHTML in all browsers:

- SET BASEURL='' (required to make embedded images work as it overrides the default setting sent from the Web Query Client).

```
TYPE={REPORT|heading}, IMAGE={url|file|(column)} [,BY=byfield]
   [,POSITION=([+|-]x [+|-]y )] [,SIZE=(w h)] [,PRESERVERATIO={ON|OFF}], $


TYPE=DATA, COLUMN=imagefield, IMAGE=(imagefield), SIZE=(wh)
[,PRESERVERATIO={ON|OFF}], $
```

where:

REPORT

Embeds an image in the body of a report. The image appears in the background of the report. REPORT is the default value (not supported for images stored in BLOB fields, which are supported for PDF output).

heading

Embeds an image in a heading or footing. Valid values are FOOTING, HEADING, SUBHEAD, and SUBFOOT.

If the image is to be embedded in a heading, subheading, footing, or subfooting rather than a column, the StyleSheet declaration is responsible for placing the image in the heading, subheading, footing, or subfooting. To make the BLOB image accessible to the StyleSheet, the BLOB field must be referenced in the PRINT or LIST command with the NOPRINT option. Do not reference the BLOB field name in the heading or footing itself.

Provide sufficient blank space in the heading or footing so that the image does not overlap the heading or footing text. Also, you may want to place heading or footing text to the right of the image using spot markers or the POSITION attribute in the StyleSheet.

file

Is the name of the image file. It must reside on the Reporting Server in a directory named on EDAPATH or APPPATH. If the file is not on the search path, supply the full path name.

When specifying a GIF file, you can omit the file extension.

column

Is a BLOB field in the data source that contains an exact binary copy of a GIF or JPG image. HTML and DHTML formats also support images in PNG format. Images of different formats (GIF, JPG, PNG) can be mixed within the same BLOB field. Web Query can determine the format from the header of the image. The image can be inserted in report columns, headings, footings, subheadings and subfootings. Use the COLUMN attribute described in *Identifying a Report Component in a Web Query StyleSheet* on page 57. Enclose *column* in parentheses.

The field containing the file name or image must be a display field or BY field referenced in the request.

byfield

Is the sort field that generated the subhead or subfoot.

imagefield

Is any valid column reference for the BLOB field that contains the image. Note that the BLOB field must be referenced in a PRINT or LIST command in the request.

If omitted, the default size is 1 inch by 1 inch. The width of the column and the spacing between the lines is automatically adjusted to accommodate the image.

`POSITION`

Is the starting position of the image.

`+|-`

Measures the horizontal or vertical distance from the upper-left corner of the report component in which the image is embedded.

x

Is the horizontal starting position of the image from the upper-left corner of the physical report page, expressed in the unit of measurement specified by the UNITS parameter.

Enclose the *x* and *y* values in parentheses (). Do not include a comma (,) between them.

y

Is the vertical starting position of the image from the upper-left corner of the physical report page, expressed in the unit of measurement specified by the UNITS parameter.

`SIZE`

Is the size of the image. By default, an image is added at its original size. Note that images stored in BLOB fields are supported only for PDF, HTML, and DHTML output.

w

Is the width of the image, expressed in the unit of measurement specified by the UNITS parameter.

Enclose the *w* and *h* values in parentheses. Do not include a comma (,) between them.

h

Is the height of the image, expressed in the unit of measurement specified by the UNITS parameter.

If SIZE is omitted, the original dimensions of the image are used (any GIF, JPG, or PNG image has an original, unscaled size based on the dimensions of its bitmap).

`[PRESERVERATIO={ON|OFF}]`

Not supported for images in PNG format. PRESERVERATIO=ON specifies that the aspect ratio (ratio of height to width) of the image should be preserved when it is scaled to the specified SIZE. This avoids distorting the appearance of the image. The image is scaled to the largest size possible within the bounds specified by SIZE for which the aspect ratio can be maintained. Supported for PDF output. OFF does not maintain the aspect ratio. OFF is the default value.

The actual size of an image stored in a BLOB field may vary from image to image, and scaling the images to a designated size allows them to better fit into a columnar report.

Note: Images stored in a BLOB field are supported only for PDF, HTML, and DHTML output.

Example: Inserting an Image From a BLOB Field Into a Report Column

The data source named retaildetail contains product information for a sports clothing and shoe retailer. The data source named retailimage has the same product ID field as retaildetail and has an image of each product stored in a field named *prodimage* whose data type is BLOB.

The following Master File describes the data source named retaildetail.

```
FILENAME=RETAILDETAIL, SUFFIX=DB2, $
  SEGMENT=SEG01, SEGTYPE=S0, $
    FIELDNAME=FOCLIST, ALIAS=FOCLIST, USAGE=I5, ACTUAL=I4, $
    FIELDNAME=PRODUCTID, ALIAS=ProductId, USAGE=A5, ACTUAL=A5,
      MISSING=ON, $
    FIELDNAME=DEPARTMENT, ALIAS=Department, USAGE=A10, ACTUAL=A10,
      MISSING=ON, $
    FIELDNAME=CATEGORY, ALIAS=Category, USAGE=A30, ACTUAL=A30,
      MISSING=ON, $
    FIELDNAME=SPORTS, ALIAS=Sports, USAGE=A30, ACTUAL=A30,
      MISSING=ON, $
    FIELDNAME=GENDER, ALIAS=Gender, USAGE=A10, ACTUAL=A10,
      MISSING=ON, $
    FIELDNAME=BRAND, ALIAS=Brand, USAGE=A25, ACTUAL=A25,
      MISSING=ON, $
    FIELDNAME=STYLE, ALIAS=Style, USAGE=A25, ACTUAL=A25,
      MISSING=ON, $
    FIELDNAME=COLOR, ALIAS=Color, USAGE=A25, ACTUAL=A25,
      MISSING=ON, $
    FIELDNAME=NAME, ALIAS=Name, USAGE=A80, ACTUAL=A80,
      MISSING=ON, $
    FIELDNAME=DESCRIPTION, ALIAS=Description, USAGE=A1000, ACTUAL=A1000,
      MISSING=ON, $
    FIELDNAME=PRICE, ALIAS=Price, USAGE=D7.2, ACTUAL=D8,
      MISSING=ON, $
```

The following Master File describes the data source named retailimage, which has the same product ID field as retaildetail and has an image of each product stored in a field named *prodimage* whose data type is BLOB.

```
FILENAME=RETAILIMAGE, SUFFIX=DB2, $
  SEGMENT=RETAILIMAGE, SEGTYPE=S0, $
    FIELDNAME=PRODUCTID, ALIAS=PRODUCTID, USAGE=A5, ACTUAL=A5, $
    FIELDNAME=PRODIMAGE, ALIAS=F02BLOB50000, USAGE=BLOB, ACTUAL=BLOB,
      MISSING=ON, $
```

The following example illustrates how to join the two data sources and print product names and prices with the corresponding image. The output is generated in DHTML format.

Report Request

```
JOIN PRODUCTID IN RETAILDETAIL TO PRODUCTID IN RETAILIMAGE
TABLE FILE RETAILDETAIL
HEADING CENTER
"Product List"
" "
PRINT NAME/A20 PRICE PRODIMAGE AS 'PICTURE'
BY PRODUCTID NOPRINT
BY NAME NOPRINT
ON NAME UNDER-LINE
-**************************
-* Lines between asterisk lines required for BLOB image support
-* for HTML and DHTML formats.
ON TABLE SET HTMLEMBEDIMG AUTO
ON TABLE SET HTMLARCHIVE ON
-**************************
ON TABLE PCHOLD FORMAT DHTML
ON TABLE SET STYLE *
INCLUDE=IBFS:/stylesheet_location/custom_stylesheet.sty,$
END
```

***custom_stylesheet*.sty**

```
TYPE=REPORT,COLOR=BLUE,FONT=ARIAL, GRID=OFF, $
TYPE=HEADING, SIZE=18, COLOR=RED, $
TYPE=DATA, COLUMN=PRODIMAGE, IMAGE=(PRODIMAGE),SIZE=(1 1), $
```

The image is placed in the report column using the following StyleSheet declaration, which names the image field, and establishes the size and position in the column for the image.

```
TYPE=DATA, COLUMN=PRODIMAGE, IMAGE=(PRODIMAGE), SIZE=(1 1), $
```

The partial output shows that DHTML format preserves the specified spacing.

Product List

NAME	PRICE	PICTURE
Mizia 9 Spike Classi	94.00	
Kine Cal Ripken Mid	49.00	
Mizia 9 Spike Classi	89.00	
Adomas Excelsior Low	49.00	
3N3 Prometal Hi Base	84.00	
3N3 Prometal Hi Base	84.00	
Adomas Spinner 7 Lo	54.00	

The following request generates the output in HTML format.

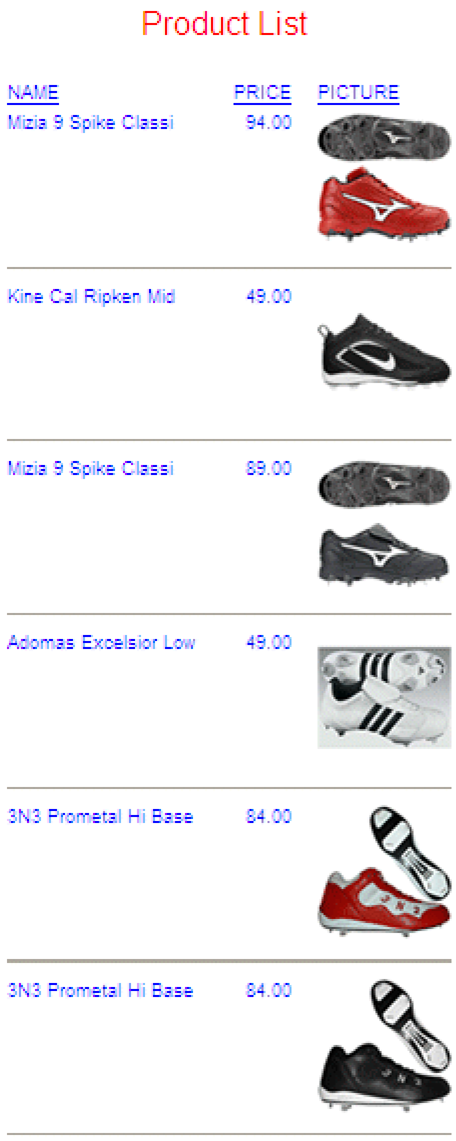
Report Request

```
JOIN PRODUCTID IN RETAILDETAIL TO PRODUCTID IN RETAILIMAGE
TABLE FILE RETAILDETAIL
HEADING CENTER
"Product List"
" "
PRINT NAME/A20 PRICE PRODIMAGE AS 'PICTURE'
BY PRODUCTID NOPRINT
BY NAME NOPRINT
ON NAME UNDER-LINE
-**************************
-* Lines between asterisk lines required for BLOB image support
-* for HTML and DHTML formats.
ON TABLE SET HTMLEMBEDIMG AUTO
ON TABLE SET HTMLARCHIVE ON
-**************************
ON TABLE SET STYLE *
INCLUDE=IBFS:/stylesheet_location/custom_stylesheet.sty,$
END
```

***custom_stylesheet*.sty**

```
TYPE=REPORT, COLOR=BLUE, GRID=OFF, FONT=ARIAL, $
TYPE=HEADING, SIZE=18, COLOR=RED, $
TYPE=DATA, COLUMN=PRODIMAGE, IMAGE=(PRODIMAGE),SIZE=(1 1), $
```

The partial output shows that the spacing is different because the browser removes blank spaces for HTML report output.



Product List

NAME	PRICE	PICTURE
Mizia 9 Spike Classi	94.00	
Kine Cal Ripken Mid	49.00	
Mizia 9 Spike Classi	89.00	
Adomas Excelsior Low	49.00	
3N3 Prometal Hi Base	84.00	
3N3 Prometal Hi Base	84.00	

The following request generates the report output in PDF format.

Report Request

```
JOIN PRODUCTID IN RETAILDETAIL TO PRODUCTID IN RETAILIMAGE
TABLE FILE RETAILDETAIL
HEADING CENTER
"Product List"
" "
PRINT NAME/A20 PRICE PRODIMAGE AS 'PICTURE'
BY PRODUCTID NOPRINT
BY NAME NOPRINT
ON NAME UNDER-LINE
-**************************
-* Lines between asterisk lines required for BLOB image support
-* for HTML and DHTML formats.
ON TABLE SET HTMLEMBEDIMG AUTO
ON TABLE SET HTMLARCHIVE ON
-**************************
ON TABLE PCHOLD FORMAT PDF
ON TABLE SET STYLE *
INCLUDE=IBFS:/stylesheet_location/custom_stylesheet.sty,$
END
```

***custom_stylesheet*.sty**

```
TYPE=REPORT, COLOR=BLUE, GRID=OFF, $
TYPE=HEADING, SIZE=18, FONT=ARIAL, COLOR=RED, $
TYPE=DATA, COLUMN=PRODIMAGE, IMAGE=(PRODIMAGE), SIZE=(1 1), $
```

The PDF partial output preserves specified spacing providing results similar to DHTML output.

Product List

NAME	PRICE	PICTURE
Mizia 9 Spike Classi	94.00	
Kine Cal Ripken Mid	49.00	
Mizia 9 Spike Classi	89.00	
Adomas Excelsior Low	49.00	
3N3 Prometal Hi Base	84.00	
3N3 Prometal Hi Base	84.00	
Adomas Spinner 7 Lo	54.00	

Example: Inserting an Image From a BLOB Field Into a Subheading

The data source named retaildetail contains product information for a sports clothing and shoe retailer. The data source named retailimage has the same product ID field as retaildetail and has an image of each product stored in a field named *prodimage* whose data type is BLOB.

The following example illustrates how to join the two data sources and print product images in a subheading. The output is generated in DHTML format. It can also be generated in HTML or PDF format.

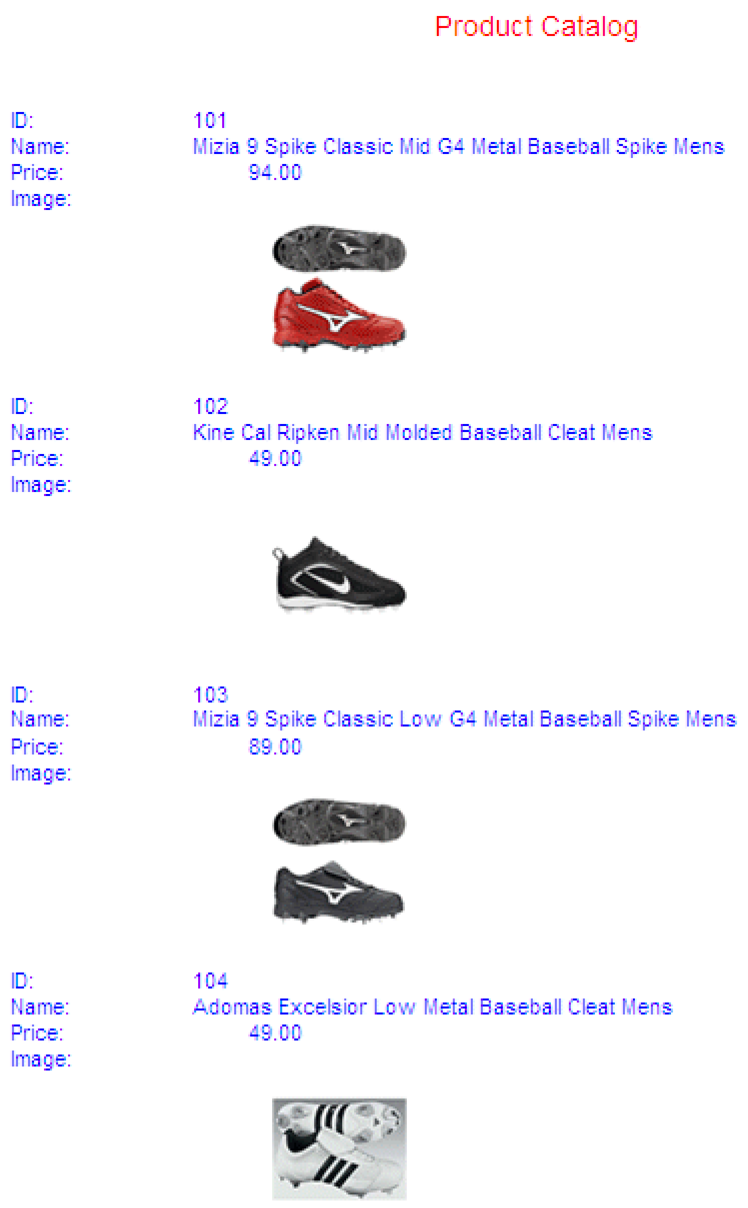
Report Request

```
SET BASEURL=''
JOIN PRODUCTID IN RETAILDETAIL TO PRODUCTID IN RETAILIMAGE
TABLE FILE RETAILDETAIL
HEADING CENTER
"Product Catalog"
" "
PRINT NAME NOPRINT PRODIMAGE NOPRINT
BY PRODUCTID NOPRINT
ON PRODUCTID SUBHEAD
""
" ID: <10<PRODUCTID "
" Name: <10<NAME "
" Price: <7<PRICE "
" Image: "
""
""
""
""
""
-**************************
-* Lines between asterisk lines required for BLOB image support
-* for HTML and DHTML formats.
ON TABLE SET HTMLEMBEDIMG AUTO
ON TABLE SET HTMLARCHIVE ON
-**************************
ON TABLE PCHOLD FORMAT DHTML
ON TABLE SET STYLE *
INCLUDE=IBFS:/stylesheet_location/custom_stylesheet.sty,$
END
```

***custom_stylesheet*.sty**

```
TYPE=REPORT, COLOR=BLUE, FONT=ARIAL, $
TYPE=HEADING, COLOR=RED, SIZE=16, JUSTIFY=CENTER, $
TYPE=SUBHEAD, BY=PRODUCTID, IMAGE=(PRODIMAGE), SIZE=(1 1),
   POSITION=(+2 +1), $
```

The partial output is:



```
                         Product Catalog

ID:            101
Name:          Mizia 9 Spike Classic Mid G4 Metal Baseball Spike Mens
Price:               94.00
Image:

ID:            102
Name:          Kine Cal Ripken Mid Molded Baseball Cleat Mens
Price:               49.00
Image:

ID:            103
Name:          Mizia 9 Spike Classic Low G4 Metal Baseball Spike Mens
Price:               89.00
Image:

ID:            104
Name:          Adomas Excelsior Low Metal Baseball Cleat Mens
Price:               49.00
Image:
```

Example: Sizing an Image From a BLOB Field

The data source named retaildetail contains product information for a sports clothing and shoe retailer. The data source named retailimage has the same product ID field as retaildetail and has an image of each product stored in a field named *prodimage* whose data type is BLOB.

The following request joins the two data sources and displays the same image on three columns of output using different sizes and different PRESERVERATIO settings. Note that PRESERVERATIO=ON is not supported with images in PNG format.

The output is generated in DHTML format. It can also be generated in HTML or PDF format.

Report Request

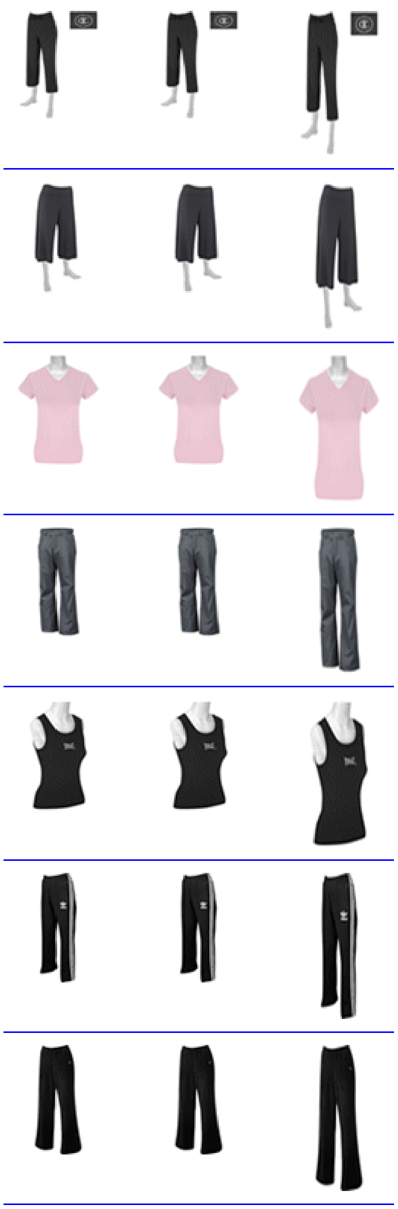
```
JOIN PRODUCTID IN RETAILDETAIL TO PRODUCTID IN RETAILIMAGE
TABLE FILE RETAILDETAIL
PRINT PRODIMAGE AS '' PRODIMAGE AS '' PRODIMAGE AS ''
BY STYLE NOPRINT
WHERE NAME CONTAINS 'Pant' OR 'Tank'
ON STYLE UNDER-LINE
ON TABLE SET PAGE NOPAGE
-**************************
-* Lines between asterisk lines required for BLOB image support
-* for HTML and DHTML formats.
ON TABLE SET HTMLEMBEDIMG AUTO
ON TABLE SET HTMLARCHIVE ON
-**************************
ON TABLE PCHOLD FORMAT DHTML
ON TABLE SET STYLE *
INCLUDE=IBFS:/stylesheet_location/custom_stylesheet.sty,$
END
```

***custom_stylesheet*.sty**

```
TYPE=REPORT, COLOR=BLUE, FONT = ARIAL, $
TYPE=DATA, COLUMN=P1, IMAGE=(PRODIMAGE), SIZE=(.75 .75), $
TYPE=DATA, COLUMN=P2, IMAGE=(PRODIMAGE), SIZE=(.75 1), PRESERVERATIO=ON, $
TYPE=DATA, COLUMN=P3, IMAGE=(PRODIMAGE), SIZE=(.75 1), PRESERVERATIO=OFF,$
```

Note that PRESERVERATIO=OFF is specified for the second column to preserve the image height and width ratio for that column even though the styling SIZE height specifies a different value than the first column image styling. In addition, PRESERVERATIO=OFF is specified for the third column, so for that column the image height to width ratio is not preserved and is rendered as specified by the styling SIZE height and width values specified in the request (FEX).

The partial output is:



Example: Inserting an Image From a BLOB Field in a Summary Report

In order to insert an image from a BLOB field in a report that displays summary data, you must include two display commands in the request, a SUM command for the summary information and a PRINT or LIST command for displaying the image and any other detail data.

The data source named retaildetail contains product information for a sports clothing and shoe retailer. The data source named retailimage has the same product ID field as retaildetail and has an image of each product stored in a field named *prodimage* whose data type is BLOB.

The following example illustrates how to join the two data sources. It contains two display commands, a SUM command and a PRINT command. The SUM command aggregates the total price for each category and displays this category name and total price in a subheading, The PRINT command displays the image for each item in the category along with its individual product number and price in a subfooting.

The output is generated in DHTML format. It can also be generated in HTML or PDF format.

Report Request

```
JOIN PRODUCTID IN RETAILDETAIL TO PRODUCTID IN RETAILIMAGE
TABLE FILE RETAILDETAIL
HEADING CENTER
"Product Price Summary"
" "
SUM PRICE NOPRINT
BY CATEGORY NOPRINT
ON CATEGORY SUBHEAD
" Category: <CATEGORY "
" Total Price: <PRICE "
" "
```

```
PRINT PRICE NOPRINT PRODIMAGE NOPRINT
BY CATEGORY NOPRINT
BY PRODUCTID NOPRINT
ON PRODUCTID SUBFOOT
" "
" "
" "
" "
" "
" "
" Product #: <PRODUCTID "
" Name: <NAME "
" Price: <FST.PRICE "
ON TABLE SET PAGE NOPAGE
-**************************
-* Lines between asterisk lines required for BLOB image support
-* for HTML and DHTML formats.
ON TABLE SET HTMLEMBEDIMG AUTO
ON TABLE SET HTMLARCHIVE ON
-**************************
ON TABLE PCHOLD FORMAT DHTML
ON TABLE SET STYLE *
INCLUDE=IBFS:/stylesheet_location/custom_stylesheet.sty,$
END
```

***custom_stylesheet*.sty**

```
TYPE=REPORT, COLOR=BLUE, FONT=ARIAL, $
TYPE=HEADING, COLOR=RED, SIZE=14, STYLE=BOLD, JUSTIFY=CENTER, $
TYPE=SUBHEAD, COLOR=RED, SIZE=12, STYLE=BOLD, JUSTIFY=CENTER, $
TYPE=SUBFOOT, BY=PRODUCTID, IMAGE=(PRODIMAGE), SIZE=(1 1),
   POSITION=(0 0), $
TYPE=SUBFOOT, BY=PRODUCTID, OBJECT=FIELD, ITEM=1, WRAP=5, $
```

The output for the first category is:

Product Price Summary

Category: Crib Shoes / Soft Bottoms
Total Price: 106.00

Product #: 206
Name: StonyBay Kids Mate Crib Shoe
Price: 30.00

Product #: 207
Name: StonyBay Kids Mate Crib Shoe
Price: 30.00

Product #: 208
Name: D2D Classic Dazzlin' Crib Shoe
Price: 24.00

Product #: 209
Name: ShoeTech KJ574 Crib
Price: 22.00

Working With Mailing Labels and Multi-Pane Pages

You can print sheets of laser printer mailing labels by dividing each page into a matrix of subpages, each corresponding to a single label. Each page break in the report positions the printer at the top of the next label.

Multi-pane printing places a whole report on a single printed page. You can create columns or rows so that when text overflows on one page, it appears in the next column or row on the same page rather than on the next page.

These features apply to a PDF report.

Reference: Attributes for Mailing Labels and Multi-Pane Printing

In addition to the attributes in the table, you can use standard margin attributes (for example, LEFTMARGIN or TOPMARGIN) to position the entire sheet of labels at once, creating an identical margin for each sheet.

Attribute	Description	Applies to
`PAGEMATRIX`	Sets the number of columns and rows of labels on a page.	PDF
`ELEMENT`	Sets the width and height of each label, expressed in the unit of measurement specified by the UNITS parameter.	PDF
`GUTTER`	Sets the horizontal and vertical distance between each label, expressed in the unit of measurement specified by the UNITS parameter.	PDF
`MATRIXORDER`	Sets the order in which the labels are printed.	PDF
`LABELPROMPT`	Sets the position of the first label on the mailing label sheet.	PDF

Procedure: How to Set Up a Report to Print Mailing Labels

1. Create the label as a page heading.
2. Sort the labels but use NOPRINT to suppress sort field display. Only the fields embedded in the page heading will print.
3. Insert a page break on a sort field to place each new field value on a separate label.

Syntax: How to Print Mailing Labels or a Multi-Pane Report

```
[TYPE=REPORT,] PAGEMATRIX=(c r), ELEMENT=(w h), [GUTTER=(x y),]
   [MATRIXORDER={VERTICAL|HORIZONTAL},] [LABELPROMPT={OFF|ON},] $
```

where:

`TYPE=REPORT`

Applies the settings to the entire report. Not required, as it is the default.

`c`

Is the number of columns of labels across the page.

Enclose the values *c* and *r* in parentheses, and do not include a comma (,) between them.

`r`

Is the number of rows of labels down the page.

`w`

Is the width of each label.

Enclose the values *w* and *h* in parentheses, and do not include a comma (,) between them.

`h`

Is the height of each label.

`GUTTER`

Is the distance between each label.

`x`

Is the horizontal distance between each label.

Enclose the values *x* and *y* in parentheses, and do not include a comma (,) between them.

`y`

Is the vertical distance between each label.

`MATRIXORDER`

Is the order in which the labels are printed.

`VERTICAL`

Prints the labels down the page.

`HORIZONTAL`

Prints the labels across the page.

`LABELPROMPT`

Is the position of the first label on the mailing label sheet.

`OFF`

Starts the report on the first label on the sheet. OFF is the default value.

`ON`

Prompts you at run time for the row and column number at which to start printing. All remaining labels follow consecutively. This feature allows partially used sheets of labels to be re-used.

Example: Printing Mailing Labels

The following example illustrates how to print on 81/2 x 11 sheets of address labels.

Report Request

```
TABLE FILE EMPLOYEE
BY LAST_NAME NOPRINT BY FIRST_NAME NOPRINT
ON FIRST_NAME PAGE-BREAK
HEADING
"<FIRST_NAME <LAST_NAME"
"<ADDRESS_LN1"
"<ADDRESS_LN2"
"<ADDRESS_LN3"
ON TABLE PCHOLD FORMAT PDF
ON TABLE SET STYLE *
INCLUDE=IBFS:/stylesheet_location/custom_stylesheet.sty,$
END
```

***custom_stylesheet*.sty**

```
TYPE=REPORT, UNITS=IN, PAGESIZE=LETTER, LEFTMARGIN=0.256, TOPMARGIN=0.5,
   PAGEMATRIX=(2 5), ELEMENT=(4 1), GUTTER=(0.188 0), $
```

The first page of labels prints as follows:

```
JOHN BANNING                    DIANE JONES
160 LOMBARDO AVE.
APT 4C                          235 MURRAY HIL PKWY
FREEPORT NY 11520               RUTHERFORD NJ 07073


ROSEMARIE BLACKWOOD             JOHN MCCOY
MRS. P. JONES                   ASSOCIATED
3704 FARRAGUT RD.               2 PENN PLAZA
BROOKLYN NY 11210               NEW YORK NY 10001


BARBARA CROSS                   ROGER MCKNIGHT
APT 2G                          APT 4D
147-15 NORTHERN BLD             117 HARRISON AVE.
FLUSHING NY 11354               ROSELAND NJ 07068


MARY GREENSPAN                  ANTHONY ROMANS

13 LINDEN AVE.                  271 PRESIDENT ST.
JERSEY CITY NJ 07300            FREEPORT NY 11520


JOAN IRVING                     MARY SMITH
APT 2J                          ASSOCIATED
123 E 32 ST.                    2 PENN PLAZA
NEW YORK NY 10001               NEW YORK NY 10001
```

Example: Printing a Multi-Pane Report

The following example illustrates how to divide the first report page in two columns so that the second report page appears in the second column of the first page. A PAGE-BREAK creates a multi-page report for the purpose of this example.

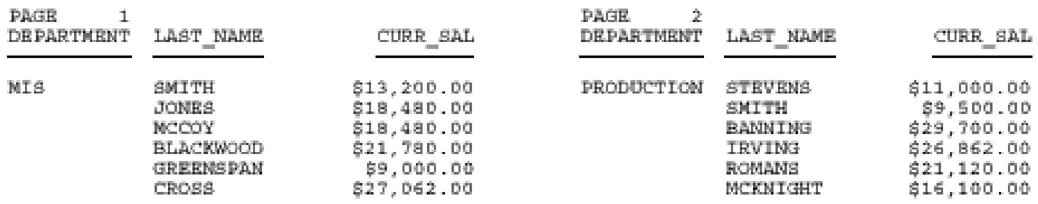
Report Request

```
TABLE FILE EMPLOYEE
PRINT LAST_NAME AND CURR_SAL
BY DEPARTMENT
ON DEPARTMENT PAGE-BREAK
HEADING
"PAGE <TABPAGENO"
ON TABLE PCHOLD FORMAT PDF
ON TABLE SET STYLE *
INCLUDE=IBFS:/stylesheet_location/custom_stylesheet.sty,$
END
```

***custom_stylesheet*.sty**

```
TYPE=REPORT, SIZE=8, UNITS=IN, PAGESIZE=LETTER, PAGEMATRIX=(2 1),
   ELEMENT=(3.5 8.0), MATRIXORDER=VERTICAL, $
```

The report prints as:



```
PAGE        1                               PAGE        2
DEPARTMENT  LAST_NAME        CURR_SAL       DEPARTMENT  LAST_NAME        CURR_SAL

MIS         SMITH          $13,200.00       PRODUCTION  STEVENS        $11,000.00
            JONES          $18,480.00                   SMITH           $9,500.00
            MCCOY          $18,480.00                   BANNING        $29,700.00
            BLACKWOOD      $21,780.00                   IRVING         $26,862.00
            GREENSPAN       $9,000.00                   ROMANS         $21,120.00
            CROSS          $27,062.00                   MCKNIGHT       $16,100.00
```

Chapter 8

Using Headings, Footings, Titles, and Labels

After you have selected the data for a report, you can make it more meaningful by adding headings, footings, titles, and labels. Headings and footings supply key information, such as the purpose of a report and its audience. They also provide structure, helping the user navigate to the detail sought. Titles and labels identify individual pieces of data, ensuring correct interpretation. These components supply context for data and enhance the visual appeal of a report.

Note: InfoAssist has built-in styling options that generate some of the StyleSheet syntax described in this chapter.

In this chapter:

Creating a Custom Report or Worksheet Title

You can create a report title that:

❏ Overrides the default report title that appears in the title bar of your browser in an HTML report.

❏ Replaces the default worksheet tab name with the name you specify in an XLSX or EXL2K report.

The worksheet tab names for an Excel Table of Contents report are the BY field values that correspond to the data on the current worksheet. If the user specifies the TITLETEXT keyword in the StyleSheet, it will be ignored.

- ❑ Excel limits the length of worksheet titles to 31 characters. The following special characters cannot be used: ':', '?', '*', and '/'.
- ❑ If you want to use date fields as the bursting BY field, you can include the dash (-) character instead of the slash (/) character. The dash (-) character is valid in an Excel tab title. However, if you do use the slash (/) character, Web Query will substitute it with the dash (-) character.

Syntax: How to Create a Custom Report Title

Add the following declaration to your Web Query StyleSheet:

```
TYPE=REPORT, TITLETEXT='title', $
```

where:

title

Is the text for your title.

The maximum number of characters for:

- ❑ The worksheet tab name in an XLSX or EXL2K report is 31. Any text that exceeds 31 characters will be truncated.
- ❑ The browser title for an HTML report is 95. This is a limit imposed by the browser.

  Text specified in the title is placed in the file as is and is not encoded. Special characters, such as <, >, and &, should not be used since they have special meaning in HTML and may produce unpredictable results.

  Note: The words "Microsoft Internet Explorer" are always appended to any HTML report title.

Example: Creating a Custom Report Title in an HTML Report

The following example illustrates how you can replace the default report title in an HTML report using the TITLETEXT attribute in your StyleSheet.

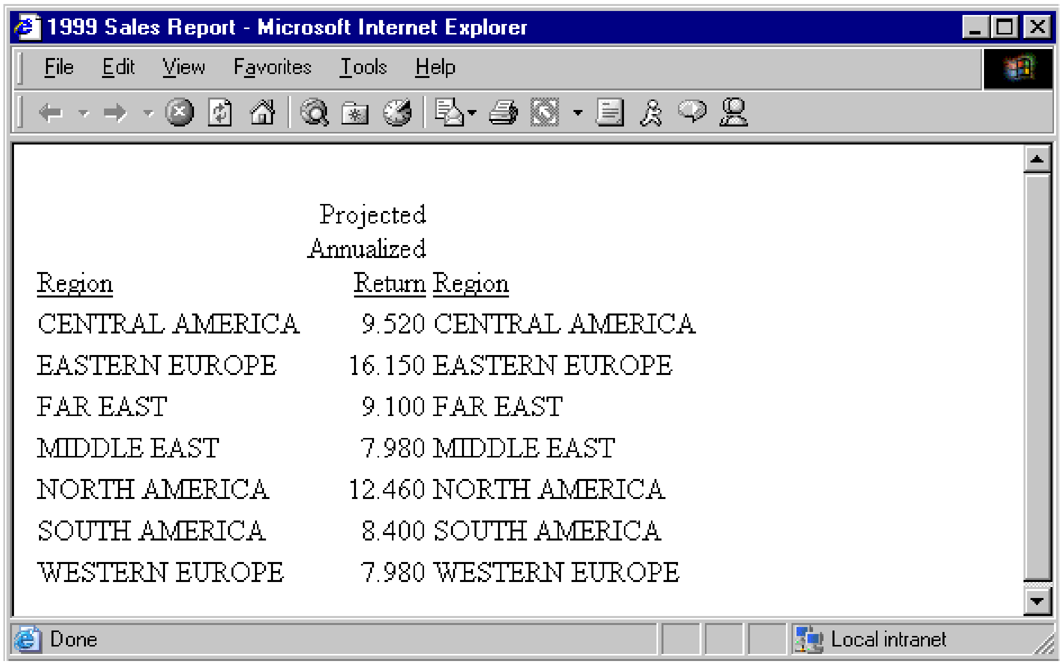
Report Request

```
TABLE FILE SHORT
SUM PROJECTED REGION
BY REGION
ON TABLE SET STYLE *
INCLUDE=IBFS:/stylesheet_location/custom_stylesheet.sty,$
END
```

***custom_stylesheet*.sty**

```
TYPE=REPORT, TITLETEXT='1999 Sales Report', $
TYPE=REPORT, GRID=OFF, $
```

The output is:



```
                        Projected
                       Annualized
Region                  Return Region
CENTRAL AMERICA          9.520 CENTRAL AMERICA
EASTERN EUROPE          16.150 EASTERN EUROPE
FAR EAST                 9.100 FAR EAST
MIDDLE EAST              7.980 MIDDLE EAST
NORTH AMERICA           12.460 NORTH AMERICA
SOUTH AMERICA            8.400 SOUTH AMERICA
WESTERN EUROPE           7.980 WESTERN EUROPE
```

Example: Creating a Custom Report Title in an EXL2K Report

The following example illustrates how you can replace the default worksheet tab name in an EXL2K report using the TITLETEXT attribute in your StyleSheet.

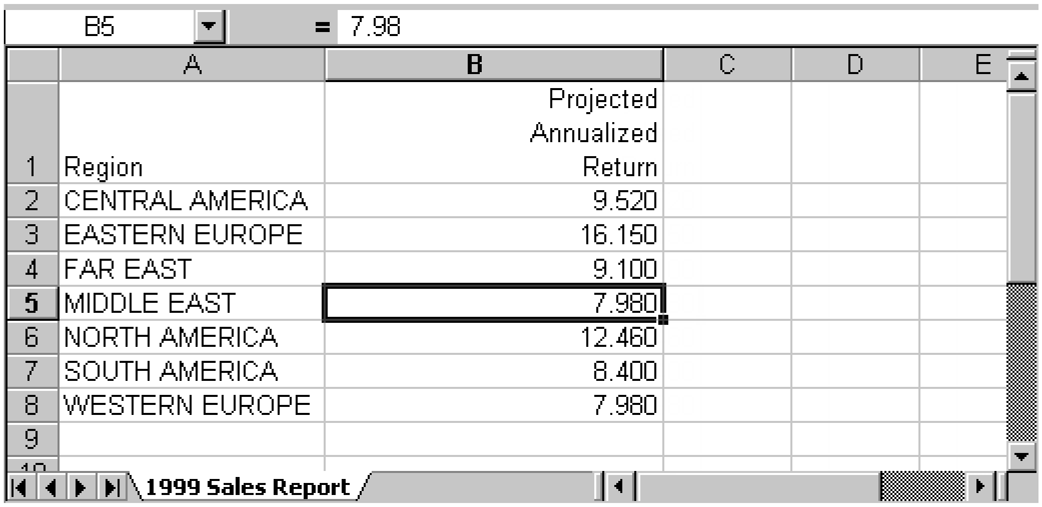
Report Request

```
TABLE FILE SHORT
SUM PROJECTED_RETURN
BY REGION
ON TABLE PCHOLD FORMAT EXL2K
ON TABLE SET STYLE *
INCLUDE=IBFS:/stylesheet_location/custom_stylesheet.sty,$
END
```

***custom_stylesheet*.sty**

```
TYPE=REPORT, TITLETEXT='1999 Sales Report', $
```

The output is:



B5 = 7.98

	A	B
1	Region	Projected Annualized Return
2	CENTRAL AMERICA	9.520
3	EASTERN EUROPE	16.150
4	FAR EAST	9.100
5	MIDDLE EAST	7.980
6	NORTH AMERICA	12.460
7	SOUTH AMERICA	8.400
8	WESTERN EUROPE	7.980
9		

Freezing HTML Headings, Footings, and Column Titles

You may want to scroll the data in a report while freezing headings, column titles, and footings in order to see the context of the report output while scrolling.

Using StyleSheet attributes, you can set aside a scrollable area for HTML report output.

Syntax: How to Create a Scrollable Area in an HTML Report

```
TYPE=REPORT,HFREEZE={OFF|ON|TOP|BOTTOM},[SCROLLHEIGHT={AUTO|nn[.n]}], $
```

where:

`HFREEZE=OFF`

Does not freeze the heading, column titles, grand totals, and footing. OFF is the default value.

`HFREEZE=ON`

Freezes the heading, column titles, grand totals, and footing.

`HFREEZE=TOP`

Freezes the heading and column titles.

`HFREEZE=BOTTOM`

Freezes the grand totals and footing.

`SCROLLHEIGHT=AUTO`

Automatically sizes the HFREEZE report within the browser page or the frame within the page.

nn`[.`*n*`]`

Is the height, in inches, of the scrollable area. The default is 4 inches.

Note: When the browser is manually resized after initial display of the HFREEZE report, the HFREEZE report size remains the same. Select the browser refresh option to reload the HFREEZE report, which will run the JavaScript that calculates the HFREEZE report size within the current page or frame.

Reference: HFREEZE With Blank Column Titles

The HTML HFREEZE reporting feature supports blank column titles. The vertical HFREEZE scroll bar will be aligned with the first row of report data.

Reference: Usage Notes for Freezing Areas of HTML Report Output

- The output format for the report must be HTML. If the format in the request is not HTML, the StyleSheet attributes to set aside a scrollable area for HTML format will be ignored.
- Accordion reports do not support column freezing.
- WRAP styling is not supported with the HTML HFREEZE reporting feature.

- ❏ The HFREEZE reporting feature does not support dynamic addition of HTML elements and tags. HTML HFREEZE alignment and scrolling features rely on a specific structure that includes both HTML and JavaScript elements. Inserting additional HTML or JavaScript elements interferes with this structure and is not supported.

Repeating Headings and Footings on Panels in PDF Report Output

When the columns presented on PDF reports cannot be displayed on a single page, the pages automatically panel. Paneling places subsequent columns for the same page on *overflow pages*. These overflow pages are generated until the entire width of the report is presented, after which the next vertical page is generated with a new page number and its associated horizontal panels.

In order to make panels following the initial panel more readable, you can designate that heading elements from the initial panel should be repeated on each subsequent panel using the HEADPANEL=ON StyleSheet attribute.

When paneling occurs, if default page numbering is used, the page number presented will include both the page number and the panel number (for example, 1.1, 1.2, 1.3). Turning HEADPANEL on will also cause the panel designation to be included in TABPAGENO.

HEADPANEL can be designated for the entire report, causing all headings and footings to be replicated on the paneled pages. It can also be turned on for just individual headings, footings, subheadings, or subfootings.

HEADPANEL causes borders from the initial page to be replicated on the paneled pages. Additional control of subheading and subfooting borders can be gained through the use of ALIGN-BORDERS which allows for the designation that subitem borders should align with the body of the data rather than the page or report headings. For more information about using ALIGN-BORDERS with HEADPANEL see *How to Align Subheading and Subfooting Margins With the Report Body* on page 165.

Syntax: How to Repeat Heading Elements on Panels

```
TYPE={REPORT|headfoot [BY=sortcolumn]}, HEADPANEL={ON|OFF}, $
```

where:

`REPORT`

Repeats all report headings, footings, page headings, page footings, subheadings, and subfootings.

headfoot

Identifies a heading or footing. Select from:

- ❑ **TABHEADING,** which is a report heading. This appears once at the beginning of the report and is generated by ON TABLE SUBHEAD.
- ❑ **TABFOOTING,** which is a report footing. This appears once at the end of the report and is generated by ON TABLE SUBFOOT.
- ❑ **HEADING,** which is a page heading. This appears at the top of every report page and is generated by HEADING.
- ❑ **FOOTING,** which is a page footing. This appears at the bottom of every report page and is generated by FOOTING.
- ❑ **SUBHEAD,** which is a sort heading. This appears at the beginning of a vertical (BY) sort group (generated by ON *sortfield* SUBHEAD).
- ❑ **SUBFOOT,** which is a sort footing. This appears at the end of a vertical (BY) sort group (generated by ON *sortfield* SUBFOOT).

BY

When there are several sort headings or sort footings, each associated with a different vertical sort (BY) column, this enables you to identify which sort heading or sort footing to format.

If there are several sort headings or sort footings associated with different vertical sort (BY) columns, and you omit this attribute and value, the formatting will be applied to all of the sort headings or footings.

sortcolumn

Specifies the vertical sort (BY) column associated with one of the report sort headings or sort footings.

ON

Repeats the specified heading or footing elements on each panel.

OFF

Displays heading or footing elements on the first panel only. OFF is the default value.

Note that the HEADPANEL=ON attribute can only be applied to the entire heading or footing, not individual lines or items within the heading or footing.

Example: Repeating All Headings and Footings on Report Panels

The following request against the GGSALES data source sums units sold, budgeted units sold, dollar sales, and budgeted sales by region, state, city, category, and product. The report has a page heading and, for each region, a subfooting.

Report Request

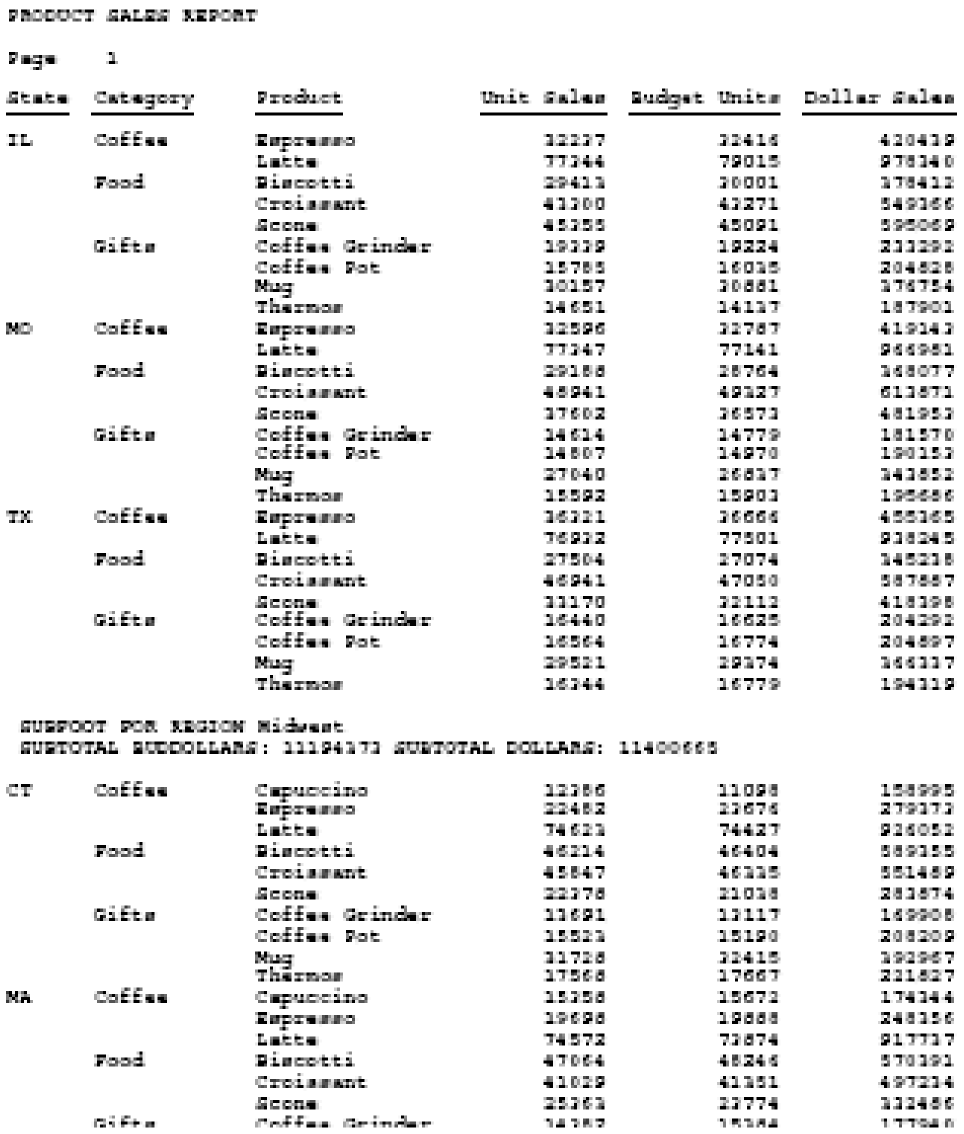
```
TABLE FILE GGSALES
HEADING
"PRODUCT SALES REPORT"
""
"Page<TABPAGENO"
""
SUM UNITS BUDUNITS DOLLARS BUDDOLLARS
BY REGION NOPRINT
BY ST BY CATEGORY BY PRODUCT
ON REGION SUBFOOT
" "
" SUBFOOT FOR REGION <REGION "
" SUBTOTAL BUDDOLLARS: <ST.BUDDOLLARS SUBTOTAL DOLLARS: <ST.DOLLARS "
" "
ON TABLE SET BYPANEL ON
ON TABLE PCHOLD FORMAT PDF
ON TABLE SET STYLE *
INCLUDE=IBFS:/stylesheet_location/custom_stylesheet.sty,$
END
```

***custom_stylesheet*.sty**

```
TYPE=REPORT, GRID=OFF, $
TYPE=REPORT, HEADPANEL=OFF, $
```

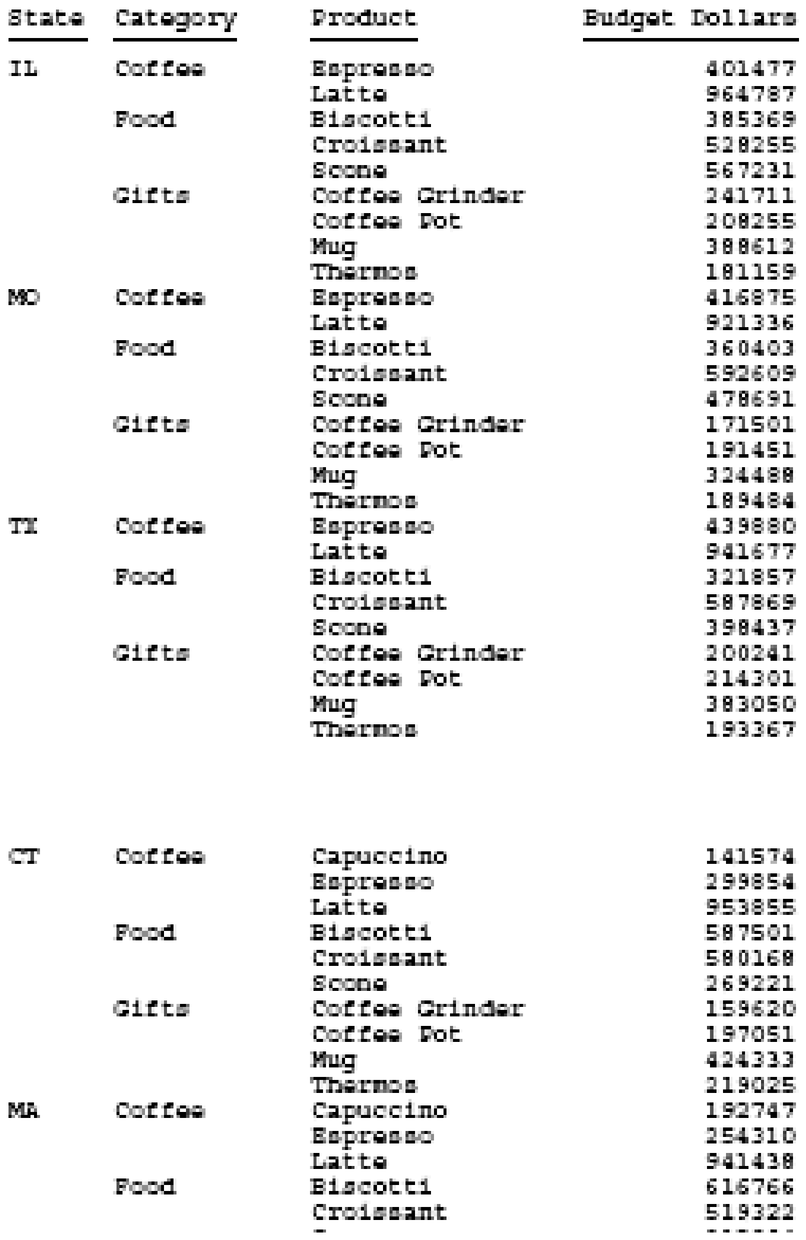
The request sets BYPANEL ON, so each panel displays the sort field values. However, since HEADPANEL=OFF for the entire report, the first panel for page 1 has the heading and the subfooting, but the second panel does not.

The output for page 1 panel 1 has the heading and subfooting, as shown in the following image. Note that with HEADPANEL=OFF, TABPAGENO does not include the panel number.



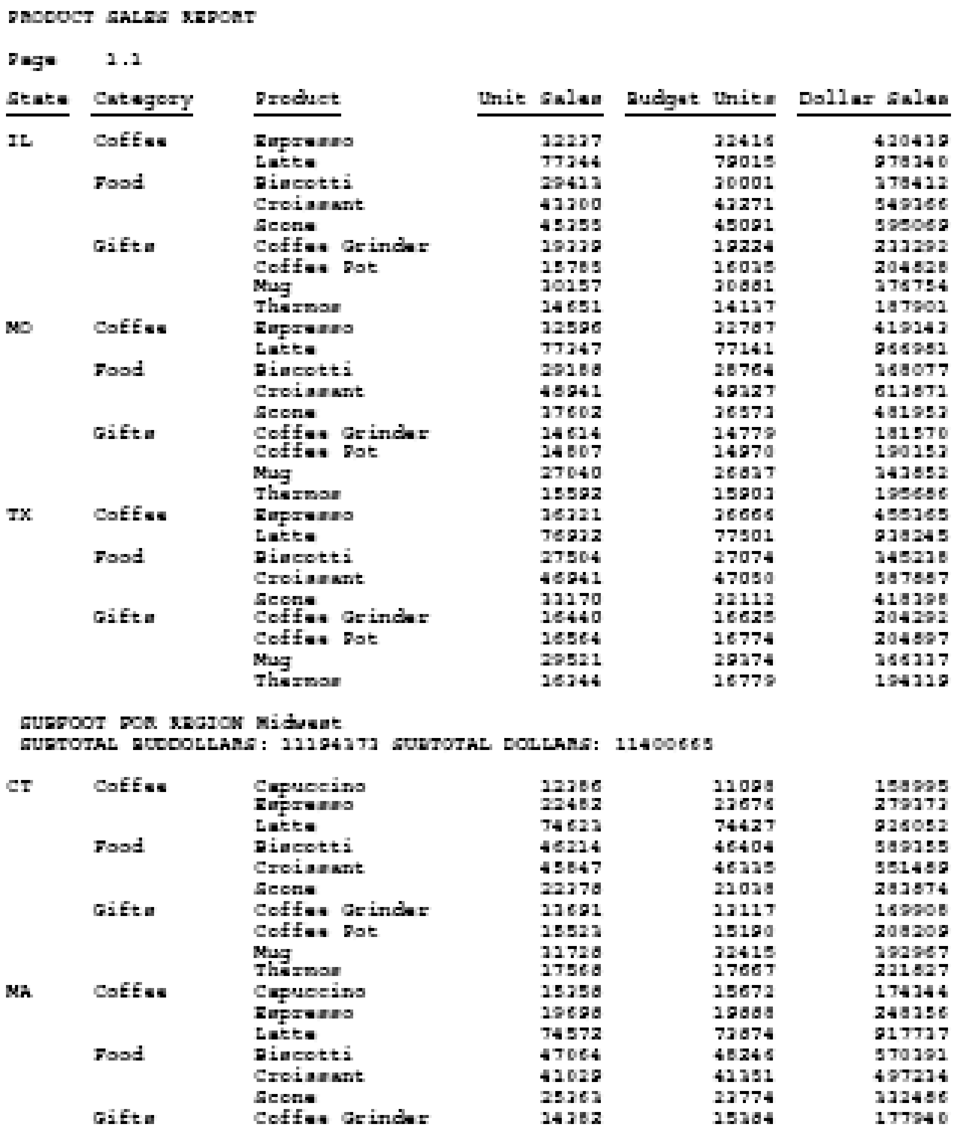
```
PRODUCT SALES REPORT

Page    1

State  Category    Product             Unit Sales  Budget Units  Dollar Sales

IL     Coffee      Espresso                 32237         32416        420439
                   Latte                    77244         79015        978340
       Food        Biscotti                 29413         30001        378412
                   Croissant                43200         42271        549366
                   Scone                    45255         45091        595069
       Gifts       Coffee Grinder           19239         19224        233292
                   Coffee Pot               15785         16035        204828
                   Mug                      30157         30881        378754
                   Thermos                  14651         14137        187901
MO     Coffee      Espresso                 32596         32787        419143
                   Latte                    77247         77141        966981
       Food        Biscotti                 29188         28764        368077
                   Croissant                48941         49327        613871
                   Scone                    37602         36573        481953
       Gifts       Coffee Grinder           14614         14779        181570
                   Coffee Pot               14807         14970        190153
                   Mug                      27040         26837        343852
                   Thermos                  15592         15903        195686
TX     Coffee      Espresso                 36221         36666        455365
                   Latte                    76932         77501        938245
       Food        Biscotti                 27504         27074        345218
                   Croissant                46941         47050        587887
                   Scone                    33170         32112        418398
       Gifts       Coffee Grinder           16440         16625        204292
                   Coffee Pot               16564         16774        204897
                   Mug                      29521         29374        366337
                   Thermos                  16344         16779        194319

 SUBFOOT FOR REGION Midwest
 SUBTOTAL BUDDOLLARS: 11194373 SUBTOTAL DOLLARS: 11400665

CT     Coffee      Capuccino                12286         11098        158995
                   Espresso                 22482         22676        279372
                   Latte                    74623         74427        926052
       Food        Biscotti                 46214         46404        589155
                   Croissant                45847         46335        551489
                   Scone                    22378         21036        283874
       Gifts       Coffee Grinder           13691         13117        169908
                   Coffee Pot               15523         15190        208209
                   Mug                      31728         32415        392967
                   Thermos                  17568         17667        221827
MA     Coffee      Capuccino                15358         15672        174344
                   Espresso                 19698         19888        248356
                   Latte                    74572         73874        917737
       Food        Biscotti                 47064         48246        570391
                   Croissant                41029         41351        497234
                   Scone                    25363         23774        332486
       Gifts       Coffee Grinder           14382         15384        177940
```

The output for page 1 panel 2 does not have the heading or subfooting, as shown in the following image.

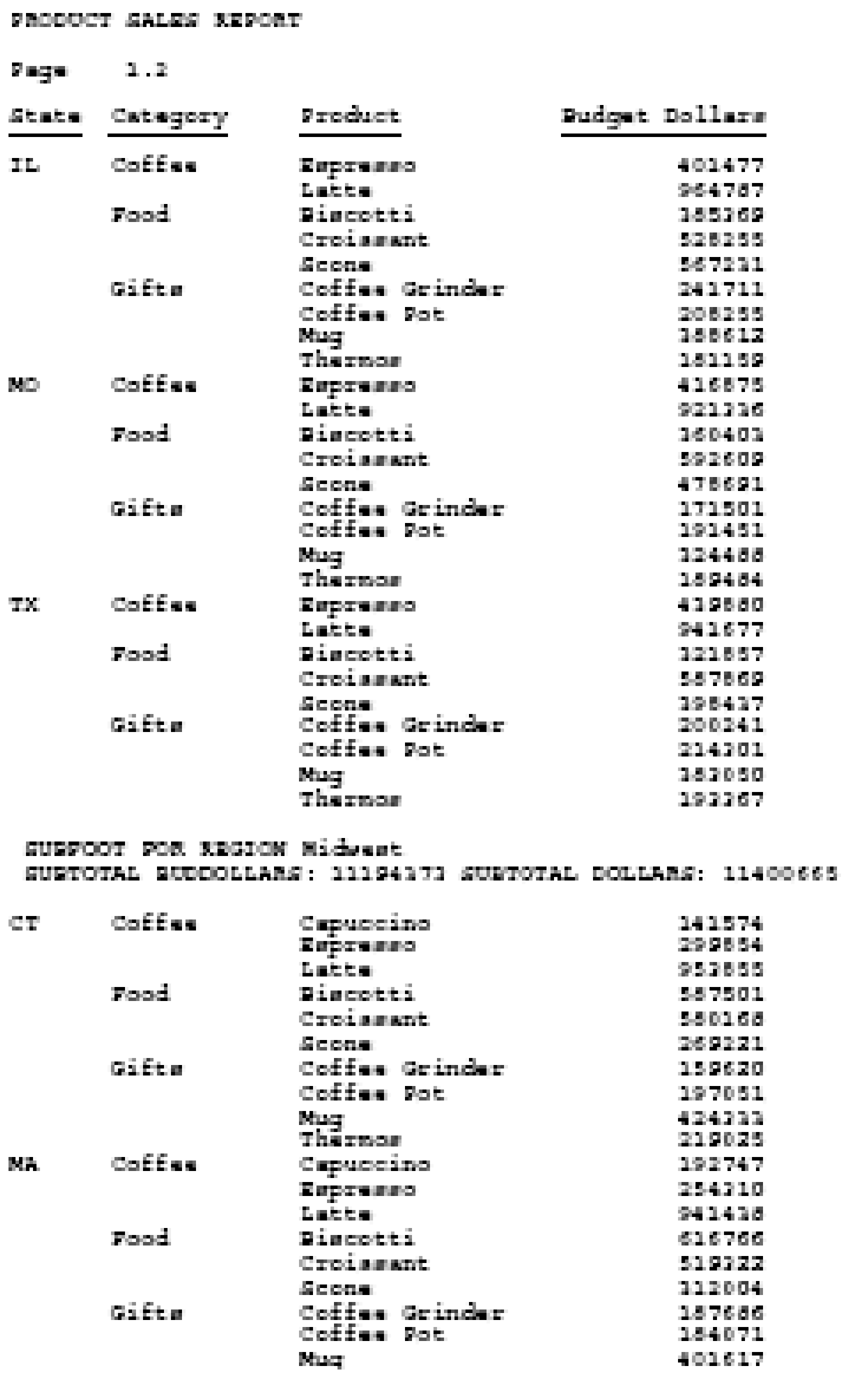


```
State  Category     Product            Budget Dollars

IL     Coffee       Espresso                   401477
                    Latte                      964787
       Food         Biscotti                   385369
                    Croissant                  528255
                    Scone                      567231
       Gifts        Coffee Grinder             241711
                    Coffee Pot                 208255
                    Mug                        388612
                    Thermos                    181159
MO     Coffee       Espresso                   416875
                    Latte                      921336
       Food         Biscotti                   360403
                    Croissant                  592609
                    Scone                      478691
       Gifts        Coffee Grinder             171501
                    Coffee Pot                 191451
                    Mug                        324488
                    Thermos                    189484
TX     Coffee       Espresso                   439880
                    Latte                      941677
       Food         Biscotti                   321857
                    Croissant                  587869
                    Scone                      398437
       Gifts        Coffee Grinder             200241
                    Coffee Pot                 214301
                    Mug                        383050
                    Thermos                    193367



CT     Coffee       Capuccino                  141574
                    Espresso                   299854
                    Latte                      953855
       Food         Biscotti                   587501
                    Croissant                  580168
                    Scone                      269221
       Gifts        Coffee Grinder             159620
                    Coffee Pot                 197051
                    Mug                        424333
                    Thermos                    219025
MA     Coffee       Capuccino                  192747
                    Espresso                   254310
                    Latte                      941438
       Food         Biscotti                   616766
                    Croissant                  519322
```

The following output shows panels 1 and 2 if the StyleSheet declaration is changed to set HEADPANEL=ON for the entire report (TYPE=REPORT, HEADPANEL=ON, $). The heading and subfooting are repeated on each panel. With HEADPANEL=ON, TABPAGENO includes the panel number.



```
PRODUCT SALES REPORT

Page    1.1

State  Category     Product             Unit Sales  Budget Units  Dollar Sales

IL     Coffee       Espresso                 32237         32416        420439
                    Latte                    77244         79015        978340
       Food         Biscotti                 29413         30001        378412
                    Croissant                43300         43271        549366
                    Scone                    45255         45091        595069
       Gifts        Coffee Grinder           19339         19224        233292
                    Coffee Pot               15785         16035        204828
                    Mug                      30157         30881        376754
                    Thermos                  14651         14137        187901
MO     Coffee       Espresso                 32596         32787        419143
                    Latte                    77247         77141        966981
       Food         Biscotti                 29188         28764        368077
                    Croissant                48941         49127        613871
                    Scone                    37602         36573        481953
       Gifts        Coffee Grinder           14614         14779        181570
                    Coffee Pot               14807         14970        190153
                    Mug                      27040         26837        343852
                    Thermos                  15592         15903        195686
TX     Coffee       Espresso                 36221         36666        455365
                    Latte                    76932         77501        938245
       Food         Biscotti                 27504         27074        345218
                    Croissant                46941         47050        587887
                    Scone                    33170         32112        418398
       Gifts        Coffee Grinder           16440         16625        204292
                    Coffee Pot               16564         16774        204897
                    Mug                      29521         29374        366337
                    Thermos                  16344         16779        194319

  SUBFOOT FOR REGION Midwest
  SUBTOTAL BUDDOLLARS: 11194373 SUBTOTAL DOLLARS: 11400665

CT     Coffee       Capuccino                12386         11098        158995
                    Espresso                 22482         23676        279373
                    Latte                    74623         74427        926052
       Food         Biscotti                 46214         46404        589355
                    Croissant                45847         46335        551489
                    Scone                    22378         21038        283874
       Gifts        Coffee Grinder           13691         13117        169908
                    Coffee Pot               15523         15190        208209
                    Mug                      31728         32415        392967
                    Thermos                  17568         17667        221827
MA     Coffee       Capuccino                15358         15672        174344
                    Espresso                 19698         19888        248356
                    Latte                    74572         73874        917737
       Food         Biscotti                 47064         48246        570391
                    Croissant                41029         41351        497234
                    Scone                    25363         23774        332486
       Gifts        Coffee Grinder           14382         15384        177940
```



```
PRODUCT SALES REPORT

Page     1.2

State  Category      Product              Budget Dollars

IL     Coffee        Espresso                     401477
                     Latte                        964787
       Food          Biscotti                     385269
                     Croissant                    528255
                     Scone                        567231
       Gifts         Coffee Grinder               241711
                     Coffee Pot                   208255
                     Mug                          368612
                     Thermos                      181159
MO     Coffee        Espresso                     416875
                     Latte                        921336
       Food          Biscotti                     360403
                     Croissant                    592609
                     Scone                        478691
       Gifts         Coffee Grinder               171501
                     Coffee Pot                   191451
                     Mug                          324488
                     Thermos                      189484
TX     Coffee        Espresso                     419880
                     Latte                        941677
       Food          Biscotti                     321857
                     Croissant                    587869
                     Scone                        398437
       Gifts         Coffee Grinder               200241
                     Coffee Pot                   214301
                     Mug                          382050
                     Thermos                      193367

 SUBFOOT FOR REGION Midwest
 SUBTOTAL BUDDOLLARS: 11194373 SUBTOTAL DOLLARS: 11400665

CT     Coffee        Capuccino                    141574
                     Espresso                     299854
                     Latte                        953855
       Food          Biscotti                     587501
                     Croissant                    580168
                     Scone                        269221
       Gifts         Coffee Grinder               159620
                     Coffee Pot                   197051
                     Mug                          424333
                     Thermos                      219025
MA     Coffee        Capuccino                    192747
                     Espresso                     254310
                     Latte                        941438
       Food          Biscotti                     616766
                     Croissant                    519222
                     Scone                        312004
       Gifts         Coffee Grinder               187686
                     Coffee Pot                   184071
                     Mug                          401617
```

Example: Repeating a Subfoot on Panels in PDF Report Output

The following request against the GGSALES data source specifies the HEADPANEL=ON attribute only for the subfoot, not for the entire report.

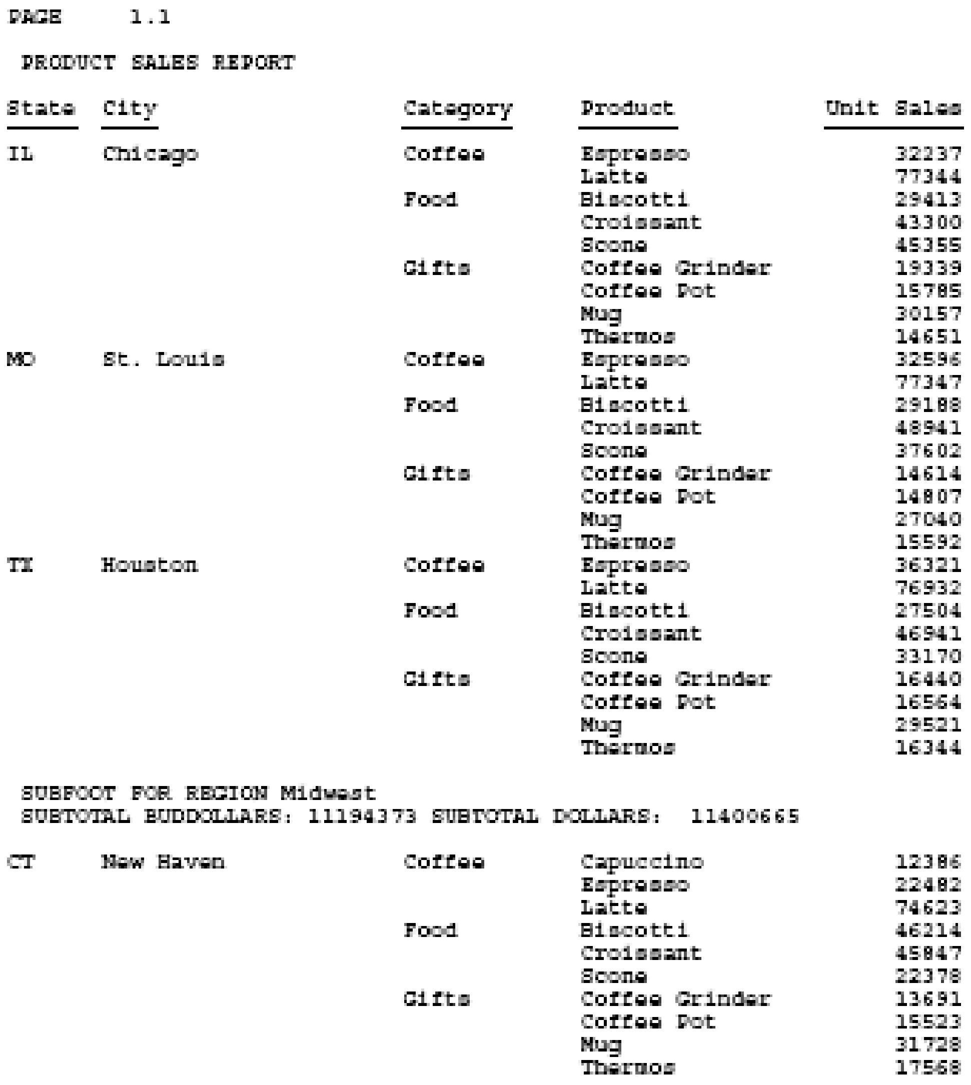
Report Request

```
TABLE FILE GGSALES
HEADING
" PRODUCT SALES REPORT"
" "
SUM UNITS BUDUNITS DOLLARS BUDDOLLARS
BY REGION NOPRINT
BY ST BY CITY  BY CATEGORY BY PRODUCT
ON REGION SUBFOOT
" "
" SUBFOOT FOR REGION <REGION "
" SUBTOTAL BUDDOLLARS: <ST.BUDDOLLARS SUBTOTAL DOLLARS:  <ST.DOLLARS "
" "
ON TABLE SET BYPANEL ON
ON TABLE PCHOLD FORMAT PDF
ON TABLE SET STYLE *
INCLUDE=IBFS:/stylesheet_location/custom_stylesheet.sty,$
END
```

***custom_stylesheet*.sty**

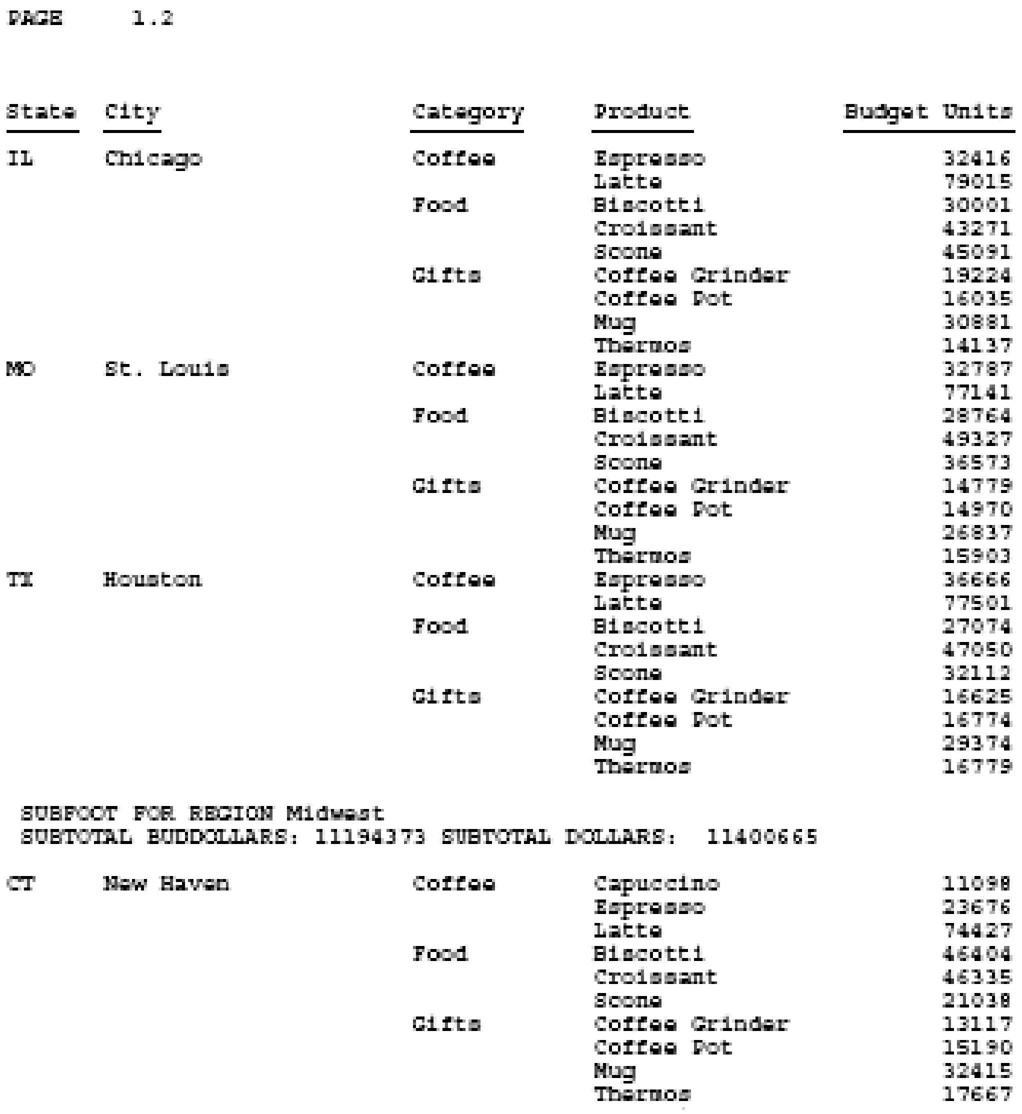
```
TYPE=SUBFOOT, HEADPANEL=ON, $
```

Panel 1 displays both the heading and the subfooting, as shown in the following image.



```
PAGE     1.1

 PRODUCT SALES REPORT

State  City                  Category     Product            Unit Sales

IL     Chicago               Coffee       Espresso                32237
                                          Latte                   77344
                             Food         Biscotti                29413
                                          Croissant               43300
                                          Scone                   45355
                             Gifts        Coffee Grinder          19339
                                          Coffee Pot              15785
                                          Mug                     30157
                                          Thermos                 14651
MO     St. Louis             Coffee       Espresso                32596
                                          Latte                   77347
                             Food         Biscotti                29188
                                          Croissant               48941
                                          Scone                   37602
                             Gifts        Coffee Grinder          14614
                                          Coffee Pot              14807
                                          Mug                     27040
                                          Thermos                 15592
TX     Houston               Coffee       Espresso                36321
                                          Latte                   76932
                             Food         Biscotti                27504
                                          Croissant               46941
                                          Scone                   33170
                             Gifts        Coffee Grinder          16440
                                          Coffee Pot              16564
                                          Mug                     29521
                                          Thermos                 16344

 SUBFOOT FOR REGION Midwest
 SUBTOTAL BUDDOLLARS: 11194373 SUBTOTAL DOLLARS:  11400665

CT     New Haven             Coffee       Capuccino               12386
                                          Espresso                22482
                                          Latte                   74623
                             Food         Biscotti                46214
                                          Croissant               45847
                                          Scone                   22378
                             Gifts        Coffee Grinder          13691
                                          Coffee Pot              15523
                                          Mug                     31728
                                          Thermos                 17568
```

Panel 2 displays only the subfooting, not the heading, as shown in the following image.



```
PAGE     1.2


State  City                 Category    Product           Budget Units

IL     Chicago              Coffee      Espresso                 32416
                                        Latte                    79015
                            Food        Biscotti                 30001
                                        Croissant                43271
                                        Scone                    45091
                            Gifts       Coffee Grinder           19224
                                        Coffee Pot               16035
                                        Mug                      30881
                                        Thermos                  14137
MO     St. Louis            Coffee      Espresso                 32787
                                        Latte                    77141
                            Food        Biscotti                 28764
                                        Croissant                49327
                                        Scone                    36573
                            Gifts       Coffee Grinder           14779
                                        Coffee Pot               14970
                                        Mug                      26837
                                        Thermos                  15903
TX     Houston              Coffee      Espresso                 36666
                                        Latte                    77501
                            Food        Biscotti                 27074
                                        Croissant                47050
                                        Scone                    32112
                            Gifts       Coffee Grinder           16625
                                        Coffee Pot               16774
                                        Mug                      29374
                                        Thermos                  16779

 SUBFOOT FOR REGION Midwest
 SUBTOTAL BUDDOLLARS: 11194373 SUBTOTAL DOLLARS:  11400665

CT     New Haven            Coffee      Capuccino                11098
                                        Espresso                 23676
                                        Latte                    74427
                            Food        Biscotti                 46404
                                        Croissant                46335
                                        Scone                    21038
                            Gifts       Coffee Grinder           13117
                                        Coffee Pot               15190
                                        Mug                      32415
                                        Thermos                  17667
```

Since the page heading is not repeated, if you use the <TABPAGENO system variable to place the page number in the heading, it will not display the panel number and will not display on the second panel.

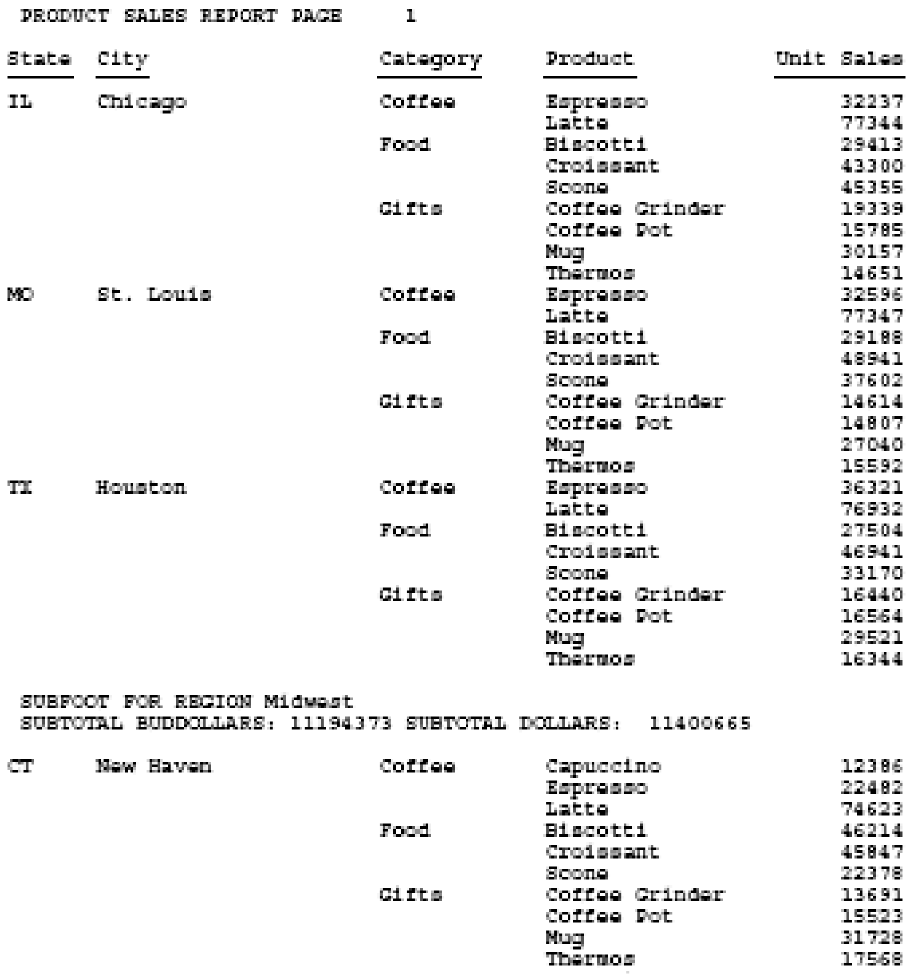
Report Request

```
TABLE FILE GGSALES
HEADING
" PRODUCT SALES REPORT PAGE <TABPAGENO"
" "
SUM UNITS BUDUNITS DOLLARS BUDDOLLARS
BY REGION NOPRINT
BY ST BY CITY  BY CATEGORY BY PRODUCT
ON REGION SUBFOOT
" "
" SUBFOOT FOR REGION <REGION "
" SUBTOTAL BUDDOLLARS: <ST.BUDDOLLARS SUBTOTAL DOLLARS:  <ST.DOLLARS "
" "
ON TABLE SET BYPANEL ON
ON TABLE PCHOLD FORMAT PDF
ON TABLE SET STYLE *
INCLUDE=IBFS:/stylesheet_location/custom_stylesheet.sty,$
END
```

***custom_stylesheet*.sty**

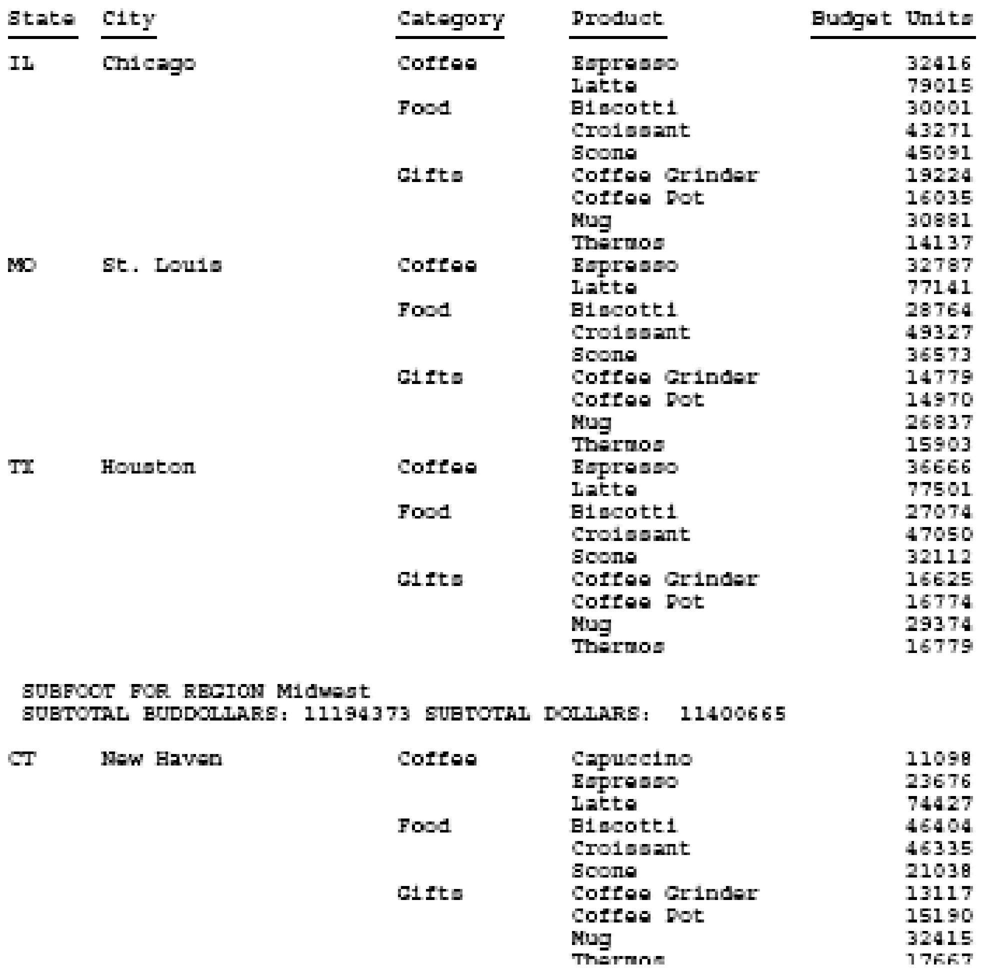
```
TYPE=REPORT, GRID=OFF, $
TYPE=SUBFOOT, HEADPANEL=ON, $
```

The first panel displays the page number in the heading, without the panel number, as shown in the following image.



```
 PRODUCT SALES REPORT PAGE     1

State  City                   Category     Product           Unit Sales

IL     Chicago                Coffee       Espresso               32237
                                           Latte                  77344
                              Food         Biscotti               29413
                                           Croissant              43300
                                           Scone                  45355
                              Gifts        Coffee Grinder         19339
                                           Coffee Pot             15785
                                           Mug                    30157
                                           Thermos                14651
MO     St. Louis              Coffee       Espresso               32596
                                           Latte                  77347
                              Food         Biscotti               29188
                                           Croissant              48941
                                           Scone                  37602
                              Gifts        Coffee Grinder         14614
                                           Coffee Pot             14807
                                           Mug                    27040
                                           Thermos                15592
TX     Houston                Coffee       Espresso               36321
                                           Latte                  76932
                              Food         Biscotti               27504
                                           Croissant              46941
                                           Scone                  33170
                              Gifts        Coffee Grinder         16440
                                           Coffee Pot             16564
                                           Mug                    29521
                                           Thermos                16344

 SUBFOOT FOR REGION Midwest
 SUBTOTAL BUDDOLLARS: 11194373 SUBTOTAL DOLLARS:  11400665

CT     New Haven              Coffee       Capuccino              12386
                                           Espresso               22482
                                           Latte                  74623
                              Food         Biscotti               46214
                                           Croissant              45847
                                           Scone                  22378
                              Gifts        Coffee Grinder         13691
                                           Coffee Pot             15523
                                           Mug                    31728
                                           Thermos                17568
```

The second panel does not display the heading and therefore, does not display the embedded page number, as shown in the following image.



```
State  City                   Category    Product            Budget Units

IL     Chicago                Coffee      Espresso                  32416
                                          Latte                     79015
                              Food        Biscotti                  30001
                                          Croissant                 43271
                                          Scone                     45091
                              Gifts       Coffee Grinder            19224
                                          Coffee Pot                16035
                                          Mug                       30881
                                          Thermos                   14137
MO     St. Louis              Coffee      Espresso                  32787
                                          Latte                     77141
                              Food        Biscotti                  28764
                                          Croissant                 49327
                                          Scone                     36573
                              Gifts       Coffee Grinder            14779
                                          Coffee Pot                14970
                                          Mug                       26837
                                          Thermos                   15903
TX     Houston                Coffee      Espresso                  36666
                                          Latte                     77501
                              Food        Biscotti                  27074
                                          Croissant                 47050
                                          Scone                     32112
                              Gifts       Coffee Grinder            16625
                                          Coffee Pot                16774
                                          Mug                       29374
                                          Thermos                   16779

 SUBFOOT FOR REGION Midwest
 SUBTOTAL BUDDOLLARS: 11194373 SUBTOTAL DOLLARS:  11400665

CT     New Haven              Coffee      Capuccino                 11098
                                          Espresso                  23676
                                          Latte                     74427
                              Food        Biscotti                  46404
                                          Croissant                 46335
                                          Scone                     21038
                              Gifts       Coffee Grinder            13117
                                          Coffee Pot                15190
                                          Mug                       32415
                                          Thermos                   17667
```

Example: Repeating Styled Headings and Footings on Paneled Pages

The following request against the GGSALES data source has a report heading, a page heading with an image, a footing, a subheading, a subfooting, and a subtotal.

Report Request

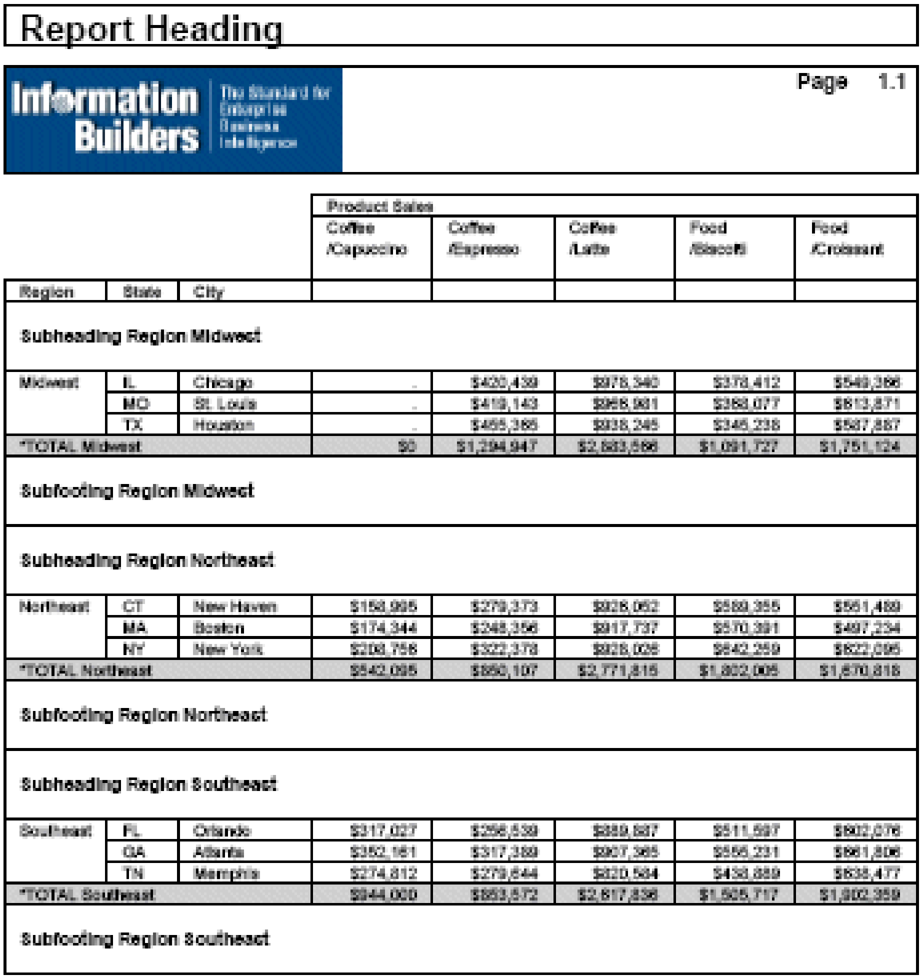
```
SET BYPANEL=ON
DEFINE FILE GGSALES
SHOWCATPROD/A30 = CATEGORY || ' / ' || PRODUCT;
END
TABLE FILE GGSALES
SUM
     DOLLARS/I8M AS ''
BY REGION
BY ST
BY CITY
ACROSS SHOWCATPROD AS 'Product Sales'

ON REGION SUBHEAD
" "
"Subheading Region <REGION"
" "
ON REGION SUBTOTAL AS '*TOTAL'
ON REGION SUBFOOT WITHIN
" "
"Subfooting Region <REGION"
" "
ON TABLE SUBHEAD
"Report Heading"
HEADING
"Page <TABPAGENO  "
" "
" "
" "
FOOTING
" "
"PAGE FOOTING "
ON TABLE SUBFOOT
" "
"Report Footing"
ON TABLE PCHOLD FORMAT PDF
ON TABLE SET STYLE *
INCLUDE=IBFS:/stylesheet_location/custom_stylesheet.sty,$
END
```

***custom_stylesheet*.sty**

```
TYPE=REPORT, UNITS=IN, SQUEEZE=ON, ORIENTATION=PORTRAIT, $
TYPE=REPORT, FONT='ARIAL', SIZE=9, HEADPANEL=ON, BORDER=ON, $
TYPE=TITLE, STYLE=BOLD,$
TYPE=TABHEADING, SIZE=20, STYLE=BOLD, $
TYPE=TABFOOTING, SIZE=20, STYLE=BOLD, $
TYPE=HEADING, SIZE=12, STYLE=BOLD, $
TYPE=HEADING, LINE=1, JUSTIFY=RIGHT, $
TYPE=HEADING, LINE=2, JUSTIFY=RIGHT, $
TYPE=HEADING, LINE=3, JUSTIFY=RIGHT, $
TYPE=HEADING, LINE=4, JUSTIFY=RIGHT, $
TYPE=HEADING, LINE=5, JUSTIFY=RIGHT, $
TYPE=HEADING, IMAGE=smplogo1.gif, POSITION=(+0.000000 +0.000000), $
TYPE=FOOTING, SIZE=12, STYLE=BOLD, JUSTIFY=RIGHT, $
TYPE=SUBHEAD, SIZE=10, STYLE=BOLD, $
TYPE=SUBFOOT, SIZE=10, STYLE=BOLD, $
TYPE=SUBTOTAL, BACKCOLOR=RGB(210 210 210), $
TYPE=ACROSSVALUE, SIZE=9, WRAP=ON, $
TYPE=ACROSSTITLE, STYLE=BOLD, $
TYPE=GRANDTOTAL, BACKCOLOR=RGB(210 210 210), STYLE=BOLD, $
```

Since HEADPANEL=ON for the entire report, both panels display all of the heading and footing elements.

The following image shows page 1 panel 1.



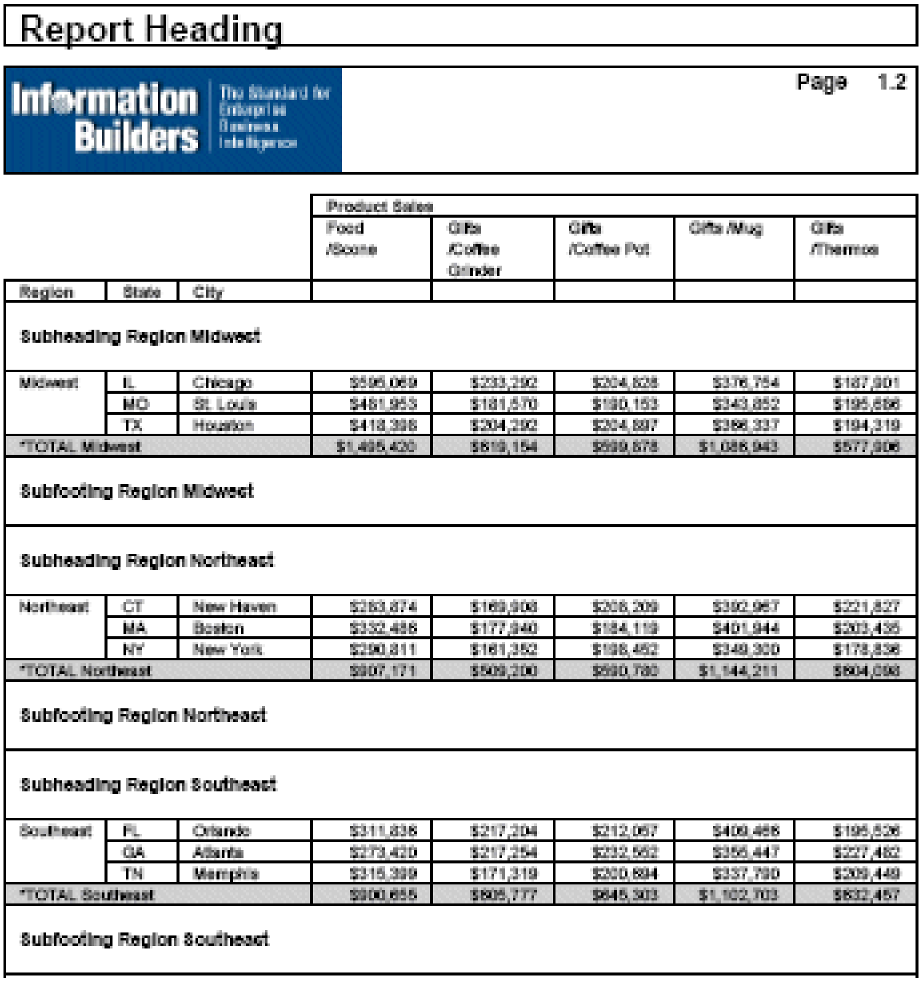
Report Heading

Information Builders — The Standard for Enterprise Business Intelligence

Page 1.1

			Product Sales				
			Coffee /Capuccino	Coffee /Espresso	Coffee /Latte	Food /Biscotti	Food /Croissant
Region	State	City					
Subheading Region Midwest							
Midwest	IL	Chicago	.	$420,439	$978,340	$378,412	$549,366
	MO	St. Louis	.	$419,143	$966,981	$368,077	$613,871
	TX	Houston	.	$455,365	$938,245	$345,238	$587,887
*TOTAL Midwest			$0	$1,294,947	$2,883,566	$1,091,727	$1,751,124
Subfooting Region Midwest							
Subheading Region Northeast							
Northeast	CT	New Haven	$158,995	$279,373	$926,052	$589,355	$551,489
	MA	Boston	$174,344	$248,356	$917,737	$570,391	$487,234
	NY	New York	$208,756	$322,378	$928,026	$642,259	$622,095
*TOTAL Northeast			$542,095	$850,107	$2,771,815	$1,802,005	$1,670,818
Subfooting Region Northeast							
Subheading Region Southeast							
Southeast	FL	Orlando	$317,027	$256,539	$889,887	$511,597	$602,076
	GA	Atlanta	$352,161	$317,389	$907,365	$556,231	$661,806
	TN	Memphis	$274,812	$279,644	$820,584	$438,889	$638,477
*TOTAL Southeast			$944,000	$853,572	$2,617,836	$1,505,717	$1,902,359
Subfooting Region Southeast							

The following image shows page 1 panel 2.



Report Heading

Page 1.2

			Product Sales				
			Food /Scone	Gifts /Coffee Grinder	Gifts /Coffee Pot	Gifts /Mug	Gifts /Thermos
Region	State	City					
Subheading Region Midwest							
Midwest	IL	Chicago	$595,069	$233,292	$204,828	$376,754	$187,901
	MO	St. Louis	$481,953	$181,570	$190,153	$343,852	$195,696
	TX	Houston	$418,398	$204,292	$204,897	$366,337	$194,319
*TOTAL Midwest			$1,495,420	$619,154	$599,878	$1,086,943	$577,906
Subfooting Region Midwest							
Subheading Region Northeast							
Northeast	CT	New Haven	$283,874	$169,908	$208,209	$392,967	$221,827
	MA	Boston	$332,486	$177,940	$184,119	$401,944	$203,435
	NY	New York	$290,811	$161,352	$198,452	$349,300	$178,836
*TOTAL Northeast			$907,171	$509,200	$590,780	$1,144,211	$604,098
Subfooting Region Northeast							
Subheading Region Southeast							
Southeast	FL	Orlando	$311,836	$217,204	$212,057	$409,488	$195,526
	GA	Atlanta	$273,420	$217,294	$232,552	$356,447	$227,482
	TN	Memphis	$315,399	$171,319	$200,894	$337,790	$209,449
*TOTAL Southeast			$900,655	$605,777	$645,303	$1,102,703	$632,457
Subfooting Region Southeast							

Controlling Column Title Underlining Using a StyleSheet Attribute

The TITLELINE attribute allows you to control whether column titles are underlined for report output.

Syntax: **How to Control Column Title Underlining Using a StyleSheet Attribute**

```
TYPE={REPORT|TITLE}, TITLELINE=(ON|OFF|SKIP), $
```

where:

ON

Underlines column titles. ON is the default value.

OFF

Replaces the underline with a blank line.

SKIP

Omits both the underline and the line on which the underline would have displayed.

Example: Controlling Column Title Underlining Using a StyleSheet Attribute

The following example illustrates how to include the TITLELINE attribute and a BY and an ACROSS field.

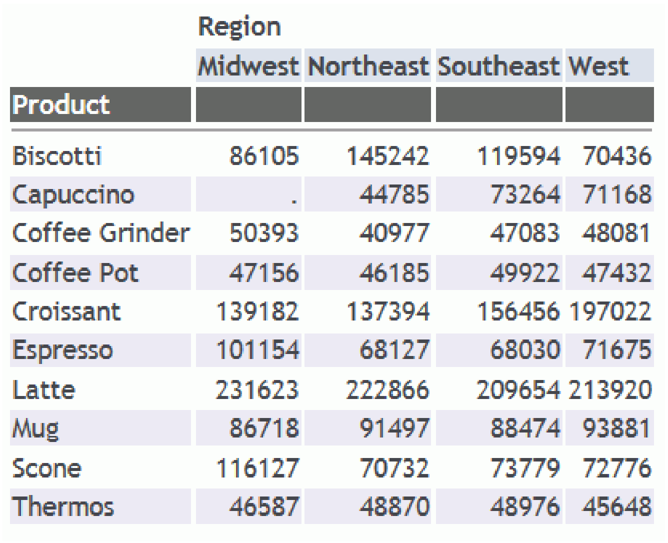
Report Request

```
TABLE FILE GGSALES
SUM UNITS BY PRODUCT
ACROSS REGION
ON TABLE SET STYLE *
INCLUDE=IBFS:/stylesheet_location/custom_stylesheet.sty,$
END
```

***custom_stylesheet*.sty**

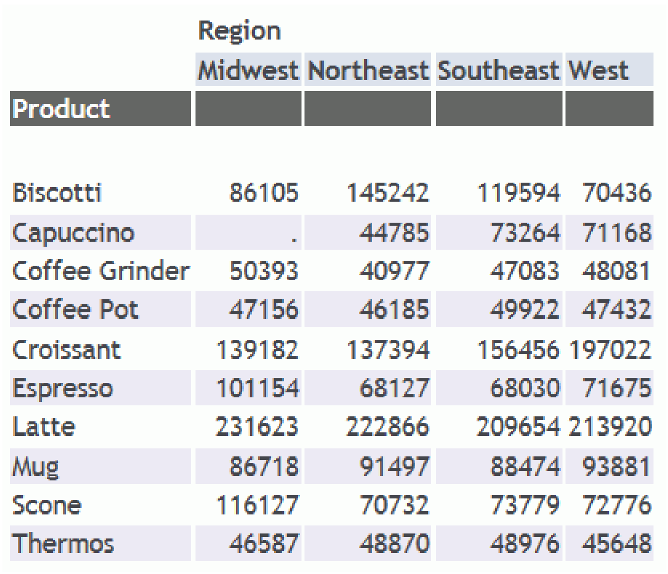
```
TYPE=REPORT, TITLELINE=ON, GRID=OFF, FONT=ARIAL, $
```

With the default value (ON) for TITLELINE, the column titles are underlined.



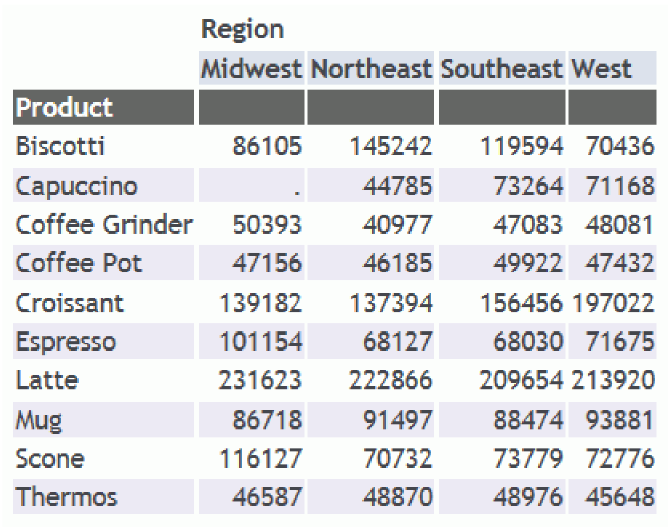
	Region			
	Midwest	Northeast	Southeast	West
Product				
Biscotti	86105	145242	119594	70436
Capuccino	.	44785	73264	71168
Coffee Grinder	50393	40977	47083	48081
Coffee Pot	47156	46185	49922	47432
Croissant	139182	137394	156456	197022
Espresso	101154	68127	68030	71675
Latte	231623	222866	209654	213920
Mug	86718	91497	88474	93881
Scone	116127	70732	73779	72776
Thermos	46587	48870	48976	45648

With TITLELINE=OFF, the column titles are not underlined, but the blank line where the underlines would have been is still there.



	Region			
	Midwest	Northeast	Southeast	West
Product				
Biscotti	86105	145242	119594	70436
Capuccino	.	44785	73264	71168
Coffee Grinder	50393	40977	47083	48081
Coffee Pot	47156	46185	49922	47432
Croissant	139182	137394	156456	197022
Espresso	101154	68127	68030	71675
Latte	231623	222866	209654	213920
Mug	86718	91497	88474	93881
Scone	116127	70732	73779	72776
Thermos	46587	48870	48976	45648

With TITLELINE=SKIP, both the underlines and the blank line are removed.



	Region			
	Midwest	Northeast	Southeast	West
Product				
Biscotti	86105	145242	119594	70436
Capuccino	.	44785	73264	71168
Coffee Grinder	50393	40977	47083	48081
Coffee Pot	47156	46185	49922	47432
Croissant	139182	137394	156456	197022
Espresso	101154	68127	68030	71675
Latte	231623	222866	209654	213920
Mug	86718	91497	88474	93881
Scone	116127	70732	73779	72776
Thermos	46587	48870	48976	45648

Formatting a Heading, Footing, Title, or Label

You can use a variety of strategies for enhancing the context-building elements in a report, that is, the headings, footings, column and row titles, and labels you assign to row and column totals and to subtotals. These additions enhance the visual appeal of a report and communicate a sense of completeness, using font and color for emphasis and distinctions, and alignment to add structural clarity and facilitate comprehension.

You can:

- Apply font characteristics to column and row titles, to headings, footings, and elements in them, and to subtotals, grand totals, and subtotal calculations. For more information, see *Applying Font Attributes to a Heading, Footing, Title, or Label* on page 255.
- Add borders around headings and footings and grid lines around headings, footings, and column titles. For more information, see *Laying Out the Report Page* on page 123.

  You can also add space between heading or footing content and grid lines. For more information, see *Controlling the Vertical Positioning of a Heading or Footing* on page 273.
- Left-justify, right-justify, or center a heading, footing, or individual lines in a multi-line heading or footing, column titles, and subtotals. For more information, see *Justifying a Heading, Footing, Title, or Label* on page 263.
- Control the vertical positioning of a heading or footing. For more information, see *Controlling the Vertical Positioning of a Heading or Footing* on page 273.

Applying Font Attributes to a Heading, Footing, Title, or Label

You can specify font family, size, color, and style for any report element you can identify in a StyleSheet:

- Row and column titles. Styling is applied to either the default title or a customized title.
- Headings and footings, and elements within them, including specific lines in a multi-line heading or footing, items in a line, text strings, and embedded fields. Note that you can also specify background color for individual elements.
- Labels for subtotals, grand totals, subtotal calculations, and row totals. Styling applies to default or customized names.
- Page numbers in a heading or footing.
- Underlines and skipped lines (not supported in HTML reports).

For detailed syntax, see *Identifying a Report Component in a Web Query StyleSheet* on page 57. For details on font options, including size, color, and style, see *Formatting Report Data* on page 279.

Example: Applying Font Characteristics to a Report Heading and Column Titles

The following example illustrates how to use a StyleSheet to select 12-point Arial bold for the report heading (Sales Report), and 10-point Arial italic for the default column titles (Category, Product, Unit Sales, Dollar Sales), based on the HTML point scale, which differs from standard point sizes. See *Formatting Report Data* on page 279.

For an HTML report, the font name must be enclosed in single quotation marks ('). The StyleSheet attribute TYPE=TABHEADING identifies the report heading, and the attribute TYPE=TITLE identifies the column titles.

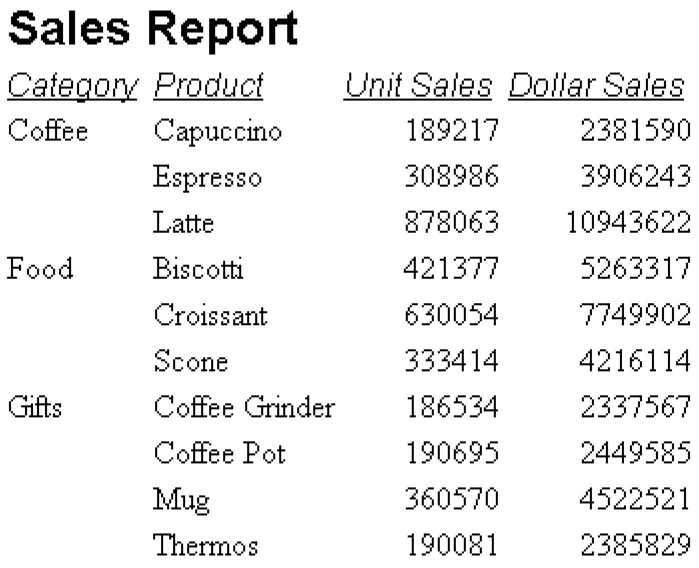
Report Request

```
TABLE FILE GGSALES
SUM UNITS DOLLARS
BY CATEGORY BY PRODUCT
ON TABLE SUBHEAD
"Sales Report"
ON TABLE SET STYLE *
INCLUDE=IBFS:/stylesheet_location/custom_stylesheet.sty,$
END
```

***custom_stylesheet*.sty**

```
TYPE=REPORT, GRID=OFF, $
TYPE=TABHEADING, FONT='ARIAL', SIZE=12, STYLE=BOLD, $
TYPE=TITLE, FONT='ARIAL', SIZE=10, STYLE=ITALIC, $
```

The output is:



Sales Report

Category	Product	Unit Sales	Dollar Sales
Coffee	Capuccino	189217	2381590
	Espresso	308986	3906243
	Latte	878063	10943622
Food	Biscotti	421377	5263317
	Croissant	630054	7749902
	Scone	333414	4216114
Gifts	Coffee Grinder	186534	2337567
	Coffee Pot	190695	2449585
	Mug	360570	4522521
	Thermos	190081	2385829

Example: Applying Font Styles to a System Variable in a Report Heading

The following example illustrates how to include the system variable &DATE in the heading. Styling is italic to distinguish it from the rest of the heading text, which is bold. The spot marker <+0> creates two items in the heading so that each one can be formatted separately.

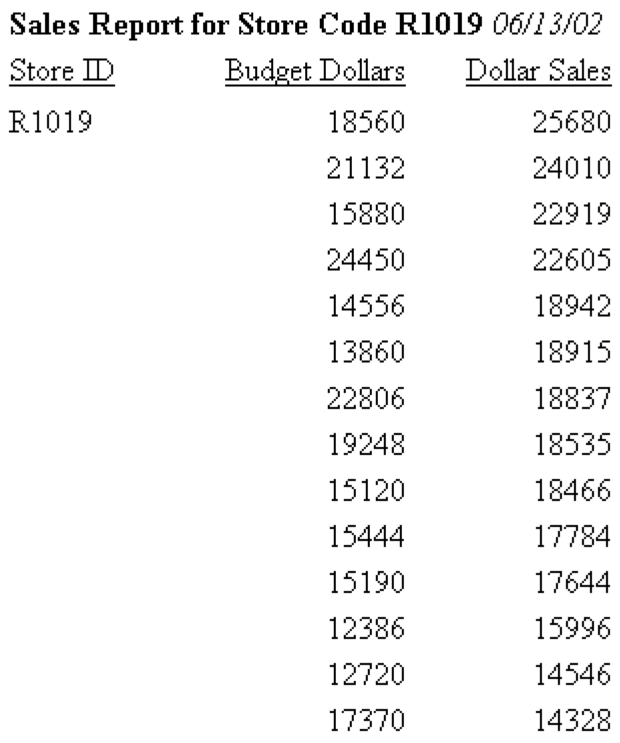
Report Request

```
TABLE FILE GGSALES
PRINT BUDDOLLARS DOLLARS
BY STCD
WHERE STCD EQ 'R1019'
ON TABLE SUBHEAD
"Sales Report for Store Code R1019 <+0>&DATE"
ON TABLE SET STYLE *
INCLUDE=IBFS:/stylesheet_location/custom_stylesheet.sty,$
END
```

***custom_stylesheet*.sty**

```
TYPE=REPORT, GRID=OFF, $
TYPE=TABHEADING, FONT='TIMES', SIZE=10, STYLE=BOLD, $
TYPE=TABHEADING, ITEM=2, STYLE=ITALIC, $
```

The partial output is:



```
Sales Report for Store Code R1019 06/13/02
Store ID        Budget Dollars    Dollar Sales

R1019                    18560           25680
                         21132           24010
                         15880           22919
                         24450           22605
                         14556           18942
                         13860           18915
                         22806           18837
                         19248           18535
                         15120           18466
                         15444           17784
                         15190           17644
                         12386           15996
                         12720           14546
                         17370           14328
```

Adding Borders and Grid Lines

You can add borders and grid lines to headings, footings, titles, and labels. For detailed syntax, see *Laying Out the Report Page* on page 123.

Example: Adding a Grid Around a Report Heading in a PDF Report

The following example illustrates how to generate a PDF report with a grid around the heading, created with the GRID attribute, to set the heading off from the body of the report.

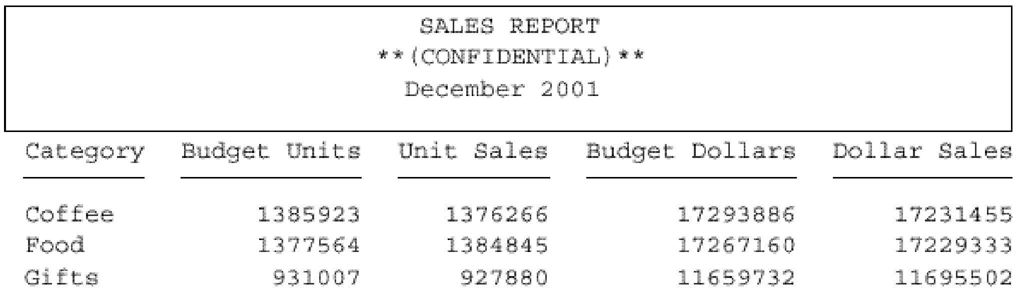
Report Request

```
TABLE FILE GGSALES
SUM BUDUNITS UNITS BUDDOLLARS DOLLARS
BY CATEGORY
ON TABLE SUBHEAD
"SALES REPORT"
"**(CONFIDENTIAL)**"
"December 2001 </1"
ON TABLE PCHOLD FORMAT PDF
ON TABLE SET STYLE *
INCLUDE=IBFS:/stylesheet_location/custom_stylesheet.sty,$
END
```

***custom_stylesheet*.sty**

```
TYPE=REPORT, SQUEEZE = ON, $
TYPE=TABHEADING, JUSTIFY=CENTER, GRID=ON, $
```

The output is:



```
                         SALES REPORT
                      **(CONFIDENTIAL)**
                        December 2001

Category  Budget Units  Unit Sales  Budget Dollars  Dollar Sales

Coffee         1385923     1376266        17293886      17231455
Food           1377564     1384845        17267160      17229333
Gifts           931007      927880        11659732      11695502
```

Example: Emphasizing Column Titles With Horizontal Lines in a PDF Report

The following example illustrates how to generate a PDF report with horizontal lines, created with the HGRID attribute, above and below the column titles.

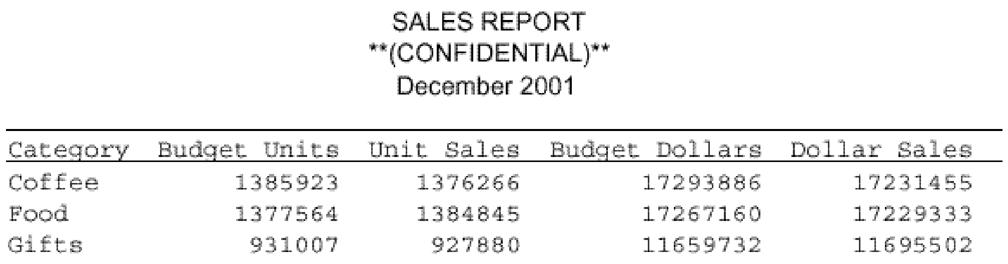
Report Request

```
TABLE FILE GGSALES
SUM BUDUNITS UNITS BUDDOLLARS DOLLARS
BY CATEGORY
ON TABLE SUBHEAD
"SALES REPORT"
"**(CONFIDENTIAL)**"
"December 2001 </1"
ON TABLE PCHOLD FORMAT PDF
ON TABLE SET STYLE *
INCLUDE=IBFS:/stylesheet_location/custom_stylesheet.sty,$
END
```

***custom_stylesheet*.sty**

```
TYPE=REPORT, SQUEEZE=ON, $
TYPE=TABHEADING, JUSTIFY=CENTER, FONT=ARIAL, SIZE=12, $
TYPE=TITLE, HGRID=ON, $
```

The output is:



```
                         SALES REPORT
                      **(CONFIDENTIAL)**
                        December 2001

Category  Budget Units  Unit Sales  Budget Dollars  Dollar Sales
Coffee         1385923     1376266        17293886      17231455
Food           1377564     1384845        17267160      17229333
Gifts           931007      927880        11659732      11695502
```

Example: Formatting a Border Around a Report Heading

The following example illustrates how to generate an HTML report with a heavy red dotted line around the entire report heading.

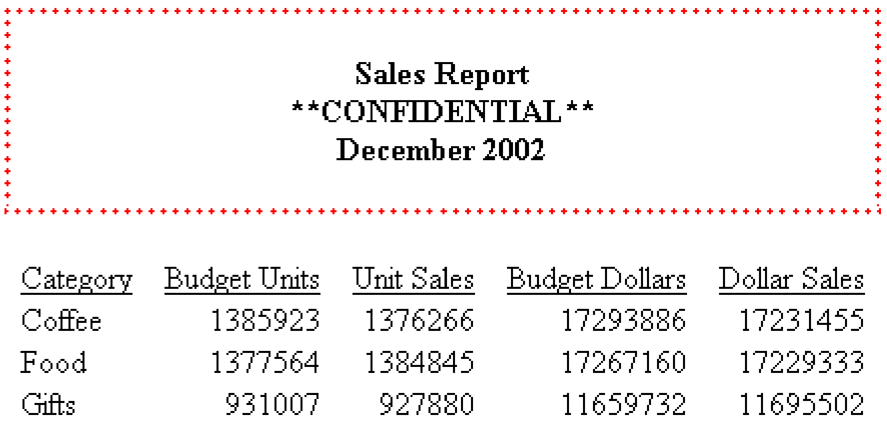
Report Request

```
TABLE FILE GGSALES
SUM BUDUNITS UNITS BUDDOLLARS DOLLARS
BY CATEGORY
ON TABLE SUBHEAD
"</1 Sales Report"
"**CONFIDENTIAL**"
"December 2002 </1"
ON TABLE SET STYLE *
INCLUDE=IBFS:/stylesheet_location/custom_stylesheet.sty,$
END
```

***custom_stylesheet*.sty**

```
TYPE=REPORT, GRID=OFF, $
TYPE=TABHEADING, STYLE=BOLD, JUSTIFY=CENTER, BORDER=HEAVY,
   BORDER-COLOR=RED, BORDER-STYLE=DOTTED, $
```

The output is:



```
                 Sales Report
               **CONFIDENTIAL**
                December 2002

Category  Budget Units  Unit Sales  Budget Dollars  Dollar Sales
Coffee         1385923     1376266        17293886      17231455
Food           1377564     1384845        17267160      17229333
Gifts           931007      927880        11659732      11695502
```

Tip: You can use the same BORDER syntax to generate this output in a PDF report.

Example: Formatting a Report Heading With Top and Bottom Borders

The following example illustrates how to generate a light blue line above the heading and a heavy double line of the same color below the heading. The request does not specify border lines for the left and right sides of the heading.

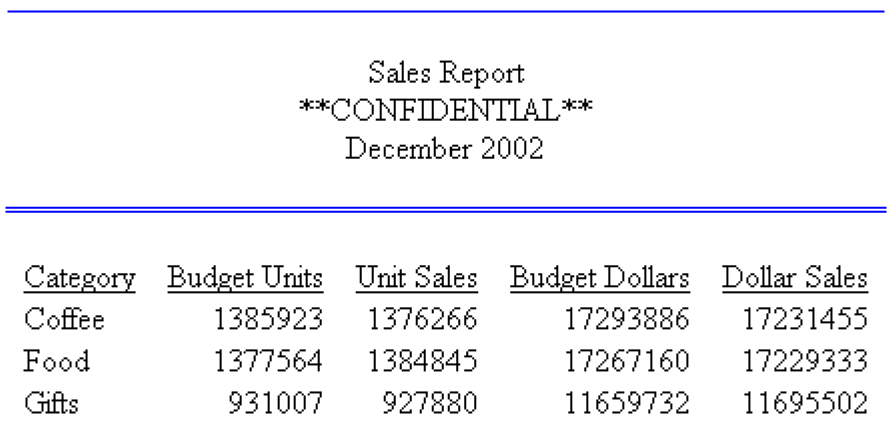
Report Request

```
TABLE FILE GGSALES
SUM BUDUNITS UNITS BUDDOLLARS DOLLARS
BY CATEGORY
ON TABLE SUBHEAD
"</1 Sales Report"
"**CONFIDENTIAL**"
"December 2002 </1"
ON TABLE SET STYLE *
INCLUDE=IBFS:/stylesheet_location/custom_stylesheet.sty,$
END
```

***custom_stylesheet*.sty**

```
TYPE=REPORT, GRID=OFF, $
TYPE=TABHEADING, JUSTIFY=CENTER, BORDER-TOP=LIGHT, BORDER-COLOR=BLUE,
   BORDER-BOTTOM=HEAVY, BORDER-BOTTOM-STYLE=DOUBLE, $
```

The output is:



Sales Report
CONFIDENTIAL
December 2002

Category	Budget Units	Unit Sales	Budget Dollars	Dollar Sales
Coffee	1385923	1376266	17293886	17231455
Food	1377564	1384845	17267160	17229333
Gifts	931007	927880	11659732	11695502

Tip: You can use the same BORDER syntax to generate this output in a PDF report.

Justifying a Heading, Footing, Title, or Label

You can left-justify, right-justify, or center the following report elements:

- A heading or footing. See *Justifying a Heading or Footing* on page 263.
- A column title. See *Justifying a Column Title* on page 267.
- A label for a subtotal or grand total. See *Justifying a Label for a Subtotal or Grand Total* on page 272.

Justifying a Heading or Footing

You can left-justify, right-justify, or center a heading or footing in a StyleSheet. By default, a heading or footing is left justified. In addition, you can justify an individual line or lines in a multiple-line heading or footing.

Justification behavior in HTML and PDF. For HTML reports, justification is implemented with respect to the report width. That means a centered heading is centered over the report content. In contrast, for PDF reports, the default justification area is the page width, rather than the report width, resulting in headings and footings that are not centered on the report. In most cases, you can achieve justification based on report width in a PDF report by adding SQUEEZE=ON to your request. This command improves the appearance of the report by eliminating excessive white space between columns and implements justification over the report content. However, if the heading is wider than the report, it will be centered on the page, even when SQUEEZE=ON.

Syntax: How to Justify a Heading or Footing in a StyleSheet

```
TYPE=headfoot, [LINE=line_#,] JUSTIFY=option, $
```

where:

headfoot

Is the type of heading or footing. Valid values are TABHEADING, TABFOOTING, HEADING, FOOTING, SUBHEAD, and SUBFOOT.

line_#

Optionally identifies a line by its position in the heading or footing so that you can individually align it. If a heading or footing has multiple lines and you omit this option, the value supplied for JUSTIFY applies to all lines.

option

Is the type of justification. Valid values are:

LEFT which left justifies the heading or footing. LEFT is the default value.

RIGHT which right justifies the heading or footing.

CENTER which centers the heading or footing.

Note: JUSTIFY is not supported with WRAP.

Example: Justifying a Report Heading

The following example illustrates how to center the report heading, PRODUCT REPORT, using the attribute JUSTIFY=CENTER.

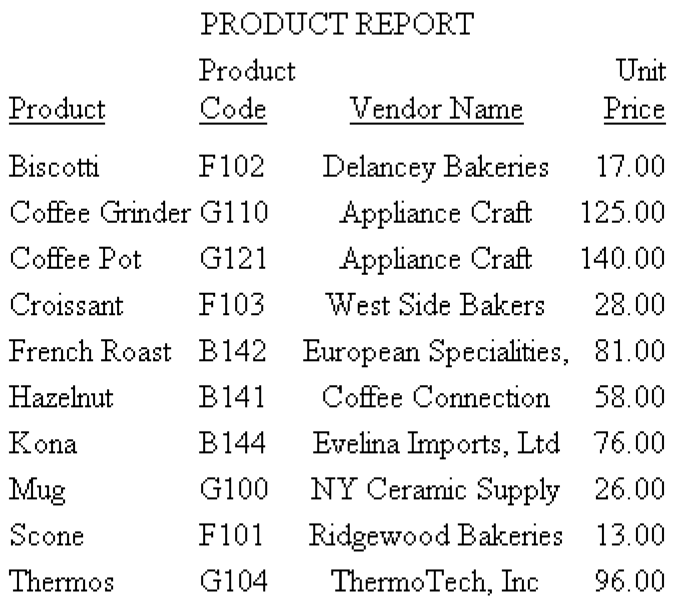
Report Request

```
TABLE FILE GGPRODS
SUM UNITS BY PRODUCT_DESCRIPTION BY PRODUCT_ID BY VENDOR_NAME
ON TABLE SUBHEAD
"PRODUCT REPORT"
ON TABLE SET STYLE *
INCLUDE=IBFS:/stylesheet_location/custom_stylesheet.sty,$
END
```

***custom_stylesheet*.sty**

```
TYPE=REPORT, GRID=OFF, $
TYPE=REPORT, COLUMN=VENDOR_NAME, JUSTIFY=CENTER, $
TYPE=TABHEADING, JUSTIFY=CENTER, $
```

The output is:



```
                   PRODUCT REPORT
                   Product                           Unit
Product            Code       Vendor Name            Price

Biscotti           F102     Delancey Bakeries        17.00
Coffee Grinder     G110     Appliance Craft         125.00
Coffee Pot         G121     Appliance Craft         140.00
Croissant          F103     West Side Bakers         28.00
French Roast       B142     European Specialities,   81.00
Hazelnut           B141     Coffee Connection        58.00
Kona               B144     Evelina Imports, Ltd     76.00
Mug                G100     NY Ceramic Supply        26.00
Scone              F101     Ridgewood Bakeries       13.00
Thermos            G104     ThermoTech, Inc          96.00
```

Tip: If you wish to run this report in PDF format, add SQUEEZE=ON to eliminate excessive white space between columns and to center the heading over the report.

For more information on justifying a column title, see *Justifying a Column Title* on page 267.

Example: **Justifying Individual Lines in a Multiple-Line Report Heading**

In the following example, heading line 1 (SALES REPORT) is centered, heading line 2 (**CONFIDENTIAL**) is also centered, and heading line 3 (December 2001) is right-justified.

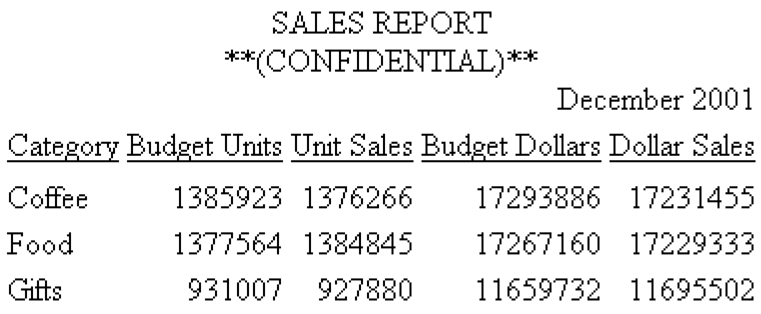
Report Request

```
TABLE FILE GGSALES
SUM BUDUNITS UNITS BUDDOLLARS DOLLARS
BY CATEGORY
ON TABLE SUBHEAD
"SALES REPORT"
"**(CONFIDENTIAL)**"
"December 2001"
ON TABLE SET STYLE *
INCLUDE=IBFS:/stylesheet_location/custom_stylesheet.sty,$
END
```

***custom_stylesheet*.sty**

```
TYPE=REPORT, GRID=OFF, $
TYPE=TABHEADING, LINE=1, JUSTIFY=CENTER, $
TYPE=TABHEADING, LINE=2, JUSTIFY=CENTER, $
TYPE=TABHEADING, LINE=3, JUSTIFY=RIGHT, $
```

The output is:



```
                        SALES REPORT
                    **(CONFIDENTIAL)**
                                              December 2001
Category Budget Units Unit Sales Budget Dollars Dollar Sales

Coffee        1385923    1376266       17293886     17231455
Food          1377564    1384845       17267160     17229333
Gifts          931007     927880       11659732     11695502
```

Tip: To run this report in PDF format, add SQUEEZE=ON to eliminate excessive white space between columns and to center the heading over the report.

Example: Centering All Lines in a Multiple-Line Report Heading

The following example illustrates how to center all lines in a multiple-line report heading, using the single StyleSheet attribute for the entire heading.

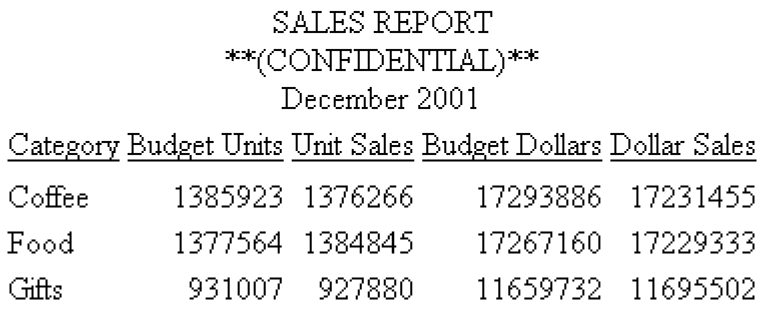
Report Request

```
TABLE FILE GGSALES
SUM BUDUNITS UNITS BUDDOLLARS DOLLARS
BY CATEGORY
ON TABLE SUBHEAD
"SALES REPORT"
"**(CONFIDENTIAL)**"
"December 2001"
ON TABLE SET STYLE *
INCLUDE=IBFS:/stylesheet_location/custom_stylesheet.sty,$
END
```

***custom_stylesheet*.sty**

```
TYPE=REPORT, GRID=OFF, $
TYPE=TABHEADING, JUSTIFY=CENTER, $
```

The output is:



```
                    SALES REPORT
                 **(CONFIDENTIAL)**
                   December 2001
Category Budget Units Unit Sales Budget Dollars Dollar Sales

Coffee        1385923    1376266       17293886     17231455
Food          1377564    1384845       17267160     17229333
Gifts          931007     927880       11659732     11695502
```

Tip: To run this report in PDF format, add SQUEEZE=ON to eliminate excessive white space between columns and to center the heading over the report.

Reference: Justification Regions and Behavior

The region in which text is justified depends on the relationship of the sizes of certain elements in the report:

- When SQUEEZE=ON, the maximum width of all the heading types in the report is calculated. This value is called MaxHeadWidth.

  If MaxHeadWidth is less than or equal to the total width of the columns of the report, headings are justified in the space over the report columns.

  If MaxHeadWidth exceeds the total width of the columns of the report, headings are centered and right-justified in the entire width of the page.

- When SQUEEZE=OFF, the maximum width of all the headings are not pre-calculated. Headings are centered in the entire width of the page.

  With a styled, multiple-panel report (in which the width exceeds one page), headings can only appear in the first panel. Thus, the preceding calculations deal with the total width of the columns in the first panel rather than the total width of all the columns in the report.

Justifying a Column Title

You can left-justify, right-justify, or center a column title for a display field, BY field, ACROSS field, or calculated value using a StyleSheet.

If a title is specified with an AS phrase in a request, or with the TITLE attribute in a Master File, that title will be justified, as specified for the field in StyleSheet syntax, if such syntax exists in the request.

Justification behavior in HTML and PDF. For HTML reports, justification is implemented with respect to the report width. That means a centered column title is centered over a report column. In contrast, for PDF reports the default justification area is the page width, rather than the report width, resulting in column titles that are not centered over the report column. You can achieve justification based on report width in a PDF report by adding SQUEEZE=ON to your request. This command improves the appearance of the report by eliminating excessive white space between columns and implements justification over the report content.

Syntax: How to Justify a Column Title Using a StyleSheet

To justify a column title for a vertical sort column (generated by BY) or a display column (generated by PRINT, LIST, SUM, or COUNT), the StyleSheet syntax is

```
TYPE=TITLE, [COLUMN=column,] JUSTIFY=option, $
TYPE=ACROSSTITLE, [ACROSS=column,] JUSTIFY=option, $
TYPE=ACROSSVALUE, [COLUMN=column,] JUSTIFY=option, $
```

To justify a horizontal sort column title (generated by ACROSS), the StyleSheet syntax is

```
TYPE=ACROSSTITLE, [ACROSS=column,] JUSTIFY=option, $
```

To justify an ACROSS value or a ROW-TOTAL column title in an HTML report, use

```
TYPE=ACROSSVALUE, [COLUMN=column,] JUSTIFY=option, $
```

where:

`TITLE`

Specifies a vertical sort (BY) title or a display field title.

`column`

Specifies the column whose title you wish to justify. If you omit this attribute and value, the formatting will be applied to all of the report column titles. For details on identifying columns, see *Identifying a Report Component in a Web Query StyleSheet* on page 57.

`ACROSSTITLE`

Specifies a horizontal sort (ACROSS) title.

`ACROSSVALUE`

Specifies a horizontal sort (ACROSS) value or a ROW-TOTAL column title.

option

Is the type of justification. Valid values are:

LEFT which left justifies the column title. This value is the default for an alphanumeric field.

RIGHT which right justifies the column title. This value is the default for a numeric or date field.

CENTER which centers the column title. You cannot center an ACROSSTITLE in a PDF report.

Note: JUSTIFY is not supported with WRAP.

Example: Using a StyleSheet to Justify Column Titles for Display and BY Fields

The following example illustrates how to center the column titles for STORE_NAME and ADDRESS1. The default column title for STORE_NAME is Store Name, as specified in the Master File with the TITLE attribute. The default column title for ADDRESS1 is Contact, also specified in the Master File. The request right-justifies the column title for STATE, which is specified in the AS phrase as St. Each column is identified by its field name and justified separately.

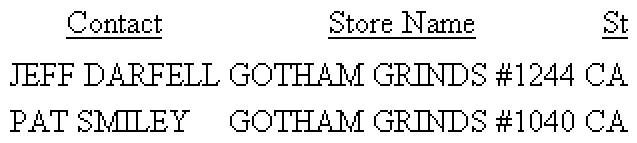
Report Request

```
TABLE FILE GGSTORES
PRINT STORE_NAME STATE AS 'St' BY ADDRESS1
WHERE STATE EQ 'CA'
ON TABLE SET STYLE *
INCLUDE=IBFS:/stylesheet_location/custom_stylesheet.sty,$
END
```

***custom_stylesheet*.sty**

```
TYPE=REPORT, GRID=OFF, $
TYPE=TITLE, SQUEEZE=ON, COLUMN=STORE_NAME, JUSTIFY=CENTER, $
TYPE=TITLE, COLUMN=STATE, JUSTIFY=RIGHT, $
TYPE=TITLE, COLUMN=ADDRESS1, JUSTIFY=CENTER, $
```

The output is:



Contact	Store Name	St
JEFF DARFELL	GOTHAM GRINDS #1244	CA
PAT SMILEY	GOTHAM GRINDS #1040	CA

Example:

Using a StyleSheet to Justify a Column Title for ACROSS and ROW-TOTAL Fields

The following example illustrates how to center the column title, State, created by the ACROSS phrase, over the two values (WY and MT) and the row total column title, Total by Gender, over the two row totals (Male Population and Female Population). Notice that each across value functions as a title for one or more columns in the report.

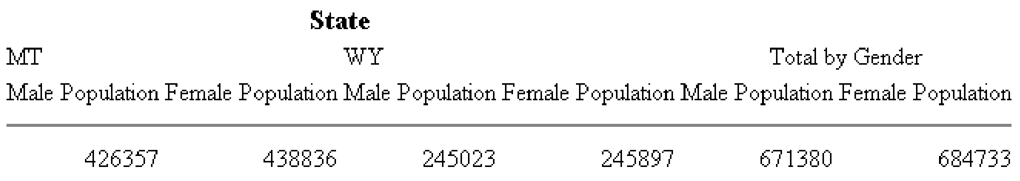
Report Request

```
TABLE FILE GGDEMOG
SUM MALEPOP98 FEMPOP98
ROW-TOTAL/D12 AS 'Total by Gender'
ACROSS ST
WHERE ST EQ 'WY' OR 'MT';
ON TABLE SET STYLE *
INCLUDE=IBFS:/stylesheet_location/custom_stylesheet.sty,$
END
```

***custom_stylesheet*.sty**

```
TYPE=REPORT, GRID=OFF, $
TYPE=ACROSSTITLE, JUSTIFY=CENTER, FONT='TIMES', SIZE=11, STYLE=BOLD, $
TYPE=ACROSSVALUE, COLUMN=N5, JUSTIFY=CENTER, $
```

The output is:



		State			
MT		WY		Total by Gender	
Male Population	Female Population	Male Population	Female Population	Male Population	Female Population
426357	438836	245023	245897	671380	684733

Example: Using a StyleSheet to Justify a Column Title for a Calculated Value

The following example illustrates how to identify the column title of the calculated value and left justify it over the data.

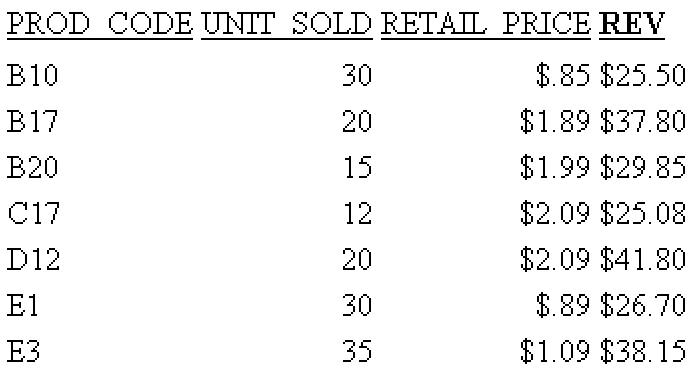
Report Request

```
TABLE FILE SALES
SUM UNIT_SOLD RETAIL_PRICE
COMPUTE REV/D12.2M = UNIT_SOLD * RETAIL_PRICE;
BY PROD_CODE
WHERE CITY EQ 'NEW YORK'
ON TABLE SET STYLE *
INCLUDE=IBFS:/stylesheet_location/custom_stylesheet.sty,$
END
```

***custom_stylesheet*.sty**

```
TYPE=REPORT, GRID=OFF, $
TYPE=TITLE, COLUMN=REV, STYLE=BOLD, JUSTIFY=LEFT, $
```

The output is:



```
PROD_CODE UNIT_SOLD RETAIL_PRICE REV
B10              30         $.85 $25.50
B17              20        $1.89 $37.80
B20              15        $1.99 $29.85
C17              12        $2.09 $25.08
D12              20        $2.09 $41.80
E1               30         $.89 $26.70
E3               35        $1.09 $38.15
```

Note: To run this report in PDF format, add SQUEEZE=ON to eliminate excessive white space between columns and to justify column titles properly over the data.

Justifying a Label for a Subtotal or Grand Total

You cannot directly justify a customized label for a subtotal. However, for HTML, EXL2K, or XLSX report output, if columns are being totaled or subtotaled by the one subtotal command, and you do not specify a column in the StyleSheet, formatting is applied to the totals and subtotals of all columns and to the labeling text that introduces the total and subtotal values. For related information, see *Identifying a Report Component in a Web Query StyleSheet* on page 57.

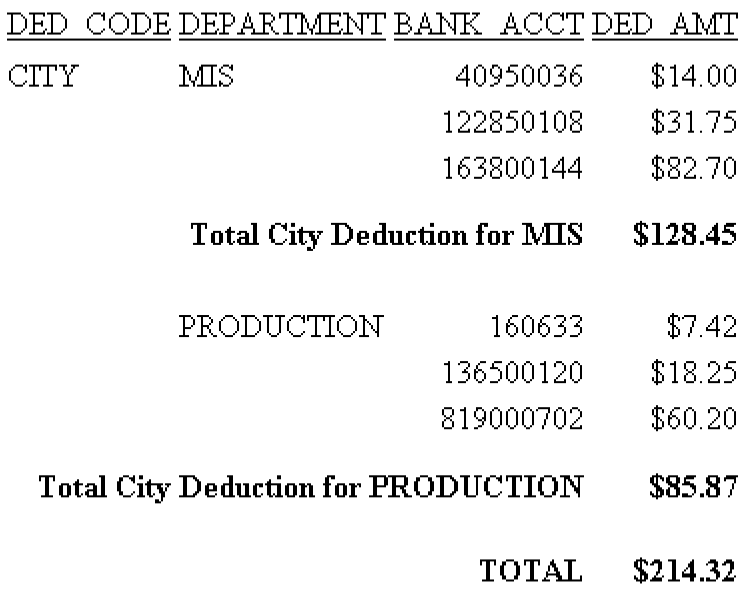
Example: Justifying Subtotal and Grand Total Labels

The following example illustrates how to subtotal the numeric columns in the report and right-justify the output, including the text of the label that precedes the values for the subtotals. Since numeric output is right-justified, by default, in this example the justification specifications in the StyleSheet are used to reposition the labels. The default label for the automatically generated grand total is also right-justified.

Report Request

```
TABLE FILE EMPLOYEE
SUM DED_AMT BY DED_CODE BY DEPARTMENT
BY BANK_ACCT
WHERE DED_CODE EQ 'CITY'
WHERE BANK_ACCT NE 0
ON DEPARTMENT SUBTOTAL AS 'Total City Deduction for'
ON TABLE SET STYLE *
INCLUDE=IBFS:/stylesheet_location/custom_stylesheet.sty,$
END

TYPE=REPORT, GRID=OFF, $
TYPE=SUBTOTAL, STYLE=BOLD, JUSTIFY=RIGHT, $
TYPE=GRANDTOTAL, STYLE=BOLD, JUSTIFY=RIGHT, $
```

The output is:



```
DED  CODE DEPARTMENT BANK  ACCT DED  AMT

CITY      MIS          40950036    $14.00
                      122850108    $31.75
                      163800144    $82.70

     Total City Deduction for MIS $128.45

          PRODUCTION     160633     $7.42
                      136500120    $18.25
                      819000702    $60.20

Total City Deduction for PRODUCTION $85.87

                          TOTAL   $214.32
```

Controlling the Vertical Positioning of a Heading or Footing

You can use several vertical positioning techniques to enhance the appearance and readability of a report:

- ❏ In a report generated in HTML, PDF, or other report formats, you can add one or more lines above or below heading or footing text using spot markers and free text lines. See *How to Add Blank Lines to a Heading or Footing* on page 274.
- ❏ In a PDF report, you can control the space above or below a heading or footing line or the distance between text and the surrounding grid lines in a heading or footing line using the TOPGAP and BOTTOMGAP attributes in a StyleSheet. See *How to Control Vertical Spacing in a Heading or Footing* on page 275. For full information on TOPGAP and BOTTOMGAP, see *Laying Out the Report Page* on page 123.
- ❏ In a PDF report, you can position a page footing at the bottom of a page, rather than directly after the report data.

Syntax: How to Add Blank Lines to a Heading or Footing

Use the following syntax options to add blank lines above or below a heading or footing, or within a heading or footing, where:

`</n`

Is a spot marker that specifies the number of lines to skip. It is best to put the spot marker on the same line as the text in the request. If you place the spot marker </*n* on a line by itself, it will add that line in addition to the designated number of skipped lines.

`" "`

Indicates a separate line in the heading or footing, with blank content.

You can use these techniques separately or in combination.

Example: Adding Blank Lines Above and Below a Report Heading

The following example illustrates how to create an HTML report with one blank line between each line of the page heading and two blank lines between the page heading and the actual report. The first blank line is added as an empty text line, The next blank lines are added with the skip-line spot marker.

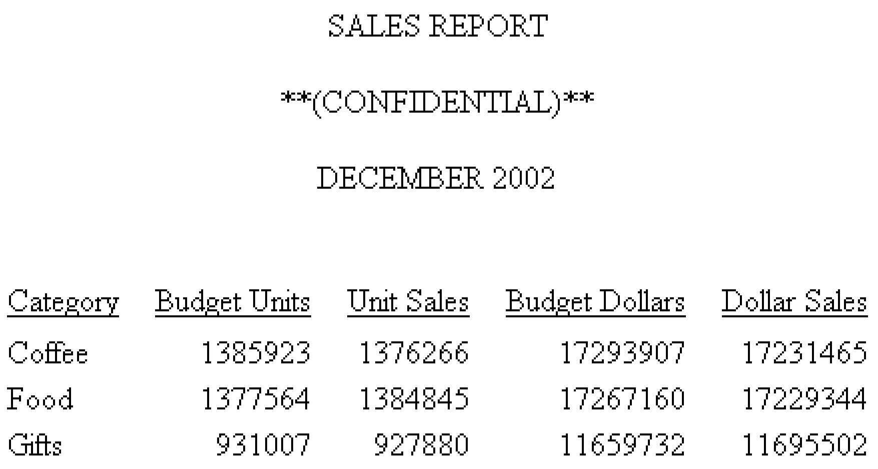
Report Request

```
TABLE FILE GGSALES
SUM BUDUNITS UNITS BUDDOLLARS DOLLARS
BY CATEGORY
ON TABLE SUBHEAD
"SALES REPORT"
" "
"**(CONFIDENTIAL)**</1"
"December 2002 </2"
ON TABLE SET STYLE *
INCLUDE=IBFS:/stylesheet_location/custom_stylesheet.sty,$
END
```

***custom_stylesheet*.sty**

```
TYPE=REPORT, GRID=OFF, TOPMARGIN=0, $
TYPE=TABHEADING, JUSTIFY=CENTER, $
```

The output is:



```
                    SALES REPORT

                 **(CONFIDENTIAL)**

                   DECEMBER 2002


Category   Budget Units   Unit Sales   Budget Dollars   Dollar Sales
Coffee          1385923      1376266         17293907       17231465
Food            1377564      1384845         17267160       17229344
Gifts            931007       927880         11659732       11695502
```

Syntax:

How to Control Vertical Spacing in a Heading or Footing

In a PDF report, you can use the TOPGAP and BOTTOMGAP attributes to control spacing above or below a heading or footing line or the distance between heading or footing text and the grid lines above and below them.

Note: You can use TOPGAP and BOTTOMGAP with multi-line headings. Keep in mind that between heading lines the top *and* bottom gap will be inserted, making the spacing between lines greater than the spacing at the top and bottom of the heading.

```
TYPE=headfoot, {TOPGAP|BOTTOMGAP}=gap, $
```

where:

headfoot

Is the type of heading or footing. Valid values are TABHEADING, TABFOOTING, HEADING, FOOTING, SUBHEAD, and SUBFOOT.

TOPGAP

Indicates how much space to add above a report component.

BOTTOMGAP

Indicates how much space to add below a report component.

gap

Is the amount of blank space, in the unit of measurement specified by the UNITS parameter (inches, by default).

In the absence of grids, the default value is 0.

In the presence of grids, the default value increases to provide space between the grid and the text.

Example: Adding Blank Space to Separate Heading Text From Grid Lines in a PDF Report

The following example illustrates how to generate a PDF report with blank space added above and below the report heading to separate the text from the upper and lower grid lines. The space above is added by the TOPGAP attribute. The space below is added by the BOTTOMGAP attribute.

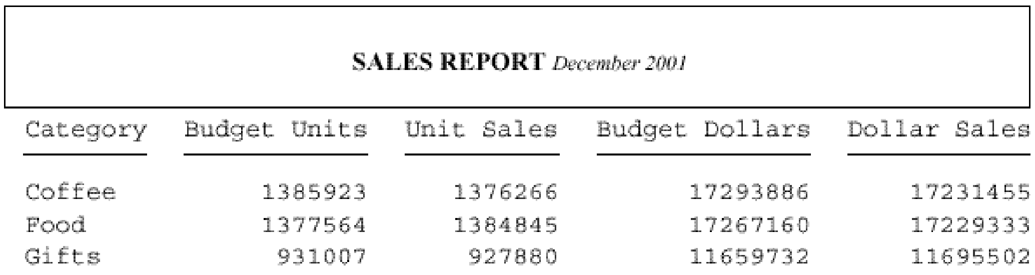
Report Request

```
TABLE FILE GGSALES
SUM BUDUNITS UNITS BUDDOLLARS DOLLARS
BY CATEGORY
ON TABLE SUBHEAD
"SALES REPORT <+0>December 2001"
ON TABLE PCHOLD FORMAT PDF
ON TABLE SET STYLE *
INCLUDE=IBFS:/stylesheet_location/custom_stylesheet.sty,$
END
```

***custom_stylesheet*.sty**

```
TYPE=REPORT, SQUEEZE=ON, $
TYPE=TABHEADING, GRID=ON, JUSTIFY=CENTER, TOPGAP=.25, BOTTOMGAP=.25, $
TYPE=TABHEADING, FONT='TIMES', SIZE=12, STYLE=BOLD, $
TYPE=TABHEADING, ITEM=2, SIZE=10, STYLE=ITALIC, $
```

The output is:



```
                    SALES REPORT December 2001

Category   Budget Units   Unit Sales   Budget Dollars   Dollar Sales

Coffee          1385923      1376266         17293886       17231455
Food            1377564      1384845         17267160       17229333
Gifts            931007       927880         11659732       11695502
```

Example: Adjusting Vertical Spacing Below a Sort Footing

The following example illustrates how to generate a PDF report in which the sort footings are bolded for emphasis and space is added below each footing to visually tie the footing text to the preceding data.

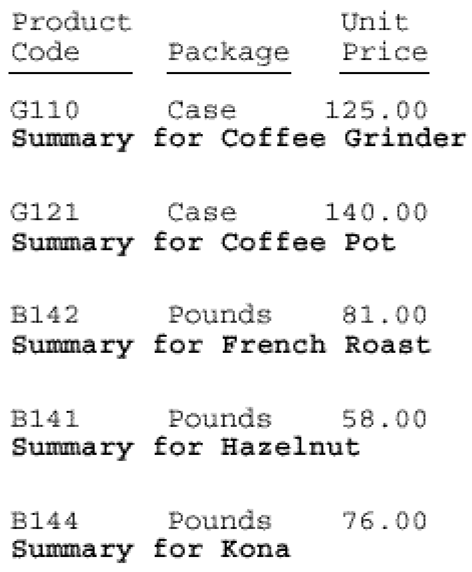
Report Request

```
TABLE FILE GGPRODS
PRINT PACKAGE_TYPE AND UNIT_PRICE
WHERE UNIT_PRICE GT 50
BY PRODUCT_DESCRIPTION NOPRINT BY PRODUCT_ID
ON PRODUCT_DESCRIPTION SUBFOOT
"Summary for <PRODUCT_DESCRIPTION"
ON TABLE PCHOLD FORMAT PDF
ON TABLE SET STYLE *
INCLUDE=IBFS:/stylesheet_location/custom_stylesheet.sty,$
END
```

***custom_stylesheet*.sty**

```
TYPE=REPORT, SQUEEZE=ON, $
TYPE=SUBFOOT, STYLE=BOLD, BOTTOMGAP=.25, $
```

The output is:



```
Product              Unit
Code       Package   Price

G110       Case      125.00
Summary for Coffee Grinder

G121       Case      140.00
Summary for Coffee Pot

B142       Pounds     81.00
Summary for French Roast

B141       Pounds     58.00
Summary for Hazelnut

B144       Pounds     76.00
Summary for Kona
```

Chapter 9

Formatting Report Data

The following topics covers information about formatting and positioning text in a report. You can select the size, color, font, and style for the text of your report, in addition to being able to adjust the position of text within a report.

Note: InfoAssist has built-in styling options that generate some of the StyleSheet syntax described in this chapter.

In this chapter:

Specifying Font Format in a Report

Using StyleSheet attributes, you can enhance the appearance of a report by specifying the font, size, and color of the font. Font format can be designated for a report as a whole, or for headings, footings, and columns individually.

Syntax: How to Specify Font Size in a Report

To specify a font size, use the following syntax within a StyleSheet.

```
TYPE=type, [subtype,] SIZE=pts, $
```

where:

type

Is the report component you wish to affect, such as REPORT, HEADING, or TITLE.

subtype

Is any additional attribute, such as COLUMN, ACROSS, or ITEM, that is needed to identify the report component that you are formatting. See *Identifying a Report Component in a Web Query StyleSheet* on page 57 for more information about how to specify different report components.

pts

Is the size of the font in points. The default value is 10, which corresponds to the HTML default font size 3. For more information on the correlation between point size and HTML font size, see *Usage Notes for Changing Font Size* on page 280.

Example: Specifying Font Size in a Report

The following example illustrates how to set the point size of column titles to 12.

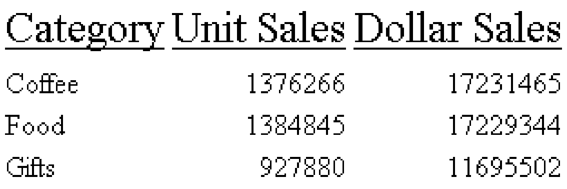
Report Request

```
TABLE FILE GGSALES
SUM UNITS DOLLARS BY CATEGORY
ON TABLE SET STYLE *
INCLUDE=IBFS:/stylesheet_location/custom_stylesheet.sty,$
END
```

***custom_stylesheet*.sty**

```
TYPE=REPORT, GRID=OFF, $
TYPE=TITLE, SIZE=12, $
```

The output is:



```
Category Unit Sales Dollar Sales
Coffee      1376266     17231465
Food        1384845     17229344
Gifts        927880     11695502
```

Reference: Usage Notes for Changing Font Size

Point size is fixed, except in an HTML report. Relative point size uses a different scale than HTML font size. The following table lists the point size and the corresponding HTML font size.

Size in Points	Corresponding HTML Font Size
8 or smaller	1
9	2
10	3
11	4

Size in Points	Corresponding HTML Font Size
12	5
13	6
14 or larger	7

Syntax: How to Specify Bold or Italic Font Style in a Report

To specify a font style, use the following syntax within a StyleSheet.

```
TYPE=type, [subtype,] STYLE=[+|-]txtsty[{+|-}txtsty], $
```

where:

type

Is the report component you wish to affect, such as REPORT, HEADING, or TITLE.

subtype

Is any additional attribute, such as COLUMN, ACROSS, or ITEM, that is needed to identify the report component that you are formatting. See *Identifying a Report Component in a Web Query StyleSheet* on page 57 for more information about how to specify different report components.

txtsty

Is one of the following values: NORMAL, BOLD, ITALIC. The default value is NORMAL.

Note that if you specify a style that is not supported for the font you are using, the specified font will display without the style.

`+`

Enables you to specify a combination of font styles. You can add additional font styles to an attribute that already has one or more font styles applied to it.

`-`

Enables you to remove a font style from an attribute.

Example: Specifying Font Style in a Report

The following example illustrates how to specify bold and italic font styles for the column titles.

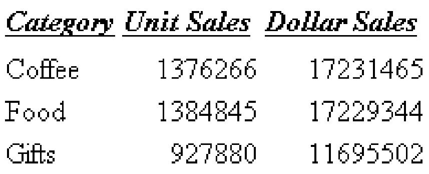
Report Request

```
TABLE FILE GGSALES
SUM UNITS DOLLARS BY CATEGORY
ON TABLE SET STYLE *
INCLUDE=IBFS:/stylesheet_location/custom_stylesheet.sty,$
END
```

***custom_stylesheet*.sty**

```
TYPE=TITLE, STYLE=BOLD+ITALIC, $
TYPE=REPORT, GRID=OFF, $
```

The output is:



```
Category  Unit Sales  Dollar Sales
Coffee       1376266      17231465
Food         1384845      17229344
Gifts         927880      11695502
```

Example: Adding and Removing Inherited Font Style in a Report

The following example illustrates how to specify bold and italic font styles for the entire report. The inherited italics are removed from the heading, and both styles are removed from the column titles.

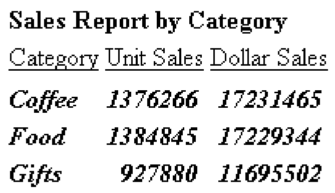
Report Request

```
TABLE FILE GGSALES
HEADING
"Sales Report by Category"
SUM UNITS DOLLARS BY CATEGORY
ON TABLE SET STYLE *
INCLUDE=IBFS:/stylesheet_location/custom_stylesheet.sty,$
END
```

***custom_stylesheet*.sty**

```
TYPE=REPORT, GRID=OFF, STYLE=BOLD + ITALIC, $
TYPE=HEADING, STYLE=-ITALIC, $
TYPE=TITLE, STYLE=-BOLD-ITALIC, $
```

The output is:



Sales Report by Category

Category	Unit Sales	Dollar Sales
Coffee	*1376266*	*17231465*
Food	*1384845*	*17229344*
Gifts	*927880*	*11695502*

Syntax: How to Specify Font Color in a Report

To specify a color for the font of a report or a report component, use the following syntax within a StyleSheet.

```
TYPE=type, [subtype,] COLOR={color|RGB({r g b|#hexcolor})}, $
```

where:

type

Is the report component you wish to affect, such as REPORT, HEADING, or TITLE.

subtype

Is any additional attribute, such as COLUMN, ACROSS, or ITEM, that is needed to identify the report component that you are formatting. See *Identifying a Report Component in a Web Query StyleSheet* on page 57 for more information about how to specify different report components.

color

Is one of the preset color values, such as GREY or GOLD. If the display or output device does not support colors, it substitutes shades of gray. The default value is BLACK. For a complete list of available color values, see *Color Values in a Report* on page 284.

RGB(*r g b*)

Specifies the font color using a mixture of red, green, and blue.

(*rgb*) Is the desired intensity of red, green, and blue, respectively. The values are on a scale of 0 to 255, where 0 is the least intense and 255 is the most intense. Note that using the three color components in equal intensities results in shades of gray.

`RGB(#hexcolor)`

Is the hexadecimal value for the color. For example, FF0000 is the hexadecimal value for red. The hexadecimal digits can be in uppercase or lowercase and must be preceded by a pound sign (#).

Reference: Color Values in a Report

The following chart lists all available color values that can be utilized with the syntax

`COLOR=color or BACKCOLOR=color,`

where *color* is one of the following values:

AQUA (CYAN)	MEDIUM FOREST GREEN (OLIVE)
AQUAMARINE	MEDIUM GOLDENROD
BLACK	MEDIUM ORCHID
BLUE VIOLET	MEDIUM SLATE BLUE
CADET BLUE	MEDIUM SPRING GREEN
CORAL	MEDIUM TURQUOISE
CORNFLOWER BLUE	MEDIUM VIOLET RED
CYAN (AQUA)	MIDNIGHT BLUE
DARK GREEN	NAVY (NAVY BLUE)
DARK OLIVE GREEN	OLIVE (MEDIUM FOREST GREEN)
DARK ORCHID	ORANGE
DARK SLATE BLUE (PURPLE)	ORANGE RED
DARK SLATE GREY	ORCHID
DARK TURQUOISE	PALE GREEN
DIM GREY (GRAY, GREY)	PINK

FIREBRICK	PLUM
FOREST GREEN (GREEN)	PURPLE (DARK SLATE BLUE)
FUCHSIA (MAGENTA)	RED
GOLD	SALMON
GOLDENROD	SEA GREEN
GRAY (DIM GREY, GREY)	SIENNA
GREEN (FOREST GREEN)	SILVER
GREEN YELLOW	SKY BLUE
GREY (DIM GREY, GRAY)	SLATE BLUE
INDIAN RED	STEEL BLUE (TEAL)
KHAKI	TAN
LIGHT BLUE	TEAL (STEEL BLUE)
LIGHT GREY	THISTLE
LIGHT STEEL BLUE	TURQUOISE
LIME	VIOLET
LIME GREEN	VIOLET RED
MAGENTA (FUCHSIA)	WHEAT
MAROON	WHITE
MEDIUM AQUAMARINE	YELLOW
MEDIUM BLUE	YELLOW GREEN

Specifying Fonts for Reports

You can specify your own fonts in a report by using the FONT attribute in a StyleSheet. If you are specifying a font for an HTML report, the web browser must support the font. If the web browser does not support the font, it reverts to its default behavior of using the default proportional font.

Syntax: How to Specify Fonts in a Report

To specify a font for your report, use the following syntax within a StyleSheet.

```
TYPE=type, [subtype,] FONT='font[,font]', $
```

where:

type

Is the report component you wish to affect, such as REPORT, HEADING, or TITLE.

subtype

Is any additional attribute, such as COLUMN, ACROSS, or ITEM, that is needed to identify the report component that you are formatting. See *Identifying a Report Component in a Web Query StyleSheet* on page 57 for more information about how to specify different report components.

font

Is the name of the font. You must enclose the value in single quotation marks ('). If you are creating an HTML report, you can specify more than one font within the single quotation marks (') to accommodate more than one browser.

Note: In an HTML report, specifying different fonts for several different report components significantly increases the size of the source code.

Example: Specifying Multiple Fonts in an HTML Report

To control how a report looks on more than one platform, you can specify multiple fonts. The web browser searches for the first font in the list. If the browser does not find the first font, it searches for the next font in the list. If none of the fonts are identified, the browser uses the default proportional font.

In the following example, the web browser first searches for the Arial font. If the browser does not find Arial, it searches for the Helvetica font. If neither font is identified, the browser uses the default proportional font.

```
TYPE=REPORT, FONT='ARIAL,HELVETICA', $
```

Syntax: How to Specify the Default Browser Fonts for HTML Reports

A browser assigns specific fonts as the default proportional and default monospaced fonts. To specify a default browser font for an HTML report, you use the reserved names, DEFAULT-PROPORTIONAL and DEFAULT-FIXED in the StyleSheet of your report. The browser displays the report accordingly.

To select the default fixed or proportional font of the browser, use the following syntax. Note that you must specify TYPE to indicate which report components you wish to affect.

```
FONT={DEFAULT-PROPORTIONAL|DEFAULT-FIXED}, $
```

where:

`DEFAULT-PROPORTIONAL`

Specifies the default proportional font of the web browser.

`DEFAULT-FIXED`

Specifies the default monospaced font of the web browser.

Example: Specifying Default Browser Fonts

In the following example, the web browser uses the default monospace font for the entire report except the report heading and the column headings. For these headings, the web browser uses the default proportional font.

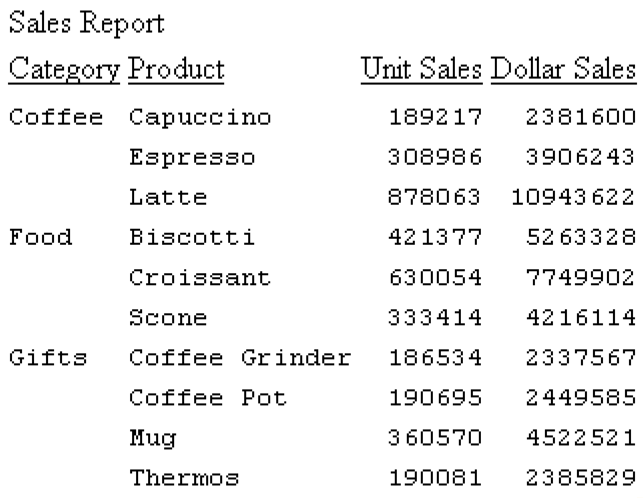
Report Request

```
TABLE FILE GGSALES
HEADING
"Sales Report"
SUM UNITS DOLLARS BY CATEGORY BY PRODUCT
ON TABLE SET STYLE *
INCLUDE=IBFS:/stylesheet_location/custom_stylesheet.sty,$
END
```

***custom_stylesheet*.sty**

```
TYPE=REPORT, GRID=OFF, FONT=DEFAULT-FIXED, $
TYPE=TITLE, FONT=DEFAULT-PROPORTIONAL, $
TYPE=HEADING, FONT=DEFAULT-PROPORTIONAL, $
```

The output is:



```
Sales Report
Category Product          Unit Sales Dollar Sales
Coffee   Capuccino          189217   2381600
         Espresso           308986   3906243
         Latte              878063  10943622
Food     Biscotti           421377   5263328
         Croissant          630054   7749902
         Scone              333414   4216114
Gifts    Coffee Grinder     186534   2337567
         Coffee Pot         190695   2449585
         Mug                360570   4522521
         Thermos            190081   2385829
```

Specifying Background Color in a Report

Using StyleSheet attributes, you can enhance the appearance of a report by specifying a background color. You can designate background colors for a report as a whole, or for headings, footings, and columns individually. In addition, alternating colors can be specified for the background of the data lines in a report.

Syntax:

How to Specify Background Color in a Report

To specify a color for the background of a report, use the following syntax within a StyleSheet.

Note that when using BACKCOLOR in a PDF report, extra space is added to the top, bottom, right, and left of each cell of data in the report. This is for readability and to prevent truncation.

```
TYPE=type, [subtype,] BACKCOLOR={color|RGB({r g b|#hexcolor})}, $
```

where:

type

Is the report component you wish to affect, such as REPORT, HEADING, or TITLE. In a HEADING, FOOTING, SUBHEADING, or SUBFOOTING, you can specify a background color for individual elements.

subtype

Is any additional attribute, such as COLUMN, ACROSS, or ITEM, that is needed to identify the report component that you are formatting. In a HEADING, FOOTING, SUBHEADING, or SUBFOOTING, you can specify a background color for individual elements. For more information about how to specify different report components, see *Identifying a Report Component in a Web Query StyleSheet* on page 57.

color

Is the background color, which fills the space of the specified report component. The default value is NONE. If you are creating a report in HTML format, background colors will only appear in web browsers that support them.

RGB (*r g b*)

Specifies the font color using a mixture of red, green, and blue.

(*rgb*) Is the desired intensity of red, green, and blue, respectively. The values are on a scale of 0 to 255, where 0 is the least intense and 255 is the most intense. Note that using the three color components in equal intensities results in shades of gray.

RGB (#*hexcolor*)

Is the hexadecimal value for the color. For example, FF0000 is the hexadecimal value for red. The hexadecimal digits can be in uppercase or lowercase and must be preceded by a pound sign (#).

Example: Specifying Background and Font Color in a Report

You can use color in a report to emphasize important information in a report. In the following example, the data in the Dollar Sales column has been specified as RED on the condition that the Dollars value is less than 2,500,000. The background color is set to LIGHT BLUE.

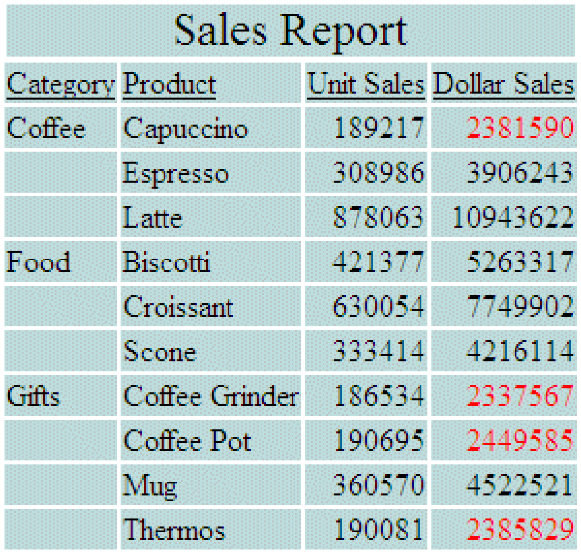
Report Request

```
TABLE FILE GGSALES
HEADING
"Sales Report"
SUM UNITS DOLLARS BY CATEGORY BY PRODUCT
ON TABLE SET STYLE *
INCLUDE=IBFS:/stylesheet_location/custom_stylesheet.sty,$
END
```

***custom_stylesheet*.sty**

```
TYPE=REPORT, GRID=OFF, BACKCOLOR=LIGHT BLUE, $
TYPE=DATA, COLUMN=DOLLARS, COLOR=RED, WHEN=DOLLARS LT 2500000, $
TYPE=HEADING, JUSTIFY=CENTER, SIZE=12, $
```

The output is:



Sales Report			
Category	Product	Unit Sales	Dollar Sales
Coffee	Capuccino	189217	2381590
	Espresso	308986	3906243
	Latte	878063	10943622
Food	Biscotti	421377	5263317
	Croissant	630054	7749902
	Scone	333414	4216114
Gifts	Coffee Grinder	186534	2337567
	Coffee Pot	190695	2449585
	Mug	360570	4522521
	Thermos	190081	2385829

Syntax: How to Specify Alternating Data Background Color in a Report

To specify alternating colors for the background of the data in a report, use the following syntax within a StyleSheet.

Note that when using BACKCOLOR in a PDF report, extra space is added to the top, bottom, right, and left of each cell of data in the report. This is for readability and to prevent truncation.

```
TYPE=DATA,BACKCOLOR=({c1|RGB({r1 g1 b1|#hc1})}
   {c2|RGB({r2 g2 b2|#hc2})}), $
```

where:

`c1, c2`

Are the background colors for the data in the report. The default value is NONE. If you are creating a report in HTML format, background colors will only appear in web browsers that support them. Color names that contain a space must be enclosed in single quotation marks ('), as a space is the delimiter between the alternating color values.

`RGB(r1 g1 b1), RGB(r2 g2 b2)`

Specifies the font colors using a mixture of red, green, and blue.

(*rgb*) Is the desired intensity of red, green, and blue, respectively. The values are on a scale of 0 to 255, where 0 is the least intense and 255 is the most intense. Note that using the three color components in equal intensities results in shades of gray.

`RGB(#hc1), RGB(#hc2)`

Are the hexadecimal values for the colors. For example, FF0000 is the hexadecimal value for red. The hexadecimal digits can be in uppercase or lowercase and must be preceded by a pound sign (#).

Example: Specifying Alternating Background Colors for the Data Lines in a Report

The following request against the GGSALES data source produces alternating light blue and white data rows on the report output.

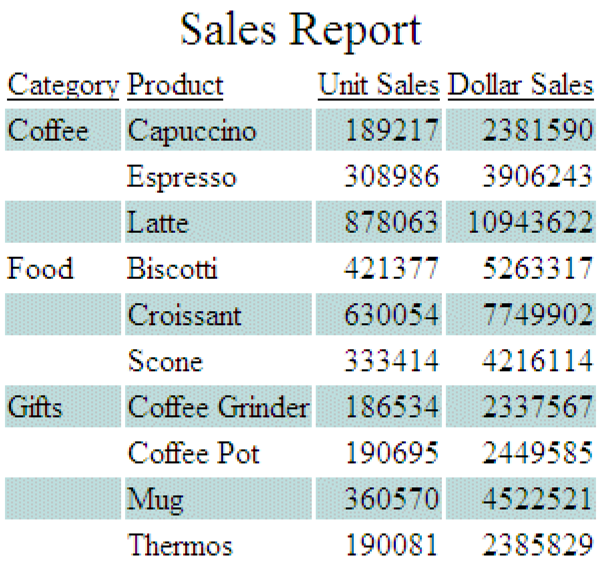
Report Request

```
TABLE FILE GGSALES
HEADING
"Sales Report"
SUM UNITS DOLLARS
BY CATEGORY
BY PRODUCT
ON TABLE PCHOLD FORMAT HTML
ON TABLE SET STYLE *
INCLUDE=IBFS:/stylesheet_location/custom_stylesheet.sty,$
END
```

***custom_stylesheet*.sty**

```
TYPE=REPORT, GRID=OFF, $
TYPE=DATA, BACKCOLOR=('LIGHT BLUE' WHITE), $
TYPE=HEADING, JUSTIFY=CENTER, SIZE=12, $
```

The output is:



Sales Report

Category	Product	Unit Sales	Dollar Sales
Coffee	Capuccino	189217	2381590
	Espresso	308986	3906243
	Latte	878063	10943622
Food	Biscotti	421377	5263317
	Croissant	630054	7749902
	Scone	333414	4216114
Gifts	Coffee Grinder	186534	2337567
	Coffee Pot	190695	2449585
	Mug	360570	4522521
	Thermos	190081	2385829

Positioning Data in a Report

You can position data within a report by selecting a justification (right, left, or center) of a column or by specifying whether or not you wish to have data wrap within a cell. For information on positioning a column on the page, see *Laying Out the Report Page* on page 123.

Controlling Wrapping of Report Data

You can control the wrapping of report data in a report, thereby preventing line breaks within report cells. When using HTML output, most web browsers will, by default, wrap alphanumeric report data that does not fit on a single line in a cell.

This bumps the contents of the cell onto a second line. A web browser wraps data based on its algorithmic settings. Use the WRAP attribute if you wish to suppress a web browser data wrapping.

By default, WRAP is set to ON for HTML output, allowing each individual browser to define the width of each column in the report. For PDF, DHTML, and PPTX output, WRAP is set OFF by default. For these positioned output formats in which the location of each item in the report is explicitly defined, WRAP=ON is not a valid value, except when specified for ACROSSVALUE. For other elements of the report, such as headers, footers, titles, or data, define the width of wrapped lines by using a numeric value, as in WRAP=*n*.

In PDF report output, you can control the line spacing in wrapped lines by using the WRAPGAP attribute.

Wrapping Data in PDF Reports That Use the OVER Phrase

OVER allows the presentation of a single data record across multiple lines within a report. By default, when OVER is defined within a request, the report shifts from a columnar presentation to a row level presentation. The field titles are displayed to the left of each value, rather than at the top of each column. This layout was not designed to be aligned in any specific fashion, but to allow for the presentation of multiple elements of data within a small area. In many cases, reports that place columns over each other use blank AS names in order to align the columns properly. You can use the WRAP attribute to wrap data in PDF reports that use OVER and this technique works well with blank AS names.

Wrapping Data in PDF Reports That Use the ACROSS Phrase

In a request that uses ACROSS, the output displays each value of the ACROSS field above the set of data columns applicable to that ACROSS value.

If the ACROSS value is longer than the width of its columns, you can wrap the ACROSS value within the width of its underlying columns.

By default, the width of each ACROSS value group (the ACROSS value and the data columns within) is defined as the largest of either the sum of the width of the data columns or the largest ACROSS value for that group. With wrapping, the size of each ACROSS wrap will be defined by the width defined based on this rule including all data columns and any non-wrapped across fields.

The width of each ACROSS column for a given ACROSS group is defined as the length of the largest value for that ACROSS group. A single width is used for each group so in groups where the values are shorter than the longest value, you will see a larger right gap within the cell.

For reports containing multiple ACROSS fields, you can wrap individual ACROSS fields or all of them. Each designated value will wrap within the defined ACROSS group.

Syntax: How to Control Wrapping of Report Data

To control wrapping of text inside a report, use the following syntax within a StyleSheet.

```
TYPE=type, [subtype,] WRAP=value, $
```

where:

type

Is the report component you wish to affect, such as REPORT, HEADING, or TITLE.

subtype

Is any additional attribute, such as COLUMN, ACROSS, or ITEM, that is needed to identify the report component that you are formatting. See *Identifying a Report Component in a Web Query StyleSheet* on page 57 for more information about how to specify different report components.

value

Is one of the following:

- **ON,** which turns on data wrapping. ON is the default value for HTML report output. For PDF, DHTML, and PPTX report output, WRAP is set OFF, by default. For these positioned output formats in which the location of each item in the report is explicitly defined, WRAP=ON is not a valid value, except when specified for ACROSSVALUE. For other elements of the report, such as headers, footers, titles, or data, define the width of wrapped lines by using a numeric value, as in WRAP=*n*. For HTML reports, WRAP is supported with all fields. For PDF reports, WRAP is supported only with embedded fields, not text.

  Note: This setting is not supported when using WRAP with OVER in PDF report output.

- **OFF,** which turns off data wrapping. This is the default value for PDF, DHTML, and PPTX report output.

- ***n*,** which represents a specific numeric value to which the column width can be set. The value represents the measure specified with the UNITS parameter. This setting is supported for wrapping data in PDF reports that use the OVER phrase.

Note: WRAP=ON and WRAP=*n* are not supported with JUSTIFY.

Example: Allowing the Web Browser to Wrap Report Data

The following example, with WRAP=ON, wraps report data based on the web browser functionality. Note that because this value is the default, there is no need to specify WRAP=ON in the report request syntax.

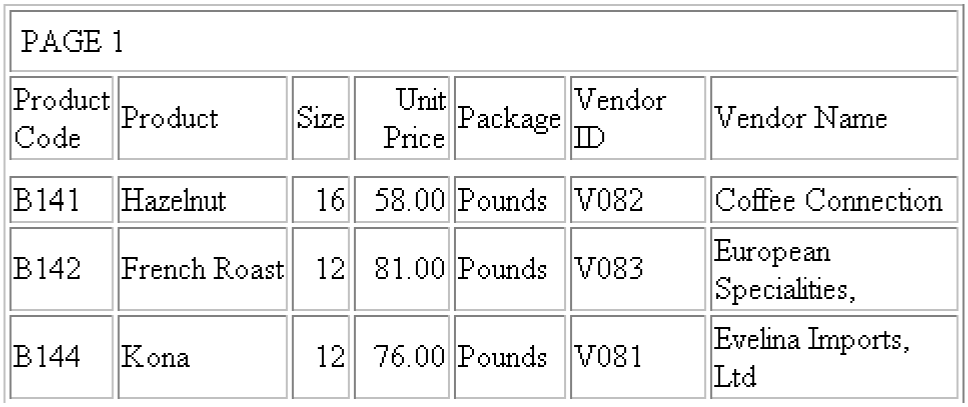
Report Request

```
TABLE FILE GGPRODS
PRINT SIZE UNIT_PRICE PACKAGE_TYPE
VENDOR_CODE VENDOR_NAME
BY PRODUCT_ID BY PRODUCT_DESCRIPTION
ON TABLE SET STYLE *
INCLUDE=IBFS:/stylesheet_location/custom_stylesheet.sty,$
END
```

***custom_stylesheet*.sty**

```
TYPE=REPORT, GRID=ON, $
```

Note: Wrap is determined by the size of your browser window, so you may need to shrink your window to see the example wrap the data, as shown in the following image.



PAGE 1						
Product Code	Product	Size	Unit Price	Package	Vendor ID	Vendor Name
B141	Hazelnut	16	58.00	Pounds	V082	Coffee Connection
B142	French Roast	12	81.00	Pounds	V083	European Specialities,
B144	Kona	12	76.00	Pounds	V081	Evelina Imports, Ltd

Notice that records in the Vendor Name column break to a second line.

Example: Suppressing the Wrapping of Report Data

The following example, with WRAP=OFF, suppresses the web browser data wrapping.

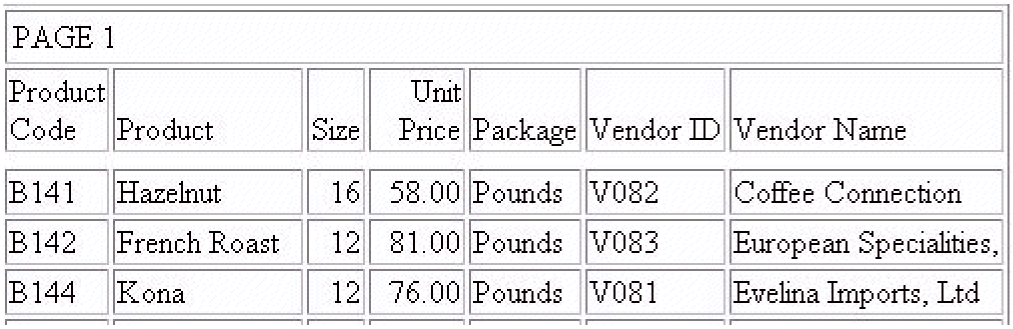
Report Request

```
TABLE FILE GGPRODS
PRINT SIZE UNIT_PRICE PACKAGE_TYPE
VENDOR_CODE VENDOR_NAME
BY PRODUCT_ID BY PRODUCT_DESCRIPTION
ON TABLE SET STYLE *
INCLUDE=IBFS:/stylesheet_location/custom_stylesheet.sty,$
END
```

***custom_stylesheet*.sty**

```
TYPE=REPORT, WRAP=OFF, GRID=ON, $
```

The output is:



PAGE 1						
Product Code	Product	Size	Unit Price	Package	Vendor ID	Vendor Name
B141	Hazelnut	16	58.00	Pounds	V082	Coffee Connection
B142	French Roast	12	81.00	Pounds	V083	European Specialities,
B144	Kona	12	76.00	Pounds	V081	Evelina Imports, Ltd

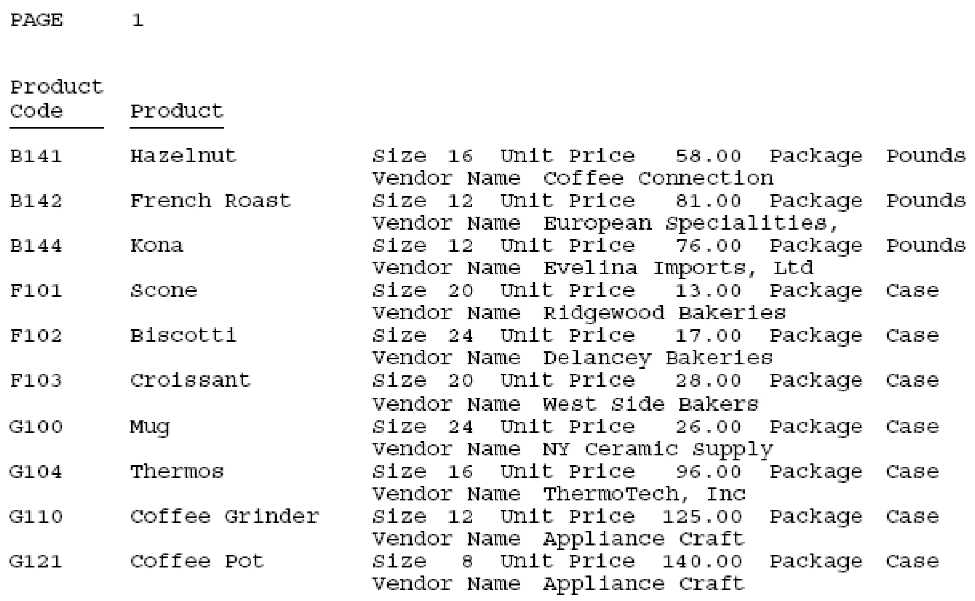
Example: **Wrapping Columns With OVER**

The following request against the GGPRODS data source places the column VENDOR_NAME on a new line with the OVER phrase. By default, wrap is turned off and must be defined explicitly within the StyleSheet.

Report Request

```
TABLE FILE GGPRODS
PRINT SIZE UNIT_PRICE PACKAGE_TYPE OVER
VENDOR_NAME
BY PRODUCT_ID BY PRODUCT_DESCRIPTION
ON TABLE PCHOLD FORMAT PDF
END
```

The partial output is shown in the following image.



```
PAGE     1


Product
Code     Product

B141     Hazelnut             Size 16  Unit Price   58.00  Package  Pounds
                              Vendor Name  Coffee Connection
B142     French Roast         Size 12  Unit Price   81.00  Package  Pounds
                              Vendor Name  European Specialities,
B144     Kona                 Size 12  Unit Price   76.00  Package  Pounds
                              Vendor Name  Evelina Imports, Ltd
F101     Scone                Size 20  Unit Price   13.00  Package  Case
                              Vendor Name  Ridgewood Bakeries
F102     Biscotti             Size 24  Unit Price   17.00  Package  Case
                              Vendor Name  Delancey Bakeries
F103     Croissant            Size 20  Unit Price   28.00  Package  Case
                              Vendor Name  West Side Bakers
G100     Mug                  Size 24  Unit Price   26.00  Package  Case
                              Vendor Name  NY Ceramic Supply
G104     Thermos              Size 16  Unit Price   96.00  Package  Case
                              Vendor Name  ThermoTech, Inc
G110     Coffee Grinder       Size 12  Unit Price  125.00  Package  Case
                              Vendor Name  Appliance Craft
G121     Coffee Pot           Size  8  Unit Price  140.00  Package  Case
                              Vendor Name  Appliance Craft
```

The following version of the request turns wrapping on and sets a column width of 1.5 for the VENDOR_NAME column, which has been placed on a new line because of the OVER phrase.

Report Request

```
TABLE FILE GGPRODS
PRINT SIZE UNIT_PRICE PACKAGE_TYPE OVER
VENDOR_NAME
BY PRODUCT_ID BY PRODUCT_DESCRIPTION
ON TABLE PCHOLD FORMAT PDF
ON TABLE SET STYLE *
INCLUDE=IBFS:/stylesheet_location/custom_stylesheet.sty,$
END
```

***custom_stylesheet*.sty**

```
TYPE=REPORT, COLUMN=VENDOR_NAME, WRAP=1.5, $
```

The partial output shows that the VENDOR_NAME column now wraps. Notice that turning WRAP ON causes the OVER value, not the OVER TITLE, to wrap:

```
PAGE     1


Product
Code      Product

B141      Hazelnut            Size  16  Unit Price   58.00  Package  Pounds
                              Vendor Name  Coffee
                                           Connection
B142      French Roast        Size  12  Unit Price   81.00  Package  Pounds
                              Vendor Name  European
                                           Specialities,
B144      Kona                Size  12  Unit Price   76.00  Package  Pounds
                              Vendor Name  Evelina
                                           Imports, Ltd
F101      Scone               Size  20  Unit Price   13.00  Package  Case
                              Vendor Name  Ridgewood
                                           Bakeries
F102      Biscotti            Size  24  Unit Price   17.00  Package  Case
                              Vendor Name  Delancey
                                           Bakeries
F103      Croissant           Size  20  Unit Price   28.00  Package  Case
                              Vendor Name  West Side
                                           Bakers
G100      Mug                 Size  24  Unit Price   26.00  Package  Case
                              Vendor Name  NY Ceramic
                                           Supply
```

Syntax: How to Wrap ACROSS Values

Wrapping ACROSS values is supported for HTML and PDF output formats.

```
TYPE=ACROSSVALUE, [ACROSS={fieldname|Nn|An}] WRAP={OFF|ON}, $
```

where:

`ACROSS`

If you have a request with multiple ACROSS fields, you can identify each field using the ACROSS identifier. You only need to include the ACROSS identifier if you have multiple ACROSS fields in your request.

`fieldname`

Specifies a horizontal sort row by its field name.

`N`*`n`*

Identifies a column by its position in the report. To determine this value, count vertical sort (BY) fields, display fields, and ROW-TOTAL fields, from left to right, including NOPRINT fields.

A*n*

Specifies a horizontal sort row by its position in the sequence of horizontal sort rows. To determine this value, count horizontal sort (ACROSS) fields. Cannot be combined with a field name specification in the same StyleSheet.

OFF

Turns off wrapping of the ACROSS values. OFF is the default value.

ON

Turns on wrapping of the ACROSS values.

Note: WRAP=ON is not supported with JUSTIFY.

Example: Wrapping ACROSS Values in PDF Report Output

In the following request against the GGPRODS data source, VENDOR_NAME is an ACROSS field.

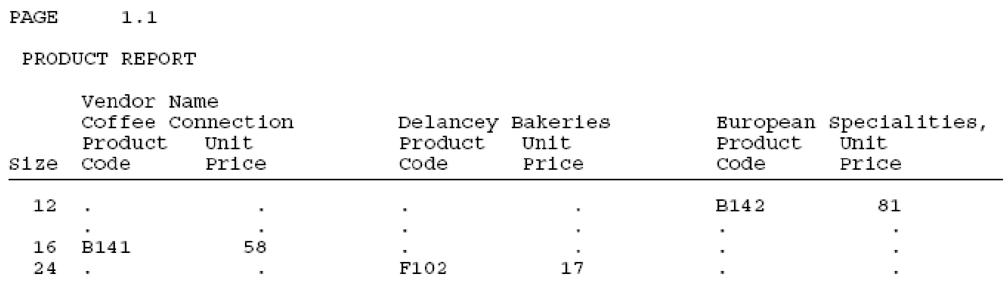
Report Request

```
TABLE FILE GGPRODS
HEADING
" PRODUCT REPORT"
" "
PRINT PRODUCT_ID UNIT_PRICE/D5
ACROSS VENDOR_NAME
BY SIZE
WHERE VENDOR_NAME GT 'B' AND VENDOR_NAME LT 'F'
ON TABLE PCHOLD FORMAT PDF
ON TABLE SET STYLE *
INCLUDE=IBFS:/stylesheet_location/custom_stylesheet.sty,$
END
```

***custom_stylesheet*.sty**

```
TYPE=REPORT, GRID=OFF, $
TYPE=REPORT, COLUMN=PRODUCT_ID, WIDTH=.25, $
TYPE=REPORT, COLUMN=UNIT_PRICE, WIDTH=.25, $
```

As shown in the following image, the output is too wide for one panel because some of the ACROSS field values (vendor names) are longer than the sum of the product code and unit price columns under them.



```
PAGE     1.1

 PRODUCT REPORT

      Vendor Name
      Coffee Connection                Delancey Bakeries                European Specialities,
      Product    Unit                  Product    Unit                  Product    Unit
Size  Code       Price                 Code       Price                 Code       Price
-------------------------------------------------------------------------------------------
  12  .              .                 .              .                 B142          81
      .              .                 .              .                 .              .
  16  B141          58                 .              .                 .              .
  24  .              .                 F102          17                 .              .
```

The following version of the request wraps the ACROSS values (TYPE=ACROSSVALUE, WRAP=ON, $).

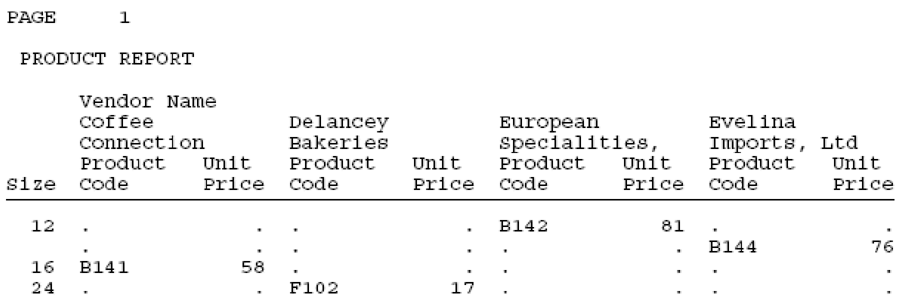
Report Request

```
TABLE FILE GGPRODS
HEADING
" PRODUCT REPORT"
" "
PRINT PRODUCT_ID UNIT_PRICE/D5
ACROSS VENDOR_NAME
BY SIZE
WHERE VENDOR_NAME GT 'B' AND VENDOR_NAME LT 'F'
ON TABLE PCHOLD FORMAT PDF
ON TABLE SET STYLE *
INCLUDE=IBFS:/stylesheet_location/custom_stylesheet.sty,$
END
```

***custom_stylesheet*.sty**

```
TYPE=REPORT, GRID=OFF, $
TYPE=REPORT, COLUMN=PRODUCT_ID, WIDTH=.25, $
TYPE=REPORT, COLUMN=UNIT_PRICE, WIDTH=.25, $
TYPE=ACROSSVALUE, WRAP=ON, $
```

The report now fits on one panel, as shown in the following image.



```
PAGE      1

 PRODUCT REPORT

      Vendor Name
      Coffee              Delancey            European            Evelina
      Connection          Bakeries            Specialities,       Imports, Ltd
      Product    Unit     Product    Unit     Product    Unit     Product    Unit
Size  Code       Price    Code       Price    Code       Price    Code       Price

  12  .              .    .              .    B142          81    .              .
      .              .    .              .    .              .    B144          76
  16  B141          58    .              .    .              .    .              .
  24  .              .    F102          17    .              .    .              .
```

Reference: OVER With Blank Column Titles

When OVER fields are defined with blank AS names (the value of the title of the column is set to empty ' '), they can be used to build a report with multiple data lines that present in an aligned grid fashion.

In this type of report, the column titles are usually indicated by adding multiple corresponding lines to the page headings, rather than using the default titles that display to the left of the column field values. To present OVER fields with unique titles that take advantage of these new alignment features, you can place the column titles in independent fields and include them as fields within the given request.

Example: Using OVER and WRAP With Blank AS Names

The following example illustrates how to use OVER with blank AS names and WRAP to build a multi-data line report.

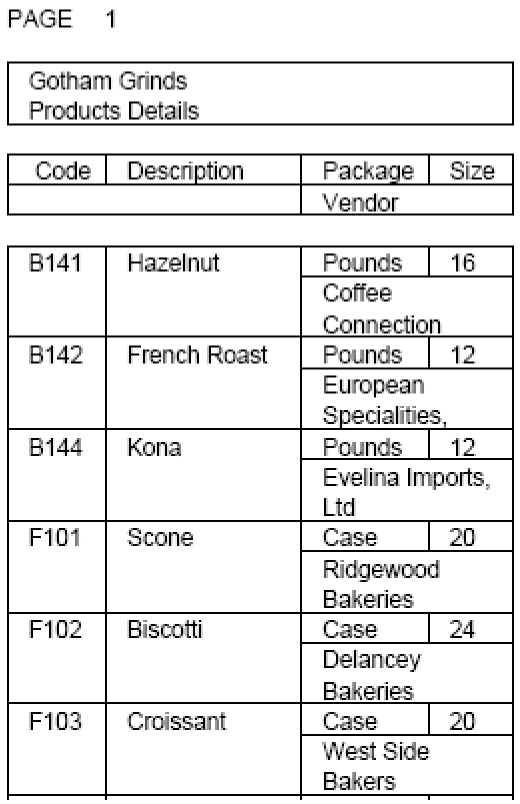
Report Request

```
TABLE FILE GGPRODS
PRINT PACKAGE_TYPE AS '' SIZE  AS '' OVER
VENDOR_NAME AS ''
BY PRODUCT_ID  AS ''
BY PRODUCT_DESCRIPTION  AS ''
ON TABLE SUBHEAD
"Gotham Grinds"
"Products Details"
HEADING
" Code <+0>Description<+0>Size  <+0>Package"
-*" <+0> <+0>Vendor"
" <+0>Vendor"
ON TABLE PCHOLD FORMAT PDF
ON TABLE SET STYLE *
INCLUDE=IBFS:/stylesheet_location/custom_stylesheet.sty,$
END
```

***custom_stylesheet*.sty**

```
TYPE=REPORT, FONT=ARIAL, SIZE=10, BORDER=ON, SQUEEZE=ON, $
TYPE=REPORT, COLUMN=PACKAGE_TYPE, SQUEEZE=.5, $
TYPE=REPORT, COLUMN=VENDOR_NAME, WRAP=1, $
TYPE=HEADING, LINE=1, ITEM=1, BORDER=ON, $
TYPE=HEADING, LINE=1, ITEM=2, BORDER=ON, POSITION=PRODUCT_DESCRIPTION, $
TYPE=HEADING, LINE=1, ITEM=3, BORDER=ON, POSITION=SIZE, $
TYPE=HEADING, LINE=1, ITEM=4, BORDER=ON, POSITION=PACKAGE_TYPE, $
TYPE=HEADING, LINE=2, ITEM=1, BORDER=ON, $
TYPE=HEADING, LINE=2, ITEM=2, BORDER=ON, POSITION=PACKAGE_TYPE, $
```

On the report output, the Package Type and Size have been placed over the vendor name. The page heading has the corresponding titles. In the heading, the titles Package and Size have also been placed over the title Vendor Name. Note that the vendor name data wraps to maintain the alignment.



PAGE 1

Gotham Grinds
Products Details

Code	Description	Package	Size
		Vendor	
B141	Hazelnut	Pounds	16
		Coffee Connection	
B142	French Roast	Pounds	12
		European Specialities,	
B144	Kona	Pounds	12
		Evelina Imports, Ltd	
F101	Scone	Case	20
		Ridgewood Bakeries	
F102	Biscotti	Case	24
		Delancey Bakeries	
F103	Croissant	Case	20
		West Side Bakers	

Reference: OVER and WRAP With Non-Blank Column Titles

The width of both the column title and the column data for each OVER value is determined by the single SQUEEZE or WRAP value. The title will automatically size to the same width as the wrapped data column. If the column title is wider than the width defined for the column wrap, you can either define a smaller title or add your titles as OVER fields that can be sized independently.

The following examples illustrate how to build a report with OVER and WRAP that have column titles longer than the designated WRAP size.

Example: Using OVER and WRAP With Column Titles

The following example illustrates how to define two virtual fields to contain the column titles for the Product Name and Vendor Name fields. It then prints each virtual field next to its related data field and gives each a blank AS name. The first virtual field and data field are placed over the second virtual field and data field.

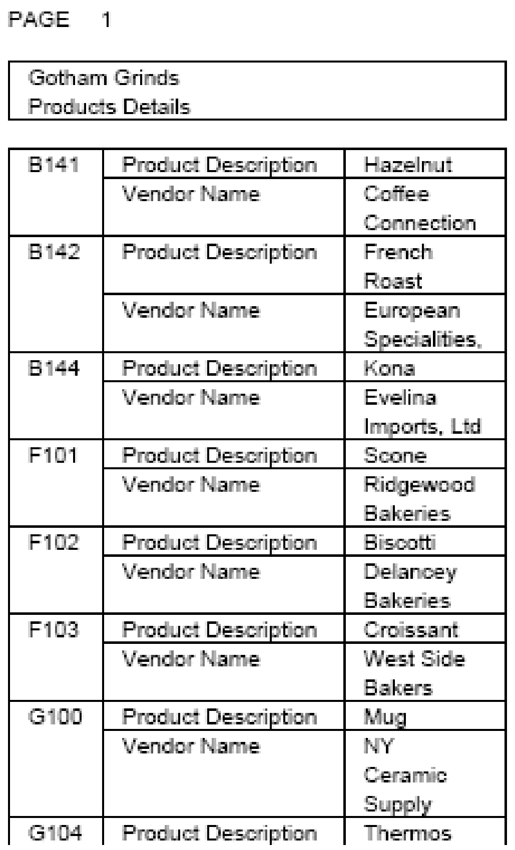
Report Request

```
DEFINE FILE GGPRODS
TITLE_PROD/A20 = 'Product Description';
TITLE_VEND/A20 = 'Vendor Name';
END
TABLE FILE GGPRODS
PRINT TITLE_PROD AS '' PRODUCT_DESCRIPTION  AS '' OVER
TITLE_VEND AS '' VENDOR_NAME  AS ''
BY PRODUCT_ID  AS ''
ON TABLE SUBHEAD
"Gotham Grinds"
"Products Details"
ON TABLE PCHOLD FORMAT PDF
ON TABLE SET STYLE *
INCLUDE=IBFS:/stylesheet_location/custom_stylesheet.sty,$
END
```

***custom_stylesheet*.sty**

```
TYPE=REPORT, FONT=ARIAL, SIZE=10, BORDER=ON, SQUEEZE=ON, $
TYPE=REPORT, COLUMN=TITLE_PROD, SQUEEZE=1.25, $
TYPE=REPORT, COLUMN=TITLE_VEND, SQUEEZE=1.25, $
TYPE=REPORT, COLUMN=PRODUCT_DESCRIPTION, WRAP=.75, $
TYPE=REPORT, COLUMN=VENDOR_NAME, WRAP=.75, $
```

The output shows that the titles and data align properly.



PAGE 1

Gotham Grinds
Products Details

B141	Product Description	Hazelnut
	Vendor Name	Coffee Connection
B142	Product Description	French Roast
	Vendor Name	European Specialities,
B144	Product Description	Kona
	Vendor Name	Evelina Imports, Ltd
F101	Product Description	Scone
	Vendor Name	Ridgewood Bakeries
F102	Product Description	Biscotti
	Vendor Name	Delancey Bakeries
F103	Product Description	Croissant
	Vendor Name	West Side Bakers
G100	Product Description	Mug
	Vendor Name	NY Ceramic Supply
G104	Product Description	Thermos

Syntax: How to Control Spacing Between Wrapped Lines

You can use the WRAPGAP attribute in a StyleSheet to control spacing between wrapped lines in PDF report output.

```
TYPE=component, WRAPGAP={ON|OFF|n}
```

where:

component

Is the component with wrapped lines.

ON

Does not leave any space between wrapped lines. ON is equivalent to specifying 0.0 for *n*.

OFF

Places wrapped data on the next line. OFF is the default value.

n

Is a number greater than or equal to zero (0) that specifies how much space to leave between wrapped lines (using the unit of measurement specified by the UNITS attribute). Setting *n* to zero (0) does not leave any space between wrapped lines, and is equivalent to specifying WRAPGAP=ON.

Example: Specifying Spacing for Wrapped Lines

In the following example, wrapping is turned on for the ADDRESS_LN3 column of the report.

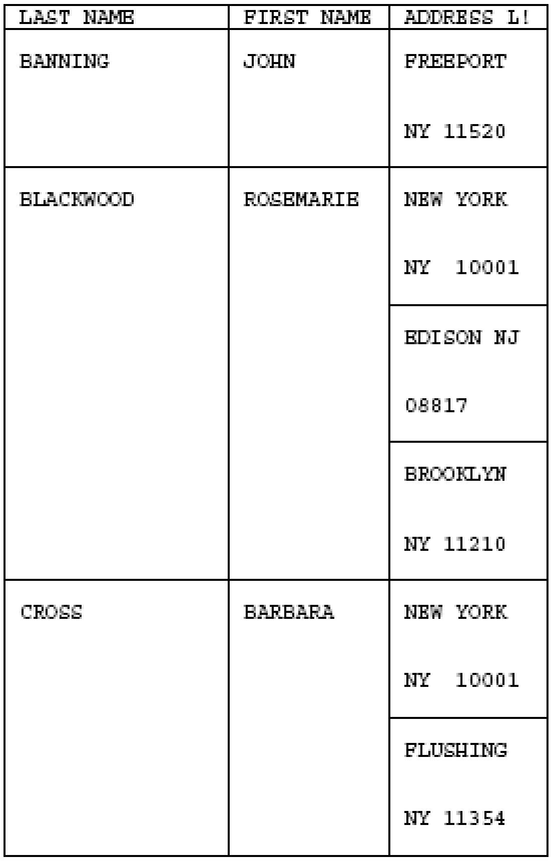
Report Request

```
TABLE FILE EMPLOYEE
PRINT ADDRESS_LN3
BY LAST_NAME BY FIRST_NAME
WHERE LAST_NAME LE 'CROSS'
ON TABLE PCHOLD FORMAT PDF
ON TABLE SET STYLE *
INCLUDE=IBFS:/stylesheet_location/custom_stylesheet.sty,$
END
```

***custom_stylesheet*.sty**

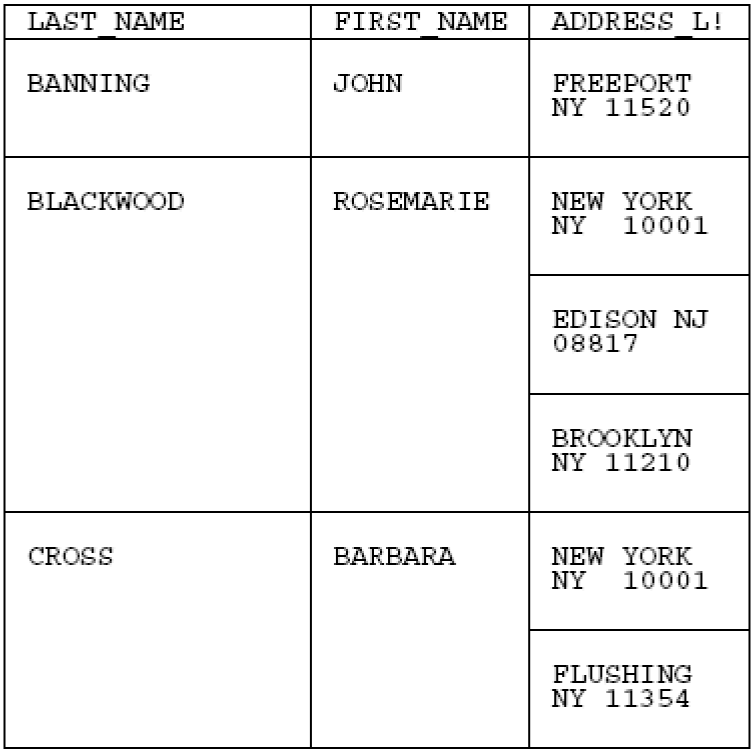
```
TYPE=REPORT, GRID=ON, $
TYPE=REPORT, COLUMN=ADDRESS_LN3, WRAP=1.0, $
TYPE=DATA, TOPGAP=0.2, BOTTOMGAP=0.2, $
TYPE=DATA, WRAPGAP=OFF, $
```

With WRAPGAP=OFF, each wrapped line is placed on the next report line.



LAST NAME	FIRST NAME	ADDRESS L!
BANNING	JOHN	FREEPORT NY 11520
BLACKWOOD	ROSEMARIE	NEW YORK NY 10001
		EDISON NJ 08817
		BROOKLYN NY 11210
CROSS	BARBARA	NEW YORK NY 10001
		FLUSHING NY 11354

With WRAPGAP=ON, the wrapped lines are placed directly under each other.



LAST_NAME	FIRST_NAME	ADDRESS_L!
BANNING	JOHN	FREEPORT NY 11520
BLACKWOOD	ROSEMARIE	NEW YORK NY 10001
		EDISON NJ 08817
		BROOKLYN NY 11210
CROSS	BARBARA	NEW YORK NY 10001
		FLUSHING NY 11354

Reference: Usage Notes for WRAPGAP

You can only specify WRAPGAP for columns that have wrapping enabled (WRAP attribute set to ON). The TOPGAP and BOTTOMGAP attributes specify how much vertical space to leave above and below a report component. Increasing the values or these attributes makes a decrease in spacing between wrapped lines more noticeable.

Justifying Report Columns

You can adjust text within a column by specifying whether report columns are left-justified, right-justified, or centered. By default, alphanumeric columns are left-justified, numeric columns are right-justified, and heading and footing elements are left-justified. However, you can change the default using the JUSTIFY attribute. For information on justifying column titles using /R /L and /C, see *Using Headings, Footings, Titles, and Labels* on page 231.

Syntax: How to Justify a Report Column

To left-justify, right-justify, or center a column, use the following syntax within a StyleSheet.

```
TYPE=type, [subtype,] [COLUMN=column,] JUSTIFY=option, $
```

where:

type

Is the report component you wish to affect, such as REPORT, HEADING, or TITLE.

subtype

Is any additional attribute, such as COLUMN, ACROSS, or ITEM, that is needed to identify the report component that you are formatting. For more information about how to specify different report components, see *Identifying a Report Component in a Web Query StyleSheet* on page 57.

column

Is the column you wish to justify. This attribute is only necessary if you wish to justify a specific column or set of columns. Omitting this attribute justifies the entire report.

option

Is the justification you wish to select:

- **LEFT,** which specifies that the column will be left-justified.
- **RIGHT,** which specifies that the column will be right-justified.
- **CENTER,** which specifies that the column will be centered.

Note: JUSTIFY is not supported with WRAP=ON or WRAP=*n*.

Example: Justifying Data in a Report Column

The following illustrates how to center the data in the Vendor Name column. The header is also center justified.

Report Request

```
TABLE FILE GGPRODS
HEADING
"PRODUCT REPORT"
SUM UNITS BY PRODUCT_DESCRIPTION BY PRODUCT_ID BY VENDOR_NAME
ON TABLE SET STYLE *
INCLUDE=IBFS:/stylesheet_location/custom_stylesheet.sty,$
END
```

***custom_stylesheet*.sty**

```
TYPE=REPORT, GRID=OFF, $
TYPE=REPORT, COLUMN=VENDOR_NAME, JUSTIFY=CENTER, $
TYPE=HEADING, JUSTIFY=CENTER, $
```

The output is:

```
               PRODUCT REPORT
               Product                           Unit
Product        Code        Vendor Name           Price

Biscotti       F102     Delancey Bakeries        17.00
Coffee Grinder G110      Appliance Craft        125.00
Coffee Pot     G121      Appliance Craft        140.00
Croissant      F103     West Side Bakers         28.00
French Roast   B142   European Specialities,     81.00
Hazelnut       B141     Coffee Connection        58.00
Kona           B144    Evelina Imports, Ltd      76.00
Mug            G100    NY Ceramic Supply         26.00
Scone          F101   Ridgewood Bakeries         13.00
Thermos        G104      ThermoTech, Inc         96.00
```

Data Visualization

To make HTML reports more powerful, you can insert visual representations of selected data directly into the report output. These visual representations are in the form of vertical or horizontal bar graphs that make relationships and trends among data more obvious.

Reference: Formatting Options for Data Visualization Bar Graphs

You can specify optional formatting attributes for data visualization bar graphs in the GRAPHTYPE declaration. The following table lists the formatting attributes and a description of each.

Formatting Attribute	Description
GRAPHCOLOR	Sets the color of the bar graphs.

Formatting Attribute	Description
GRAPHLENGTH	Sets the length of the longest bar graph. The value for GRAPHLENGTH determines the length in measurement units (inches, centimeters, and so on) of the longest bar graph in a vertical or horizontal bar graph. The length value is expressed in the current units, which is set using the UNITS StyleSheet attribute. The GRAPHLENGTH value is then converted into pixels.
GRAPHWIDTH	Sets the width of the bar graphs. The width value is expressed in the current units. See GRAPHLENGTH, above, for more information about units.

Note: Graphcolor=orange is not supported.

Syntax: How to Incorporate Data Visualization Formatting Attributes

```
GRAPHTYPE=DATA, {COLUMN|ACROSSCOLUMN|FIELD}=identifier,
    [GRAPHCOLOR=graphcolor,] [GRAPHLENGTH=lengthvalue,]
    [GRAPHWIDTH=widthvalue,] $
```

where:

GRAPHCOLOR

Specifies the color of the bar graphs. Black is the default color if you omit this attribute from the declaration.

graphcolor

Is one of the following values:

AQUA	NAVY
BLACK	OLIVE
BLUE	PURPLE
FUCHSIA	RED
GRAY	SILVER
GREEN	TEAL
LIME	WHITE
MAROON	YELLOW

GRAPHLENGTH

Specifies the length of the longest bar graph. The default length is 60 pixels for a vertical bar graph and 80 pixels for a horizontal bar graph.

`lengthvalue`

Sets the value used to display the vertical or horizontal bar graph for the maximum data value in the associated report column. This value must be a positive number.

This value is initially expressed in the current units (using the UNITS attribute). This value is then converted into the corresponding number of pixels.

`GRAPHWIDTH`

Specifies the width of the bar graphs in a report.

`widthvalue`

Sets the value used to display the width of the bar graphs in a report. This value must be a positive number.

This value is initially expressed in the current units (defined by the UNITS attribute). This value is then converted into the corresponding number of pixels.

Displaying Multi-Line An and AnV Fields

Using StyleSheet attributes, you can display An (character) and AnV (varchar) fields that contain line breaks on multiple lines in a PDF report. Line breaks can be based on line-feeds, carriage-returns, or a combination of both. If you do not add these StyleSheet attributes, all line-feed and carriage-return formatting within these fields will be ignored, and all characters will be displayed on one line that wraps to fit the width of the report.

Syntax: How to Display An and AnV Fields Containing Line Breaks on Multiple Lines

```
TYPE=REPORT, LINEBREAK='type', $
```

where:

`REPORT`

Is the type of report component. TYPE must be REPORT. Otherwise, an error will result.

`'`*`type`*`'`

Specifies that line breaks will be inserted in a report based on the following:

`LF` inserts a line break after each line-feed character found in all An and AnV fields.

`CR` inserts a line break after each carriage-return character found in all An and AnV fields.

`LFCR` inserts a line break after each combination of a line-feed character followed by a carriage-return character found in all An and AnV fields.

`CRLF` inserts a line break after each combination of a carriage-return character followed by a line-feed character found in all An and AnV fields.

Note: The report output must be formatted as PDF.

Example: Displaying an Alphanumeric Field With Line Breaks in a PDF Report

The following example defines an alphanumeric field named ANLB with a semicolon (;) in an EDCDIC environment, or a circumflex in an ASCII environment, in the middle. The CTRAN function then replaces the semicolon or circumflex with a carriage return character and stores this string in a field named ANLBC. On the report output, this field displays on two lines.

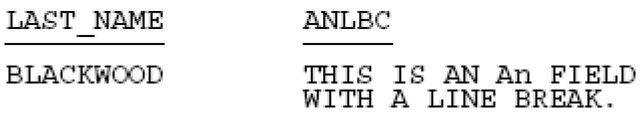
Report Request

```
DEFINE FILE EMPLOYEE
ANLB/A40 ='THIS IS AN An FIELD;WITH A LINE BREAK.';
ANLBC/A40 = CTRAN(40, ANLB, 094, 013  , ANLBC);
END
TABLE FILE EMPLOYEE
PRINT LAST_NAME ANLBC
WHERE LAST_NAME EQ 'BLACKWOOD'
ON TABLE HOLD FORMAT PDF
ON TABLE SET STYLE *
INCLUDE=IBFS:/stylesheet_location/custom_stylesheet.sty,$
END
```

***custom_stylesheet*.sty**

```
TYPE=REPORT, LINEBREAK='CR', $
```

The output is:



```
LAST_NAME          ANLBC

BLACKWOOD          THIS IS AN An FIELD
                   WITH A LINE BREAK.
```

Example: Using an Alphanumeric Field With a Line Break in a Subfoot

The following request defines an alphanumeric named ANLB field with a semicolon (;) in the middle. The CTRAN function then replaces the semicolon (;) in an EDCDIC environment, or a circumflex in an ASCII environment, with a carriage return character and stores this string in a field named ANLBC. In the subfoot, this field displays on two lines.

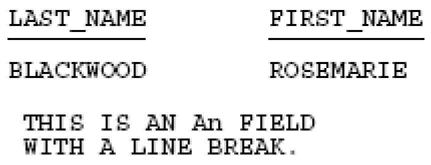
Report Request

```
DEFINE FILE EMPLOYEE
ANLB/A40 ='THIS IS AN An FIELD;WITH A LINE BREAK.';
ANLBC/A40 = CTRAN(40, ANLB, 094, 013  , ANLBC);
END
TABLE FILE EMPLOYEE
PRINT FIRST_NAME
BY LAST_NAME
WHERE LAST_NAME EQ 'BLACKWOOD'
ON LAST_NAME SUBFOOT
  " "
  " <ANLBC "
ON TABLE HOLD FORMAT PDF
ON TABLE SET STYLE *
INCLUDE=IBFS:/stylesheet_location/custom_stylesheet.sty,$
END
```

***custom_stylesheet*.sty**

```
TYPE=REPORT, LINEBREAK='CR', $
```

The output is:



```
LAST_NAME          FIRST_NAME

BLACKWOOD          ROSEMARIE

 THIS IS AN An FIELD
 WITH A LINE BREAK.
```

Chapter 10

Choosing a Display Format

You can choose from several different display formats when you display a report on the screen. Some display formats are best suited for particular kinds of uses. For example, you can choose to display the report as:

- An HTML page, which is optimized for display in a web browser.
- A PDF document, which is useful when you want the report to look the same whether displayed on a screen or printed.
- A DHTML file, which is HTML output that has most of the features normally associated with output formatted for printing, such as PDF output.
- An Excel worksheet, where you can work with data in Excel.
- A PowerPoint presentation, where you can work with data to create PowerPoint formatted output.

You can learn which display formats are available in *Report Display Formats* on page 315.

Note: InfoAssist has built-in styling options that generate some of the StyleSheet syntax described in this chapter.

In this chapter:

Report Display Formats

You can choose from among several display formats for your report:

- **Web format:** HTML. You can display a report as an HTML page. HTML supports most style sheet options (especially when used with an internal cascading style sheet), allowing for full report formatting.

- **Print formats:** PDF (Adobe Acrobat Portable Document Format). For more information, see *Using Print Display Formats: PDF* on page 316.
- **Worksheet formats:** Excel 2007/2010 XML-based format, Excel 2000/2003 HTML-based format, with variations for Excel 2000 FORMULA and Excel binary format. For more information, see *Saving Report Output in Excel XLSX Format* on page 320.
- **Display Formats:** PowerPoint 2007, 2010, and 2013 PPTX format. For more information, see *Saving Report Output in PPTX Format* on page 376.

A note about DHTML and HTML: DHTML is the absolute positioning version of HTML. As architected, format HTML generates output in a table-based format that leaves the exact positioning to the browser that is presenting the report. Format DHTML, on the other hand, is designed to render with the user-defined positioning in the same way as PDF. This means objects should position on the page precisely as defined in the report procedure. PDF, DHTML, and PPTX are position-based. HTML and EXL2K are table-based or cell-based. Therefore, DHTML output looks more like PDF rather than HTML.

Using Print Display Formats: PDF

PDF (Adobe Acrobat Portable Document Format) is most often used to distribute and share electronic documents through the web. It is especially useful if you want a report to maintain its presentation and layout regardless of a browser or printer type. For details, see *Using PDF Display Format* on page 316.

You can combine multiple styled reports into a single PDF file. For details, see *Laying Out the Report Page* on page 123.

PDF reports, including compound reports, can be distributed using Report Broker. See the Report Broker documentation for details.

Using PDF Display Format

You can display a report as a PDF (Adobe Acrobat Portable Document Format) document. PDF supports most StyleSheet attributes, allowing for full report formatting. The wide range of StyleSheet features supported for PDF are described throughout this documentation.

PDF prints and displays a document consistently, regardless of the application software, hardware, and operating system used to create or display the document.

The report opens in Adobe Acrobat or Acrobat Reader within a web browser. To display a PDF report, a computer must have Adobe Acrobat Reader installed. For free downloads of Acrobat Reader, go to *http://www.adobe.com*.

Limit: Adobe Acrobat PDF format limits the number of pages, hyperlinks, and images in a document.

Displaying Watermarks in PDF Output

Watermarks are images or text strings that are placed on the bottom layer of a document and displayed through the transparent layered content.

Web Query backcolor does not support transparency. Therefore, standard images placed below it on the page may be obscured. To resolve this, in PDF reports, Web Query mirrors the approach taken by standard printers, and places an opaque image on the top of the document layers. With this approach, the layers of the document will be visible beneath the transparent watermark image.

Watermark images are provided by the report developer. When creating a transparent image, the image needs to be created in GIF format with a transparent background.

Reference: Inserting Images in PDF Reports With Backcolor

Watermarks are supported for PDF output in compound reports (InfoAssist Documents) and in single TABLE requests. Each document supports a single active watermark image. This image is designated as the watermark image, by defining the placement order within the Z-INDEX attribute.

The first image with a Z-INDEX value will be considered the active watermark for the current document. Any subsequent images, defined with style sheet attributes for Z-INDEX or OPACITY, will be displayed as standard Web Query images.

Watermark images are designated by defining the following attributes in the style sheet for the transparent GIF.

Z-INDEX=TOP	Designates that the image is to be handled as a watermark image and should always be placed on top of all other objects on the page. This value will be respected as the top-most layer and will be supported with other layers in future releases.
OPACITY=*n*	Where *n* represents the percent (%) of OPACITY to be applied to the image. The greater the OPACITY, the less transparent the image. Less of the underlying report will be visible below the image. The value for *n* can be any number from 0 through 100. If a value is not specified, it defaults to 100%, presenting a fully opaque image.

In a Report Request

```
TYPE=[REPORT|HEADING[, OBJECT=IMAGE, IMAGE=<image.gif>,
   Z-INDEX=TOP, OPACITY=15, POSITION=(.25 .25), DIMENSION=(8 10.5), $
```

Example: Inserting Transparent Images Into a PDF Report

The following request against the GGSALES data source places the coffee image (coffee.gif) on the page and layers the watermark image (watermark.gif) on top. These images are displayed on every page of the report.

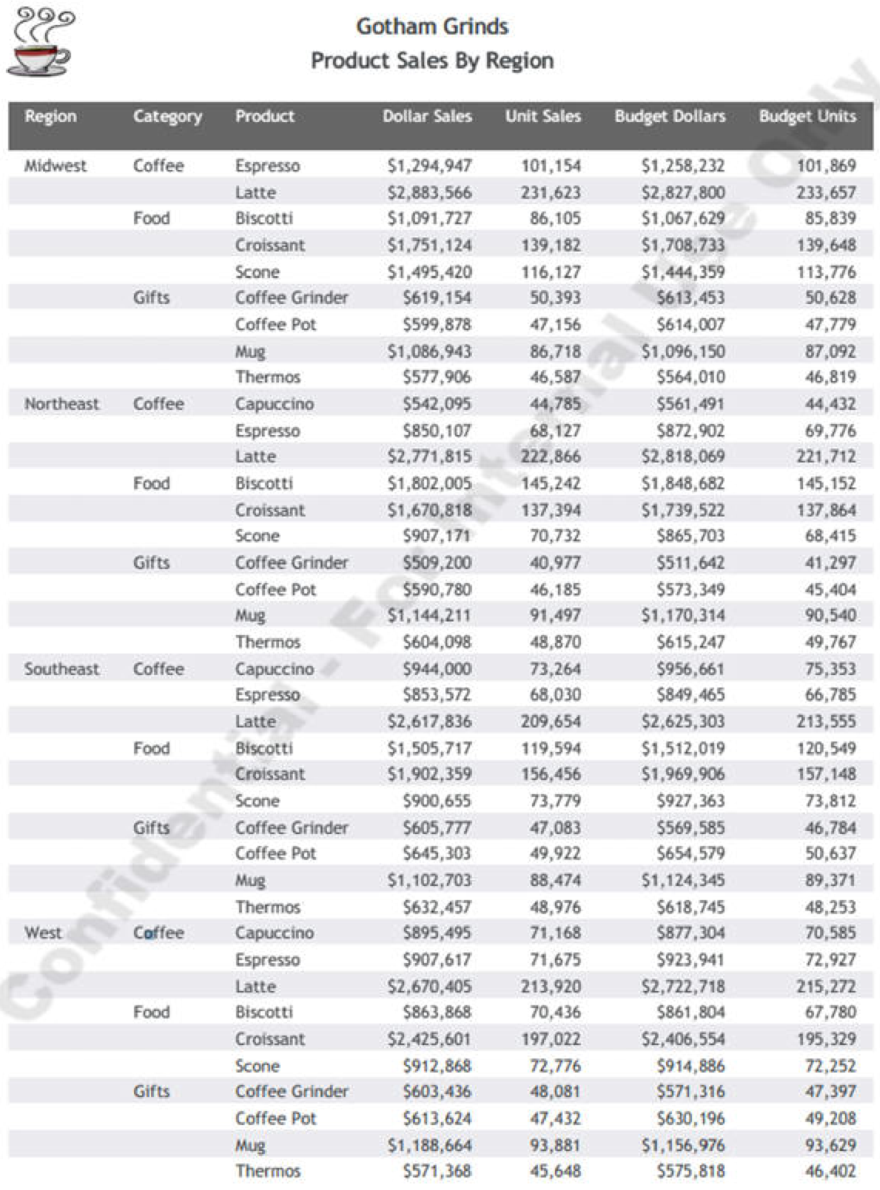
Report Request

```
TABLE FILE GGSALES
SUM
GGSALES.SALES01.DOLLARS/D12CM
GGSALES.SALES01.UNITS/D12C
GGSALES.SALES01.BUDDOLLARS/D12CM
GGSALES.SALES01.BUDUNITS/D12C
BY GGSALES.SALES01.REGION
BY GGSALES.SALES01.CATEGORY
BY GGSALES.SALES01.PRODUCT
HEADING
"Gotham Grinds"
"Product Sales By Region"
ON TABLE NOTOTAL
ON TABLE PCHOLD FORMAT PDF
ON TABLE SET STYLE *
INCLUDE=IBFS:/stylesheet_location/custom_stylesheet.sty,$
END
```

***custom_stylesheet*.sty**

```
TYPE=REPORT, TOPMARGIN=.5, BOTTOMMARGIN=.5, LEFTMARGIN=1,
   RIGHTMARGIN=1, $
TYPE=REPORT, OBJECT=IMAGE, IMAGE=watermark.gif,
   POSITION=(+0.70000 +0.70000), SIZE=(7 7.5), Z-INDEX=TOP, OPACITY=15, $
TYPE=REPORT, OBJECT=IMAGE, IMAGE=coffee.gif, POSITION=(+1.0 +0.5),
   SIZE=(.5 .5), $
```

The output is:



Gotham Grinds
Product Sales By Region

Region	Category	Product	Dollar Sales	Unit Sales	Budget Dollars	Budget Units
Midwest	Coffee	Espresso	$1,294,947	101,154	$1,258,232	101,869
		Latte	$2,883,566	231,623	$2,827,800	233,657
	Food	Biscotti	$1,091,727	86,105	$1,067,629	85,839
		Croissant	$1,751,124	139,182	$1,708,733	139,648
		Scone	$1,495,420	116,127	$1,444,359	113,776
	Gifts	Coffee Grinder	$619,154	50,393	$613,453	50,628
		Coffee Pot	$599,878	47,156	$614,007	47,779
		Mug	$1,086,943	86,718	$1,096,150	87,092
		Thermos	$577,906	46,587	$564,010	46,819
Northeast	Coffee	Capuccino	$542,095	44,785	$561,491	44,432
		Espresso	$850,107	68,127	$872,902	69,776
		Latte	$2,771,815	222,866	$2,818,069	221,712
	Food	Biscotti	$1,802,005	145,242	$1,848,682	145,152
		Croissant	$1,670,818	137,394	$1,739,522	137,864
		Scone	$907,171	70,732	$865,703	68,415
	Gifts	Coffee Grinder	$509,200	40,977	$511,642	41,297
		Coffee Pot	$590,780	46,185	$573,349	45,404
		Mug	$1,144,211	91,497	$1,170,314	90,540
		Thermos	$604,098	48,870	$615,247	49,767
Southeast	Coffee	Capuccino	$944,000	73,264	$956,661	75,353
		Espresso	$853,572	68,030	$849,465	66,785
		Latte	$2,617,836	209,654	$2,625,303	213,555
	Food	Biscotti	$1,505,717	119,594	$1,512,019	120,549
		Croissant	$1,902,359	156,456	$1,969,906	157,148
		Scone	$900,655	73,779	$927,363	73,812
	Gifts	Coffee Grinder	$605,777	47,083	$569,585	46,784
		Coffee Pot	$645,303	49,922	$654,579	50,637
		Mug	$1,102,703	88,474	$1,124,345	89,371
		Thermos	$632,457	48,976	$618,745	48,253
West	Coffee	Capuccino	$895,495	71,168	$877,304	70,585
		Espresso	$907,617	71,675	$923,941	72,927
		Latte	$2,670,405	213,920	$2,722,718	215,272
	Food	Biscotti	$863,868	70,436	$861,804	67,780
		Croissant	$2,425,601	197,022	$2,406,554	195,329
		Scone	$912,868	72,776	$914,886	72,252
	Gifts	Coffee Grinder	$603,436	48,081	$571,316	47,397
		Coffee Pot	$613,624	47,432	$630,196	49,208
		Mug	$1,188,664	93,881	$1,156,976	93,629
		Thermos	$571,368	45,648	$575,818	46,402

Features Supported

The following core PDF features are supported with watermarks:

- Standard reporting language requests, including reports with paneling
- Drilldown
- Bookmarks
- Borders/backcolor

Limits

- OPACITY must be between 0 and 100, inclusive.
- A single watermark image is supported for a single document.

Usage Notes

- The first watermark image found in the syntax will be used for the report. Any other watermark images found in the code will be ignored or displayed as standard Web Query images.

Saving Report Output in Excel XLSX Format

With Excel® 2007, Microsoft® introduced enhanced spreadsheet functionality in a new workbook file format. Using Web Query, you can retrieve data from any Web Query supported data source and generate a native XLSX format (Excel 2013, Excel 2010, and Excel 2007) workbook for data analysis and distribution.

The Web Query XLSX format supports the following Microsoft Office software products:

- Microsoft Office 2013/2010/2007 and Microsoft Office 2000/2003 with the Microsoft Office Compatibility Pack.
- Open Office Support (FORMAT XLSX). Core Excel functionality generated by the XLSX format is supported for Open Office as of Web Query 2.1.x. For details on Open Office, see *http://www.openoffice.org/*.

Web Query generates XLSX workbooks based on the Microsoft XLSX standard. These workbooks are accessible through all browsers and mobile applications that support native Microsoft XLSX files.

Note: This section, which describes important XLSX features, applies to Excel 2013, Excel 2010, and Excel 2007, unless otherwise indicated.

Building the .xlsx Workbook File

Microsoft changed the format and structure of the Excel workbook in Excel 2007. The new .xlsx file is a binary compilation of a group of xml files. Generating this new file format using Web Query is a two-step process that consists of generating the xml files containing the report output and zipping the xml documents into the binary .xlsx format. The Reporting Server performs the xml generation process. The zipping process can be completed either by the client (Web Query Servlet) or the server (JSCOM3):

- ❑ **Web Query Servlet.** The Web Query Client within the application server performs the zipping process. This can be done within the local client or through a remotely accessed client.
- ❑ **JSCOM3.** The Java layer of the Reporting Server performs the zipping operation. This option should be used when the Web Query Servlet is configured on a secured web or application server. This is because JSCOM3 does not require URL access to a remote Web Query Client. JSCOM3 is the default for Web Query 2.2.1.

Overview of XLSX Format

The Web Query procedure generates a new workbook containing a single worksheet with the report output containing your defined report elements (headings and subtotals), as well as StyleSheet syntax (such as conditional styling and drill downs):

- ❑ You can define a new default font for XLSX in the fontuser.xml file. This allows ease in customizing the look and feel of XLSX workbooks. Web Query uses Arial as the default font. However, you can change the default font to match the Microsoft Office standard font, Calibri, or your corporate standard.
- ❑ XLSX format accurately displays formatted numeric, character, and date formats.
- ❑ XLSX FORMULA enables you to convert summed information (such as column totals, row totals, and calculated values) into Excel formulas that will automatically update as you edit the Excel worksheet.
- ❑ Report Broker supports distribution of XLSX workbooks and XLSX FORMULA workbooks.

❑ Within each generated worksheet, the columns in the report are automatically sized to fit the largest value in the column (SQUEEZE=ON). Web Query calculates the width of each data column based on the font and size requirement of all cells in that column using font metrics developed for other styled formats, including PDF and DHTML. Calculations are based on the data and title elements of the report. Heading and footing elements are not used in the sizing calculation and will be sized based on the data column requirements.

❑ By default, there is a standard height for the data and Title rows. Heading, Footing, Subhead, and Subfoot rows are taller than the data rows to support wrapping and for a clearer distinction between headings and data.

❑ Using the TITLETEXT StyleSheet attribute, tab names within the workbook can be customized to provide better descriptions of the worksheet content.

❑ Unlike the HTML-based (EXL2K) format, which removes all blanks, XLSX, by default, retains leading, internal, and trailing blanks in cells within the worksheet.

❑ An XLSX worksheet can contain 1,048,576 rows by 16,384 columns. Web Query will generate worksheets larger than these defined limits, but Excel is not able to open the workbook.

❑ Because of the new format of the zipped XLSX files, native HTML symbols, such as a caret (<), cannot be supported as tag characters. For XLSX, unlike other output formats, HTMLENCODE defaults to ON. HTMLENCODE set to OFF will cause any data containing HTML tag characters to be omitted from the cell.

Opening XLSX Report Output

To open XLSX workbooks, Excel 2013, 2010, or 2007 must be installed on the desktop.

Reference: Opening XLSX Report Output in Excel 2000/2003

Excel 2000 and Excel 2003 can be updated to read Excel XLSX workbooks using the Microsoft Office Compatibility Pack available from the Microsoft download site (*http://www.microsoft.com/downloads/en/default.aspx*). When the file extension of the file being opened is .xlsx (XLSX workbook), the Microsoft Office Compatibility Pack performs the necessary conversion to allow Excel 2000/2003 to read and open it.

In addition to the Microsoft Office Compatibility Pack, it is important to enable the Web Query Client Redirection Settings *Save As* option so that Excel 2000/2003 will be able to open the XLSX report output without users first having to save it to their machine with the .xlxs file extension. The Web Query Client processing Redirection Settings *Save As* option configures how the Web Query Client sends each report output file type to the user machine. This option can be set as follows:

❑ **Save As Option disabled (NO).** The Web Query Client Redirection Setting *Save As* is disabled by default. When the *Save As* option is disabled, the Web Query Client sends report output to the user machine in memory with the application association specified for the report format in the Web Query Client Redirection Settings configuration file (mime.wfs).

  A user machine that does not have Excel 2007/2010 installed will not recognize the application association for Excel 2007/2010 and Excel will display a message.

  The Excel 2000/2003 user can select *Save* and provide a file name with the .xlsx extension to save the report output to their machine. The user can then open the .xlsx file directly from Excel 2000/2003.

- **Save As Option enabled (YES).** When the Web Query Redirection *Save As* option is enabled, the Web Query Client sends the report output to the user as a file with the extension specified in the Web Query Client Redirection Settings configuration file (mime.wfs).

  Upon receiving the file, Windows will display the File Download prompt asking the user to Open or Save the file with the identified application type. The File Download prompt displays the Name with the .xlsx file extension for the report output that is recognized as an Excel XLSX file type.

  Note: The download prompt will display for all users, including users who have Excel 2007/2010 installed on their machines.

  If an Excel 2000/2003 user chooses to open the file, the Microsoft Office Compatibility Pack will recognize the .xlsx file extension and perform the necessary conversion to allow Excel 2000/2003 to read the Excel XLSX workbook.

  If an Excel 2007/2010 user chooses to open the file, Excel will recognize the .xlsx file extension and read the Excel XLSX workbook.

Reference: Viewing Excel Workbooks in the Browser or Excel Application

Your Operating System and desktop settings determine whether Excel output sent to the client is displayed in an Internet Browser window or within the Excel application. When Excel output has been defined within the Windows environment to *Browse in same window*, the workbook generated by a Web Query request is opened within an Internet Explorer® browser window. When the *Browse in same window* option is not selected for the .xls file type, the browser window created by Web Query is blank because the report output is displayed in the stand-alone Excel application window.

- In Windows 7, Microsoft removed the desktop settings that support opening worksheets in the browser. This means that to change this behavior, you can no longer simply navigate to the Folder Options dialog box, but that you must change a registry setting. This change is documented in the Microsoft Knowledge Base Article ID 927009 at the following website:

  http://support.microsoft.com/kb/927009

Note: This works the same for both EXL2K and XLSX formats. The only difference is the selection of file type based on the version of Excel output you will be generating.

Formatting Values Within Cells in XLSX Report Output

Web Query formats defined in Master Files or within a FOCEXEC will be represented in the resulting cells in an Excel XLSX worksheet. Where possible, the Web Query formats are translated to custom Excel formats and applied to values passed as raw data. Each data value passed to a cell in Excel is defined with a value and a format mask pair. The data format is associated with the cell rather than embedded in the value. This technique provides enhanced support for editing worksheets generated by Web Query. New values entered into existing cells will retain the cell formats and continue to display in the style defined for the column within the report.

The following types of data can be passed to Excel:

- **Numeric.** Where corresponding Excel format masks can be defined, numeric values are passed as raw values with associated format masks. In instances where an equivalent format mask cannot be defined, the numeric value is passed as a text string.

- **Alphanumeric.** Alphanumeric formats are passed to Excel as text strings, with General format defined. By default, General format presents all text fields as left-justified. Alignment and other styling attributes can be applied to these cells to override the default.

- **Date formats.** Data that contain sufficient elements to define a valid Excel date format are passed as raw date values with the Web Query formats translated to Excel date format masks. In Web Query formats that do not contain sufficient information to create valid Excel date values, the dates are converted to text strings.

- **Date-Time formats.** Date-time values are passed as raw date-time values with Web Query formats translated to Excel date-time format masks using Custom formats.

- **Text.** Text values are passed as strings with General Format defined (as with alphanumeric data).

Note: This behavior is a change from EXL2K format, where cells containing dates and more complex numeric formats were passed as formatted text.

Displaying Formatted Numeric Values in XLSX Report Output

Each numeric Web Query format is translated to a custom numeric Excel format. The numeric value is displayed in the Excel formula bar for the selected cell. Within the actual cell, the value with the format mask applied displays.

The Web Query formats for the following numeric data types are translated into Excel XLSX format masks supporting full editing within the resulting workbook:

- Data types: E, F, D, I, P
- Comma edit option (C)
- Zero suppression (S)
- Leading zero (L)
- Floating currency symbol (M)
- Comma suppression (c)
- Right-side minus sign (-)
- Credit negative (CR)
- Bracket negative (B)
- Fixed extended currency symbol (!d, !e, !l, !y)
- Floating extended currency symbol (!D, !E, !L, !Y)
- Percent (%)

Example:

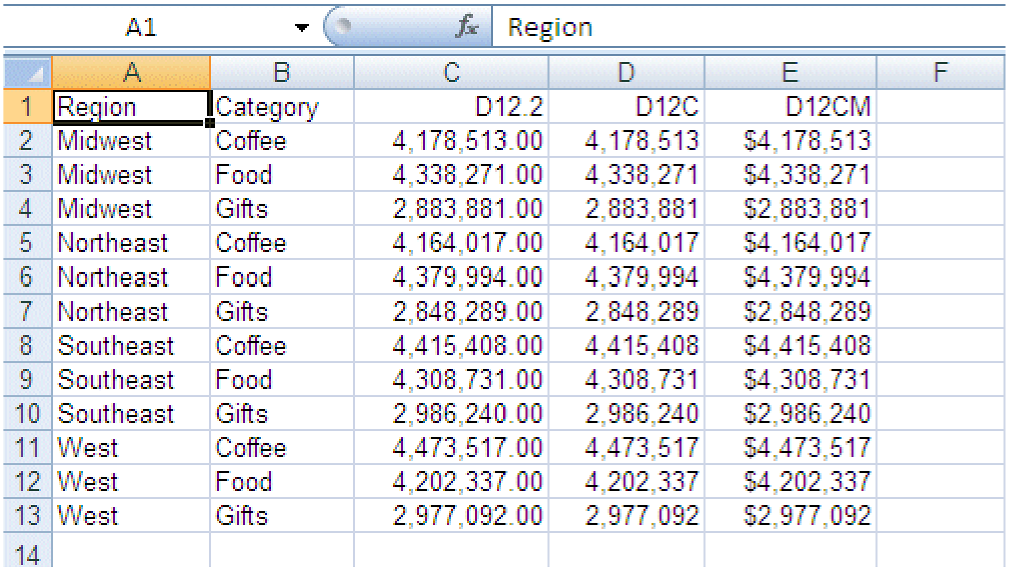
Passing Numeric Formats to XLSX Report Output

The following example illustrates how to assign the DOLLARS field to different numeric formats to demonstrate different available options. The column titles have been edited to display the Web Query format options that have been applied.

```
TABLE FILE GGSALES
SUM DOLLARS/D12.2 AS 'D12.2'
    DOLLARS/D12C  AS 'D12C'
    DOLLARS/D12CM AS 'D12CM'
BY REGION
BY CATEGORY
ON TABLE PCHOLD FORMAT XLSX
ON TABLE SET BYDISPLAY ON
END
```

In the resulting worksheet, notice that cell C2 containing the DOLLAR value for *Midwest Coffee* presents the value with the Web Query format D12.2, which presents the comma (,) and two decimal places. On the formula bar, the actual value is presented without any formatting. Examine each of the DOLLAR values in each row to see that the value as displayed in the formula bar remains the same, and only the display values presented in each cell change.

Also notice that with SET BYDISPLAY ON, the BY field values are repeated for every row on the worksheet. This creates fully qualified data rows that can be used with various data sorting, filtering, and table features in Excel without losing valuable information. This setting is recommended as a best practice for all worksheets.

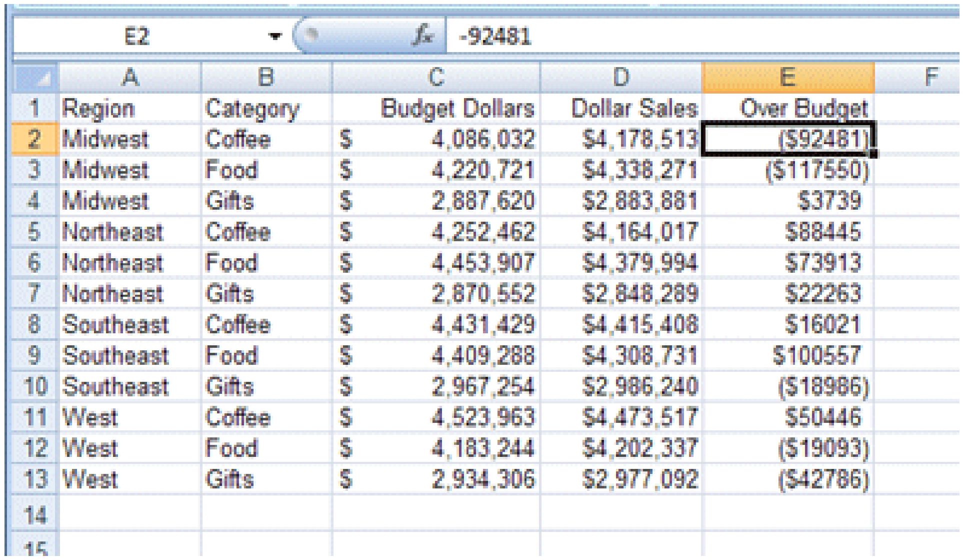

A1 Region

	A	B	C	D	E	F
1	Region	Category	D12.2	D12C	D12CM	
2	Midwest	Coffee	4,178,513.00	4,178,513	$4,178,513	
3	Midwest	Food	4,338,271.00	4,338,271	$4,338,271	
4	Midwest	Gifts	2,883,881.00	2,883,881	$2,883,881	
5	Northeast	Coffee	4,164,017.00	4,164,017	$4,164,017	
6	Northeast	Food	4,379,994.00	4,379,994	$4,379,994	
7	Northeast	Gifts	2,848,289.00	2,848,289	$2,848,289	
8	Southeast	Coffee	4,415,408.00	4,415,408	$4,415,408	
9	Southeast	Food	4,308,731.00	4,308,731	$4,308,731	
10	Southeast	Gifts	2,986,240.00	2,986,240	$2,986,240	
11	West	Coffee	4,473,517.00	4,473,517	$4,473,517	
12	West	Food	4,202,337.00	4,202,337	$4,202,337	
13	West	Gifts	2,977,092.00	2,977,092	$2,977,092	
14						

The following example illustrates how to use a Fixed Dollar (N) format, as well as multiple combined format options. Each Web Query format option is translated to the appropriate Excel XLSX format mask and applied to the cell value.

```
TABLE FILE GGSALES
SUM BUDDOLLARS/D12N
    DOLLARS/D12M
COMPUTE OVERBUDGET/D12BMc = BUDDOLLARS-DOLLARS; AS 'Over Budget'
BY REGION
BY CATEGORY
ON TABLE PCHOLD FORMAT XLSX
ON TABLE SET BYDISPLAY ON
END
```

Notice the fixed numeric format defined for the BUDDOLLARS column (Column C) presents the local currency symbol in a fixed position within each cell, regardless of the size of the data value. On the formula bar, the values in the *Over Budget* calculated field is passed as a negative value where appropriate. In the actual cells, the bracketed styling is applied to the negative values as part of the custom Excel XLSX format mask.



E2 -92481

	A	B	C	D	E	F
1	Region	Category	Budget Dollars	Dollar Sales	Over Budget	
2	Midwest	Coffee	$ 4,086,032	$4,178,513	($92481)	
3	Midwest	Food	$ 4,220,721	$4,338,271	($117550)	
4	Midwest	Gifts	$ 2,887,620	$2,883,881	$3739	
5	Northeast	Coffee	$ 4,252,462	$4,164,017	$88445	
6	Northeast	Food	$ 4,453,907	$4,379,994	$73913	
7	Northeast	Gifts	$ 2,870,552	$2,848,289	$22263	
8	Southeast	Coffee	$ 4,431,429	$4,415,408	$16021	
9	Southeast	Food	$ 4,409,288	$4,308,731	$100557	
10	Southeast	Gifts	$ 2,967,254	$2,986,240	($18986)	
11	West	Coffee	$ 4,523,963	$4,473,517	$50446	
12	West	Food	$ 4,183,244	$4,202,337	($19093)	
13	West	Gifts	$ 2,934,306	$2,977,092	($42786)	
14						
15						

Using Numeric Formats in Report Headings and Footings

By default, headings and footings are passed to Excel as a single character string. Spot markers are not supported for positioning within each line. Numeric fields and dates passed in headings and footings are passed as text strings within the overall heading or footing contents.

To display numeric fields and dates within headings and footings as numeric or date values, use HEADALIGN=BODY in the StyleSheet to define each of the items in the heading as an individual cell. Each cell containing numeric or date values will then be passed as the appropriate value with the associated format mask.

Using Numeric Format Punctuation in Headings and Footings

For data columns, all currency formats are translated using the Excel XLSX format masks that use the punctuation rules defined by the regional settings of the desktop.

In languages that use Continental Decimal Notation, the currency definitions designate that a comma (,) is used as the decimal separator, and a period (.) is used as the thousands separator, so D12.2CM may present the value as $ 9.999,99 rather than the English (United States) value $ 9,999.99. In headings and footings, you can designate that punctuation should be converted to Continental Decimal notation by issuing the SET CDN=ON command. With this setting in effect, the data embedded within heading and footing text strings will be formatted using the converted punctuation. Specify HEADALIGN=BODY to delineate items as individual cells and to retain the numeric formatting within the field, which will follow the same rules as the report data within the data columns.

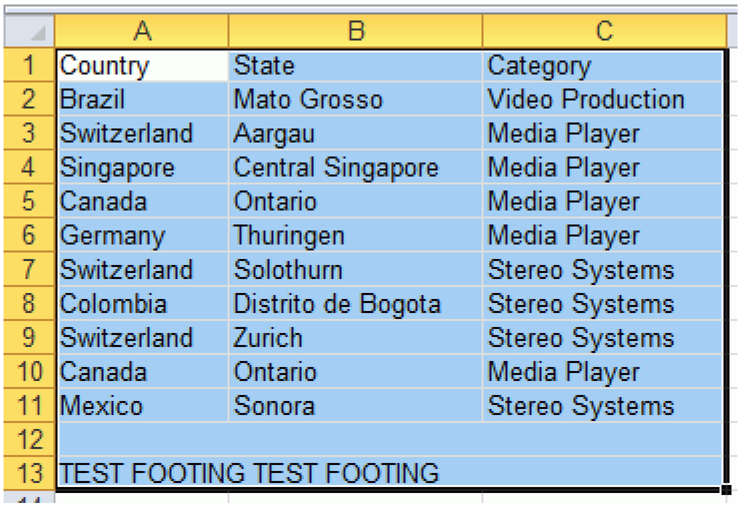
Reference: Usage Note for Sorting an XSLX Report That Contains a Footing

In XLSX format, the report footer is included as a part of the data table in Excel. This is not the same behavior in EXL2K. In EXL2K format, the footer is not included as a part of the table.

For example, if you run the following request and sort the data table, the report footer is part of the data table.

```
TABLE FILE WF_RETAIL_LITE
PRINT COUNTRY_NAME AS Country
STATE_PROV_NAME AS State
PRODUCT_CATEGORY AS Category
WHERE RECORDLIMIT EQ 10
FOOTING
""
"TEST FOOTING TEST FOOTING"
ON TABLE PCHOLD FORMAT XLSX
END
```

The output is:



	A	B	C
1	Country	State	Category
2	Brazil	Mato Grosso	Video Production
3	Switzerland	Aargau	Media Player
4	Singapore	Central Singapore	Media Player
5	Canada	Ontario	Media Player
6	Germany	Thuringen	Media Player
7	Switzerland	Solothurn	Stereo Systems
8	Colombia	Distrito de Bogota	Stereo Systems
9	Switzerland	Zurich	Stereo Systems
10	Canada	Ontario	Media Player
11	Mexico	Sonora	Stereo Systems
12			
13	TEST FOOTING TEST FOOTING		

The workaround is to add a named data range, as shown in the following request.

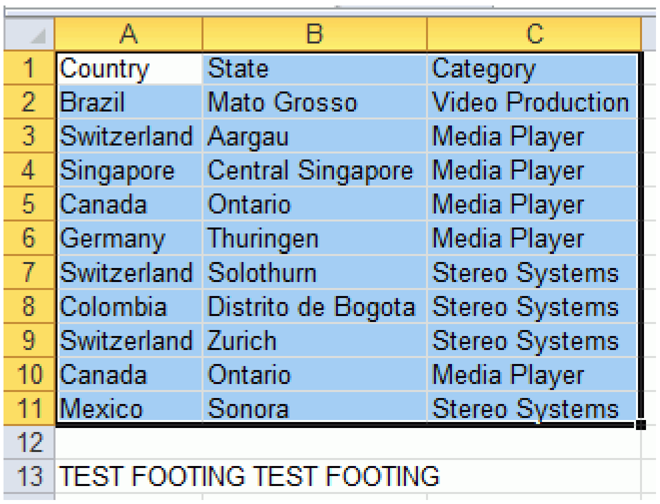
Report Request

```
TABLE FILE WF_RETAIL_LITE
PRINT COUNTRY_NAME AS Country
STATE_PROV_NAME AS State
PRODUCT_CATEGORY AS Category
WHERE RECORDLIMIT EQ 10
FOOTING
""
"TEST FOOTING TEST FOOTING"
ON TABLE PCHOLD FORMAT XLSX
ON TABLE SET STYLE *
INCLUDE=IBFS:/stylesheet_location/custom_stylesheet.sty,$
END
```

***custom_stylesheet*.sty**

```
TYPE=DATA, IN-RANGES=DATA, $
TYPE=TITLE, IN-RANGES=DATA, $
```

The report footer is not part of the data table, as shown in the following image.



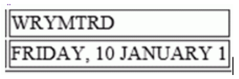
	A	B	C
1	Country	State	Category
2	Brazil	Mato Grosso	Video Production
3	Switzerland	Aargau	Media Player
4	Singapore	Central Singapore	Media Player
5	Canada	Ontario	Media Player
6	Germany	Thuringen	Media Player
7	Switzerland	Solothurn	Stereo Systems
8	Colombia	Distrito de Bogota	Stereo Systems
9	Switzerland	Zurich	Stereo Systems
10	Canada	Ontario	Media Player
11	Mexico	Sonora	Stereo Systems
12			
13	TEST FOOTING TEST FOOTING		

Passing Dates to XLSX Report Output

Most translated and smart dates can be sent to Excel as standard date values with format masks, enabling Excel to use them in functions, formulas, and sort sequences.

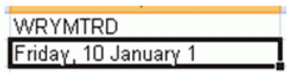
Excel 2007 only supports mixed-case date text strings so all month and day names are displayed in mixed-case, regardless of how the case has been specified in the Web Query format. For example, the Web Query date format WRYMTRD presents the date text information in uppercase in all non-Excel formats. Excel transforms this value to mixed-case automatically.

In HTML, the date format displays as:



WRYMTRD
FRIDAY, 10 JANUARY 1

In XLSX, the date format displays as:



WRYMTRD
Friday, 10 January 1

Example: Translating Web Query Dates to Excel XLSX Dates

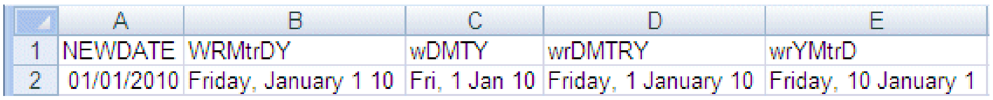
The following request against the GGSALES data source creates the date January 1, 2010 and converts it to four date formats with translated text.

```
DEFINE FILE GGSALES
NEWDATE/MDYY = '01/01/2010';
WRMtrDY/WRMtrDY = NEWDATE;
wDMTY/wDMTY = NEWDATE;
wrDMTRY/wrDMTRY = NEWDATE;
wrYMtrD/wrYMtrD = NEWDATE;
END
TABLE FILE GGSALES
SUM DATE NOPRINT
NEWDATE WRMtrDY wDMTY wrDMTRY wrYMtrD
ON TABLE PCHOLD FORMAT XLSX
END
```

The following table shows how the dates should appear.

Web Query Format	Web Query Display	XLSX Display	XLSX Value
WRMtrDY	FRIDAY, January 1 10	Friday, January 1 10	1/1/2010
wDMTY	Fri, 1 JAN 10	Fri, 1 Jan 10	1/1/2010
wrDMTY	Friday, 1 JANUARY 10	Friday, 1 January 10	1/1/2010
wrYMtrD	FRIDAY, 10 JANUARY 1	Friday, 10 January 1	1/1/2010

In Excel 2007/2010, all of the cells have a date value with format masks, and all month and day names are in mixed-case, regardless of how the case has been specified in the Web Query format. The output is:



	A	B	C	D	E
1	NEWDATE	WRMtrDY	wDMTY	wrDMTRY	wrYMtrD
2	01/01/2010	Friday, January 1 10	Fri, 1 Jan 10	Friday, 1 January 10	Friday, 10 January 1

Passing Dates Without a Day Component

Date formats that do not specify the day value explicitly are defined as the date value of the first day of the month. Therefore, the value placed in the cell may be different from the day component value in the source data field and may produce unexpected results when used for sorting or date calculations in an Excel formula.

The following table shows how Web Query date formats are represented in XLSX. The table shows how the value is preserved in the cell and how the display is generated using the format mask that corresponds to the Web Query date format.

```
DATEFLD/MDYY = '01/02/2010'
```

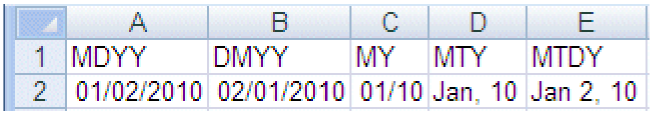
Web Query Format	XLSX Display	XLSX Value
DMYY	02/01/2010	1/2/2010
MY	01/10	1/1/2010
MTY	Jan, 10	1/1/2010
MTDY	Jan 2, 10	1/2/2010

Example: Passing Web Query Dates With and Without a Day Component to XLSX Report Output

The following request against the GGSALES data source creates the date January 2, 2010 and passes it to Excel with formats MDYY, DMYY, MY, and MTDY.

```
DEFINE FILE GGSALES
NEWDATE/MDYY = '01/02/2010';
END
TABLE FILE GGSALES
SUM DATE NOPRINT
NEWDATE AS 'MDYY' NEWDATE/DMYY AS 'DMYY' NEWDATE/MY AS 'MY'
NEWDATE/MTY AS 'MTY' NEWDATE/MTDY AS 'MTDY'
ON TABLE PCHOLD FORMAT XLSX
END
```

Columns D and E have actual date values with format masks, displayed by Excel 2007/2010 in mixed-case. Since the MTY format does not have a day component, the date value stored is the first of January 2010 (1/1/2010), not the second of January 2010 (1/2/2010).



	A	B	C	D	E
1	MDYY	DMYY	MY	MTY	MTDY
2	01/02/2010	02/01/2010	01/10	Jan, 10	Jan 2, 10

Passing Date Components for Use in Excel Formulas

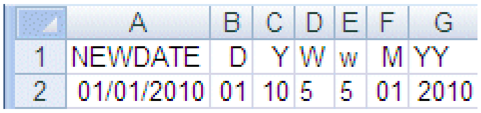
Dates formatted as individual components (for example, D, Y, M, W) are passed to Excel as numeric values that can be used as parameters to Excel date functions. The values are passed as General format that are recognized by Excel as numbers.

Example: Passing Numeric Date Components to XLSX Report Output

The following request against the GGSALES data source creates the date January 1, 2010 and extracts numeric date components, passing them to Excel 2007/2010.

```
DEFINE FILE GGSALES
NEWDATE/MDYY = '01/01/2010';
D/D = NEWDATE;
Y/Y = NEWDATE;
W/W = NEWDATE;
w/w = NEWDATE;
M/M = NEWDATE;
YY/YY = NEWDATE;
END
TABLE FILE GGSALES
SUM DATE NOPRINT
NEWDATE D Y W w M YY
ON TABLE PCHOLD FORMAT XLSX
END
```

The output is:



	A	B	C	D	E	F	G
1	NEWDATE	D	Y	W	w	M	YY
2	01/01/2010	01	10	5	5	01	2010

Passing Quarter Formats

Date formats that contain a Quarter component are always passed to Excel as text strings since Excel does not support Quarter formats.

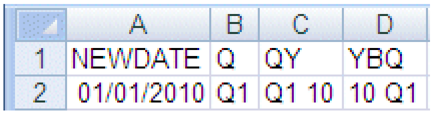
Example: Passing Dates With a Quarter Component to XLSX Report Output

The following request against the GGSALES data source creates the date January 1, 2010 and converts it to date formats that contain a Quarter component.

```
DEFINE FILE GGSALES
NEWDATE/MDYY = '01/01/2010';
Q/Q = NEWDATE;
QY/QY = NEWDATE;
YBQ/YBQ = NEWDATE;
END
TABLE FILE GGSALES
SUM DATE NOPRINT
NEWDATE Q QY YBQ
ON TABLE PCHOLD FORMAT XLSX
END
```

In XLSX, the cells containing dates with Quarter components have General format. To see this, open the Format Cells dialog box.

The output is:



	A	B	C	D
1	NEWDATE	Q	QY	YBQ
2	01/01/2010	Q1	Q1 10	10 Q1

Passing Date Components Defined as Translated Text

Date formats that do not contain sufficient information to present a valid date result in Excel are not translated to a value, including formats that do not contain year and/or month information. These dates will be sent to Excel as text. In the absence of complete information, the year defaults to the current year, so the value sent would be incorrect if this type of format was passed as a date value. The following formats will not be sent as values:

- MT, MTR, Mt, Mtr
- W, w, WR, wr

When date formats are passed to XLSX with format masks, all month and day names are in mixed-case, regardless of how the case has been specified in the Web Query format. However, since the values in this example are always sent as text, the casing defined in the Web Query format is applied in the resulting cell.

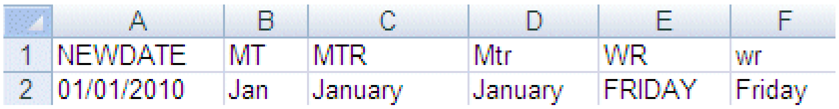
Example: **Passing Date Components Defined as Translated Text to XLSX Report Output**

The following request against the GGSALES data source creates the date January 1, 2010 and converts it to date formats that are defined as either month name or day name.

```
DEFINE FILE GGSALES
NEWDATE/MDYY = '01/01/2010';
MT/MT = NEWDATE;
MTR/MTR = NEWDATE;
Mtr/Mtr = NEWDATE;
WR/WR = NEWDATE;
wr/wr = NEWDATE;
END
TABLE FILE GGSALES
SUM DATE NOPRINT
NEWDATE MT MTR Mtr WR wr
ON TABLE PCHOLD FORMAT XLSX
END
```

In Excel 2007 or 2010, the cells containing the days have General format. To see this, open the Format Cells dialog box.

The output is:



	A	B	C	D	E	F
1	NEWDATE	MT	MTR	Mtr	WR	wr
2	01/01/2010	Jan	January	January	FRIDAY	Friday

Reference: **Usage Notes for Date Values in XLSX Report Output**

The following date formats are not supported in XLSX. They will translate into Excel General format and possibly produce unpredictable results:

- JUL, YYJUL, and I2MT.
- Dates stored as a packed or alphanumeric field with date display options.

Passing Date-Time to XLSX

Most Web Query date-time formats can be sent to XLSX as standard date/time values with format masks, enabling Excel to use them in functions, formulas, and sort sequences.

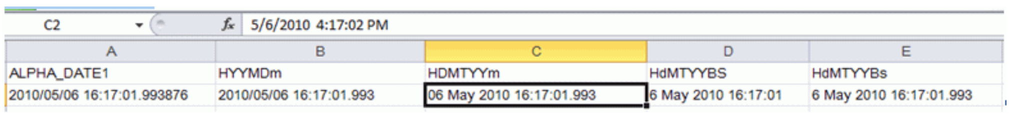
As with the Date formats, Excel only supports mixed-case to date-time fields, so if the date-time format contains text and is supported by Excel, the text will be in mixed-case, regardless of the casing defined within the Web Query format.

Example: **Passing Date-Time to XLSX**

The following request against the GGSALES data source illustrates how to pass a date-time format to XLSX.

```
DEFINE FILE GGSALES
DT1/HYYMDm WITH REGION = DT(20100506 16:17:01.993876);
DPT1/HDMTYYm = DT1;
ALPHA_DATE1/A30 = HCNVRT(DT1,'(HYYMDm)',30,'A30');
END
TABLE FILE GGSALES
PRINT
ALPHA_DATE1
DT1 AS 'HYYMDm'
DPT1 AS 'HDMTYYm'
DT1/HdMTYYBS AS 'HdMTYYBS'
DT1/HdMTYYBs AS 'HdMTYYBs'
ON TABLE SET SPACES 1
IF RECORDLIMIT EQ 1
ON TABLE PCHOLD FORMAT XLSX
END
```

The output is:



C2	5/6/2010 4:17:02 PM			
A	B	C	D	E
ALPHA_DATE1	HYYMDm	HDMTYYm	HdMTYYBS	HdMTYYBs
2010/05/06 16:17:01.993876	2010/05/06 16:17:01.993	06 May 2010 16:17:01.993	6 May 2010 16:17:01	6 May 2010 16:17:01.993

Note: Minutes by themselves are not supported in Excel and will be sent as an integer to XLSX with a Custom format.

Also, Excel time formats only support to the milliseconds. Web Query formats that display microseconds will send the value to Excel, but the value will be rounded to milliseconds within the worksheet if the cell is edited.

The following table shows how the date-time values appear.

Web Query Format	XLSX Displays	XLSX Value
HYYMDm	2010/05/06 16:17:01.993	5/6/2010 4:17:02 PM
HDMTYYm	06 May 2010 16:17:01.993	5/6/2010 4:17:02 PM
HdMTYYBS	6 May 2010 16:17:01	5/6/2010 4:17:01 PM
HdMTYYBs	6 May 2010 16:17:01.993	5/6/2010 4:17:02 PM

Generating Native Excel Formulas in XLSX Report Output

When you display or save a tabular report request using XLSX FORMULA, the resulting worksheet contains an Excel formula that computes and displays the results of any type of summed information, such as column totals, row totals, subtotals, and calculated values, rather than static numbers. A formula for a calculated value is generated by translating the internal form of the Web Query expression into an Excel formula. Worksheets saved using the XLSX FORMULA format are interactive, allowing for "what if" scenarios that immediately reflect any additions or modifications made to the data.

Understanding Formula Versus Value

The XLSX FORMULA format will generate formulas rather than values for the following Web Query TABLE commands: ROW-TOTAL, COLUMN-TOTAL, SUB-TOTAL, SUBTOTAL, and SUMMARIZE, as well as for calculations performed by functions.

- ❑ A DEFINE field will always generate a constant value and not a formula.
- ❑ COMPUTE will generate the formula, except when the COMPUTE is equal to a single variable. In that case, the constant is placed and not the formula.
- ❑ If your report contains a calculated value (generated by the COMPUTE or RECOMPUTE command), all of the fields referenced by the calculated value must be displayed in the report in order for a cell reference to be included in the formula. If the referenced column is not displayed in the workbook, the data value will be placed in the formula, rather than a cell reference. Additionally, if the value cannot be reliably calculated based on the information passed to Excel, the value, rather than an expression, will be used. For example, using the LAST function in Web Query cannot be translated correctly into Excel. In this instance, the LAST value is used in the expression, rather than a cell reference.

Reference: Translation Support for FORMAT XLSX FORMULA

This topic describes translation support for FORMAT XLSX FORMULA. Use of unsupported Web Query features may produce unreliable results.

- ❑ All standard operators are supported. These include arithmetic operators, relational operators, string operators, IF/THEN/ELSE, and logical operators. However, column notation is not supported.

  The IS-PRESENT, IS-MISSING, IS-FROM, FROM, NOT-FROM, IS-MORE-THAN, IS-LESS-THAN, CONTAINS, and OMITS operators are not supported.

  The logical operators AND and OR are not supported in conditional (IF-THEN-ELSE) or logical expressions.

- ❑ The following functions are supported:

  ABS, ARGLEN, ATODBL, BYTVAL, CHARGET, CTRAN, DMOD, DOWK, DOWK, DOWKL, EXP, FMOD, HEXBYT, HHMMSS, IMOD, LCWORD, LOCASE, LOG, MAX, MIN, OVRLAY, POSIT, RDUNIF, SQRT, SUBSTR, TODAY, and UPCASE. The EDIT function is supported for converting formats (one argument variant). It is not supported for editing strings.

  The functions CTRFLD, LJUST, and RJUST are not recommended for justifying data in Excel columns. With the use of Excel proportional fonts, the StyleSheet JUSTIFY attribute is more appropriate.

Be cautious when using functions that use decimal values as an argument (BYTVAL, CTRAN, HEXBYT). Based on whether the operating environment is EBCDIC or ASCII, the results may be different.

- XLSX FORMULA is not supported with the following Web Query commands and phrases:
  - DEFINE
  - OVER
  - FOR
  - NOPRINT
  - Multiple display (PRINT, LIST, SUM, and COUNT) commands
  - SEQUENCE StyleSheet attribute
  - RECAP
  - SET HIDENULLACRS
  - SET SUBTOTALS = ABOVE
  - LAST
- The BYDISPLAY ON setting is recommended to allow the sort field value to be available on all rows for recalculations.
- If an expression requires more than 1024 characters, Web Query will place the value into the cell, and not the formula.
- Conditional styling is based on the values in the original report. If the worksheet values are changed and the formulas are recomputed, the styling will not reflect the updated information.

Example: Generating Native Excel Formulas for Column Totals

The following example illustrates how a column total in a report request is translated to an Excel formula when you use the XLSX FORMULA format. Notice that the formatting of the column total (TYPE=GRANDTOTAL) is retained in the Excel workbook. When you select the total in the report, the equation =SUM(B4:B10) displays in the formula bar, representing the column total as a sum of cell ranges.

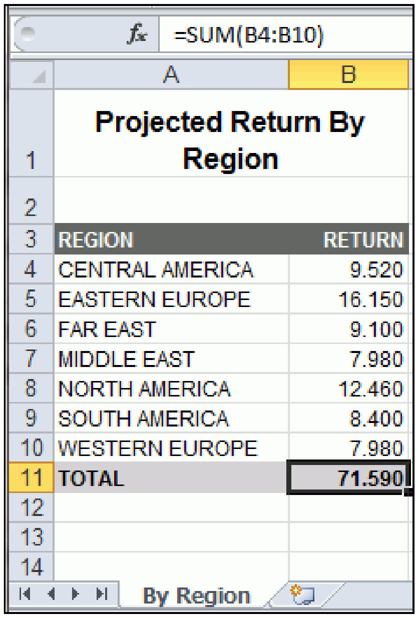
Report Request

```
TABLE FILE SHORT
HEADING
"Projected Return By Region"
" "
SUM PROJECTED_RETURN AS 'RETURN'
BY REGION AS 'REGION'
ON TABLE COLUMN-TOTAL
ON TABLE PCHOLD FORMAT XLSX FORMULA
ON TABLE SET STYLE *
INCLUDE=IBFS:/stylesheet_location/custom_stylesheet.sty,$
END
```

***custom_stylesheet*.sty**

```
TYPE=REPORT, GRID=OFF, FONT='ARIAL', SIZE=9, TITLETEXT=’By Region’, $
TYPE=TITLE, BACKCOLOR=RGB(102 102 102), COLOR=RGB(255 255 255), $
TYPE=HEADING, SIZE=12, STYLE=BOLD, JUSTIFY=CENTER, $
TYPE=GRANDTOTAL, BACKCOLOR=RGB(210 210 210), STYLE=BOLD, $
```

The output is:



=SUM(B4:B10)

	A	B
1	**Projected Return By Region**	
2		
3	REGION	RETURN
4	CENTRAL AMERICA	9.520
5	EASTERN EUROPE	16.150
6	FAR EAST	9.100
7	MIDDLE EAST	7.980
8	NORTH AMERICA	12.460
9	SOUTH AMERICA	8.400
10	WESTERN EUROPE	7.980
11	**TOTAL**	**71.590**
12		
13		
14		

By Region

Web Query can translate any total (subtotal, row total, or column total) to an Excel formula. For related information, see *Translation Support for FORMAT XLSX FORMULA* on page 338.

Example: Generating Native Excel Formulas for Row Totals

The following example illustrates how to calculate totals for returns and balances across continents. The row totals are represented as sums of cell ranges.

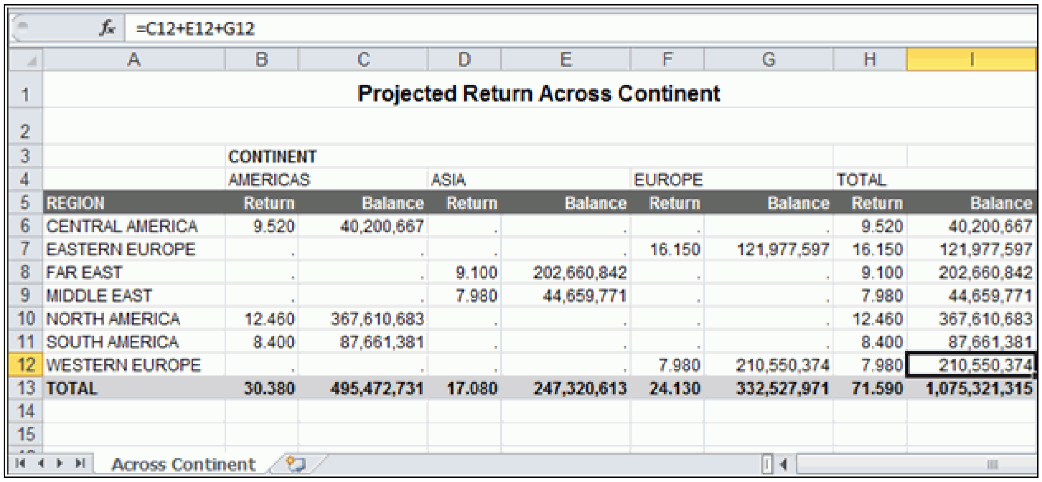
Report Request

```
TABLE FILE SHORT
HEADING
"Projected Return Across Continent"
" "
SUM PROJECTED_RETURN AS 'Return' AND BALANCE AS 'Balance'
ACROSS CONTINENT AS 'CONTINENT'
BY REGION AS 'REGION'
ON CONTINENT ROW-TOTAL AS 'TOTAL'
ON TABLE COLUMN-TOTAL AS 'TOTAL'
ON TABLE PCHOLD FORMAT XLSX FORMULA
ON TABLE SET STYLE *
INCLUDE=IBFS:/stylesheet_location/custom_stylesheet.sty,$
END
```

***custom_stylesheet*.sty**

```
TYPE=REPORT, GRID=OFF, FONT='ARIAL', SIZE=9, TITLETEXT='Across Continent', $
TYPE=TITLE, BACKCOLOR=RGB(102 102 102), COLOR=RGB(255 255 255), $
TYPE=HEADING, SIZE=12, STYLE=BOLD, JUSTIFY=CENTER, $
TYPE=ACROSSTITLE, STYLE=BOLD, $
TYPE=GRANDTOTAL, BACKCOLOR=RGB(210 210 210), STYLE=BOLD, $
```

The following output highlights the formula that calculates the row total in cell I12=C12+E12+G12.



=C12+E12+G12

	A	B	C	D	E	F	G	H	I
1	Projected Return Across Continent								
2									
3		CONTINENT							
4		AMERICAS		ASIA		EUROPE		TOTAL	
5	REGION	Return	Balance	Return	Balance	Return	Balance	Return	Balance
6	CENTRAL AMERICA	9.520	40,200,667	.	.	.	.	9.520	40,200,667
7	EASTERN EUROPE	.	.	.	.	16.150	121,977,597	16.150	121,977,597
8	FAR EAST	.	.	9.100	202,660,842	.	.	9.100	202,660,842
9	MIDDLE EAST	.	.	7.980	44,659,771	.	.	7.980	44,659,771
10	NORTH AMERICA	12.460	367,610,683	.	.	.	.	12.460	367,610,683
11	SOUTH AMERICA	8.400	87,661,381	.	.	.	.	8.400	87,661,381
12	WESTERN EUROPE	.	.	.	.	7.980	210,550,374	7.980	210,550,374
13	TOTAL	30.380	495,472,731	17.080	247,320,613	24.130	332,527,971	71.590	1,075,321,315
14									
15									

Across Continent

Example: Generating Native Excel Formulas for Calculated Values

The following example illustrates how to total the columns for retail cost and dealer cost, and calculate the value of a field called PROFIT by subtracting the DOLLARS field from the BUDDOLLARS field.

The formula for the calculated values is generated by translating the internal form of the Web Query expression (PROFIT/D12.2MC = BUDDOLLARS - DOLLARS;) into an Excel formula. In this example, the formulas appear in cells B8, C8, and D8.

All fields referenced in the calculation should be displayed in the report for a valid formula to be created using cell references. Otherwise, it may be created using values not in the report. If the fields used in the calculation are not present in the report and there is a subsequent RECOMPUTE, the formula created for the RECOMPUTE will not be correct.

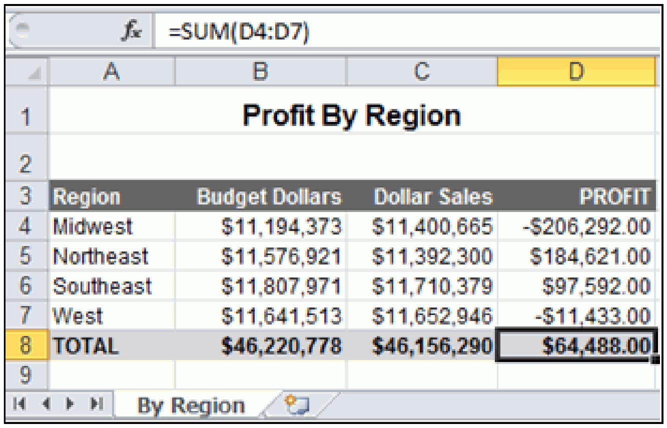
Report Request

```
TABLE FILE GGSALES
ON TABLE SET PAGE-NUM OFF
SUM BUDDOLLARS/I8MC AND DOLLARS/I8MC
COMPUTE PROFIT/D12.2MC = BUDDOLLARS - DOLLARS;
BY REGION
HEADING
"Profit By Region"
" "
ON TABLE COLUMN-TOTAL
ON TABLE PCHOLD FORMAT XLSX FORMULA
ON TABLE SET STYLE *
INCLUDE=IBFS:/stylesheet_location/custom_stylesheet.sty,$
END
```

***custom_stylesheet*.sty**

```
TYPE=REPORT, GRID=OFF, FONT='ARIAL', SIZE=9, TITLETEXT=’By Region’, $
TYPE=TITLE, BACKCOLOR=RGB(102 102 102), COLOR=RGB(255 255 255), $
TYPE=HEADING, SIZE=12, STYLE=BOLD, JUSTIFY=CENTER, $
TYPE=GRANDTOTAL, BACKCOLOR=RGB(210 210 210), STYLE=BOLD, $
```

The following output highlights the formula that calculates for the column total of PROFIT: D8=SUM(D4:D7).



=SUM(D4:D7)

	A	B	C	D
1	Profit By Region			
2				
3	Region	Budget Dollars	Dollar Sales	PROFIT
4	Midwest	$11,194,373	$11,400,665	-$206,292.00
5	Northeast	$11,576,921	$11,392,300	$184,621.00
6	Southeast	$11,807,971	$11,710,379	$97,592.00
7	West	$11,641,513	$11,652,946	-$11,433.00
8	TOTAL	$46,220,778	$46,156,290	$64,488.00
9				

By Region

Example:

Generating a Native Excel Formula for a Function

The following example illustrates how functions are translated to Excel reports. The function IMOD divides ACCTNUMBER by 1000 and returns the remainder to LAST3_ACCT. The Excel formula corresponds to =TRUNC((MOD($C3,(1000)))). TRUNC is used when the answer returned from an equation is being placed into an Integer field, to be sure there are no decimals.

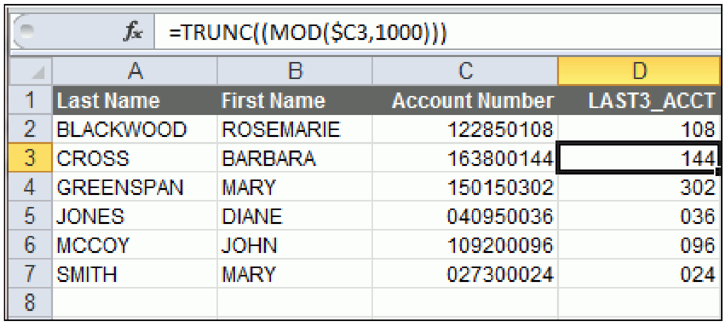
Report Request

```
TABLE FILE EMPLOYEE
PRINT ACCTNUMBER AS 'Account Number'
COMPUTE LAST3_ACCT/I3L = IMOD(ACCTNUMBER, 1000, LAST3_ACCT);
BY LAST_NAME AS 'Last Name'
BY FIRST_NAME AS 'First Name'
WHERE (ACCTNUMBER NE 000000000) AND (DEPARTMENT EQ 'MIS');
ON TABLE PCHOLD FORMAT XLSX FORMULA
ON TABLE SET STYLE *
INCLUDE=IBFS:/stylesheet_location/custom_stylesheet.sty,$
END
```

***custom_stylesheet*.sty**

```
TYPE=REPORT, GRID=OFF, FONT='ARIAL', SIZE=9, $
TYPE=TITLE, BACKCOLOR=RGB(102 102 102), COLOR=RGB(255 255 255),
   STYLE=BOLD, $
```

The output is:



=TRUNC((MOD($C3,1000)))

	A	B	C	D
1	Last Name	First Name	Account Number	LAST3_ACCT
2	BLACKWOOD	ROSEMARIE	122850108	108
3	CROSS	BARBARA	163800144	144
4	GREENSPAN	MARY	150150302	302
5	JONES	DIANE	040950036	036
6	MCCOY	JOHN	109200096	096
7	SMITH	MARY	027300024	024
8				

Reference: Generating a Formula With Recomputed Values

If your report contains a calculated value (generated by the COMPUTE or RECOMPUTE command), all of the fields referenced by the calculated value must be displayed in the report in order for cell references to be included in the formula. If a referenced column is not displayed in the workbook, the data value will be placed in the formula, rather than a cell reference. In the case of RECOMPUTE, the value used may be an incorrect value from the last detail record of the sort break.

Example: Generating a Formula With Recomputed Values

The following example illustrates how to compute the difference (DIFF) by subtracting budgeted dollars from dollar sales. The budgeted dollars field used in the expression is not included in the SUM command. The value of DIFF is recomputed on the region level.

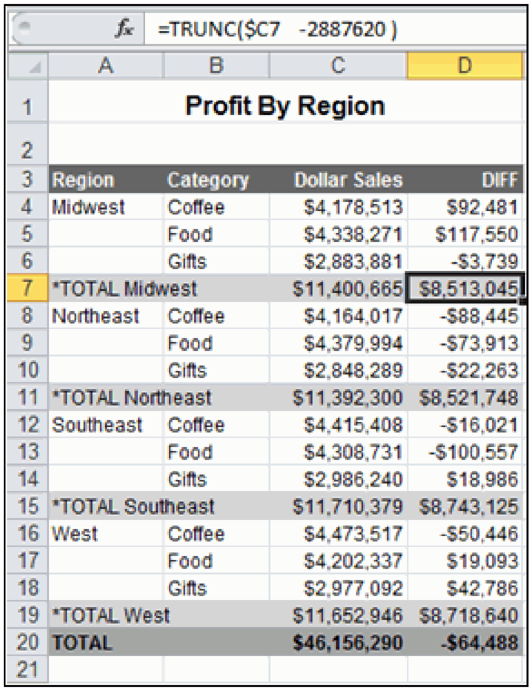
Report Request

```
TABLE FILE GGSALES
HEADING
"Profit By Region"
" "
SUM DOLLARS/I8CM
COMPUTE DIFF/I8CM=DOLLARS - BUDDOLLARS;
BY REGION
BY CATEGORY
ON REGION RECOMPUTE
ON TABLE PCHOLD FORMAT XLSX FORMULA
ON TABLE SET STYLE *
INCLUDE=IBFS:/stylesheet_location/custom_stylesheet.sty,$
END
```

***custom_stylesheet*.sty**

```
TYPE=REPORT, GRID=OFF, FONT='ARIAL', SIZE=9, TITLETEXT=’By Region’, $
TYPE=TITLE, BACKCOLOR=RGB(102 102 102), COLOR=RGB(255 255 255), $
TYPE=HEADING, SIZE=12, STYLE=BOLD, JUSTIFY=CENTER, $
TYPE=SUBTOTAL, BACKCOLOR=RGB(210 210 210), $
TYPE=GRANDTOTAL, BACKCOLOR=RGB(166 166 166), STYLE=BOLD, $
```

The output shows that the formula is subtracting a data value that is not displayed on the worksheet. It is actually the BUDDOLLARS value from the current hardcoded value, since there is no cell reference.

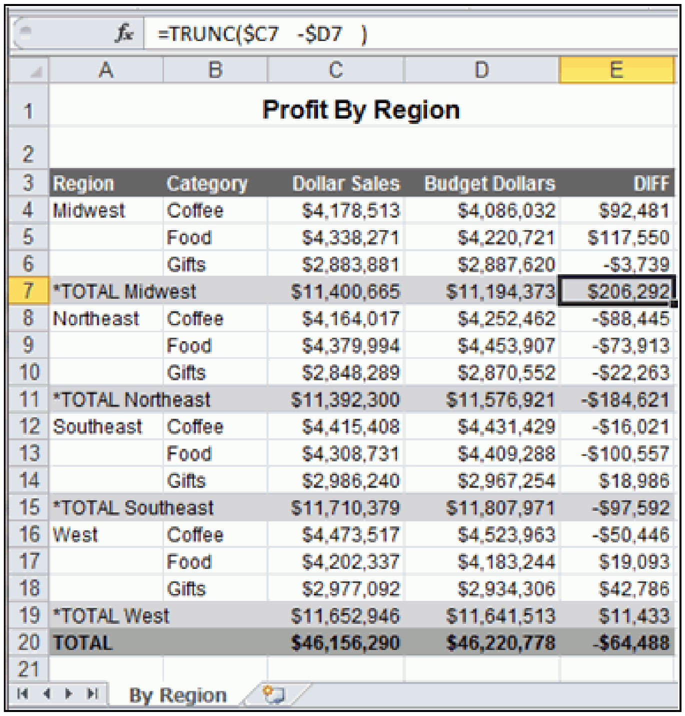

=TRUNC($C7 -2887620)

	A	B	C	D
1	**Profit By Region**			
2				
3	Region	Category	Dollar Sales	DIFF
4	Midwest	Coffee	$4,178,513	$92,481
5		Food	$4,338,271	$117,550
6		Gifts	$2,883,881	-$3,739
7	*TOTAL Midwest		$11,400,665	$8,513,045
8	Northeast	Coffee	$4,164,017	-$88,445
9		Food	$4,379,994	-$73,913
10		Gifts	$2,848,289	-$22,263
11	*TOTAL Northeast		$11,392,300	$8,521,748
12	Southeast	Coffee	$4,415,408	-$16,021
13		Food	$4,308,731	-$100,557
14		Gifts	$2,986,240	$18,986
15	*TOTAL Southeast		$11,710,379	$8,743,125
16	West	Coffee	$4,473,517	-$50,446
17		Food	$4,202,337	$19,093
18		Gifts	$2,977,092	$42,786
19	*TOTAL West		$11,652,946	$8,718,640
20	**TOTAL**		**$46,156,290**	**-$64,488**
21				

If you add the BUDDOLLARS column to the request, the formula can be recomputed correctly.

```
SUM DOLLARS/I8MC BUDDOLLARS/I8MC
```

The formula generated with the new SUM command contains cell references for both fields used in the calculation.



=TRUNC($C7 -$D7)

	A	B	C	D	E
1	Profit By Region				
2					
3	Region	Category	Dollar Sales	Budget Dollars	DIFF
4	Midwest	Coffee	$4,178,513	$4,086,032	$92,481
5		Food	$4,338,271	$4,220,721	$117,550
6		Gifts	$2,883,881	$2,887,620	-$3,739
7	*TOTAL Midwest		$11,400,665	$11,194,373	$206,292
8	Northeast	Coffee	$4,164,017	$4,252,462	-$88,445
9		Food	$4,379,994	$4,453,907	-$73,913
10		Gifts	$2,848,289	$2,870,552	-$22,263
11	*TOTAL Northeast		$11,392,300	$11,576,921	-$184,621
12	Southeast	Coffee	$4,415,408	$4,431,429	-$16,021
13		Food	$4,308,731	$4,409,288	-$100,557
14		Gifts	$2,986,240	$2,967,254	$18,986
15	*TOTAL Southeast		$11,710,379	$11,807,971	-$97,592
16	West	Coffee	$4,473,517	$4,523,963	-$50,446
17		Food	$4,202,337	$4,183,244	$19,093
18		Gifts	$2,977,092	$2,934,306	$42,786
19	*TOTAL West		$11,652,946	$11,641,513	$11,433
20	TOTAL		$46,156,290	$46,220,778	-$64,488
21					

Using XLSX FORMULA With Prefix Operators

XLSX FORMULA output supports prefix operators that are used on summary lines generated by Web Query commands, such as SUBTOTAL and RECOMPUTE. Where a corresponding formula exists in Excel, these prefix operators are translated into the equivalent Excel summarization formula. The results of prefix operators used directly against retrieved data continue to be passed to Excel as values, not formulas.

The following table identifies the prefix operators supported by XLSX FORMULA when used on summary lines, and the Excel formula equivalent placed in the generated worksheet.

Prefix Operator	Excel Formula Equivalent
SUM.	=SUM()
AVE.	=AVERAGE()

Prefix Operator	Excel Formula Equivalent
CNT.	=COUNT()
MIN.	=MIN()
MAX.	=MAX()

The following prefix operators are not translated to formulas when used on summary lines in XLSX FORMULA.

- ASQ.
- FST.
- LST.

Note:

- When using a prefix operator on a field specified directly against retrieved data, there is no space between the prefix operator and the field on which it operates.

  For example, in the following aggregating display command, the AVE. prefix operator operates on the DOLLARS field.

  ```
  SUM AVE.DOLLARS
  ```

- When using a prefix operator on a summary line, you must leave a space between the prefix operator and the aggregated field on which it operates.

  In the following summary command, the MAX. prefix operator operates on the DOLLARS field at the REGION sort break. Note the required blank space between the prefix operator and the field name.

  ```
  ON REGION RECOMPUTE MAX. DOLLARS
  ```

Example: Using a Summary Prefix Operator With FORMAT XLSX FORMULA

In the following request against the GGSALES data source, the RECOMPUTE command for the REGION sort field calculates the maximum of the aggregated DOLLARS field and the minimum of the aggregated BUDDOLLARS field.

Report Request

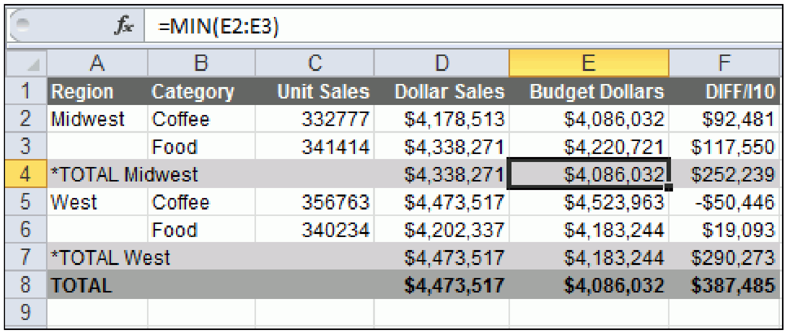
```
TABLE FILE GGSALES
SUM UNITS DOLLARS/I8MC BUDDOLLARS/I8MC
AND COMPUTE DIFF/I8MC= DOLLARS-BUDDOLLARS;
BY REGION
BY CATEGORY
WHERE CATEGORY EQ 'Food' OR 'Coffee'
WHERE REGION EQ 'West' OR 'Midwest'
ON REGION RECOMPUTE MAX. DOLLARS MIN. BUDDOLLARS DIFF
ON TABLE PCHOLD FORMAT XLSX FORMULA
ON TABLE SET STYLE *
INCLUDE=IBFS:/stylesheet_location/custom_stylesheet.sty,$
END
```

***custom_stylesheet*.sty**

```
TYPE=REPORT, GRID=OFF, FONT='ARIAL', SIZE=9, $
TYPE=TITLE, BACKCOLOR=RGB(102 102 102), COLOR=RGB(255 255 255), $
TYPE=SUBTOTAL, BACKCOLOR=RGB(210 210 210), $
TYPE=GRANDTOTAL, BACKCOLOR=RGB(166 166 166), STYLE=BOLD, $
```

In the output, shown in the following image, the cell that represents the recomputed DOLLARS for the Midwest region has been generated as the following formula.

```
=MIN(E2:E3)
```



=MIN(E2:E3)

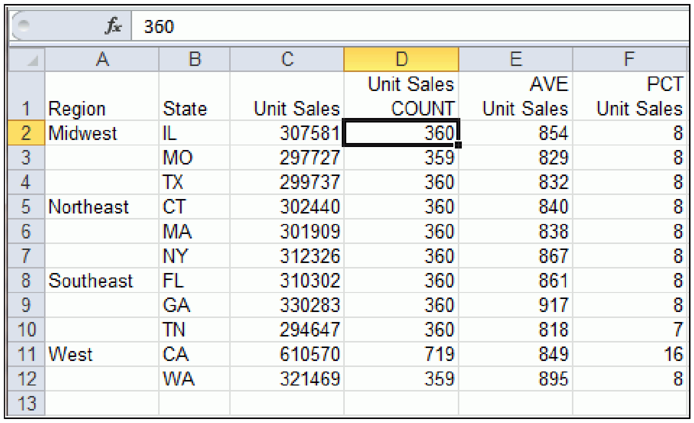
	A	B	C	D	E	F
1	Region	Category	Unit Sales	Dollar Sales	Budget Dollars	DIFF/I10
2	Midwest	Coffee	332777	$4,178,513	$4,086,032	$92,481
3		Food	341414	$4,338,271	$4,220,721	$117,550
4	*TOTAL Midwest			$4,338,271	$4,086,032	$252,239
5	West	Coffee	356763	$4,473,517	$4,523,963	-$50,446
6		Food	340234	$4,202,337	$4,183,244	$19,093
7	*TOTAL West			$4,473,517	$4,183,244	$290,273
8	**TOTAL**			**$4,473,517**	**$4,086,032**	**$387,485**
9						

Example: Using a Prefix Operator on a Display Command With FORMAT XLSX FORMULA

In the following request against the GGSALES data source, the CNT., AVE., and PCT. prefix operators are used in the SUM display command.

```
TABLE FILE GGSALES
SUM UNITS
CNT.UNITS
AVE.UNITS
PCT.UNITS
BY REGION
BY ST
ON TABLE PCHOLD FORMAT XLSX FORMULA
END
```

The output shows that the prefix operators were not passed to Excel as formulas. They were passed as data values.



fx 360

	A	B	C	D	E	F
1	Region	State	Unit Sales	Unit Sales COUNT	AVE Unit Sales	PCT Unit Sales
2	Midwest	IL	307581	360	854	8
3		MO	297727	359	829	8
4		TX	299737	360	832	8
5	Northeast	CT	302440	360	840	8
6		MA	301909	360	838	8
7		NY	312326	360	867	8
8	Southeast	FL	310302	360	861	8
9		GA	330283	360	917	8
10		TN	294647	360	818	7
11	West	CA	610570	719	849	16
12		WA	321469	359	895	8
13						

NODATA With Formulas

Support for full Excel functionality requires that only valid numeric values are placed into cells that will be used for formula references.

The null value (NODATA='') is supported for calculations. When cells containing the default NODATA symbol (.) are used in a formula, they will cause a formula error.

For example:

```
SET NODATA=''
TABLE FILE GGSALES
SUM DOLLARS/D12CM UNITS/D12C AND ROW-TOTAL AND COLUMN-TOTAL
COMPUTE REVENUE/D12CM=DOLLARS*UNITS; AS 'Revenue'
BY LOWEST GGSALES.SALES01.CATEGORY
BY GGSALES.SALES01.PRODUCT
ACROSS REGION
ON TABLE PCHOLD FORMAT XLSX FORMULA
END
------------------------
SET NODATA=''
DEFINE FILE GGSALES
DOLLARMOD/D12CM MISSING ON=IF REGION GT 'V' THEN MISSING ELSE DOLLAR;
END
TABLE FILE GGSALES
SUM DOLLARMOD/D12CM UNITS/D12C AND ROW-TOTAL AND COLUMN-TOTAL
COMPUTE REVENUE/D12CM=DOLLARMOD*UNITS; AS 'Revenue'
BY REGION
BY LOWEST GGSALES.SALES01.CATEGORY
BY GGSALES.SALES01.PRODUCT
ON TABLE SET PAGE-NUM NOLEAD
ON TABLE PCHOLD FORMAT XLSX FORMULA
END
```

Reference: Usage Notes for XLSX With Formulas

- Formulas are defined within a single worksheet. They will not be assigned across worksheets.

Controlling Column Width and Wrapping in XLSX Report Output

- Column width and data wrapping can be controlled in an Excel worksheet when using FORMAT XLSX.
- To size the column without wrapping and define the exact size width, use SQUEEZE=ON. If a data value is wider than the specified width of the column, a portion of the data will be hidden from view, but fully visible in the formula bar. You can adjust the column width in Excel after the worksheet has been generated.
- The default behavior is for all data to wrap within the defined column width. You can also specify the exact width of a column using WRAP=ON.
- WRAP is not supported for Date format fields.

Syntax: How to Set Column Width in XLSX Report Output

```
TYPE=REPORT, [COLUMN=column,] SQUEEZE=value,$
```

where:

`column`

Identifies a particular column. If COLUMN is not included in the declaration, default SQUEEZE behavior is applied to the entire report.

`value`

Is one of the following:

`ON`

Automatically sizes the columns based on the largest data value in the column. This is the default behavior.

`OFF`

Sizes the columns based on the maximum size defined for the field in the Master File or Define.

`n`

Represents a specific numeric value for which the column width can be set. The value represents the measure specified with the UNITS parameter (the default is inches). This is the most commonly used SQUEEZE setting in an XLSX report. This turns off data wrapping.

Note: SQUEEZE can be applied to the entire report by using the ON TABLE SET SQUEEZE ON command.

Syntax: How to Wrap Data in XLSX Report Output

```
TYPE=REPORT, [COLUMN=column,] WRAP=value,$
```

where:

`column`

Designates a particular column to apply wrapping behavior to. If COLUMN is not included in the declaration, wrapping will be applied to the entire report.

value

Is one of the following:

ON

Turns on data wrapping. ON is the default value. With this setting, the column width is determined by the client (Excel). Data wraps if it exceeds the width of the column and the row height expands to meet the new height of the wrapped data.

OFF

Turns off data wrapping. Data will not wrap in any cell in the column.

n

Represents a specific numeric value that the column width can be set to. The value represents the measure specified with the UNITS parameter (the default is inches).

This setting implies ON. However, the column width is set to the specified width unless the data is wider than the column width, in which case, wrapping will occur as for ON.

Note: WRAP is not supported for Date format fields.

Example: Controlling Column Width and Wrapping in XLSX Report Output

The following example illustrates how to turn on and turn off data wrapping in a column and how to set the column width for a particular column. The UNITS in this example are set to inches (the default).

Report Request

```
DEFINE FILE GGSALES
PROFIT/D14.3 = BUDDOLLARS-DOLLARS;
DESCRIPTION/A80 = 'Subtract Total Sales Quota from Reported Sales to
calculate profit.';
END


TABLE FILE GGSALES
SUM
DESCRIPTION AS 'DEFAULT'
DESCRIPTION AS 'WRAP = 2'
DESCRIPTION AS 'WRAP = OFF'
DESCRIPTION AS 'SQUEEZE = 1.5'
PROFIT
BY REGION NOPRINT
ON TABLE PCHOLD FORMAT XLSX
ON TABLE SET STYLE *
INCLUDE=IBFS:/stylesheet_location/custom_stylesheet.sty,$
END
```

***custom_stylesheet*.sty**

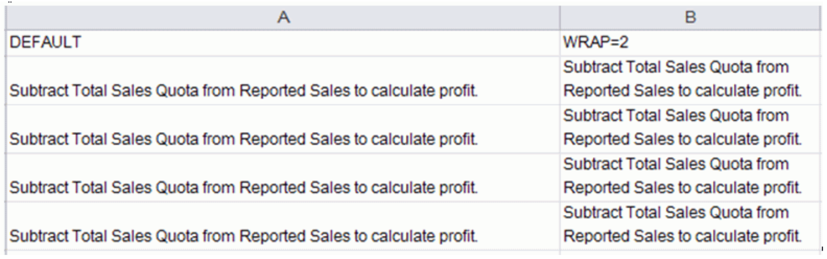
```
TYPE=REPORT, COLUMN=DESCRIPTION(2), WRAP=2, $
TYPE=REPORT, COLUMN=DESCRIPTION(3), WRAP=OFF, $
TYPE=REPORT, COLUMN=DESCRIPTION(4), SQUEEZE=1.5, $
```

where:

- ❑ The column titled "DEFAULT" illustrates the default column width and wrapping behavior.
- ❑ The column titled "WRAP=2" sets the column width to 2 inches with data wrapping on.
- ❑ The column titled "WRAP=OFF" turns off data wrapping for that column.
- ❑ The column titled "SQUEEZE=1.5" sets the column width to 1.5 inches with data wrapping off.

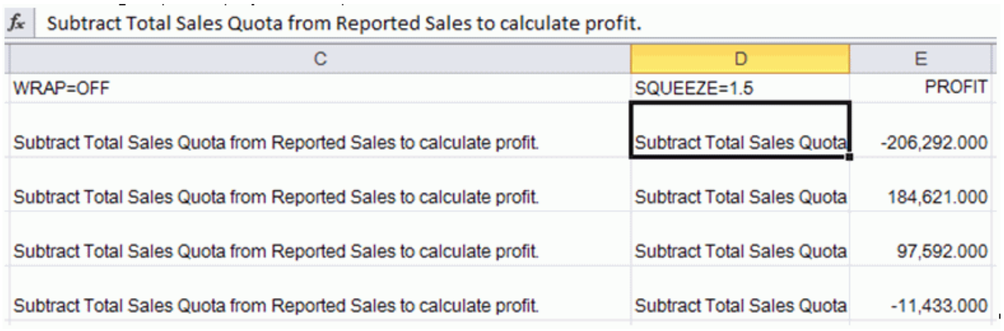
Since the output spans two pages, the output is shown below in two separate images.

The following output displays the different behavior for the "DEFAULT" and "WRAP=2" columns.



A	B
DEFAULT	WRAP=2
Subtract Total Sales Quota from Reported Sales to calculate profit.	Subtract Total Sales Quota from Reported Sales to calculate profit.
Subtract Total Sales Quota from Reported Sales to calculate profit.	Subtract Total Sales Quota from Reported Sales to calculate profit.
Subtract Total Sales Quota from Reported Sales to calculate profit.	Subtract Total Sales Quota from Reported Sales to calculate profit.
Subtract Total Sales Quota from Reported Sales to calculate profit.	Subtract Total Sales Quota from Reported Sales to calculate profit.

The following output displays the output for the "WRAP=OFF" and "SQUEEZE=1.5" columns.



Subtract Total Sales Quota from Reported Sales to calculate profit.

C	D	E
WRAP=OFF	SQUEEZE=1.5	PROFIT
Subtract Total Sales Quota from Reported Sales to calculate profit.	Subtract Total Sales Quota	-206,292.000
Subtract Total Sales Quota from Reported Sales to calculate profit.	Subtract Total Sales Quota	184,621.000
Subtract Total Sales Quota from Reported Sales to calculate profit.	Subtract Total Sales Quota	97,592.000
Subtract Total Sales Quota from Reported Sales to calculate profit.	Subtract Total Sales Quota	-11,433.000

Support for Drill Downs With XLSX Report Output

Drill downs are supported within the data elements in a report in XLSX format in the same manner as they are supported in EXL2K format. Hyperlink connections can be defined in the StyleSheet declaration of any data column to provide access to any external web source or to execute a FOCEXEC. Drill downs to FOCEXECs can contain data-driven parameters and can generate any of the supported output formats, including XLSX, PDF, HTML, DHTML, and PPT.

Drill downs within text embedded in headings, subheadings, subfootings, and footings will be implemented for XLSX format in a future release.

Note:

- ❏ When the limit of 65530 hyperlinks for a worksheet is reached, a warning message displays and no further links can be inserted. For more information on drill downs, see *Linking a Report to Other Resources* on page 397.
- ❏ The JAVASCRIPT and IMAGE drill-down options are not supported with FORMAT XLSX.

Redirection and Excel Drill-Down Reports

The Web Query Client can use redirection when passing the report output to the client application. When redirection is enabled, the Web Query Client saves report output in a temporary directory when a request is executed. Then, an HTTP call is made from the browser to retrieve the temporary stored output for display. When redirection is disabled, the report output is sent directly to the browser without any buffering.

Redirection is disabled by default for the .xlsx file extension because this enables drill downs to run successfully whether the user machine is configured to launch Excel in the browser or as an application outside of the browser.

For information about launching Excel in the browser or as an application, see *Viewing Excel Workbooks in the Browser or Excel Application* on page 324.

- ❏ **For workbooks opened outside the browser in the Excel application:** The current security context and any previously established session-related cookies are not retained, changing the user authorization, so drill-down reports will not have the information required to access the redirected files. The initial workbook will open within Excel, but the target drill-down workbook will not open and you will receive a message stating *You are not allowed to access this viewer file*. The drill-down feature in Microsoft Office products functioned in Web Query Release 1.1.x because anonymous drill-down access was permitted.

For more information on how Microsoft Office products work with session related information, see the Microsoft Office support site at *http://support.microsoft.com/kb/218153*.

Excel Page Settings

Excel page settings for the XLSX workbook default to the Web Query standards:

- Orientation: Portrait
- Page Size: Letter
- ,75 inches (Excel default)
- .75 inches (Excel default)

To customize these page settings, turn the XLSXPAGESETS attribute ON and define individual attributes.

If XLSXPAGESETS is turned on, but the page margin attributes are not defined within the procedure, the values will be set to the Web Query default of .25 inches.

Syntax: How to Define Excel Page Settings

```
[TYPE=REPORT,] XLSXPAGESETS={ON|OFF} [,PAGESIZE={pagesize|LETTER}]
   [,ORIENTATION={PORTRAIT|LANDSCAPE}] [,TOPMARGIN=n] [,BOTTOMMARGIN=m],$
```

where:

`XLSPAGESETS={ON|OFF}`

ON causes the page settings defined in the Web Query request to be applied to the Excel worksheet page settings. OFF retains the default page settings defined in the standard Excel workbook. OFF is the default value.

`n`

Defines the top margin for the worksheet in the units identified by the UNITS parameter (inches, by default). The default value is .25.

`m`

Defines the bottom margin for the worksheet in the units identified by the UNITS parameter (inches, by default). The default value is .25.

`pagesize`

Is one of the PAGESIZE values supported in a Web Query StyleSheet. LETTER is the default page size.

```
PORTRAIT|LANDSCAPE
```

PORTRAIT displays the report across the narrower dimension of a vertical page, producing a page that is longer than it is wide. PORTRAIT is the default value.

LANDSCAPE displays the report across the wider dimension of a horizontal page, producing a page that is wider than it is long.

Adding an Image to a Report

Web Query supports the placement of images within each area or node of the report on the worksheet. An image, such as a logo, gives corporate identity to a report, or provides visual appeal. Data specific images can be placed in headers, footers, and data columns to provide additional clarity and style.

The image must reside on the Web Query Reporting Server in a directory named on EDAPATH or APPPATH. If the file is not on the search path, supply the full path name.

Inserting Images Into Excel XLSX Reports

Images can be placed in any available Web Query reporting node or element of a worksheet. Supported image formats include .gif and .jpg.

Usage Considerations

- All images will be placed in the top-left corner of the first cell of the defined area, based on the top and left gap. Defined explicit positioning and justification have not been implemented yet.
- Standard page setting keywords can be used in conjunction with XLSXPAGESETS to control the page layout in standard reports (not compound).
- Images placed within a report cell in a row or column is anchored to the top-left corner of the cell. The cell is automatically sized to the height and width to fit the largest image (SQUEEZE=ON).
- Additional lines may need to be added within a heading, footing, subhead, or subfoot to accommodate the placement of the image.

Syntax: How to Insert Images Into Web Query Report Elements in XLSX Reports

```
TYPE={REPORT|heading|data}, IMAGE={url|file|(column)} [,BY=byfield]
   [,SIZE=(w h)] ,$
```

where:

`REPORT`

Embeds an image in the body of a report. The image appears in the background of the report. REPORT is the default value.

`heading`

Embeds an image in a heading or footing. Valid values are TABHEADING, TABFOOTING, FOOTING, HEADING, SUBHEAD, and SUBFOOT. Provide sufficient blank space in the heading or footing so that the image does not overlap the heading or footing text. You may also want to place heading or footing text to the right of the image using spot markers.

`data`

Defines a cell within a data column to place the image. Must be used with COLUMNS= attributes to identify the specific report column where the image should be anchored.

`url`

Is the URL of the image file.

`file`

Is the name of the image file. It must reside on the Web Query Reporting Server in a directory named on EDAPATH or APPPATH. If the file is not on the search path, supply the full path name. When specifying a GIF file, you can omit the file extension.

`column`

Is an alphanumeric field in the data source that contains the name of an image file. Enclose the column in parentheses ( ). The field containing the file name or image must be a display field or BY field referenced in the request. Note that the value of the field is interpreted exactly as if it were typed as the URL of the image in the StyleSheet. If you omit the suffix, .GIF is supplied, by default. You can use the SET BASEURL command for supplying the base URL of the images. This way, the value of the field does not have to include the complete URL. This syntax is useful, for example, if you want to embed an image in a SUBHEAD, and you want a different image for each value of the BY field on which the SUBHEAD occurs.

`byfield`

Is the sort field that generates the subhead or subfoot.

`SIZE`

Is the size of the image. By default, an image is added at its original size.

w

Is the width of the image, expressed in the unit of measurement specified by the UNITS parameter. Enclose the *w* and *h* values in parentheses (). Do not include a comma (,) between them.

h

Is the height of the image, expressed in the unit of measurement specified by the UNITS parameter.

Example: Adding a GIF Image to a Single Table Request

The following example illustrates how to add a GIF image to a request.

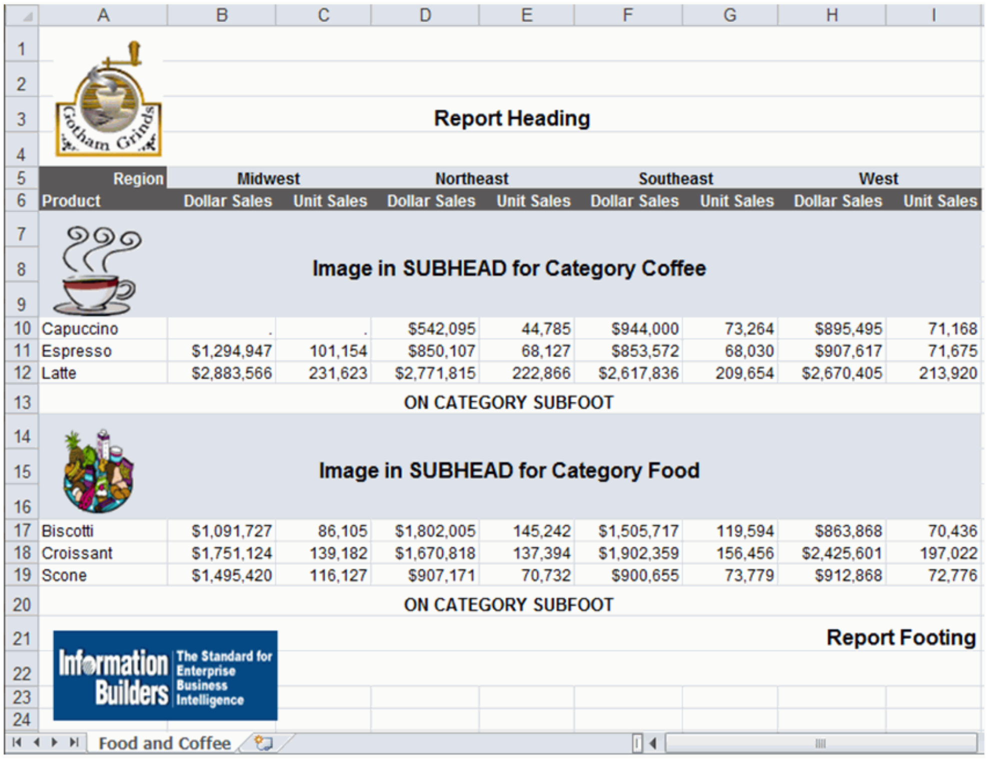
Report Request

```
DEFINE FILE GGSALES
SHOWCAT/A100=CATEGORY || '.GIF';
END
TABLE FILE GGSALES
SUM DOLLARS/D12CM UNITS/D12C
BY LOWEST CATEGORY NOPRINT
BY SHOWCAT NOPRINT
BY PRODUCT
ACROSS REGION
WHERE CATEGORY NE 'Gifts'

ON CATEGORY SUBHEAD
" "
“Image in SUBHEAD for Category <CATEGORY "
" "
ON TABLE SUBHEAD
" "
" "
" Report Heading "
" "
ON CATEGORY SUBFOOT
"ON CATEGORY SUBFOOT"

ON TABLE SUBFOOT
"Report Footing"
" "
ON TABLE SET PAGE-NUM NOLEAD
ON TABLE NOTOTAL
ON TABLE SET ACROSSTITLE SIDE
ON TABLE PCHOLD FORMAT XLSX
ON TABLE SET STYLE *
INCLUDE=IBFS:/stylesheet_location/custom_stylesheet.sty,$
END
```

***custom_stylesheet*.sty**

```
TYPE=REPORT, GRID=OFF, FONT='ARIAL', SIZE=9, TITLETEXT='Food and Coffee', $
TYPE=REPORT, COLUMN=PRODUCT, SQUEEZE=1, $
TYPE=TITLE, BACKCOLOR=RGB(90 90 90), COLOR=RGB(255 255 255), STYLE=BOLD, $
TYPE=ACROSSTITLE, STYLE=BOLD, BACKCOLOR=RGB(90 90 90),
   COLOR=RGB(255 255 255), $
TYPE=ACROSSVALUE, BACKCOLOR=RGB(218 225 232), STYLE=BOLD, JUSTIFY=CENTER, $
TYPE=HEADING, STYLE=BOLD, COLOR=RGB(0 35 95), SIZE=12, JUSTIFY=CENTER, $
TYPE=FOOTING, BACKCOLOR=RGB(90 90 90), SIZE=12, COLOR=RGB(255 255 255),
   STYLE=BOLD, JUSTIFY=CENTER, $
TYPE=SUBHEAD, SIZE=12, STYLE=BOLD, BACKCOLOR=RGB(218 225 232),
   JUSTIFY=CENTER, $
TYPE=SUBHEAD, IMAGE=(SHOWCAT), SIZE=(.6 .6), $
TYPE=SUBFOOT, SIZE=10, STYLE=BOLD, JUSTIFY=CENTER, $
TYPE=TABHEADING, SIZE=12, STYLE=BOLD, JUSTIFY=CENTER, $
TYPE=TABHEADING, IMAGE=gglogo.gif, $
TYPE=TABFOOTING, SIZE=12, STYLE=BOLD, JUSTIFY=RIGHT, $
TYPE=TABFOOTING, IMAGE=logo.gif, SIZE=(1.67 .6), $
```

In the following request, since the referenced images are not part of the existing GGSALES table, the image files (.gif) are being built in the DEFINE and then referenced in the TABLE request. You can NOPRINT fields if you do not want them to display as columns, but the fields must be referenced in the table to include them in the internal matrix. This will allow the images to be placed in the headings, footings, or data cells. The specific location is defined using StyleSheet definitions for attaching the image based on field value.



Report Heading

Region	Midwest		Northeast		Southeast		West	
Product	Dollar Sales	Unit Sales	Dollar Sales	Unit Sales	Dollar Sales	Unit Sales	Dollar Sales	Unit Sales
Image in SUBHEAD for Category Coffee								
Capuccino	.	.	$542,095	44,785	$944,000	73,264	$895,495	71,168
Espresso	$1,294,947	101,154	$850,107	68,127	$853,572	68,030	$907,617	71,675
Latte	$2,883,566	231,623	$2,771,815	222,866	$2,617,836	209,654	$2,670,405	213,920
ON CATEGORY SUBFOOT								
Image in SUBHEAD for Category Food								
Biscotti	$1,091,727	86,105	$1,802,005	145,242	$1,505,717	119,594	$863,868	70,436
Croissant	$1,751,124	139,182	$1,670,818	137,394	$1,902,359	156,456	$2,425,601	197,022
Scone	$1,495,420	116,127	$907,171	70,732	$900,655	73,779	$912,868	72,776
ON CATEGORY SUBFOOT								

Report Footing

Inserting Images Into XLSX Workbook Headers and Footers

Web Query supports the insertion of images into Excel headers and footers and the definition of key page settings to support the placement of these images in relationship to the overall worksheet and the Excel generated page breaks. This new access to the Excel page functionality is designed to enhance overall usability of the worksheets for users who will be printing these reports. Page settings including orientation, page size, and page margins will directly affect the layout of each Excel page based on values defined within the FOCEXEC. Images can be included on headers and footers on every printed page, on the first page of the report only, or only on all subsequent pages. The Web Query headings and footings continue to display within the worksheet. With this feature, Web Query can insert logos to be printed once at the top of a report and watermark images that need to be displayed on every printed page.

Syntax: How to Insert Images Into Excel Headers and Footers

```
TYPE={PAGEHEADER|PAGEFOOTER}, OBJECT=IMAGE,
   IMAGE=imagename, JUSTIFY={LEFT|CENTER|RIGHT}
   [,DISPLAYON={FIRST|NOT-FIRST}] [,SIZE=(w h)],$
```

where:

`PAGEHEADER`

Places the image in the worksheet header.

`PAGEFOOTER`

Places the image in the worksheet footer.

`imagename`

Is the name of a valid image file to be placed in the header or footer. The image must be located in the defined application path on the Reporting Server. The image types supported are GIF and JPEG.

`JUSTIFY={LEFT|CENTER|RIGHT}`

Identifies the area in the header or footer to contain the image and the justification or placement within that defined area.

`DISPLAYON`

Defines whether the image should be placed on the first page only or on all pages except the first. Omit this attribute to place the image on all pages.

Valid values are:

`FIRST` places the image only on the first page.

`NOT-FIRST` places the image on every page, except the first page.

`SIZE=(`*`w h`*`)`

Is the size of the image. By default, an image is added at its original size.

w is the width of the image, expressed in the unit of measurement specified by the UNITS parameter.

h is the height of the image, expressed in the unit of measurement specified by the UNITS parameter.

Example: Inserting Images in Excel Headers and Footers and Defining Page Settings

The following request against the GGSALES data source places the ibi_logo.gif image on the left header area of the first page and the right header area of every subsequent page of the resulting worksheet. It places the WebQuery1.gif image in the center area of the footer on every page.

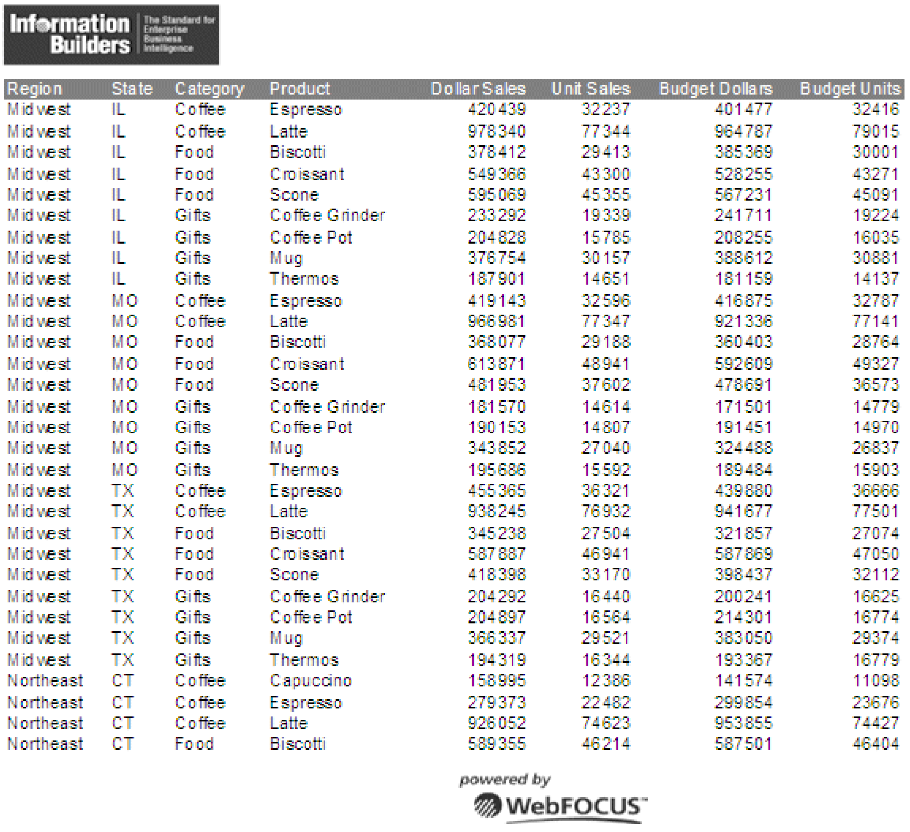
Report Request

```
TABLE FILE GGSALES
SUM DOLLARS UNITS BUDDOLLARS BUDUNITS
BY REGION
BY ST
BY CATEGORY
BY PRODUCT
ON TABLE SET BYDISPLAY ON
ON TABLE PCHOLD FORMAT XLSX
ON TABLE SET STYLE *
INCLUDE=IBFS:/stylesheet_location/custom_stylesheet.sty,$
END
```

***custom_stylesheet*.sty**

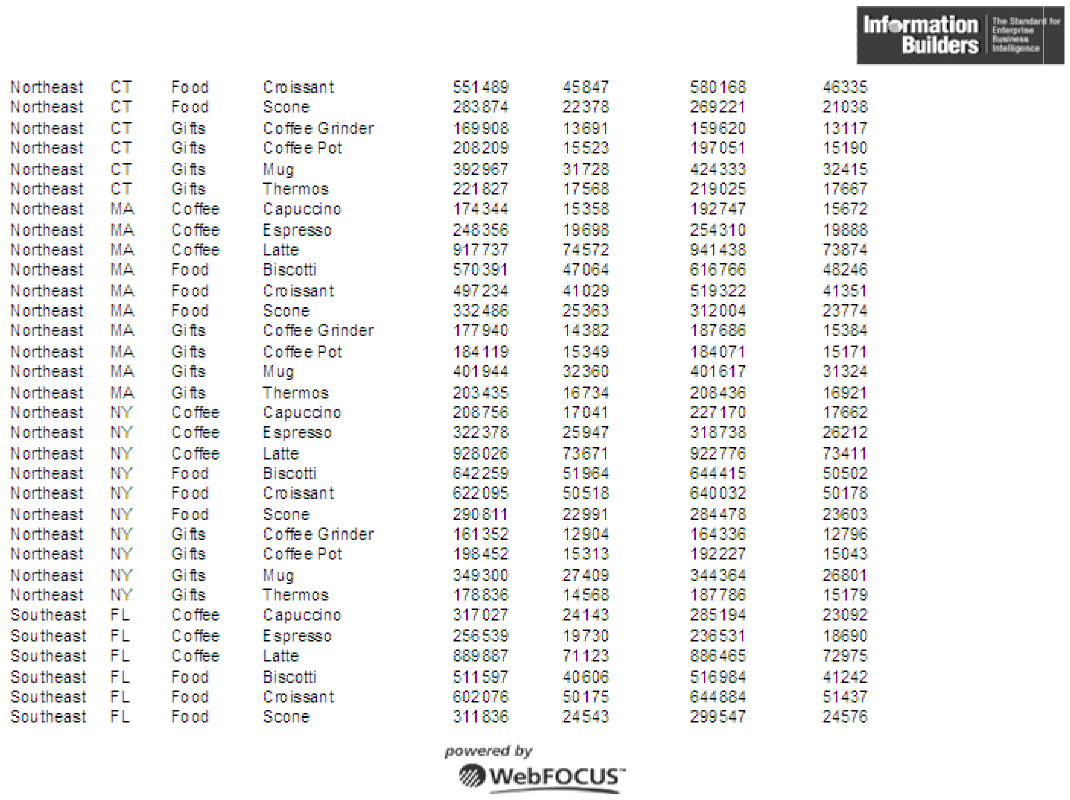
```
TYPE=REPORT, FONT=ARIAL, SIZE=12,
XLSXPAGESETS=ON,TOPMARGIN=1,BOTTOMMARGIN=1,ORIENTATION=LANDSCAPE,
   PAGESIZE=LETTER, $
TYPE=TITLE, COLOR=WHITE, BACKCOLOR=GREY, $
TYPE=PAGEHEADER, OBJECT=IMAGE, JUSTIFY=LEFT, IMAGE=IBI_LOGO.GIF,
   DISPLAYON=FIRST, $
TYPE=PAGEHEADER, OBJECT=IMAGE, JUSTIFY=RIGHT, IMAGE=IBI_LOGO.GIF,
   DISPLAYON=NOT-FIRST, $
TYPE=PAGEFOOTER, OBJECT=IMAGE, JUSTIFY=CENTER, IMAGE=WebQuery1.GIF, $
```

The first page of output has the ibilogo.gif image in the left area of the header and the WebQuery1.gif image in the center area of the footer.



Region	State	Category	Product	Dollar Sales	Unit Sales	Budget Dollars	Budget Units
Midwest	IL	Coffee	Espresso	420439	32237	401477	32416
Midwest	IL	Coffee	Latte	978340	77344	964787	79015
Midwest	IL	Food	Biscotti	378412	29413	385369	30001
Midwest	IL	Food	Croissant	549366	43300	528255	43271
Midwest	IL	Food	Scone	595069	45355	567231	45091
Midwest	IL	Gifts	Coffee Grinder	233292	19339	241711	19224
Midwest	IL	Gifts	Coffee Pot	204828	15785	208255	16035
Midwest	IL	Gifts	Mug	376754	30157	388612	30881
Midwest	IL	Gifts	Thermos	187901	14651	181159	14137
Midwest	MO	Coffee	Espresso	419143	32596	416875	32787
Midwest	MO	Coffee	Latte	966981	77347	921336	77141
Midwest	MO	Food	Biscotti	368077	29188	360403	28764
Midwest	MO	Food	Croissant	613871	48941	592609	49327
Midwest	MO	Food	Scone	481953	37602	478691	36573
Midwest	MO	Gifts	Coffee Grinder	181570	14614	171501	14779
Midwest	MO	Gifts	Coffee Pot	190153	14807	191451	14970
Midwest	MO	Gifts	Mug	343852	27040	324488	26837
Midwest	MO	Gifts	Thermos	195686	15592	189484	15903
Midwest	TX	Coffee	Espresso	455365	36321	439880	36666
Midwest	TX	Coffee	Latte	938245	76932	941677	77501
Midwest	TX	Food	Biscotti	345238	27504	321857	27074
Midwest	TX	Food	Croissant	587887	46941	587869	47050
Midwest	TX	Food	Scone	418398	33170	398437	32112
Midwest	TX	Gifts	Coffee Grinder	204292	16440	200241	16625
Midwest	TX	Gifts	Coffee Pot	204897	16564	214301	16774
Midwest	TX	Gifts	Mug	366337	29521	383050	29374
Midwest	TX	Gifts	Thermos	194319	16344	193367	16779
Northeast	CT	Coffee	Capuccino	158995	12386	141574	11098
Northeast	CT	Coffee	Espresso	279373	22482	299854	23676
Northeast	CT	Coffee	Latte	926052	74623	953855	74427
Northeast	CT	Food	Biscotti	589355	46214	587501	46404

The second page of output has the ibilogo.gif image in the right area of the header and the WebQuery1.gif image in the center area of the footer.



Northeast	CT	Food	Croissant	551489	45847	580168	46335
Northeast	CT	Food	Scone	283874	22378	269221	21038
Northeast	CT	Gifts	Coffee Grinder	169908	13691	159620	13117
Northeast	CT	Gifts	Coffee Pot	208209	15523	197051	15190
Northeast	CT	Gifts	Mug	392967	31728	424333	32415
Northeast	CT	Gifts	Thermos	221827	17568	219025	17667
Northeast	MA	Coffee	Capuccino	174344	15358	192747	15672
Northeast	MA	Coffee	Espresso	248356	19698	254310	19888
Northeast	MA	Coffee	Latte	917737	74572	941438	73874
Northeast	MA	Food	Biscotti	570391	47064	616766	48246
Northeast	MA	Food	Croissant	497234	41029	519322	41351
Northeast	MA	Food	Scone	332486	25363	312004	23774
Northeast	MA	Gifts	Coffee Grinder	177940	14382	187686	15384
Northeast	MA	Gifts	Coffee Pot	184119	15349	184071	15171
Northeast	MA	Gifts	Mug	401944	32360	401617	31324
Northeast	MA	Gifts	Thermos	203435	16734	208436	16921
Northeast	NY	Coffee	Capuccino	208756	17041	227170	17662
Northeast	NY	Coffee	Espresso	322378	25947	318738	26212
Northeast	NY	Coffee	Latte	928026	73671	922776	73411
Northeast	NY	Food	Biscotti	642259	51964	644415	50502
Northeast	NY	Food	Croissant	622095	50518	640032	50178
Northeast	NY	Food	Scone	290811	22991	284478	23603
Northeast	NY	Gifts	Coffee Grinder	161352	12904	164336	12796
Northeast	NY	Gifts	Coffee Pot	198452	15313	192227	15043
Northeast	NY	Gifts	Mug	349300	27409	344364	26801
Northeast	NY	Gifts	Thermos	178836	14568	187786	15179
Southeast	FL	Coffee	Capuccino	317027	24143	285194	23092
Southeast	FL	Coffee	Espresso	256539	19730	236531	18690
Southeast	FL	Coffee	Latte	889887	71123	886465	72975
Southeast	FL	Food	Biscotti	511597	40606	516984	41242
Southeast	FL	Food	Croissant	602076	50175	644884	51437
Southeast	FL	Food	Scone	311836	24543	299547	24576

Reference: Usage Notes for Inserting Images Into XLSX Worksheet Headers and Footers

- The Excel headers and footers are not automatically sized based on contents of the areas. Define page margins within the page settings (XLSPAGESETS) to account for the space required to display the images within each page of the report.
- The image sizing based on the specified height and width is not proportional. Sizing may cause image distortion.
- BLOB image fields are not supported in this release.

Reference: Displaying Watermarks on XLSX Report Output

Watermark images can be placed into the Excel headers to display on every printed page of the generated worksheet.

Excel places images on the page starting in the header from left to right and then the footer from left to right. Large images placed in the header may overlap images before them in the presentation order. For page layouts with a logo in the left area and watermark centered on the page, the watermark image background must be transparent so it does not overlay the logo image.

In Excel, images are placed first on the page. All other contents of the worksheet are then placed on top of the images. Text in cells and styling, such as background color and drawing objects, are placed on top of the images. Excel supports transparency in drawing objects and images, but not in cell background color. BACKCOLOR will cover over images placed on the page.

Example: Placing a Watermark in an XLSX Header

The following request against the GGSALES data source uses the internaluseonly.gif image as a watermark to display in the background of every page of the worksheet. Although the image is placed in the center area of the header, it is large enough to span the entire worksheet page. It has a transparent background, so it does not cover the logo images placed at the left in the header and the center in the footer.

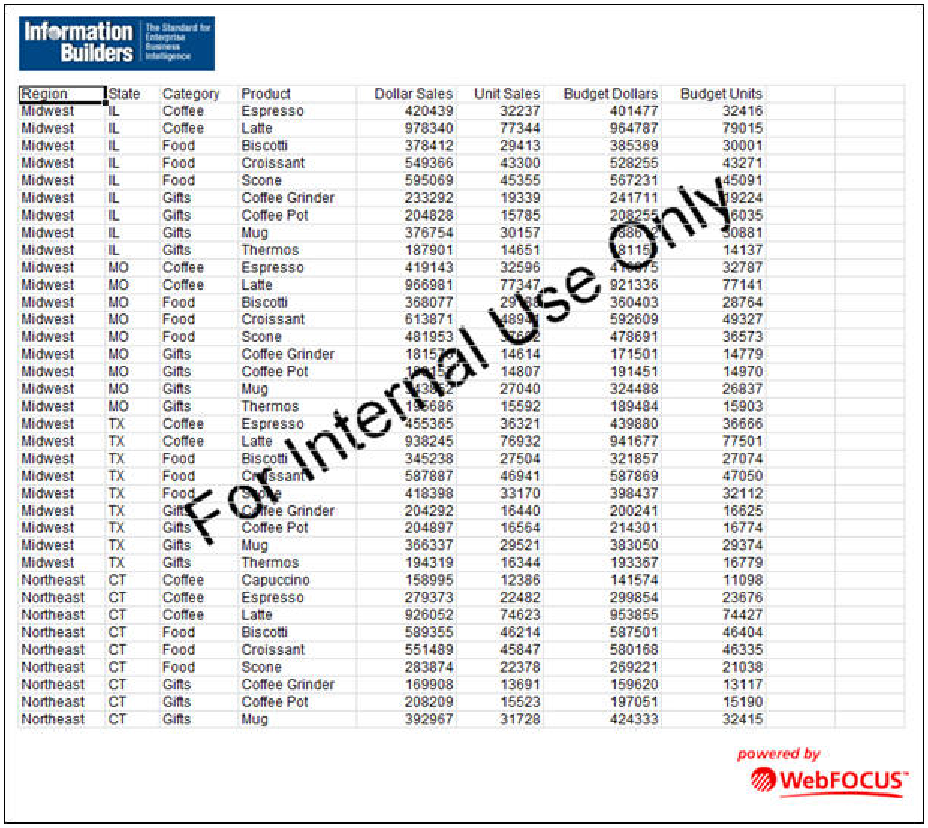
Report Request

```
TABLE FILE GGSALES
SUM DOLLARS UNITS BUDDOLLARS BUDUNITS
BY REGION
BY ST
BY CATEGORY
BY PRODUCT
ON TABLE SET BYDISPLAY ON
ON TABLE PCHOLD FORMAT XLSX
ON TABLE SET STYLE *
INCLUDE=IBFS:/stylesheet_location/custom_stylesheet.sty,$
END
```

***custom_stylesheet*.sty**

```
TYPE=REPORT, XLSXPAGESETS=ON,
TOPMARGIN=1,BOTTOMMARGIN=1,LEFTMARGIN=1, RIGHTMARGIN=1,
ORIENTATION=LANDSCAPE,PAGESIZE=LETTER, $
TYPE=PAGEHEADER, OBJECT=IMAGE, JUSTIFY=LEFT, IMAGE=IBI_LOGO.GIF,
   DISPLAYON=FIRST, $
TYPE=PAGEHEADER, OBJECT=IMAGE, JUSTIFY=CENTER,
   IMAGE=WQINTERNALUSEONLY.GIF, $
TYPE=PAGEFOOTER, OBJECT=IMAGE, JUSTIFY=RIGHT, IMAGE=WebQuery1.GIF, $
```

The first page of the generated worksheet shows the watermark image beneath the data. This image is displayed on every page of the worksheet.



Region	State	Category	Product	Dollar Sales	Unit Sales	Budget Dollars	Budget Units
Midwest	IL	Coffee	Espresso	420439	32237	401477	32416
Midwest	IL	Coffee	Latte	978340	77344	964787	79015
Midwest	IL	Food	Biscotti	378412	29413	385369	30001
Midwest	IL	Food	Croissant	549366	43300	528255	43271
Midwest	IL	Food	Scone	595069	45355	567231	45091
Midwest	IL	Gifts	Coffee Grinder	233292	19339	241711	19224
Midwest	IL	Gifts	Coffee Pot	204828	15785	208255	16035
Midwest	IL	Gifts	Mug	376754	30157	[illegible]	30881
Midwest	IL	Gifts	Thermos	187901	14651	[illegible]	14137
Midwest	MO	Coffee	Espresso	419143	32596	[illegible]	32787
Midwest	MO	Coffee	Latte	966981	77347	921336	77141
Midwest	MO	Food	Biscotti	368077	[illegible]	360403	28764
Midwest	MO	Food	Croissant	613871	[illegible]	592609	49327
Midwest	MO	Food	Scone	481953	[illegible]	478691	36573
Midwest	MO	Gifts	Coffee Grinder	[illegible]	14614	171501	14779
Midwest	MO	Gifts	Coffee Pot	[illegible]	14807	191451	14970
Midwest	MO	Gifts	Mug	[illegible]	27040	324488	26837
Midwest	MO	Gifts	Thermos	[illegible]	15592	189484	15903
Midwest	TX	Coffee	Espresso	455365	36321	439880	36666
Midwest	TX	Coffee	Latte	938245	76932	941677	77501
Midwest	TX	Food	Biscotti	345238	27504	321857	27074
Midwest	TX	Food	Croissant	587887	46941	587869	47050
Midwest	TX	Food	Scone	418398	33170	398437	32112
Midwest	TX	Gifts	Coffee Grinder	204292	16440	200241	16625
Midwest	TX	Gifts	Coffee Pot	204897	16564	214301	16774
Midwest	TX	Gifts	Mug	366337	29521	383050	29374
Midwest	TX	Gifts	Thermos	194319	16344	193367	16779
Northeast	CT	Coffee	Capuccino	158995	12386	141574	11098
Northeast	CT	Coffee	Espresso	279373	22482	299854	23676
Northeast	CT	Coffee	Latte	926052	74623	953855	74427
Northeast	CT	Food	Biscotti	589355	46214	587501	46404
Northeast	CT	Food	Croissant	551489	45847	580168	46335
Northeast	CT	Food	Scone	283874	22378	269221	21038
Northeast	CT	Gifts	Coffee Grinder	169908	13691	159620	13117
Northeast	CT	Gifts	Coffee Pot	208209	15523	197051	15190
Northeast	CT	Gifts	Mug	392967	31728	424333	32415

Overcoming the Excel 2007/2010 Row Limit Using Overflow Worksheets

The maximum number of rows supported by Excel 2007/2010 on a worksheet is 1,048,576 (1MB). When you create an XLSX output file from a Web Query report, the number of rows generated can be greater than this maximum.

To avoid creating an incomplete output file, you can have extra rows flow onto a new worksheet, called an *overflow worksheet*. The name of each overflow worksheet will be the name of the original worksheet appended with an increment number.

In addition, when the overflow worksheet feature is enabled, you can set a target value for the maximum number of rows to be included on a worksheet. By default, the row limit will be set to the default value for the LINES parameter (57).

Note: By default, when generating XLSX output, the Web Query page heading and page footing commands generate only worksheet headings and worksheet footings.

Syntax: How to Enable Overflow Worksheets

Add the ROWOVERFLOW attribute to your Web Query StyleSheet:

```
TYPE=REPORT, ROWOVERFLOW={ON|OFF|PBON}, [ROWLIMIT={n|MAX},]$
```

where:

`ON`

Enables overflow worksheets.

`OFF`

Disables overflow worksheets. OFF is the default value.

`PBON`

Inserts Web Query page breaks that display the page heading, footing, and column titles at the appropriate places within the worksheet rows. This option does not cause a new worksheet to start when a Web Query page break occurs.

`ROWLIMIT=`*n*

Sets a target value for the number of rows to be included on a worksheet to *n* rows. The default value is the LINES value (by default, 57).

`ROWLIMIT=MAX`

Sets a target value for the number of rows to be included on a worksheet to 1,048,000 rows for XLSX output.

Reference: Usage Notes for XLSX Overflow Worksheets

- The report heading is placed once at the start of the first sheet. The report footing is placed once at the bottom of the last overflow sheet.
- Unless the PBON setting is used, worksheet headings and column titles are repeated at the top of the original sheet and each subsequent overflow sheet. Worksheet footings are placed at the bottom of the original sheet and each subsequent overflow sheet. The data values are displayed on the top data row of each overflow sheet as they would be on a standard new page.
- Report total lines are displayed at the bottom of the last overflow sheet directly above the final page and table footings.

- Subheadings, subfootings, and subtotal lines display within the data flow as normal. No special consideration is made to retain groupings within a given sheet.
- If ROWOVERFLOW=PBON, the page headings and footings and column titles display within the worksheet when a Web Query command causes a page break.
- For XLSX output, if the ROWOVERFLOW attribute is specified in the StyleSheet and ROWLIMIT is greater than 1MB, the following message is presented and no output file is generated:

  ```
  (FOC3338) The row limit for EXCEL XLSX worksheets is 1048576.
  ```

- As formula references are not automatically updated to reflect placement on new overflow worksheets, the EXL2K FORMULA output type, which contains formula references, is not supported.
- The overflow worksheet feature applies to rows only, not columns. A new worksheet will not automatically be created if a report generates more than the Excel 2007/2010 limit or 16,384 columns.
- As named ranges in Excel cannot run across multiple worksheets, the IN-RANGES phrase that defines named ranges in the resulting workbook is not supported with the ROWOVERFLOW feature. When they exist together in the same request, ROWOVERFLOW takes precedence and the IN-RANGES phrase is ignored.

Example: Creating Overflow Worksheets

The following request illustrates how to create XLSX report output with overflow worksheets. The ROWOVERFLOW=ON attribute in the StyleSheet activates the overflow feature. Without this attribute, one worksheet would have been generated instead of three.

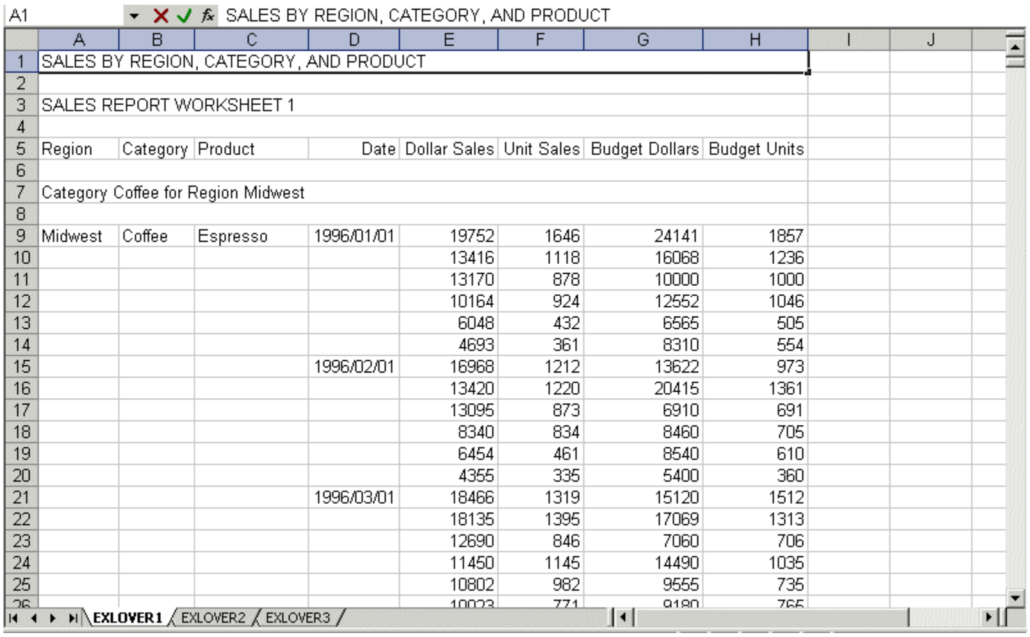
Report Request

```
TABLE FILE GGSALES
-* ****Report Heading****
ON TABLE SUBHEAD
"SALES BY REGION, CATEGORY, AND PRODUCT"
" "
-* ****Worksheet Heading****
HEADING
"SALES REPORT WORKSHEET <TABPAGENO"
" "
-* ****Worksheet Footing****
FOOTING
" "
"END OF WORKSHEET <TABPAGENO"
PRINT DOLLARS UNITS BUDDOLLARS BUDUNITS
BY REGION
BY CATEGORY
BY PRODUCT
BY DATE

-* ****Subfoot****
ON REGION SUBFOOT
" "
" End of Region <REGION"
" "
-* ****Subhead****
ON REGION SUBHESD
" "
"Category <CATEGORY for Region <REGION"
" "
-* ****Report Footing****
ON TABLE SUBFOOT
" "
"END OF REPORT"
ON TABLE PCHOLD FORMAT XLSX
ON TABLE SET STYLE *
INCLUDE=IBFS:/stylesheet_location/custom_stylesheet.sty,$
END
```

***custom_stylesheet*.sty**

```
TYPE=REPORT, TITLETEXT=EXLOVER, ROWOVERFLOW=ON, ROWLIMIT=2000, $
```

The report heading displays on the first worksheet only, the page heading and column titles display on each worksheet, and the subhead and subfoot display whenever the associated sort field changes value. The following image shows the top of the first worksheet, displaying the report heading, page heading, column titles, and first subhead.

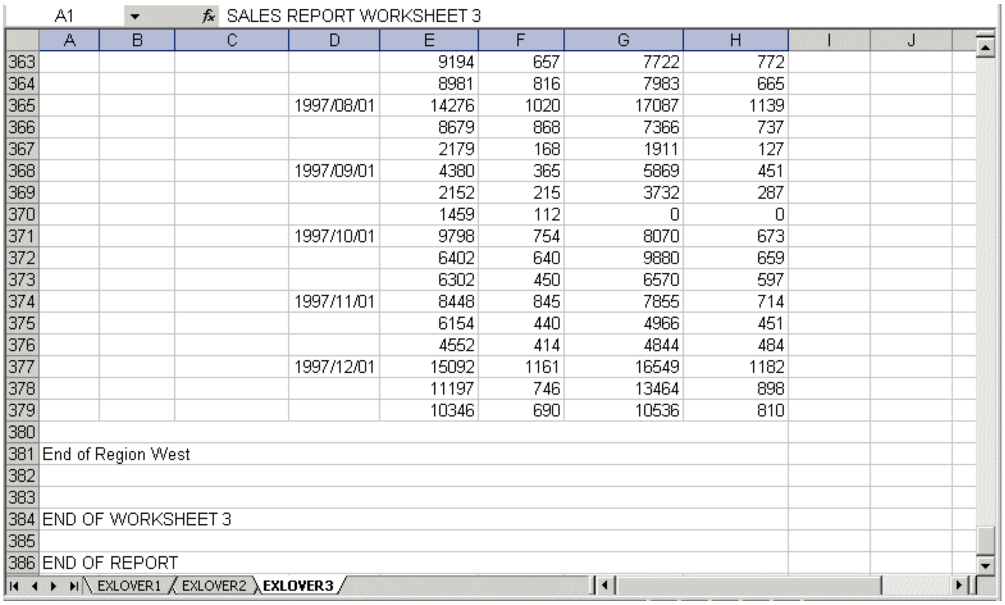


A1 SALES BY REGION, CATEGORY, AND PRODUCT

	A	B	C	D	E	F	G	H	I	J
1	SALES BY REGION, CATEGORY, AND PRODUCT									
2										
3	SALES REPORT WORKSHEET 1									
4										
5	Region	Category	Product	Date	Dollar Sales	Unit Sales	Budget Dollars	Budget Units		
6										
7	Category Coffee for Region Midwest									
8										
9	Midwest	Coffee	Espresso	1996/01/01	19752	1646	24141	1857		
10					13416	1118	16068	1236		
11					13170	878	10000	1000		
12					10164	924	12552	1046		
13					6048	432	6565	505		
14					4693	361	8310	554		
15				1996/02/01	16968	1212	13622	973		
16					13420	1220	20415	1361		
17					13095	873	6910	691		
18					8340	834	8460	705		
19					6454	461	8540	610		
20					4355	335	5400	360		
21				1996/03/01	18466	1319	15120	1512		
22					18135	1395	17069	1313		
23					12690	846	7060	706		
24					11450	1145	14490	1035		
25					10802	982	9555	735		
26					10023	771	9180	765		

Note that the TITLETEXT attribute in the StyleSheet specified the name EXLOVER, so the three worksheets were generated with the names EXLOVER1, EXLOVER2, and EXLOVER3. If there had been no TITLETEXT attribute, the sheets would have been named SHEET1, SHEET2, and SHEET3.

The worksheet footing displays at the bottom of each worksheet and the report footing displays at the bottom of the last worksheet. The following image shows the bottom of the last worksheet, displaying the last subfoot, the page footing, and the report footing.



A1 SALES REPORT WORKSHEET 3

	A	B	C	D	E	F	G	H	I	J
363					9194	657	7722	772		
364					8981	816	7983	665		
365				1997/08/01	14276	1020	17087	1139		
366					8679	868	7366	737		
367					2179	168	1911	127		
368				1997/09/01	4380	365	5869	451		
369					2152	215	3732	287		
370					1459	112	0	0		
371				1997/10/01	9798	754	8070	673		
372					6402	640	9880	659		
373					6302	450	6570	597		
374				1997/11/01	8448	845	7855	714		
375					6154	440	4966	451		
376					4552	414	4844	484		
377				1997/12/01	15092	1161	16549	1182		
378					11197	746	13464	898		
379					10346	690	10536	810		
380										
381	End of Region West									
382										
383										
384	END OF WORKSHEET 3									
385										
386	END OF REPORT									

Example: Creating Overflow Worksheets With Web Query Page Breaks

The following request illustrates how to create XLSX report output with overflow worksheets. The ROWOVERFLOW=PBON attribute in the StyleSheet activates the overflow feature, and the ROWLIMIT=250 sets the maximum number of rows in each worksheet to approximately 250. Without this attribute, one worksheet would have been generated. The PRODUCT sort phrase specifies a page break.

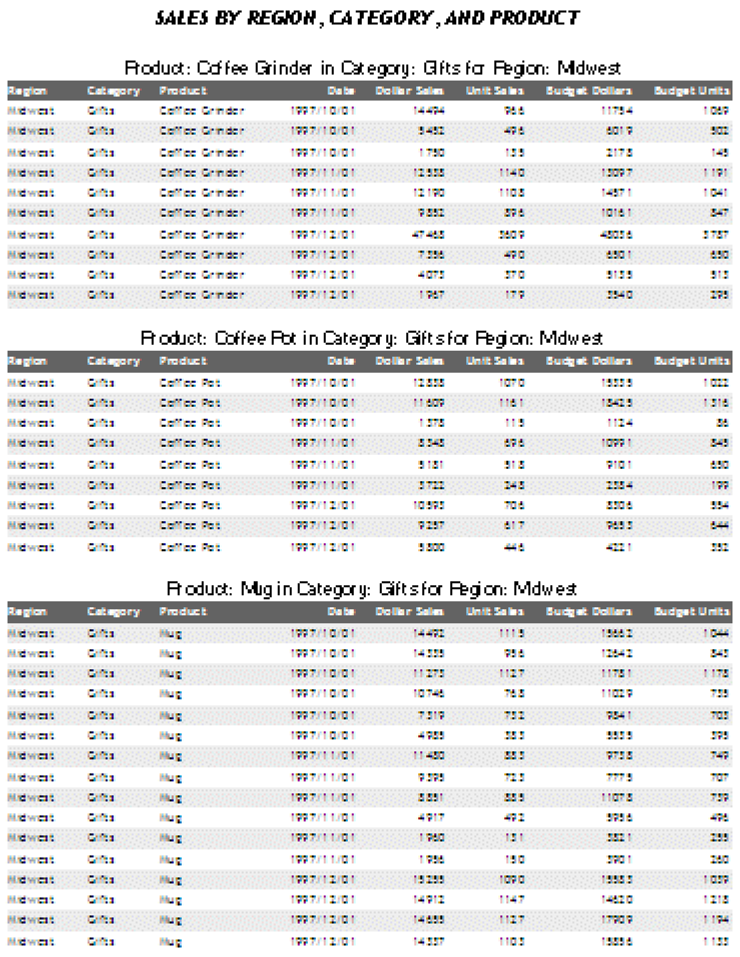
Report Request

```
TABLE FILE GGSALES
-* ****Report Heading****
ON TABLE SUBHEAD
"SALES BY REGION, CATEGORY, AND PRODUCT"
" "
PRINT DOLLARS UNITS BUDDOLLARS BUDUNITS
BY REGION
BY HIGHEST CATEGORY
BY PRODUCT PAGE-BREAK
BY DATE
WHERE DATE GE '19971001'
-* ****Page Heading****
HEADING
" Product: <PRODUCT in Category: <CATEGORY for Region: <REGION"
-* ****Page Footing****
FOOTING
" "
-* ****Report Footing****
ON TABLE SUBFOOT
" "
"END OF REPORT"
ON TABLE SET BYDISPLAY ON
ON TABLE PCHOLD FORMAT XLSX
ON TABLE SET STYLE *
INCLUDE=IBFS:/stylesheet_location/custom_stylesheet.sty,$
END
```

***custom_stylesheet*.sty**

```
TITLETEXT=EXLOVER, ROWOVERFLOW=PBON, ROWLIMIT=250, $
```

The report heading displays on the first worksheet only, the page heading, footing, and column titles display on each worksheet and at each Web Query page break (each time the product changes), and the subhead and subfoot display whenever the associated sort field changes value. The following image shows the top of the first worksheet.



SALES BY REGION, CATEGORY, AND PRODUCT

Product: Coffee Grinder in Category: Gifts for Region: Midwest

Region	Category	Product	Date	Dollar Sales	Unit Sales	Budget Dollars	Budget Units
Midwest	Gifts	Coffee Grinder	1997/10/01	14494	986	11734	1069
Midwest	Gifts	Coffee Grinder	1997/10/01	5432	496	6019	502
Midwest	Gifts	Coffee Grinder	1997/10/01	1750	135	2178	149
Midwest	Gifts	Coffee Grinder	1997/11/01	12533	1140	13097	1191
Midwest	Gifts	Coffee Grinder	1997/11/01	12190	1103	14571	1041
Midwest	Gifts	Coffee Grinder	1997/11/01	9832	896	10181	847
Midwest	Gifts	Coffee Grinder	1997/12/01	47468	3609	48036	3787
Midwest	Gifts	Coffee Grinder	1997/12/01	7336	490	6501	650
Midwest	Gifts	Coffee Grinder	1997/12/01	4075	370	5135	513
Midwest	Gifts	Coffee Grinder	1997/12/01	1967	179	3340	298

Product: Coffee Pot in Category: Gifts for Region: Midwest

Region	Category	Product	Date	Dollar Sales	Unit Sales	Budget Dollars	Budget Units
Midwest	Gifts	Coffee Pot	1997/10/01	12838	1070	15539	1022
Midwest	Gifts	Coffee Pot	1997/10/01	11609	1161	15429	1316
Midwest	Gifts	Coffee Pot	1997/10/01	1378	115	1124	86
Midwest	Gifts	Coffee Pot	1997/11/01	8348	696	10991	849
Midwest	Gifts	Coffee Pot	1997/11/01	5131	513	9101	690
Midwest	Gifts	Coffee Pot	1997/11/01	3722	248	2384	199
Midwest	Gifts	Coffee Pot	1997/12/01	10593	706	8306	554
Midwest	Gifts	Coffee Pot	1997/12/01	9257	617	9653	644
Midwest	Gifts	Coffee Pot	1997/12/01	5800	446	4221	352

Product: Mug in Category: Gifts for Region: Midwest

Region	Category	Product	Date	Dollar Sales	Unit Sales	Budget Dollars	Budget Units
Midwest	Gifts	Mug	1997/10/01	14492	1113	15662	1044
Midwest	Gifts	Mug	1997/10/01	14339	956	12642	843
Midwest	Gifts	Mug	1997/10/01	11273	1127	11781	1178
Midwest	Gifts	Mug	1997/10/01	10746	768	11029	735
Midwest	Gifts	Mug	1997/10/01	7319	732	9541	703
Midwest	Gifts	Mug	1997/10/01	4958	553	5535	398
Midwest	Gifts	Mug	1997/11/01	11480	853	9738	749
Midwest	Gifts	Mug	1997/11/01	9393	723	7773	707
Midwest	Gifts	Mug	1997/11/01	8851	885	11078	739
Midwest	Gifts	Mug	1997/11/01	4917	492	5956	496
Midwest	Gifts	Mug	1997/11/01	1960	131	3321	233
Midwest	Gifts	Mug	1997/11/01	1956	150	3901	260
Midwest	Gifts	Mug	1997/12/01	15233	1090	15553	1039
Midwest	Gifts	Mug	1997/12/01	14912	1147	14620	1212
Midwest	Gifts	Mug	1997/12/01	14655	1127	17909	1194
Midwest	Gifts	Mug	1997/12/01	14337	1103	15556	1133

Excel Compound Reports Using XLSX

Excel compound reports (InfoAssist Documents) generate compound workbooks that can contain multiple worksheet reports using the XLSX output format.

You can use standard Compound Layout syntax to generate XLSX compound workbooks. By default, each of the component reports from the compound report is placed in a new Excel worksheet (analogous to a new page in PDF).

The components of an Excel compound report can include standard tables and ROWOVERFLOW reports.

Component graphs will be added to worksheets as images.

Reference: Usage Notes for Excel Compound Reports Using XLSX

- Images and graphs can be embedded within a component, but images and drawing objects (lines, boxes, strings) defined in Compound Layout syntax on a page layout will not be included in the generated workbook.
- Graphs and images are not supported in Excel headers and footers within XLSX compound workbooks.
- Coordinated compound reports that generate individual instances of the overall report for each unique primary key are not available in XLSX.

Note: Since multiple tables are generated, Web Query will ensure that each tab name is unique.

Example: Creating a Simple Compound Report Using XLSX

The following request illustrates how to create a compound report using XLSX.

Report Request Report 1

```
SET PAGE-NUM=OFF
TABLE FILE GGSALES
HEADING
"Report 1: Coffee - Budget"
" "
SUM BUDDOLLARS BUDUNITS COLUMN-TOTAL AS 'Total'
BY REGION
ON TABLE SET STYLE *
INCLUDE=IBFS:/stylesheet_location/custom_stylesheet.sty,$
ON TABLE PCHOLD AS EX1 FORMAT XLSX OPEN
END
```

***custom_stylesheet*.sty**

```
TYPE=REPORT, TITLETEXT=Coffee Budget, $
TYPE=HEADING, SIZE=14, $
```

Report Request Report 2

```
TABLE FILE GGSALES
HEADING
"Report 2: Coffee - Actual "
SUM DOLLARS UNITS COLUMN-TOTAL AS 'Total'
BY REGION
ON TABLE PCHOLD FORMAT XLSX
ON TABLE SET STYLE *
INCLUDE=IBFS:/stylesheet_location/custom_stylesheet.sty,$
END
```

***custom_stylesheet*.sty**

```
TYPE=REPORT, TITLETEXT=Coffee Actual, $
TYPE=HEADING, SIZE=14, $
```

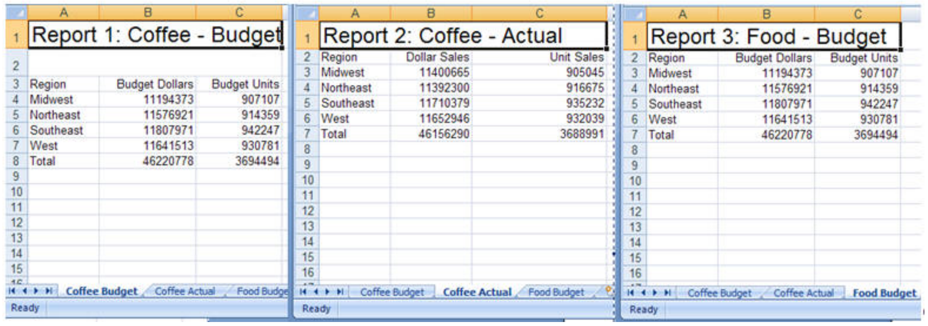
Report Request Report 3

```
TABLE FILE GGSALES
HEADING
"Report 3: Food - Budget"
SUM BUDDOLLARS BUDUNITS COLUMN-TOTAL AS 'Total'
BY REGION
ON TABLE SET STYLE *
INCLUDE=IBFS:/stylesheet_location/custom_stylesheet.sty,$
ON TABLE PCHOLD FORMAT XLSX CLOSE
END
```

***custom_stylesheet*.sty**

```
TYPE=REPORT, TITLETEXT=Food Budget, $
TYPE=HEADING, SIZE=14, $
```

The output is:



Report 1: Coffee - Budget

Region	Budget Dollars	Budget Units
Midwest	11194373	907107
Northeast	11576921	914359
Southeast	11807971	942247
West	11641513	930781
Total	46220778	3694494

Report 2: Coffee - Actual

Region	Dollar Sales	Unit Sales
Midwest	11400665	905045
Northeast	11392300	916675
Southeast	11710379	935232
West	11652946	932039
Total	46156290	3688991

Report 3: Food - Budget

Region	Budget Dollars	Budget Units
Midwest	11194373	907107
Northeast	11576921	914359
Southeast	11807971	942247
West	11641513	930781
Total	46220778	3694494

Reference: Guidelines for Producing Excel Compound Reports Using XLSX

- **Naming of Worksheets.** The default worksheet tab names will be Sheet1, Sheet2, and so on. You have the option to specify a different worksheet tab name by using the TITLETEXT keyword in the StyleSheet. For example:

  ```
  TYPE=REPORT, TITLETEXT='Summary Report',$
  ```

  Excel limits the length of worksheet titles to 31 characters. The following special characters cannot be used: ':', '?', '*', and '/'.

- **File Names and Formats.** The output file name (AS name, or HOLD by default) is obtained from the first report of the compound report (the report with the OPEN keyword). Output file names on subsequent reports are ignored.

  The HOLD FORMAT syntax used in the first component report in a compound report applies to all subsequent reports in the compound report, regardless of their format.

FORMAT XLSX Limitations

Format XLSX does not support the following feature, currently supported for EXL2K:

- Cell locking

Saving Report Output in PPTX Format

Web Query Release 2.1.x introduced the capability to retrieve data from any Web Query supported data source and to generate a PPTX formatted (PowerPoint 2007, PowerPoint 2010, and PowerPoint 2013) presentation.

The PPTX format in Web Query supports the following Microsoft Office software products:

- Microsoft Office 2013/2010/2007.
- Microsoft Office 2000/2003 with the Microsoft Office Compatibility Pack.
- Open Office Support (FORMAT PPTX). Core PowerPoint functionality generated by the PPTX format is supported for Open Office as of Web Query 2.1.x. For details on Open Office, see *http://www.openoffice.org/*.

Web Query generates PPTX presentations based on the Microsoft PPTX standard. These presentations are accessible through all browsers and the Microsoft PowerPoint mobile application.

Note: This section, which describes important PPTX features, applies to PowerPoint 2007, PowerPoint 2010, and PowerPoint 2013, unless otherwise indicated.

Building the .pptx Presentation File

Microsoft changed the format and structure of the PowerPoint presentation file in PowerPoint 2007. The new .pptx file is a binary compilation of a group of .xml files. Generating this new file format using Web Query is a two-step process that consists of generating the .xml files containing the report output and zipping the .xml documents into the binary .pptx format. The Reporting Server performs the xml generation process. The zipping process can be completed either by the Client (Web Query Servlet) or the Server (JSCOM3):

- ❑ **Web Query Servlet.** The Web Query Client within the application server performs the zipping process. This can be done within the local client or through a remotely accessed client.
- ❑ **JSCOM3.** The Java layer of the Reporting Server performs the zipping operation. This option should be used when the Web Query Servlet is configured on a secured web or application server. This is because JSCOM3 does not require URL access to a remote Web Query Client. JSCOM3 is the default for Web Query 2.2.1.

Opening PPTX Report Output

To open PPTX presentations, you must have an account for Microsoft Office 365 or Microsoft PowerPoint 2013, 2010, or 2007 must be installed on the desktop.

Upon execution of a report with FORMAT PPTX, the user is prompted to Open or Save the PPTX file. The file name displayed before the .pptx extension is an internally generated name.

The Web Query procedure generates a presentation containing as many slides as required to display output. A report may contain defined elements, such as headings, subtotals, and titles, as well as StyleSheet syntax, such as conditional styling and drill downs.

Opening PPTX Report Output in Microsoft PowerPoint 2000/2003

PowerPoint 2000 and PowerPoint 2003 can be updated to read PowerPoint PPTX presentations using the Microsoft Office Compatibility Pack available from the Microsoft download site (*http://www.microsoft.com/downloads/en/default.aspx*). When the file extension of the file being opened is .pptx (PPTX presentation), the Microsoft Office Compatibility Pack performs the necessary conversion to allow PowerPoint 2000/2003 to read and open it.

In addition to the Microsoft Office Compatibility Pack, it is important to enable the Web Query Client Redirection Settings Save As option so that PowerPoint 2000/2003 will be able to open the PPTX report output without users first having to save it to their machine with the .pptx file extension. The Web Query Client processing Redirection Settings Save As option configures how the Web Query Client sends each report output file type to the user machine.

This option can be set as follows:

- ❑ **Save As Option disabled (NO).** The Web Query Client Redirection Setting Save As is disabled, by default. When the Save As option is disabled, the Web Query Client sends report output to the user machine in memory with the application association specified for the report format in the Web Query Client Redirection Settings configuration file (mime.wfs).

  A user machine that does not have PowerPoint 2007 or higher installed will not recognize the application association for PowerPoint and PowerPoint will display a message.

  The PowerPoint 2000/2003 user can select Save and provide a file name with the .pptx extension to save the report output to their machine. The user can then open the .pptx file directly from PowerPoint 2000/2003.

- ❑ **Save As Option enabled (YES).** When the Web Query Redirection Save As option is enabled, the Web Query Client sends the report output to the user as a file with the extension specified in the Web Query Client Redirection Settings configuration file (mime.wfs).

  Upon receiving the file, Windows will display the File Download prompt asking the user to Open or Save the file with the identified application type. The File Download prompt displays the Name with the .pptx file extension for the report output that is recognized as a PowerPoint PPTX file type.

  Note: The download prompt will display for all users, including users who have PowerPoint 2007 or higher installed on their machines.

  If a PowerPoint 2000/2003 user chooses to open the file, the Microsoft Office Compatibility Pack will recognize the .pptx file extension and perform the necessary conversion to allow PowerPoint 2000/2003 to read the PowerPoint PPTX presentation.

  If a PowerPoint 2007 or higher user chooses to open the file, PowerPoint will recognize the .pptx file extension and read the PowerPoint PPTX presentation.

Viewing PowerPoint Presentations in the Browser or PowerPoint Application

Your Operating System and desktop settings determine whether PowerPoint output sent to the client is displayed in an Internet Browser window or within the PowerPoint application. When PowerPoint output has been defined within the Windows environment to Browse in same window, the workbook generated by a Web Query request is opened within an Internet Explorer® browser window. When the Browse in same window option is not selected for the .ppt file type, the browser window created by Web Query is blank because the report output is displayed in the stand-alone PowerPoint application window.

❑ In Windows 7, Microsoft removed the desktop settings that support opening worksheets in the browser. This means that to change this behavior, you can no longer simply navigate to the Folder Options dialog box, but you must change a registry setting. This change is documented in the Microsoft Knowledge Base Article ID 927009 at the following website:

  http://support.microsoft.com/kb/927009

  Note: This works the same for both PPT and PPTX formats. The only difference is the selection of file type based on the version of PowerPoint output you will be generating.

Grouping Tables and Components in a PowerPoint Slide

When table elements are placed on a PowerPoint slide, the elements are placed in individual text boxes to allow for explicit positioning to match the other positioned drivers, such as PDF and DHTML.

The PPTXGROUP parameter enables you to group elements together in a PPTX report. You can rotate, flip, move, or resize objects within a group at the same time as though they were a single object. You can also change the attributes of all of the objects in a group at one time, including font, color, or size, and you can ungroup a group of objects at any time, and then regroup them later.

In Web Query, grouping is done within each report component. Objects within the report component (or stand-alone report), including data, all headings and footings, and images, are grouped together. In compound reports (InfoAssist Documents), each component report is grouped individually and non-component elements, such as drawing objects, lines, and images, are not included in any group.

Syntax: How to Group Tables and Components in a PowerPoint Slide

```
SET PPTXGROUP={ON|OFF}
```

The command can also be issued from within a report using:

```
ON TABLE SET PPTXGROUP={ON|OFF}
```

In a StyleSheet:

```
TYPE=REPORT, PPTXGROUP={ON|OFF}
```

where:

`ON`

Enables you to group elements together in a PPTX report.

`OFF`

Indicates no grouping of elements, which is the legacy behavior. OFF is the default value.

Example: Displaying Group Tables and Components in a Standard Report

The following example illustrates how to define grouping at the TYPE=REPORT level for the core report elements, excluding images and drawing objects.

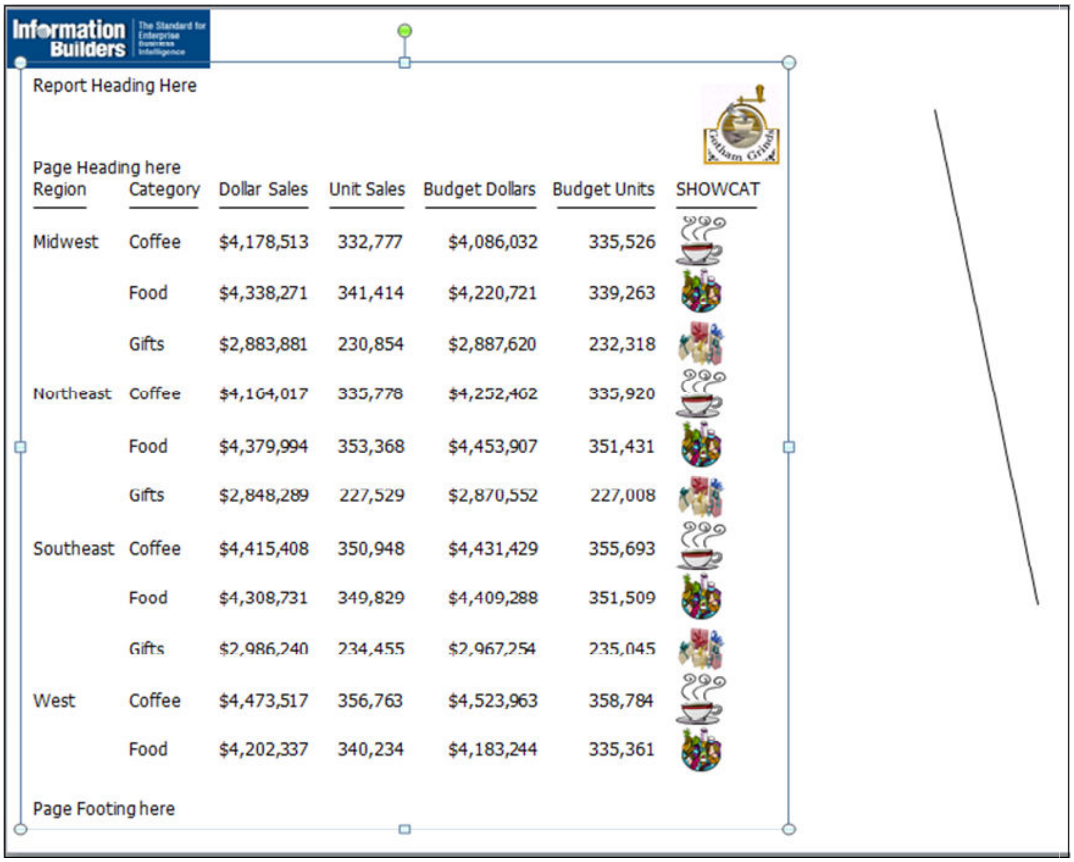
Report Request

```
TABLE FILE GGSALES
SUM DOLLARS/D12CM UNITS/D12C BUDDOLLARS/D12CM BUDUNITS/D12C
COMPUTE SHOWCAT/A100=CATEGORY||'.GIF';
BY REGION
BY CATEGORY
ON TABLE SUBHEAD
"Report Heading Here"
" "
" "
" "
HEADING
"Page Heading here"
FOOTING
"Page Footing here"
ON TABLE SUBFOOT
"Report Footing Here"
ON TABLE PCHOLD AS GROUPTEST FORMAT PPTX
ON TABLE SET STYLE *
INCLUDE=IBFS:/stylesheet_location/custom_stylesheet.sty,$
END
```

***custom_stylesheet*.sty**

```
TYPE=REPORT, PPTXGROUP = ON, SQUEEZE=ON, FONT=TAHOMA,SIZE=12,
   ORIENTATION=LANDSCAPE, $
TYPE=REPORT, OBJECT=LINE, POSITION=(9 1), DIMENSION=(1 5), COLOR=BLUE, $
TYPE=REPORT, IMAGE=IBILOGO, POSITION=(0 0), $
TYPE=TABHEADING, IMAGE=GGLOGO.GIF, POSITION=(6.5 .10), $
TYPE=DATA,COLUMN=SHOWCAT, IMAGE=(SHOWCAT), PRESERVERATIO=ON,
   SIZE=(.5 .5), $
```

The output is:



Report Heading Here

Page Heading here

Region	Category	Dollar Sales	Unit Sales	Budget Dollars	Budget Units	SHOWCAT
Midwest	Coffee	$4,178,513	332,777	$4,086,032	335,526	
	Food	$4,338,271	341,414	$4,220,721	339,263	
	Gifts	$2,883,881	230,854	$2,887,620	232,318	
Northeast	Coffee	$4,164,017	335,778	$4,252,462	335,920	
	Food	$4,379,994	353,368	$4,453,907	351,431	
	Gifts	$2,848,289	227,529	$2,870,552	227,008	
Southeast	Coffee	$4,415,408	350,948	$4,431,429	355,693	
	Food	$4,308,731	349,829	$4,409,288	351,509	
	Gifts	$2,986,240	234,455	$2,967,254	235,045	
West	Coffee	$4,473,517	356,763	$4,523,963	358,784	
	Food	$4,202,337	340,234	$4,183,244	335,361	

Page Footing here

Date and Page/Slide Number

You can add the date and page numbers to both single and compound reports.

- For single reports, use the TABPAGENO feature and the associated attributes.
- For compound reports, add your data and page number to text objects. For more information, see *Text Formatting Markup Tags for a Text Object* on page 381.

Text Formatting Markup Tags for a Text Object

Note: If your text contains any open caret characters (<), you must put a blank space after each open caret that is part of the text, for example, "< 250". If you do not, everything following the open caret will be interpreted as the start of a markup tag and will not display as text.

Font Properties

The font tag supports three attributes: face, size, and color (where the color must be specified as the hexadecimal number code for the color):

`<font face="`*`font`*`" size=[+|-]`*`n`*` color=`*`color_code`*`>text</font>`

For example:

```
<font face="New Century Schoolbook">Test1</font>
<font face="Times" size=12>test2</font>
<font face="Times New Roman" color=#0000FF size=+4>Test3</font>
<font size=-2 face="Times New Roman" color=#0000FF >Test4</font>
```

Text Styles

The supported text styles are bold, italic, underline, and superscript:

Bold: `<b>`*`text`*`</b>`

Italic: `<i>`*`text`*`</i>`

Underline: `<u>`*`text`*`</u>`

Superscript: `<sup>`*`text`*`</sup>`

Line Breaks

The line break tag after a portion of text begins the next portion of text on a new line. Note that there is no closing tag for a line break:

`<br>`

Text Alignment

The alignment options pertain to wrapped text, as well as specified line breaks. Both horizontal justification and vertical alignment are supported.

- **Horizontal Justification**

  Left Justification:

  `<left>`*`text`*`</left>`

  Right Justification:

  `<right>`*`text`*`</right>`

  Center Justification:

  `<center>`*`text`*`</center>`

Full Justification:

```
<full>text</full>
```

- **Vertical Alignment**

  Top Alignment:

  ```
  <top>text</top>
  ```

  Middle Alignment:

  ```
  <mid>text</mid>
  ```

  Bottom Alignment:

  ```
  <bottom>text</bottom>
  ```

Unordered (Bullet) List

The unordered (ul) list tag encloses a bullet list. Each item is enclosed in a list item tag (li). The start tag and end tag for the list must each be on its own line. Each list item must start on a new line:

```
<ul>
<li>list item1</li>
<li>list item2</li>
  .
  .
  .
</ul>
```

By default, the bullet type is disc. You can also specify circle or square:

```
<ul type=disc>

<ul type=circle>

<ul type=square>
```

Ordered (Number or Letter) List

The ordered (ol) list tag encloses a list in which each item has a consecutive number or letter. Each item is enclosed in a list item tag (li). The start tag and end tag for the list must each be on its own line. Each list item must start on a new line:

```
<ol>
<li>list item1</li>
<li>list item2</li>
   .
   .
   .
</ol>
```

By default, Arabic numerals (type=1) are used for the ordering of the list. You can specify the following types of order:

Arabic numerals (the default): `<ol type=1>`

Lowercase letters: `<ol type=a>`

Uppercase letters: `<ol type=A>`

Lowercase Roman numerals: `<ol type=I>`

Uppercase Roman numerals: `<ol type=I>`

Hyperlinks

Hyperlinks can be included within text markup in a PPTX output file.

The syntax for the anchor markup tag is a subset of the HTML anchor syntax:

```
<a href="hyperlink">Text to display</a>
```

where:

`hyperlink`

Is the hyperlink to jump to when the text is clicked.

`Text to display`

Is the text to display for the hyperlink.

For example:

```
<a href="http://www.example.com/help.htm">Click here for help</a>
```

No other attributes are supported in the anchor markup tag.

Page Numbering

There are two pseudo-HTML tags for embedding page numbers in text on a Page Master for a Coordinated Compound Layout report:

Current page number: `<ibi-page-number/>`

Total number of pages: `<ibi-total-pages/>`

Note that when MARKUP=ON, space is allocated for the largest number of pages, so there may be a wide gap between the page number and the text that follows. To remove the extra space in the text object that has the page numbering tags:

❑ If specific styling of the text object is not required, do not insert markup tags, and turn MARKUP=OFF.

```
MARKUP=OFF, TEXT='Page <ibi-page-number/> of <ibi-total-pages/> of Sales
Report', $
```

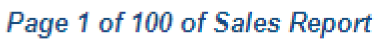
This displays the following output:

```
Page 1 of 100 of Sales Report
```

❑ If specific styling of the text object is required, you must set MARKUP=ON. With MARKUP=ON, set WRAP=OFF and do not place any styling tags between the page number variables within the string. Tags can be used around the complete *Page n of m* string. The following code produces a page number string without the extra spaces:

```
MARKUP=ON, WRAP=OFF, TEXT='<font face="ARIAL" size=10><i>Page <ibi-page-
number/> of <ibi-total-pages/> of Sales Report </i>', $
```

This displays the following output:


Page 1 of 100 of Sales Report

Dates

To display a date in the report output, insert a Web Query date variable in a text object on a Page Master (such as &DATEtrMDYY) in the text object.

Example: Formatting a Compound Layout Text Object With Markup Tags

The following example illustrates a text object with markup tags in a PPTX output file.

Important: Text markup syntax cannot contain hidden carriage return or line feed characters. For purposes of presenting the example in this documentation, line feed characters have been added so that the sample code wraps to fit within the printed page. To run this example in your environment, copy the code into a text editor and delete any line feed characters within the text markup object by going to the end of each line and pressing the Delete key. In some instances, you may need to add a space to maintain the structure of the string. For additional information on displaying carriage returns within the text object, see *Text Formatting Markup Tags for a Text Object* on page 381.

Report Request

```
SET PAGE-NUM=OFF
SET LAYOUTGRID=ON
TABLE FILE GGSALES
BY REGION NOPRINT
ON TABLE PCHOLD AS LINESP1 FORMAT PPTX
ON TABLE SET STYLE *
INCLUDE=IBFS:/stylesheet_location/custom_stylesheet.sty,$
END
```

***custom_stylesheet*.sty**

```
TYPE=REPORT, SIZE=8, $
OBJECT=STRING, POSITION=(1 1), DIMENSION=(7 3), WRAP=ON, MARKUP=ON,
 LINESPACING=MULTIPLE(3),
 TEXT='<b><FONT FACE="Arial" SIZE=12>This paragraph is triple-spaced
 (LINESPACING=MULTIPLE(3)):</font></b>
 <full>Our <i>primary</I> goal for fiscal 2006 was to accelerate our
 transformation to customer centricity. In this letter, I’d like to
 give you an update on this work, which contributed to the 22-percent
 increase in earnings from continuing operations we garnered for fiscal
 2006. Since the past is often prologue to the future, I’d like to
 describe how customer centricity is influencing not only our goals for
 fiscal 2007, but also our long-term plans. At Gotham Grinds, customer
 centricity means treating each customer as a unique individual, meeting
 their needs with end-to-end solutions, and engaging and energizing our
 employees to serve them.</full>', $
```

In this request:

- No fields from the data source are displayed. Only the text object displays on the output.
- The SET LAYOUTGRID command displays a grid to indicate the coordinates and dimensions of the text object.
- The OBJECT=STRING declaration specifies triple spacing: LINESPACING=MULTIPLE(3).

- The following text is displayed in boldface, in the Arial font face, and with a font size of 12 (the default is 8 from the TYPE=REPORT declaration):

  'This paragraph is triple-spaced (LINESPACING=MULTIPLE(3)):'

  The markup for this formatting is:

  ```
  <b><font face="Arial" size=12>This paragraph is triple-spaced
  (LINESPACING=MULTIPLE(3)):</font></b>
  ```

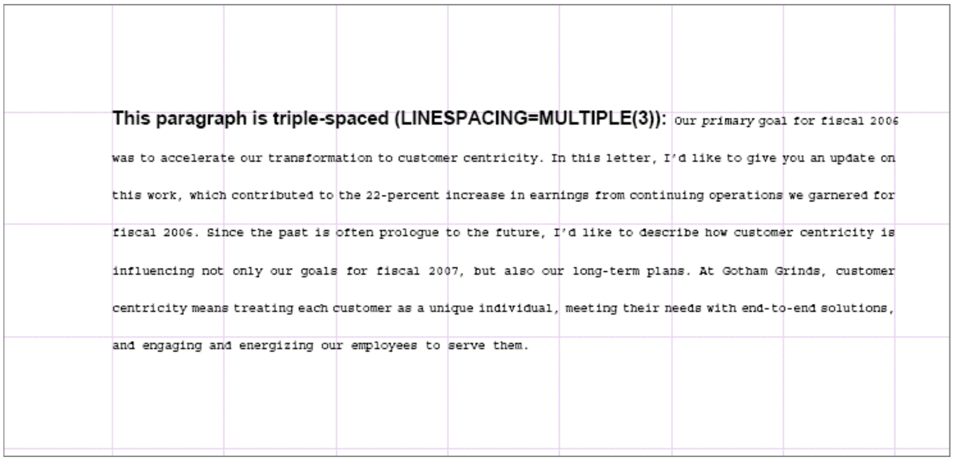
- The remainder of the text is displayed with full justification (left and right sides align):

  ```
  <full>Our ... </full>
  ```

- The following markup displays the text 'primary' in italics:

  ```
  <i>primary</I>
  ```

The output is:



This paragraph is triple-spaced (LINESPACING=MULTIPLE(3)): Our *primary* goal for fiscal 2006 was to accelerate our transformation to customer centricity. In this letter, I'd like to give you an update on this work, which contributed to the 22-percent increase in earnings from continuing operations we garnered for fiscal 2006. Since the past is often prologue to the future, I'd like to describe how customer centricity is influencing not only our goals for fiscal 2007, but also our long-term plans. At Gotham Grinds, customer centricity means treating each customer as a unique individual, meeting their needs with end-to-end solutions, and engaging and energizing our employees to serve them.

Example: Vertically Aligning Text Markup in PPTX Report Output

The following example illustrates how to create three boxes and place a text string object within each of them:

- In the left box, the text is aligned vertically at the top.
- In the middle box, the text is aligned vertically at the middle.

- In the right box, the text is aligned vertically at the bottom.

Important: Text markup syntax cannot contain hidden carriage return or line feed characters. For purposes of presenting the example in this documentation, line feed characters have been added so that the sample code wraps to fit within the printed page. To run this example in your environment, copy the code into a text editor and delete any line feed characters within the text markup object by going to the end of each line and pressing the Delete key. In some instances, you may need to add a space to maintain the structure of the string. For additional information on displaying carriage returns within the text object see *Text Formatting Markup Tags for a Text Object* on page 381.

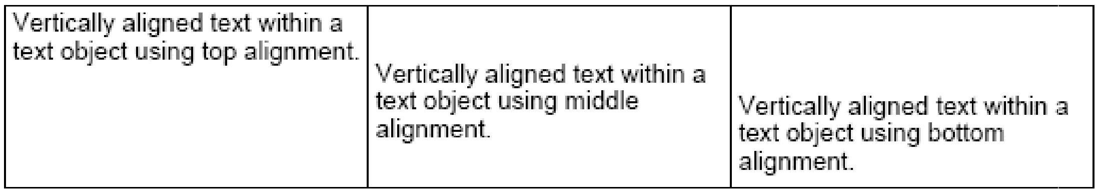
Report Request

```
SET PAGE-NUM=OFF
TABLE FILE GGSALES
BY REGION NOPRINT
ON TABLE PCHOLD FORMAT PPTX
ON TABLE SET STYLE *
INCLUDE=IBFS:/stylesheet_location/custom_stylesheet.sty,$
END
```

***custom_stylesheet*.sty**

```
TYPE=REPORT, FONT=Arial, SIZE=10, $
OBJECT=BOX, POSITION=(1 1), DIMENSION=(6 1), $
OBJECT=LINE, POSITION=(3 1), ENDPOINT=(3 2), $
OBJECT=LINE, POSITION=(5 1), ENDPOINT=(5 2), $
OBJECT=STRING, TEXT='<top>Vertically aligned text within a text object
using top alignment.</top>', POSITION=(1.05 1), DIMENSION=(2 1),
LINESPACING=EXACT(.15), MARKUP=ON, WRAP=ON, $
OBJECT=STRING, TEXT='<mid>Vertically aligned text within a text object
using middle alignment.</mid>', POSITION=(3.05 1), DIMENSION=(2 1),
LINESPACING=EXACT(.15), MARKUP=ON, WRAP=ON, $
OBJECT=STRING, TEXT='<bottom>Vertically aligned text within a text object
using bottom alignment.</bottom>', POSITION=(5.05 .9),
DIMENSION=(2 1), LINESPACING=EXACT(.15), MARKUP=ON, WRAP=ON, $
```

The output is:



Vertically aligned text within a text object using top alignment.

Vertically aligned text within a text object using middle alignment.

Vertically aligned text within a text object using bottom alignment.

Inserting Images In Various Elements of PowerPoint PPTX Reports

Web Query supports the placement of images within each element or node of the report. An image, such as a logo, gives corporate identity to a report, or provides visual appeal. Data specific images can be placed in headers and footers to provide additional clarity and style. The image must reside on the Web Query Reporting Server in a directory named on EDAPATH or APPPATH. If the file is not on the search path, supply the full path name.

All images will be placed in the defined area, based on the explicit positioning defined by the POSITION attribute within the style sheet.

Images can be placed in any available Web Query reporting node or element. Supported image formats include .gif, .jpg, and .png. Images may be positioned and resized by using the POSITION and SIZE attributes to set the x, y coordinates and height, width settings, respectively. Justification of images is not supported.

Note: The highest quality image format for charts is PNG, which allows for transparency, as well as better integration with the styling within slide backgrounds.

Syntax: How to Insert Images Into Web Query PPTX Reports

```
TYPE={REPORT|HEADING|data}, IMAGE={file|(column)}
   [,BY=byfield] [,SIZE=(w h)] ,$
```

where:

REPORT

Embeds an image in the body of a report. The image appears in the background of the report. REPORT is the default value.

HEADING

Embeds an image in a heading or footing. Valid values are TABHEADING, TABFOOTING, FOOTING, HEADING, SUBHEAD, and SUBFOOT. Provide sufficient blank space in the heading or footing so that the image does not overlap the heading or footing text. You may also want to place heading or footing text to the right of the image using spot markers.

data

Defines a data column in which to place the image. Must be used with COLUMNS=*column title* to identify the specific report column where the image should be anchored.

file

Is the name of the image file. It must reside on the Web Query Reporting Server in a directory named on EDAPATH or APPPATH. If the file is not on the search path, supply the full path name. When specifying a GIF file, you can omit the file extension.

Example: Inserting Images in the Headers and Footers of a Report

The following example illustrates how to insert images in the headers and footers of a report.

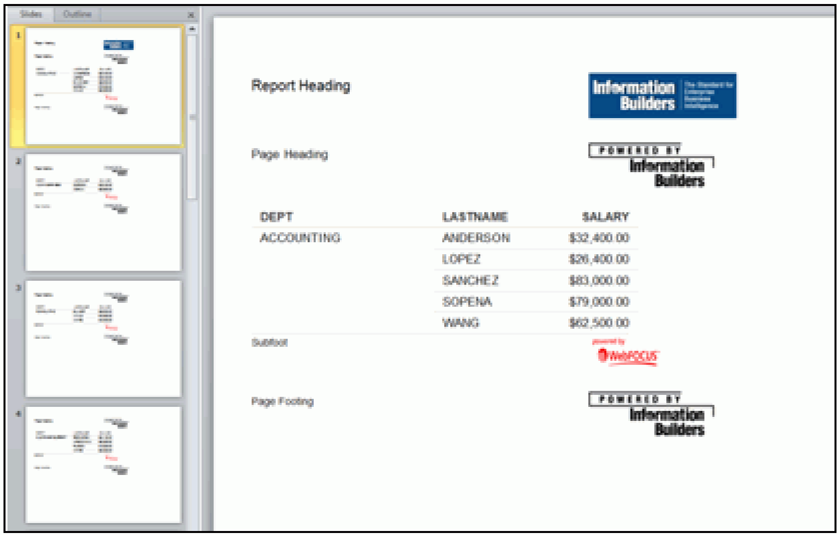
Report Request

```
TABLE FILE EMPDATA
SUM
EMPDATA.EMPDATA.SALARY
BY LOWEST EMPDATA.EMPDATA.DEPT
BY EMPDATA.EMPDATA.LASTNAME
ON EMPDATA.EMPDATA.DEPT SUBFOOT "Subfoot"
" "
ON EMPDATA.EMPDATA.DEPT PAGE-BREAK
ON TABLE SUBHEAD
"Report Heading"
" "
HEADING
"Page Heading"
" "
FOOTING
"Page Footing"
" "
ON TABLE SUBFOOT "Report Footing"
" "
ON TABLE SET PAGE-NUM NOLEAD
ON TABLE NOTOTAL
ON TABLE PCHOLD FORMAT PPTX
ON TABLE SET STYLE *
INCLUDE=IBFS:/stylesheet_location/custom_stylesheet.sty,$
END
```

***custom_stylesheet*.sty**

```
TYPE=REPORT, PAGESIZE='PPT Slide', ORIENTATION=LANDSCAPE, $
TYPE=REPORT, OBJECT=STATUS-AREA, JUSTIFY=LEFT, PAGE-LOCATION=BOTTOM, $
TYPE=REPORT, GRID=OFF, FONT='ARIAL', SIZE=12, STYLE=NORMAL, SQUEEZE=ON,
   TOPGAP=0.05, BOTTOMGAP=0.05, BORDER-COLOR=RGB(219 219 219),
TITLELINE=SKIP,
TOPMARGIN=.75, LEFTMARGIN=.5, $
TYPE=TITLE, COLOR=RGB(51 51 51), STYLE=-UNDERLINE +BOLD, $
TYPE=DATA, BORDER-TOP=LIGHT, BORDER-TOP-COLOR=RGB(219 219 219), $
TYPE=SUBTOTAL, STYLE=BOLD, BORDER-TOP=LIGHT,
   BORDER-TOP-COLOR=RGB(219 219 219), $
TYPE=TABHEADING, SIZE=14, JUSTIFY=LEFT, $
TYPE=TABHEADING, IMAGE=smplogo1.gif, POSITION=(+4.500000 +0.000000), $
TYPE=TABFOOTING, SIZE=10, $
TYPE=SUBHEAD, BACKCOLOR=RGB(246 246 246), BORDER-TOP=LIGHT,
   BORDER-TOP-COLOR=RGB(219 219 219), $
TYPE=SUBHEAD, BY=1, BORDER-TOP=LIGHT, BORDER-TOP-COLOR=RGB(102 102 102), $
TYPE=SUBHEAD, OBJECT=FIELD, STYLE=BOLD, $
TYPE=SUBFOOT, SIZE=9, BORDER-TOP=LIGHT, BORDER-TOP-COLOR=RGB(219 219 219), $
TYPE=TABFOOTING, IMAGE=smplogo1.gif, POSITION=(+4.500000 +0.000000), $
TYPE=HEADING, SIZE=12, $
TYPE=HEADING, IMAGE=poweredbyibi.gif, POSITION=(+4.500000 +0.000000), $
TYPE=FOOTING, SIZE=10, $
TYPE=FOOTING, STYLE=BOLD, JUSTIFY=LEFT, IMAGE=poweredbyibi.gif,
   POSITION=(+4.500000 +0.000000), $
TYPE=SUBFOOT, SIZE=10, $
TYPE=SUBFOOT, IMAGE=Web Query1.gif, POSITION=(+4.500000 +0.000000),
   SIZE=(1.000000 0.500000), $
```

The output is:


Slides
Outline
Information Builders
POWERED BY Information Builders
powered by WebFOCUS
POWERED BY Information Builders

Report Heading

Page Heading

DEPT	LASTNAME	SALARY
ACCOUNTING	ANDERSON	$32,400.00
	LOPEZ	$26,400.00
	SANCHEZ	$83,000.00
	SOPENA	$79,000.00
	WANG	$62,500.00

Subfoot

Page Footing

Example: Inserting Images in the Data Cells of a Report

The following example illustrates how to insert images in the data cells of a report.

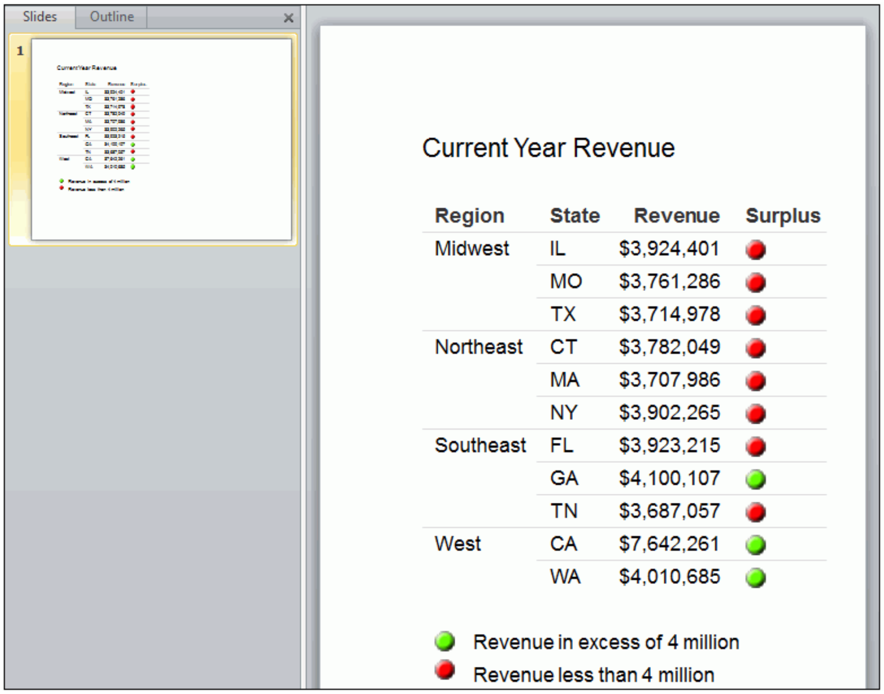
Report Request

```
APP PATH IBISAMP IBIDEMO
TABLE FILE GGSALES
SUM DOLLARS/D17M AS 'Revenue'
COMPUTE Surplus/A15 = IF DOLLARS GE 4000000 THEN 'g1.gif' ELSE 'r1.gif';
BY REGION
BY ST
ON TABLE SUBHEAD
"Current Year Revenue"
FOOTING
"Revenue in excess of 4 million"
"Revenue less than 4 million"
ON TABLE SET PAGE-NUM NOLEAD
ON TABLE PCHOLD FORMAT PPTX
ON TABLE SET SQUEEZE ON
ON TABLE SET STYLE *
INCLUDE=IBFS:/stylesheet_location/custom_stylesheet.sty,$
END
```

***custom_stylesheet*.sty**

```
TYPE=REPORT, GRID=OFF, FONT='ARIAL', SIZE=14, STYLE=NORMAL, SQUEEZE=ON,
   TOPGAP=0.05, BOTTOMGAP=0.05, BORDER-COLOR=RGB(219 219 219),
   TITLELINE=SKIP, ORIENTATION=LANDSCAPE, TOPMARGIN=1, LEFTMARGIN=1, $
TYPE=DATA, COLUMN=Surplus, IMAGE=(Surplus), SIZE=(.2 .2),$
TYPE=FOOTING, IMAGE=g1.gif, POSITION=(.122 .055), SIZE=(.2 .2 ),$
TYPE=FOOTING, LINE=1,ITEM=1, POSITION=.5,$
TYPE=FOOTING, IMAGE='r1.gif', POSITION=(.122 .33), SIZE=(.2 .2),$
TYPE=FOOTING, LINE=2, ITEM=1, POSITION=.5,$
TYPE=DATA, BORDER-TOP=LIGHT, BORDER-TOP-COLOR=RGB(219 219 219), $
TYPE=TITLE, COLOR=RGB(51 51 51), STYLE=-UNDERLINE +BOLD, $
TYPE=TABHEADING, SIZE=18, JUSTIFY=LEFT, $
TYPE=TABFOOTING, SIZE=10, $
TYPE=HEADING, JUSTIFY=LEFT, SIZE=12, $
TYPE=SUBHEAD, BACKCOLOR=RGB(246 246 246), BORDER-TOP=LIGHT,
   BORDER-TOP-COLOR=RGB(219 219 219), $
TYPE=SUBHEAD, BY=1, BORDER-TOP=LIGHT, BORDER-TOP-COLOR=RGB(102 102 102), $
TYPE=SUBHEAD, OBJECT=FIELD, STYLE=BOLD, $
TYPE=SUBFOOT, SIZE=9, BORDER-TOP=LIGHT, BORDER-TOP-COLOR=RGB(219 219 219), $
```

The output is:



Current Year Revenue

Region	State	Revenue	Surplus
Midwest	IL	$3,924,401	🔴
	MO	$3,761,286	🔴
	TX	$3,714,978	🔴
Northeast	CT	$3,782,049	🔴
	MA	$3,707,986	🔴
	NY	$3,902,265	🔴
Southeast	FL	$3,923,215	🔴
	GA	$4,100,107	🟢
	TN	$3,687,057	🔴
West	CA	$7,642,261	🟢
	WA	$4,010,685	🟢

🟢 Revenue in excess of 4 million
🔴 Revenue less than 4 million

Drill Down From Microsoft PowerPoint

Two types of drill downs are supported:

- ❏ Web Query content
- ❏ External URL

When working in the Web Query Repository or Content environment, drill-down hyperlinks in PPTX reports will not work when Microsoft PowerPoint opens in a PowerPoint application window instead of in a browser. The current security context and any previously established session-related cookies are not retained and this changes user authorization.

Example: Drilling Down to an External URL

The following example illustrates how to place a reference to an external URL on the grand total line tag.

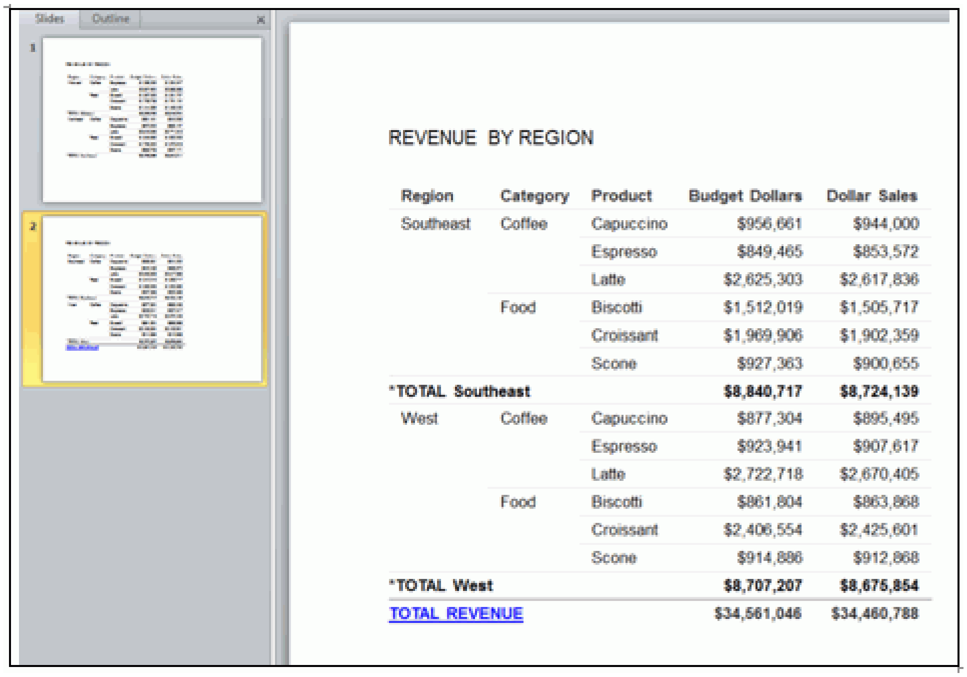
Report Request

```
TABLE FILE GGSALES
SUM
  GGSALES.SALES01.BUDDOLLARS/D12CM
  GGSALES.SALES01.DOLLARS/D12CM
BY GGSALES.SALES01.REGION
BY GGSALES.SALES01.CATEGORY
BY GGSALES.SALES01.PRODUCT
HEADING
"REVENUE BY REGION "
ON GGSALES.SALES01.REGION SUBTOTAL AS '*TOTAL'
WHERE GGSALES.SALES01.CATEGORY NE 'Gifts';
ON TABLE SET PAGE-NUM NOLEAD
ON TABLE COLUMN-TOTAL AS 'TOTAL'
ON TABLE PCHOLD FORMAT PPTX
ON TABLE SET STYLE *
INCLUDE=IBFS:/stylesheet_location/custom_stylesheet.sty,$
END
```

***custom_stylesheet*.sty**

```
TYPE=REPORT, PAGESIZE='PPT Slide', ORIENTATION=LANDSCAPE, $
TYPE=REPORT, OBJECT=STATUS-AREA, JUSTIFY=LEFT, PAGE-LOCATION=BOTTOM, $
TYPE=REPORT, GRID=OFF, FONT='ARIAL', SIZE=12, STYLE=NORMAL, SQUEEZE=ON,
   TOPGAP=0.05, BOTTOMGAP=0.05, BORDER-COLOR=RGB(219 219 219),
   TITLELINE=SKIP, TOPMARGIN=1, LEFTMARGIN=1, $
TYPE=TITLE, COLOR=RGB(51 51 51), STYLE=-UNDERLINE +BOLD, $
TYPE=DATA, BORDER-TOP=LIGHT, BORDER-TOP-COLOR=RGB(219 219 219), $
TYPE=HEADING, JUSTIFY=LEFT, SIZE=14, $
TYPE=SUBTOTAL, STYLE=BOLD, BORDER-TOP=LIGHT,
   BORDER-TOP-COLOR=RGB(219 219 219), $
TYPE=GRANDTOTAL, OBJECT=TAG, URL=http://www.ibi.com, $
TYPE=REPORT, OBJECT=STATUS-AREA, JUSTIFY=LEFT, PAGE-LOCATION=BOTTOM, $
TYPE=GRANDTOTAL, COLOR=RGB(51 51 51), STYLE=BOLD, BORDER-TOP=LIGHT,
   BORDER-TOP-COLOR=RGB(102 102 102), $
```

The output is:



Slides Outline

REVENUE BY REGION

Region	Category	Product	Budget Dollars	Dollar Sales
Southeast	Coffee	Capuccino	$956,661	$944,000
		Espresso	$849,465	$853,572
		Latte	$2,625,303	$2,617,836
	Food	Biscotti	$1,512,019	$1,505,717
		Croissant	$1,969,906	$1,902,359
		Scone	$927,363	$900,655
***TOTAL Southeast**			**$8,840,717**	**$8,724,139**
West	Coffee	Capuccino	$877,304	$895,495
		Espresso	$923,941	$907,617
		Latte	$2,722,718	$2,670,405
	Food	Biscotti	$861,804	$863,868
		Croissant	$2,406,554	$2,425,601
		Scone	$914,886	$912,868
***TOTAL West**			**$8,707,207**	**$8,675,854**
TOTAL REVENUE			$34,561,046	$34,460,788

Chapter 11

Linking a Report to Other Resources

You can use StyleSheet declarations to define links from any report component. You can use links to:

- ❏ Create a series of drill-down reports by linking the procedures that generate these reports.
- ❏ Link to URLs. These can be other webpages, websites, Servlet programs, or non-World Wide Web resources, such as an email application.
- ❏ Execute JavaScript functions to perform additional analysis of the report data.

You can create links from report data, as well as graphical images within a report. You can also create links from a graph.

Note: InfoAssist has built-in styling options that generate some of the StyleSheet syntax described in this chapter.

In this chapter:

Linking Using StyleSheets

You can use StyleSheets to define a link from any report component. You can create links from report data (including headings and footings), as well as graphic images (such as a company logo or product image), to other reports, procedures, URLs, or JavaScript functions.

The links you create can be dynamic. With a dynamic link, your selection passes the value of the selected report component to the linked report (procedure, URL, or JavaScript function). The resource uses the passed value to dynamically determine the results that are returned. You can pass one or more parameters.

Procedure: How to Create Links Using StyleSheets

This procedure is a basic overview of how to create links using StyleSheets.

1. Identify the report component that the user selects in the web browser to execute the link.
2. Specify the name of the embedded procedure, URL, or JavaScript function to execute.
3. Identify the parameters that define the specifics of your link, if necessary.

Linking to Another Report

A link allows you to drill down to a report for more details or execute a procedure by selecting a designated hot spot (the link) in the report. By linking reports, you provide easy access to more detailed data that supplements the information in your base report. The drill-down report can contain information that is either independent of the data in the base report or depends and expands on a specific data value in the base report.

To create a link, you must have a report to link from (the base report) and a report to link to (the drill-down report). If the drill-down report depends on a specific data value in the base report, you also need to pass that value to the drill-down report by creating parameters.

Syntax: How to Link to Reports and Procedures

```
TYPE=type, [subtype], FOCEXEC=fex[(parameters ...)], [TARGET=frame,]
   [ALT = 'description',] $
```

where:

type

Identifies the report component that you select in the web browser to execute the link. The TYPE attribute and its value must appear at the beginning of the declaration.

`subtype`

Are any additional attributes, such as COLUMN, LINE, or ITEM, that are needed to identify the report component that you are formatting. For information on identifying report components, see *Identifying a Report Component in a Web Query StyleSheet* on page 57.

`fex`

Identifies the file name of the linked procedure to run when you select the report component.

The maximum length of a FOCEXEC=*fex* argument, including any associated parameters, is 2400 characters. The FOCEXEC argument can span more than one line, as described in *Introducing Formatting and Style Sheets* on page 27.

`parameters`

Values that are passed to the report, URL, or JavaScript function.

`frame`

Identifies the target frame in the webpage in which the output from the drill-down link is displayed. For details, see *Specifying a Target Frame* on page 414.

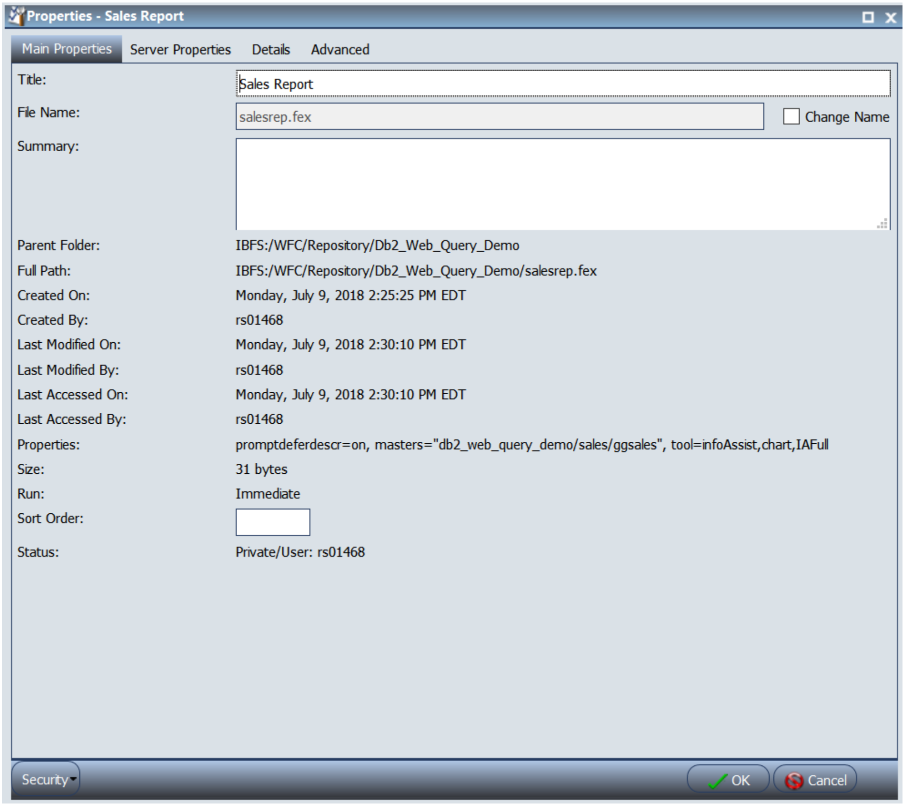
`description`

Is a textual description of the link supported in an HTML report for compliance with Section 508 accessibility. Enclose the description in single quotation marks (').

The description also displays as a pop-up description when your mouse or cursor hovers over the link in the report output.

Procedure: How to Determine a Web Query File Name in a Dashboard

1. Right-click the report name and select *Properties*. The Report Properties dialog box opens.
2. The file name appears under *File Name*. In the example below, the name of the file is *salesrep*. Do not include the file extension (.fex).



Title:	Sales Report
File Name:	salesrep.fex
Summary:	
Parent Folder:	IBFS:/WFC/Repository/Db2_Web_Query_Demo
Full Path:	IBFS:/WFC/Repository/Db2_Web_Query_Demo/salesrep.fex
Created On:	Monday, July 9, 2018 2:25:25 PM EDT
Created By:	rs01468
Last Modified On:	Monday, July 9, 2018 2:30:10 PM EDT
Last Modified By:	rs01468
Last Accessed On:	Monday, July 9, 2018 2:30:10 PM EDT
Last Accessed By:	rs01468
Properties:	promptdeferdescr=on, masters="db2_web_query_demo/sales/ggsales", tool=infoAssist,chart,IAFull
Size:	31 bytes
Run:	Immediate
Sort Order:	
Status:	Private/User: rs01468

Example: Linking to a Report From a Footing

The following example illustrates how to summarize product sales and sort the data by region, state, and store code. The store code also displays in the subfootings where links to detailed reports about the store sales (by product or by date) display. Each line of the subfoot contains two text objects and one embedded field.

Report Request

```
TABLE FILE GGSALES
HEADING
"Sales Report"
SUM DOLLARS/I08M
BY REGION BY ST BY STCD
ON STCD SUBFOOT
"View Store <STCD Sales By Product"
" "
"View Store <STCD Sales By Date"
ON REGION PAGE-BREAK
ON TABLE SET STYLE *
INCLUDE=IBFS:/stylesheet_location/custom_stylesheet.sty,$
END
```

***custom_stylesheet*.sty**

```
TYPE=HEADING, SIZE=12, STYLE=BOLD, $
TYPE=SUBFOOT, LINE=1, OBJECT=TEXT, ITEM=2, COLOR=GREEN,
   FOCEXEC=PRDSALES(STOREID=STCD), $
TYPE=SUBFOOT, LINE=3, OBJECT=TEXT, ITEM=2, COLOR=BLUE,
   FOCEXEC=HSTSALES(STOREID=STCD), $
```

Using StyleSheet declarations, the subfoot phrase *Sales By Product* links to a second procedure named PRDSALES and passes it the value of STCD displayed in the subfoot. The subfoot phrase *Sales By Date* links to a procedure named HSTSALES and passes it the value of STCD displayed in the subfoot.

Report Request for the Linked Report HSTSALES

```
TABLE FILE GGSALES
SUM UNITS
BY STCD
BY DATE
WHERE STCD = '&STOREID'
ON TABLE SET STYLE *
INCLUDE=IBFS:/stylesheet_location/custom_stylesheet.sty,$
END
```

***custom_stylesheet*.sty**

```
TYPE=REPORT, GRID=OFF, $
```

Report Request for the Linked Report PRDSALES

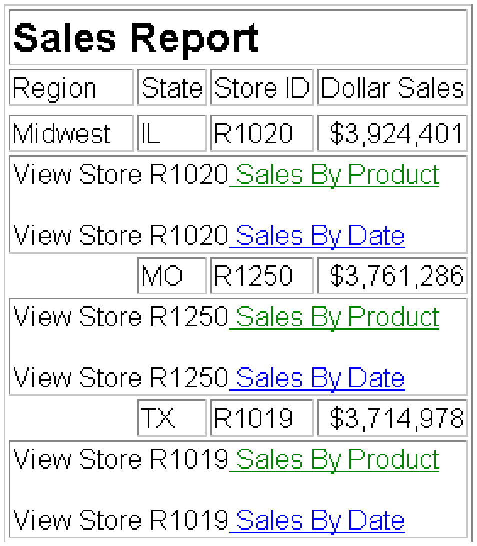
```
TABLE FILE GGSALES
SUM UNITS
BY STCD
BY PRODUCT
WHERE STCD = '&STOREID'
ON TABLE SET STYLE *
INCLUDE=IBFS:/stylesheet_location/custom_stylesheet.sty,$
END
```

***custom_stylesheet*.sty**

```
TYPE=REPORT, GRID=OFF, $
```

The first page of output for the main report follows. If you select *Sales By Product* for Store R1020, the value R1020 is passed to the PRDSALES procedure. If you select *Sales By Date* for Store R1019, the value R1019 is passed to the HSTSALES procedure.

The output is:



Sales Report

Region	State	Store ID	Dollar Sales
Midwest	IL	R1020	$3,924,401
View Store R1020 Sales By Product View Store R1020 Sales By Date			
	MO	R1250	$3,761,286
View Store R1250 Sales By Product View Store R1250 Sales By Date			
	TX	R1019	$3,714,978
View Store R1019 Sales By Product View Store R1019 Sales By Date			

If you click the *Sales By Product* link for store R1020, the output is:

```
Store ID      Product           Unit Sales
R1020         Biscotti               29413
              Coffee Grinder         19339
              Coffee Pot             15785
              Croissant              43300
              Espresso               32237
              Latte                  77344
              Mug                    30157
              Scone                  45355
              Thermos                14651
```

Linking to a URL

You can define a link from any report component to any URL including webpages, websites, Servlet programs, or non-World Wide Web resources, such as an email application. After you have defined a link, you can select the report component to access the URL.

The links you create can be dynamic. With a dynamic link, your selection passes the value of the selected report component to the URL. The resource uses the passed value to dynamically determine the results that are returned. You can pass one or more parameters.

Syntax: How to Link to a URL

```
TYPE=type, [subtype], URL=url[(parameters ...)], [TARGET=frame,]
   [ALT='description',] $
```

where:

type

Identifies the report component that you select in the web browser to execute the link. The TYPE attribute and its value must appear at the beginning of the declaration.

subtype

Are any additional attributes, such as COLUMN, LINE, or ITEM, that are needed to identify the report component that you are formatting. For information on identifying report components, see *Identifying a Report Component in a Web Query StyleSheet* on page 57.

url

Identifies any valid URL, including a URL that specifies a Web Query Servlet program, or the name of a report column enclosed in parentheses () whose value is a valid URL to which the link will jump.

Note:

- ❑ The maximum length of a URL=*url* argument, including any associated variable=object parameters, is limited by the maximum number of characters allowed by the browser. For information about this limit for your browser, search your browser support site. The URL argument can span more than one line, as described in *Introducing Formatting and Style Sheets* on page 27

  Note that the length of the URL is limited by the maximum number of characters allowed by the browser. For information about this limit for your browser, search your browser support site.

- ❑ If the URL refers to a Web Query Servlet program that takes parameters, the URL must end with a question mark (?).

parameters

Values that are passed to the URL.

frame

Identifies the target frame in the webpage in which the output from the drill-down link is displayed. For details, see *Specifying a Target Frame* on page 414.

description

Is a textual description of the link supported in an HTML report for compliance with Section 508 accessibility. Enclose the description in single quotation marks (').

The description also displays as a pop-up description when your mouse or cursor hovers over the link in the report output.

Example: Linking to a URL

The following example illustrates how to link to a URL from a report. The heading *Click here to access the IBM homepage* is linked to the URL https://www.ibm.com/us-en/.

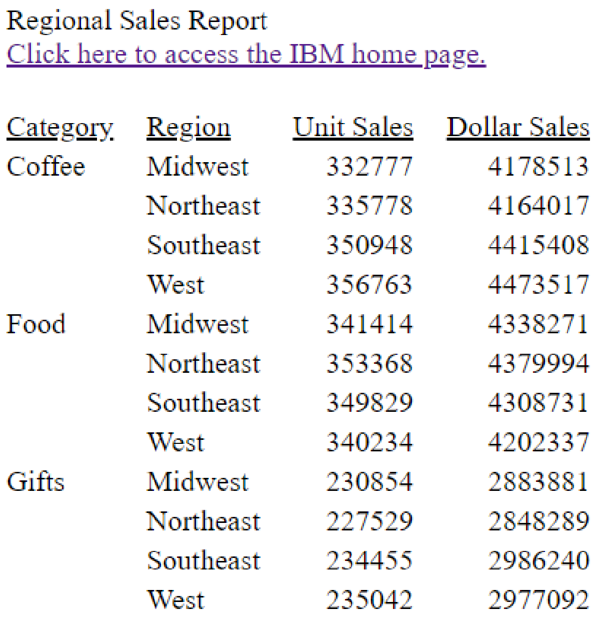
Report Request

```
TABLE FILE GGSALES
SUM UNITS AND DOLLARS
BY CATEGORY BY REGION
HEADING
"Regional Sales Report"
"Click here to access the IBM support page."
ON TABLE SET STYLE *
INCLUDE=IBFS:/stylesheet_location/custom_stylesheet.sty,$
END
```

***custom_stylesheet*.sty**

```
TYPE=REPORT, GRID=OFF, $
TYPE=HEADING, LINE=2, OBJECT=TEXT, ITEM=1,
   URL=https://www.ibm.com/us-en/, $
```

The output is:



```
Regional Sales Report
Click here to access the IBM home page.

Category   Region      Unit Sales   Dollar Sales
Coffee     Midwest         332777        4178513
           Northeast       335778        4164017
           Southeast       350948        4415408
           West            356763        4473517
Food       Midwest         341414        4338271
           Northeast       353368        4379994
           Southeast       349829        4308731
           West            340234        4202337
Gifts      Midwest         230854        2883881
           Northeast       227529        2848289
           Southeast       234455        2986240
           West            235042        2977092
```

When you click the link the site displays in your browser.

Linking to a JavaScript Function

You can use a StyleSheet to define a link to a JavaScript function from any report component. After you have defined the link, you can select the report component to execute the JavaScript function.

Just as with drill-down links to procedures and URLs, you can specify optional parameters that allow values of a report component to be passed to the JavaScript function. The function will use the passed value to dynamically determine the results that are returned to the browser.

Note:

- JavaScript functions can, in turn, call other JavaScript functions.
- You cannot specify a target frame if you are executing a JavaScript function. However, the JavaScript function itself can specify a target frame for its results.

Syntax:

How to Link to a JavaScript Function

```
TYPE=type, [subtype], JAVASCRIPT=function[(parameters ...)], $
```

where:

`type`

Identifies the report component that you select in the web browser to execute the link. The TYPE attribute and its value must appear at the beginning of the declaration.

`subtype`

Are any additional attributes, such as COLUMN, LINE, or ITEM, that are needed to identify the report component that you are formatting. See *Identifying a Report Component in a Web Query StyleSheet* on page 57 for details.

`function`

Identifies the JavaScript function to run when you select the report component.

The maximum length of a JAVASCRIPT=*function* argument, including any associated parameters, is 2400 characters and can span more than one line. If you split a single argument across a line, you need to use the slash (\) character at the end of the first line, as continuation syntax. If you split an argument at a point where a space is required as a delimiter, the space must be before the slash (\) character or be the first character on the next line. The slash (\) character does not act as the delimiter.

In the following example, the argument correctly spans two lines.

```
JAVASCRIPT=myfunc(COUNTRY \
CAR MODEL 'ABC'),$
```

parameters

Values that are passed to the JavaScript function.

Linking With Conditions

You can create conditions when linking to a report, URL, or JavaScript function from a report or chart. For example, you may only be interested in displaying current salaries for a particular department. You can accomplish this by creating a WHEN condition.

For complete details on WHEN, see *Controlling Report Formatting* on page 41.

Note: Linking with conditions is not supported in GRAPH requests.

Syntax: How to Link With Conditions

To specify a conditional link to a report use:

```
TYPE=type, [subtype], FOCEXEC=fex[(parameters...)],
   WHEN=expression, [TARGET=frame,] $
```

To specify a conditional link to a URL use:

```
TYPE=type, [subtype], URL=url[(parameters...)],
   WHEN=expression, [TARGET=frame,] $
```

To specify a conditional link to a JavaScript function use:

```
TYPE=type, [subtype], JAVASCRIPT=function[(parameters...)],
   WHEN=expression, [TARGET=frame,] $
```

where:

type

Identifies the report component that you select in the web browser to execute the link. The TYPE attribute and its value must appear at the beginning of the declaration.

subtype

Are any additional attributes, such as COLUMN, LINE, or ITEM, that are needed to identify the report component that you are formatting. For information on identifying report components, see *Identifying a Report Component in a Web Query StyleSheet* on page 57.

fex

Identifies the file name of the linked procedure to run when you select the report component. For details about linking to another procedure, see *Linking to Another Report* on page 398.

url

Identifies any valid URL, or the name of a report column enclosed in parentheses () whose value is a valid URL. For details about linking to a URL, see *Linking to a URL* on page 403.

function

Identifies the JavaScript function to run when you select the report component. For details about calling a JavaScript function, see *Linking to a JavaScript Function* on page 406.

parameters

Values that are passed to the report, URL, or JavaScript function.

expression

Is any Boolean expression that would be valid on the right side of a COMPUTE expression.

Note: IF... THEN... ELSE logic is not necessary in a WHEN clause and is not supported. All non-numeric literals in a WHEN expression must be specified within single quotation marks (’).

frame

Identifies the target frame in the webpage in which the output from the drill-down link is displayed. For details, see *Specifying a Target Frame* on page 414.

Example: Linking With Conditions

The following example illustrates how to link the MIS value of the DEPARTMENT field to REPORT3. To do this, include the phrase WHEN=DEPARTMENT EQ 'MIS' in the StyleSheet declaration.

Report Request

```
TABLE FILE EMPLOYEE
SUM CURR_SAL AS 'Total,Current,Salaries'
BY DEPARTMENT AS 'Department'
ON TABLE SET STYLE *
INCLUDE=IBFS:/stylesheet_location/custom_stylesheet.sty,$
END
```

***custom_stylesheet*.sty**

```
TYPE=REPORT, GRID=OFF, $
TYPE=DATA, COLUMN=DEPARTMENT, FOCEXEC=REPORT3(DEPARTMENT=N1),
   WHEN=DEPARTMENT EQ 'MIS', $
```

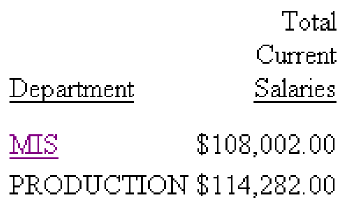
Drill-Down Report Request (REPORT3)

```
TABLE FILE EMPLOYEE
PRINT SALARY
BY DEPARTMENT
BY LAST_NAME
WHERE DEPARTMENT EQ '&DEPARTMENT'
ON TABLE SET STYLE *
INCLUDE=IBFS:/stylesheet_location/custom_stylesheet.sty,$
END
```

***custom_stylesheet*.sty**

```
TYPE=REPORT, GRID=OFF, $
```

In the following output, note that only the MIS department is linked:



Department	Total Current Salaries
MIS	$108,002.00
PRODUCTION	$114,282.00

When you click *MIS*, the following output displays:

```
DEPARTMENT     LAST NAME      SALARY

MIS              BLACKWOOD    $21,780.00

                 CROSS        $27,062.00

                              $25,755.00

                 GREENSPAN     $9,000.00

                               $8,650.00

                 JONES        $18,480.00

                              $17,750.00

                 MCCOY        $18,480.00

                 SMITH        $13,200.00
```

Linking From a Graphic Image

You can link to a report or procedure from an image in an HTML report. The image can be attached to the entire report or to the report heading or footing (this includes table headings/table footings, and sub-headings/sub-footings).

The syntax for linking from a graphic image is the same as for linking from a report component. The only difference is adding IMAGE=*image* to the StyleSheet declaration.

Note: You can only link to a report or procedure from an image when you are using HTML format.

Syntax: How to Specify Links From a Graphic Image

To specify a link from an image in a report or procedure use:

```
TYPE=type, [subtype], IMAGE=image,
   FOCEXEC=fex[(parameters ...)], [TARGET=frame,] $
```

To specify a link from an image in a URL use:

```
TYPE=type, [subtype], IMAGE=image,
URL=url[(parameters ...)], [TARGET=frame,] $
```

To specify a link from an image in a JavaScript function use:

```
TYPE=type, [subtype], IMAGE=image,
JAVASCRIPT=function[(parameters ...)],$
```

where:

`type`

Identifies the report component that the user selects to execute the link. The TYPE attribute and its value must appear at the beginning of the declaration. You can specify the following types of components:

`REPORT` enables you to drill down from a graphical image that is attached to the entire report.

`TABHEADING` or `TABFOOTING` enables you to drill down from a graphical image that is attached to a report heading or footing.

`HEADING` or `FOOTING` enables you to drill down from a graphical image that is attached to a page heading or footing.

`SUBHEAD` or `SUBFOOT` enables you to drill down from a graphical image that is attached to a subheading or subfooting.

Report components are described in *Identifying a Report Component in a Web Query StyleSheet* on page 57.

`subtype`

Are any additional attributes, such as COLUMN, LINE, or ITEM, that are needed to identify the report component that you are formatting. See *Identifying a Report Component in a Web Query StyleSheet* on page 57 for information on identifying report components.

`image`

Specifies the file name of a graphical image file. The image must exist as a separate graphic file in a format that your browser supports. Most browsers support GIF and JPEG file types.

You can specify a local image file, or identify an image elsewhere on the network using a URL. URLs can be absolute, such as, http://www.ibm.com/graphic.gif, or relative alias that can be identified to the application server or web server, such as, /webquery/ibi_html/ce_logo.gif.

Alternatively, you can specify an alphanumeric field in the report (either a BY sort field or a display field) whose value corresponds to the name of the image file. For information about using StyleSheets to incorporate and position graphical images in a report, see *Laying Out the Report Page* on page 123.

fex

Identifies the file name of the linked procedure to run when the user selects the report component. For details about linking to another procedure, see *Linking to Another Report* on page 398.

url

Identifies any valid URL, or the name of a report column enclosed in parentheses () whose value is a valid URL. For details about linking to an URL, see *Linking to a URL* on page 403.

function

Identifies the JavaScript function to run when the user selects the report component. For details about calling a JavaScript function, see *Linking to a JavaScript Function* on page 406.

parameters

Are values that are passed to the report, URL, or JavaScript function. You can pass one or more parameters. The entire string of parameters must be enclosed in parentheses (), and separated from each other by a blank space.

frame

Identifies the target frame in the webpage in which the output from the drill-down link is displayed. For details, see *Specifying a Target Frame* on page 414.

Note: You cannot specify a target frame if you are executing a JavaScript function. However, the JavaScript function itself can specify a target frame for its results.

Example: Specifying a Link From an Image

The following example illustrates how to link a report from an image.

Report Request

```
TABLE FILE EMPLOYEE
PRINT LAST_NAME BY EMP_ID
HEADING
"List Of Employees By Employee ID"
ON TABLE SET STYLE *
INCLUDE=IBFS:/stylesheet_location/custom_stylesheet.sty,$
END
```

***custom_stylesheet*.sty**

```
TYPE=REPORT, GRID=OFF, $
TYPE=REPORT, IMAGE=/qibm/userdata/qwebqry/apps/century_electronics/
   ce_logo.gif, $
TYPE=HEADING, STYLE=BOLD, $
```

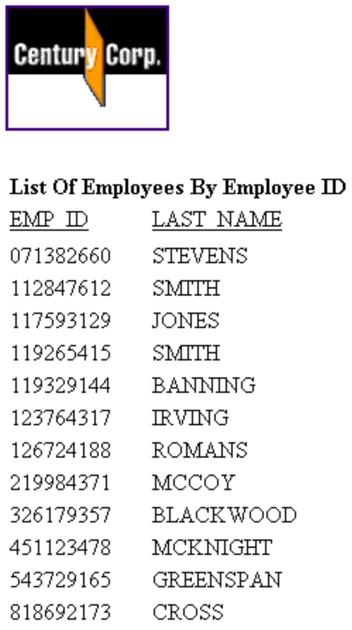
Drill-Down Report Request (IMAGE-D)

```
TABLE FILE EMPDATA
PRINT SALARY
BY DIV
WHERE DIV LE 'CORP';
ON TABLE SET STYLE *
INCLUDE=IBFS:/stylesheet_location/custom_stylesheet.sty,$
END
```

***custom_stylesheet*.sty**

```
TYPE=REPORT, GRID=OFF, $
```

The output for the main report is:



```
List Of Employees By Employee ID
EMP ID        LAST NAME
071382660     STEVENS
112847612     SMITH
117593129     JONES
119265415     SMITH
119329144     BANNING
123764317     IRVING
126724188     ROMANS
219984371     MCCOY
326179357     BLACKWOOD
451123478     MCKNIGHT
543729165     GREENSPAN
818692173     CROSS
```

When you click the graphic, the output is:

```
DIV            SALARY
CE               $62,500.00
                 $54,100.00
                 $25,400.00
                $115,000.00
                 $33,300.00
                 $25,000.00
                 $49,000.00
                 $40,900.00
                 $43,000.00
                 $45,000.00
CORP             $55,500.00
                 $83,000.00
                 $32,000.00
                 $62,500.00
                 $79,000.00
                 $35,200.00
                 $62,500.00
                 $26,400.00
```

Specifying a Target Frame

You can use frames to subdivide application HTML pages into separate scrollable sections. Frames enable users to explore various information items on a page by scrolling through a section, instead of linking to a separate page. When defining a link from a report component to a report procedure or URL, you can specify that the results of the drill-down link be displayed in a target frame on a webpage.

You can specify a target frame in a StyleSheet declaration using the TARGET attribute. You can use StyleSheets to specify that drill-down links from a report or chart are displayed in a target frame on the webpage displaying the report or chart. However, using StyleSheets to specify target frames adds extra HTML syntax to every HREF that is generated.

To use the TARGET attribute, you must create multiple frames on the webpage.

Note: You cannot specify a target frame if you are executing a JavaScript function. However, the JavaScript function itself can specify a target frame for its results.

Syntax: How to Specify a Target Frame

To specify a target frame in a report or procedure use:

```
TYPE=type, [subtype], FOCEXEC=fex[(parameters ...)], [TARGET=frame,] $
```

To specify a target frame for an URL use:

```
TYPE=type, [subtype], URL=url[(parameters ...)], [TARGET=frame,] $
```

where:

type

Identifies the report component that the user selects in the web browser to execute the link. The TYPE attribute and its value must appear at the beginning of the declaration.

subtype

Are any additional attributes, such as COLUMN, LINE, or ITEM, that are needed to identify the report component that you are formatting. For information on identifying report components, see *Identifying a Report Component in a Web Query StyleSheet* on page 57.

fex

Identifies the file name of the linked procedure to run when the user selects the report component. For details about linking to another procedure, see *Linking to Another Report* on page 398.

url

Identifies any valid URL, or the name of a report column enclosed in parentheses () whose value is a valid URL to which the link will jump. For details about linking to an URL, see *Linking to a URL* on page 403.

parameters

Are values being passed to the procedure or URL. You can pass one or more parameters. The entire string of values must be enclosed in parentheses (), and separated from each other by a blank space.

frame

Identifies the target frame in the webpage in which the output from the drill-down link (either a FOCEXEC or URL) is displayed.

If the name of the target frame contains embedded spaces, the name will be correctly interpreted without enclosing the name in quotation marks.

For example:

```
TYPE=DATA, COLUMN=N1,
FOCEXEC=MYREPORT, TARGET=MY FRAME, $
```

The name of the target frame is correctly interpreted to be MY FRAME.

You can also use the following standard HTML frame names: _BLANK, _SELF, _PARENT, _TOP.