Oracle® Reports

Building Reports

10*g* (9.0.4)

Part No. B10602-01

September 2003

ORACLE®

Oracle Reports Building Reports, 10*g* (9.0.4)

Part No. B10602-01

Primary Authors: Vanessa Wang, Ingrid Snedecor, and Frank Rovitto

Contributors: Christian Bauwens, Reena John, Shaun Lin, Rohit Marwaha, Darren McBurney, Paul Narth, Praveen Padmanabhan, Danny Richardson, Jim Safcik, Anjana Suparna Sriram, Toby Shimizu, Ravikumar Venkatesan, Philipp Weckerle, Stewart Wilson

Contents

2 Advanced Concepts

3 How To...

4 Visual Index

30 Building a Check Printing Report with Spelled-Out Cash Amounts

31 Building a Report Using a Pre-Printed Form

32 Building an Invoice Report

33 Building a Ranking Report

Send Us Your Comments

Oracle Reports Building Reports, 10*g* (9.0.4)

Part No. B10602-01

Oracle Corporation welcomes your comments and suggestions on the quality and usefulness of this document. Your input is an important part of the information used for revision.

- Did you find any errors?
- Is the information clearly presented?
- Do you need more information? If so, where?
- Are the examples correct? Do you need more examples?
- What features did you like most?

If you find any errors or have suggestions for improvement to this documentation, please send us your comments via the Oracle Reports discussion group forum (`http://otn.oracle.com/products/reports/`). If you have problems with the software, contact your local Oracle Support Services representative.

Preface

Reports Builder is the report-building component of Oracle Reports (part of the Oracle Developer Suite). The information and example reports in this manual are intended to help you learn about the extensive capabilities of Reports Builder, how to build different types of reports, and how to customize your reports to meet a wide variety of requirements.

Intended Audience

This manual is intended for anyone who uses Oracle Reports to build reports. The needs of both novice and advanced users are addressed. Following the step-by-step instructions, you can build the example reports from start to finish. Each report that you build will demonstrate how to use many of the powerful features in Reports Builder.

Documentation Accessibility

Our goal is to make Oracle products, services, and supporting documentation accessible, with good usability, to the disabled community. To that end, our documentation includes features that make information available to users of assistive technology. This documentation is available in HTML format, and contains markup to facilitate access by the disabled community. Standards will continue to evolve over time, and Oracle Corporation is actively engaged with other market-leading technology vendors to address technical obstacles so that our documentation can be accessible to all of our customers. For additional information, visit the Oracle Accessibility Program Web site at `http://www.oracle.com/accessibility/`.

Accessibility of Code Examples in Documentation JAWS, a Windows screen reader, may not always correctly read the code examples in this document. The conventions for writing code require that closing braces should appear on an otherwise empty line; however, JAWS may not always read a line of text that consists solely of a bracket or brace.

Accessibility of Links to External Web Sites in Documentation This documentation may contain links to Web sites of other companies or organizations that Oracle Corporation does not own or control. Oracle Corporation neither evaluates nor makes any representations regarding the accessibility of these Web sites.

Structure

This manual contains the following chapters and parts:

Chapter 1, "Basic Concepts"

This chapter describes basic Oracle Reports concepts.

Chapter 2, "Advanced Concepts"

This chapter describes advanced Oracle Reports concepts.

Chapter 3, "How To..."

This chapter provides procedures for using Reports Builder to create objects and design your reports.

Chapter 4, "Visual Index"

This chapter contains a visual index of all the example reports described in this manual.

The remainder of the chapters describe, step-by-step, how to build the example reports described in each chapter. Refer to the Visual Index for images of the individual report output.

Part 1, "Building Basic Reports"

- Chapter 5, "Building a Tabular Report"
- Chapter 6, "Building a Mailing Label Report"
- Chapter 7, "Building a Form Letter Report"

- Chapter 30, "Building a Check Printing Report with Spelled-Out Cash Amounts"
- Chapter 31, "Building a Report Using a Pre-Printed Form"
- Chapter 32, "Building an Invoice Report"
- Chapter 33, "Building a Ranking Report"
- Chapter 34, "Building a Paper Report with a Simple Table of Contents and Index"
- Chapter 35, "Building a Paper Report with a Multilevel Table of Contents"
- Chapter 36, "Bursting and Distributing a Report"

Part 6, "Building Reports with PL/SQL and Java"

- Chapter 37, "Building a PL/SQL Report"
- Chapter 38, "Building a Paper Report with Ref Cursors"
- Chapter 39, "Building a Simple Parameter Form for a JSP-based Web Report"
- Chapter 40, "Building a Report with a Barcode"

Part 7, "Building Reports with Pluggable Data Sources"

- Chapter 41, "Building a Report with an XML Pluggable Data Source"
- Chapter 42, "Building a Report with a Text Pluggable Data Source"
- Chapter 43, "Building a Report Using Oracle Express Data"

Appendix A, "Tool Palette and Toolbar Reference"

This appendix describes the tools in the Reports Builder interface.

"Glossary"

Provides definitions for terminology used in this manual.

Accessing the example reports

The reports you will build are also available for viewing on the Oracle Technology Network (`http://otn.oracle.com/products/reports/`) in *Getting Started with Oracle Reports*. Once you access the *Getting Started with Oracle Reports* Web site, click **Index**, then browse the **Examples** list. All of the latest example files available

for Oracle Reports are accessible from this list. These example reports are documented in this manual.

From the same Web site, you can download all the example files for this manual, including text files that contain the code you will enter for each report. These files are located in a single zip file, called `buildingreports_examples.zip`. This zip file contains sub-folders whose names correspond to the name of the chapter. Each of the sub-folders contains all the supporting files (images, code, and so on) necessary for building the individual reports.

Accessing the data sources

The data sources used in this manual are:

- SCOTT schema
- sample Human Resources and Order Entry schemas
- SUMMIT schema

The first two schemas are available with the Oracle9*i* database. You can contact your database administrator for more information on where to find those schemas. You can download the SQL scripts to install the SUMMIT schema from the *Getting Started with Oracle Reports* Web site (`http://otn.oracle.com/products/reports/)`, then click ***Getting Started with Oracle Reports***). Click **Index**, then navigate to **Tools and Utilities**.

Other resources

You can find more information about using Oracle Reports on the Oracle Technology Network, the *Reports Builder Online Help*, and *Oracle Application Server Reports Services Publishing Reports to the Web*. The latest information is available on the Oracle Technology Network (`http://otn.oracle.com/products/reports/`), under ***Getting Started with Oracle Reports***.

The Oracle Technology Network

The Oracle Technology Network is located at `http://otn.oracle.com`. You can access general information about the product, as well as delve into more detail by navigating to ***Getting Started with Oracle Reports***. The *Getting Started with Oracle Reports* Web site provides you with up-to-date example reports, as well as the latest white papers and demonstrations. Here, you can access the latest version of the

Building Reports and *Oracle Application Server Reports Services Publishing Reports to the Web* manuals.

Oracle Reports Tutorial

The tutorial describes how to build a simple JSP-based Web report with a graph, and how to generate a paper report from the same data model. You can access this manual via the ***Getting Started with Oracle Reports*** Web site. Simply go to the Web site, click **Index**, and navigate to the *Oracle Reports Tutorial*.

Reports Builder Online Help

You can access the *Reports Builder Online Help* by clicking **Help** in any dialog box, or by choosing **Help > Help Contents**.

Oracle Application Server Reports Services Publishing Reports to the Web

You can access this manual on the Oracle Application Server product CD, as well as on the Oracle Technology Network. Simply go to the Web site, click **Index**, and navigate to the *Oracle Application Server Reports Services Publishing Reports to the Web*. This manual describes how to set up Oracle Application Server Reports Services to perform the tasks you require. If you are setting up your Reports Server for distribution, for example, you should refer to this manual for further information.

1

Basic Concepts

This chapter introduces the fundamental concepts of the Reports Builder component of Oracle Reports. Each topic in this chapter is also included in the **Basic Concepts** section of *Reports Builder online help* (see Section 3.1.1, "Using the online help").

Topics are grouped into the following sections:

- Section 1.1, "Reports Builder"
- Section 1.2, "Reports"
- Section 1.3, "Report Styles"
- Section 1.4, "Wizards"
- Section 1.5, "The Object Navigator"
- Section 1.6, "The Report Editor"
- Section 1.7, "Data Model Objects"
- Section 1.8, "Layout Objects"
- Section 1.9, "Parameter Form Objects"
- Section 1.10, "The Property Inspector"
- Section 1.11, "Runtime Views"
- Section 1.12, "Executables"

1.1 Reports Builder

The topics in this section discuss the features and functionality in Reports Builder.

1.1.1 About Reports Builder

Reports Builder is the report-building component of Oracle Reports Developer (a component of the Oracle Developer Suite), a powerful enterprise reporting tool that enables you to rapidly develop and deploy sophisticated Web and paper reports against any data source (including an Oracle database, JDBC, XML, text files, and Oracle OLAP). Leveraging the latest J2EE technologies such as JSP and XML, you can publish your reports in a variety of formats (including HTML, XML, PDF, delimited text, Postscript, PCL, and RTF) to any destination (including e-mail, Web browser, OracleAS Portal, and file system) in a scalable, efficient manner.

Recognizing the differences between Web publishing and paper publishing, Reports Builder provides the power to develop high quality output for the Web and e-business requirements, as well as high-fidelity printed reports. Reports Builder includes:

- user-friendly wizards that guide you through the report design process
- pluggable data sources (PDSs), such as JDBC and XML, that provide access to data from any source for your reports
- a query builder with a graphical representation of the SQL statement to obtain report data
- default report templates and layout styles that can be customized if needed
- a live editor that allows you to modify paper report layouts in WYSIWYG mode
- the ability to add dynamic report output to an HTML page by embedding custom JSP tags within an HTML document
- an integrated graph builder to graphically represent report data
- the ability to generate code to customize how reports will run
- tools that dynamically generate Web pages based on your data
- standard report output formats such as HTML, HTMLCSS, XML, PDF, PCL, PostScript, and ASCII

- client-side parameter validation using JavaScript
- the ability to execute dynamic SQL statements within PL/SQL procedures
- support for Oracle database objects
- event-based reporting (report execution based on database events)
- seamless integration of Oracle Reports Developer with OracleAS Portal for administering report security and publishing report output to portlets

1.1.2 About this release

Oracle Reports focuses heavily on Web publishing, moving more fully into its role as a universal publishing solution. In prior releases, Reports Builder's Web feature simply displayed paper reports (that is, multiple pages) in HTML or PDF. This moves corporate data onto the Web, but also results in large and somewhat inflexible HTML pages.

One of the major new features of Oracle Reports is the incorporation of JavaServer pages (JSPs). Through the use of JSP technology, Oracle Reports embeds report data directly into Web pages.

For a detailed summary of new features in this release, as well as deprecated, obsolete, and changed functionality and components, see the topic "About this release" in the **Welcome** section of the *Reports Builder online help*.

1.2 Reports

The topics in this section discuss basic concepts of reports; for more advanced concepts, see Section 2.1, "Reports".

- About reports
- About Web reports

1.2.1 About reports

A report consists of objects that collectively define the report:

- data model objects (queries, groups, columns, links, user parameters)
- layout objects (repeating frames, frames, fields, boilerplate, anchors)
- parameter form objects (parameters, fields, boilerplate)
- PL/SQL objects (program units, triggers)
- references to external PL/SQL libraries, if any
- code shown in the Web Source view (for JSP-based Web reports)

Using the Property Inspector, you define report properties. The document taxonomy (classification) properties (Title, Author, Subject, and Keywords) assist in cataloguing and searching a report document.

When you first start Reports Builder, you can choose to open an existing report, create a new report using the Report Wizard, or create a new report manually.

1.2.1.1 Creating a new report using the Report Wizard

Using the Report Wizard, you can quickly and easily accomplish the steps to build a report for both Web and paper layouts:

1. Create a new report definition.
2. Define the data model (choose the data, data relationships, and calculations you will use to produce the report output).
3. Specify a layout. You can use a default, customizing it if desired, or create your own. Reports Builder provides the default layout styles described in Section 1.3, "Report Styles".
4. Then, you can modify your report using the different views of the Report Editor.

1.2.1.2 Creating a new report manually

If you choose to create a new report manually, Reports Builder creates a new default report definition for you. The first window you see is called the Object Navigator. This window displays a comprehensive list of report objects. Initially, it shows all objects that Reports Builder has created for you, as part of the report definition. As you define your report, the Object Navigator provides a central location to access and modify all objects in your report(s), including attached libraries and program units. You can change the Object Navigator view to list object hierarchically or by object type.

See also

Section 3.5.1, "Creating a report"

1.2.2 About Web reports

With a focus on Web publishing, Oracle Reports has moved more fully into its role as a universal publishing solution. In prior releases, Reports Builder's Web feature simply displayed paper reports (that is, multiple pages) in HTML or PDF. This moves corporate data onto the Web, but also results in large and somewhat inflexible HTML pages.

While all the prior Web report functionality remains (that is, hyperlinks, bookmarks, hyperlink destinations, before and after escapes, and so on), Oracle Reports uses JavaServer Pages (JSPs) as the underlying technology to enable you to enhance Web pages with information retrieved using Reports Builder. This introduces the Web Source view of a report, and allows you to have both JSP-based and paper-based definitions in a single report. In other words, you can either publish your paper reports to the Web or take more advantage of Web features using the Web Source view. For example, you can create a report that has a paper PDF version and a JSP-based Web version; what you choose depends on your needs and whether you are able to produce the desired results more easily in the Web Source view or in the Paper Design view.

Oracle Reports also includes servlet technology. Servlets provide a Java-based alternative to CGI programs. Servlets provide a platform-independent method for building Web-based applications, without the performance limitations of CGI programs.

You can create a Web report in any of the following ways:

- When you create a report using the Report Wizard, you can select on the first page of the wizard whether the report is both Web and paper, Web only, or paper only. The data model is added to both Web and paper reports, providing the data to be used to build the report. The layout for both Web and paper reports defaults in the Paper Design view. You can view the source for the Web report in the Web Source view.
- Open an existing HTML document (Web page) and imbed a report in your Web page using the Report Block Wizard. This provides tremendous flexibility in creating reports that meet the demands of completely integrating multiple sources of information within a single Web page. See and Section 3.6.4, "Adding a report block to a Web page".
- Display the Web Source view (see Section 3.6.2, "Viewing the source code for a Web report") and manually insert the Oracle Reports custom JSP tags. See the topic "Custom JSP tags for Oracle Reports" in the **Reference** section of the *Reports Builder online help*.
- Insert an existing report into an existing Web page, by displaying the Web Source view for both the report and the Web page, then copying and pasting the report block into the desired position in the Web page.
- Use the functionality available since Oracle Reports 6*i* to add HTML and hyperlinks to an existing paper-based report. See Section 3.6.7, "Adding Web links to paper-based reports".

To preview your report output in a Web browser, use **Program > Run Web Layout** in the Paper Design view to run the Web Source. This enables you to immediately see the effect of your changes on the output.

Note: If Netscape 7.0 is your default browser, the browser may not display. You can work around this bug by making a copy of the Netscape 7.0 executable, naming it `netscape.exe`; with this name, the browser will display as expected.

See also

Section 2.2, "Web Reports"

Section 1.9.4, "About Parameter Forms for Web reports"

Section 2.8.8, "About HTML output"

Section 2.8.9, "About PDF output"

Section 3.6.1, "Creating a Web report"

Section 3.7.15.4, "Displaying report output in your Web browser"

Section 3.7.16.3, "Printing a report from your Web browser"

1.3 Report Styles

The topics in this section describe the built-in report styles in Reports Builder.

- About tabular reports
- About group above reports
- About group left reports
- About form-like reports
- About form letter reports
- About mailing label reports
- About matrix reports

1.3.1 About tabular reports

A tabular report is the most basic type of report. Each column corresponds to a column selected from the database.

See also

Section 3.5.1, "Creating a report"

1.3.2 About group above reports

A group above report contains multiple groups in its data model. It is a "master/detail" report, where there may be a lot of information in the master group. For every master group, the related values of the detail group(s) are fetched from the database and are displayed below the master information.

See also

Section 3.5.1, "Creating a report"

Section 1.7.2, "About groups"

Section 3.8.4, "Creating a break group"

1.3.3 About group left reports

A group left report also contains multiple groups in its data model, dividing the rows of a table based on a common value in one of the columns. Use this type of report to restrict a column from repeating the same value several times while values

of related columns change. The data model for group above and group left reports is the same, but the layouts differ; group above reports display the master information at the top while group left reports display break columns to the side.

See also

Section 3.5.1, "Creating a report"

Section 1.7.2, "About groups"

Section 3.8.4, "Creating a break group"

1.3.4 About form-like reports

A form-like report displays one record per page, displaying field values to the right of field labels.

See also

Section 3.5.1, "Creating a report"

1.3.5 About form letter reports

A form letter report contains database values embedded in boilerplate text (any text that you enter or import into a Report Editor.)

See also

Section 3.5.1, "Creating a report"

1.3.6 About mailing label reports

A mailing label report prints mailing labels in multiple columns on each page. Using the Report Wizard, you can specify the format for your mailing labels.

See also

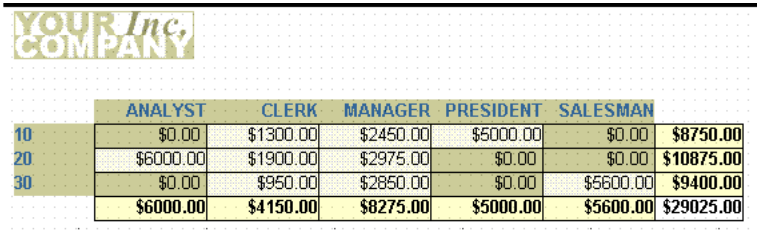
Section 3.5.1, "Creating a report"

1.3.7 About matrix reports

A matrix (cross-product) report is a cross-tabulation of four groups of data:

- One group of data is displayed across the page.
- One group of data is displayed down the page.
- One group of data is the cross-product, which determines all possible locations where the across and down data relate and places a cell in those locations.
- One group of data is displayed as the "filler" of the cells.

Figure 1–1 Example matrix report



	ANALYST	CLERK	MANAGER	PRESIDENT	SALESMAN	
10	$0.00	$1300.00	$2450.00	$5000.00	$0.00	**$8750.00**
20	$6000.00	$1900.00	$2975.00	$0.00	$0.00	**$10875.00**
30	$0.00	$950.00	$2850.00	$0.00	$5600.00	**$9400.00**
	$6000.00	**$4150.00**	**$8275.00**	**$5000.00**	**$5600.00**	**$29025.00**

Thus, to create a matrix report, you need at least four groups in the data model: one group must be a cross-product group, two of the groups must be within the cross-product group to furnish the "labels," and at least one group must provide the information to fill the cells. The groups can belong to a single query or to multiple queries.

A distinguishing feature of matrix reports is that the number of columns is not known until the data is fetched from the database.

With Reports Builder, you can create many different matrix reports. The four general types of matrix reports are simple matrix, nested matrix, multi-query matrix with break, and matrix break, but you are not confined to these designs.

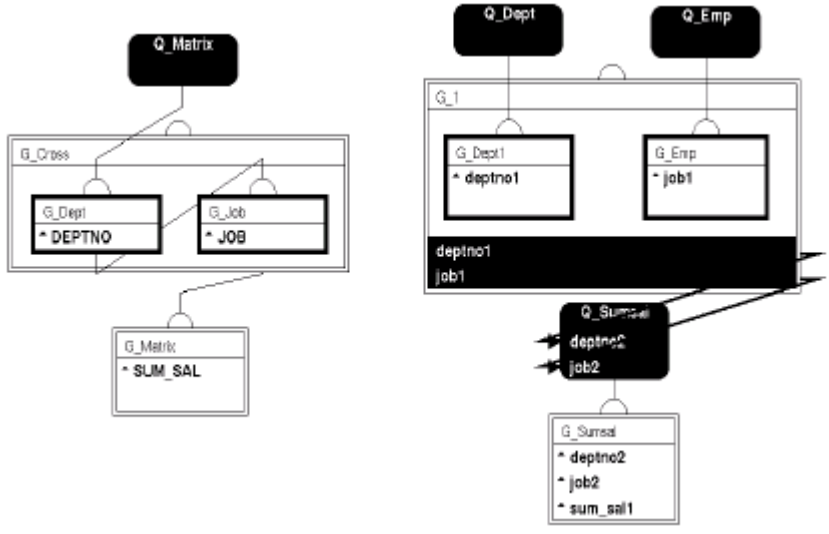
1.3.7.1 Matrix data model

In building your matrix data model, you should consider the following:

- number of queries
- group structure
- summary settings

1.3.7.1.1 Number of queries Although matrix reports always require at least four groups, they can be built with any number of queries. If you build a matrix report with only one query (usually the most efficient structure), you must create at least three groups in addition to the one created by default. If you build a matrix report with multiple (three or more) queries, you still need to create at least one additional group. The figure below illustrates the two types of query structures that you can use for your matrix data model.

Figure 1–2 Matrix data models



One advantage to a one-query data model is that the resulting report is generally more efficient than a report based on a multi-query data model.

Multi-query matrix. You may consider using a multi-query data model, as it often has simple queries and can be easier to maintain than the equivalent one-query data model. In addition, a multi-query data model is required for some types of matrices (e.g., some nested matrix reports).

1.3.7.1.2 Group structure Matrix reports are built with four or more groups:

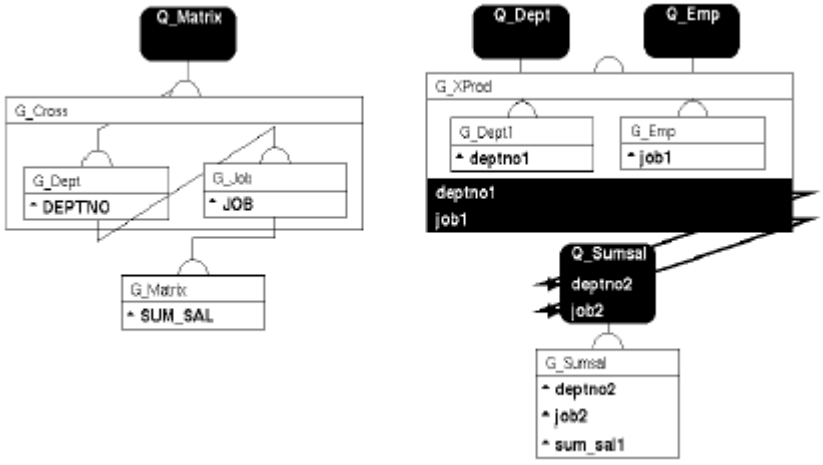
- **Two or more dimension groups.** Dimension groups are contained within the cross product group. In the layout, the information in at least one group goes across the page, and the information in at least one group goes down the page, forming a grid. The information in these groups is sometimes referred to as "matrix labels", as they appear to provide column and row labels for the matrix.
- **One or more cross product groups.** The cross product group represents all possible combinations of the values of the dimension groups. In the layout, the cross product group is represented by the intersection of the repeating frames for the across and down dimension groups. When the report is run, it expands, and each instance of data intersection becomes a separate cell. This concept is sometimes best understood graphically, as in the figure below. The rectangles are cells, and show where each department/job combination is valid.

Figure 1–3 Conceptual matrix

- **One cell, or "filler" group.** The cell group contains the actual information that is represented by the cells of the cross product. For each intersection of the values of the dimension groups (cell), the cell group contains zero, one, or multiple values. When the report is run, these values appear in the appropriate cells.

The following figure shows graphical representations of how groups are related for both single and multi-query data models.

Figure 1–4 Matrix data relationships



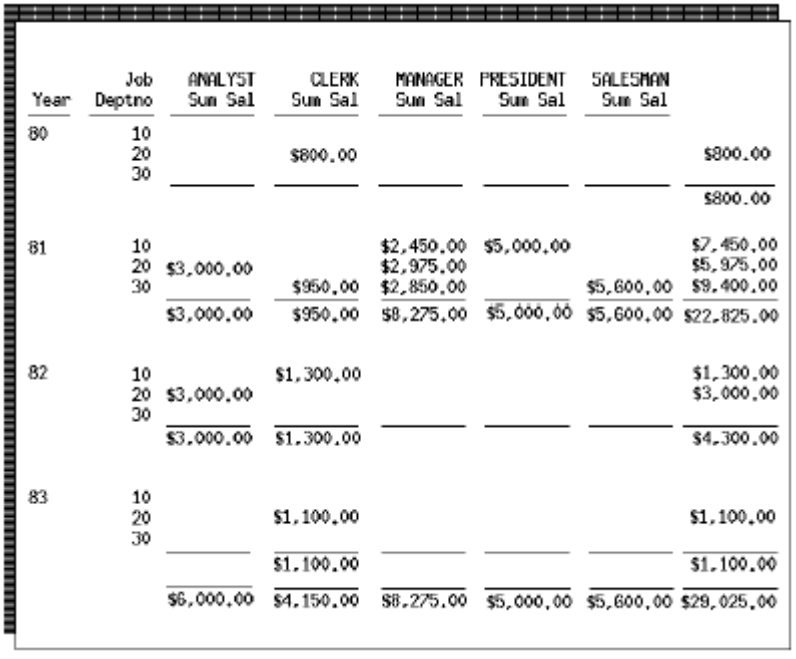
Notice that for each data model the cross product group is represented by the large rectangle, while the dimension groups are the smaller rectangles enclosed by it, and the cell group is shown outside of the cross product group.

1.3.7.1.3 Summary settings Creating a summary for a matrix requires more information than creating a summary for other kinds of reports. When you create summary columns for your matrix, you need to indicate the following:

- The frequency of the summary. The frequency specifies the dimension groups for which to compute the summary.
- The order in which to compute the summary. The order specifies how to traverse the matrix in calculating the summary (top to bottom or left to right).

In Reports Builder, you specify this information by setting the Product Order property for your summary. All summaries that are owned by the cross product group require that a Product Order be specified. Suppose that you have a matrix report that looks something like the one below.

Figure 1–5 Sample nested matrix report



Year	Job / Deptno	ANALYST Sum Sal	CLERK Sum Sal	MANAGER Sum Sal	PRESIDENT Sum Sal	SALESMAN Sum Sal	
80	10						
	20		$800.00				$800.00
	30						
							$800.00
81	10			$2,450.00	$5,000.00		$7,450.00
	20	$3,000.00		$2,975.00			$5,975.00
	30		$950.00	$2,850.00		$5,600.00	$9,400.00
		$3,000.00	$950.00	$8,275.00	$5,000.00	$5,600.00	$22,825.00
82	10		$1,300.00				$1,300.00
	20	$3,000.00					$3,000.00
	30						
		$3,000.00	$1,300.00				$4,300.00
83	10						
	20		$1,100.00				$1,100.00
	30						
			$1,100.00				$1,100.00
		$6,000.00	$4,150.00	$8,275.00	$5,000.00	$5,600.00	$29,025.00

This is a nested matrix report. Assume that group G_YEAR contains the YEAR column, G_DEPT contains the DEPTNO column, G_JOB contains the JOB column, and G_CROSS is the cross product group.

To create the summary of salaries by job that appears at the bottom of the matrix for each job, you create a summary column in G_CROSS with the following property settings:

Figure 1–6 Matrix summary settings

Product Order	*Reset At*
G_JOB	G_JOB

These settings specify that the summary should be calculated for each job in the G_JOB group and the summary should be reset to zero for each job in the G_JOB group.

To create the summary of salaries by year which appears at the right of the matrix directly underneath the boilerplate lines, you create a summary column in G_CROSS with the following property settings:

Figure 1–7 Matrix summary settings

Product Order	*Reset At*
G_YEAR	G_YEAR

These settings specify that the summary should be calculated for each year in the G_YEAR group and the summary should be reset to zero for each year in the G_YEAR group. To create the summary of salaries by year and department that appears at the right of the matrix for each department row of the matrix that contains a value, you create a summary column in G_CROSS with the following property settings:

Figure 1–8 Matrix summary settings

Product Order	*Reset At*
G_YEAR, G_DEPT	G_DEPT

These settings specify that the summary should be calculated for each record of G_DEPT within each record of G_YEAR. G_DEPT comes after G_YEAR in the Product

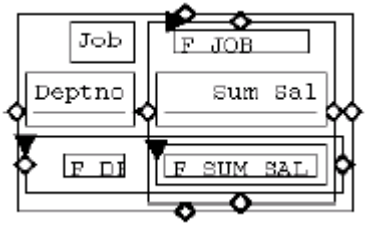
Order because, in this report, it changes more frequently than G_YEAR. If G_YEAR changed more frequently (i.e., years were listed for each department), it would make more sense to have a Product Order of G_DEPT, G_YEAR.

This summary is reset to zero for each record in the G_DEPT group. When you have multiple groups listed in Product Order and you want a non-running summary, the reset group should be the same as the last group in Product Order. When you have multiple groups listed in the Product Order and you want a running summary, the reset group should be a group other than the last one in the Product Order.

1.3.7.2 Matrix layout

The figure below shows the objects that make up a simple, two-dimensional matrix. All of these objects are necessary for a matrix, except for the boilerplate field labels.

Figure 1–9 Matrix Layout Model view



A matrix layout model must consist of the following layout objects:

- at least three repeating frames: one with the Print Direction property set to *Down*, one with the Print Direction property set to *Across*, and one for the cell in the matrix object
- several group, header, and footer (if summaries are included) frames
- a matrix object created for the cross product group, inside of which are the cells of the matrix (e.g., the rectangle in the figure above with R_SUMSAL and F_SUMSAL inside it)
- boilerplate for each column and row of values, as well as for summaries (e.g., Job and Deptno)

Note: Displaying the boilerplate is optional, but Reports Builder will generate it by default.

1.3.7.2.1 Matrix object The matrix object defines the intersection of at least two repeating frames. The repeating frames are the dimensions of the matrix and the matrix object contains the repeating frame that will hold the "filler" or values of the cell group. You need one matrix object for each pair of intersecting repeating frames in the layout. One of the repeating frames must have the Print Direction property set to *Down* and the other must have the Print Direction property set to *Across* in order to form a matrix.

For more details, see Section 2.3.7, "About matrix objects".

Usage notes

You can improve the performance of matrix reports with:

- fewer queries
- WHERE clauses

Note: Cross-product groups always cause "fetching ahead." The reason for this is that to cross-tabulate the data in a cross-product group, Reports Builder must first fetch all of the data.

See also

Section 2.3.7, "About matrix objects"

Section 2.1.7, "About nested matrix reports"

Section 2.1.8, "About matrix with group reports"

Section 2.4.2, "About layout defaulting"

Section 3.5.1, "Creating a report"

1.4 Wizards

Reports Builder includes wizards to help you quickly and easily define a report and add objects to it. The topics in this section describe each Reports Builder wizard.

- About the Report Wizard
- About the Report Block Wizard
- About the Data Wizard
- About the Graph Wizard

1.4.1 About the Report Wizard

The Report Wizard helps you to quickly and easily define a single-query report for both Web and paper layouts. Reports Builder uses what you specify on each page of the wizard to create a data model and layout for your report. After the wizard has created the report, you can modify it using the Report Editor views. See Section 1.6.1, "About the Report Editor".

Re-entrancy

To re-enter the Report Wizard for an existing report:

- Click the report, then choose **Tools > Report Wizard**.

Using the Report Wizard

For help on the fields on any tab page of the wizard, click **Help** at the bottom of the tab page.

1.4.2 About the Report Block Wizard

The Report Block Wizard is a version of the Report Wizard. You can open an existing HTML document (Web page) and imbed a report in your Web page using the Report Block Wizard to specify the layout of the data.

Access

To display the Report Block Wizard:

- In the Web Source view, choose **Insert > Report Block**.

Using the Report Block Wizard

For help on the fields on any tab page of the wizard, click **Help** at the bottom of the tab page.

1.4.3 About the Data Wizard

The Data Wizard helps you to quickly and easily define a query, break groups, and totals for a multi-query report. After the wizard has created the data model, you can:

- modify the data model in the Data Model view.
- use the Data Wizard to create additional queries.
- use the Report Wizard to default the layout.
- modify the layout in the Paper Layout view, including reordering or adding new layout sections.

Access

To display the Data Wizard:

- In the Data Model view, choose **Insert > Query**.

> **Note:** The Data Wizard does not support creating links between queries. To define parent/child query relationships, you can create a data link manually.

Re-entrancy

To re-enter the Data Wizard for an existing query, do either of the following:

- Right-click the query, and choose **Data Wizard**.
- Click the query, then choose **Edit > Settings**.

Using the Data Wizard

For help on the fields on any tab page of the wizard, click **Help** at the bottom of the tab page.

1.4.4 About the Graph Wizard

The Graph Wizard produces graphs that are automatically translated into JSP tags to enable you to add graphics to Web reports. Reports Builder uses what you specify on each page of the wizard to create the desired graph.

The Graph Wizard replaces Oracle6*i* Graphics, or Graphics Builder. There is no separate graphics tool; all of the graphing options and controls are now available in Reports Builder through the Graph Wizard. Using the Graph Wizard, you can generate more complex graphs with a larger variety of graph types. For backward compatibility, Oracle6*i* Graphics charts are supported in Oracle Reports 10*g*.

Primary Graph Types

For most graphing needs, the following primary graph types will provide the best means to represent data:

- bar graph
- line graph
- area graph
- pie graph
- combination graph

Secondary Graph Types

These are special usage or less common graphs that are associated with particular data types or ways to display unique cases of data:

- scatter graph
- bubble graph
- radar graph
- polar graph
- pareto graph
- stock graph
- 3-D graph

Reports built with previous versions of Oracle Reports containing Oracle6*i* Graphics charts will continue to run if the Oracle6*i* Graphics runtime is installed on the same machine in a separate `ORACLE_HOME`. For more information, see

"Displaying Oracle6*i* Graphics charts in Oracle Reports 10*g*" in the **Migration Issues** section of the *Reports Builder online help*.

Access

To display the Graph Wizard for paper reports:

- In the Paper Layout view, choose **Insert > Graph**.

To display the Graph Wizard for JSP-based Web reports:

- In the Web Source view, choose **Insert > Graph**.

Re-entrancy

To re-enter the Graph Wizard for an existing graph in paper reports:

1. In the Paper Layout view, click the graph.
2. To display the Graph Wizard in re-entrant mode, do either of the following:
   - Right-click and choose **Graph Wizard**.
   - Choose **Edit > Settings**.

To re-enter the Graph Wizard for an existing graph in JSP-based Web reports:

- In the Web Source view, place your cursor anywhere between the `<rw:graph>` and `</rw:graph>` tags, then choose **Edit > Settings**.

Using the Graph Wizard

For help on the fields on any tab page of the wizard, click **Help** at the bottom of the tab page.

1.5 The Object Navigator

The topics in this section discuss the Object Navigator in Reports Builder.

- About the Object Navigator
- About Object Navigator views

1.5.1 About the Object Navigator

The Object Navigator provides a hierarchical display of all objects in a report or template, including attached libraries and program units. Using the Object Navigator, you can:

- create reports, parameters, PL/SQL program units, and attached libraries.
- select and work with reports, queries, PL/SQL program units and libraries, data model objects, layout objects, and parameter form objects.
- display Report Editor views.
- expand and collapse nodes.
- search for objects.
- view objects by hierarchy or type.
- display properties.
- customize the Object Navigator.
- drag and drop PL/SQL program units.

See also

Section 3.2.4, "Setting preferences for the Object Navigator"

1.5.2 About Object Navigator views

The Object Navigator enables you to view objects by ownership hierarchy or by object type.

- **View > Change View > Ownership View** displays objects in a parent-child hierarchy. For example, a query's columns would appear underneath the query and a field would appear underneath the repeating frame that encloses it. This view can be a useful debugging tool because it enables you to easily see the relationships between objects.

- **View > Change View > Object Type View** displays objects by their type. For example, all queries would appear underneath the Queries heading and all database columns underneath the Database Columns heading. This view can be useful when you want to quickly find objects in the Object Navigator.

See also

Section 3.2.4, "Setting preferences for the Object Navigator"

1.6 The Report Editor

The topics in this section discuss the Report Editor and its different views of a report.

- About the Report Editor
- About the Data Model view
- About the Paper Layout view
- About the Paper Design view
- About the Paper Parameter Form view
- About the Web Source view
- About the tool palette and toolbar

1.6.1 About the Report Editor

The Report Editor is a work area in which you can manipulate the objects in your report directly or by changing properties in the Property Inspector. In the Report Editor window, you can navigate between different views of your report: the Data Model view, Paper Layout view, Paper Design view, Paper Parameter Form view, and Web Source view.

1.6.2 About the Data Model view

The Report Editor's Data Model view is a work area in which you create, define, and modify data model objects (queries, groups, columns, and data links) to be used in your report. In this view, objects and their property settings are represented symbolically to highlight their types and relationships. To create the query objects for your data model, you can use the Report Wizard, Data Wizard, or the Query tools in the tool palette.

Reports Builder uses the data model to determine what data to retrieve for the report. The data retrieved from the database may or may not appear in the report output.

Access

To display the Data Model view:

- Choose **View > Change View> Data Model**.
- Click the Data Model view button in the toolbar.

- In the Object Navigator, double-click the Data Model view icon next to the **Data Model** node.

1.6.3 About the Paper Layout view

The Report Editor's Paper Layout view is a work area in which you can modify the format of your paper report. The default format for your report is defined by the information you specify in the Report Wizard. You can modify the format by working with layout objects, such as frames, repeating frames, fields, boilerplate, anchors, and graph objects. In this view, objects and their property settings are represented symbolically to highlight their types and relationships.

The Paper Layout view is similar to the Paper Design view, in that it is a work area in which you modify the format of your report. However, the Paper Design view allows you to preview your report and manipulate the actual, or live, data at the same time. In the Paper Design view, you can customize reports interactively, meaning that you can see the results immediately as you make each change. The Paper Design view is displayed only after you run a report.

Access

To display the Paper Layout view:

- Choose **View > Change View > Paper Layout**.
- Click the Paper Layout view button in the toolbar.
- In the Object Navigator, double-click the Paper Layout view icon next to the **Paper Layout** node.

1.6.4 About the Paper Design view

The Report Editor's Paper Design view is a work area in which you can preview your paper report and manipulate the actual, or live, data at the same time. In the Paper Design view, you can customize reports interactively, meaning that you can see the results immediately as you make each change.

To edit your report, such as changing column size, the Paper Design view must be in Flex mode.

Access

The Paper Design view is displayed whenever you run a report. To run a report from the Object Navigator or any editor, click the Run Paper Layout button in the

toolbar or choose **Program > Run Paper Layout**. To run a report from the Report Wizard, click **Finish**.

You can also display the Paper Design view in these ways:

- click the Paper Design view button in the toolbar
- choose **View > Change View > Paper Design**

> **Note:** In the Object Navigator, there is no **Paper Design** node; this view is displayed only after running a report.

Usage notes

To speed the execution of your report, avoid "fetching ahead" when sending report output to the Paper Design view. The following items can result in fetching ahead when referenced before the data on which they rely:

- total number of pages/panels
- grand totals
- break columns that are formulas
- break columns that have Value if Null specified

Matrix (cross-product) groups also cause fetching ahead. In order to cross-tabulate the data, Reports Builder must first fetch all of the data.

It should be noted that while these items slow down the Paper Design view, they do not affect performance when writing to a file or some other destination.

> **Note:** A column can cause fetching ahead even if it is not displayed. For example, a grand total may not appear in the report output, but since it is in the report, fetching ahead may still occur when Reports Builder calculates it.

1.6.5 About the Paper Parameter Form view

The Report Editor's Paper Parameter Form view is a work area in which you define the format of the report's Runtime Parameter Form. To do this, you define and modify parameter form objects (fields and boilerplate). You can select pre-defined system parameters for your report using the Parameter Form Builder, or you can create your own.

When you run a report, Reports Builder uses the Paper Parameter Form view as a template for the Runtime Parameter Form. Fields and boilerplate appear in the Runtime Parameter Form exactly as they appear in the Paper Parameter Form view. If you do not define a Runtime Parameter Form in the Paper Parameter Form view, Reports Builder displays a default Runtime Parameter Form for you at runtime.

> **Note:** At runtime, the Runtime Parameter Form displays only when running the paper report layout. For JSP-based Web reports, the Runtime Parameter Form displays when you run your report within Reports Builder for debugging purposes, but will not display at runtime. For more information, see Section 1.9.4, "About Parameter Forms for Web reports".

Access

To display the Paper Parameter Form view:

- Choose **View > Change View > Paper Parameter Form**.
- Click the Paper Parameter Form view button in the toolbar.
- In the Object Navigator, double-click the Paper Parameter Form view icon next to the **Paper Parameter Form** node.

1.6.6 About the Web Source view

The Report Editor's Web Source view displays the source code for your Web report, including HTML tags and JSP tags. It is a Web page where you can view and add dynamic report blocks using the Report Block Wizard, and graphs using the Graph Wizard. Experienced Web developers can edit the Web source directly in this view using the Oracle Reports custom JSP tags.

Access

To display the Web Source view, do any of the following:

- Choose **View > Change View > Web Source**.
- Click the Web Source view button in the toolbar.
- In the Object Navigator, double-click the Web Source view icon next to the **Web Source** node.

See also

Section 1.2.2, "About Web reports"

Section 2.2.1, "About JavaServer Pages (JSPs) and servlets"

Section 3.6.3, "Adding report data to an existing Web page (HTML file)"

Section 3.6.4, "Adding a report block to a Web page"

Section 3.6.5, "Adding a graph to a Web report"

"Custom JSP tags for Oracle Reports" in the **Reference** section of the *Reports Builder online help*

1.6.7 About the tool palette and toolbar

The tool palette and toolbar contain tools used to manually create or manipulate objects in the Report Editor views (excluding the Web Source view). Each tool appears as an icon. Some tools, such as the Select tool, are common to tool palettes in all Report Editor views. Other tools are specific to the views in which they appear. In Reports Builder, run your cursor over a tool to display the hint text that identifies the tool.

Tool palette

The tool palette is positioned along the left side of the Report Editor views. Click a tool to activate it for a single operation, or double-click a tool to "lock" it for multiple operations. You can hide the tool palette by choosing **View > Tool Palette** to deselect it.

Toolbar

The toolbar is positioned along the top of the Report Editor views. To use the tools in the toolbar, click the desired tool to perform the action. All toolbar tools also have menu equivalents. If the tool is designed to perform an action on a group of objects, it appears grayed out until one or more objects are selected. Then clicking the tool performs the action on the selected object(s).

See also

Appendix A, "Tool Palette and Toolbar Reference"

1.7 Data Model Objects

The topics in this section discuss basic concepts of data model objects; for more advanced concepts, see Section 2.3, "Data Model Objects".

- About queries
- About groups
- About database columns
- About data links
- About Query Builder

1.7.1 About queries

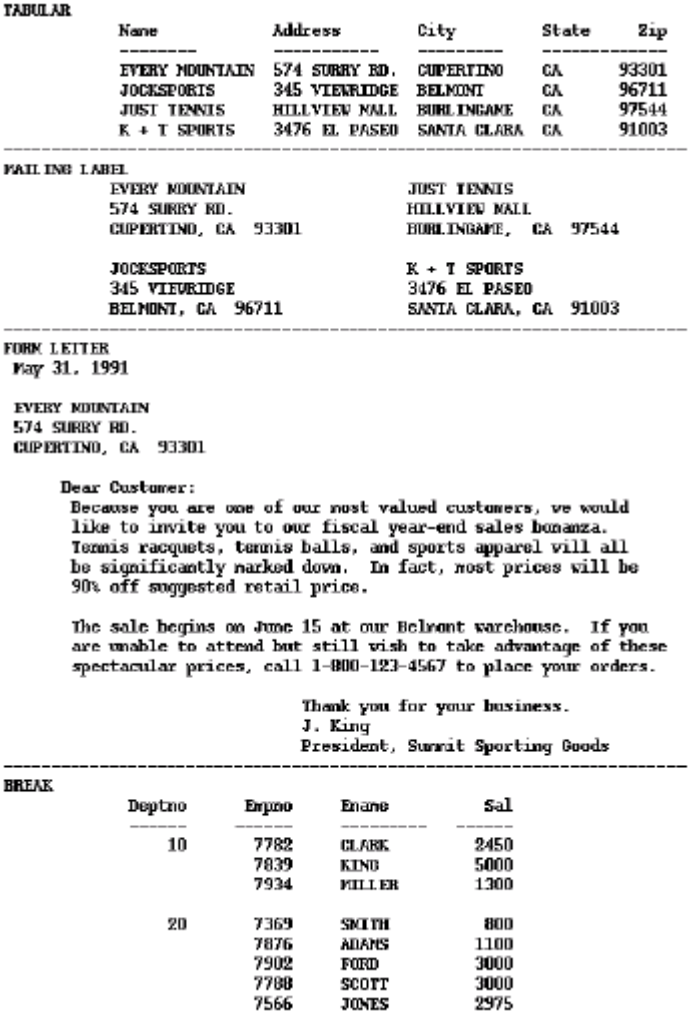
Queries provide the data for your report. You create a query using the Report Wizard, Data Wizard, or manually using the query tools in the Data Model tool palette. Queries can select data from any data source (Oracle, XML, JDBC, Text, Oracle Express, or your own data source that you can access via the pluggable data source (PDS) API). You can also use ref cursors to create queries.

Single-query reports

Reports built using one query are the simplest reports. The most popular formats for single-query reports are tabular, mailing label, form letter, and break (either group above or group left). In one report, you can display one query's data any number of times, even in different formats. You can query data without including it in the report output. This is useful for establishing relationships between multiple queries, performing calculations, and so on.

The figure below shows the data of one query formatted various ways.

Figure 1–10 Formatting the same data in different ways



Notice that the last report style (group left break) seems to use an entirely different query for its data than the first three. However, the data it displays was hidden in the other reports, and vice versa. With Reports Builder, you can query data without

including it in the report output. This is useful for establishing relationships between multiple queries, performing calculations, and so on.

Multi-query reports

A report may contain any number of queries. Multi-query reports are useful when you want to:

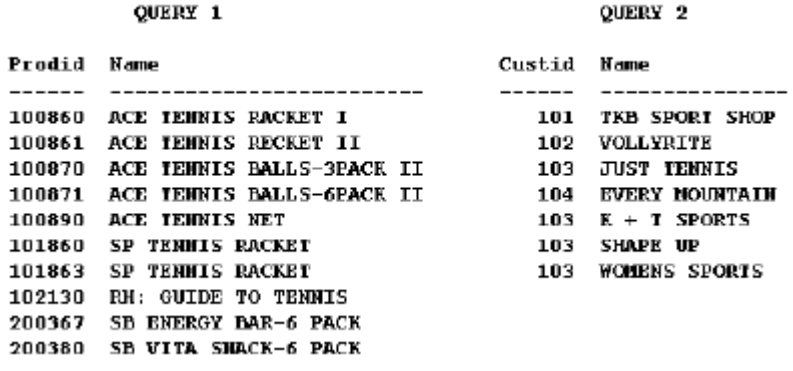
- produce multi-part unrelated query reports.
- produce multi-part related query reports.
- make your queries easier to maintain (often a report with a complex query containing embedded SELECT statements and joins can also be created with multiple simple queries that are linked. The latter are often easier for others to understand and maintain.)
- display the same data twice in a report with different sorting criteria.

If you create a report with multiple queries, you can either leave the queries unrelated, or establish a relationship between them using a data link.

Multi-part unrelated query reports

If you do not link the queries, you'll produce a multi-part unrelated query report (commonly called a master/master report). These types of reports display lists of unrelated data. For example, in the report below, one query selects products and another selects customers. Notice that there is no relationship between the products and customers.

Figure 1–11 Unrelated queries



QUERY 1

Prodid	Name
100860	ACE TENNIS RACKET I
100861	ACE TENNIS RECKET II
100870	ACE TENNIS BALLS-3PACK II
100871	ACE TENNIS BALLS-6PACK II
100890	ACE TENNIS NET
101860	SP TENNIS RACKET
101863	SP TENNIS RACKET
102130	RH: GUIDE TO TENNIS
200367	SB ENERGY BAR-6 PACK
200380	SB VITA SNACK-6 PACK

QUERY 2

Custid	Name
101	TKB SPORT SHOP
102	VOLLYRITE
103	JUST TENNIS
104	EVERY MOUNTAIN
103	K + T SPORTS
103	SHAPE UP
103	WOMENS SPORTS

Multi-part related query reports

In many reports, the data fetched for one part of the report is determined by the data fetched for another part. This is termed a ”master/detail,” or ”parent/child,” relationship, and is defined with a data link between two queries. When you run a master/detail report, each row of the master (or parent) query will cause the detail (or child) query to be executed, retrieving only matching rows.

Usage notes

- External queries are no longer supported.
- Reduce the number of queries in your report as much as possible. In general, the fewer queries you have, the faster your report will run. While multi-query data models are often easier to understand, single-query data models tend to execute more quickly.
- Processing done in the query is done on the server.
- To restrict the number of pages that a certain user can produce, insert a record into the PRODUCT_PROFILE table for that user.
- The only times you should use multi-query data models are:
  - when you're fetching many large columns from the parent and only a few small columns from the child.
  - when you're trying to do things that the query type, such as a SQL query, does not support directly (e.g., multi-way outer join).
  - when you have complex views (e.g., distributed queries or GROUP BY queries).
  - when you need but do not have or want to use a view.

Rationale

For a single-query report, Reports Builder opens only one cursor to fetch all of the master and detail records. For a two-query report, Reports Builder opens two cursors--one for each query--after appending the detail query's link to the WHERE clause of the detail query. Therefore, for each master record fetched in the master query, Reports Builder must rebind, execute, and fetch data from the detail query.

See also

Section 2.3.5, "About non-linkable queries"

Section 3.8.1, "Creating a query"

1.7.2 About groups

Groups are created to organize the columns in your report. Groups can do two things: separate a query's data into sets, and filter a query's data.

When you create a query, Reports Builder automatically creates a group that contains the columns selected by the query. You create additional groups to produce break levels in the report, either manually in the Data Model view or by using the Report Wizard to create a group above or group left report.

Create groups when you want to treat some columns differently than others. For example, you create groups to:

- produce subtotals (i.e., totals at a more granular level).
- create breaks or cross products in your report output.

With the exception of cross-product groups, all user-created groups are called *break groups*.

Break groups

You create break groups to produce subtotals, print columns in a different direction, create breaks, and so on. A break group suppresses duplicate values in sequential records. For example, Reports Builder can select the department number for each record; however, the duplicate department numbers are not printed.

Cross-product groups

You create cross-product groups to perform mathematical cross products, which are generally used to create matrix reports.

Group filters

Filters enable you to conditionally remove records selected by your queries. Groups can have two types of filters:

- Reports Builder packaged filters:
  - **First**, to display only the first n records for the group (e.g., the first 5 records)
  - **Last**, to display only the last n records for the group
- User-created filters, using PL/SQL.

See also

Section 2.3.6, "About links versus groups"

Section 2.6.9, "About group filters"

Section 1.3.2, "About group above reports"

Section 1.3.3, "About group left reports"

Section 2.1.8, "About matrix with group reports"

Section 3.5.1, "Creating a report"

Section 3.5.2, "Creating a multiquery group above report"

Section 3.8.4, "Creating a break group"

Section 3.8.5, "Creating a matrix (cross-product) group"

1.7.3 About database columns

A database column represents a column that is selected by the query, containing the data values for a report. For each column that you select in your query, Reports Builder automatically creates a column in the data model of your report. If you want to perform summaries and computations on database column values, you can create new columns manually in the Data Model view (for summary and formula columns) or by using the Report Wizard (for summary columns). You can also reassign one or more columns to a group or groups you've created.

In addition to the traditional column types (e.g., date, number, character), Reports Builder also supports graphic columns, which are columns whose values can be:

- **graphics stored directly in the database.** Such a column is usually of data type LONG or LONG RAW and contains a graphic. In this case, specify the graphic's format in the column's properties.
- **file names.** The values of such a column are pointers to files stored in the operating system. In this case, specify in the column's properties that its values are file contents (Read from File) and the format of the files. Many graphics formats are supported, including BMP, CALS, CGM, GIF, JFIF, PCD, PCX, PICT, RAS, TIFF, etc.)

See also

Section 3.8.1, "Creating a query"

Section 3.9.8.7, "Selecting an image from the database"

Section 2.3.4, "About referencing columns and parameters"

Section 2.3.1, "About summary columns"

Section 2.3.2, "About formula columns"

Section 2.3.3, "About placeholder columns"

1.7.4 About data links

A data link (or parent-child relationship) relates the results of multiple queries. A data link can establish these relationships:

- between one query's column and another query's column.
- between one query's group and another query's group (this is useful when you want the child query to know about its parent's data).

A data link causes the child query to be executed once for each instance of its parent group. The child query is executed with the values of the primary key used by the parent.

When a report with a data link is executed, the data link is converted into a SQL clause (as specified in the link's Property Inspector) and appended to the child query *if* Reports Builder is able to parse the query. **Important note**: If the query cannot be parsed, the query is not modified (for example, Reports Builder has no way to modify how the data is fetched when the query is defined for a pluggable data source).

Although links are commonly equijoins (e.g., WHERE DEPTNO =DEPTNO), you can create links with various SQL clauses (i.e., WHERE, HAVING, or START WITH) and conditions. If your database has database constraints, you can create a data link that derives its SQL clause and condition from the constraints. You can view the SELECT statements for the individual parent and child queries in Reports Builder, but can not view the SELECT statement that includes the clause created by the data link you define.

Example

For the report below, the following link was defined:

Figure 1–12 Sample data link

Parent Group	*SQL Clause*	*Parent Column(s)*	*Condition*	*Child Column(s)*
G_ORD	WHERE	ORDID	=	ORDID

The default group of the master query (containing the columns "Order ID" and "Customer") is the parent group; the detail query (the query to which "Item", "Product" and "Amount" belong) is the child query.

Figure 1–13 Sample data link output

MASTER QUERY

Order ID	Customer
613	108
615	107

DETAIL QUERY

Item	Product	Amount
1	100871	560.00
2	101860	4800.00
3	200380	600.00
4	200376	440.00

Item	Product	Amount
1	100861	180.00
2	100870	280.00
3	100871	250.00

Note: Reports Builder does not support data links between queries that contain column objects. If you attempt to create such a link, a message dialog box displays, which enables you to choose whether to create a group-to-group query instead (using the parent groups), or to cancel the operation. If you want to create a link between any type of column and a column object, you can manually modify the SQL statement in the child query, adding in the appropriate WHERE clause based on columns in the master query.

See also

Section 2.3.6, "About links versus groups"

Section 2.3.5, "About non-linkable queries"

Section 3.8.6, "Creating a data link"

1.7.5 About Query Builder

Query Builder is an easy-to-use data access tool designed for analysts, managers, and other business professionals. It provides a logical and intuitive means to access information from your organization's databases for analysis and reporting.

Query Builder is designed for professionals who do not have a computer programming or database background. Because of its powerful query features and its support of Structured Query Language (SQL) statements, experienced database users and programmers will find that Query Builder serves many of their needs as well.

In Query Builder's graphical Query window, you can specify a request for data from your organization's database(s). The request for data is called a query.

You can use Query Builder to define almost any query that you would build using a SQL SELECT statement. Query Builder automatically generates the appropriate SELECT FROM [table.column] clause based on columns displayed in the Query Builder workspace.

See also

Section 3.8.3, "Using Query Builder"

1.8 Layout Objects

The topics in this section discuss basic concepts of layout objects; for more advanced concepts, see Section 2.4, "Layout Objects".

- About frames
- About repeating frames
- About frame and repeating frame sizing
- About fields
- About boilerplate objects

1.8.1 About frames

Frames surround other objects and protect them from being overwritten or pushed by other objects. For example, a frame might be used to surround all objects owned by a group, to surround column headings, or to surround summaries.

When you default the layout for a report, Reports Builder creates frames around report objects as needed; you can also create a frame manually in the Paper Layout view. See Section 1.6.3, "About the Paper Layout view".

Create frames when you want to:

- group together objects to ensure they maintain their relative positions during printing.
- delineate sections in your report.

  Example: You want the top of each page to have a tabular format, but the bottom of each page to have a matrix format.
- protect other objects from being overwritten.

  Example: A summary is centered under two repeating frames. Defaulting rules state that the summary must remain at a fixed distance from only the first object that can overwrite it; therefore, the summary is in danger of being overwritten by the second repeating frame. Enclosing both repeating frames with a frame will force the summary to maintain a fixed distance from both of them, and it will not be overwritten.
- prevent an object from printing until other objects finish printing.

  Example: A summary is centered under two repeating frames. According to defaulting rules, the summary will print as soon as the first repeating frame

finishes printing. Create a frame around the two repeating frames to ensure the summary prints after both repeating frames have finished printing.

See also

Section 3.9.1.2, "Creating a frame or repeating frame"

1.8.2 About repeating frames

Repeating frames surround all of the fields that are created for a group's columns. The repeating frame prints (is fired) once for each record of the group.

When you default the layout for a report, Reports Builder generates one repeating frame for each group in the data model, and places one field inside it for each of the group's columns. Repeating frames can enclose any layout object, including other repeating frames. Nested repeating frames are typically used to produce master/detail and break reports. For each record of the outer repeating frame, Reports Builder will format all related records of the enclosed repeating frame.

You can also create a repeating frame manually in the Paper Layout view. See Section 1.6.3, "About the Paper Layout view".

See also

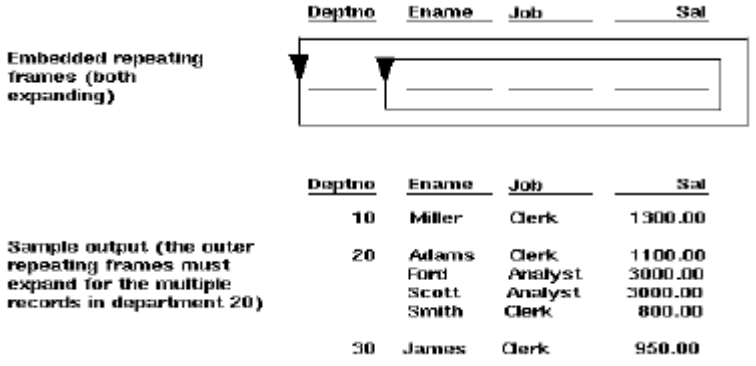
Section 3.9.1.2, "Creating a frame or repeating frame"

1.8.3 About frame and repeating frame sizing

For each object or record, a frame or repeating frame's size can be expandable, contractible, variable, or fixed (specified by the Horizontal Elasticity and Vertical Elasticity properties). For example, you can set the properties to specify that it should be fixed in size horizontally, but expand vertically if a record requires more space.

In the figure below, notice that there are four records for department 20, but only one record each for departments 10 and 30. The repeating frame has expanded vertically to accommodate department 20's additional records.

Figure 1–14 Repeating frame sizing



Frames or repeating frames that contract horizontally or vertically reduce in size when an object or record requires less space than the frame or repeating frame's initial size. A frame or repeating frame may also be variable in size: it expands or contracts based on the size of the value it displays. Also, if a frame or repeating frame is fixed in size and an object or record's data requires a larger field than it can contain, the remaining data will be pushed onto the following page(s) in the same x- and y-coordinates.

See also

Section 3.9.1.2, "Creating a frame or repeating frame"

1.8.4 About fields

Fields are placeholders for parameters, columns, and such values as the page number, current date, etc. If a parameter or column does not have an associated field, its values will not appear in the report output. A field is owned by the object surrounding it, which is the first enclosing object (either a frame or repeating frame).

When you default the layout for a report, Reports Builder generates one field for each column, and places each field inside of a repeating frame. You can also create a

field manually in the Paper Design view, Paper Layout view or Paper Parameter Form view.

See also

Section 1.9.2, "About Parameter Form fields"

Section 3.9.1.1, "Creating a field object"

Section 3.9.2.4, "Referencing a field in boilerplate text"

1.8.5 About boilerplate objects

A boilerplate object is any text, lines, or graphics that appear in a report every time it is run.

Reports Builder creates one boilerplate object for each label selected in the Report Wizard (it is named *B_columnname*). For example, if a column is named ENAME, a boilerplate object containing "Ename" is generated for the column. For some report types, Reports Builder also generates lines under the labels.

You can also create a boilerplate objects manually in the Paper Design view, Paper Layout view, or Paper Parameter Form view using any of the following tools in the tool palette:

- Arc tool
- Ellipse tool
- File Link
- Freehand tool
- Line tool
- Polygon tool
- Polyline tool
- Rectangle tool
- Rounded Rectangle tool
- Text tool

See Section 1.6.4, "About the Paper Design view", Section 1.6.3, "About the Paper Layout view", and Section 1.6.5, "About the Paper Parameter Form view".

Boilerplate from files

If you have text, graphics, or HTML in a file that you would like to display as boilerplate in your report, you can link to the file, as described in the following procedures:

- Section 3.9.2.5, "Linking a boilerplate text object to a file"
- Section 3.9.8.9, "Linking an image object to a file"
- Section 3.6.7.1.15, "Linking an HTML object to a file"

Linking to a file means that the contents of the file are pulled into the boilerplate object each time the report is run. In this way, you can ensure that your output includes the most recent changes to the file.

You can also link a boilerplate image object to a URL where the image is located. See Section 3.9.8.10, "Linking an image object to a URL".

See also

Section 1.9.3, "About Parameter Form boilerplate"

Section 3.9.2.1, "Creating a boilerplate object for text"

Section 3.9.2.2, "Creating a boilerplate object for text that displays every other page"

Section 3.9.2.4, "Referencing a field in boilerplate text"

Section 3.9.11.8, "Rotating a boilerplate object"

1.9 Parameter Form Objects

The topics in this section discuss basic concepts of Parameter Form objects; for more advanced concepts, see Section 2.5, "Parameter Form Objects".

- About parameters
- About Parameter Form fields
- About Parameter Form boilerplate
- About Parameter Forms for Web reports

1.9.1 About parameters

A parameter is a variable whose value can be set at runtime (e.g., from the Runtime Parameter Form or the command line). Parameters are especially useful for modifying SELECT statements and setting PL/SQL variables at runtime.

Reports Builder automatically creates a set of system parameters at runtime, but you can create your own as well. You can create parameters to replace either single literal values or entire expressions in any part of a query. You can reference parameters elsewhere in the report, such as in PL/SQL constructs providing conditional logic for the report.

Note: While you can delete or rename a user parameter, you cannot delete or rename a system parameter.

Parameter values can be specified in these ways:

- accepting the default parameter values (default values are set in the Parameter properties, and you can control whether the values are displayed at runtime on the Runtime Parameter Form)
- typing the parameter value(s) as arguments on the command line (where applicable)
- choosing from a list or entering the parameter value(s) in the Runtime Parameter Form

System parameters

Reports Builder is shipped with the following parameters. You can change their default values as described above.

Table 1–1 System parameter descriptions

System Parameter	Description
COPIES	Is the number of report copies that should be made when the report is printed.
CURRENCY	Is the symbol for the currency indicator (e.g., "$"). This system parameter is deprecated, and will be obsolete in a subsequent release of Reports Builder.
DECIMAL	Is the symbol for the decimal indicator (e.g., "."). This system parameter is deprecated, and will be obsolete in a subsequent release of Reports Builder.
DESFORMAT	Is either: ▪ the output format for the report (e.g., PDF, HTML, HTMLCSS, RTF, XML, or DELIMITED for bitmapped reports). ▪ the printer definition to use when formatting the report when DESTYPE=FILE and DESNAME=*filename*. If MODE=BITMAP, this is the name of the printer. If MODE=CHARACTER, this is the character mode printer definition file (.prt file).
DESNAME	Is the name of the output device (e.g., the file name, printer's name, mail userid).
DESTYPE	Is the type of device that will receive the report output (e.g., screen, file, printer, mail, sysout, cache, localfile, or preview (to format the report using screen fonts and size the fonts using the printer font metrics)).
MODE	Is whether the report should run in character mode or bitmap.
ORIENTATION	Is the print direction for the report (landscape, portrait, default).
PRINTJOB	Is whether the Print Job dialog box should appear before the report is run.
THOUSANDS	Is the symbol for the thousand's indicator (e.g., ","). This system parameter is deprecated, and will be obsolete in a subsequent release of Reports Builder.

User parameters

You can create a user parameter in the following ways:

- create a parameter in the Object Navigator
- use a bind parameter reference in a query, which causes Reports Builder to automatically create the parameter the first time it is referenced (see Section 2.3.4.1, "About bind references")

See also

Section 2.3.4, "About referencing columns and parameters"

Section 3.11.1, "Using a pre-defined system parameter"

Section 3.11.2, "Creating a user parameter"

Section 3.11.3, "Creating a list of values (LOV) for a parameter"

Section 3.11.4, "Validating a parameter value at runtime"

Section 3.11.6, "Selecting parameters to include in the Runtime Parameter Form"

Section 3.11.9, "Passing parameters to reports running in batch mode"

1.9.2 About Parameter Form fields

Fields in the Paper Parameter Form view (Section 1.6.5, "About the Paper Parameter Form view") act as placeholders for parameters. They define the formatting attributes for the parameters displayed in the Runtime Parameter Form. By default, one field is created for each parameter.

See also

Section 1.8.4, "About fields"

Section 3.11.5, "Creating a default Parameter Form"

Section 3.11.7, "Displaying the Parameter Form at runtime"

1.9.3 About Parameter Form boilerplate

Boilerplate in the Paper Parameter Form view refers to text and graphics that appear in the Runtime Parameter Form each time it is run; for example, a label denoting a particular parameter is boilerplate text created by Reports Builder. Lines or boxes that you create in the layout are also considered boilerplate, as well as any added text.

Boilerplate enables you to customize the Runtime Parameter Form. By default, a boilerplate label is produced for each field that appears on the Runtime Parameter Form.

See also

Section 1.8.5, "About boilerplate objects"

Section 3.11.5, "Creating a default Parameter Form"

Section 3.11.7, "Displaying the Parameter Form at runtime"

1.9.4 About Parameter Forms for Web reports

You can design a Parameter Form for both Web and paper reports. However, at runtime, the Runtime Parameter Form displays only when running the paper report layout. For JSP-based Web reports, the Runtime Parameter Form displays when you run your report within Reports Builder for debugging purposes, but will not display at runtime.

Because you now have the flexibility to display your reports on any Web page, the report may be just one object on a Web page containing many other portlets and objects, and parameters for the report may be retrieved from sources other than the Parameter Form. For example, parameters might be provided by the Web page to all portlets on the page. Therefore, it does not make sense to display a Runtime Parameter Form for JSP-based Web reports before the report is formatted.

In the absence of the Runtime Parameter Form, you will need to use an alternate method to provide required parameters to a JSP-based Web report designed with a Parameter Form. For example:

- When you design the report, set all parameters to a default value.
- If you run your report using a URL, provide the parameters via the URL.
- Create an HTML form that your report calls to provide parameter values, either as static values, or as a list of values using the Oracle Reports custom JSP tags.
- If the report displays as a portlet in a Web page, you can pass the page level parameters to the report.
- Use the JSP tag <rw:reports id="*myReport*" parameters="*yourParameterList*">, where *yourParameterList* can be a Java variable. For example:

```
<% String myParameterList = "userid=scott/tiger&p_deptno+10"; %>
<rw:report id="myReport" parameters="<%= myParameterList %>">
```

For more information on creating a Parameter Form for a JSP-based Web report, see Chapter 39, "Building a Simple Parameter Form for a JSP-based Web Report".

Note: If you display your paper-based report on the Web, you can create an HTML Parameter Form by adding HTML header and footer tags using either:

- **the Property Inspector**: set the Before Form Value property and After Form Value property for the report
- **PL/SQL**: set format triggers using the `SRW.SET_BEFORE_FORM_HTML` and `SRW.SET_AFTER_FORM_HTML` procedures.

See also

Section 1.2.2, "About Web reports"

Section 1.6.5, "About the Paper Parameter Form view"

Section 1.11.1, "About the Runtime Parameter Form"

Section 2.5.1, "About Parameter Form HTML extensions"

1.10 The Property Inspector

The topics in this section discuss the Property Inspector in Reports Builder.

- About the Property Inspector
- About making multiple selections in the Property Inspector

1.10.1 About the Property Inspector

The Property Inspector is a windows that enables you to access the properties of the currently selected object(s) in the Object Navigator, Report Editor, and Template Editor. Every Reports Builder object (query, group, frame, parameter, etc.) has associated properties that can be viewed using the Property Inspector.

See Section 1.5.1, "About the Object Navigator", Section 1.6.1, "About the Report Editor", and Section 2.7.5, "About the Template Editor".

To get help on any property, click the property in the Property Inspector and press F1.

1.10.1.1 About making multiple selections in the Property Inspector

You can select multiple objects at the same time by using Shift+Click or Ctrl+Click in the navigators or the editors. When two or more objects are selected, a list of object names is displayed at the top of the Property Inspector.

The Intersection/Union button on the Property Inspector toolbar determines which properties are displayed in the property list when more than one object is selected. Toggling between Intersection and Union changes the list of properties displayed in the Property Inspector, but does not affect the setting of any property.

Table 1–2 Property Inspector Intersection/Union

Button	Description
Intersection	The default. Only properties common to all selected objects are displayed.
Union	All properties of every object selected are displayed.

Properties that are common to two or more objects in a multiple selection are listed only once, and the property setting is displayed as follows:

- If the property has the same setting for all of the selected objects that have it in common, the common setting is displayed.

- If the property is set differently for the objects that have it in common, the string ***** is displayed.

See also

Section 3.2.2, "Setting report properties"

Section 3.2.5, "Setting properties for an ASCII (character-mode) report"

Section 3.2.7, "Setting properties of multiple objects"

Section 3.2.8, "Comparing the properties of one object to another"

1.11 Runtime Views

The topic in this section discuss the views of a report shown at runtime.

- About the Runtime Parameter Form
- About the Previewer

1.11.1 About the Runtime Parameter Form

The Runtime Parameter Form is a dialog box that optionally displays at runtime in which you can override default parameter values (e.g., values that modify SELECT statements, route the report output to a specified device, etc.). You define the format of the Runtime Parameter Form in the Paper Parameter Form view. If you do not define a Runtime Parameter Form in the Paper Parameter Form view, Reports Builder displays a default Runtime Parameter Form for you at runtime.

Change the parameters as desired and then click the Run Paper Layout button in the toolbar to run the report.

Alternatively, you can click the Cancel Run button in the toolbar to cancel.

See also

Section 1.6.5, "About the Paper Parameter Form view"

Section 1.9.4, "About Parameter Forms for Web reports"

Section 3.11.7, "Displaying the Parameter Form at runtime"

1.11.2 About the Previewer

The Previewer displays on your screen how the printed version of your report will look. In the Previewer, you can scroll though a single page of report output, page through the entire report, and split the screen to view different sections of the same report concurrently. You can also perform the following actions:

Table 1–3 Print previewer actions

To...	Click in the toolbar...
Print the report	the Print button
Specify page setup settings	the Page Setup button
Open a new Previewer	the New Previewer button
Close the Previewer	the Close Previewer button
Zoom in	the Zoom In button
Zoom out	the Zoom Out button

A physical page (or panel) is the size of a page that will be output by your printer. A logical page is the size of one page of your actual report (it can be any number of physical pages wide or long). The Previewer displays the logical pages of your report output, one at a time.

Access

To display the Previewer:

- choose **File > Print Preview**.

Usage notes

To speed the execution of your report, avoid "fetching ahead" when sending report output to the Previewer or Paper Design view. The following items can result in fetching ahead when referenced before the data on which they rely:

- total number of pages/panels
- grand totals
- break columns that are formulas
- break columns that have Value if Null specified

Matrix (cross-product) groups also cause fetching ahead. In order to cross-tabulate the data, Reports Builder must first fetch all of the data.

It should be noted that while these items slow down the Previewer or Paper Design view, they do not affect performance when writing to a file or some other destination.

Note: A column can cause fetching ahead even if it is not displayed. For example, a grand total may not appear in the report output, but since it is in the report, fetching ahead may still occur when Reports Builder calculates it.

See also

Section 3.7.15.3, "Displaying report output in the Previewer"

Section 3.7.16.2, "Printing a report from the Previewer"

1.12 Executables

The Oracle Reports executables are:

Table 1–4 Executable descriptions

Executable	Description
`rwbuilder`	starts Reports Builder
`rwrun`	runs a report using the OracleAS Reports Services in-process server
`rwclient`	parses and transfers a command line to the specified (or default) Reports Server
`rwcgi`	translates and delivers information between either a Web server or a J2EE Container (for example, OC4J) and the Reports Server, enabling you to run a report dynamically from your Web browser (supported for backward compatibility with previous releases of Oracle Reports)
`rwserver`	invokes the Reports Server
`rwservlet`	translates and delivers information between either a Web server or a J2EE Container (for example, OC4J) and the Reports Server, enabling you to run a report dynamically from your Web browser
`rwconverter`	converts one or more report definitions or PL/SQL libraries from one storage format to another

All executables can be run from the command line. See the **Reference > Command Line** section of the *Reports Builder online help* for detailed information about the executables and command line keywords.

See also

Section 3.7.2, "Running a report from the command line"

Section 3.7.3, "Running a report using a command file"

2

Advanced Concepts

This chapter introduces the concepts for advanced users of Oracle Reports. Each topic in this chapter is also included in the **Advanced Concepts** section of *Reports Builder online help* (see Section 3.1.1, "Using the online help").

Topics are grouped into the following sections:

- Section 2.1, "Reports"
- Section 2.2, "Web Reports"
- Section 2.3, "Data Model Objects"
- Section 2.4, "Layout Objects"
- Section 2.5, "Parameter Form Objects"
- Section 2.6, "PL/SQL"
- Section 2.7, "Templates"
- Section 2.8, "Output Formats and Capabilities"
- Section 2.9, "Data Sources"
- Section 2.10, "Debugging Tools"

2.1 Reports

The topics in this section build on the basic concepts discussed in Section 1.2, "Reports".

- About report titles
- About report sectioning and sections
- About the report unit of measurement
- About the report dimensions
- About conditional formatting
- About nested matrix reports
- About matrix with group reports

2.1.1 About report titles

You can add a title to a report in either of the following ways:

- manually, by creating a boilerplate text object in the margin of the Paper Layout view.
- by typing in the **Title** field on the Style page of the Report Wizard.

When you use the Report Wizard to add a title and *do not* select a template for your report output, the title is inserted into the margin of the report with default attributes defined by Reports Builder. You can modify the attributes in the Paper Layout view.

When you use the Report Wizard to add a title, and *do* select a predefined template or a user-defined template file for your report output, Reports Builder looks for a boilerplate text object named B_OR$REPORT_TITLE defined for the selected template:

- If B_OR$REPORT_TITLE exists, the title is displayed using the attributes of this object, with the text you typed in the Report Wizard.
- If B_OR$REPORT_TITLE is not found, the title is displayed using the attributes defined by the Default properties (under the Title node in the Property Inspector) of the selected template.

Note: If you do not specify a title in the Report Wizard, the B_OR$REPORT_TTILE object is not copied to your report.

For layouts created using the Report Block Wizard, the title is inserted into the new layout as a group title rather than into the margin of the report. In this case, the attributes are set per the Default properties (under the Title node in the Property Inspector) of the selected template, and B_OR$REPORT_TITLE is ignored. If you do not select a template, the title uses the default attributes defined by Reports Builder.

See also

Section 3.5.6, "Adding a title to a report"

Section 3.12.5, "Formatting the report title in a template"

Section 2.7.1, "About templates"

2.1.2 About report sectioning and sections

Report sectioning enables you to define multiple layouts in the same report, each with a different target audience, output format, page layout, page size, or orientation. You can define up to three report sections, each with a body area and a margin area: the names of the sections are Header, Main, and Trailer. By default, a report is defined in the Main section. In the other sections, you can define different layouts, rather than creating multiple separate reports. For example, a single report can include an executive summary for senior management in one section and also a detailed breakdown for individual managers in another section. If you wish, you can use the margin and body of the Header and Trailer sections to create a Header and Trailer page for your reports.

In the Object Navigator, the report sections are exposed in the Object Navigator under the **Paper Layout** node as **Header Section**, **Main Section**, and **Trailer Section**.

For detailed information about section-level distribution, see Chapter 36, "Bursting and Distributing a Report". This chapter also covers distribution based on a repeating section, then e-mail those sections based on the distribution.xml.

Examples

Example 1

You can use sectioning and distribution to publish your report output in HTML, and also send a Postscript version to the printer.

Example 2

You can send an executive summary of the report to a senior management, and also mail detailed breakdowns to individual managers. In this example, a single report with two report sections needs to be created: a portrait-sized summary section and a landscape-sized detail Section. Associating the detail section with a data model group that lists the managers and then altering the destination on each instance of the data model group sends the output to the appropriate managers.

See also

Section 2.8.3, "About report distribution"

Section 3.10.1, "Displaying a section layout view"

Section 3.10.2, "Creating a default layout for a section"

2.1.3 About the report unit of measurement

A report can be defined using inches, centimeters, or points. The unit of measurement is independent of the device on which you build the report. As a result, you can design reports that will run on any platform on which Reports Builder runs. You can change a report's unit of measurement in these ways:

- Setting the Unit of Measurement property.
- Converting the report using rwconverter, specifying a different unit of measurement with the DUNIT keyword.
- Opening the report in a different environment. For example, if you open a character-mode report, Reports Builder will change the report's unit of measurement to the bit-mapped environment's default. If you then save the report, it will be saved with the new unit of measurement.

2.1.4 About the report dimensions

A report page can have any length and any width. Because printer pages may be smaller or larger than your paper report's "page," the concept of physical and logical pages is used. A physical page is the size of a page that is output by your printer. A logical page is the size of one page of your report; one logical page may be made up of multiple physical pages.

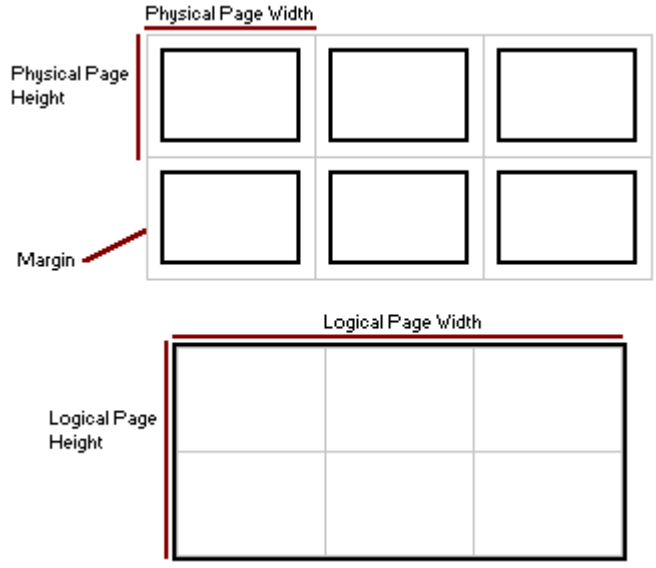
For each section (header, main, trailer) of a report:

- you specify the dimensions of the physical page (including the margin) using the Width property and Height property.

- you specify the dimensions of the logical page (report page) in physical pages (printer pages) using the Horizontal Panels per Page property (width) and the Vertical Panels per Page property (height). For example, a Horizontal Panels per Page size of 1 means that each logical page is one physical page wide, and a Vertical Panels per Page size of 2 means that each logical page is two physical pages in height.

In this example, one logical page is made up of six physical pages. The logical page is three physical pages wide and two physical pages high. Consequently, Horizontal Panels per Page size is 3 and Vertical Panels per Page size is 2. If you wanted the logical page to be two physical pages wide and three physical pages high, you would specify a Horizontal Panels per Page size of 2 and a Vertical Panels per Page size of 3.

Figure 2–1 Report dimensions



2.1.5 About fonts in reports

For detailed information about using and adding fonts in Oracle Reports, including font configuration files, font aliasing, troubleshooting font issues, and font types, refer to the chapter "Fonts in Oracle Reports" in the *Oracle Application Server Reports Services Publishing Reports to the Web* manual, available on the Oracle Technology Network Oracle Reports Documentation page (`http://otn.oracle.com/docs/products/reports/content.html`).

2.1.6 About conditional formatting

Using the Conditional Formatting and Format Exception dialog boxes, you can specify output formatting attributes (font and/or color) for a selected layout object based on conditions that exist. The conditions that you define are called format exceptions.

You can display the Conditional Formatting dialog box from the Paper Layout view or Paper Design view in any of the following ways:

- Double-click the object to display the Property Inspector. Under the **General Layout** node, click the Conditional Formatting value field (labeled**...**).
- Display the pop-up menu (right-click in Windows) for the object.
- Click the object, then choose **Format > Conditional Formatting**.

The Format Exception dialog box displays when you click **New** or **Edit** in the Conditional Formatting dialog box, and enables you to quickly and easily specify output formatting attributes for a selected layout object based on defined conditions. After you specify conditions and formatting for the current layout object in the Format Exception dialog box, the entire definition is exported to a PL/SQL format trigger. If a format trigger already exists for the layout object, the definition in the Format Exception dialog box overwrites the existing trigger code when you confirm the Reports Builder prompt.

You can edit the format trigger manually via the PL/SQL Editor; however, if you subsequently modify the definition using the Format Exception dialog box, Reports Builder displays a prompt to overwrite the existing format trigger.

See also

Section 3.9.1.5, "Applying conditional formatting to a layout object"

2.1.7 About nested matrix reports

A nested matrix report is a matrix report in which at least one parent/child relationship appears within the matrix grid.

A nested matrix report has more than two dimensions; therefore, it has multiple dimensions going across and/or down the page. For example, look at the report below (Figure 2–2). Notice that for each year there is a nested list of related departments. Also notice that the list of jobs (the across values) appears only once. Because the job values appear only once, a summary of each category of jobs can be made to line up with the values it summarizes.

For an example, see the example report in Chapter 26, "Building a Nested Matrix Report".

See also

Section 2.3.7, "About matrix objects"

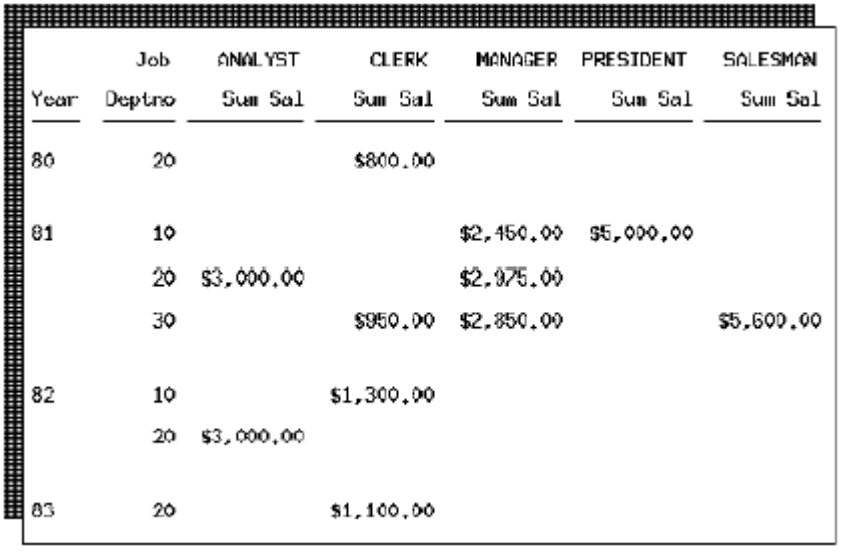
Section 1.3.7, "About matrix reports"

2.1.8 About matrix with group reports

A matrix with group report is a group above report with a separate matrix for each value of the master group. For example, for each year (master) in the report below there is a unique matrix that contains only that year's departments and jobs. This means that a summary of each job category may not line up with the values it summarizes because the position of each job category in the matrix may vary for each year.

A multi-query matrix with group report is similar to a nested matrix report in that it has more than two dimensions. For example, in the following report, notice that for each year there is a nested list of related departments.

Figure 2–2 Sample matrix with group and nested matrix report



Year	Job Deptno	ANALYST Sum Sal	CLERK Sum Sal	MANAGER Sum Sal	PRESIDENT Sum Sal	SALESMAN Sum Sal
80	20		$800.00			
81	10			$2,450.00	$5,000.00	
	20	$3,000.00		$2,975.00		
	30		$950.00	$2,850.00		$5,600.00
82	10		$1,300.00			
	20	$3,000.00				
83	20		$1,100.00			

The advantage of using multiple queries is that you get a real break, or master/detail relationship, for the nesting groups (e.g., notice that in the multi-query example above, Year 80 shows only record 20; with a single query, Year 80 would show all records whether or not they contain data for Year 80). If you want to suppress detail records that do not contain data for a particular master record, you must use multiple queries.

For a complete example, see the example report in Chapter 27, "Building a Matrix with Group Above Report".

See also

Section 2.3.7, "About matrix objects"

Section 1.3.7, "About matrix reports"

2.2 Web Reports

The topics in this section build on the basic concepts discussed in Section 1.2.2, "About Web reports".

- About JavaServer Pages (JSPs) and servlets
- About previewing JSP-based Web reports
- About Web links for HTML output
- About Web links for PDF output
- About hyperlinks
- About hyperlink destinations
- About bookmarks
- About application command line links
- About before and after escapes
- About style sheets

See also

Section 1.2.2, "About Web reports"

2.2.1 About JavaServer Pages (JSPs) and servlets

Oracle Reports supports JavaServer Pages (JSPs) as the underlying technology to enable you to enhance Web pages with information retrieved using Reports Builder. JSP technology is an extension to the Java servlet technology from Sun Microsystems that provides a simple programming vehicle for displaying dynamic content on a Web page. A JSP is an HTML page with embedded Java source code that is executed in the Web server or application server. The HTML provides the page layout that is returned to the Web browser, and the Java provides the business logic.

JSPs keep static page presentation and dynamic content generation separate. Because JSPs cleanly separate dynamic application logic from static HTML content, Web page designers who have limited or no Java programming expertise can modify the appearance of the JSP page without affecting the generation of its content, simply using HTML or XML tags to design and format the dynamically-generated Web page. JSP-specific tags or Java-based scriptlets can be utilized to call other components that generate the dynamic content on the page.

JSPs have the .jsp extension. This extension notifies the Web server that the page should be processed by a JSP container. The JSP container interprets the JSP tags and scriptlets, compiles the JSP into a Java servlet and executes it, which generates the content required, and sends the results back to the browser as an HTML or XML page.

A JSP can be accessed and run from a browser-based client, typically over the Internet or a corporate intranet. Unlike traditional client-server applications, JSP applications:

- run on a wider variety of client machines and browsers.
- run on thinner clients, thereby consuming fewer client-machine resources.
- scale to a larger number of simultaneous users.
- require less effort to install and maintain.

When a JSP is called for the first time, it is compiled into a Java servlet class and stored in the Web server's memory. Because it is stored in memory, subsequent calls to that page are very fast, thereby avoiding the performance limitations seen with traditional Common Gateway Interface (CGI) programs, which spawn a new process for each HTTP request.

For additional background information about JSP technology, see the JavaSoft web site at http://www.javasoft.com/products/jsp/.

2.2.1.1 Using JSPs in Oracle Reports

In Oracle Reports, you use JSPs to embed data retrieved using the data model into an existing Web page to create a JSP-based Web report. You can create new JSP reports, or save existing reports as JSP reports. New reports are by default saved as JSP reports. The benefit of saving reports as JSPs is that JSPs are text files that are easy to edit as opposed to, for example, the binary .rdf format. When a report is saved as a JSP file, the data model is embedded using XML tags. The entire report can now be defined using XML tags and saved as an XML file.

Using the Oracle Reports custom JSP tags, you can easily add report blocks and graphs to existing JSP files. These tags can be used as templates to enable you to build and insert your own data-driven Java component into a JSP-based Web report. Not only can you edit the HTML or XML code that encapsulates the report block, but you can also edit the report block in the JSP itself, by modifying, adding or deleting their bodies and attributes.

Not only can you edit the HTML code that encapsulates the report block, but you can also edit the report block in the JSP itself, by modifying, adding or deleting their bodies and attributes.

The Report Editor's Web Source view displays the source code for your Web report, including HTML, XML, and JSP tags.

In prior releases, Oracle Reports introduced Web links that you can add to paper-based reports, which become active when you display your paper report in a Web browser or PDF viewer. For JSP reports, hyperlinks have to be created manually, and if the hyperlinks need to substitute data values, the data values must be provided via the rw:field JSP tag. For example:

```
<a href="http://server/path/rwservlet?report=department.jsp&p_deptno=<rw:field
  id="F_Deptno" src="Deptno"/>">
<rw:field id="F_Deptno" src="Deptno">10</rw:field>
</a>
```

See also

- *Oracle Reports New Features*, available on the Oracle Technology Network Oracle Reports main page (`http://otn.oracle.com/products/reports/content.html`).
- Topic "Custom JSP tags for Oracle Reports" and "XML tag list" in the **Reference** section of the *Reports Builder online help*.

2.2.2 About previewing JSP-based Web reports

You can preview a JSP-based Web report by clicking the Run Web Layout button in the toolbar, or by choosing **Program > Run Web Layout**, to run the Web Source. Reports Builder displays Web reports in your default browser.

> **Note:** If Netscape 7.0 is your default browser, the browser may not display. You can work around this bug by making a copy of the Netscape 7.0 executable, naming it `netscape.exe`; with this name, the browser will display as expected.

You do not need to have the Reports Server configured to use this functionality. Reports Builder includes an embedded OC4J (Oracle Container for Java) server in it. If you are using this option to preview a JSP report locally, if your JSP depends on external files, such as images, or if you want to check the generated Java files, it is important to understand what Reports Builder does with the temporary files. Each

instance of the Builder has its own OC4J server listening on different port, so you can have multiple Builder sessions at a time. If not specified, the Builder automatically looks for a free port in the default range.

A JSP gets converted into a .java file and compiled into a class file. When the class file is executed, it will return HTML in an .html file. This file and the .java and .class files are all located in the $REPORTS_TEMP/docroot directory. $REPORTS_TEMP can be passed in as a command line parameter to Reports Builder, thus allowing you to override the default location for the docroot directory. The contents of the docroot directory are cleaned up when you exit the Reports Builder.

Document Root

By default, the Reports Builder document root is the docroot directory under the directory specified by the $REPORTS_TMP environment variable (e.g., c:/temp/docroot). The end user can override this default docroot from the command line (using the WEBSERVER_DOCROOT command line keyword). If your JSP depends on external files, such as images, style sheets, and so on, make sure you copy them into the docroot directory. Better yet, you can specify the WEBSERVER_DOCROOT command line value to be your document root directory.

Document Root Structure

A JSP gets translated into a .java file and compiled into a .class file. When the class file is executed, it will return HTML in an .html file. This file and the .java and .class files are all located in the docroot directory. The docroot directory structure looks as follows after running emp.jsp (note that we use the default docroot, which is $REPORTS_TMP/docroot):

```
temp
  docroot
    3000                        working directory for instance of the Builder
      default
        defaultWebApp            temporary JSP working directory
          temp
            _pages
            _empxxx.class       compiled Java class
            _empxxx.java         translated Java file
      log                        OC4J log directory
        global-application.log
        server.log
        orion-conf               OC4J configuration files directory
      stderr.log                 debug log when WEBSERVER_LOG=yes
      stdout.log
    3002                         another instance's working directory
```

```
css                          template style sheets
images                       template images
WEB-INF
  lib
  reports_tld.jar
web.xml
rwerror.jsp                  template error JSP
empxxx.jsp                   working copy of emp.jsp
empxxx.html                  resulting output
```

Usage notes

- The large numbers in generated filenames are simply unique IDs, and have no particular meaning (e.g., emp012345678.jsp).
- The temporary files created in the docroot directory get cleaned up when you exit Reports Builder.

See also

Section 3.7.15.4, "Displaying report output in your Web browser"

Section 3.7.16.3, "Printing a report from your Web browser"

2.2.3 About Web links for HTML output

This topic discusses the Web links that you can add to paper-based reports that will become active when you run your report to an HTML file and display it in a Web browser.

In most cases, you can define Web links in an object's Property Inspector. You can specify column and field names in the link value to create dynamic links at runtime. If you require more complex implementation of Web links, such as conditional settings, you must specify the link using PL/SQL and the Reports Builder built-in packaged procedure SRW.SET_ATTR.

Reports output in HTML format can include the following types of Web links:

- A link from an object to another object within the same report, or to another HTML or PDF document (see Section 2.2.5, "About hyperlinks").
- An identifier for the destination of a Web link (see Section 2.2.6, "About hyperlink destinations"). The destination can be any printable object (field, boilerplate, frame, etc.) in your report layout.

- A string in a frame of the master HTML document that links to an associated object (see Section 2.2.7, "About bookmarks"). You can associate a bookmark with any printable object (field, boilerplate, frame, etc.) in your report layout. In the formatted report, you can click on a bookmark to display the associated object at the top of the window.
- URLs that specify image resources. The URLs must be available to your Web server so that the images can be located when the HTML output is displayed by the server.

Additionally, your report can include the following headers and footers that use escapes to add HTML tags to your paper-based report:

- a document header (a before report escape) for placing a logo or some standard links at the beginning of an HTML document (see Section 2.2.9, "About before and after escapes").
- a document footer (an after report escape) for placing a logo or some standard links at the end of an HTML document.
- a page header (a before page escape) for placing a logo or some standard links at the beginning of one page or all pages in an HTML document.
- a page footer (an after page escape) for placing a logo or some standard links at the end of one page or all pages in an HTML document.
- a Parameter Form header (a before form escape) for placing a logo or some standard links in the header of the HTML Parameter Form.
- a Parameter Form footer (an after form escape) for placing a logo or some standard links in the footer of the HTML Parameter Form.

2.2.4 About Web links for PDF output

This topic discusses the Web links that you can add to paper-based reports that will become active when you run your report to a PDF file and display it in a PDF viewer.

Reports output in PDF format can include the following types of Web links:

- A link from an object to another object within the same report, or to another HTML or PDF document (see Section 2.2.5, "About hyperlinks").
- An identifier for the destination of a Web link (see Section 2.2.6, "About hyperlink destinations"). The destination can be any printable object (field, boilerplate, frame, etc.) in your report layout.

- A string in the bookmark area of the PDF viewer that links to an associated object (see Section 2.2.7, "About bookmarks"). You can associate a bookmark with any printable object (field, boilerplate, frame, etc.) in your report layout. In the formatted report, you can click on a bookmark to display the associated object at the top of the window.
- A link that executes a command when clicked (see Section 2.2.8, "About application command line links"). You can associate a command with any printable object (field, boilerplate, frame, etc.) in your report layout. In the formatted report, you can click on the object to execute the associated command.

In most cases, you can define Web links in an object's Property Inspector. You can specify column and field names in the link value to create dynamic links at runtime. If you require more complex implementation of Web links, such as conditional settings, you must specify the link using PL/SQL.

2.2.5 About hyperlinks

A hyperlink is an attribute of an object that specifies a hypertext link to either of the following destinations:

- an object identified with a hyperlink destination within the same report
- another HTML or PDF document on the same machine or on a remote Web server

You can set the Additional Hyperlink Attributes property to specify additional HTML to be applied to the hyperlink.

See also

Section 3.6.7.1.7, "Creating a hyperlink using the Property Inspector"

Section 3.6.7.2.7, "Creating a hyperlink using PL/SQL"

2.2.6 About hyperlink destinations

A hyperlink destination is an attribute of an object that identifies the destination of a hypertext link.

See also

Section 3.6.7.1.8, "Creating a hyperlink destination using the Property Inspector"

Section 3.6.7.2.8, "Creating a hyperlink destination using PL/SQL"

2.2.7 About bookmarks

A bookmark is an attribute of an object that specifies a string that is a link to the object.

See also

Section 3.6.7.1.10, "Creating a bookmark using the Property Inspector"

Section 3.6.7.2.10, "Creating a bookmark using PL/SQL"

Section 3.6.7.1.11, "Creating a bookmark on break columns using the Property Inspector"

2.2.8 About application command line links

(PDF output only) An application command line link is an attribute of an object that specifies a command line to be executed when the object is clicked.

Restrictions

An object that is associated with a application command line link cannot also be the source of a Web link (a hyperlink).

See also

Section 3.6.7.1.9, "Creating an application command line link using the Property Inspector"

Section 3.6.7.2.9, "Creating an application command line link using PL/SQL"

2.2.9 About before and after escapes

- A *before report escape* specifies any text, graphics, or HTML commands that you want to appear at the beginning of your document.
- An *after report escape* specifies any text, graphics, or HTML commands that you want to appear at the end of your document.
- A *before page escape* specifies any text, graphics, or HTML commands that you want to appear at the beginning of one page or all pages of your document.
- An *after page escape* specifies any text, graphics, or HTML commands that you want to appear at the end of one page or all pages of your document.
- A *before form escape* specifies any text, graphics, or HTML commands that you want to appear at the top of the HTML Parameter Form.

- An *after form escape* specifies any text, graphics, or HTML commands that you want to appear at the bottom of the HTML Parameter Form.

See also

Section 3.6.7.1.1, "Creating an HTML document header using the Property Inspector"

Section 3.6.7.2.1, "Creating an HTML document header using PL/SQL"

Section 3.6.7.1.2, "Creating an HTML document footer using the Property Inspector"

Section 3.6.7.2.2, "Creating an HTML document footer using PL/SQL"

Section 3.6.7.1.3, "Creating an HTML page header using the Property Inspector"

Section 3.6.7.2.3, "Creating an HTML page header using PL/SQL"

Section 3.6.7.1.4, "Creating an HTML page footer using the Property Inspector"

Section 3.6.7.2.4, "Creating an HTML page footer using PL/SQL"

Section 3.6.7.1.5, "Creating an HTML Parameter Form header using the Property Inspector"

Section 3.6.7.2.5, "Creating an HTML Parameter Form header using PL/SQL"

Section 3.6.7.1.6, "Creating an HTML Parameter Form footer using the Property Inspector"

Section 3.6.7.2.6, "Creating an HTML Parameter Form footer using PL/SQL"

2.2.10 About style sheets

Style sheets (or cascading style sheets) refer to HTML extensions that provide powerful formatting flexibility. With style sheet support, your HTML documents can include any of the following:

- any font size or style
- overlapping objects
- horizontal and vertical lines and rectangles of any color or width
- precise object positioning on a page
- pagination
- printing from a Web browser
- inline image maps

This means that the sophisticated formatting in a report is preserved when you format the report as an HTML document. Without style sheet extensions, your HTML documents display only basic text formats and imported images. With style sheets, images of highly formatted text can be replaced with text objects of equivalent style, color, and font. Text objects can be positioned to overlay image objects. All text is fully searchable, and fewer images have to be downloaded.

To view an HTML document that takes advantage of style sheets, you must display it in a browser that supports style sheets.

Restrictions

The following elements are not supported by HTML style sheet extensions:

- ellipses, arcs, polygons/polylines, and diagonal lines
- rounded rectangles (formatted as rectangles)
- arrows on lines
- dashes on lines or borders of objects

See also

Section 3.7.15.4, "Displaying report output in your Web browser"

Section 3.7.16.3, "Printing a report from your Web browser"

2.3 Data Model Objects

The topics in this section build on the basic concepts discussed in Section 1.7, "Data Model Objects".

- About summary columns
- About formula columns
- About placeholder columns
- About referencing columns and parameters
- About non-linkable queries
- About links versus groups
- About matrix objects

2.3.1 About summary columns

A summary column performs a computation on another column's data. Using the Report Wizard or Data Wizard, you can create the following summaries: sum, average, count, minimum, maximum,% total. You can also create a summary column manually in the Data Model view, and use the Property Inspector to create the following additional summaries: first, last, standard deviation, variance.

If your report requires a customized computation, for example, one that computes sales tax, create a formula column (see Section 3.8.7, "Creating or editing a formula column").

> **Note:** For group reports, the Report Wizard and Data Wizard create *n* summary fields in the data model for each summary column you define: one at each group level above the column being summarized, and one at the report level. For example, if a report is grouped by division, and further grouped by department, then a summary column defined for a salary total would create fields for the sum of salaries for each division and each department group (group-level summaries), and the sum of all salaries (report-level summary).

See also

Section 3.8.8, "Creating a summary column"

2.3.2 About formula columns

A formula column performs a user-defined computation on the data of one or more column(s), including placeholder columns. For example, :ITEMTOT *.07 is a formula that performs a computation on one column, while :SAL + :COMM performs a computation using two columns in a record. You create formulas in PL/SQL using the PL/SQL Editor.

> **Note:** Formula columns should not be used to set values for parameters.

See also

Section 3.8.7, "Creating or editing a formula column"

Section 2.6.8, "About formulas"

2.3.3 About placeholder columns

A placeholder is a column for which you set the datatype and value in PL/SQL that you define. Placeholder columns are useful when you want to selectively set the value of a column (e.g., each time the nth record is fetched, or each time a record containing a specific value is fetched, etc.). You can set the value of a placeholder column in the following places:

- the Before Report trigger, if the placeholder is a report-level column
- a report-level formula column, if the placeholder is a report-level column
- a formula in the placeholder's group or a group below it (the value is set once for each record of the group)

See also

Section 3.8.9, "Creating or editing a placeholder column"

Section 2.6.8, "About formulas"

2.3.4 About referencing columns and parameters

You can reference user parameters, system parameters and columns as either *bind references* or *lexical references*.

2.3.4.1 About bind references

Bind references (or bind variables) are used to replace a single value in SQL or PL/SQL, such as a character string, number, or date. Specifically, bind references may be used to replace expressions in SELECT, WHERE, GROUP BY, ORDER BY, HAVING, CONNECT BY, and START WITH clauses of queries. Bind references may not be referenced in FROM clauses or in place of reserved words or clauses.

You create a bind reference by typing a colon (:) followed immediately by the column or parameter name. If you do not create a column or parameter before making a bind reference to it in a SELECT statement, Reports Builder will create a parameter for you by default.

Restrictions

Bind references must not be the same name as any reserved SQL keywords. For more information, see the *Oracle9i Server SQL Language Reference Manual*.

Examples

Example 1: SELECT clause

In the following example, the value of DFLTCOMM replaces null values of COMMPLAN in the rows selected.

```
SELECT CUSTID, NVL(COMMPLAN, :DFLTCOMM) COMMPLAN
    FROM ORD;
```

Example 2: WHERE clause

The value of CUST is used to select a single customer.

```
SELECT ORDID, TOTAL
    FROM ORD
    WHERE CUSTID = :CUST;
```

Example 3: GROUP BY clause

All non-aggregate expressions such as NVL(COMMPLAN, :DFLTCOMM) in the SELECT clause must be replicated in the GROUP BY clause.

```
SELECT NVL(COMMPLAN, :DFLTCOMM) COMMPLAN, SUM(TOTAL) TOTAL
    FROM ORD
    GROUP BY NVL(COMMPLAN, :DFLTCOMM);
```

Example 4: HAVING clause

The value of MINTOTAL is used to select customers with a minimum total of orders.

```
SELECT CUSTID, SUM(TOTAL) TOTAL
    FROM ORD
    GROUP BY CUSTID HAVING SUM(TOTAL) > :MINTOTAL;
```

Example 5: ORDER BY clause

The value of SORT is used to select either SHIPDATE or ORDERDATE as the sort criterion. Note that this is not the same as ORDER BY 1 because :SORT is used as a value rather than to identify the position of an expression in the SELECT list. Note that DECODE is required in this example. You cannot use a bind variable in an ORDER BY clause unless it is with DECODE.

```
SELECT ORDID, SHIPDATE, ORDERDATE, TOTAL
    FROM ORD
    ORDER BY DECODE(:SORT, 1, SHIPDATE, 2, ORDERDATE);
```

Example 6: CONNECT BY and START WITH clauses

References in CONNECT BY and START WITH clauses are used in the same way as they are in the WHERE and HAVING clauses.

Example 7: PL/SQL

```
procedure double is begin; :my_param := :my_param*2; end;
```

The value of `myparam` is multiplied by two and assigned to myparam.

2.3.4.2 About lexical references

Lexical references are placeholders for text that you embed in a SELECT statement. You can use lexical references to replace the clauses appearing after SELECT, FROM, WHERE, GROUP BY, ORDER BY, HAVING, CONNECT BY, and START WITH. Use a lexical reference when you want the parameter to substitute multiple values at runtime.

You cannot make lexical references in a PL/SQL statement. You can, however, use a bind reference in PL/SQL to set the value of a parameter that is then referenced lexically in SQL. Look at the example below.

You create a lexical reference by typing an ampersand (&) followed immediately by the column or parameter name. A default definition is not provided for lexical references. Therefore, you must do the following:

- Before you create your query, define a column or parameter in the data model for each lexical reference in the query. For columns, you must set the Value if Null property, and, for parameters, you must set the Initial Value property. Reports Builder uses these values to validate a query with a lexical reference.
- Create your query containing lexical references.

Restrictions

- You cannot make lexical references in a PL/SQL statement.
- If a column or parameter is used as a lexical reference in a query, its Datatype must be Character.
- If you want to use lexical references in your SELECT clause, you should create a separate lexical reference for each column you will substitute. In addition, you should assign an alias to each lexical reference. This enables you to use the same layout field and boilerplate label for whatever value you enter for the lexical reference on the Runtime Parameter Form.
- If you use lexical references in your SELECT clause, you must specify the same number of items at runtime as were specified in the report's data model. Each value you specify for your lexical references at runtime must have the same datatype as its Initial Value.
- If you use lexical references in your SELECT clause, the width of the column is derived from the Initial Value of the parameter. Consequently, you should ensure that the Initial Value of the parameter corresponds to the widest column that you intend to use.

- A Reports Builder link should not depend upon a lexical reference. That is, neither the child column of a link or its table name should be determined by a lexical reference. To achieve this functionality, you need to create a link with no columns specified and then enter the SQL clause (e.g., WHERE) for the link directly in the query. For example, your parent and child queries might be written as follows:

  Parent Query:

  ```
  SELECT DEPTNO FROM EMP
  ```

  Child Query:

  ```
  SELECT &PARM_1 COL_1, &PARM2 COL_2
      FROM EMP
      WHERE &PARM_1 = :DEPTNO
  ```

  Note how the WHERE clause makes a bind reference to DEPTNO, which was selected in the parent query. Also, this example assumes that you have created a link between the queries in the Data Model editor with no columns specified.

- A lexical reference cannot be used to create additional bind variables after the After Form trigger fires. For example, suppose you have a query like the following (note that the WHERE clause is replaced by a lexical reference):

  ```
  SELECT ENAME, SAL FROM EMP
  &where_clause
  ```

  If the value of the WHERE_CLAUSE parameter contains a reference to a bwind variable, you must specify the value in the After Form trigger or earlier. You would get an error if you supplied the following value for the parameter in the Before Report trigger:

  ```
  WHERE SAL = :new_bind
  ```

If you supplied this same value in the After Form trigger, the report would run.

Examples

Example 1: SELECT clause

```
SELECT &P_ENAME NAME, &P_EMPNO ENO, &P_JOB ROLE
    FROM EMP;
```

P_ENAME, P_EMPNO, and P_JOB can be used to change the columns selected at runtime. For example, you could enter DEPTNO as the value for P_EMPNO on the Runtime Parameter Form. Note that in this case, you should use aliases for your

columns. Otherwise, if you change the columns selected at runtime, the column names in the SELECT list will not match the Reports Builder columns and the report will not run.

Example 2: FROM clause

```
SELECT ORDID, TOTAL
    FROM &ATABLE;
```

ATABLE can be used to change the table from which columns are selected at runtime. For example, you could enter ORD for ATABLE at runtime. If you dynamically change the table name in this way, you may also want to use lexical references for the SELECT clause (look at the previous example) in case the column names differ between tables.

Example 3: WHERE clause

```
SELECT ORDID, TOTAL
    FROM ORD
    WHERE &CUST;
```

CUST can be used to restrict records retrieved from ORD. Any form of the WHERE clause can be specified at run-time.

Example 4: GROUP BY clause

```
SELECT NVL(COMMPLAN, DFLTCOMM) CPLAN, SUM(TOTAL) TOTAL
    FROM ORD
    GROUP BY &NEWCOMM;
```

The value of NEWCOMM can be used to define the GROUP BY clause.

Example 5: HAVING clause

```
SELECT CUSTID, SUM(TOTAL) TOTAL
    FROM ORD
    GROUP BY CUSTID HAVING &MINTOTAL;
```

The value of MINTOTAL could, for example, be used to select customers with a minimum total of orders.

Example 6: ORDER BY clause

```
SELECT ORDID, SHIPDATE, ORDERDATE, TOTAL
    FROM ORD
    ORDER BY &SORT;
```

The value of SORT can be used to select SHIPDATE, ORDERDATE, ORDID, or any combination as the sort criterion. It could also be used to add on to the query, for example to add a CONNECT BY and START WITH clause.

Example 7: CONNECT BY and START WITH clauses

Parameters in CONNECT BY and START WITH clauses are used in the same way as they are in the WHERE and HAVING clauses.

Example 8: Multiple clauses

```
SELECT &COLSTABLE;
```

COLSTABLE could be used to change both the SELECT and FROM clauses at runtime. For example, you could enter DNAME ENAME, LOC SAL FROM DEPT for COLSTABLE at runtime.

```
SELECT * FROM EMP &WHEREORD;
```

WHEREORD could be used to change both the WHERE and ORDER BY clauses at runtime. For example, you could enter WHERE SAL > 1000 ORDER BY DEPTNO for &WHEREORD at runtime.

Example 9: PL/SQL and SQL

```
SELECT &BREAK_COL C1, MAX(SAL)
    FROM EMP
    GROUP BY &BREAK_COL;
```

BREAK_COL is used to change both the SELECT list and the GROUP BY clause at runtime. The Initial Value of the parameter &BREAK_COL is JOB. At runtime, the user of the report can provide a value for a parameter called GROUP_BY_ COLUMN (of Datatype Character).

In the Validation Trigger for GROUP_BY_COLUMN, you call the following PL/SQL procedure and pass it the value of GROUP_BY_COLUMN:

```
procedure conv_param (in_var IN char) is
begin
    if upper(in_var) in ('DEPTNO','EMPNO','HIREDATE') then
        :break_col := 'to_char('||in_var||')' ;
    else
        :break_col := in_var;
    end if;
end;
```

This PL/SQL ensures that, if necessary, a TO_CHAR is placed around the break column the user chooses. Notice how in SQL, you make a lexical reference to BREAK_COL. In PL/SQL, you must make a bind reference to BREAK_COL because lexical references are not allowed in PL/SQL.

2.3.4.3 Differences between bind and lexical references

Bind references are used to replace a single value in SQL or PL/SQL. Specifically, bind references may be used to replace expressions in SELECT, WHERE, GROUP BY, ORDER BY, HAVING,CONNECT BY, and START WITH clauses of queries. Binds may not be referenced in the FROM clause. An example is:

```
SELECT ORDID,TOTAL
    FROM ORD
    WHERE CUSTID = :CUST
```

Lexical references are placeholders for text that you embed in a SELECT statement, when you want the parameter to substitute multiple values at runtime. You can use lexical references to replace the clauses appearing after SELECT, FROM, WHERE, GROUP BY, ORDER BY, HAVING, CONNECT BY, and START WITH. You cannot make lexical references in PL/SQL. Before you reference a lexical parameter in a query you must have predefined the parameter and given it an initial value. An example is:

```
SELECT ORDID, TOTAL
    FROM &ATABLE
```

2.3.5 About non-linkable queries

A non-linkable query is a detail query that contains column objects that prevent the query from being linked to via a column-to-column link (when you create a column-to-column link, Reports Builder adds a WHERE clause to your query). If you attempt to create such a link, a message dialog box displays, which prompts you to choose whether to create a group-to-group query (using the parent groups), or to cancel the operation. A non-linkable query displays the non-linkable query icon in its title bar.

Instead, you can create a group-to-group link (when you create a group-to-group link, Reports Builder does not add a WHERE clause to your query) between the two queries and add a WHERE clause to the child query's SELECT statement, using a bind variable to reference the parent column. See Section 3.8.6, "Creating a data link".

For example, suppose you want to create a column-to-column link between the ADDRESS.STREET column in your child query and the LOC1 column in your parent query. You can create a group-to-group link, and then modify the child query SQL statement to say:

```
SELECT * FROM EMP E WHERE E.ADDRESS.STREET = :LOC1
```

See also

Section 1.7.4, "About data links"

Section 1.7.1, "About queries"

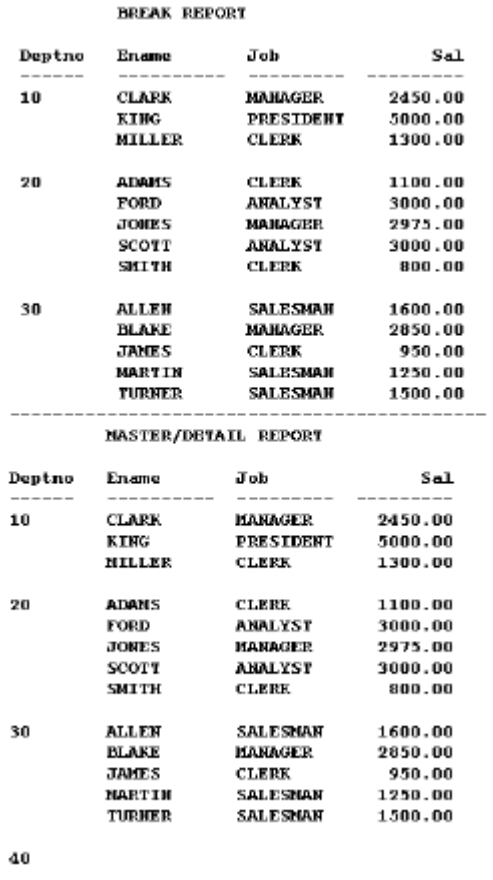
Section 2.3.4.1, "About bind references"

2.3.6 About links versus groups

In Reports Builder, data is defined independent of format (layout). Therefore, you should be aware of when to use data links instead of groups.

The layouts of a master/detail report that uses two queries and a data link, and a group report that uses one query and two groups can be identical. Following is an example of a default master/detail report and a group report that query the same data. Notice the difference between the two reports: unlike the group report, the master/detail report displays department 40. This is because the data link in the master/detail report causes an outer-join: the link automatically fetches unrelated data. If you are designing a group report that requires an outer-join, explicitly add it to your SELECT statement via (+).

Figure 2–3 Default master/detail and group report that query same data



BREAK REPORT

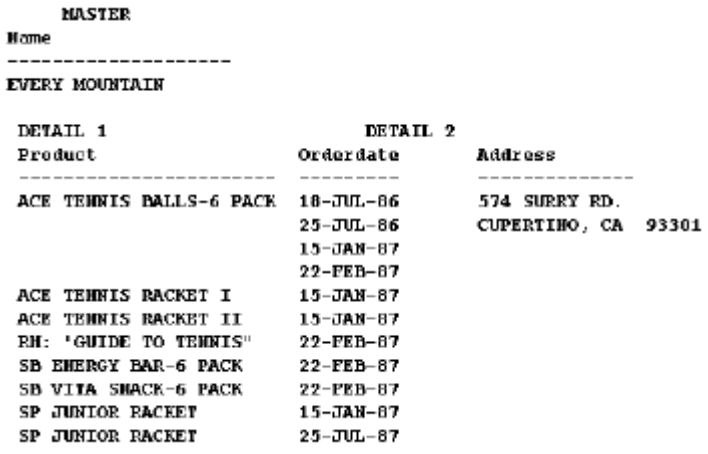
Deptno	Ename	Job	Sal
10	CLARK	MANAGER	2450.00
	KING	PRESIDENT	5000.00
	MILLER	CLERK	1300.00
20	ADAMS	CLERK	1100.00
	FORD	ANALYST	3000.00
	JONES	MANAGER	2975.00
	SCOTT	ANALYST	3000.00
	SMITH	CLERK	800.00
30	ALLEN	SALESMAN	1600.00
	BLAKE	MANAGER	2850.00
	JAMES	CLERK	950.00
	MARTIN	SALESMAN	1250.00
	TURNER	SALESMAN	1500.00

MASTER/DETAIL REPORT

Deptno	Ename	Job	Sal
10	CLARK	MANAGER	2450.00
	KING	PRESIDENT	5000.00
	MILLER	CLERK	1300.00
20	ADAMS	CLERK	1100.00
	FORD	ANALYST	3000.00
	JONES	MANAGER	2975.00
	SCOTT	ANALYST	3000.00
	SMITH	CLERK	800.00
30	ALLEN	SALESMAN	1600.00
	BLAKE	MANAGER	2850.00
	JAMES	CLERK	950.00
	MARTIN	SALESMAN	1250.00
	TURNER	SALESMAN	1500.00
40			

A master/detail/detail report, as shown in the figure below, is a report that contains three groups of data: for each master group, two unrelated detail groups are displayed. To produce a master/detail/detail report or any variation of it, you must use data links. If you try to produce this report with a control break using a single query and three groups the query will establish a relationship between the two detail groups.

Figure 2–4 Master/detail/detail report



```
     MASTER
Name
--------------------
EVERY MOUNTAIN

 DETAIL 1                      DETAIL 2
 Product                 Orderdate       Address
 ----------------------  ---------       --------------
 ACE TENNIS BALLS-6 PACK 18-JUL-86       574 SURRY RD.
                         25-JUL-86       CUPERTINO, CA  93301
                         15-JAN-87
                         22-FEB-87
 ACE TENNIS RACKET I     15-JAN-87
 ACE TENNIS RACKET II    15-JAN-87
 RH: 'GUIDE TO TENNIS"   22-FEB-87
 SB ENERGY BAR-6 PACK    22-FEB-87
 SB VITA SNACK-6 PACK    22-FEB-87
 SP JUNIOR RACKET        15-JAN-87
 SP JUNIOR RACKET        25-JUL-87
```

See also

Section 1.7.2, "About groups"

Section 1.7.4, "About data links"

2.3.7 About matrix objects

A matrix object merely defines a relationship between two repeating frames: it isn't really owned by anything, nor does it own anything. A matrix object is created only for layouts with a Matrix layout style. A report may have multiple matrices within it, provided that the data model contains the necessary groups. Reports Builder creates one matrix object for each pair of intersecting, perpendicular repeating frames.

The repeating frames are the dimensions of the matrix and the matrix object contains the field that will hold the ”filler” or values of the cell group. One of the

repeating frames must have the Print Direction property set to *Down* and the other must have the Print Direction property set to *Across* in order to form a matrix.

Restrictions

- The down repeating frames must be below the across repeating frames in a matrix.
- A matrix object must always be on top of the repeating frames that form it (i.e., it must be one or more layers above its horizontal and vertical repeating frames). Reports Builder prevents you from moving the matrix below its horizontal and vertical repeating frames.
- Moving a matrix also causes its two repeating frames to move.
- A matrix object cannot be anchored to another object and other objects cannot be anchored to it (i.e., a matrix object cannot be the parent or child object for an anchor).
- To copy a matrix, you must select the matrix and its two repeating frames. If you select the matrix object by itself, nothing will be copied to the paste buffer. If you select the matrix and one of the repeating frames, only the repeating frame is placed in the paste buffer.
- A matrix object can only be resized by resizing its associated repeating frames.
- You cannot use Alignment or Size Objects from the Layout menu on matrix objects.
- The source groups of the repeating frames that make up the dimensions of a matrix must be from the same cross-product group.
- Repeating frames whose source groups are in the same "family" hierarchy (i.e., are descendants or ancestors of each other) must have the same Print Direction. Parent-child relationships within a cross-product group are used to create nesting in the matrix. As a result, the repeating frames associated with such groups must print in the same direction on the page.
- You can put a border on a matrix object just as you would any other object, but the width will always be the minimum width possible. You cannot widen the border due to the closeness of the objects in a matrix layout.

Example

Suppose that you have a group named Group1 that contains a column called C_DEPTNO, which gets its values from the database column DEPTNO. A group called Group2, contains column C_JOB, which gets its values from the database

column JOB, and column C_DEPTNO1, which is used for linking to Group1's query. A group called Group3 contains a column called SUMSAL, which is a summary of the database column SAL.

```
                  Job
          Analyst  Clerk  Manager

      10          $1300   $2450
Dept  20    $6000 $1900   $2975
      30          $ 950   $2850
```

In this example:

- The Vertical Repeating Frame is the repeating frame that contains Group2 (the job titles).
- The Horizontal Repeating Frame is the repeating frame that contains Group1 (the department numbers).
- The Cross Product Group is Group4 (the group that is the parent of Group1 and Group2).

If you need to build a more complex matrix, you can do so by adding more columns to Group1 and Group2. For example, instead of having Group1 just contain department numbers, it could also contain the locations (LOC) of the departments. The matrix might then look something like this:

```
                          Job
Loc         Dept    Analyst  Clerk  Manager

New York     10             $1300   $2450
Dallas       20      $6000  $1900   $2975
Chicago      30             $ 950   $2850
```

See also

Section 1.3.7, "About matrix reports"

Section 2.1.7, "About nested matrix reports"

Section 2.1.8, "About matrix with group reports"

Section 3.9.1.3, "Creating a matrix object"

Section 3.8.5, "Creating a matrix (cross-product) group"

Section 3.5.3, "Creating a nested matrix report"

2.4 Layout Objects

The topics in this section build on the basic concepts discussed in Section 1.8, "Layout Objects".

- About layout objects
- About layout defaulting
- About images
- About anchors
- About changing colors and patterns
- About resizing objects
- About moving and layering objects in the Paper Layout view

2.4.1 About layout objects

Several important concepts and properties apply to layout objects:

- the frequency with which you want the object to appear in the report, specified by the Print Object On property
- how Reports Builder fetches and formats data for instances of repeating frames, specified by the Column Mode property
- whether to keep an object and the object to which it is anchored on the same logical page, specified by the Keep With Anchoring Object property
- whether to try to keep the entire object and its contents on the same logical page, specified by the Page Protect property
- format triggers, which are PL/SQL functions executed before an object is formatted that can dynamically change the formatting attributes of objects
- report layout, generated by defaulting applied by Reports Builder, modified in the Paper Layout view, or created from scratch.

See also

Section 3.5.4, "Creating a default layout for a report"

Section 3.10.2, "Creating a default layout for a section"

Section 2.6.12.2, "About format triggers"

The **Properties** section of the *Reports Builder online help*

2.4.2 About layout defaulting

When you select one of the default layout styles in the Report Wizard, Reports Builder creates the necessary layout objects, based upon the report's data model. For example, if you want to build a mailing label report and have defined an appropriate data model, simply choose the mailing label default style. Reports Builder automatically creates the report's layout objects and displays them in the Layout Model view. You can completely customize any default layout you create. You can cut, copy, paste, move, resize, and edit each layout object that Reports Builder generates for you.

Layout defaulting is governed by the following rules:

- All previously-defined layout objects for the report will be overwritten (including format triggers) unless you define the area in which you wish to create the layout as one which does not already contain layout objects.
- One report can have any number of different formats. For example, you can build a report that has a tabular format on the top of the first page, and a matrix format on the bottom of the same page by creating an additional report layout (see Section 3.5.5, "Creating an additional report layout").
- Once you have created a report layout, any further changes you make to the data model will not automatically be included in the layout. For example, if you create a query after you have created a report layout and then run your report, the data from the new query will not appear in the report output. To incorporate your changes, you need to either redefault or modify the layout.

Reports Builder defaults report layout according to the following rules:

1. Unless otherwise noted, a group with a Print Direction of *Across* defaults identically to a group with a Print Direction of *Down*, except that the default format is transposed. To quickly determine the defaulting of an Across group, do the following:
   - Draw the down layout for the group on translucent paper.
   - Turn the page over as if it was a page in a book.
   - Rotate the page counter-clockwise 90 degrees.
2. In form letter reports, all default fields are hidden, and have a Horizontal Elasticity property setting of *Variable* and a Vertical Elasticity property setting of *Fixed*. In all other reports, all default fields have a Horizontal and Vertical Elasticity property setting of *Fixed*.

Exception: For tabular, form-like, group left/above, and matrix reports, the default for CHAR (if you reduce the default width) and LONG fields have a Horizontal Elasticity property setting of *Fixed* and a Vertical Elasticity property setting of *Variable*. As a result, all of the field's value will be displayed, instead of truncated, by word-wrapping any data to the next lines.

Caution: For form letter and mailing label reports, the default for CHAR (if you reduce the default width) and LONG fields have a Horizontal Elasticity and Vertical Elasticity property setting of Fixed. As a result, the field value will be truncated if the size of the data is greater than the size of the field.

---

3. Summaries that are owned by the report, not by a group, are allowed in all report layout styles. They are all formatted in the following way:
   - If the layout style is form-like, form letter, or mailing label, the summary is a report column. A report column is formatted like any other database column for that report style.
   - If the layout style is tabular, group left/above, or matrix, and the column on which the summary performs its function is not selected, the summary is a report column. (It is formatted as stated in the bullet above.) Otherwise, the summary is a report summary. It is formatted left-justified at the end of the report, with the label to the left of the field (if there is room).
4. If the layout style is tabular, group left/above, or matrix, a summary is defaulted like a database column if the column it summarizes is not selected. Otherwise, a summary is defaulted like summaries (i.e., appearing inside the M_*groupname*_FTR frames).
5. If the layout style is tabular, group left/above, or matrix, Reports Builder places one summary type per line, in the following order:
   - SUM
   - AVERAGE
   - MINIMUM
   - MAXIMUM
   - COUNT
   - FIRST

- LAST
- % OF TOTAL
- STANDARD DEVIATION
- VARIANCE

6. If the layout style is tabular, group left/above, or matrix, the summary label will appear to the far-left of the group footer frame (M_*groupname*_FTR). If there is not room for the full label, the label will be truncated.

See also

Section 3.5.4, "Creating a default layout for a report"

Section 3.10.2, "Creating a default layout for a section"

2.4.3 About images

You can add an image to a report by:

- selecting a column in the database that contains images (see Section 3.9.8.7, "Selecting an image from the database")
- importing the image from a file into the report layout (see Section 3.9.8.6, "Importing a drawing or image")
- linking an image object to a file (see Section 3.9.8.9, "Linking an image object to a file")
- for HTML output, selecting a column in the database that contains URL links to images or linking an image object to a URL (see Section 3.9.8.8, "Selecting an image URL from the database" and Section 3.9.8.10, "Linking an image object to a URL")

By default, images display in fields so that they appear in the printed report, not only in the Previewer.

Oracle Reports 10*g* (9.0.4) enhances imaging support via the REPORTS_OUTPUTIMAGEFORMAT environment variable and OUTPUTIMAGEFORMAT command line keyword. The image formats supported in the report definition are: JPEG (all types), PNG, BMP, TIFF, GIF, CGM, and OGD. The enhancements in imaging support provide the capability to generate complex graphics-intensive reports with high fidelity image output. Additionally on UNIX, the dependency on a windowing system for displaying images is removed; the PostScript printer driver `screenprinter.ppd` provides surface resolution for images.

You can include an unlimited number of image objects without running out of local disk space by using non-caching references. A non-caching reference causes objects to be read from the database only when needed while a report is processing. You must be connected to an ORACLE V7.1 or later database to use this feature.

If you reference a URL for an image, the image is displayed when you format your report for HTML output. For other output formats, the URL text displays in the Paper Design view; in the output destination (for example, a file or PDF document), nothing is displayed. It is your responsibility to verify that the URL exists; Reports Builder does not validate the existence of the resource nor the syntax of the protocol. The size of the object that contains the URL defines the size of the image in the HTML output. Any elasticity properties applied to the object are ignored.

Limitations

- If the input image includes more than 256 colors, and the output image format is set to GIF (via the OUTPUTIMAGEFORMAT command line keyword or the REPORTS_OUTPUTIMAGEFORMAT environment variable), Oracle Reports implements a color reduction to 256 colors to successfully generate the GIF.
- On UNIX, CGM and OGD (Oracle6*i* Graphics) formats are not supported in HTML output. This limitation does not apply on the Windows platform.

See also

Section 3.9.8.6, "Importing a drawing or image"

Section 3.9.8.7, "Selecting an image from the database"

Section 3.9.8.8, "Selecting an image URL from the database"

Section 3.9.8.9, "Linking an image object to a file"

Section 3.9.8.10, "Linking an image object to a URL"

2.4.4 About anchors

Anchors fasten an edge of one object to an edge of another object, ensuring that they maintain their relative positions. For example, you can anchor boilerplate text to the edge of a variable-sized repeating frame, guaranteeing the boilerplate's distance and position in relation to the repeating frame, no matter how the frame's size might change.

Anchors determine the vertical and horizontal positioning of a child object relative to its parent. The child object may be either outside of or contained within the parent.

Since the size of some layout objects may change when the report runs (and data is actually fetched), you need anchors to define where you want objects to appear relative to one another. An anchor defines the relative position of an object to the object to which it is anchored. Positioning is based on the size of the objects after the data has been fetched rather than on their size in the editor. It should also be noted that the position of the object in the Paper Layout view affects the final position in the report output. Any physical offset in the layout is incorporated into the percentage position specified in the Anchor properties.

There are two types of anchors:

- **Implicit anchors**. At runtime, Reports Builder generates an implicit anchor for each layout object that does not already have an explicit anchor. It determines for each layout object which objects, if any, can overwrite it, then creates an anchor from the layout object to the closest object that can overwrite it. This prevents the object from being overwritten. The implicit anchor functionality saves you from having to define the positioning of each object. Implicit anchors are not visible in the Paper Layout view. However, you can specify in the Object Navigator Options dialog box that the Object Navigator display anchoring information. By default, objects are anchored to the upper left corner of their enclosing object. If this view of the Object Navigator does not show anchoring information for an object, you can assume that the object is anchored to its enclosing object, which might be the frame or the body.
- **Explicit anchors**. You can create an anchor in the Layout editor using the Anchor tool, dragging from one edge of the child to the one of the parent's edges. Any anchor you create for an object will override its implicit anchoring. Explicit anchors are always visible in the Paper Layout view unless you specify otherwise in the Layout Options dialog box.

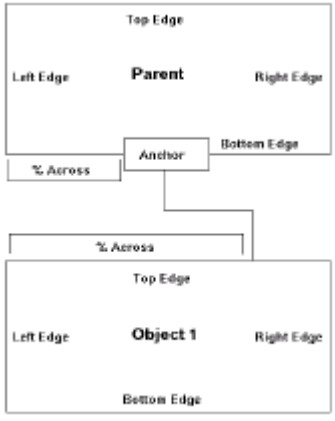
Relative positioning of anchors

When you anchor a child object to a parent object, the x and y coordinates of the anchor's attachments are important.

If the parent object is located above or below the child object:

- the vertical distance between the two objects is fixed. For example, in the figure below, the vertical spacing between the parent and Object 1 is fixed.
- the horizontal positioning of the anchor's x-coordinate on the child object is relative to the anchor's x-coordinate on the parent object. For example, in the figure below, the anchor is 50% from the edge of the parent and 75% from the left edge of Object 1. Therefore, when this report is run, Reports Builder will shift Object 1 25% to the left of the center of the parent.

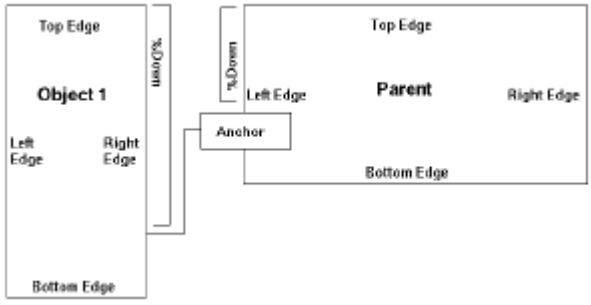
Figure 2–5 Parent object above child object



If the parent object is located to the right or left of the child object:

- the vertical positioning of the two objects is relative. For example, in the figure below, both ends of the anchor are about 80% down from the top edges of the objects. Therefore, when the report is run, Reports Builder will calculate the length of the two objects (as they may expand), calculate the y coordinate that is 80% down for both objects, and position the two objects so that those two points are separated by the amount of space separating them in the Paper Layout view.
- the horizontal positioning between the two objects is fixed. For example, in the figure below, the horizontal spacing between the parent and Object 1 is fixed.

Figure 2–6 Parent object to the right of child object



If you need to position an object outside a repeating frame or frame, but you want the object to be "owned" by the repeating frame or frame (i.e., to be formatted when its "owner" is formatted), create an anchor that is attached to an object inside the frame or repeating frame.

Collapsing Anchors

You can create anchors to be "collapsible." Collapsing anchors help avoid unnecessary empty space in your report. Such empty space can occur when the parent object does not print on the same page as the child object, either because the parent and child can not fit on the same page or because of an assigned Print Condition. A collapsing anchor allows the child object to move into the position that would have been taken by the parent had it printed. The child object will also maintain its relative position as defined by the anchor.

2.4.4.1 Implicit anchoring algorithm

Reports Builder creates implicit anchors at runtime in the body region. The margin algorithm differs slightly.

Body algorithm

1. Determine which objects are not entirely enclosed by a repeating frame of frame (directly or indirectly), or explicitly anchored to an object that is enclosed by a frame or repeating frame (directly or indirectly). We'll call these objects Type A objects. (Type A objects are typically group frames, repeating frames, other objects you create that are not owned by a frame or repeating frame, etc.) An object is considered to be enclosed by another object only if all of the following are true:
   - Both objects belong to the same region (Body or Margin).
   - The outermost of the two objects is a frame or repeating frame.
   - The outermost of the two objects is behind the other object.
   - The innermost of the two objects lies entirely within the borders of the other object.
2. Determine all children objects of a frame or repeating frame (these are the non-Type A objects). We'll call these objects Type B objects.
3. Follow this procedure for Type A and Type B objects independently:
   - Find all objects that are of the same type (e.g., Type A), and are on the same layer.
   - Determine which of those objects have potential to "push" other objects of that type. An object has potential to "push" object of that type if it has a Horizontal and/or Vertical Elasticity setting of Variable or Expand, and a second object is located in its "push path" (i.e., in the area in which it can possibly grow). Also, a repeating frame with a Horizontal and/or Vertical Elasticity setting of Fixed or Contract has a "push path": its Print Direction.
   - Create pairs of objects. Each pair must contain a pusher (i.e., the object that will grow) and a pushee (i.e., the object that will be pushed). When creating these pairs, a pushee object cannot be a child of an explicit anchor--those objects are ignored.
   - Go through this loop. For each pair, determine the distance in the "push path" between the pusher and pushee. Next, find the pair with the shortest

distance. Finally, create an implicit anchor between those two objects using this algorithm:

If the "push path" direction is *Down*, anchor the pushee object's top 0% to the pusher object's bottom 0%.

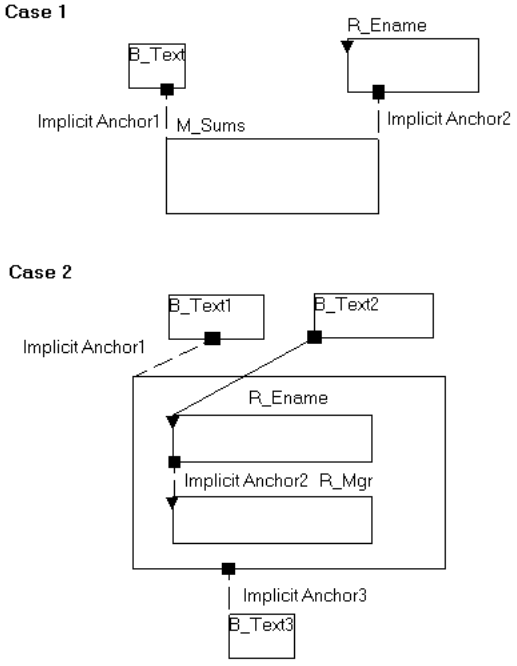
If the "push path" direction is *Across*, anchor the pushee object's left 0% to the pushed object's right 0%.

- That pair is now treated like one object, and the loop continues until either all objects have one anchor, or nothing will push the remaining, unanchored objects.
- For each remaining, unanchored object, create an implicit anchor from the top-left corner of the object to the top-left corner of the body region.
- Move to the next layer, and follow the procedure starting at step 1.

Rules:

- If an object is in the "push path" of two other objects and it is equidistant from the other two objects, the implicit anchoring of the object may vary between executions of the report. For example, the drawing that follows shows two cases where this could occur:

Figure 2–7 Object in push path of two other objects



In the first case, M_Sums is in the "push path" of both B_Text1 and R_Ename. Because M_Sums is equidistant from B_Text1 and R_Ename, though, the normal criteria (shortest distance) for determining implicit anchors does not work in this case. Consequently, the formatting algorithm will randomly create an implicit anchor between M_Sums and either B_Text1 or R_Ename at runtime. To avoid this behavior, you could create an explicit anchor between M_Sums and B_Text1 or R_Ename.

In the second case, B_Text3 is in the "push path" of M_Emp. Since the bottom edges of M_Emp and R_Mgr are virtually in the same position, though, B_Text3 could be implicitly anchored to either M_Emp or R_Mgr. Consequently, the formatting algorithm will randomly create an implicit anchor between B_Text3 and either M_Emp or R_Mgr at runtime. To avoid this behavior, you could create an explicit anchor from B_Text3 to one of the objects or remove the explicit anchor between R_Ename and B_Text2. Removing the explicit anchor would cause R_Mgr to be treated as a descendant of M_Emp and, therefore, the implicit anchor would always be created between B_Text3 and M_Emp.

(Note that Case 2 is most likely to occur in character mode, where it is common to have the edges of objects overlap in the Paper Layout view.)

Margin algorithm

Reports Builder creates implicit anchors for all Type B objects in the margin region using the Body algorithm. For each Type A object, however, Reports Builder creates an implicit anchor from the top-left corner of the object to the top-left corner of the margin. No Type A object will be implicitly anchored to another Type A object. (This ensures that Type A objects will not be pushed off the page. However, they may be overwritten by another Type A object, if they are found on the same layer.)

See also

Section 3.9.5.1, "Anchoring objects together"

Section 3.9.5.2, "Viewing implicit anchors"

Section 3.9.5.3, "Moving an anchor"

2.4.5 About changing colors and patterns

Color and pattern selections are applied to an entire object (e.g., you can apply a color to all the text in the object but not to a segment of the text).

You can change colors and patterns in your report in the following ways:

- In the Reports Builder user interface, use the following tools in the Paper Layout view's tool palette:
  - The **Line Color** tool is used to customize the color of borders around layout objects.

    Note: The Windows platform does not support a border pattern (that is, patterns for the Line Color tool).

  - The **Fill Color** tool is used to fill layout objects with colors and patterns.
  - The **Text Color** tool is used to change the default text color.
  - The **Fill/Line/Text Display**, the box directly above the three Color tools, shows the currently selected fill, border, and text. The default fill and border for objects created by Reports Builder is transparent, while the default for objects you create is a black, one point line around a white fill.

- In PL/SQL, use the following SRW packaged procedures:
  - SRW.SET_BACKGROUND_BORDER_COLOR
  - SRW.SET_BACKGROUND_FILL_COLOR
  - SRW.SET_FOREGROUND_BORDER_COLOR
  - SRW.SET_FOREGROUND_FILL_COLOR
  - SRW.SET_TEXT_COLOR
  - SRW.SET_FILL_PATTERN
  - SRW.SET_BORDER_PATTERN
- For templates, set the following properties in the template Property Inspector:
  - The Fill Pattern property defines the pattern to use for the space enclosed by the objects. You can define the background and foreground colors of the fill pattern using the Foreground Color and Background Color properties.
  - The Edge Pattern property defines the pattern to use for the borders of the objects. You can define the background and foreground colors of the edge pattern using the Edge Foreground Color and Edge Background Color properties.

    Note: The Windows platform does not support a border pattern.

  - The Text Color property specifies the text color to use for the object(s).

Additionally, you can set color palette preferences to specify how it is used by a report (see Section 3.2.6, "Setting color palette preferences") and modify the color palette to change the definition of individual colors (see Section 3.9.6.5, "Modifying the color palette").

To change the color palette being used by the current report, you can import a new color palette. You can also export the current color palette for use by other reports. (See Section 3.9.6.6, "Importing or exporting a color palette".)

See also

Section 3.9.6.2, "Changing colors"

Section 3.9.6.3, "Changing patterns"

Section 3.9.6.4, "Changing colors and patterns using PL/SQL"

Section 3.9.4.2, "Changing object border attributes"

Topics "Oracle CDE1 color palette", "Default color palette", "Grayscale color palette", and "Pattern color palette" in the **Reference > Color and Pattern Palettes** section of the *Reports Builder online help*

Topic "SRW built-in package" in the **Reference > PL/SQL Reference > Built-in Packages** section of the *Reports Builder online help*

Topic "Template properties" in the **Properties** section of the *Reports Builder online help*

2.4.6 About resizing objects

You can resize queries, groups, frames, repeating frames, fields, matrix objects, and boilerplate objects. You cannot resize anchors. However, an anchor is automatically resized if you move one of the objects it anchors.

Caution: When you resize boilerplate text, be very careful that all of the text fits within the object. If font descends (the space left for the lower parts of letters like g and q) do not fit, the line of text will appear in the Report Editor view, but, when the report is run, the line will not appear in the output. When you click a handle and drag it, the two edges that join at the corner will be resized; that is, the object will grow or reduce in both the x and y directions.

See also

Section 3.9.12.1, "Resizing objects"

Section 3.9.12.2, "Making multiple objects the same size"

Section 3.9.11.3, "Adjusting parent borders automatically"

2.4.7 About moving and layering objects in the Paper Layout view

In the Paper Layout view, objects must be on a layer above the objects that enclose them. For example, the fields that belong to a repeating frame must be at least one layer above the repeating frame in the Paper Layout view. If not, then they are not considered to be enclosed by the repeating frame any longer and will cause a frequency error at runtime. When you move or group objects in the Paper Layout view, it is possible to change the layering such that you will get frequency errors

when you run the report. To avoid this problem, you should take advantage of Confine or Flex mode when moving objects in the Paper Layout view.

See also

Section 3.9.4.3, "Changing the current mode (Confine or Flex)"

Section 3.9.11.7, "Changing object layering"

Section 3.9.11.2, "Moving an object outside its parent"

Section 3.9.11.1, "Moving multiple objects"

Section 3.9.11.6, "Aligning objects"

2.5 Parameter Form Objects

The topic in this section builds on the basic concepts discussed in Section 1.9, "Parameter Form Objects".

2.5.1 About Parameter Form HTML extensions

Parameter Form HTML extensions enable you to enhance your Runtime Parameter Form with HTML tagged text and JavaScript when your paper reports are run via the Web. To enhance your Paper Parameter Form for displaying on the Web, you can:

- create boilerplate text with HTML tags for adding hyperlinks or any other HTML tagged text to your Parameter Form (see Section 3.6.7.1.14, "Creating a boilerplate text object for HTML tags").
- insert parameter fields with JavaScript for defining input or select events, such as raising errors when users enter invalid data in a parameter field (see.Section 3.11.12, "Creating HTML Parameter Form input or select events").
- create a Parameter Form header (a before form escape) for placing a logo or some standard links in the header of the HTML Parameter Form (see Section 3.6.7.1.5, "Creating an HTML Parameter Form header using the Property Inspector" to use the Property Inspector, or Section 3.6.7.2.5, "Creating an HTML Parameter Form header using PL/SQL").
- create a Parameter Form footer (an after form escape) for placing a logo or some standard links in the footer of the HTML Parameter Form (see Section 3.6.7.1.6, "Creating an HTML Parameter Form footer using the Property Inspector" to use the Property Inspector, or Section 3.6.7.2.6, "Creating an HTML Parameter Form footer using PL/SQL").

You can access the Parameter Form Builder from the Object Navigator or by choosing **Tools > Parameter Form Builder**.

See also

Section 1.2.2, "About Web reports"

Section 1.9.4, "About Parameter Forms for Web reports"

Section 2.2.3, "About Web links for HTML output"

2.6 PL/SQL

The topics in this section discuss the use of PL/SQL in Reports Builder.

- About the PL/SQL Editor
- About the Stored PL/SQL Editor
- About the Syntax Palette
- About program units
- About stored program units
- About stored program units
- About external PL/SQL libraries
- About attached libraries
- About formulas
- About group filters
- About ref cursor queries
- About built-in packages
- About triggers

2.6.1 About the PL/SQL Editor

The PL/SQL Editor enables you to create and edit PL/SQL program units.

Usage notes

When you make changes to a program unit, dependent program units lose their compiled status, which is indicated by an asterisk (*) after their name under the **Program Units** node in the Object Navigator. You can navigate to those program units directly in the PL/SQL Editor using the **Name** drop-down list to recompile them.

Restrictions

- If you delete a PL/SQL package, function, or procedure, you must also delete all references to it in your report. Otherwise, you will get an error when you compile, generate, or run the report.

- PL/SQL package, function, and procedure names must be unique within the report and may not duplicate the names of any columns, groups, queries, or printable objects.

See also

Section 3.13.3.1, "Creating a local program unit"

2.6.2 About the Stored PL/SQL Editor

The Stored PL/SQL Editor enables you to create and edit stored PL/SQL program units in a database (listed under the **Database Objects** node in the Object Navigator).

See also

Section 3.13.3.2, "Creating a stored program unit"

2.6.3 About the Syntax Palette

The Syntax Palette is a programming tool that enables you to display and copy the constructs of PL/SQL language elements and built-in packages into the PL/SQL Editor and Stored PL/SQL Editor.

See also

Section 3.13.2.4, "Inserting syntax into the PL/SQL Editor"

2.6.4 About program units

Program units are packages, functions, or procedures that you can reference from any PL/SQL within the current report.

Note: Program units cannot be referenced from other documents. If you want to create a package, function, or procedure that can be referenced from multiple documents, create an external PL/SQL library (see Section 3.13.5.1, "Creating an external PL/SQL library").

For a detailed example of using PL/SQL in a report, see Chapter 37, "Building a PL/SQL Report".

Restrictions

- If you delete a PL/SQL package, function, or procedure, you must also delete all references to it in your report. Otherwise, you will get an error when you compile, generate, or run the report.
- PL/SQL package, function, and procedure names must be unique within the report and may not duplicate the names of any columns, groups, queries, or printable objects.

Example: Referencing a PL/SQL function in formulas

Suppose that you have a report with the following groups and columns:

```
Groups    Columns         Summary
-----------------------------------------
RGN       REGION
          RGNSUMSAL       SUM(DEPTSUMSAL)
          COSTOFLIVING

DEPT      DNAME
          DEPTNO
          DEPTSUMSAL      SUM(EMP.SAL)

JOB       JOB
          HEADCOUNT       COUNT(EMP.EMPNO)

EMP       ENAME
          EMPNO
          SAL
```

```
        COMM
```

Given these groups and columns, you might create multiple formulas that apply the cost of living factor (COSTOFLIVING) to salaries. To avoid duplication of effort, you could create the following PL/SQL function and reference it from the formulas:

```
function CompSal(salary number) return number is
begin
  return (salary*CostofLiving);
end;
```

Following are some examples of how you might reference the PL/SQL function in formulas:

CompSal(:RGNSUMSAL)

or

CompSal(:SAL) + COMM

See also

Section 3.13.3.1, "Creating a local program unit"

2.6.5 About stored program units

Stored program units (also known as stored subprograms, or stored procedures) can be compiled separately and stored permanently in an Oracle database, ready to be executed. Once compiled and stored in the data dictionary, they are schema objects, which can be referenced by any number of applications connected to that database.

Stored program units offer higher productivity, better performance, memory savings, application integrity, and tighter security. For example, by designing applications around a library of stored procedures and functions, you can avoid redundant coding and increase your productivity.

Stored program units are stored in parsed, compiled form. So, when called, they are loaded and passed to the PL/SQL engine immediately. Also, they take advantage of shared memory. So, only one copy of a program unit need be loaded into memory for execution by multiple users.

Because stored program units run in ORACLE, they can perform database operations more quickly than PL/SQL that is local to your report. Therefore, in general, use stored program units for PL/SQL that performs database operations. Use local program units for PL/SQL that does not involve database operations.

However, if you are on a heavily loaded network with very slow response time, using stored program units may not be faster for database operations. Similarly, if your server is significantly faster than your local machine, then using local program units may not be faster for non-database operations.w

See also

Section 3.13.3.2, "Creating a stored program unit"

2.6.6 About external PL/SQL libraries

External PL/SQL libraries are collections of PL/SQL procedures, functions, and packages that are independent of a report definition. By attaching an external library to a report, you can reference its contents any number of times. For example, you could reference a procedure in an attached library from both a Before Report trigger and a format trigger. This eliminates the need to re-enter the same PL/SQL for each application.

When you associate an external PL/SQL library with a report or another external library, it is called an *attached library*.

See also

Section 3.13.5.1, "Creating an external PL/SQL library"

2.6.7 About attached libraries

Attached libraries are external PL/SQL libraries that you have associated with a report or another external library. When an external library is attached, you can reference its packages, functions, and procedures from within your report. For example, if you attached an external library name MYLIB to your report and it contained a function named ADDXY, then you could reference ADDXY from any PL/SQL in the report.

External PL/SQL libraries are independent of a report definition

Usage notes

Local PL/SQL executes more quickly than a reference to a procedure or function in an external PL/SQL library. As a result, you should only use external PL/SQL libraries when the benefits of sharing the code across many applications outweigh the performance overhead.

Restrictions

- If Reports Builder cannot find a library that you specify in the Attached Libraries list, a warning will be raised when you accept the dialog box, save the report, or open the report. If you try to run the report or compile the PL/SQL in it, an error will be raised.
- The Attached Libraries list is saved. The next time you open the report or library the list will have the same contents it did when you last saved the report.
- If an external library references another library, you must attach both libraries to the report even if the first library already has the second one attached.

See also

Section 3.13.5.5, "Attaching a PL/SQL library"

2.6.8 About formulas

Formulas are PL/SQL functions that populate formula or placeholder columns. You can access the PL/SQL for formulas from the Object Navigator, the PL/SQL Editor, or the Property Inspector (i.e., the PL/SQL Formula property).

A column of datatype Number can only have a formula that returns a value of datatype NUMBER. A column of Datatype Date can only have a formula that returns a value of datatype DATE. A column of Datatype Character can only have a formula that returns a value of datatype CHARACTER, VARCHAR, or VARCHAR2.

Restrictions

- You can read and assign values to a column in a formula, if the column is a placeholder or parameter column; you cannot change the value of database columns (values retrieved from the database). For example, you can use the value of a column called COMP in a condition (e.g., IF :COMP = 10) and you can directly set its value in an assignment statement (e.g.,:COMP:= 15).
- A formula can only make reference to columns that are in the same or a higher group in the group hierarchy. For example, a formula for a report-level column can only reference other report-level columns.
- Formulas are calculated such that any column referenced in the formula will be calculated first. To do so, Reports Builder builds a dependency list, to guarantee proper ordering of calculations. Note that circular dependencies, in which a

column references another column which in turn references the first column, either directly or indirectly, are not allowed.

- When using SRW.DO_SQL, we recommend that you do not read database values that are updated or inserted in the same report. There is no guarantee of the exact time Reports Builder will fetch records from the database for formatting the output. Reports Builder does internal "data look-ahead" to optimize performance. Thus, a particular record might already have been accessed before an update is issued to the same record. Reports Builder builds internal dependency lists which guarantee that events, such as invocation of user exits, calculation of summaries, etc., happen in the correct order. However, Reports Builder cannot guarantee these events will be synchronized with its internal data access or with the formatting of data.

Examples

Example 1: Adding values

The following example populates the column with the value of the salary plus the commission.

```
function salcomm return NUMBER is
begin
  return(:sal + :comm);
end;
```

Example 2: Using conditions

The following code adds the commission to the salary if the value for the commission is not null.

```
function calcomm return NUMBER is
temp number;
begin
  if :comm IS NOT NULL then
    temp := :sal + :comm;
  else
    temp := :sal;
  end if;
  return (temp);
end;
```

See also

Section 2.3.2, "About formula columns"

Section 3.13.4.3, "Creating or editing a formula column"

Section 3.13.4.4, "Creating a placeholder column"

2.6.9 About group filters

A group filter determines which records to include in a group. You can use the packaged filters, First and Last, to display the first n or last n records for the group, or you can create your own filters using PL/SQL. You can access group filters from the Object Navigator, the Property Inspector (the PL/SQL Filter property), or the PL/SQL Editor.

The function must return a boolean value (TRUE or FALSE). Depending on whether the function returns TRUE or FALSE, the current record is included or excluded from the report.

Difference between group filters and Maximum Rows to Fetch property

The Maximum Rows to Fetch property restricts the actual number of records fetched by the query. A group filter determines which records to include or exclude, after all the records have been fetched by the query. Since Maximum Rows to Fetch actually restricts the amount of data retrieved, it is faster than a group filter in most cases. If you use a Filter of Last or Conditional, Reports Builder must retrieve all of the records in the group before applying the filter. Also, you should be aware that when using Maximum Rows to Fetch for queries. It can effect summaries in other groups which depend on this query. For example, if you set Maximum Rows to Fetch to 8 any summaries based on that query will only use the 8 records retrieved.

Restrictions

Group filters cannot be added to groups if Filter Type is First or **Last**.

- Group filters cannot be added to cross-product groups.
- The function that you enter for a group filter can only depend upon the following columns:
  - a database column owned by the group's query or a query above it in the data model hierarchy
  - computed columns (formulas or summaries) that depend on unrelated queries (i.e., computed columns that do not depend upon columns in the group, the group's ancestors, or the group's descendants)

- In a group filter, you can read the values of Reports Builder columns and parameters of the correct frequency, but you cannot directly set their values. For example, you can use the value of a parameter called COUNT1 in a condition (e.g., IF :COUNT1 = 10), but you cannot directly set its value in an assignment statement (e.g.,:COUNT1:= 10). Note also that the use of PL/SQL global variables to indirectly set the values of columns or parameters is not supported. If you do this, you may get unpredictable results. You also cannot reference any page-dependent columns (i.e., Reset At of Page) or columns that rely on page-dependent columns in a group filter.

Example

```
function filter_comm return boolean is
begin
  if :comm IS NOT NULL then
    if :comm < 100 then
      return (FALSE);
    else
      return (TRUE);
    end if;
  else
    return (FALSE); -- for rows with NULL commissions
  end if;
end;
```

See also

Section 3.13.4.2, "Creating or editing a group filter"

2.6.10 About ref cursor queries

A ref cursor query uses PL/SQL to fetch data. Each ref cursor query is associated with a PL/SQL function that returns a cursor value from a cursor variable. The function must ensure that the ref cursor is opened and associated with a SELECT statement that has a SELECT list that matches the type of the ref cursor.

Usage notes

- When you make a ref cursor query the child in a data link, the link can only be a group to group link. It cannot be a column to column link.
- If you use a stored program unit to implement ref cursors, you receive the added benefits that go along with storing your program units in the Oracle database.

- You base a query on a ref cursor when you want to:
  1. more easily administer SQL.
  2. avoid the use of lexical parameters in your reports.
  3. share data sources with other applications, such as Form Builder.
  4. increase control and security.
  5. encapsulate logic within a subprogram.

Furthermore, if you use a stored program unit to implement ref cursors, you receive the added benefits that go along with storing your program units in the Oracle database.

For more information about ref cursors and stored subprograms, refer to the PL/SQL User's Guide and Reference.

Examples

Example 1: Package with ref cursor example

```
/* This package spec defines a ref cursor
** type that could be referenced from a
** ref cursor query function.
** If creating this spec as a stored
** procedure in a tool such as SQL*Plus,
** you would need to use the CREATE
** PACKAGE command.
*/

PACKAGE cv IS
type comp_rec is RECORD
  (deptno number,
   ename varchar(10),
   compensation number);
type comp_cv is REF CURSOR return comp_rec;
END;
```

Example 2: Package with ref cursor and function

```
/* This package spec and body define a ref
** cursor type as well as a function that
** uses the ref cursor to return data.
** The function could be referenced from
** the ref cursor query, which would
** greatly simplify the PL/SQL in the
** query itself. If creating this spec
** and body as a stored procedure in a
** tool such as SQL*Plus, you would need
** to use the CREATE PACKAGE and CREATE
** PACKAGE BODY commands.
*/

PACKAGE cv IS
type comp_rec is RECORD
  (deptno number,
   ename varchar(10),
   compensation number);
type comp_cv is REF CURSOR return comp_rec;
function emprefc(deptno1 number) return comp_cv;
END;

PACKAGE BODY cv IS
function emprefc(deptno1 number) return comp_cv is
  temp_cv cv.comp_cv;
begin
  if deptno1 > 20 then
    open temp_cv for select deptno, ename,
    1.25*(sal+nvl(comm,0)) compensation
    from emp where deptno = deptno1;
  else
    open temp_cv for select deptno, ename,
    1.15*(sal+nvl(comm,0)) compensation
    from emp where deptno = deptno1;
  end if;
  return temp_cv;
end;
END;
```

Example 3: Ref cursor

```
empCur rcPackage.empCurType;
BEGIN
  OPEN empCur FOR SELECT * FROM emp;
  RETURN empCur;
END;

/* Note, rcPackage is a local program unit defined as: */

PACKAGE rcPackage IS
  TYPE empCurType IS REF CURSOR RETURN emp%ROWTYPE;
END;
```

Example 4: Ref cursor query

```
/* This ref cursor query function would be coded
** in the query itself. It uses the cv.comp_cv
** ref cursor from the cv package to return
** data for the query.
*/
function DS_3RefCurDS return cv.comp_cv is
  temp_cv cv.comp_cv;
begin
  if :deptno > 20 then
    open temp_cv for select deptno, ename,
    1.25*(sal+nvl(comm,0)) compensation
    from emp where deptno = :deptno;
  else
   open temp_cv for select deptno, ename,
   1.15*(sal+nvl(comm,0)) compensation
   from emp where deptno = :deptno;
  end if;
  return temp_cv;
end;
```

Example 5: Ref cursor query calling function

```
/* This ref cursor query function would be coded
** in the query itself. It uses the cv.comp_cv
** ref cursor and the cv.emprefc function from
** the cv package to return data for the query.
** Because it uses the function from the cv
** package, the logic for the query resides
** mainly within the package. Query
** administration/maintenance can be
** done at the package level (e.g.,
** modifying SELECT clauses could be done
** by updating the package). You could also
** easily move the package to the database.
** Note this example assumes you have defined
** a user parameter named deptno.
*/

function DS_3RefCurDS return cv.comp_cv is
  temp_cv cv.comp_cv;
begin
  temp_cv := cv.emprefc(:deptno);
  return temp_cv;
end;
```

See also

Section 3.8.1.9, "Creating a query: Ref Cursor Query tool"

2.6.11 About built-in packages

A built-in package is a group of logically related PL/SQL types, objects, and functions or procedures. It generally consists of two parts: the package spec (including data declarations) and the package body. Packages are especially useful because they allow you to create global variables.

Oracle provides several packaged procedures that you can use when building or debugging your PL/SQL-based applications. Your PL/SQL code can make use of the procedures, functions, and exceptions in the Reports Builder built-in package (SRW), and numerous Tools built-in packages, as described below.

2.6.11.1 About the Reports Builder built-in package (SRW)

Reports Builder is shipped with a built-in package (SRW), a collection of PL/SQL constructs that include many functions, procedures, and exceptions you can reference in any of your libraries or reports.

The PL/SQL provided by the SRW package enables you to perform such actions as change the formatting of fields, run reports from within other reports, create customized messages to display in the event of report error, and execute SQL statements.

You can reference the contents of the SRW package from any of your libraries or reports without having to attach it. However, you cannot reference its contents from within another product, e.g., from SQL*Plus.

Constructs found in a package are commonly referred to as "packaged"; i.e., packaged functions, packaged procedures, and packaged exceptions.

See also

Topic "SRW built-in package" in the **Reference** section of the *Reports Builder online help*

2.6.11.2 About the Tools built-in packages

Several client-side built-in packages are provided that contain many PL/SQL constructs you can reference while building applications or debugging your application code. These built-in packages are not installed as extensions to the package STANDARD. As a result, any time you reference a construct in one of the packages, you must prefix it with the package name (e.g., TEXT_IO.PUT_LINE).

The Tools built-in packages are:

- **DDE**

  Provides Dynamic Data Exchange support within Reports Builder components.

- **DEBUG**

  Provides procedures, functions, and exceptions for when debugging your PL/SQL program units. Use these built-in subprograms to create debug triggers and set breakpoints with triggers.

- **EXEC_SQL**

  Provides procedures and functions for executing dynamic SQL within PL/SQL code written for Reports Builder applications.

- **LIST**

Provides procedures, functions, and exceptions you can use to create and maintain lists of character strings (VARCHAR2). This provides a means of creating arrays in PL/SQL Version 1.

- **ORA_FFI**

  Provides a foreign function interface for invoking C functions in a dynamic library.

- **ORA_JAVA**

  Provides an interface for invoking Java classes from PL/SQL.

- **ORA_NLS**

  Enables you to extract high-level information about your current language environment. This information can be used to inspect attributes of the language, enabling you to customize your applications to use local date and number format. Information about character set collation and the character set in general can also be obtained. Facilities are also provided for retrieving the name of the current language and character set, allowing you to create applications that test for and take advantage of special cases.

- **ORA_PROF**

  Provides procedures, functions, and exceptions you can use for tuning your PL/SQL program units (e.g. examining how much time a specific piece of code takes to run).

- **TEXT_IO**

  Provides constructs that allow you to read and write information from and to files. There are several procedures and functions available in Text_IO, falling into the following categories:

  - **file operations**. The FILE_TYPE record, the FOPEN and IS_OPEN functions, and the FCLOSE procedure enable you to define FILE_TYPE variables, open files, check for open files, and close open files, respectively.
  - **output (write) operations**. The PUT, PUTF, PUT_LINE, and NEW_LINE procedures enable you to write information to an open file or output it to the PL/SQL Interpreter.
  - **input (read) operations**. The GET_LINE procedure enables you to read a line from an open file

- **TOOL_ENV**

  Allows you to interact with Oracle environment variables by retrieving their values for use in subprograms.

- **TOOL_ERR**

  Allows you to access and manipulate the error stack created by other built-in packages such as DEBUG.

  In addition to using exceptions to signal errors, some built-in packages (e.g., the DEBUG package) provide additional error information. This information is maintained in the form of an "error stack".

  The error stack contains detailed error codes and associated error messages. Errors on the stack are indexed from zero (oldest) to n-1 (newest), where n is the number of errors currently on the stack. Using the services provided by the TOOL_ERR package, you can access and manipulate the error stack.

- **TOOL_RES**

  Provides a means of extracting string resources from a resource file with the goal of making PL/SQL code more portable by isolating all textual data in the resource file.

The following packages are used only internally by Oracle Reports. There are no subprograms available for external use with these packages.

- **ORA_DE**

  Contains constructs used by Reports for private PL/SQL services.

- **STPROC**

  Calls subprograms stored in the database. Calls to this package are automatically generated.

- **JNI**

  Facilitates calling Java from PL/SQL.

See also

Topics for each of the Tools built-in packages under in the **Reference > PL/SQL Reference > Built-in Packages** section of the *Reports Builder online help*

2.6.12 About triggers

Triggers check for an event. When the event occurs they run the PL/SQL code associated with the trigger.

Report triggers are activated in response to report events such as the report opening and closing rather that the data that is contained in the report. They are activated in a predefined order for all reports.

Format triggers are executed before an object is formatted. A format trigger can be used to dynamically change the formatting attributes of the object.

Validation triggers are PL/SQL functions that are executed when parameter values are specified on the command line and when you accept the Runtime Parameter Form.

Database triggers are procedures that are stored in the database and implicitly executed when a triggering statement such as INSERT, UPDATE, or DELETE is issued against the associated table.

2.6.12.1 About report triggers

Report triggers execute PL/SQL functions at specific times during the execution and formatting of your report. Using the conditional processing capabilities of PL/SQL for these triggers, you can do things such as customize the formatting of your report, perform initialization tasks, and access the database. To create or modify a report trigger, use Report Triggers in the Object Navigator. Report triggers must explicitly return TRUE or FALSE. Reports Builder has five global report triggers (you cannot create new global report triggers):

- *Before Report trigger*
- *After Report trigger*
- *Between Pages trigger*
- *Before Parameter Form trigger*
- *After Parameter Form trigger*

For information about these triggers, see the **Reference** section of the *Reports Builder online help*.

Usage notes

The Before Report trigger fires before the report is executed but after the queries are parsed.

You can think of order in this way:

1. Queries are parsed.
2. Before Report trigger fires.
3. Report is executed (i.e., fetch data+format output.)

Consistency is guaranteed if you use DML, DDL in (or before) the After Form Trigger. However, consistency is not guaranteed in the Before Report trigger, since Reports Builder may have to start work on data cursors before that trigger based on the definition of the report. Before the Before Report trigger, Reports Builder describes the tables involved and opens cursors. Any change to the tables after that will not be seen by the report.

See also

Section 3.13.3.5, "Creating a report trigger"

Section 3.13.3.6, "Deleting a report trigger"

2.6.12.2 About format triggers

A format trigger is a PL/SQL function executed before an object is formatted. A trigger can be used to dynamically change the formatting attributes of the object. For example, you can use a format trigger to cause a value to display in bold if it is less than zero. Another example is to use a format trigger to use scientific notation for a field if its value is greater than 1,000,000.

A format trigger can fire multiple times for a given object, whenever Reports Builder attempts to format the object. Consider the case where Reports Builder starts to format the object at the bottom of a page. If the object does not fit on the page, Reports Builder stops formatting and reformats on the following page. In this case, the format trigger will fire twice. It is therefore not advisable to do any kind of "persistence" operation, such as logging, in this trigger.

The Reports Builder SRW built-in package contains PL/SQL procedures with which you can quickly change the format attributes of an object. These include procedures to:

- change the border pattern and color of an object
- change the interior pattern and color of an object

- change the font size, style, weight, spacing, and justification of a field or boilerplate text
- change the format mask of a field
- access a field's value

Examples

See the topic "Format trigger" in the **Reference** section of the *Reports Builder online help*.

See also

Section 3.13.4.1, "Creating or editing a format trigger"

2.6.12.3 About validation triggers

Validation triggers are PL/SQL functions that are executed when parameter values are specified on the command line and when you accept the Runtime Parameter Form.

> **Note:** For JSP-based Web reports, the Runtime Parameter Form displays when you run a report in Reports Builder, but does not display in the runtime environment. If parameters are not specified on the Runtime Parameter Form, the validation trigger returns false and generates error message `rep-546 Invalid Parameter Input error`. Thus, you need to provide the parameters in an alternate way, as described in Section 1.9.4, "About Parameter Forms for Web reports".

Validation triggers are also used to validate the Initial Value property of the parameter. Depending on whether the function returns TRUE or FALSE, the user is returned to the Runtime Parameter Form.

Example

See the topic "Validation trigger" in the **Reference** section of the *Reports Builder online help*.

See also

Section 3.11.4, "Validating a parameter value at runtime"

2.6.12.4 About database triggers

Database triggers are procedures that are stored in the database and implicitly executed when a triggering statement such as INSERT, UPDATE, or DELETE is issued against the associated table. Triggers can be defined only on tables, not on views. However, triggers on the base table of a view are fired if an INSERT, UPDATE, or DELETE statement is issued against a view.

A trigger can include SQL and PL/SQL statements that execute as a unit, and can invoke other stored procedures. Use triggers only when necessary. Excessive use of triggers can result in cascading or recursive triggers. For example, when a trigger is fired, a SQL statement in the trigger body potentially can fire other triggers.

By using database triggers, you can enforce complex business rules and ensure that all applications behave in a uniform manner. Use the following guidelines when creating triggers:

- Use triggers to guarantee that when a specific operation is performed, related actions are performed.
- Use database triggers only for centralized, global operations that should be fired for the triggering statement, regardless of which user or database application issues the statement.
- Do not define triggers that duplicate the functionality already built into Oracle. For example, do not define triggers to enforce data integrity rules that can be easily enforced using declarative integrity constraints.
- Limit the size of triggers (60 lines or fewer is a good guideline). If the logic for your trigger requires much more than 60 lines of PL/SQL code, it is better to include most of the code in a stored procedure, and call the procedure from the trigger.
- Be careful not to create recursive triggers. For example, creating an AFTER UPDATE statement trigger on the EMP table that itself issues an UPDATE statement on EMP causes the trigger to fire recursively until it has run out of memory.

For additional information about how triggers are used in applications, see the *Oracle9i Application Developer's Guide*. See the *Oracle9i Concepts Manual* for more information about the different types of triggers.

See also

Section 3.13.3.7, "Creating a database trigger"

2.7 Templates

The topics in this section discuss the use of templates in Reports Builder.

- About templates
- About template attributes
- About applying templates
- About inheritance in templates
- About the Template Editor

2.7.1 About templates

Templates define common characteristics and objects that you want to apply to multiple paper-based reports. For example, you can define a template that includes the company logo and sets fonts and colors for selected areas of a report.

When you use the Report Wizard to create a paper-based report, you use the Templates page of the wizard to apply a template (.tdf file) to the report. The Templates page lists the default templates, as well as any templates that you have created.

Note: If the list of templates does not appear, make sure that the REPORTS_PATH environment variable includes the location of the templates (e.g., *ORACLE_HOME*\reports\templates).

When you choose a template, objects in the margin area of a template are imported into the same locations in the current report section, overwriting any existing objects. The characteristics (formatting, fonts, colors, etc.) of objects in the body area of the template are applied to objects in the body area of the current report section. Any template properties, parameters, report triggers, program units, and attached libraries you have defined are also applied. You can apply different templates to each section of the report. However, if you are applying one of the default templates, you cannot combine two report blocks that use different default templates in a single report. All of your report blocks in any one report must use the same default template.

If you later apply another template to a report, the existing template objects will be deleted in the current report section.

See also

Section 3.12, "Define a Template"

2.7.2 About template attributes

In the Layout Body area of a template, you can define Default and Override attributes under the following Object Navigator nodes:

- **Frames**, which contains the following:
  - a **Section Frame** node, which defines attributes for the parent frame surrounding the currently selected section.
  - a **Headings Frame** node, which defines attributes for the parent frame surrounding the column headings.
  - a **Fields Frame** node, which defines attributes for the parent frame surrounding the fields.
  - a **Summaries Frame** node, which defines attributes for the parent frame surrounding the summaries (totals).
- **Field Labels/Headings**, which contains the following:
  - a **Character** node, which defines attributes for the labels or column headings of character fields.
  - a **Number** node, which defines attributes for the labels or column headings of number fields.
  - a **Date** node, which defines attributes for the labels or column headings of date fields.
- **Fields**, which contains the following:
  - a **Character** node, which defines attributes for character fields.
  - a **Number** node, which defines attributes for number fields.
  - a **Date** node, which defines attributes for date fields.
- **Summary Labels**, which contains the following:
  - a **Character** node, which defines attributes for the labels of summaries on character fields.
  - a **Number** node, which defines attributes for the labels of summaries on numeric fields.

- a **Date** node, which defines attributes for the labels of summaries on date fields.

- **Summaries**, which contains the following:
  - a **Character** node, which defines attributes for summaries on character fields.
  - a **Number** node, which defines attributes for summaries on number fields.
  - a **Date** node, which defines attributes for summaries on date fields.

Default attributes

The **Default** node in the Object Navigator defines the default visual attributes (formatting, fonts, colors, etc.) for all report styles. If you want to define attributes for individual report styles, you do so under the **Override** node. When you apply a template to a report, all Default attributes are applied to the report, except for attributes that are localized under the **Override** node.

Override attributes

Under the **Override** node in the Object Navigator, you can define attributes for individual report styles. Each report style contains one or more sections that map to groups in the report:

Single-section report styles: Tabular, Form, Mailing Label, Form Letter

Multiple section report styles: Group Left, Group Above, Matrix, Matrix with Group

For the report styles that support multiple groups, you can create additional sections as needed. Sections are mapped to groups as follows:

Same number of groups as sections: one-to-one mapping (the first section is mapped to the first group, the second section to the second group, and so on).

More groups than sections: one-to-one mapping until the next-to-last section. Then, all subsequent groups are mapped to the next-to-last section, and the last group is mapped to the last section. If only one section exists, all groups are mapped to it.

More sections than groups: one-to-one mapping until the next-to-last group. Then, the last group is mapped to the last section.

See also

Section 3.12.2, "Defining default template attributes"

Section 3.12.3, "Defining override template attributes"

2.7.3 About applying templates

When you apply a template to a report, all of the following objects, properties, and attributes from the template are applied to the current report section:

- parameters and their validation triggers
- physical page size
- logical page size
- character/bitmap mode
- margin position
- panel print order
- Maximum Horizontal Body Pages property
- Maximum Vertical Body Pages property

In addition, all of the layout objects in the margin of the template are copied into the same location in the current report section.

Usage notes

- The template will be applied to the current section of the report (the section displayed in the Paper Layout view) by default. If no section is displayed, the template will be applied to the Main Section of the report by default. If you select the node for the entire report in the Object Navigator, the template will be applied to the Main Section of the report by default. To apply the template to a specific section of your report, select the node for that section in the Object Navigator.
- You can apply different templates to each section of the report. However, if you are applying one of the default templates, you cannot combine two report blocks that use different default templates in a single report. All of your report blocks in any one report must use the same default template.

See also

Section 3.12.4, "Applying a template to a report"

2.7.4 About inheritance in templates

In templates, Sections, Frames, Fields, Labels, Headings, and Summaries properties all may inherit their values.

Default properties

The Default properties inherit the values preset by Reports Builder. When a property is set to its default Reports Builder value, the icon next to it in the Property Inspector is a small circle. Default properties become localized when you change their values, or select them and click the Localize Property button in the toolbar. When a property is localized, the icon next to it changes to a square. To return the properties to their inherited values, select the properties and click the Inherit Property button in the toolbar.

Override properties

The properties of Override Sections inherit their values from the Default properties. When a property inherits from a Default property, the icon next to it in the Property Inspector is an arrow. Properties in the Override Sections become localized when you change their values, or select them and click the Localize Property button in the toolbar. When a property is localized, the icon next to it changes to an arrow with a red cross through it. To return the values of properties in the Override Sections to their inherited values, select the properties and click the Inherit Property button in the toolbar.

2.7.5 About the Template Editor

The Template Editor is a work area in which you can define objects and formatting properties for your templates. It is similar to the Paper Layout view of the Report Editor. You can create, delete, and modify objects (e.g., page numbers, text, and graphics) in the margin area. You cannot create and delete objects in the body area, but you can modify the properties of body objects in the Property Inspector (**Tools > Property Inspector**).

The **Report Style** drop-down list allows you to view the layout for a selected report style. To define default settings for all report styles, you can choose Default from the **Report Style** drop-down list. To make changes for an individual report style, you can select that report style from the **Report Style** drop-down list to specify settings that override the default.

Access

You can access the Template Editor in the following ways:

When creating a new template:

- Choose **File > New > Template**.
- In the Object Navigator, click the **Templates** node, then click the Create button in the toolbar.

When displaying an existing template:

- From the **Window** menu, choose a window displaying Template Editor Paper Layout view.
- In the Object Navigator, double-click the view icon next to the **Paper Layout** node for a template.

2.8 Output Formats and Capabilities

The topics in this section discuss the various output formats and capabilities in Reports Builder.

- About format order
- About batch reporting
- About report distribution
- About pluggable destinations
- About event-driven publishing
- About switching the printer tray
- About XML in reports
- About HTML output
- About PDF output
- About RTF output
- About delimited output
- About text output
- About creating an ASCII (character-mode) report

2.8.1 About format order

In prior releases, Oracle Reports formatted the sections of a report in sequential order: Header section, followed by Main section, followed by Trailer section. This release introduces the capability to change the order in which the three sections of a report are formatted.

> **Note:** Regardless of the order in which the report sections are formatted, the output order is unchanged: Header section first, then Main section, then Trailer section.

The format order can be set in either of the following ways:

- Format Order of Sections property
- SRW.SET_FORMAT_ORDER built-in procedure (if defined, overrides the Format Order of Sections property setting)

This feature is useful for formatting any report section first to retrieve information that is known only at the time of formatting, such as page numbers, then using that information in the formatting of a previous section.

For example, to create a table of contents (TOC) for a report, you can format the Main section first and use report triggers to build a table containing the TOC entries. When the first element for the TOC is formatted, a trigger fires and creates a row in the TOC table containing the TOC entry and the page number. After the Main section has completed formatting, the format order setting can define that the Header section is formatted next. The Header section can contain a report block based on the TOC table. After formatting, you can output your report with a TOC (the Header section), followed by the report body (the Main section), followed by the Trailer section.

For the steps to create a TOC for a report, see the example reports in Chapter 34, "Building a Paper Report with a Simple Table of Contents and Index" and Chapter 35, "Building a Paper Report with a Multilevel Table of Contents".

A note about page numbering:

The page numbering of a report follows the format order. For example, in a report with a Header section of 2 pages, a Main section of 8 pages, and a Trailer section of 3 pages, with Format Order set to Main-Trailer-Header, the page numbering will be as follows in the report output: 12, 13 (Header pages, which were formatted last), 1, 2, 3, 4, 5, 6, 7, 8, (Main pages, which were formatted first) 9, 10, 11(Trailer pages, which were formatted second).

2.8.2 About batch reporting

If you do not need to examine report output in the Previewer (e.g., you may have to generate large volumes of output from a fully-tested report or run several reports in succession), you can run your report in batch using `rwrun`. This leaves you free to pursue other tasks while your reports are running.

You can run reports in batch mode from the command line, or use a command file to specify arguments. A command file can save you a great deal of typing if you wish to run several reports using the same set of arguments.

You can also use the Reports Server to batch process reports by specifying BACKGROUND=YES on the command line (valid for `rwclient`, `rwcgi`, or `rwservlet`) to run reports asynchronously (the client sends the call to the server, then continues with other processes without waiting for the report job to complete; if the client process is killed, the job is canceled).

See also

Section 3.7.2, "Running a report from the command line"

Section 3.7.3, "Running a report using a command file"

The **Reference > Command Line** section of the *Reports Builder online help* (for information about BATCH and BACKGROUND)

2.8.3 About report distribution

Report distribution enables you to design a report that can generate multiple output formats and be distributed to multiple destinations from a single run of the report. You can create distributions for an entire report, and for individual sections of the report. For example, in a single run of a report, you can generate HTML output, send a PostScript version to the printer, and also e-mail any or all sections of the report to a distribution list.

You can define the distribution for a report in any of the following ways:

- using XML, as described in the chapter "Creating Advanced Distributions" in *Oracle Application Server Reports Services Publishing Reports to the Web*, available on the Oracle Technology Network Oracle Reports Documentation page (http://otn.oracle.com/docs/products/reports/content.html).
- in the Distribution dialog box to define a distribution list for the entire report and/or any report section.

  Note: To display the Distribution dialog box: In the report or section Property Inspector, under the Report node, click the Distribution property value field

- on the command line, via the DESTINATION keyword that specifies a DST file or XML file.

  Note: The DST file method is supported for backward compatibility; the preferred and recommended method of distributing reports is via XML or the Distribution dialog box.

For detailed information about when you'd use the report-level vs. section-level distribution, see Chapter 36, "Bursting and Distributing a Report". This chapter also

covers distribution based on a repeating section, then e-mail those sections based on the distribution.xml.

To distribute a report, you first *define* the distribution, then *enable* the distribution, as described in Section 3.7.11, "Distributing a report to multiple destinations".

Usage notes

- You cannot mix character mode and bit-mapped output in one report. The MODE parameter can only be set to one value per the entire report (DEFAULT, BITMAP, or CHARACTER). As a result, you cannot have both within one report.

2.8.3.1 About the DST file

As an alternative to defining the distribution for a report or report section in the Distribution dialog box, you can also create a DST file and specify its name on the command line via the DESTINATION keyword to distribute the report.

> **Note:** This method is supported for backward compatibility; the preferred and recommended method of distributing reports is via the Distribution dialog box or using XML, as described in the chapter "Creating Advanced Distributions" in the *Oracle Application Server Reports Services Publishing Reports to the Web* manual, available on the Oracle Technology Network Oracle Reports Documentation page (http://otn.oracle.com/docs/products/reports/content.html).

If a DST file is specified on the command line, the distribution that it defines overrides the distribution defined using the Distribution dialog box.

> **Note:** If you trace report distribution to identify distribution errors (see Section 3.14.19, "Tracing report distribution"), the trace file format is very similar to the DST file, so you can cut and paste to generate a DST file from the trace file.

The format of each line of a DST file is as follows:

```
dist_ID: output_def
```

where

dist_ID is an identifier for a distribution destination.

output_def is a series of `rwrun` or `rwclient` command line keywords that specify the distribution definition. In addition, the following parameter is valid:

LEVEL specifies the scope of the distribution.

Values for LEVEL

`REPORT` means that the distribution applies to the entire report.

`Header_Section` means that the distribution applies to the header section only.

`Main_Section` means that the distribution applies to the main (body) section only.

`Trailer_Section` means that the distribution applies to the trailer section only.

Default

`REPORT`

Example

The definition in this example sends report output to an HTML file, 3 copies of the main section to a printer, and the header section to a PDF file.

```
;dst file (specified via the DESTINATION keyword on the command line)
DEST1: DESNAME=testdst1.HTM DESTYPE=file
DESFORMAT=HTML COPIES=1 LEVEL=Report
DEST2: DESNAME=\\luna\2op813 DESTYPE=printer
DESFORMAT=BITMAP COPIES=3
LEVEL=Main_Section
DEST3: DESNAME=SECTION1.pdf DESTYPE=file
DESFORMAT=pdf COPIES=1
LEVEL=Header_Section
```

See also

Section 2.1.2, "About report sectioning and sections"

2.8.4 About pluggable destinations

Pluggable destinations can be used to distribute any content that an engine (not only the Oracle Reports engine) has created in the Reports Server's cache. Oracle Reports provides several out-of-the-box destinations:

- Web
- printer
- e-mail
- file
- OracleAS Portal

You can also define access to your own custom destination by using the Oracle Reports Java APIs to implement a new destination component in the Reports

Server. You can choose for your jobs to use an out-of-the-box destination or your customized destination to determine the destination for the output in the cache.

For information and steps to implement and register a destination class, then use the destination with Oracle Reports, see the FTP Destination tutorial available via the Oracle Reports Plugin Exchange on OTN (http://otn.oracle.com/products/reports/pluginxchange/content.html).

See also

"Configuring Destinations for OracleAS Reports Services" in the *Oracle Application Server Reports Services Publishing Reports to the Web* manual, available on the Oracle Technology Network Oracle Reports Documentation page (http://otn.oracle.com/docs/products/reports/content.html).

2.8.5 About event-driven publishing

Event-driven publishing enables you to set up a report to execute when a certain action has been performed. For example, when an employee submits an expense report, new data is being inserted into the database. When this insert event (e.g., a database trigger or an Advanced Queuing (AQ) message) occurs, a report is sent to the employee's manager via their portal page or e-mail notifying them to approve/reject this expense report.

For detailed information, refer to the chapter "Event-Driven Publishing" in the *Oracle Application Server Reports Services Publishing Reports to the Web* manual, available on the Oracle Technology Network Oracle Reports Documentation page (http://otn.oracle.com/docs/products/reports/content.html).

2.8.6 About switching the printer tray

Using the Before Report, Between Pages, or format triggers, you can switch to different printer trays as your report formats. This allows you to easily print pages of the same report on different sheets of paper (e.g., letterhead, forms, colored).

You can determine the names of the printer trays defined for your printer in the Page Setup dialog box, then use SRW.SET_PRINTER_TRAY to set the printer tray as desired.

See also

Section 3.7.16.5, "Switching the printer tray"

2.8.7 About XML in reports

Oracle Reports uses XML (Extensible Markup Language) in the following ways:

- XML tags are used to define Web-based reports (see the topic "Oracle Reports XML tags" in the **Reference** section of the *Reports Builder online help*).
- XML is used to define tag-delimited, structured information.
- XML is a supported pluggable data source (PDS). For more information, see Chapter 41, "Building a Report with an XML Pluggable Data Source".

XML is a form of encoding text formats that can be read by many different applications. The XML tags can be used to output information or as a basis for building a pluggable data source to exchange electronic data with a third-party application (EDI).

You may change the XML properties that control XML output for your report at three levels: report, group, and column. Note that in any Reports Builder-generated XML file, your output mimics the data model, structured by groups and columns. For information on how to view your changes in XML output, see Section 3.7.7, "Generating XML output".

For detailed information about using XML for report distribution and customizing reports through XML, see the chapters "Creating Advanced Distributions" and "Customizing Reports with XML" in the *Oracle Application Server Reports Services Publishing Reports to the Web* manual, available on the Oracle Technology Network Oracle Reports Documentation page (http://otn.oracle.com/docs/products/reports/content.html).

See also

Section 3.7.7, "Generating XML output"

Section 3.7.1, "Running and dispatching a report from the user interface"

The **XML PDS** section of the *Reports Builder online help*.

2.8.8 About HTML output

Reports Builder can generate report output to Hypertext Markup Language (HTML) and Hypertext Markup Language with a Cascading Style Sheet (HTMLCSS) files, containing the formatted data and all objects. HTML is a form of encoding text formats that can be read by many different Web page developing software packages, such as Microsoft Front Page, and Web browsers. You can use the software's editing and graphics features to modify and enhance your report

output. When you generate your report output to an HTML or HTMLCSS file, you can distribute the output to any HTML destination, including e-mail, printer, OracleAS Portal, and Web browser.

> **Note:** This topic discusses the HTML generated when you run a paper-based report to an HTML or HTMLCSS file. This is completely separate from HTML that might be generated when you run a JSP-based Web report (when you click the Run Web Layout button in the toolbar, or choose **Program > Run Web Layout**).

Usage notes

- You can preview your HTML or HTMLCSS report output in your Web browser by choosing **File > Preview Format > Paginated HTML** (or **Paginated HTMLCSS**).
- Bookmarks cause multiple HTML files to be created. One master file is created with two frames: one for bookmarks and one for the report output. One HTML file is created for each of these frames. The master document filename is the name specified in the DESNAME parameter. The bookmark filename is *desname*`b.htm`. The report output filename is *desname*`d.htm`.
- Linked images, image fields, and graphs in a report cause GIF files to be created and referenced from the HTML document. Note that even if the linked boilerplate or image field refers to an external GIF file, a new GIF file is generated.
- If an image is stored in the database, one GIF file may be generated for each occurrence of the image in the report. If an image is stored in a file (e.g., imported images, linked boilerplate, or image fields that reference files), only one GIF file will be generated per image regardless of how many times it is repeated in the report.
- Any GIF image files generated for HTML output have a number sequence (e.g. *desname*`0.gif`... *desname*`17.gif`).
- A comment block in the master document contains the names of all of the files that are associated with the master document.
- HTML has seven sizes for text. The font sizes in the report are mapped according to the following table. Note that the user can override the size specified in the HTML file from their browser.

Table 2–1 Report to HTML font sizes

Report font size	HTML font size
less than 8	1
8 through 9	2
10 through 12	3
13 through 15	4
16 through 20	5
21 through 30	6
more than 30	7

Note: If the font used in the report is non-proportional, TeleType mode is turned on for the generated HTML file.

- HTML does not have the concept of a page. A separator line is placed between each page of the report. Depending upon the browser, you may or may not need to scroll to see the entire report page. Furthermore, if you print the HTML document from your browser, the printer will not necessarily print the separator lines at the bottom of each page. If you do not want the separator line or you want to use a different separator line, you can use SRW.SET_AFTER_PAGE_HTML to change it.
- For HTMLCSS, graphics and text can be overlapped.
- Once you have generated your report to an HTML or HTMLCSS file, the data model and looping tags are removed and replaced with the data. You can open the HTML or HTMLCSS file in Reports Builder, but it will be a static text file and not a report.
- For information about paginating HTML or HTMLCSS output, see Section 2.8.8.1, "About HTML page streaming".

Restrictions

- Objects cannot overlap one another. For example, you could not have text on top of an image.
  - If objects overlap slightly (two characters or less), then the underneath object is truncated to prevent overlap.

  - If objects overlap significantly (or one is completely on top of the other), then the underneath object is removed altogether. In this case, any linking information of the removed object is transferred using the same rules as if it were a frame (see the rules about frames below).
  - Text always takes precedence over horizontal lines, regardless of which is on top. This prevents the line underneath a column label from eliminating the label text.
- When multiple output files are generated (e.g., when bookmarks are used), any file except the master file will be overwritten without confirmation. For example, GIF files and bookmark files would be overwritten without prompting.
- Report frames are not visually represented in the HTML output. Any fill or border attributes of frames do not appear in HTML output.
- If the frame in a report is the target of a link or a bookmark, that attribute is transferred to the visible object nearest to the upper left corner of the frame in HTML output. If the frame object is a hyperlink, that attribute will be transferred to all the child (interior) objects that are not hyperlinks. If the hyperlink attribute cannot be transferred to the child objects, the frame's hyperlink is lost.
- The only drawn object supported in HTML is a solid, black, horizontal line. The line width specified in the report may be honored depending upon the browser. All other drawn objects (e.g., rectangles or circles) in the report layout will not show up in the HTML output. Space for these drawn objects is reserved, but there is no visible representation in the HTML output.
- Background (fill) and border (line) colors/patterns for text are not available in HTML. Bold, italic, underline, and foreground (text) color are supported if the browser supports them.
- The PDF action attribute is ignored for HTML output.
- Any browser customizations that have been made will affect how the generated HTML is displayed.

See also

Section 1.2.2, "About Web reports"

Section 2.2.3, "About Web links for HTML output"

Section 3.7.5, "Generating HTML or HTMLCSS output"

2.8.8.1 About HTML page streaming

HTML page streaming enables you to display individual pages of your HTML/HTMLCSS report output in your Web browser, without having to download the entire report. From the first page of the report, you can navigate to any page in the rest of the report. When you click a bookmark or hyperlink with a destination:

- within the report, the frame that contains the current page will update with the destination page.
- outside the report, the entire base frame (including the bookmark frame, the page, and the navigation frame) will reload.

Navigation controls

You can specify the navigation controls script for a report in either of the following ways:

- via PL/SQL (SRW.SET_PAGE_NAVIGATION_HTML in a Before Report trigger)
- in the Report Property Inspector via the Page Navigation Control Type and Page Navigation Control Value properties

Output file names

With HTML page streaming, each page is formatted as a separate HTML document. If your HTML file is named myreport.htm and there are no bookmarks, the new files are named as follows:

- myreport.htm (for the base frame)
- myreportb.htm (for the bookmark file, present only if bookmarks are used in the HTML files)
- myreport_1.htm through myreport_*n*.htm (for the pages)
- myreportj.htm (for the navigation JavaScript)

Scope of HTML output

To specify HTML to be displayed on only the first (header) or last (footer) pages of your report, set the Before Report or After Report properties or use the SRW.SET_BEFORE_REPORT_HTML or SRW.SET_AFTER_REPORT_HTML PL/SQL procedures. To specify global HTML to apply to the entire report, such as background colors or images, set the Before Page properties or SRW.SET_BEFORE_

PAGE_HTML PL/SQL procedure. The Reports Builder-generated HTML logo appears only on the last page of your report.

Enabling page-streamed output

To enable page streaming when you format your report as HTML or HTMLCSS output, you must specify PAGESTREAM=YES on the command line. This option can not be set via the Reports Builder user interface.

See also

Section 3.7.15.5, "Displaying individual pages of HTML report output"

Section 3.6.7.1.12, "Adding navigation controls for HTML page-streamed output using the Property Inspector"

Section 3.6.7.2.11, "Adding navigation controls for HTML page-streamed output using PL/SQL"

2.8.9 About PDF output

Reports Builder can generate report output to PDF files, containing the formatted data and all objects. When you generate your report output to a PDF file, you can distribute the output to any PDF destination, including e-mail, printer, OracleAS Portal, and Web browser.

> **Note:** Oracle Reports does not support Windows UDC for PDF output. For the user-defined characters to be printed and/or rendered, all the glyphs must be within a single TTF or TTC file.

For detailed information about PDF enhancements and capabilities in Oracle Reports, which include compression, font aliasing, font subsetting, font embedding and accessibility tags, refer to the chapter "PDF in Oracle Reports" in the *Oracle Application Server Reports Services Publishing Reports to the Web* manual, available on the Oracle Technology Network Oracle Reports Documentation page (http://otn.oracle.com/docs/products/reports/content.html).

Usage notes

- You can preview your PDF report output in your Web browser by choosing **File > Preview Format > PDF**.
- If you are building a multibyte report for multi-byte languages, such as Chinese or Japanese, and you need to alias the font in PDF output, you need the CID fonts named within the Acrobat 4.0 packs. Otherwise, you do not need the CID fonts in the Acrobat 4.0 packs.
- Graphics and text can be overlapped.
- The foreground color of the object will be used as the fill color (regardless of a specified pattern).

Restrictions

- For PDF output, the bit-mapped drivers (e.g., PostScript) for the currently selected printer are used to produce the output.

See also

Section 1.2.2, "About Web reports"

Section 2.2.4, "About Web links for PDF output"

Section 3.7.6, "Generating PDF output"

2.8.10 About RTF output

Reports Builder can generate report output to Rich Text Format (RTF) files, containing the formatted data and all objects. RTF can be read by many different word processing software packages, such as Microsoft Word. You can use the software's editing and graphics features to modify and enhance your report output. When you generate your report output to an RTF file, you can distribute the output to any RTF destination, including e-mail, printer, OracleAS Portal, and Web browser.

Usage notes

- Graphics and text can be overlapped.
- Text can only be rotated by 90-degree variations.
- As with PDF output, the foreground color of the object will be used as the fill color (regardless of a specified pattern).

Note: When you view the report in Microsoft Word in Office 95, you must choose **View > Page Layout** to see all the graphics and objects in your report.

See also

Section 3.7.8, "Generating RTF output"

Section 3.7.1, "Running and dispatching a report from the user interface"

2.8.11 About delimited output

Reports Builder can generate report output that includes a delimiter to delimited files (e.g., files that contain comma-separated or tab-separated data), which are easily imported into spreadsheets or for use with word processors.

Oracle Reports provides two choices for generating delimited output:

- Delimited
- DelimitedData (for use when you have problems running large volume reports with Delimited)

You can specify a delimiter (a character or string of characters) to separate the data (boilerplate or field objects) in your report output in either of the following ways:

- on the command line via the DELIMITER keyword.
- in the Delimited Output dialog box or DelimitedData Output dialog box (displayed via **File > Generate to File > Delimited** or **DelimitedData**).

When you generate delimited output, you can further distinguish the cells by using a cell wrapper. A cell wrapper can consist of any series of characters, such as a comma or parentheses.

For example, if the data in your report output include the same character as the delimiter (e.g. a comma), you can use the parentheses cell wrapper to distinguish each cell:

```
(1,000,000),(3,6000),(543),(2,003,500)…
```

Running macros on delimited output in Microsoft Excel

You can run predefined macros on report output generated in delimited (.csv) format when opening it using Microsoft Excel. This is similar to generating an Excel file on a predefined template with embedded macros to manipulate the worksheet. This is useful if you intend to use Excel to open a .csv file generated by Oracle Reports, and further manipulate it on Windows clients.

To create and load macros when opening a .csv file using Microsoft Excel:

1. In Microsoft Excel, use the record macro functionality (choose ToolsMacroRecord New Macro) to create the macros in a new worksheet.
2. Save the macros in `personal.xls`, in the *`root_directory`*`\`*`operating_system`*`\Profiles\`*`user_name`*`\Application Data\Microsoft\Excel\Xlstart` or *`root_directory`*`\Program Files\Microsoft Office\Office\XLStart` directory.

   Note: You can save the macro in `personal.xls` via the Record Macro dialog box or you can manually add your macros to `personal.xls`.

3. Choose **Window > Unhide** to ensure that `personal.xls` is opened by default whenever you use Excel, allowing you to use the macros you created and saved.

   Note: You can also specify an additional startup folder by choosing **Tools > Options**; on the **General** tab, type the folder path in **Alternate startup file location**. If a file with the same name is in both the `XLStart` folder and the additional startup folder, the file in the `XLStart` folder opens. Microsoft Excel will attempt to open every file in the `XLStart` and the additional startup folder, so make sure you specify an empty folder or a folder that contains only files that Excel can open.

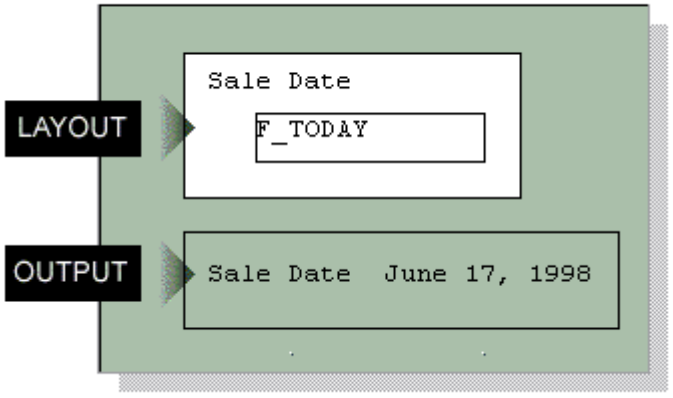
4. Save the template with embedded macros in one of the startup folders. Now, when you open your .csv file generated by Oracle Reports using Excel, you can run your macros on your report data (e.g., format data, perform calculations/manipulations, and so on).

Usage notes

When you generate delimited output, the data displays according to the positions of the objects in the Paper Layout view.

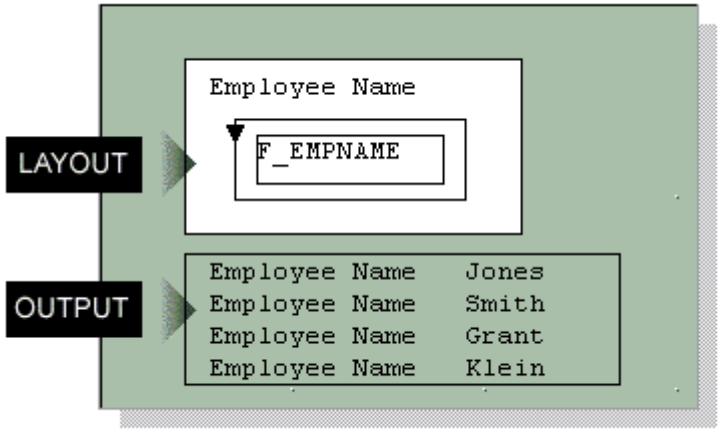
- If you place A above or to the left of B (where A and B are any boilerplate or field objects) Reports Builder displays each instance of A before each instance of B in every line of output.

Figure 2–8 Delimited output of A above B



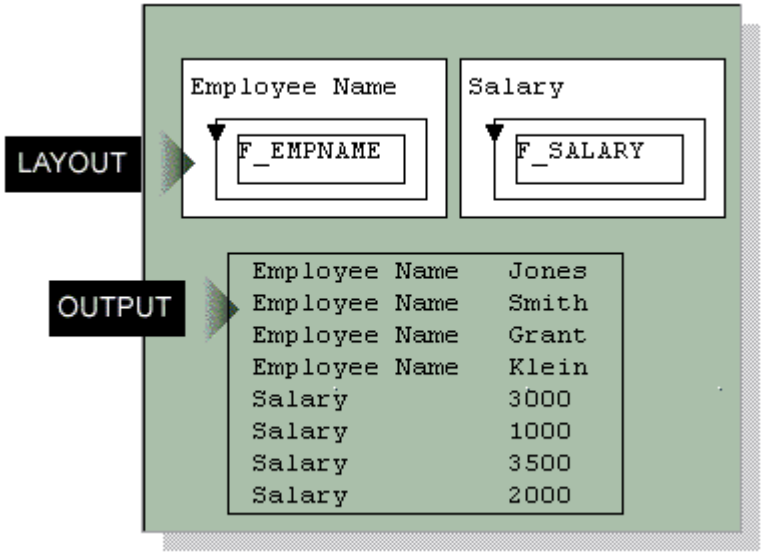
- If you create a frame that contains a boilerplate object (A) and encloses a repeating frame that contains a field object (B), each instance of A displays with each instance of B.

Figure 2–9 Delimited output of frame enclosing repeating frame



- In the previous layout, if you add another frame that contains a boilerplate object (C) and encloses another repeating frame that contains a field object (D), A displays for every instance of B, and then C displays for every instance of D.

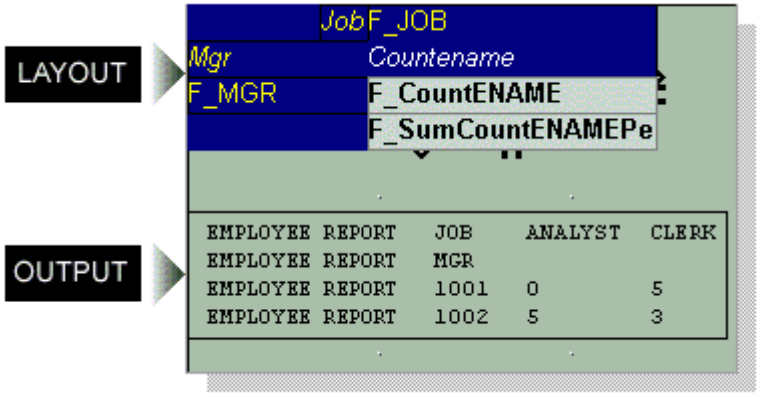
Figure 2–10 Delimited output of two frames enclosing repeating frames



- If you create a matrix in your report, be sure to align your objects carefully according to the grid in the Layout Model view. If the objects are not aligned, Reports Builder may interpret the extra space as an extra row or column and disrupt your report output.

- If you create a boilerplate object outside of a matrix, each instance of the boilerplate repeats with every row (not column) of the matrix. Note that boilerplates contained in the matrix will not be repeated with field objects outside of the matrix.

Figure 2–11 Delimited output of boilerplate outside matrix



Restrictions

If the text file contains a field labeled "ID" (in uppercase) as the first field, you will be unable to open the file in Microsoft Excel. The following delimited output causes an error in Excel:

ID, name, title, dept, etc.

If you want to generate delimited output that contains an ID field, try changing the database column name to lowercase, i.e., id, or re-arranging the order of the fields.

See also

Section 3.7.9, "Generating delimited output"

Section 3.7.1, "Running and dispatching a report from the user interface"

Section 1.2.2, "About Web reports"

2.8.12 About text output

Reports Builder can generate report output to text files, containing the formatted data and all objects. When you generate your report output to text, and the running mode is character (MODE=CHARACTER, or MODE system parameter Initial Value property set to *Character*), the result is pure text output, which can be read by many different applications. If the running mode is bitmap (MODE=BITMAP, or MODE system parameter Initial Value property set to *Bitmap*), the result is PostScript output, which can be read and rendered only by PostScript-compatible applications (such as a PostScript printer).

See also

Section 3.7.10, "Generating text output"

2.8.13 About creating an ASCII (character-mode) report

To create a character-mode report, you first create a bit-mapped report, then convert that report to an ASCII (character-mode) report. The process will create a new character-mode version of your bit-mapped report; the original bit-mapped report remains unchanged.

After conversion, many of your fields and text objects may need to be resized. Also, graphical objects such as images and drawings will not be included in your character-mode report. The following lists summarize what is supported in each output format:

Table 2–2 Supported items in bit-mapped and character-mode reports

Bit-mapped	Character-mode
Images	Boxes
Colors	Horizontal lines
Drawings	Vertical lines
Ellipses/Circles	ASCII text
Italicized text	Boldface text
Diagonal lines	Underlines
Bit-mapped patterns	
Multimedia objects	

See also

Section 3.5.9, "Creating an ASCII (character-mode) report"

Section 3.2.5, "Setting properties for an ASCII (character-mode) report"

Section 3.7.1, "Running and dispatching a report from the user interface"

2.9 Data Sources

Oracle Reports allows you to access any data source. See Section 3.15.1, "Accessing non-Oracle data sources".

The topics in this section discuss information related to accessing other data sources in Reports Builder.

- About database roles
- About Oracle Net Services
- About user exits
- About the Oracle Call Interface (OCI)

See also

The **Pluggable Data Sources** section of the *Reports Builder online help*, including the topics:

- About pluggable data sources
- Adding a pluggable data source
- Connecting to a pluggable data source
- Adding Online Help to a pluggable data source
- Pluggable data source interface definition
- Troubleshooting PDS problems

2.9.1 About database roles

Database roles provide a way for end users to run reports that query tables to which they do not have access privileges. For example, a report may query a table that includes sensitive information such as salary data, even though the final report does not display this data.

Database roles are checked in the runtime environment only. If a table requires special access privileges, end users without those privileges cannot run a report that retrieves data from that table. However, if a database role is defined for the report, end users with privileges for that role can run the report using Reports Runtime (`rwrun`).

Note: To run a report for which a database role is defined, the end user must run the .rdf file, not the .rep file. When running multiple reports, Reports Runtime automatically switches to the role defined for the current report.

If you try to open a report in Reports Builder for which a database role has been defined, you will be prompted for the role password. Typically, only the report designer and DBA have this information.

See also

Section 3.16.1, "Setting a database role"

2.9.2 About Oracle Net Services

Oracle Net Services is Oracle's remote data access software that enables both client-server and server-server communications across any network. It supports distributed processing and distributed database capability. Oracle Net Services runs over and interconnects many communication protocols. Oracle Net Services is backwardly compatible with Net8 and SQL*Net.

2.9.3 About user exits

In prior releases, user exits provided a way to pass control from Reports Builder to a program you have written, which performs some function, and then returns control to Reports Builder. You could write ORACLE Precompiler user exits, OCI (ORACLE Call Interface) user exits, or non-ORACLE user exits to perform tasks such as complex data manipulation, passing data to Reports Builder from operating system text files, manipulating LONG RAW data, supporting PL/SQL blocks, or controlling real time devices, such as a printer or a robot.

Now, you can call Java methods using the ORA_JAVA built-in package and the Java Importer. This reduces the need to have user exits in a report and allows for a more open and portable deployment. You may also use the ORA_FFI built-in package, which provides a foreign function interface for invoking C functions in a dynamic library. With the availability of these new built-in packages, the use of user exits is being deprecated in Oracle Reports, though makefiles will still be supplied to permit you to continue to work with existing user exits.

2.9.4 About the Oracle Call Interface (OCI)

In prior releases, the Oracle Call Interface (OCI) provided a set of standard procedures that you could call in your 3GL programs to call Oracle Reports executables. These procedures (written in C) were shipped with the Reports Builder, Reports Runtime, and Reports Converter executables. For example, to run a Reports Builder report from a Pro*FORTRAN program, you could add a RWCRRB procedure call to your program to run the report using the Reports Runtime executable.

Now, the OCI is obsolete. Instead, use the rwclient.exe command line interface or the JSP tag library.

2.10 Debugging Tools

The topics in this section discuss debugging reports in Reports Builder.

- About the debugging process
- About the PL/SQL Interpreter
- About the Source pane
- About debug commands in the PL/SQL Interpreter
- About debug actions
- About the current execution location
- About the current scope location
- About debug levels
- About modifying code at runtime

2.10.1 About the debugging process

Debugging an application is an iterative process in which application errors are identified and corrected. In general, quickly identifying and locating failing code is essential to successfully debugging your application.

See also

Section 3.14.1, "Debugging a report"

Section 3.14.2, "Running a report in debug mode"

2.10.2 About the PL/SQL Interpreter

The PL/SQL Interpreter is your debugging workspace, where you can display source code, create debug actions, run program units, and execute Interpreter commands, PL/SQL, and SQL statements.

By default, two panes are always open in the PL/SQL Interpreter: Source pane and Interpreter pane.

Debugging features include the following:

- **Direct manipulation debugging**: insert debug actions and inspect program data by directly manipulating displayed source text.

- **Dynamic execution feedback**: Reports Builder automatically displays and tracks the current PL/SQL source location as program execution is interrupted by debug actions or incrementally advanced during program stepping.
- **Browsing of interrupted program state**: once execution has been interrupted, it is possible to browse the current stack, browse and modify variable state, and execute arbitrary PL/SQL statements. All information is accessed symbolically (i.e., by name as opposed to by address or number).

2.10.3 About the Source pane

The PL/SQL Interpreter's Source pane displays a read-only copy of the program unit currently selected in the Object Navigator pane.

The numbers along the left hand margin correspond to the line numbers of the displayed program unit.

In addition, the symbols described below may appear in the margin.

Table 2–3 Symbols in the margin of the Source pane

Symbol	Description
|	Specifies the current source location.
=>	Specifies the current scope location.
-	Specifies the current execution location (if different from the current scope location).
B(*n*)	Specifies the location of a breakpoint, where n is the corresponding debug action ID. It appears in the line number column.
T(*n*)	Specifies the location of a trigger, where n is the corresponding debug action ID. It appears in the line number column.

2.10.4 About debug commands in the PL/SQL Interpreter

The following commands are available when using the PL/SQL Interpreter:

Table 2–4 PL/SQL Interpreter Commands

Command	Description
CREATE	Creates a new library that can be stored in either the file system or the current database.
DELETE	Deletes: ■ libraries that reside in the current database ■ library program units ■ program units
DESCRIBE	Inspects a variable or parameter that is local to the current scope location. The description includes the name, type, and value of the specified local.
EXPORT	Writes the source of one or more program units to a text file.
LIST	Displays the source text for program units, triggers, and debug actions.
LOG	Saves a transcript of PL/SQL Interpreter input and output to the specified log file.
RESET	Returns control to an outer debug level without continuing execution in the current debug level.
SET	Changes the current scope location to a specified frame of the stack. You can specify relative motion from the current stack frame to any other frame, or move to a particular subprogram on the stack. There are several ways to invoke SET: ■ select a frame entry in the Object Navigator ■ enter the SET command in the PL/SQL Interpreter
SHOW	Lists the name, type, and value of all variables and parameters at the current scope location.

2.10.5 About debug actions

The PL/SQL Interpreter can be invoked from report code (triggers, user-named program units, libraries, etc.) by creating *debug actions* in the code. These are instructions which that the execution of PL/SQL program units so they can be monitored.

Each debug action you create is automatically assigned a unique numeric ID. While debugging, you can refer to this ID to browse, display, or modify a specific debug action via Reports Builder debug commands.

You can display detailed information about one or more debug actions, including its ID, source location, and whether or not it is enabled. You can temporarily disable specific debug actions and then re-enable them later if necessary.

There are two types of debug actions: *breakpoints* and *debug triggers*.

Breakpoints suspend execution at a specific source line of a program unit, passing control to the PL/SQL Interpreter.

Create breakpoints to identify specific debugging regions. For example, create a breakpoint at lines 10 and 20 to debug the code within this region.

With breakpoints, suspension occurs just *before* reaching the line on which the breakpoint is specified. At this point, use the PL/SQL Interpreter's features to inspect and/or modify program state. Once satisfied, resume execution with the GO or STEP commands, or abort execution using the RESET command.

Debug Triggers are a general form of debug action that associate a block of PL/SQL code with a specific source line within a program unit. When a debug trigger is encountered, Reports Builder executes the debug trigger code.

Create a debug trigger to execute a block of PL/SQL code provided at debug time:

1. when program execution reaches a single line in a program unit (e.g., the current source location, line 5, line 23, etc.)
2. every time the PL/SQL Interpreter takes control (i.e., whenever it suspends program execution due to a breakpoint, program stepping, etc.)
3. at *every* PL/SQL source line being run

Debug triggers are especially useful as conditional breakpoints. You can raise the exception DEBUG.BREAK from within a trigger. For example, the debug trigger shown below establishes a conditional breakpoint on line 10 of `my_proc`, which will be reached only if the local NUMBER variable `my_sal` exceeds 5000:

```
PL/SQL>line 10 is
+> IF Debug.Getn('my_sal') > 5000 THEN
+> Raise Debug.Suspend;
+> END IF;
```

2.10.5.1 About creating a debug action

You can create debug actions (breakpoints and debug triggers) in the PL/SQL Interpreter in the following ways:

- choosing **Program > Breakpoint** or **Program > Debugging Triggers** on the Reports Builder menu bar while a program unit is open in the PL/SQL Interpreter
- right-clicking in the Source pane of the PL/SQL Interpreter and choosing **Breakpoint** or **Debug Trigger**
- inserting debug actions in the Object Navigator pane
- entering commands in the Interpreter pane

When you create a debug action, attach the breakpoint or debug trigger to a program unit source line that is "executable." A source line is considered executable if it contains one or more statements for which the PL/SQL compiler generates code. For example, source lines containing assignment statements and procedure calls are executable, while source lines containing comments, blank lines, declarations, or the NULL statement are not executable.

See also

Section 3.14.3, "Setting a breakpoint"

Section 3.14.4, "Setting a debug trigger"

2.10.6 About the current execution location

The current execution location specifies the next PL/SQL source line to be executed. It corresponds to what is commonly referred to as the program counter, or PC.

When control passes to the PL/SQL Interpreter while running a program (e.g., when a breakpoint is encountered or following a step operation), the Source pane in

the PL/SQL Interpreter automatically displays the source line associated with the current execution location.

Use the LIST command in the Interpreter pane to manually display the current execution location.

For example, entering:

```
.LIST PC
```

will list the current execution location in the Source pane.

2.10.7 About the current scope location

The current scope location dictates where the PL/SQL Interpreter looks for local variables and parameters. It corresponds to the current execution location of one of the PL/SQL subprograms on the stack.

Each time a program unit's execution is interrupted (e.g., by a debug action), the scope location is initialized to the execution location of the subprogram at the bottom of the stack.

Once execution has been interrupted, you can change the current scope location to another frame on the stack. This enables you to view local variables in another subprogram in the call chain.

See also

Section 3.14.14, "Displaying the current scope location"

2.10.8 About debug levels

When a debug action interrupts program execution, the PL/SQL Interpreter takes control and establishes what is known as a *debug level*. At a debug level, you can enter commands and PL/SQL statements to inspect and modify the state of the interrupted program unit as well as resume execution.

Since any PL/SQL code interactively entered at a debug level may itself be interrupted (for example, by encountering another breakpoint), it is possible for debug levels to nest. To facilitate distinguishing one debug level from another, the levels are numbered. The most deeply nested level is assigned the highest number. Numbering starts at zero with the outermost level.

The 0th or outermost level is commonly referred to as top level. Top level has no associated program state since it is the outermost level at which program units are originally invoked. When code invoked from top level is interrupted, debug level 1

is established. Similarly, interrupting code invoked from debug level 1 establishes debug level 2, and so on.

The PL/SQL Interpreter command prompt reflects the current debug level. When the PL/SQL Interpreter enters levels below top level, the prompt includes a prefix containing the current debug level number. For example, the PL/SQL Interpreter command prompt at debug level 1 appears as:

```
(debug 1)PL/SQL>
```

2.10.9 About modifying code at runtime

At runtime, you can modify and compile any program unit, menu item command, or trigger that is not on the current stack.

> **Note:** To modify an item on the current stack, first clear the stack by issuing the RESET command.

Although runtime code modification is not communicated back to Reports Builder, you can interactively test possible fixes, before returning to implement the eventual fix.

See also

Section 3.14.13, "Modifying code at runtime"

3

How To...

This chapter provides procedures for using Oracle Reports to create objects and design your reports. Each topic in this chapter is also included in the **How To...** section of the *Reports Builder online help* (see Section 3.1.1, "Using the online help").

The procedures are grouped into the following sections:

3.1 Access Oracle Reports Documentation

This section provides procedures for the following tasks that you may perform to access the Oracle Reports documentation:

- Using the online help
- Locating other documentation

3.1.1 Using the online help

You can access the *Reports Builder online help* in any of the following ways:

- Choose **Help > Help Contents.**
- Click **Help** or press F1 in any dialog box.
- In the Property Inspector, click a property, then press F1 to display the property's help topic.

The following guidelines will help you to make effective use of the online help system:

- Familiarize yourself with the **Contents** tab to get an idea of the scope of the topics covered in the online help.
- Click the **Index** tab to locate specific topics on a subject.
- The Help Navigator window and Help Topic window can be separated or kept together. Move the windows as appropriate for your use.

3.1.2 Locating other documentation

To get started and become proficient with building reports, refer to the following resources:

Table 3–1 Oracle Reports documentation roadmap

For...	Refer to...
an overview of Oracle Reports	■ Getting Started with Oracle Reports, available on the Oracle Technology Network Oracle Reports Documentation page (`http://otn.oracle.com/docs/products/reports/content.html`). ■ the online help topics under the **Welcome** node in the Help Navigator window (choose **Help > Help Contents**): About Reports Builder and About this release.
an overview of new features in this release	■ *Oracle Reports New Features*, available on the Oracle Technology Network Oracle Reports main page (`http://otn.oracle.com/docs/products/reports/content.html`). ■ the online help topic under the **Welcome** node in the Help Navigator window (choose **Help > Help Contents**): About this release.
an overview of deprecated, obsolete, and changed functionality in this release	■ *Oracle Reports Obsolescence Plan*, available on the Oracle Technology Network Oracle Reports main page (`http://otn.oracle.com/docs/products/reports/content.html`). ■ the online help topic under the **Welcome** node in the Help Navigator window (choose **Help > Help Contents**): About this release.
instructions and hands-on lessons about building paper and Web-based reports	■ This manual, and *Oracle Reports Tutorial*, available on the Oracle Technology Network Oracle Reports Documentation page (`http://otn.oracle.com/docs/products/reports/content.html`). ■ the online help (choose **Help > Help Contents**) for specific tasks and reference information.

Table 3–1 Oracle Reports documentation roadmap

For...	Refer to...
context-sensitive help on dialog boxes, messages, and properties	■ for dialog boxes and messages, the online help topics displayed when you click the **Help** button, or press F1. ■ for properties, the online help topic displayed when you click a property in the Property Inspector, then press F1.
Information about OracleAS Reports Services and configuring and using the Reports Server	■ *Oracle Application Server Reports Services Publishing Reports to the Web*, available on the Oracle Technology Network Oracle Reports Documentation page (`http://otn.oracle.com/docs/products/reports/content.html`).
information about using the Oracle Reports application program interface (API)	■ *Oracle Reports JavaAPI Reference*, available on the documentation CD and the Oracle Technology Network Oracle Reports Documentation page (`http://otn.oracle.com/docs/products/reports/content.html`).
examples and demos	■ Getting Started with Oracle Reports, available on the Oracle Technology Network Oracle Reports Documentation page (`http://otn.oracle.com/docs/products/reports/content.html`). Use the Index to find the list of all examples and demos for Oracle Reports. ■ A subset of the examples is also available on the Oracle Developer Suite product CD: Tutorial example files Express example files Building a Report with a Barcode Bean example Building a Report with an XML Pluggable Data Source example Building a Report with a Text Pluggable Data Source example Bursting and Distributing a Report example Oracle Reports Demos
other resources, such as white papers	■ the Oracle Technology Network (`http://otn.oracle.com`).

3.2 Set Properties and Preferences

This section provides procedures for the following tasks that you may perform as you work with property and preference settings:

- Displaying the Property Inspector
- Setting report properties
- Setting report preferences
- Setting preferences for the Object Navigator
- Setting properties for an ASCII (character-mode) report
- Setting color palette preferences
- Setting properties of multiple objects
- Comparing the properties of one object to another

3.2.1 Displaying the Property Inspector

To display the Property Inspector for an object:

- In the Object Navigator, do either of the following:
  - Double-click the icon immediately to the left of the object name.
  - Click the object name, then right-click and choose **Property Inspector.**
  - Click the object name, then choose **Tools > Property Inspector**.
- In the Report Editor views, do either of the following:
  - Double-click the object.
  - Click the object, then right-click and choose **Property Inspector**.
  - Click the object, then choose **Tools > Property Inspector.**

See also

Section 1.10.1, "About the Property Inspector"

3.2.2 Setting report properties

To set the properties for a report:

1. In the Object Navigator, double-click the properties icon next to the report name to display the Property Inspector.
2. Set report properties as desired.

3.2.3 Setting report preferences

To specify the preferences for a report:

1. Choose **Edit > Preferences**.
2. To specify preferences for designing reports, set values on the **General**, **Access**, and **Wizards** tabs.
3. To specify preferences for running reports, set values on the **Runtime Values** and **Runtime Settings** tabs.
4. For a detailed description of settings in the Preferences dialog box, click **Help**.

3.2.4 Setting preferences for the Object Navigator

To specify options for the Object Navigator display:

1. In the Object Navigator, choose **Tools > Options > Navigator**.
2. In the Object Navigator Options dialog box, change the settings as desired.
3. Click **OK**.

See also

Section 1.5.1, "About the Object Navigator"

3.2.5 Setting properties for an ASCII (character-mode) report

To set properties for an ASCII (character-mode) report:

1. Double-click the properties icon next to the report name to display the Property Inspector.
2. Under the **Character Mode** node, set Design In Character Units to Yes.

3. In the Object Navigator, under the **Paper Layout** node, double-click the Header Section, Main Section, or Trailer Section properties icon for the pertinent report section (Header, Main, or Trailer) to display the Property Inspector.

   Note: By default, a report is defined in the Main section.

4. In the Property Inspector, under the **Section** node:
   - set the Report Width and Report Height to the appropriate character-mode dimensions for the report. For example, 132 (or 180) width x 66 height for landscape or 102 width x 85 (or 116) height for portrait.
   - set the Orientation property to the desired value.
5. In the Object Navigator, expand the **Data Model** node, then the **System Parameters** node.
6. Double-click the properties icon next to **MODE** to display the Property Inspector, and set properties:
   - Under the **Parameter** node, set the Initial Value property to Character.
7. In the Paper Layout view, choose **Tools > Options > Rulers** to display the Ruler Settings dialog box:
   - Set **Units** to Character Cells and **Number of Snap Points Per Grid Spacing** to 1.
   - Click **OK**.
8. Click **View** in the menu bar and make sure that **Snap to Grid** is checked.
9. Choose **Edit > Preferences** to display the Preferences dialog box:
   - On the **Wizards** page, set **Horizontal Interfield** to 1 and **Vertical Interfield** to 0.
   - Click **OK**.
10. Choose **Format > Font**, and select the font, style, and size that most closely approximates the character-mode font. For example, Courier, Regular, 12 points.

See also

Section 3.5.9, "Creating an ASCII (character-mode) report"

3.2.6 Setting color palette preferences

To set color palette preferences:

1. Choose **Edit > Preferences**.
2. In the Preferences dialog box, on the **General** tab page, set the **Color Mode** as desired:
   - **Editable**. Reports Builder temporarily replaces your system's color palette with the palette of the active report. Each time a new report is made active, its color palette replaces the system palette. The active report will then be shown accurately, although the appearance of the inactive reports may suffer. You must choose Editable if you want to modify the color palette or import or export a color palette.
   - **Read Only - Shared**. (Default) Reports Builder continues to append each active report's color palette to your system's original palette until there is no room left in the palette for any more colors. That palette becomes the single palette that all open reports share. If you create a report that uses entirely different colors, it may not be shown accurately.
   - **Read Only - Private**. Each time you make a report active, Reports Builder appends that report's color palette to your system's original palette until there is no room left in the palette for any more colors. That palette becomes the palette that is used whenever that report is active. The active report is then shown accurately, although the appearance of the inactive reports may suffer.
3. If you change the **Color Mode**, shut down and restart Reports Builder to enable the new mode.

See also

Section 2.4.5, "About changing colors and patterns"

Section 3.9.6.6, "Importing or exporting a color palette"

Section 3.9.6.5, "Modifying the color palette"

Section 3.9.6.2, "Changing colors"

Topics "Oracle CDE1 color palette", "Default color palette", and "Grayscale color palette" in the **Reference > Color and Pattern Palettes** section of the *Reports Builder online help*

3.2.7 Setting properties of multiple objects

To set the properties of multiple objects:

1. In the Object Navigator or any Report Editor view, select the objects whose properties you want to set. The objects can be of different types and can even be in different documents.
2. Choose **Tools > Property Inspector**.
3. In the Property Inspector, click the Union button to see all properties of all objects or leave the tool as the Intersection button to see only the properties the objects have in common.
4. Set the properties as desired.

> **Note:** Any change you make to a property setting is applied to all of the objects in the current selection to which that property applies.

For example, a report includes several fields, each of which displays the date. The Datatype property of each field is DATE, which causes dates to be displayed in the default ORACLE format DD-MON-YY.

To use a different date format throughout the application, you need to set the Format Mask property for each field that displays dates. Rather than setting each field's Format Mask property individually, select all of the items, then set the Format Mask property once to change the format mask for each item selected.

See also

Section 1.10.1.1, "About making multiple selections in the Property Inspector"

3.2.8 Comparing the properties of one object to another

To compare the properties of one object to another:

1. In the Object Navigators or the editors, double-click the first object so that its properties are displayed in the Property Inspector.
2. In the Property Inspector, click the Pin button to attach this copy of the Property Inspector to the current object.
3. In the Object Navigator, click the second object, then choose **Property Inspector** from the pop-up menu.

   A second Property Inspector is displayed. If the second window is on top of the first, drag it to move it alongside the first window.

3.3 Perform Common Tasks

As you work with Reports Builder, you will become very familiar with the following tasks:

- Connecting to a database
- Opening a report
- Saving a report
- Renaming a report
- Deleting a report
- Archiving a report
- Selecting and deselecting objects
- Deleting an object
- Hiding or showing components

Usage notes

Beginning with Oracle Reports 10*g*, reports are no longer stored in the database (including the obsolescence of Rename, Grant, and Get Info functionality for reports in the database). Instead, reports are saved to files or into source control using the Check In and Check Out capabilities of Oracle Software Configuration Manager (SCM).

3.3.1 Connecting to a database

To connect to a database when designing a report in Reports Builder:

1. Choose **File > Connect**.
2. In the Connect dialog box, type the required information in the **User Name**, **Password**, and **Database** fields (click **Help** for more information), then click **Connect**.

To connect to a database using the command line:

- On the `rwbuilder` or `rwrun` command line, use the USERID keyword to specify connection information (`USERID=username[/password][@database]`). For information about USERID, see the **Reference > Command Line** section of the *Reports Builder online help*.

3.3.2 Opening a report

To open a report:

1. Choose **File > Open**.
2. In the Open dialog box, browse to the report, click its name, then click **Open** to open the report in Reports Builder.

3.3.3 Saving a report

To save a report:

1. Choose **File > Save** or **File > Save As** or click the **Save** button in the toolbar.
2. In the Save dialog box, browse to the desired location, and type a name for the report, if it has never been saved before. Click **Save**.

3.3.4 Copying a report

To copy a report:

1. In the Object Navigator, click the report you want to copy.
2. Choose **File > Save As**.
3. In the Save As dialog box, browse to the desired location, and type a name for the new copy of the report. Click **Save**.

3.3.5 Renaming a report

To rename a report in the Object Navigator:

1. In the Object Navigator, double-click the properties icon next to the report name to display the Property Inspector.
2. In the Property Inspector, under the **General Information** node, set the Name property by typing a new name.

To rename a report in the file system:

1. In your operating system's file system (e.g., Windows Explorer on Windows), browse to the location of the report.
2. Click the report name, then click again to make its name active.
3. Type a new name for the report.

3.3.6 Deleting a report

To delete a report in the Object Navigator:

1. In the Object Navigator, click the report name, then choose **File > Delete**, or click the Delete button in the toolbar.
2. In the alert dialog box, click **Yes** to delete the report. To cancel the delete operation, click **Cancel**.

To delete a report in the file system:

1. In your operating system's file system, for example, Windows Explorer on Windows, browse to the location of the report.
2. Click the report name, then choose **File > Delete**.
3. In the alert dialog box, click **Yes** to delete the report. The cancel the delete operation, click **Cancel**.

3.3.7 Archiving a report

To archive a report in Oracle Reports 10*g*, it can be saved into source control using the Check In and Check Out capabilities of Oracle Software Configuration Manager (SCM).

SCM is a single central location where all application components, including business logic, can be stored. This provides complete control over your application development environment. SCM includes features such as source/version control, impact analysis, and check in and check out of all the related application components. Oracle Reports is integrated with Oracle SCM, which allows you to easily control your reports in a seamless integrated environment.

3.3.8 Selecting and deselecting objects

To manipulate objects or define their properties, you must first select them.

3.3.8.1 Selecting single objects

To select a single object in the Object Navigator:

- Click the object. If the object is displayed in one of the Report Editor views, it is also selected in the corresponding view.

To select a single object in the Data Model view, Paper Layout view, Paper Design view, or Parameter Form view:

1. If a tool in the tool palette is active, first click the Select tool in the tool palette.
2. Click the object.

3.3.8.2 Selecting multiple objects

To select multiple objects in the Object Navigator:

- Click the first object, then either:
  - Shift-click another object to select all objects between the first and current object.
  - Or, control-click additional objects to add them to the selection group.

  Objects that are displayed in one of the Report Editor views are also selected in the corresponding view.

To select multiple objects in the Data Model view, Paper Layout view, Paper Design view, or Parameter Form view:

1. If a tool in the tool palette is active, first click the Select tool in the tool palette.
2. Do either of the following:
   - Shift-click each object.
   - Or, click and drag a region that includes all of the objects you want to select. (In the Data Model view, note that if the first object you select is a column within a group, everything else you select must be a column within the same group; you cannot select columns within groups at the same time as you select queries, groups, report-level columns, or parameters.)

3.3.8.3 Selecting objects owned by a frame

To select a set of objects owned by a frame or repeating frame:

1. In the Paper Layout view, either:
   - Click the Confine On button in the toolbar.
   - Or, if you want to explicitly select all objects to set common properties, click the Frame Select button in the toolbar.
2. Click the frame or repeating fame.

   All objects within the frame or repeating frame are selected, depending upon their explicit anchors.

3.3.8.4 Selecting grouped objects

To select grouped objects (grouped using **Layout > Group Operations > Group**), click the Select tool in the tool palette and click on one of the grouped objects.

3.3.8.5 Selecting all objects in a report region

To select all objects within the region (Body or Margin of the Header Section, Main Section, or Trailer Section) you are currently editing, choose **Edit > Select All**. (In the Data Model view, note that this does not select any columns within groups.)

3.3.8.6 Selecting overlapped objects

To select any object that is partially or completely overlapped by another object:

1. Click the object that obscures the object you want to select.
2. Choose **Layout > Send to Back** to move the object that you want on top of the object that obscured it.
3. Click the object.

3.3.8.7 Deselecting single objects

To deselect a single object in the Object Navigator:

- Control-click the object. If the object is displayed in one of the Report Editor views, it is also deselected in the corresponding view.

To deselect a single object in the Data Model view, Paper Layout view, Paper Design view, or Parameter Form view:

- Shift-click the object.

3.3.8.8 Deselecting multiple objects

To deselect all selected objects:

- Click anywhere in a blank area.

3.3.9 Deleting an object

To delete an object:

1. Click the object(s).

> **Tip:** To select multiple objects, click one object, then Shift-click all other objects. To select all objects, choose **Edit > Select All**.

2. Choose **Edit > Clear**.

3.3.10 Hiding or showing components

To hide or show components in the Object Navigator or Report Editor views:

- Click the **View** menu, then choose the component you want to hide or show. A checkmark indicates that the component is shown; no checkmark indicates that the component is hidden.

3.4 Work with the Object Navigator

This section provides procedures for the following tasks that you may perform as you work with the Object Navigator:

- Displaying a Report Editor view from the Object Navigator
- Expanding and collapsing nodes
- Searching for nodes
- Changing Object Navigator views
- Setting preferences for the Object Navigator

See also

Section 1.5.1, "About the Object Navigator"

3.4.1 Displaying a Report Editor view from the Object Navigator

To display a Report Editor view from the Object Navigator, do any of the following:

- Double-click the icon next to the node of the view you want to display. For example, to display the Paper Layout view, double-click the view icon next to the **Paper Layout** node.
- Click the node, then click the right mouse button and choose **Report Editor**.
- Click the node, then choose **Tools > Report Editor**.

Note: In the Object Navigator, there is no **Paper Design** node; the Paper Design view is displayed only after running a report.

See also

Section 1.6.2, "About the Data Model view"

Section 1.6.3, "About the Paper Layout view"

Section 1.6.5, "About the Paper Parameter Form view"

Section 1.6.6, "About the Web Source view"

3.4.2 Expanding and collapsing nodes

To expand or collapse a node one level, do either of the following:

- In the Object Navigator, click the plus or minus sign next to the node.
- Click the node, then click the Expand or Collapse tool in the tool palette.

To fully expand or collapse a node, do either of the following:

- In the Object Navigator, click the plus or minus sign next to the node repeatedly until fully expanded.
- Click the node, then click the Expand All or Collapse All tool in the tool palette.
- Click the node, then choose **View > Expand All** or **View > Collapse All**.

3.4.3 Searching for nodes

To search for a specific node in the Object Navigator:

1. Type the full of partial name of the node in the **Find** field.
2. To search for the next match, click the **Find Next** button in the toolbar.
3. To search for the previous match, click the **Find Previous** button in the toolbar.

3.4.4 Changing Object Navigator views

To view objects in the Object Navigator by object hierarchy to see the parent-child relationships:

- Choose **View > Change View > Ownership View**.

To view objects in the Object Navigator by their type such as all queries under a single heading:

- Choose **View > Change View > Object Type View**.

See also

Section 1.5.2, "About Object Navigator views"

3.4.5 Setting preferences for the Object Navigator

See Section 3.2.4, "Setting preferences for the Object Navigator".

3.5 Work with Reports

This section provides procedures for the following tasks that you may perform as you work with reports (either paper-based or Web-based):

- Creating a report
- Creating a multiquery group above report
- Creating a nested matrix report
- Creating a default layout for a report
- Creating an additional report layout
- Adding a title to a report
- Adding a table of contents to a report
- Adding index to a report
- Creating an ASCII (character-mode) report
- Preparing a multiplatform report
- Preparing a report for translation into other languages

3.5.1 Creating a report

To create a single-query report:

1. In the Object Navigator, click the **Reports** node, then click the Create button in the toolbar.
2. In the New Report dialog box, click **Use the Report Wizard**, then click **OK**.
3. Follow the Report Wizard to create your report, clicking **Help** for assistance on any tab page.
4. Modify the resulting report output in the Web Source view or Paper Design view, or choose **Tools > Report Wizard** to re-enter the wizard.

To build a multiquery report:

5. Create each query using the Data Wizard, clicking **Help** for assistance on any tab page (see Section 3.8.1.2, "Creating a query: Data Wizard").
6. Create a layout for the report (see Section 3.5.4, "Creating a default layout for a report").

See also

Section 1.2.1, "About reports"

Section 1.3.1, "About tabular reports"

Section 1.3.2, "About group above reports"

Section 1.3.3, "About group left reports"

Section 1.3.4, "About form-like reports"

Section 1.3.5, "About form letter reports"

Section 1.3.6, "About mailing label reports"

Section 1.3.7, "About matrix reports"

Section 2.1.7, "About nested matrix reports"

Section 2.1.8, "About matrix with group reports"

3.5.2 Creating a multiquery group above report

See the example report in Chapter 11, "Building a Two-Query Group Report".

3.5.3 Creating a nested matrix report

See the example report in Chapter 26, "Building a Nested Matrix Report".

3.5.4 Creating a default layout for a report

To create a default layout for a report:

1. To default the layout for the current report (by default, the Main section of the report), choose **Tools > Report Wizard**, then follow the wizard to create the layout for the report style you choose.
2. To add another layout section to the current report layout, create an additional report layout, as described below.
3. Make further modifications to the default layout manually in the Paper Layout view.

Caution: If you re-enter the Report Wizard after making manual adjustments to your layout in the Paper Layout or Paper Design view, you will lose these layout changes when you click **Finish** in the Report Wizard, which redefaults the layout.

See also

Section 2.4.2, "About layout defaulting"

Section 3.10.2, "Creating a default layout for a section"

Chapter 3.9.13.3, "Changing the default layout spacing"

3.5.5 Creating an additional report layout

To add a new layout section to a report without overriding existing layouts:

1. In the Paper Layout view, click the Report Block tool in the tool palette.
2. Drag a rectangular area for the new layout to display the Report Block Wizard.
3. Follow the wizard to select the data to display in the new layout section.
4. To reorder the layout sections, click and drag them to new positions in the Paper Layout view.
5. Modify the report output in the Paper Design view.

Caution: If you re-enter the Report Wizard to make modifications, then click **Finish** in the Report Wizard, your entire layout will be overwritten with a new default layout. You will lose the additional report layout you have created, and any manual changes made to the layout in the Paper Layout or Paper Design view.

3.5.6 Adding a title to a report

To add a title to a report:

1. In the Object Navigator, select or open the report.
2. Choose **Tools > Report Wizard**.
3. On the **Style** page, type a title for the report in the **Title** field.
4. If the report title is to be used in a template, see Section 3.12.5, "Formatting the report title in a template".

See also

Section 2.1.1, "About report titles"

Section 2.7.1, "About templates"

3.5.7 Adding a table of contents to a report

To create a table of contents (TOC) for a report, you can use the Format Order property or the SRW.SET_FORMAT_ORDER built-in procedure to format the Main section of the report first and use report triggers to build a table containing the TOC entries. When the first element for the TOC is formatted, a trigger fires and creates a row in the TOC table containing the TOC entry and the page number. After the Main section has completed formatting, the format order setting can define that the Header section is formatted next. The Header section can contain a report block based on the TOC table. After formatting, you can output your report with a TOC (the Header section), followed by the report body (the Main section), followed by the Trailer section.

For two step-by-step examples, see Chapter 34, "Building a Paper Report with a Simple Table of Contents and Index" and Chapter 35, "Building a Paper Report with a Multilevel Table of Contents".

See also

Section 2.8.1, "About format order"

3.5.8 Adding index to a report

To create an index for a report, you can use report triggers to build a table containing the index entries as you format the Main section of your report. When the first element for the index is formatted, a trigger fires and creates a row in the index table containing the index entry and the page number. After the Main section has completed formatting, the Trailer section is formatted next by default. The Trailer section can contain a report block based on the index table. After formatting, you can output your report with the report body (the Main section), followed by an index (the Trailer section).

For a step-by-step example, see Chapter 34, "Building a Paper Report with a Simple Table of Contents and Index".

3.5.9 Creating an ASCII (character-mode) report

To create an ASCII (character-mode) report:

1. First, create and save a report using the Report Wizard.
2. In the Object Navigator, under the **Reports** node, click the report you want to convert to character mode.
3. Choose **Tools > File Conversion**.
4. In the Convert dialog box, on the **Conversion** page:
   - set **Document Type** to Report.
   - set **Source** to the name of the existing bit-mapped report.
   - set **Destination Type** to Report Binary File (RDF).
   - set **Destination** to the name of the new character-mode report.
5. On the **Options** page, set **Destination Unit** to Character.
6. Click **OK**.
7. Set properties for your new character-mode report.

See also

Section 2.8.13, "About creating an ASCII (character-mode) report"

Section 3.2.5, "Setting properties for an ASCII (character-mode) report"

3.5.10 Preparing a multiplatform report

To prepare a report to run on multiple platforms, consider the following GUI differences:

Fonts: A font type, style, or size might not be available in the target GUI. You can handle this in one of two ways:

- Use a font that you know exists on the target GUI or one that maps well to the default font of the target GUI.
- Modify the font mapping file, `uifont.ali`, to ensure that the fonts map correctly.

Colors: A color might not be available in the target GUI. If possible, use a color that you know exists on the target GUI; otherwise, use one that maps well to the default color of the target GUI. The following colors are typically available on many platforms: blue, magenta, red, cyan, green, yellow.

DPI: The dots-per-inch (DPI) that your monitor uses may not be the same as the DPI used by the person who runs the report. The DPI only affects how alpha-numeric characters word-wrap on the screen. If you design a report that may be displayed in the Paper Design view, try to use the same DPI as the people who will run it. Also avoid giving layout objects fixed sizing.

3.5.11 Preparing a report for translation into other languages

For detailed information about National Language Support (NLS), review the topics in the **National Language Support** section of the *Reports Builder online help*.

3.6 Work with Web Reports

This section provides procedures for the following tasks that you may perform as you work with Web reports:

- Creating a Web report
- Viewing the source code for a Web report
- Adding report data to an existing Web page (HTML file)
- Adding a report block to a Web page
- Adding a graph to a Web report
- Preparing a paper-based report for the Web
- Adding Web links to paper-based reports

See also

Section 1.2.2, "About Web reports"

3.6.1 Creating a Web report

You can create a Web report in any of the following ways:

- Create a report using the Report Wizard, (see Section 3.5.1, "Creating a report"), choosing to create a report for both Web and paper, or Web only.
- Insert a report in a Web page by adding report data, then adding a report block to a Web page, as described in Section 3.6.3, "Adding report data to an existing Web page (HTML file)" and Section 3.6.4, "Adding a report block to a Web page".
- Display the Web Source view and manually insert the Oracle Reports custom JSP tags (see the **Reference** section of the *Reports Builder online help*).
- Insert an existing report into an existing Web page, by displaying the Web Source view for both the report and the Web page, then copying and pasting the report block into the desired position in the Web page.
- Prepare a paper-based report for the Web by adding hyperlinks to an existing report (see Section 3.6.6, "Preparing a paper-based report for the Web").

3.6.2 Viewing the source code for a Web report

To view the source code for a Web report:

- In the Object Navigator, double-click the view icon next to the **Web Source** node for the report to display the source code in the Web Source view.

See also

Topic "Custom JSP tags for Oracle Reports" in the **Reference** section of the *Reports Builder online help*.

3.6.3 Adding report data to an existing Web page (HTML file)

To retrieve the data to be used to build a report to add to an existing Web page (HTML file):

1. In the Object Navigator, choose **File > Open**.
2. Navigate to your Web page (HTML file), and click **Open**.
3. In the Object Navigator, find the report created when you opened the HTML file (Reports Builder may give the report a default name, such as **REP1**), then double-click the view icon next to the **Data Model** node to display the Data Model view.

   Caution: If you right-click the **Data Model** node, then choose **Report Wizard** to use the Report Wizard to select the data for a report, the report will overwrite all of the data in your existing Web page.

4. In the Data Model view, choose **Insert > Query** to select data for the report using the Data Wizard.

   After you click **Finish**, the data you have selected is available to your Web report.

Next step

See Section 3.6.4, "Adding a report block to a Web page"

3.6.4 Adding a report block to a Web page

To add a report block to an existing Web page:

1. In the Object Navigator, choose **File > Open**.
2. Navigate to your Web page (HTML file), and click **Open**.
3. In the Object Navigator, find the report created when you opened the HTML file (Reports Builder may give the report a default name, such as **REP1**).
4. If the Web page has never had a report added to it, you must first add the data to be used to build the report, as described above.
5. After you have added data to be used to build the report, double-click the view icon next to the **Web Source** node to display the source code for the Web page in the Web Source view.
6. Locate the section in the source code where you want to add the report block.

> **Note:** Adding some comment text such as "Place the report block here" to your Web page allows you to easily locate the correct position for your report block.

7. Choose **Insert > Report Block**.
8. In the Report Block Wizard, specify the information for the report block.

3.6.5 Adding a graph to a Web report

See Section 3.9.8.4, "Adding a graph to a Web report"

See also

Section 3.9.8.5, "Editing a graph in a Web report"

Section 3.9.8.2, "Adding a graph to a paper report"

3.6.6 Preparing a paper-based report for the Web

To prepare a paper-based report for the Web:

1. Optionally, add desired Web links to a new or existing report:
   - Create an HTML document header (see Section 3.6.7.1.1
   - Create an HTML document footer (see Section 3.6.7.1.2)
   - Create an HTML page header (see Section 3.6.7.1.3
   - Create an HTML page footer (see Section 3.6.7.1.4)
   - Create an HTML Parameter Form header (see Section 3.6.7.1.5)
   - Create an HTML Parameter Form footer (see Section 3.6.7.1.6)
   - Create a hyperlink (and, if linking to an object within the report, create an associated hyperlink destination) (see Section 3.6.7.1.7 and Section 3.6.7.1.8)
   - Create an application command line link (PDF output only) (see Section 3.6.7.1.9)
   - Create a bookmark (see Section 3.6.7.1.10)
2. Select a printer or use the default printer setup (the drivers for the currently selected printer are used to produce the output; you must have a printer configured for the machine on which you are running the report).
3. Display your report output in your Web browser (see Section 3.7.15.4, "Displaying report output in your Web browser"

See also

Section 1.2.2, "About Web reports"

Section 2.2.3, "About Web links for HTML output"

Section 2.2.4, "About Web links for PDF output"

3.6.7 Adding Web links to paper-based reports

This section provides procedures for adding links to paper-based reports that will become active when you run your report to an HTML or PDF file. This is completely separate from HTML that might be generated when you run a JSP-based Web report (when you click the Run Web Layout button in the toolbar, or choose **Program > Run Web Layout**).

You can add Web links to paper-based reports using either:

- the user interface.
- PL/SQL.

3.6.7.1 Using the user interface

This section provides procedures for the following tasks that you may perform using the Reports Builder user interface to add Web links to paper-based reports:

- Creating an HTML document header using the Property Inspector
- Creating an HTML document footer using the Property Inspector
- Creating an HTML page header using the Property Inspector
- Creating an HTML page footer using the Property Inspector
- Creating an HTML Parameter Form header using the Property Inspector
- Creating an HTML Parameter Form footer using the Property Inspector
- Creating a hyperlink using the Property Inspector
- Creating a hyperlink destination using the Property Inspector
- Creating an application command line link using the Property Inspector
- Creating a bookmark using the Property Inspector
- Creating a bookmark on break columns using the Property Inspector
- Adding navigation controls for HTML page-streamed output using the Property Inspector
- Linking an image object to a URL
- Creating a boilerplate text object for HTML tags
- Linking an HTML object to a file
- Selecting HTML tags from the database

See also

Section 1.2.2, "About Web reports"

Section 2.2.3, "About Web links for HTML output"

Section 2.2.4, "About Web links for PDF output"

Section 2.2.9, "About before and after escapes"

3.6.7.1.1 Creating an HTML document header using the Property Inspector

Note: This procedure is for HTML output only.

To insert an HTML file or text on the header page of your HTML document:

1. In the Object Navigator, double-click the properties icon for the report to display the Property Inspector.
2. Under the **Report Escapes** node, set the Before Report Type property to Text (if you will type the header) or File (if you will import the header from a file).
3. Set the Before Report Value property by clicking the**...** button to either type HTML code in the dialog box or select an HTML file to import.

See also

Section 3.6.7.2.1, "Creating an HTML document header using PL/SQL"

3.6.7.1.2 Creating an HTML document footer using the Property Inspector

Note: This procedure is for HTML output only.

To insert an HTML file or text on the footer page of your HTML document:

1. In the Object Navigator, double-click the properties icon for the report to display the Property Inspector.
2. Under the **Report Escapes** node, set the After Report Type property to Text (if you will type the footer) or File (if you will import the footer from a file).
3. Set the After Report Value property by clicking the**...** button to either type HTML code in the dialog box or select an HTML file to import.

See also

Section 3.6.7.2.2, "Creating an HTML document footer using PL/SQL"

3.6.7.1.3 Creating an HTML page header using the Property Inspector

> **Note:** This procedure is for HTML output only.

To add a page header to *every page* of your HTML document:

1. In the Object Navigator, double-click the properties icon for the report to display the Property Inspector.
2. Under the **Report Escapes** node, set the Before Page Type property to Text (if you will type the header) or File (if you will import the header from a file).
3. Set the Before Page Value property by clicking the**...** button to either type HTML code in the dialog box or select an HTML file to import.

To add a page header to a *single page* of your HTML document:

- See Section 3.6.7.2.3, "Creating an HTML page header using PL/SQL".

3.6.7.1.4 Creating an HTML page footer using the Property Inspector

> **Note:** This procedure is for HTML output only.

To add a page footer to *every page* of your HTML document:

1. In the Object Navigator, double-click the properties icon for the report to display the Property Inspector.
2. Under the **Report Escapes** node, set the After Page Type property to Text (if you will type the footer) or File (if you will import the footer from a file).
3. Set the After Page Value property by clicking the**...** button to either type HTML code in the dialog box or select an HTML file to import.

To add a page footer to a *single page* of your HTML document:

- See Section 3.6.7.2.4, "Creating an HTML page footer using PL/SQL".

3.6.7.1.5 Creating an HTML Parameter Form header using the Property Inspector

Note: This procedure is for HTML output only.

To add items to the top of the HTML Parameter Form:

1. In the Object Navigator, double-click the properties icon next to the report name to display the Property Inspector.
2. Under the **Report Escapes** node, set the Before Form Type property to Text (if you will type the header) or File (if you will import the header from a file).
3. Set the Before Form Value property by clicking the**...** button to either type HTML code in the dialog box or select an HTML file to import.

See also

Section 2.5.1, "About Parameter Form HTML extensions"

Section 3.6.7.2.5, "Creating an HTML Parameter Form header using PL/SQL"

3.6.7.1.6 Creating an HTML Parameter Form footer using the Property Inspector

Note: This procedure is for HTML output only.

To add items to the bottom of the HTML Parameter Form:

1. In the Object Navigator, double-click the properties icon next to the report name to display the Property Inspector.
2. Under the **Report Escapes** node, set the After Form Type property to Text (if you will type the footer) or File (if you will import the footer from a file).
3. Set the After Form Value property by clicking the **...** button to either type HTML code in the dialog box or select an HTML file to import.

See also

Section 2.5.1, "About Parameter Form HTML extensions"

Section 3.6.7.2.6, "Creating an HTML Parameter Form footer using PL/SQL"

3.6.7.1.7 Creating a hyperlink using the Property Inspector

To add a hyperlink to your report:

1. Create a hyperlink destination, which will be the target of the Web link.
2. In the Object Navigator, expand the **Paper Layout** node, then expand the node that contains the object that will be the source of the Web link.

   Note: If you are defining a template, you can select objects in the margin. Objects in the body are unknown until the template is applied to a report.

3. Double-click the object that will be the source of the Web link to display the Property Inspector.
4. Under the **Web Settings** node, set the Hyperlink property to the destination of the link.

   For examples, see the description of the Hyperlink property in the **Reference** section of the *Reports Builder online help*.

   Note: A report output in PDF format can include both hyperlinks and application command line links. If the Application Command Line property is set for an object, it will be applied to the object. Otherwise, the Hyperlink property is applied.

5. To apply additional HTML attributes to the hyperlink, set the Additional Hyperlink Attributes property accordingly.

See also

Section 2.2.5, "About hyperlinks"

Section 3.6.7.2.7, "Creating a hyperlink using PL/SQL"

3.6.7.1.8 Creating a hyperlink destination using the Property Inspector

To add a hyperlink destination to your report:

1. In the Object Navigator, expand the **Paper Layout** node, then expand the node that contains the object that will be the destination of a Web link.

   Note: If you are defining a template, you can select objects in the margin. Objects in the body are unknown until the template is applied to a report.

2. Double-click the object that will be the destination of a Web link to display the Property Inspector.
3. Under the **Web Settings** node, set the Hyperlink Destination property to an identifier for the object.

See also

Section 2.2.6, "About hyperlink destinations"

Section 3.6.7.2.8, "Creating a hyperlink destination using PL/SQL"

3.6.7.1.9 Creating an application command line link using the Property Inspector

Note: This procedure is for PDF output only.

To associate a command with an object in your report:

1. In the Object Navigator, expand the **Paper Layout** node, then expand the node that contains the object with which you want to associate a command.

   Note: If you are defining a template, you can select objects in the Margin. Objects in the body are unknown until the template is applied to a report.

2. Double-click the object with which you want to associate a command to display the Property Inspector.
3. Under the **Web Settings** node, set the Application Command Line property to the command you want to execute when the object is clicked.

Note: A report output in PDF format can include both hyperlinks and application command line links. If the Application Command Line property is set for an object, it will be applied to the object. Otherwise, the Hyperlink property is applied.

Restrictions

An object that is associated with an application command line link cannot also be the source of a Web link (a hyperlink).

See also

Section 2.2.8, "About application command line links"

Section 3.6.7.2.9, "Creating an application command line link using PL/SQL"

3.6.7.1.10 Creating a bookmark using the Property Inspector

To create a bookmark on objects *other than break columns* in the bookmark area of your HTML or PDF document:

1. In the Object Navigator, expand the **Paper Layout** node, then expand the node that contains the object with which you want to associate a bookmark (typically, the object is a repeating frame or frame that encloses the relevant section of the report).

   Note: If you are defining a template, you can select objects in the Margin. Objects in the body are unknown until the template is applied to a report.

2. Double-click the object with which you want to associate a bookmark to display the Property Inspector, and set properties:
   - Under the **Web Settings** node, set the Bookmark property to the string you want to appear in the bookmark area of the formatted report.

See also

Section 2.2.7, "About bookmarks"

Section 3.6.7.2.10, "Creating a bookmark using PL/SQL"

3.6.7.1.11 Creating a bookmark on break columns using the Property Inspector

To create a bookmark on break columns in the bookmark area of your paginated HTMLCSS or paper PDF document:

1. In the Paper Layout view, choose **Insert > Bookmark**.
2. In the Insert Bookmarks dialog box, move the desired column(s) to the Bookmarks list.
3. Click **OK**.

See also

Section 2.2.7, "About bookmarks"

Section 3.6.7.2.10, "Creating a bookmark using PL/SQL"

3.6.7.1.12 Adding navigation controls for HTML page-streamed output using the Property Inspector

> **Note:** This procedure is for HTML output only.

To add navigation controls for HTML page-streamed (paginated) output:

1. In the Object Navigator, double-click the properties icon next to the report name to display the Property Inspector.
2. Set the Page Navigation Control Type and Page Navigation Control Value properties.

> **Note:** If you do not change the default Page Navigation Control Value, Reports Builder will use a default built-in JavaScript for implementing the navigation between the pages of output.

See also

Section 2.8.8.1, "About HTML page streaming"

Section 3.7.15.5, "Displaying individual pages of HTML report output"

Section 3.6.7.2.11, "Adding navigation controls for HTML page-streamed output using PL/SQL"

3.6.7.1.13 Linking an image object to a URL

> **Note:** This procedure is for HTML output only.

To link an image object to a URL:

1. In the Paper Design view or Paper Layout view, click the Link File tool in the tool palette
2. Click and drag a rectangle.
3. Double-click the link file object to display the Property Inspector.
4. Under the **Link File Boilerplate** node:
   - set the Source File Format property to *Image URL*.
   - set the Source Filename property to the URL where the image is located with the required protocol.

Example 1

```
HTTP://www.oracle.com/images/logo.gif
```

Example 2

```
HTTP://&<P_SERVER_NAME>/images/logo.gif
where P_SERVER_NAME is a user parameter of type CHAR
At runtime, the end user can specify a value for the parameter (e.g., P_SERVER_
NAME = www.oracle.us.com for a dynamic URL link of
HTTP://www.oracle.us.com/images/logo.gif).
```

Example 3

```
FILE://c:/images/logo.gif
```

> **Note:** If you click Browse to find a file, Reports Builder automatically prepends FILE:// to the returned path.

See also

Section 2.4.3, "About images"

Section 1.8.5, "About boilerplate objects"

Section 3.9.8.9, "Linking an image object to a file"

Section 3.9.8.8, "Selecting an image URL from the database"

3.6.7.1.14 Creating a boilerplate text object for HTML tags

> **Note:** This procedure is for HTML output only.

To create a boilerplate text object for HTML tags:

1. In the Paper Design view, click the Text tool in the tool palette.
2. Click and drag a rectangle.
3. In the new boilerplate object, type the desired HTML.

> **Note:** The text you type will format for HTML output only; for all other output formats, the text object will not appear in the report.

4. Click outside the boilerplate text object.
5. Double-click the boilerplate text object (or right-click the object, and choose **Property Inspector**).
6. In the Property Inspector, under the **Web Settings** node, set the Contains HTML Tags property to Yes.

Example 1

This example shows a boilerplate text object tagged as a hyperlink.

Create the following as a boilerplate text object:

```
<a href=http://your_webserver/reports/my_report.html><img src=oracle.gif> </a>
```

Double-click the boilerplate object to display the Property Inspector. Under the **Web Settings** node, set the Contains HTML tags property to *Yes*.

Example 2

In a boilerplate text object, you can type the following Java Applet, called NervousText.class, which takes the object width and height as parameters:

```
<base href=http://cagney.uk.oracle.com/java/NervousText/>
<applet code="NervousText.class" width=&ObjectWidth height=&<ObjectHeight>>
<param name=text value="&deptno">
</applet>
```

This is output to HTML as:

```
<base href=http://cagney.uk.oracle.com/java/NervousText/>
<applet code="NervousText.class" width=84 height=72>
<param name=text value="10">
</applet>
```

See also

Section 1.8.5, "About boilerplate objects"

Section 2.5.1, "About Parameter Form HTML extensions"

3.6.7.1.15 Linking an HTML object to a file

> **Note:** This procedure is for HTML output only.

To link an HTML object to a file:

1. In the Paper Design view or Paper Layout view, click the Link File tool in the tool palette.
2. Click and drag a rectangle.
3. Double-click the link file object to display the Property Inspector.
4. Under the **Link File Boilerplate** node:
   - set the Source File Format property to Text.
   - set the Source Filename property to the name of the file containing the HTML tags.
5. Under the **Web Settings** node, set the Contains HTML Tags property to Yes.

See also

Section 1.8.5, "About boilerplate objects"

3.6.7.1.16 Selecting HTML tags from the database

> **Note:** This procedure is for HTML output only.

To select HTML tags from the database:

1. Create a query, with a SELECT statement that selects a column containing HTML tags or the names of files that contain HTML tags.
2. In the Data Model view, double-click the HTML tags column to display the Property Inspector.
3. If the column contains the names of files that contain the HTML tags, rather than the HTML tags themselves:
   - Under the **Column** node, set the Read from File property to Yes.
   - Set the File Format property to Text.
4. In the Paper Layout view, create a field object that references the column.
5. Double-click the field object to display the Property Inspector.
6. In the Property Inspector, under the **Web Settings** node, set the Contains HTML Tags property to Yes.

3.6.7.2 Using PL/SQL

This section provides procedures for the following tasks that you may perform as you work with PL/SQL to add Web links to paper-based reports:

- Creating an HTML document header using PL/SQL
- Creating an HTML document footer using PL/SQL
- Creating an HTML page header using PL/SQL
- Creating an HTML page footer using PL/SQL
- Creating an HTML Parameter Form header using PL/SQL
- Creating an HTML Parameter Form footer using PL/SQL
- Creating a hyperlink using PL/SQL
- Creating a hyperlink destination using PL/SQL
- Creating an application command line link using PL/SQL

- Creating a bookmark using PL/SQL
- Adding navigation controls for HTML page-streamed output using PL/SQL

See also

Section 1.2.2, "About Web reports"

Section 2.2.3, "About Web links for HTML output"

Section 2.2.4, "About Web links for PDF output"

Section 2.2.9, "About before and after escapes"

3.6.7.2.1 Creating an HTML document header using PL/SQL

> **Note:** This procedure is for HTML output only. A before report escape should be set in a trigger that fires before the report starts formatting, such as the Before Report trigger.

To add items to the header page of your HTML document using PL/SQL instead of the Property Inspector:

1. In the Object Navigator, expand the **Report Triggers** node.
2. Double-click the PL/SQL icon next to the Before Report trigger.
3. In the PL/SQL Editor, include SRW.SET_BEFORE_REPORT_HTML to define the PL/SQL for the format trigger.

See also

Section 3.6.7.1.1, "Creating an HTML document header using the Property Inspector"

3.6.7.2.2 Creating an HTML document footer using PL/SQL

> **Note:** This procedure is for HTML output only. An after report escape should be set in a trigger that fires before the report is done formatting, such as the Before Report trigger.

To add items to the footer page of your HTML document using PL/SQL instead of the Property Inspector:

1. In the Object Navigator, expand the **Report Triggers** node.
2. Double-click the PL/SQL icon next to the Before Report trigger.
3. In the PL/SQL Editor, include SRW.SET_AFTER_REPORT_HTML to define the PL/SQL for the format trigger.

See also

Section 3.6.7.1.2, "Creating an HTML document footer using the Property Inspector"

3.6.7.2.3 Creating an HTML page header using PL/SQL

Note: This procedure is for HTML output only. If you want the escape to apply to every page, set it in a trigger that fires before the report begins formatting, such as the Before Report trigger. If you want the escape to apply to a single page, set it in a Format trigger

---

To add a page header to *every page* of your HTML document using PL/SQL instead of the Property Inspector:

1. In the Object Navigator, expand the **Report Triggers** node.
2. Double-click the PL/SQL icon next to the Before Report trigger.
3. In the PL/SQL Editor, include SRW.SET_BEFORE_PAGE_HTML to define the PL/SQL for the format trigger.

To add a page header to a *single page* of your HTML document using PL/SQL:

1. In the Object Navigator, expand the **Paper Layout** node, then expand the node that contains the object that should fire the trigger when formatted.
2. Double-click the PL/SQL icon next to the object that should fire the trigger when formatted.
3. In the PL/SQL Editor, include SRW.SET_BEFORE_PAGE_HTML to define the PL/SQL for the format trigger.

See also

Section 3.6.7.1.3, "Creating an HTML page header using the Property Inspector"

3.6.7.2.4 Creating an HTML page footer using PL/SQL

> **Note:** This procedure is for HTML output only. If you want the escape to apply to every page, set it in a trigger that fires before the report begins formatting, such as the Before Report trigger. If you want the escape to apply to a single page, set it in a Format trigger.

To add a page footer to *every page* of your HTML document using PL/SQL instead of the Property Inspector:

1. In the Object Navigator, expand the **Report Triggers** node.
2. Double-click the PL/SQL icon next to the **Before Report** trigger.
3. In the PL/SQL Editor, include SRW.SET_AFTER_PAGE_HTML to define the PL/SQL for the format trigger.

To add a page footer to a *single page* of your HTML document using PL/SQL:

1. In the Object Navigator, expand the **Paper Layout** node, then expand the node that contains the object that should fire the trigger when formatted.
2. Double-click the PL/SQL icon next to the object that should fire the trigger when formatted.
3. In the PL/SQL Editor, include SRW.SET_AFTER_PAGE_HTML to define the PL/SQL for the format trigger.

See also

Section 3.6.7.1.4, "Creating an HTML page footer using the Property Inspector"

3.6.7.2.5 Creating an HTML Parameter Form header using PL/SQL

> **Note:** This procedure is for HTML output only. A before form escape should be set in a trigger that fires before the Parameter Form starts formatting, such as the Before Parameter Form trigger.

To add items to the top of the HTML Parameter Form using PL/SQL instead of the Property Inspector:

1. In the Object Navigator, expand the **Report Triggers** node.
2. Double-click the PL/SQL icon next to the **Before Parameter Form** trigger.

3. In the PL/SQL Editor, include SRW.SET_BEFORE_FORM_HTML to define the PL/SQL for the format trigger.

See also

Section 2.5.1, "About Parameter Form HTML extensions"

Section 3.6.7.1.5, "Creating an HTML Parameter Form header using the Property Inspector"

3.6.7.2.6 Creating an HTML Parameter Form footer using PL/SQL

> **Note:** This procedure is for HTML output only. A before form escape should be set in a trigger that fires before the Parameter Form starts formatting, such as the Before Parameter Form trigger.

To add items to the bottom of the HTML Parameter Form using PL/SQL instead of the Property Inspector:

1. In the Object Navigator, expand the **Report Triggers** node.
2. Double-click the PL/SQL icon next to the **Before Parameter Form** trigger.
3. In the PL/SQL Editor, include SRW.SET_AFTER_FORM_HTML to define the PL/SQL for the format trigger.

See also

Section 2.5.1, "About Parameter Form HTML extensions"

Section 3.6.7.1.6, "Creating an HTML Parameter Form footer using the Property Inspector"

3.6.7.2.7 Creating a hyperlink using PL/SQL

To add a hyperlink to your report using PL/SQL instead of the Property Inspector:

1. In the Object Navigator, expand the **Paper Layout** node, then expand the node that contains the object that will be the source of the Web link.

> **Note:** If you are defining a template, you can select objects in the Margin. Objects in the body are unknown until the template is applied to a report

2. Double-click the PL/SQL icon next to the object that will be the source of the Web link.
3. In the PL/SQL Editor, include SRW.SET_HYPERLINK to define the PL/SQL for the format trigger.
4. To apply additional HTML attributes to the hyperlink, use SRW.SET_HYPERLINK_ATTRS.

Next step

Section 3.6.7.2.8, "Creating a hyperlink destination using PL/SQL"

See also

Section 2.2.5, "About hyperlinks"

Section 3.6.7.1.7, "Creating a hyperlink using the Property Inspector"

3.6.7.2.8 Creating a hyperlink destination using PL/SQL

To add a hyperlink destination to your report using PL/SQL instead of the Property Inspector:

1. In the Object Navigator, expand the **Paper Layout** node, then expand the node that contains the object that will be the destination of a Web link.

> **Note:** If you are defining a template, you can select objects in the Margin. Objects in the body are unknown until the template is applied to a report.

2. Double-click the PL/SQL icon next to the object that will be the destination of a Web link.
3. In the PL/SQL Editor, include SRW.SET_LINKTAG to define the PL/SQL for the format trigger.

Next step

Section 3.6.7.2.7, "Creating a hyperlink using PL/SQL"

See also

Section 2.2.6, "About hyperlink destinations"

Section 3.6.7.1.8, "Creating a hyperlink destination using the Property Inspector"

3.6.7.2.9 Creating an application command line link using PL/SQL

Note: This procedure is for PDF output only.

To associate a command with an object in your report using PL/SQL instead of the Property Inspector:

1. In the Object Navigator, expand the **Paper Layout** node, then expand the node that contains the object with which you want to associate an application command line link.

   Note: If you are defining a template, you can select objects in the Margin. Objects in the body are unknown until the template is applied to a report.

2. Double-click the PL/SQL icon next to the object with which you want to associate an application command line link.
3. In the PL/SQL Editor, include SRW.SET_PDF_ACTION to define the PL/SQL for the format trigger.

Restrictions

An object that is associated with an application command line link cannot also be the source of a Web link (a hyperlink).

See also

Section 2.2.8, "About application command line links"

Section 3.6.7.1.9, "Creating an application command line link using the Property Inspector"

3.6.7.2.10 Creating a bookmark using PL/SQL

To add a bookmark string in the bookmark area of your report using PL/SQL:

1. In the Object Navigator, expand the **Paper Layout** node, then expand the node that contains the object with which you want to associate a bookmark (typically, the object is a repeating frame or frame that encloses the relevant section of the report).

Note: If you are defining a template, you can select objects in the Margin. Objects in the body are unknown until the template is applied to a report.

2. Double-click the PL/SQL icon next to the object with which you want to associate a bookmark.
3. In the PL/SQL Editor, include SRW.SET_BOOKMARK to define the PL/SQL for the format trigger.

See also

Section 2.2.7, "About bookmarks"

Section 3.6.7.1.10, "Creating a bookmark using the Property Inspector"

Section 3.6.7.1.11, "Creating a bookmark on break columns using the Property Inspector"

3.6.7.2.11 Adding navigation controls for HTML page-streamed output using PL/SQL

Note: This procedure is for HTML output only.

To add navigation controls for HTML page-streamed output using PL/SQL instead of the Property Inspector:

- in a Before Report trigger, use the SRW.SET_PAGE_NAVIGATION_HTML PL/SQL procedure

See also

Section 2.8.8.1, "About HTML page streaming"

Section 3.7.15.5, "Displaying individual pages of HTML report output"

Section 3.6.7.1.12, "Adding navigation controls for HTML page-streamed output using the Property Inspector"

3.7 Run and Dispatch a Report

This section provides procedures for the following tasks that you may perform as you run and dispatch a report:

- Running and dispatching a report from the user interface
- Running a report from the command line
- Running a report using a command file
- Running a report to a remote Reports Server
- Generating HTML or HTMLCSS output
- Generating XML output
- Generating RTF output
- Generating text output
- Generating delimited output
- Distributing a report to multiple destinations
- Deploying a report
- Changing orientation
- Suppressing the Parameter Form
- Viewing report output
- Printing a report
- E-mailing a report

3.7.1 Running and dispatching a report from the user interface

To run and dispatch a report:

1. In the Object Navigator, click the report name, or choose **File > Open** to open it.
2. If you are not already connected, connect to a database.
3. Click the Run Paper Layout button in the toolbar.
4. If the Runtime Parameter Form displays, specify settings for the desired output:

Note: When you create the report, you must use the Parameter Form Builder (choose **Tools > Parameter Form Builder**) to select the system and user parameters you want to appear in the Runtime Parameter Form.

- To preview output in the Paper Design view using screen fonts: set **Destination Type** to *Screen*.
- To preview output in the Paper Design view using screen fonts and size the fonts using the printer font metrics: set **Destination Type** to *Preview*.
- To send output to a file: set **Destination Type** to *File*, **Destination Name** to the path and filename, **Destination Format** to the output format for the report (e.g., *PDF*, *HTML*, *HTMLCSS*, *RTF*, *XML*, or *DELIMITED* for bitmapped reports, or the character-mode printer definition file (.prt file) for character-mode reports).

Note: You can also automatically display report output in your Web browser (see Section 3.7.15.4, "Displaying report output in your Web browser"), then save the temporary file as an HTML or PDF file.

Note: When you view the report output in Microsoft Word, you must choose **View > Page Layout** to see all the graphics and objects in your report.

- To send output to a printer: set **Destination Type** to *Printer*, **Destination Name** to the printer driver name.
- To send output via e-mail using your Internet standard protocol SMTP e-mail application: set **Destination Type** to *Mail*, leave **Destination Name** blank. In the Mail dialog box, specify the recipients and subject line for your e-mail report.

Note: The configuration file `rwbuilder.conf` must include the pluginParam mailServer with the outgoing mail server name. This applies in both the Windows NT and Solaris environments.

5. Optionally, display output in your Web browser (see Section 3.7.15.4, "Displaying report output in your Web browser").
6. In the Runtime Parameter Form, click the Run Report button in the toolbar.
7. If a dialog box appears, enter required information.

Usage notes

If you want the orientation of the report to be landscape, you must change the orientation for the report (see Section 3.7.13, "Changing orientation"). However, if your printer does not support the specified page size, you may get truncated results. To enable character-mode printers to support landscape (or other page sizes), you can set printer escape sequences in the printer definition files. Refer to your printer documentation for the escape sequences to use.

See also

Section 3.11.6, "Selecting parameters to include in the Runtime Parameter Form"

3.7.2 Running a report from the command line

To run a runfile (.rep) or a report definition file (.rdf) from the command line:

1. Type %ORACLE_HOME%\BIN\rwrun, followed by the report name and desired arguments.
2. If you sent the output to a file (DESTYPE=FILE DESNAME=*filename* DESFORMAT=*fileformat*), open or print the file to verify the output.

See also

Section 1.12, "Executables"

Topic "rwrun" in the **Reference** section of the *Reports Builder online help*.

Section 2.8.2, "About batch reporting"

3.7.3 Running a report using a command file

To run a report using a command file:

1. Create a text file using a word processor of your choice.
2. In the file, type %ORACLE_HOME%\BIN\rwrun, followed by the report name and desired arguments.
3. Name the file *filename*.bat.

4. To run the report, type *filename.bat* on the command line.
5. If you sent the output to a file (DESTYPE=FILE DESNAME=*filename* DESFORMAT=HTML), open or print the file to verify the output.

See also

Section 2.8.2, "About batch reporting"

Section 1.12, "Executables"

Topic "rwrun" in the **Reference** section of the *Reports Builder online help*.

3.7.4 Running a report to a remote Reports Server

To run a report to a remote Reports Server:

- Type %ORACLE_HOME%\bin\rwclient, followed by the report name, server name, and desired arguments.

For additional information, see the *Oracle Application Server Reports Services Publishing Reports to the Web* manual, available on the Oracle Technology Network Oracle Reports Documentation page (http://otn.oracle.com/docs/products/reports/content.html).

3.7.5 Generating HTML or HTMLCSS output

To generate HTML or HTMLCSS output:

1. In the Object Navigator, select or open the report, then click the Run Paper Layout button in the toolbar to run the report.
2. In the Paper Design view, specify the scope of output you want to display:
   - To display only the current page of the report, choose **File > Preview Format > Current Page**.
   - To display all pages, choose **File > Preview Format > All Pages**.

Note: If you choose to show all pages of your report output, and you have HTML page streaming enabled, the first page will display in your Web browser, and you will be able to quickly navigate to any other page in the rest of the report.

---

3. To preview your HTML or HTMLCSS report output in a Web browser, choose **File > Preview Format > Paginated HTML** or **Paginated HTMLCSS** (to format with style sheet extensions).
4. To preview your report output with Web layout in a Web browser, choose **Program > Run Web Layout**.

> **Note:** If Netscape 7.0 is your default browser, the browser may not display. You can work around this bug by making a copy of the Netscape 7.0 executable, naming it `netscape.exe`; with this name, the browser will display as expected.

5. To save your report output as an HTML or HTMLCSS file, choose **File > Generate to File**.

See also

Section 2.8.8, "About HTML output"

Section 2.8.8.1, "About HTML page streaming"

Section 2.2.10, "About style sheets"

3.7.6 Generating PDF output

To generate PDF output:

1. In the Object Navigator, select or open the report, then click the Run Paper Layout button in the toolbar to run the report.
2. In the Paper Design view, specify the scope of output you want to display:
   - To display only the current page of the report, choose **File > Preview Format > Current Page**.
   - To display all pages, choose **File > Preview Format > All Pages**.
3. To preview your PDF report output in your Web browser, choose **File > Preview Format > PDF**.
4. To preview your report output with Web layout in a Web browser, choose **Program > Run Web Layout**.

Note: If Netscape 7.0 is your default browser, the browser may not display. You can work around this bug by making a copy of the Netscape 7.0 executable, naming it `netscape.exe`; with this name, the browser will display as expected.

5. To save your report output as a PDF file, choose **File > Generate to File > PDF**.

See also

Section 2.8.9, "About PDF output"

3.7.7 Generating XML output

In any Reports Builder-generated XML file, your output mimics the data model; that is, it is structured by groups and columns. Follow the steps below to:

- produce an XML output file from your current report
- preview the contents of an XML file in a Web browser in conjunction with command line invocation:
- change the XML properties that control XML output for the entire report
- change the XML properties that control XML output for a column or group

To produce an XML output file from your current report:

1. Optionally, in the Object Navigator, select or open the report, then click the Run Paper Layout button in the toolbar to run the report. Currently, any report style can be generated to XML output, but with limited support for matrix-style reports.
2. Optionally, in the Paper Design view, choose **File > Preview Format > XML** to preview XML data in your default XML viewer (e.g., your Web browser).

Tip: An .XML extension is required by XML-supporting browsers to recognize the format as XML and display it accordingly.

3. Click the report in the Object Navigator, and choose **File > Generate to File > XML** to save the XML output file to the directory of your choice.

Notes:

- Reports Builder creates "well-formed" XML output. This means that there is an XML Declaration present in the prolog of the XML document, but not necessarily a DTD (document type definition). If you wish to create "valid" XML, you must add the DTD in the XML Prolog Value property field at the report level.
- If your report includes a graph, the graph information will not be saved in the XML file (XML is text-based output, which means images are not included).

To preview the contents of an XML file in a Web browser in conjunction with command line invocation:

1. On the command line, execute your report with the arguments `DESFORMAT=XML`, `DESTYPE=FILE`, and `DESNAME=`*`filename`*`.XML`.
2. Open an XML-supporting Web browser.
3. Choose **File > Open** to navigate to the XML file that you just generated. The XML file will appear in the browser window.
4. If you make changes to the XML output in your report and regenerate the XML file with the same name, click the Reload button in your browser to view the updated XML output.

To change the XML properties of the entire report:

1. In the Object Navigator double-click the properties icon next to the report name to display the Property Inspector.
2. Under **XML Settings**, set the XML properties as required for this report.

To change the XML properties of a column or group:

1. In the Data Model view, double-click the column or the title bar of the group to display the Property Inspector.
2. Under **XML Settings**, set the XML properties as required.

See also

Section 2.8.7, "About XML in reports"

Topic "Oracle Reports XML tags" in the **Reference** section of the *Reports Builder online help*.

The chapter "Customizing Reports with XML" in the *Oracle Application Server Reports Services Publishing Reports to the Web* manual, available on the Oracle Technology Network Oracle Reports Documentation page (`http://otn.oracle.com/docs/products/reports/content.html`).

3.7.8 Generating RTF output

To generate RTF output:

1. In the Object Navigator, select or open the report, then click the Run Paper Layout button in the toolbar to run the report.
2. To preview your report output in a Microsoft Word document, choose **File > Preview Format > RTF**.
3. To save your report output as an RTF file, choose **File > Generate to File > RTF**.
4. In the Save dialog box, specify a location and file name. Click **Save**.

See also

Section 2.8.10, "About RTF output"

3.7.9 Generating delimited output

To generate delimited output:

1. In the Object Navigator, select or open the report, then click the Run Paper Layout button in the toolbar to run the report.
2. To preview your delimited report output in a spreadsheet, choose **File > Preview Format > Delimited** or **File > Preview Format > DelimitedData** (for large volume reports).
3. To save your delimited report output to a file, choose **File > Generate** to **File > Delimited** or **File > Generate** to **File > DelimitedData** (for large volume reports).
4. In the Delimited Output dialog box or DelimitedData Output dialog box, choose from the drop-down list or type the delimiter that will separate the cells in your report output.
5. If you want to use formatting options, choose from the drop-down list or type a Date Format Mask or a Number Format Mask.

6. If you want to use a cell wrapper, choose from the drop-down list or type the cell wrapper that will be placed around the cell data in your report output.
7. Click **OK**.
8. In the Save dialog box, specify a location and file name. Click **Save**.

See also

Section 2.8.11, "About delimited output"

Section 3.9.7.1, "Specifying date and time format masks"

Section 3.9.7.2, "Specifying number format masks"

3.7.10 Generating text output

To generate text output:

1. To generate pure text output, which can be read by many different applications, set the Initial Value property of the MODE system parameter to Character (see Section 3.11.1, "Using a pre-defined system parameter").

   Note: If MODE is set to Bitmap (the default), the result is PostScript output, which can be read and rendered only by PostScript-compatible applications (such as a PostScript printer).

2. In the Object Navigator, select or open the report, then click the Run Paper Layout button in the toolbar to run the report.
3. To preview your report output in a text viewer, choose **File > Preview Format > Text**.
4. To save your report output as a text file, choose **File > Generate to File > Text**.
5. In the Save dialog box, specify a location and file name. Click **Save**.

See also

Section 2.8.12, "About text output"

3.7.11 Distributing a report to multiple destinations

To distribute a report to multiple destinations, you first *define* the distribution, then *enable* the distribution.

You can *define* the distribution for a report in any of the following ways:

- using XML, as described in the chapter "Creating Advanced Distributions" in the *Oracle Application Server Reports Services Publishing Reports to the Web* manual, available on the Oracle Technology Network Oracle Reports Documentation page (http://otn.oracle.com/docs/products/reports/content.html).
- in the Distribution dialog box to define a distribution list for the entire report and/or any report section.

  Note: To display the Distribution dialog box: In the report or section Property Inspector, under the **Report** node, double-click the Distribution property value field.

- on the command line, including the `DESTINATION` keyword to specify a DST file or XML file.

  Note: This method is supported for backward compatibility; the preferred and recommended method of distributing reports is via XML or the Distribution dialog box.

- To *enable* the distribution of a report, you can do either of the following:
  - Specify `DISTRIBUTE=YES` on the command line.
  - Choose **File > Distribute**.

You can also trace the report distribution to verify a successful distribution.

See also

Chapter 36, "Bursting and Distributing a Report"

Section 2.8.3, "About report distribution"

Section 3.14.19, "Tracing report distribution"

3.7.12 Deploying a report

To deploy a report so that end users can view it, refer to the section "Deploying Your Reports" in the chapter "Running Report Requests" in the *Oracle Application Server Reports Services Publishing Reports to the Web* manual, available on the Oracle Technology Network Oracle Reports Documentation page (`http://otn.oracle.com/docs/products/reports/content.html`). This section describes how to deploy a report with a paper layout (i.e., a REP, RDF, XML, or JSP report) and how to deploy a report with a Web layout (i.e., a JSP report).

3.7.13 Changing orientation

To change the orientation (portrait or landscape) of a report section:

1. In the Object Navigator, under the **Paper Layout** node, double-click the Header Section, Main Section, or Trailer Section properties icon for the pertinent report section (Header, Main, or Trailer) to display the Property Inspector (by default, a report is defined in the Main Section).
2. In the Property Inspector, under the **Section** node:
   - set the width and height properties as desired (e.g., 11 (or 15) width x 8.5 height for landscape or 8.5 width x 11 (or 15) height for portrait). Make sure that you subtract the margin depths (e.g., for an 8.5 x 11 page with top, bottom, left, and right margins of .25, set Width to 8 and Height to 10.5).
   - set the Orientation property to the desired value.
3. If you want to be able to modify the orientation at runtime, choose **Tools > Parameter Form Builder** to display the Parameter Form Builder:
   - Click the ORIENTATION system parameter.
   - Click **OK** to display the Paper Parameter Form view.
   - Double-click the Orientation value field (PF_ORIENTATION) to display the Property Inspector.
   - Under the **Parameter** node, set the Initial Value property to Portrait or Landscape.
   - Click the Run Paper Layout button in the toolbar.
   - From the **Orientation** drop-down list, change the orientation if desired.

See also

Section 2.1.2, "About report sectioning and sections"

3.7.14 Suppressing the Parameter Form

If the report is run from another product, you may not want the Parameter Form to appear.

To suppress the Parameter Form and have the report assign a default value to DESTYPE, do either of the following:

- Create a list parameter to explicitly pass the values of DESTYPE and PARAMFORM.
- In the Preferences dialog box (**Edit > Preferences**), on the Runtime Settings page, clear the Parameter Form check box to specify a value of No for the PARAMFORM keyword.

See also

Section 1.6.5, "About the Paper Parameter Form view"

Section 1.11.1, "About the Runtime Parameter Form"

Section 1.9.4, "About Parameter Forms for Web reports"

3.7.15 Viewing report output

This section provides procedures for various ways to view report output:

- Viewing the printable area
- Displaying report output in the Paper Design view
- Displaying report output in the Previewer
- Displaying report output in your Web browser
- Displaying individual pages of HTML report output
- Scrolling and paging
- Splitting the viewing region
- Magnifying or reducing the output

3.7.15.1 Viewing the printable area

The printable area is the physical region in which your printer can print on a physical page. Most printers usually cannot print up to the edge of a page, leaving a blank "unprintable" area.

To view the printable area of your report:

1. Choose **File > Page Setup** to specify your page settings.
2. In the Paper Layout view, click the Edit Margin button in the toolbar.

   The printable area is displayed as the area within the dotted lines in the margin area.
3. Choose **View > Zoom > Zoom Out** to see more.

3.7.15.2 Displaying report output in the Paper Design view

The Paper Design view is displayed whenever you *run a report*.

To run a report from the Object Navigator or any editor:

- Click the Run Paper Layout button or choose **Program > Run Paper Layout**.

To run a report from the Report Wizard:

- Click **Finish**.

You can also display the Paper Design view in these ways:

- Click the Paper Design View button in the toolbar.
- Choose **View > Change View > Paper Design**.
- In the Object Navigator, double-click the view icon next to the **Paper Design** node.

In the Paper Design view, manipulate the output as desired, and use any of the following features:

- scrolling and paging (see Section 3.7.15.6, "Scrolling and paging")
- splitting horizontally and vertically (see Section 3.7.15.7, "Splitting the viewing region")
- magnifying or reducing the output (see Section 3.7.15.8, "Magnifying or reducing the output")

See also

Section 3.7.16.1, "Printing a report from the Paper Design or Paper Layout view"

3.7.15.3 Displaying report output in the Previewer

To display your report output in the Previewer:

1. In the Paper Design view, choose **File > Print Preview**.
2. In the Previewer, use any of the following features:
   - scrolling and paging (see Section 3.7.15.6, "Scrolling and paging")
   - splitting horizontally and vertically (see Section 3.7.15.7, "Splitting the viewing region")
   - magnifying or reducing the output (see Section 3.7.15.8, "Magnifying or reducing the output")

See also

Section 1.11.2, "About the Previewer"

Section 3.7.16.2, "Printing a report from the Previewer"

3.7.15.4 Displaying report output in your Web browser

To display report output in your Web browser:

1. In the Object Navigator, select or open the report, then click the Run Paper Layout button in the toolbar to run the report.
2. In the Paper Design view, specify the scope of output you want to display:
   - To display only the current page of the report, choose **File > Preview Format > Current Page**.
   - To display all pages, choose **File > Preview Format > All Pages**.

   Note: If you choose to show all pages of your report output, and you have HTML page streaming enabled, the first page will display in your Web browser, and you will be able to quickly navigate to any other page in the rest of the report.

3. To preview your report output in a Web browser, choose **File > Preview Format > Paginated HTML**, **File > Preview Format > Paginated HTMLCSS** (to format with style sheet extensions), **File > Preview Format > PDF**, or **File > Preview Format > XML** (if you have an XML-supporting browser set as your default XML viewer).

4. To preview your report output with Web layout in a Web browser, choose **Program > Run Web Layout**.

> **Note:** If Netscape 7.0 is your default browser, the browser may not display. You can work around this bug by making a copy of the Netscape 7.0 executable, naming it `netscape.exe`; with this name, the browser will display as expected only.

5. To save your report output as an HTML, HTMLCSS, PDF, or XML file, choose **File > Generate to File**.

See also

Section 1.2.2, "About Web reports"

Section 2.2.2, "About previewing JSP-based Web reports"

Section 2.2.10, "About style sheets"

Section 3.7.16.3, "Printing a report from your Web browser"

3.7.15.5 Displaying individual pages of HTML report output

To paginate the HTML output for a report:

1. Specify navigation controls for HTML page-streamed output using either of the following methods:
   - in the Report Property Inspector (see Section 3.6.7.1.12, "Adding navigation controls for HTML page-streamed output using the Property Inspector")
   - using PL/SQL (see Section 3.6.7.2.11, "Adding navigation controls for HTML page-streamed output using PL/SQL")
2. Enable HTML page streaming by running your report from the command line, specifying PAGESTREAM=YES.

See also

Section 1.2.2, "About Web reports"

Section 2.8.8.1, "About HTML page streaming"

3.7.15.6 Scrolling and paging

To move around pages of the report using the scroll bars and paging buttons:

- Use the mouse cursor to drag the horizontal and vertical scroll bars to move around the current page.
- Click the First Page, Previous Page, Next Page, and Last Page buttons in the toolbar to display the indicated pages of the report.
- To display a specific page, type the page number in the **Page** field.

3.7.15.7 Splitting the viewing region

To create two views of the report output by splitting the display horizontally and/or vertically:

- In the Data Model view, Paper Layout view, Paper Design view, or Paper Parameter Form view, drag the black bar *below* the vertical scroll bar (to split horizontally), or to the *right* of the horizontal scroll bar (to split vertically).
- In the Previewer, drag the grey bar *above* the vertical scroll bar (to split horizontally), or to the *left* of the horizontal scroll bar (to split vertically)

You can scroll and page within each view to move the contents of that view while the other view remains unchanged.

3.7.15.8 Magnifying or reducing the output

To magnify a hard-to-see portion of your report:

- In the Data Model view, Paper Layout view, Paper Design view, or Paper Parameter Form view, click the Magnify tool in the tool palette (or, choose **View > Zoom > Zoom In**).
- In the Previewer, click the Zoom In button in the toolbar.

To reduce the image to get a sense of your report's overall layout:

- In the Data Model view, Paper Layout view, Paper Design view, or Paper Parameter Form view, choose **View > Zoom > Zoom Out** or **Normal Size**.
- To magnify the output displayed in the viewing region,
- In the Previewer, click the Zoom Out button in the toolbar.

3.7.16 Printing a report

This section provides procedures for the following tasks that you may perform to print a report:

- Printing a report from the Paper Design or Paper Layout view

- Printing a report from the Previewer
- Printing a report from your Web browser
- Printing a report on a pre-printed form
- Switching the printer tray
- Printing a report on UNIX

3.7.16.1 Printing a report from the Paper Design or Paper Layout view

To print a report from the Paper Design or Paper Layout view:

1. In the Paper Design view or Paper Layout view choose **File > Print.**
2. In the Print dialog box, type the number of pages and copies you want to print.
3. Click **OK**.

See also

Section 3.7.15.2, "Displaying report output in the Paper Design view"

3.7.16.2 Printing a report from the Previewer

To print a report from the Previewer:

1. In the Previewer, click the Page Setup button in the toolbar to verify your printer setup.
2. Click the Print button in the toolbar.
3. In the Print dialog box, specify the pages and copies you want to print.

   If the report has a Destination Type of *Screen*, a warning appears that you should run the report with a Destination Type of *Preview* before printing. *Preview* creates PostScript output, which is typically more desirable for printing bit-mapped reports.
4. Click **OK**.

See also

Section 3.7.15.3, "Displaying report output in the Previewer"

3.7.16.3 Printing a report from your Web browser

> **Note:** Formatting *with* style sheet extensions paginates the HTML document in the same way the report is paginated; formatting *without* style sheet extensions generates HTML output that does not break between the pages of the report.

To print a report from your Web browser:

1. If your report is formatted with HTML style sheet extensions (HTMLCSS), set up your browser to print the HTML document in the same way the report is paginated:
   - In your browser, choose **File > Page Setup** and modify the margin settings as desired.
   - In the Object Navigator, under the **Paper Layout** node, double-click the Header Section, Main Section, or Trailer Section properties icon for the pertinent report section (Header, Main, or Trailer) to display the Property Inspector (by default, a report is defined in the Main Section).
   - In the Property Inspector, under the **Section** node:

     set the Width and Height properties as desired (e.g., 11 (or 15) width x 8.5 height for landscape or 8.5 width x 11 (or 15) height for portrait). Make sure that you subtract the margin depths (e.g., for an 8.5 x 11 page with top, bottom, left, and right margins of .25, set Width to 8 and Height to 10.5).

     set the Orientation property to the desired value.
2. Display your report output in your Web browser (see Section 3.7.15.4, "Displaying report output in your Web browser").
3. Choose **File > Print**.

See also

Section 1.2.2, "About Web reports"

Section 2.2.2, "About previewing JSP-based Web reports"

Section 2.2.10, "About style sheets"

3.7.16.4 Printing a report on a pre-printed form

See the example report in Chapter 31, "Building a Report Using a Pre-Printed Form".

3.7.16.5 Switching the printer tray

To switch the printer tray:

1. Choose **File > Page Setup** and in the **Source** drop-down list, note the names of the printer trays defined for your printer.
2. Use SRW.SET_PRINTER_TRAY to define the desired trigger:
   - To change the printer tray before the report begins formatting, create a Before Report trigger (see Section 3.13.3.5, "Creating a report trigger").

     Note: The printer tray specified in the Page Setup dialog box is used as the default printer tray; create a Before Report trigger to change this default.

   - To change the printer tray between pages of the report, create a Between Pages trigger (see Section 3.13.3.5, "Creating a report trigger").
   - To change the printer tray for a specific page on which a particular object prints, create a format trigger for that object (see Section 3.13.4.1, "Creating or editing a format trigger").

See also

Section 2.8.6, "About switching the printer tray"

3.7.16.6 Printing a report on UNIX

Printing on UNIX requires some setup and configuration to create the proper printing environment. For detailed information about printing your reports on UNIX, refer to the appendix titled "Printing on UNIX with Oracle Reports" in the *Oracle Application Server Reports Services Publishing Reports to the Web* manual, available on the Oracle Technology Network Oracle Reports Documentation page (`http://otn.oracle.com/docs/products/reports/content.html`).

Oracle Reports 10*g* (9.0.4) provides the following enhancements for printing reports on UNIX:

- The REPORTS_DEFAULT_DISPLAY environment variable, which provides for the removal of dependency on the DISPLAY environment variable. Prior releases of Oracle Reports required the DISPLAY environment variable to be set to run and print reports on UNIX.
- The PostScript printer driver screenprinter.ppd, which provides for the removal of dependency on having a valid printer available for Reports Runtime on UNIX. This driver provides surface resolution for images and specifies font information. Prior releases of Oracle Reports required a printer to resolve fonts to run and print reports on UNIX.

Limitations

- If your report includes an Oracle6*i* Graphics graph, a printer is required to generate the graph. However, TK_PRINT_STATUS=echo is set in g90runm.sh. Thus, to run a report with an Oracle6*i* Graphics graph without a valid printer, set the PRINTER environment variable to any value.

> **Note:** Oracle6*i* Graphics, or Graphics Builder, is obsolete in Oracle Reports 10*g*; instead, use the Graph Wizard, which produces graphs that are automatically translated into JSP tags to enable you to add graphics to Web reports (for additional information, see the *Reports Builder online help* topic "Displaying Oracle6*i* Graphics charts in Oracle Reports 10*g*").

3.7.17 E-mailing a report

To e-mail a report from Reports Builder:

1. Log onto your Internet Standard Protocol SMTP mail application.
2. Select or open the report.
3. If you are not already connected, connect to a database.
4. Choose **File > Mail**.
5. In the Mail dialog box, specify the recipients and subject line for your e-mail report.
6. Click **OK**.

To e-mail a report in batch mode:

Note: When using BATCH=yes, Reports Builder will use a blank subject line and cc list. To specify the subject line and cc lists, you must run with BATCH=no (interactively).

1. Log onto your Internet Standard Protocol SMTP mail application.
2. Run the report from the command line, specifying BATCH=YES, DESTYPE=MAIL, and DESNAME=recipient1, recipient2, recipient3, etc.

Usage notes

The configuration file rwbuilder.conf must include the pluginParam mailServer with the outgoing mail server name. This applies in both the Windows NT and Solaris environments.

3.8 Work with the Data Model

This section provides procedures for the following tasks that you may perform as you work with the data model of your report:

- Creating a query
- Modifying a query
- Using Query Builder
- Creating a break group
- Creating a matrix (cross-product) group
- Creating a data link
- Creating or editing a formula column
- Creating a summary column
- Creating or editing a placeholder column

3.8.1 Creating a query

To create a query, you can use any of the following tools:

- Report Wizard (single-query reports only)
- Data Wizard
- SQL Query tool (to create a query that selects data from an Oracle relational database)
- XML Query tool (to access an XML data file, if you have the DTD file)
- JDBC Query tool (to access any JDBC-enabled data source)
- Text Query tool (to create a query that selects data from a text pluggable data source)
- Express Server Query tool (to create a query that selects data from an Express Server)
- OLAP Query tool (to create a query that selects from multidimensional Oracle OLAP (on-line analytical processing) data stored in an Oracle database)
- Ref Cursor Query tool (to use ref cursors)

See also

Section 1.7.1, "About queries"

Section 1.7.3, "About database columns"

3.8.1.1 Creating a query: Report Wizard

To create a query using the Report Wizard:

- On the Data page of the Report Wizard, enter the query statements for the type of data source previously selected. For example, if you selected the SQL Query data source, you can enter your SQL query in any of the following ways:
  - Type the SELECT statement in the **Data Source definition** field.
  - Click **Query Builder** for a graphical method of creating a query without a knowledge of SQL.
  - Click **Import Query** to use a query that has been written by someone else or to use the text editor of your choice.

3.8.1.2 Creating a query: Data Wizard

To create a query using the Data Wizard:

1. In the Data Model view, choose **Insert > Query** to display the Data Wizard.
2. Follow the wizard to create the first query for the data model.
3. Repeat Steps 1 and 2 for each query you want to create.
4. Modify the resulting data model in the Data Model view.
5. To re-enter the Data Wizard, do either of the following:
   - Right-click, and choose **Data Wizard**.
   - Click the query, then choose **Edit > Settings**.

Note: The Data Wizard does not support creating links between queries. To define parent/child query relationships, you can create a data link manually.

3.8.1.3 Creating a query: SQL Query tool

To create a SQL query:

1. In the Data Model view, single-click the SQL Query tool in the tool palette.
2. Click in the main area (canvas region) of the window.
3. In the SQL Query Statement dialog box, define a SELECT statement for the query:
   - To use Query Builder for an easy graphical method of creating a query without a knowledge of SQL, click **Query Builder**.
   - To import a query from a file, click **Import SQL Query**.
   - To enter the SELECT statement yourself, type it in the SQL Query Statement area.

   Tip: Select the columns in the order you want them to appear in the report output.

4. Click **OK**.
5. Refine the query using either of the following methods:
   - Right-click the query object, then choose **Property Inspector** to set desired properties
   - Right-click the query object, then choose **Data Wizard** to specify which fields to display, group fields, and any totals.
6. Repeat Steps 1 through 5 for each query you want to create.

   Note: If you define multiple queries in the Data Model view of your report, the **Data** page does not appear when you invoke the Report Wizard to default the layout.

3.8.1.4 Creating a query: XML Query tool

To create an XML query:

1. In the Data Model view, single-click the XML Query tool in the tool palette.
2. Click in the main area (canvas region) of the window.
3. In the Define XML Query dialog box, define the XML query based on the fields defined in the data definition file (DTD) selected. The query is run against the data in the XML data file. If an XSL file is specified, it translates the XML data file before running the query.
4. Click **OK**.

3.8.1.5 Creating a query: JDBC Query tool

To create a JDBC query:

1. In the Data Model view, single-click the JDBC Query tool in the tool palette.
2. Click in the main area (canvas region) of the window.
3. In the JDBC Query dialog box, define the JDBC query or procedure and the connection parameters for the data source. Click **Help** for assistance.
4. Click **OK**.

3.8.1.6 Creating a query: Text Query tool

To create a Text query:

1. In the Data Model view, single-click the Text Query tool in the tool palette.
2. Click in the main area (canvas region) of the window.
3. In the Define Text Query dialog box, specify the data definition and data source for the text query. Click **Help** for assistance.
4. Click **OK**.

3.8.1.7 Creating a query: Express Server Query tool

To create an Express Server query:

1. In the Data Model view, single-click the Express Server Query tool in the tool palette.
2. Click in the main area (canvas region) of the window.
3. In the dialog box, specify the definition for the Express Server query. Click **Help** for assistance.
4. Click **OK**.

3.8.1.8 Creating a query: OLAP Query tool

To create an OLAP query:

1. In the Data Model view, single-click in the tool palette.
2. Click in the main area (canvas region) of the window.
3. In the dialog box, specify the definition for the OLAP query. Click **Help** for assistance.
4. Click **OK**.

3.8.1.9 Creating a query: Ref Cursor Query tool

To create a ref cursor query:

1. Create a package that defines a ref cursor type in one of the following ways:
   - Create a local program unit (see Section 3.13.3.1, "Creating a local program unit")
   - Create an external PL/SQL library (see Section 3.13.5.1, "Creating an external PL/SQL library")
   - Create a stored program unit (see Section 3.13.3.2, "Creating a stored program unit")
2. If the package created in Step 1 is an external PL/SQL library, you must attach it to the report before referencing it.
3. In the Data Model view, single-click the Ref Cursor Query tool in the tool palette.
4. Click in the main area (canvas region) of the window.

5. In the PL/SQL Editor, type the PL/SQL for a function that opens a cursor and returns a cursor variable of the ref cursor type you defined in the package. For example:

```
empCur rcPackage.empCurType;

BEGIN
  OPEN empCur FOR SELECT * FROM emp;
  RETURN empCur;
END;

/* Note, rcPackage is a local program unit defined as: */
PACKAGE rcPackage IS
  TYPE empCurType IS REF CURSOR RETURN emp%ROWTYPE;
END;
```

6. Refine the query as desired:
   - Click the ref cursor query object, then choose **Tools > Property Inspector** to modify properties.
   - Click the query object, then choose **Tools > Data Wizard** to specify which fields to display, group fields, and any totals.
7. Repeat Steps 1 through 6 for each ref cursor query you want to create.

See also

Section 2.6.10, "About ref cursor queries"

3.8.2 Modifying a query

To modify a SQL query statement or the columns displayed in your report:

1. In the Data Model view, click the query object, then right-click and choose **Data Wizard**.
2. To change the SELECT statement, click the **Data** tab.
3. To change the break groups, click the **Groups** tab.
4. To add or change summary columns for totals, click the **Totals** tab.
5. To change which selected database columns are displayed in your report, choose **Tools > Report Wizard**, then click the **Fields** tab.

To modify the properties of a query:

1. In the Data Model view, click the query object, then right-click and choose **Property Inspector**.
2. In the Property Inspector, modify the properties as desired.

3.8.3 Using Query Builder

You can use Query Builder to define almost any query that you would build using a SQL SELECT statement. Query Builder automatically generates the appropriate SELECT FROM [table.column] clause based on columns displayed in the Query Builder workspace.

See the **How To** section of the *Reports Builder online help* for topics about using Query Builder.

See also

Section 1.7.5, "About Query Builder"

3.8.4 Creating a break group

You can define break groups in the Report Wizard or create them manually.

To create a break group manually:

1. In the Data Model view, drag the group that contains the column at which you want to break your report down from the query object about 2 inches.
2. Select the column that you want to use to divide your report (e.g. if you want to group a list of employee data by department number, select the department number column) and drag it out of and above the group to create a new group. This is the *break column*.
3. Double-click the title bar of the new group object to display its Property Inspector, where you can set properties for the break group.

See also

Section 1.7.2, "About groups"

Section 1.3.2, "About group above reports"

Section 1.3.3, "About group left reports"

3.8.5 Creating a matrix (cross-product) group

You can define matrix groups in the Data Wizard or create them manually.To create a matrix (cross-product) group using the Data Wizard:

1. In the Data Model view, right-click the query object, then choose **Data Wizard**.
2. On the **Query** page, select the **Matrix query** check box.
3. Follow the wizard to select the columns for the matrix group(s), rows, columns, and cells.

   To create a matrix (cross-product) group manually:

4. In the Data Model view, drag the title bar of the group object down.
5. Drag the “row” column and “column” column out of the group object into the space to create two new groups.
6. Single-click the Cross Product tool in the tool palette.
7. Drag a box around the groups you want to include in the matrix (cross-product) group.
8. If the cell group is a separate query from the matrix (cross-product) groups, create the appropriate links between the columns in the matrix groups and the cell group.

See also

Section 2.3.7, "About matrix objects"

Section 1.3.7, "About matrix reports"

Section 2.1.7, "About nested matrix reports"

Section 2.1.8, "About matrix with group reports"

Section 1.7.2, "About groups"

Section 3.9.1.3, "Creating a matrix object"

Section 3.5.3, "Creating a nested matrix report"

Section 1.3.2, "About group above reports"

Section 1.3.3, "About group left reports"

3.8.6 Creating a data link

To create a data link:

1. In the Data Model view, single-click the Data Link tool in the tool palette. A link is always drawn from the parent group to the child query.
2. Create the link:

   Create a Group to Group Link: To create a link between one query's group and another query's group, which is useful when you want the child query to know about the parent's data, click the parent group (avoiding the columns in the group) and drag a link to the child group.

   Create a Column to Column Link: To create a link between columns, click a column of the parent query and drag a link to a column of the child query.
3. Double-click the new link object to display the Property Inspector, and set the desired properties.

See also

Section 1.7.4, "About data links"

Section 2.3.5, "About non-linkable queries"

3.8.7 Creating or editing a formula column

To create or edit a formula column:

1. In the Data Model view, single-click the Formula Column in the tool palette, then:
   - To create a column within a group, click in the group at the position you want the column placed in the hierarchy.
   - To create a report-level column, click in an open area of the canvas region.
2. Double-click the formula column object to display the Property Inspector.
3. Under the **Placeholder/Formula** node, double click the PL/SQL Formula property field.
4. In the PL/SQL Editor, define the PL/SQL for the formula for example SAL * 0.07.

Example: Referencing a PL/SQL function in formulas

Suppose that you have a report with the following groups and columns:

```
Groups  Columns         Summary
---------------------------------------
RGN     REGION
        RGNSUMSAL       SUM(DEPTSUMSAL)
        COSTOFLIVING

DEPT    DNAME
        DEPTNO
        DEPTSUMSAL      SUM(EMP.SAL)

JOB     JOB
        HEADCOUNT       COUNT(EMP.EMPNO)

EMP     ENAME
        EMPNO
        SAL
        COMM
```

Given these groups and columns, you might create multiple formulas that apply the cost of living factor (COSTOFLIVING) to salaries. To avoid duplication of effort, you could create the following PL/SQL function and reference it from the formulas:

```
function CompSal(salary number) return number is
begin
  return (salary*CostofLiving);
end;
```

Following are some examples of how you might reference the PL/SQL function in formulas:

```
CompSal(:RGNSUMSAL)
```

or

```
CompSal(:SAL) + COMM
```

See also

Section 2.3.2, "About formula columns"

3.8.8 Creating a summary column

To create a summary column (for totals or subtotals) *using the Data Wizard*:

1. In the Data Model view, click the query that contains the column you want to total.
2. Choose **Tools > Data Wizard**.
3. On the **Totals** page, follow the wizard to add the desired summary to your report.

> **Note:** For group reports, the Report Wizard and Data Wizard create *n* summary fields in the data model for each summary column you define: one at each group level above the column being summarized, and one at the report level. For example, if a report is grouped by division, and further grouped by department, then a summary column defined for a salary total would create fields for the sum of salaries for each division and each department group (group-level summaries), and the sum of all salaries (report-level summary).

To create a summary column (for totals or subtotals) using the tool palette:

4. In the Data Model view, single-click the Summary Column tool in the tool palette, then:
   - To create a column within a group, click in the group at the position you want the column placed in the hierarchy.
   - To create a report-level column, click in an open area of the canvas region.
5. Double-click the new summary column object to display the Property Inspector.
6. Under the **Summary** node:
   - set the Function property by choosing the type of summary you want from the drop-down list.
   - set the Source property to the column you want to summarize.

See also

Section 2.3.1, "About summary columns"

3.8.9 Creating or editing a placeholder column

To create or edit a placeholder column:

1. In the Data Model view, single-click the Placeholder Column tool in the tool palette.
   - To create a column within a group, click in the group at the position you want the column placed in the hierarchy.
   - To create a report-level column, click in an open area of the canvas region.
2. Double-click the placeholder column object.
3. In the Property Inspector, set the desired properties for the placeholder column.
4. Set the value of a placeholder column in:
   - the Before Report trigger, if the placeholder is a report-level column
   - a report-level formula column, if the placeholder is a report-level column
   - a formula in the placeholder's group or a group below it (the value is set once for each record of the group)

See also

Section 2.3.3, "About placeholder columns"

3.9 Work with the Report Layout

This section provides procedures for creating and modifying objects in your report layout. The procedures are grouped into the following sections:

- General Layout Objects
- Text Objects
- Page Numbers or Date/Time Stamps
- Borders
- Anchors
- Colors, Patterns, and Highlighting
- Format Masks
- Graphic or Image Objects
- Page or Group Headers or Footers
- Margin, Header Page, or Trailer Page Objects
- Move Objects
- Resize Objects
- Change Spacing
- Modify the Page Layout

3.9.1 General Layout Objects

This section provides procedures for the following tasks that you may perform as you work with container objects:

- Creating a field object
- Creating a frame or repeating frame
- Creating a matrix object
- Creating a barcode using a barcode font
- Applying conditional formatting to a layout object

3.9.1.1 Creating a field object

To create a field object manually in the report layout:

1. In the Paper Design view, Paper Layout view or Paper Parameter Form view, click the Field tool in the tool palette.
2. Click and drag a rectangle.
3. Double-click the new field object.
4. In the Property Inspector, under the **Field** node, set the Source property to the column or parameter that will provide the value for the field.
5. Set other properties as desired.

See also

Section 1.8.4, "About fields"

Section 1.9.2, "About Parameter Form fields"

3.9.1.2 Creating a frame or repeating frame

To create a frame or repeating frame manually:

1. In the Paper Layout view, click the Frame tool or the Repeating Frame tool in the tool palette.
2. Click and drag a rectangle.

> **Note:** The size of the rectangle must be large enough to include the objects that it will contain.

3. Double-click the new frame object.
4. In the Property Inspector, set the desired properties.

See also

Section 1.8.1, "About frames"

Section 1.8.2, "About repeating frames"

Section 1.8.3, "About frame and repeating frame sizing"

3.9.1.3 Creating a matrix object

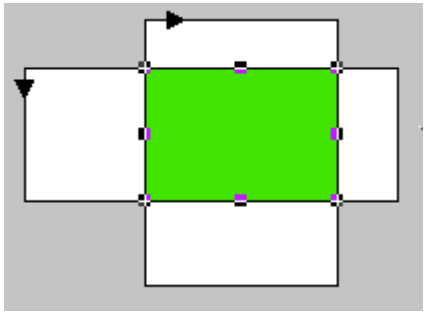
> **Note:** This procedure provides steps to create a matrix object manually. The recommended method for creating a matrix object is to use the Report Block Wizard (choose **Insert > Report Block** in the Paper Layout view).

To create a matrix object:

1. In the Paper Layout view, click the Repeating Frame tool in the tool palette, then drag a rectangle to create a repeating frame.
2. In the Property Inspector, under **Repeating Frame**, set the Source property to the "column" for the matrix, and set the Print Direction property to *Down*.
3. Click the Repeating Frame tool in the tool palette again, then drag a rectangle to create a second repeating frame, intersecting the first repeating frame.
4. In the Property Inspector, under **Repeating Frame**, set the Source property to the "row" for the matrix, and set the Print Direction property to *Across*.
5. Drag a rectangle around both repeating frames to select them, then choose **Insert > Layout Matrix**.

   This creates a matrix object that is the intersection of the two repeating frames. For example:

Figure 3–1 Example matrix object



6. Double-click the matrix object to set its properties.

See also

Section 2.3.7, "About matrix objects"

Section 1.3.7, "About matrix reports"

Section 2.1.7, "About nested matrix reports"

Section 2.1.8, "About matrix with group reports"

Section 3.8.5, "Creating a matrix (cross-product) group"

Section 3.5.3, "Creating a nested matrix report"

3.9.1.4 Creating a barcode using a barcode font

Note: Beginning with Oracle9*i* Reports Release 1 (9.0.2), you can use a barcode JavaBean that automatically generates the barcode for you. See the example report in Chapter 40, "Building a Report with a Barcode".

To create a barcode using a barcode font:

1. Install a barcode font on your machine (e.g., on Windows, install the barcode font in the Windows Control Panel **Font**). Barcode fonts are available from most computer stores.

   Note: For portability between platforms, you can use a PostScript barcode font to allow you to print your report to any PostScript-enabled output device.

   ---

2. Create a query that selects a column from the database containing the numbers for the barcodes you want included in your report.
3. In the Paper Layout view, click the barcode field, and choose **Format > Font** to apply the barcode font to the field.

3.9.1.5 Applying conditional formatting to a layout object

To apply conditional formatting to a layout object:

1. Click the object, then choose **Format > Conditional Formatting**.

> **Tip:** To select multiple objects, click one object, then Shift-click all other objects. To select all objects, choose **Edit > Select All**.

2. In the Conditional Formatting dialog box, click **New** to display the Format Exception dialog box.
3. Specify the conditions and formatting attributes to be applied when the condition(s) evaluate to TRUE.
4. Click **OK**.

Usage notes

- For more complex conditional formatting, you can define conditions against an object using a PL/SQL format trigger in the PL/SQL Editor.

See also

Section 2.1.6, "About conditional formatting"

Chapter 3.9.6.7, "Highlighting a value"

Chapter 3.9.6.8, "Highlighting a row"

3.9.2 Text Objects

This section provides procedures for the following tasks that you may perform as you work with text objects:

- Creating a boilerplate object for text
- Creating a boilerplate object for text that displays every other page
- Editing text
- Referencing a field in boilerplate text
- Linking a boilerplate text object to a file
- Wrapping text in a field
- Changing text attributes

3.9.2.1 Creating a boilerplate object for text

To create a boilerplate object for text:

1. In the Paper Design view, Paper Layout view or Paper Parameter Form view, click the Text tool in the tool palette.
2. Click and drag a rectangle.
3. In the new boilerplate object, type the desired text.

   The text you type is composed of paragraphs which are separated by new lines. Text within a paragraph is word-wrapped inside the boilerplate object's horizontal size.
4. Click outside the boilerplate text object.

See also

Section 1.8.5, "About boilerplate objects"

3.9.2.2 Creating a boilerplate object for text that displays every other page

To create a boilerplate object for text that displays every other page:

1. Create a boilerplate text object where you want it to appear on the page, either in the margin, or in a repeating frame (expand the repeating frame to make room for the boilerplate text, below the fields in the repeating frame).
2. Double-click the boilerplate text object to display the Property Inspector. Under the Advanced Layout node, set the Format Trigger property by typing the following code in the PL/SQL Editor:

```
function XXX_HDRFormatTrigger return boolean is page_num number;
begin
  srw.get_page_num(page_num);
  if mod(page_num, 2) = 0 then
    return(false);
  else
    return (true);
  end if;
end;
```

3. Run the report to see the boilerplate text appear every other page.

See also

Section 1.8.5, "About boilerplate objects"

3.9.2.3 Editing text

To insert and replace text:

1. In the Paper Design view, Paper Layout view, or Paper Parameter Form view, single-click the text object.
2. Click the text to change to edit mode.
3. Modify the text as desired, then click outside the text object.

To delete text:

1. In the Paper Design view, Paper Layout view, or Paper Parameter Form view, single-click the text object.
2. To cut all the text, choose **Edit > Clear**.
3. To cut portions of text, click and drag to mark the text you want to cut, then press the Delete key.

To cut, copy, or paste text:

1. In the Paper Layout view or Parameter Form view, single-click the text object.
2. Click and drag to mark the text you want to cut or copy.
3. Choose **Edit > Cut** or **Edit > Copy**.
4. To paste text, choose **Edit > Paste**.

3.9.2.4 Referencing a field in boilerplate text

To reference a field in boilerplate text:

1. In the Paper Design view or Paper Layout view, click in the boilerplate text where you want the reference to a field to appear.
2. Type an ampersand (*&*) followed by the name of the field.

   If you want to place the field reference in front of other text with no spaces in between, enclose the field name in angle brackets to separate it from the text (e.g., `&<`*`fieldname`*`>Oracle`). If the field reference follows other text, no angle brackets are needed (e.g., `Oracle&`*`fieldname`*). You can include field references right next to each other with no spaces in between and without angle brackets (e.g., *`&field1&field2&field3`*).

Usage notes

- A reference to a field includes the field's properties. Therefore, if the Horizontal Elasticity property is set to *Fixed*, any extra spaces in the field will appear in the report output. For example, if field `f_sal` is fixed horizontally, `$&<f_sal>/week` may produce `$800/week` in the report output.
- In addition to referencing fields in a boilerplate text object, you can directly reference a database column (e.g., `&SAL`). For example, `$&<SAL>/week` may produce `$800/week` in the report output.

See also

Section 1.8.4, "About fields"

Section 1.8.5, "About boilerplate objects"

3.9.2.5 Linking a boilerplate text object to a file

To link a boilerplate text object to a file:

1. In the Paper Design view, click the Link File tool in the tool palette.
2. Click and drag a rectangle.
3. Double-click the link file object to display the Property Inspector.
4. Under the **Link File Boilerplate** node:
   - Set the Source File Format property to *Text*.
   - Set the Source Filename property to the name of the file containing the text.

See also

Section 1.8.5, "About boilerplate objects"

3.9.2.6 Wrapping text in a field

To wrap text in a field:

1. Choose **Tools > Report Wizard**.
2. In the Report Wizard, click the **Labels** tab.
3. Change the value in the **Width** column for the field in which you want to wrap text, as desired.
4. Click **Apply**.

Caution: If you have made manual adjustments to your layout in the Paper Layout view or Paper Design view, you will lose these layout changes when you click **Apply** or **Finish** in the Report Wizard, which redefaults the layout. If you do not want to overwrite your current layout, you can manually drag the column to the desired width in the Paper Layout view or Paper Design view.

5. In the Paper Layout view or Parameter Form view, double-click the field to display the Property Inspector, and set properties:
   - Under **General Layout**, verify that the Vertical Elasticity property is set to Expand.

3.9.2.7 Changing text attributes

To change the font, justification, spacing, or reading direction of a text object:

1. In the Paper Design view, click the text object(s).

   Tip: To select multiple objects, click one object, then Shift-click all other objects. To select all objects, choose **Edit > Select All**.

2. From the **Format** menu, choose **Font**, **Justify**, **Text Spacing**, or **Direction**.

3.9.2.8 Changing text attributes using PL/SQL

To change the font or justification of a text object using PL/SQL instead of the user interface:

1. In the Object Navigator, expand the **Paper Layout** node.
2. Double-click the PL/SQL icon next to the text object(s).

   Tip: To select multiple objects, click one object, then Shift-click all other objects. To select all objects, choose **Edit > Select All**.

3. In the PL/SQL Editor, use the following built-in procedures to change the font or justification of the object as desired:

   SRW.SET_CHARMODE_TEXT

   SRW.SET_FONT_FACE

   SRW.SET_FONT_STYLE

   SRW.SET_FONT_WEIGHT

   SRW.SET_FONT_SIZE

   SRW.SET_JUSTIFICATION

See also

Topic "SRW built-in package" in the **Reference > PL/SQL Reference > Built-in Packages** section of the *Reports Builder online help*.

3.9.3 Page Numbers or Date/Time Stamps

This section provides procedures for the following tasks that you may perform as you work with page numbers or date/time stamps:

- Creating page numbers
- Resetting page numberss
- Creating a time and/or date stamp

3.9.3.1 Creating page numbers

To create default page numbers:

1. In the Paper Layout view or Paper Design view, choose **Insert > Page Number**.
2. In the Insert Page Number dialog box, choose from the drop-down list the location for the page number.
3. Click the desired page number format: **Page Number Only** or **Page Number and Total Pages**.
4. Optionally, change the default attributes of the page number text as desired.

To create customized page numbers:

1. In the Paper Layout view or Paper Design view, click the Edit Margin button in the toolbar.

   Note: The margin area is defined by a thick black line that separates it from the body. If you create objects in the body portion of a report while displaying the margin area, you can only edit those objects when the margin is displayed.

2. Create a field object (see Section 3.9.1.1, "Creating a field object") in the margin area for each page number value you require, for example Physical Page Number, Total Physical Pages.
3. In the field's Property Inspector, under the **Field** node, set the Source property to the source for the page number, and set the Visible property to *No*. Under the **General Layout** node, set the Horizontal Elasticity property to *Variable*.
4. Create a boilerplate text object (see Section 3.9.2.1, "Creating a boilerplate object for text") in the margin area, and reference the page number field(s) (see Section 3.9.2.4, "Referencing a field in boilerplate text") using `&fieldname`.

Usage notes

- To generate odd page numbers on the right and even pages numbers on the left, you have to create two fields: one on the right side, one on the left side. In the format trigger of each field, you test if it is an odd or an even page by using SRW.GET_PAGE_NUM. Then, you specify whether or not to display the field.

3.9.3.2 Resetting page numbers

To reset default page numbers (created with **Insert > Page Number**):

1. In the Paper Layout view or Paper Design view, click the default page number field, then choose **Edit > Clear** to delete it.
2. Create customized page numbers (see Section 3.9.3.1, "Creating page numbers").
3. In the page number field's Property Inspector, under the **Field** node, double-click the Page Numbering property field (labeled...).

4. In the Page Numbering dialog box, click the desired **Reset At** setting. Page numbers will reset to the **Start at** value each time the selected frame is formatted. Click **OK**.

3.9.3.3 Creating a time and/or date stamp

To add a date and/or time stamp to your report:

1. In the Paper Layout view Paper Design view, choose I**nsert > Date and Time**.
2. In the Insert Date and Time dialog box, choose from the drop-down list the location for the date or time stamp.
3. Click the desired date and/or time format, or click **Custom** to define your own format.

3.9.4 Borders

This section provides procedures for the following tasks that you may perform as you work with borders:

- Showing or hiding object borders
- Changing object border attributes
- Changing the current mode (Confine or Flex)

3.9.4.1 Showing or hiding object borders

To show or hide the borders around an object:

1. Click the object(s).

> **Tip:** To select multiple objects, click one object, then Shift-click all other objects. To select all objects, choose **Edit > Select All**.

2. Click the Line color tool in the tool palette.
3. To add a border, click the solid black square.
4. To make a border transparent in the report output, click **No Line** at the bottom of the color palette.

Note: If you make the borders transparent for all the objects, this includes the underlines beneath the column headings. To add the underlines, click the underline objects, and use the Line Color in the tool palette to select a line color.

5. To show or hide part of the border, choose **Format > Line > Border**, and select from the menu to toggle between showing and hiding the sides of the border.

3.9.4.2 Changing object border attributes

To change the borders around an object:

1. Click the object(s).

Tip: To select multiple objects, click one object, then Shift-click all other objects. To select all objects, choose **Edit > Select All**.

2. To change the line width, choose **Format > Line > Line Width**.
3. To select a pattern, choose **Format > Line > Dash**. Even though it appears in the output, the dash pattern is sometimes obscured by the object's outline. You can choose **Tools > Options > Paper Layout** and clear the Options:Frame Outlines check box.
4. To "frame" the border, choose **Format > Bevel**.

See also

Section 2.4.5, "About changing colors and patterns"

Section 3.12.8, "Modifying the color, pattern, or border of body objects in a template"

3.9.4.3 Changing the current mode (Confine or Flex)

To set or override Confine mode:

- In the Paper Layout view, click the Confine On button or the Confine Off button in the toolbar to toggle the mode:

  On: child objects cannot be moved outside their enclosing parent objects.

  Off: child objects can be moved outside their enclosing parent objects.

To set or override Flex mode:

- In the Paper Layout view, click the Flex On button or the Flex Off button in the toolbar to toggle the mode:

  On: parent borders "stretch" when child objects are moved against them. The child object maintains the same distance from the side it moves against.

  Off: parent borders remain fixed when child objects are moved against them.

See also

Section 2.4.7, "About moving and layering objects in the Paper Layout view"

Chapter 3.9.11.3, "Adjusting parent borders automatically"

Chapter 3.9.11.2, "Moving an object outside its parent"

3.9.5 Anchors

This section provides procedures for the following tasks that you may perform as you work with anchors:

- Anchoring objects together
- Viewing implicit anchors
- Moving an anchor

See also

Section 2.4.4, "About anchors"

3.9.5.1 Anchoring objects together

Anchoring objects assures that the anchored object will move with the parent object. An object can be anchored to only one other object.

To anchor objects together:

1. In the Paper Layout view, click the Anchor tool in the tool palette.
2. Click an edge of the child object and double-click an edge of the parent object.

   A line is drawn from the child to the parent. A small box appears at the end of the line that is attached to the parent object.

3.9.5.2 Viewing implicit anchors

By default, you see the explicit anchors created in the Paper Layout view of the Report Editor.

To view information on both implicit and explicit anchors:

1. In the Object Navigator, choose **Tools > Options > Navigator** to display the Object Navigator Options dialog box.
2. Click the **Layout** tab, and select the **Anchoring Information** check box.

   With this option selected, you can see all information on both implicit and explicit anchors in the Object Navigator.

> **Note:** By default, objects are anchored to the upper left corner of their enclosing object. If this view of the Object Navigator does not show anchoring information for an object, you can assume that the object is anchored to its enclosing object, which might be the frame or the body.

See also

Section 2.4.4.1, "Implicit anchoring algorithm"

3.9.5.3 Moving an anchor

Moving an anchor changes how the objects will be displayed in relationship to each other.

To move an anchor:

1. In the Paper Layout view, click the anchor.
2. Click the Reshape tool in the tool palette, then drag one of the anchor endpoints to its new location on the object edge.

To move an anchor along an object edge:

- Press the constrain (e.g., shift) key when moving the anchor.

To change the position of the anchor on the object edge, as a percentage down or across from top to bottom or left to right:

1. Double-click the anchor object to display the Property Inspector.
2. Set the Child Edge Percent property and/or Parent Edge Percent property to a new value.

3. To change the edge on which the anchor is positioned, set the Child Edge Type property and/or Parent Edge Type property as desired.

3.9.6 Colors, Patterns, and Highlighting

This section provides procedures for the following tasks that you may perform as you work with colors, patterns, and highlighting:

- Setting color palette preferences
- Changing colors
- Changing patterns
- Changing colors and patterns using PL/SQL
- Modifying the color palette
- Importing or exporting a color palette
- Highlighting a value
- Highlighting a row

3.9.6.1 Setting color palette preferences

See Section 3.2.6, "Setting color palette preferences".

3.9.6.2 Changing colors

To change the color of an object or text:

1. In the Paper Layout view, click the object(s).

> **Tip:** To select multiple objects, click one object, then Shift-click all other objects. To select all objects, choose **Edit > Select All**.

2. Click the Fill Color tool, the Line Color tool, or the Text Color tool in the tool palette, depending on which part of the object you want to apply a color to.

Note: The Fill/Line/Text Display, the box directly above the three Color tools, shows the currently selected fill, border, and text. The default fill and border for objects created by Reports Builder is transparent, while the default for objects you create is a black, one-point line around a white.

3. On the color palette, click a color.

See also

Section 2.4.5, "About changing colors and patterns"

Section 3.12.8, "Modifying the color, pattern, or border of body objects in a template"

3.9.6.3 Changing patterns

To change the pattern of an object:

1. In the Paper Layout view, click the object(s).

Tip: To select multiple objects, click one object, then Shift-click all other objects. To select all objects, choose **Edit > Select All**.

2. Click the Fill Color tool or the Line Color tool in the tool palette, depending on which part of the object you want to apply a pattern to.

Note: The Windows platform does not support a border pattern (that is, patterns for the Line Color tool).

3. On the color palette, click **Patterns**.
4. On the pattern palette, click a pattern. To change the foreground and background colors, choose from the drop-down color palettes at the bottom of the pattern palette.

See also

Section 2.4.5, "About changing colors and patterns"

Section 3.12.8, "Modifying the color, pattern, or border of body objects in a template"

Topic "Pattern color palette" in the **Reference > Color and Pattern Palettes** section of the *Reports Builder online help*.

3.9.6.4 Changing colors and patterns using PL/SQL

To change the color of an object using PL/SQL instead of the user interface:

1. In the Object Navigator, expand the **Paper Layout** node.
2. Double-click the PL/SQL icon next to the object for which you want to change the color.
3. In the PL/SQL Editor, use the following built-in procedures to change the color of the object as desired:

   SRW.SET_BACKGROUND_BORDER_COLOR

   SRW.SET_BACKGROUND_FILL_COLOR

   SRW.SET_FOREGROUND_BORDER_COLOR

   SRW.SET_FOREGROUND_FILL_COLOR

   SRW.SET_TEXT_COLOR

   SRW.SET_BORDER_PATTERN

See also

Section 2.4.5, "About changing colors and patterns"

Topic "SRW built-in package" in the **Reference > PL/SQL Reference > Built-in Packages** section of the *Reports Builder online help.*

3.9.6.5 Modifying the color palette

To modify the color palette:

1. First, make the color palette editable:
   - Choose **Edit > Preferences**, and set **Color Mode** to **Editable**, as described in Section 3.2.6, "Setting color palette preferences".

     Note: By default, color palettes are read-only.

   - Shut down and restart Reports Builder to enable the Editable mode.
2. In the Paper Layout view, choose **Format > Color Palette > Edit**.
3. In the Color Palette dialog box, modify the color palette as desired:
   - Click **Edit** to alter the settings of the current color.
   - Type a new name in **Current Color**, then click **Rename** to rename the current color.
   - **Select Color to Edit** contains the current color palette being used in the Report Editor for the current report. Select a color from the palette to alter it.
   - Click **OK** to apply your changes to the current report.

See also

Section 2.4.5, "About changing colors and patterns"

Topics "Oracle CDE1 color palette", "Default color palette", and "Grayscale color palette" in the **Reference > Color and Pattern Palettes** section of the *Reports Builder online help*.

3.9.6.6 Importing or exporting a color palette

To import or export a color palette:

1. First, make the color palette editable:
   - Choose **Edit > Preferences**, and set **Color Mode** to **Editable**, as described in Section 3.2.6, "Setting color palette preferences".

     Note: By default, color palettes are read-only.

   - Shut down and restart Reports Builder to enable the Editable mode.
2. In the Paper Layout view, choose **Format > Color Palette > Import** or **Format > Color Palette > Export**.
3. In the dialog box, specify the name and format of the file.
4. Click **OK**.

See also

Section 2.4.5, "About changing colors and patterns"

Topics "Oracle CDE1 color palette", "Default color palette", and "Grayscale color palette" in the **Reference > Color and Pattern Palettes** section of the *Reports Builder online help*.

3.9.6.7 Highlighting a value

To highlight a value in a report:

1. In the Paper Layout view or Paper Design view, click the field that contains the value to be highlighted.
2. Choose **Format > Conditional Formatting**.
3. In the Conditional Formatting dialog box, click **New** to display the Format Exception dialog box.
4. Select the field and define the condition(s) when the value should be highlighted.
5. In the Format group box, select the **Fill Color** to be used to highlight the value.

Note: If you can't select the condition you want to use in the dialog box, select the formatting you want and a placeholder condition. The condition can be edited in the PL/SQL Editor for the format trigger that is created.

6. Click **OK** to close the Format Exception dialog box, then click **OK** again to close the Conditional Formatting dialog box. If the code compiles without errors, the new formatting is reflected in the Paper Layout view. The code is stored as a format trigger for the field.

Example

The following code determines the monthly compensation and changes the background color to red if the compensation has exceeded four thousand dollars a month and the employee is not a manager.

```
function R_G_EMPNOFormatTrigger return boolean is varcomm number;
begin
 if :comm is null then
   varcomm := 0;
 else
   varcomm := :comm;
 end if;
  if (:sal * 2 + varcomm > 4000) and :job != 'MANAGER' then
   srw.set_background_fill_color('red');
 end if;
 return (TRUE);
end;
```

See also

Section 2.1.6, "About conditional formatting"

Section 3.9.1.5, "Applying conditional formatting to a layout object"

3.9.6.8 Highlighting a row

To highlight an entire row:

1. In the Paper Layout view, click the repeating frame that contains the fields that make up the rows to be highlighted.
2. Choose **Format > Conditional Formatting**.

3. In the Conditional Formatting dialog box, click New to display the Format Exception dialog box.
4. Select the field(s) and define the condition(s) that describe the rows to be highlighted.
5. In the **Format** group box, select the **Fill Color** to be used to highlight the row.

> **Note:** If you can't select the condition you want to use in the dialog box, select the formatting you want and a placeholder condition.The condition can be edited in the PL/SQL Editor for the format trigger that is created.

6. Click **OK** to close the Format Exception dialog box, then click **OK** again to close the Conditional Formatting dialog box. If the code compiles without errors, the new formatting is reflected in the Paper Layout view. The code is stored as a format trigger for the repeating frame.

Example

The following code determines the monthly compensation and changes the background color to red if the compensation has exceeded four thousand dollars a month and the employee is not a manager.

```
function R_G_EMPNOFormatTrigger return boolean is varcomm number;
begin
if :comm is null then
  varcomm := 0;
else
  varcomm := :comm;
end if;

  if (:sal * 2 + varcomm > 4000) and :job != 'MANAGER' then
  srw.set_background_fill_color('red');
end if;
  return (TRUE);
end;
```

See also

Section 2.1.6, "About conditional formatting"

Section 3.9.1.5, "Applying conditional formatting to a layout object"

3.9.6.9 Alternating row colors

To alternate the colors of either the text or the fill of a row:

- See Chapter 41, "Building a Report with an XML Pluggable Data Source". This chapter provides detailed steps for using PL/SQL procedures and format triggers to apply alternating row colors.

3.9.7 Format Masks

This section provides procedures for the following tasks that you may perform as you work with format masks:

- Specifying date and time format masks
- Specifying number format masks
- Applying a format mask to a numeric object
- Applying a format mask to a date object
- Adding a custom format mask
- Changing the format mask for multiple fields

3.9.7.1 Specifying date and time format masks

The following tables describe the date and time format masks, and the suffixes you can add to date format masks:

Table 3–2 Data and time format masks

Format Mask	Description
SCC or CC	Century, abbreviated;'S' prefixes "BC" with (-)
SYYYY or YYYY	Year;'S' prefixes "BC" "BC" date with a (-)
I or IY or IYY	Last 1, 2, or 3 digit(s) of year
Y or YY or YYY	Last 1, 2, or 3 digit(s) of year
Y,YYY	Year with comma
SYEAR or YEAR	Year, spelled out; 'S' prefixes "BC" date with (-)
RRRR or SRRRR	Year; 'S' prefixes "BC" date with a (-)
RR	Last 2 digit(s) of year
BC, AD, or B.C., A.D.	Century indicator

Table 3–2 Data and time format masks

Format Mask	Description
Q	Quarter of year (Jan-Mar= Quarter 1)
MM	Month in digits (Jan = 01)
MONTH or MON	Name of month, or 3-letter abbreviation
WW, IW	Week in year
W	Week in Julian days
J	Julian day; the number of days since January 1,4712 BC
DDD, DD, or D	Day in year, month, or week
DAY	Day of week fully spelled out (e.g., MONDAY)
DY	Name of day, 3-letter abbreviation (e.g., MON)
AM, PM, or A.M., P.M.	Meridian indicator
HH or HH12	Hour of day (1-12)
HH24	Hour of day (0-23)
MI	Minute
SS; SSSSS	Second in minute; seconds in day
FM	Toggles fill mode which replaces multiple spaces before or between dates, numbers, or words with a single space

The following suffixes may be added to the format masks:

Table 3–3 Date format mask suffixes

Suffix	Description
TH	Suffixed number ("DDth" for "4th")
SP	Spelled out number ("DDSP" for "FOUR")
SPTH or THSP	Spelled and suffixed number ("DDSPTH" for "FOURTH")

Examples

Table 3–4 Date format mask examples

Sample Date Format	Date Displayed
MM/DD/RR	03/04/02
DD MON RRRR	04 MAR 2002
Mon. DD, RRRR	Mar. 4, 2002
Day Month DD fmHH:MI AM	Monday March 4 11:35 AM
Dy Mon ddth fmHH24:MI:SS	Mon Mar 4th 23:35:22
Day "the" ddthsp "of" Month	Monday the fourth of March

3.9.7.2 Specifying number format masks

The following table describes the tokens you can use in creating a number format mask:

Table 3–5 Tokens for number format masks

Format Token	Description
0	Prints one digit.
N	Prints one digit, unless it is a leading zero to the left of the decimal point or a trailing zero to the right of the decimal point.
*	Prints one digit, unless it is a leading zero to the left of the decimal point, in which case an asterisk (*) is printed. Trailing zeros to the right of the decimal point are printed.
9	Prints one digit, unless it is a leading zero to the left of the decimal point, in which case a space is printed. Trailing zeros to the right of the decimal point are printed.
+	Prints a leading plus (+) for positive values and a leading minus (-) for negative values. This token must lead the mask.
-	Prints a leading minus (-) for negative values and nothing for positive values. This token must lead the other tokens.
MI	Prints a minus (-) after negative values and a space after positive values. This token must trail the other tokens.

Table 3–5 Tokens for number format masks

Format Token	Description
S	Prints a minus (-) for negative values and a plus (+) for positive values (wherever the S appears in the mask). This token must lead or trail the other tokens.
PR	Prints angle brackets (<>) around negative values and spaces around positive values. This token must trail the other tokens.
( )	Prints parentheses around negative values and spaces around positive values. The parentheses must surround the other tokens.
DB	Prints a "DB" after positive values. This token must trail the other tokens.
CR	Prints a "CR" after negative values. This token must trail the other tokens.
Y	Causes no sign symbol to be printed for negative or positive values.
V	Causes the number to be multiplied by 10N, where N is the number of 0, 9, *, and S tokens that appear to the right of the V.
EEEE	Causes the number to be printed in scientific notation. All digit tokens refer to digits of the mantissa. There must be exactly one digit to the left of the decimal point (displayed or implicit). The token EEEE prints as E followed by a plus (+), if the ordinate is positive or zero, and a minus (-), if the ordinate is negative, and two digits representing the ordinate (e.g., E-99).
"string"	Prints the string between the double quotes. To have double-quotes inside the string, type double-quotes back to back ("").
. (period)	Prints a period (.) to separate the integral and fractional parts of a number.
D	Prints the local decimal character to separate the integral and fractional parts of a number.
, (comma)	Prints a comma (,) as the group/thousands separator.
G	Prints the local group/thousands separator.
$	Prints $
L	Prints the local currency symbol.

Table 3–5 Tokens for number format masks

Format Token	Description
C	Prints the ISO currency symbol.
%	Prints %.
" "	Prints a blank space. (Do not include quotes in mask.)
v	Prints a blank space for all zero values, regard-less of other tokens.
K	Prints a blank space.
<>	Delineates the beginning and ending of the decimal-aligned region (i.e., that part of the number that you want aligned with the decimal point in the format mask). Angle brackets indicate that the number should always occupy the same amount of space. If necessary, values are padded with blanks to the left or right of the decimal point.
RN, rn	Prints values in uppercase or lowercase Roman numerals, respectively. You cannot enter any other tokens with this token.

Restrictions

- For number format masks, if the actual value is longer than the specified format mask, the value will appear as a string of asterisks in the report output, regardless of the field's width. For example, if a fixed field's width is 8, the value is 1234567, and the format mask is <NNNNNN>, your output will be *******.
- Similarly, if the number format mask causes the value to be larger than the field width, asterisks will appear in place of the value. For example, if a fixed field's width is 6, the value is 1234, and the format mask is -99999999, your output will be ******. This occurs because the format token 9 prints a blank for leading zeros to the left of the decimal. As a result, the value is too long to be displayed in a field of width 6.
- If you do not specify a sign token in the format mask, positive values are preceded by a space and negative values are preceded by a minus (-).
- After you create a format mask it will display in the list of values only if it is an appropriate format mask for the Datatype of the Source --i.e., format masks for numbers are displayed when the Source is a number, and format masks for dates are displayed when the Source is a date.

- Format masks that print spaces for zeros (e.g., 9) increase the number of bytes per page taken up by your output.

Examples

Table 3–6 Number format mask examples

Sample Number Format	Number	Number Displayed
-0000	7934	"7934"
	-7934	"-7934"
-00000	7934	"07934"
-NNNN	7639	"7639"
	535	"535"
-NNN	7639	"***"
-****	7902	"7902"
-*****	7902	"*7902"
+NNNN	100	"+100"
	-99	"-99"
(NNNN)	1600	" 1600 "
	-800	"(800)"
NNNNPR	1600	" 1600 "
	-800	"<800>"
NNNNMI	1600	"1600 "
	-800	"800-"
NNNVNN	343	"34300"
N.NNEEEE	7369	"7.37E+03"
"SRW"-0000	7782	"SRW7782"
-$NNNN.NN	800	"$800"
	1600	"$1600"
-%NNN	10	"%10"
-NN NNN.NN	3000	"3 000"

Table 3–6 Number format mask examples

Sample Number Format	Number	Number Displayed
+KKNNNNN.00	1950	"+ 1950.00"
	900	"+ 900.00"
$<NNNNN.00>	1200	"$1200.00"
	500	"$500.00"
$<NNNNN.00> DB	1200	"$1200.00 DB"
	-500	"$500.00"
$<NNNNN.00> CR	1200	"$1200.00"
	-500	"$500.00 CR"

* The quotation marks will not appear in your output. They are used here to make it clear where there are leading or trailing spaces.

3.9.7.3 Applying a format mask to a numeric object

1. To apply a format mask to a numeric object:
2. In the Paper Design view, click the object(s).

> **Tip:** To select multiple objects, click one object, then Shift-click all other objects. To select all objects, choose **Edit > Select All**.

3. To apply the currency format to the object, click the Currency button in the toolbar.
4. To apply the percentage format to the object, click the Percentage button in the toolbar.
5. To add commas to the value of the object, click the Commas button in the toolbar. To move the comma right or left, click the Remove Decimal Place button or the Add Decimal Place button.
6. Choose **Tools > Property Inspector**.
7. Under the **Field** node, verify the Format Mask property is set to the desired format.
8. Set other properties as desired.

3.9.7.4 Applying a format mask to a date object

To apply a format mask to a date object:

1. In the Paper Design view, click the object(s).

   Tip: To select multiple objects, click one object, then Shift-click all other objects. To select all objects, choose **Edit > Select All**.

2. Choose **Tools > Property Inspector**.
3. Under the **Field** node, set the Format Mask property to the desired date format.
4. Set other properties as desired.

3.9.7.5 Adding a custom format mask

To add a custom format mask to the default format masks list:

1. Choose **Edit > Preferences**.
2. In the Preferences dialog box, click **Edit Masks**.
3. In the Format Masks dialog box, set **Display** to the type of format mask you want to add.
4. In the Mask field, type the format mask to add, then click **Add**.
5. Click **OK**.

   Now, you will see your new custom format masks in the list of values for the Format Mask property for fields and the Input Mask property for parameters.

3.9.7.6 Changing the format mask for multiple fields

To change the format mask for multiple fields at one time:

1. In the Paper Layout view, click the fields you want to change.
2. Choose **Tools > Property Inspector**.
3. In the Property Inspector, set the Format Mask property to a new format mask.

3.9.8 Graphic or Image Objects

This section provides procedures for the following tasks that you may perform as you work with graphic or image objects:

- Creating a drawing object
- Adding a graph to a paper report
- Editing a graph in a paper report
- Adding a graph to a Web report
- Editing a graph in a Web report
- Importing a drawing or image
- Selecting an image from the database
- Selecting an image URL from the database
- Linking an image object to a file
- Linking an image object to a URL

3.9.8.1 Creating a drawing object

To create a drawing object:

1. In the Paper Layout view, click a drawing tool in the tool palette (e.g., Rectangle, Ellipse, Polyline, etc.).
2. To create a line, rounded rectangle, rectangle, ellipse, arc, or freehand object, click in the main area (canvas region) of the window and drag to create the object.
3. To create a polygon or polyline, click in the main area (canvas region) of the window where you want each point of the object, then double-click to create the object.
4. To draw constrained objects (i.e., perfect circles and squares, etc.), hold down the constrain key (e.g., the shift key) when drawing the graphic.
5. Double-click the drawing object.
6. In the Property Inspector, set the desired properties.

3.9.8.2 Adding a graph to a paper report

To add a graph to your paper report:

1. In the Paper Layout view, click the Graph tool in the tool palette.
2. Drag a square in the area where the graph should appear to display the Graph Wizard.
3. Follow the wizard to create the desired graph and position it in your report.
4. Double-click the graph object you have created to display the Property Inspector, and set properties as desired.
5. To re-enter the Graph Wizard, do either of the following:
   - Right-click, and choose **Graph Wizard**.
   - Click the graph, then choose **Edit > Settings**.

3.9.8.3 Editing a graph in a paper report

To edit a graph in a paper report:

1. In the Paper Layout view, click the graph.

   Note: The graph is represented as a bar graph even if the graph is of another type.

2. To edit the XML definition of the graph:
   - Right-click and choose **Property Inspector**.
   - In the Property Inspector, under the **Graph** node, click the Graph Settings property value field to display the Graph Settings dialog box where you can edit the XML.
3. To re-enter the Graph Wizard to redefine the graph, do either of the following:
   - Right-click and choose **Graph Wizard**.
   - Choose **Edit > Settings**.

3.9.8.4 Adding a graph to a Web report

To add a graph to a JSP-based Web report:

1. In the Object Navigator, double-click the view icon next to the **Web Source** node to display the source code in the Web Source view.
2. Locate the section in the source code where you want to add the graph.

> **Note:** Adding some text such as "Place the graph here" to your Web page allows you to easily locate the correct position for your graph.

3. Choose **Insert > Graph**.
4. In the Graph Wizard, specify the information for the graph. Click **Help** on any tab page for assistance.
5. Click the Run Web Layout button in the toolbar to display your report and graph in your Web browser.

> **Note:** If Netscape 7.0 is your default browser, the browser may not display. You can work around this bug by making a copy of the Netscape 7.0 executable, naming it `netscape.exe`; with this name, the browser will display as expected.

6. To re-enter the Graph Wizard, place your cursor anywhere between the `<rw:graph>` and `</rw:graph>` tags in the Web Source view, then choose **Edit > Settings**.

3.9.8.5 Editing a graph in a Web report

To edit a graph in a JSP-based Web report, do any of the following in the Web Source view:

- Between the `<rw:graph>` and `</rw:graph>` tags that define the graph, make changes directly to the JSP and XML tags.

> **Note:** This is not recommended as errors can result in messages or an unreadable graph.

- Place your cursor anywhere between the <rw:graph> and </rw:graph> tags, then choose **Edit > Settings** to re-enter the Graph Wizard.

  Note: Any manual changes and additions you've made to the graph XML will be lost if you click **Finish**, which redefaults the graph to the definition in the Graph Wizard.

- Delete and re-create the graph as follows:

  1. Delete all lines from the <rw:graph> tag to the </rw:graph> tag.
  2. Leaving the cursor in the position where the graph was cut, choose **Insert > Graph** and follow the Graph Wizard to re-create the graph.

3.9.8.6 Importing a drawing or image

1. To import a drawing or image:
2. In the Paper Layout view, choose **Insert > Image** and click the type of object you want to import.
3. In the dialog box, specify the name and format of the file. Click **OK**.
4. Move the object to the desired position.

See also

Section 2.4.3, "About images"

3.9.8.7 Selecting an image from the database

To select an image from the database:

1. Create a query that selects a column containing images or the filenames of image objects.
2. In the Data Model view, double-click the image column to display the Property Inspector.
3. If the column contains the filenames of image objects, rather than the images themselves:
   - Under the **Column** node, set the Read from File property to Yes.
4. Under the **Column** node, set the File Format property to Image.

See also

Section 2.4.3, "About images"

Section 1.7.3, "About database columns"

3.9.8.8 Selecting an image URL from the database

To select a URL from the database for including an image in an HTML report:

1. Create a query, with a SELECT statement that selects a column containing URLs.
2. In the Data Model view, double-click the column containing the URLs to display the Property Inspector.
3. Under **Column**:
   - set the Read from File property to Yes.
   - set the File Format property to Image URL.

See also

Section 2.4.3, "About images"

Section 2.8.8, "About HTML output"

3.9.8.9 Linking an image object to a file

To link an image object to a file:

1. In the Paper Design view or Paper Layout view, click the Link File tool in the tool palette.
2. Click and drag a rectangle.
3. Double-click the link file object to display the Property Inspector.
4. Under **Link File Boilerplate**:
   - set the Source File Format property to Image.
   - set the Source Filename property to the name of the file containing the image.

See also

Section 2.4.3, "About images"

Section 1.8.5, "About boilerplate objects"

3.9.8.10 Linking an image object to a URL

Note: This procedure is for HTML output only.

To link an image object to a URL:

1. In the Paper Design view or Paper Layout view, click the Link File tool in the tool palette
2. Click and drag a rectangle.
3. Double-click the link file object to display the Property Inspector.
4. Under the **Link File Boilerplate** node:
   - set the Source File Format property to Image URL.
   - set the Source Filename property to the URL where the image is located with the required protocol.

Example 1

```
HTTP://www.oracle.com/images/logo.gif
```

Example 2

```
HTTP://&<P_SERVER_NAME>/images/logo.gif
```

where `P_SERVER_NAME` is a user parameter of type CHAR

At runtime, the end user can specify a value for the parameter (e.g., `P_SERVER_NAME = www.oracle.us.com` for a dynamic URL link of `HTTP://www.oracle.us.com/images/logo.gif`).

Example 3

```
FILE://c:/images/logo.gif
```

Note: If you click **Browse** to find a file, Reports Builder automatically prepends `FILE://` to the returned path.

See also

Section 2.4.3, "About images"

Section 2.8.8, "About HTML output"

3.9.9 Page or Group Headers or Footers

This section provides procedures for the following tasks that you may perform as you work with page or group headers or footers:

- Creating a text heading
- Creating a heading that includes database values
- Creating a group header or footer

3.9.9.1 Creating a text heading

To create an object or heading in the margin of a report:

1. In the Paper Layout view, click the Edit Margin button in the toolbar.

> **Note:** The margin area is defined by a thick black line that separates it from the body. If you create objects in the body portion of a report while displaying the margin area, you can only edit those objects when the margin is displayed.

2. To adjust the margin, click the margin border, then drag a handle to the desired position. You can adjust the margin on all four sides of a report.
3. Create required objects in the margin area. They will appear on all pages of the report.
4. Click the Header Section, Main Section, or the Trailer Section buttons in the toolbar to reactivate the appropriate section of the body area of the report.

3.9.9.2 Creating a heading that includes database values

To create a heading that includes database values:

1. Choose **Tools > Report Wizard** to re-enter the Report Wizard for the current report.
2. On the **Totals** page, create any totals that you want to include in the heading.

> **Note:** Any totals that you intend to place in the margin area of your report must be report-level totals. Group totals (e.g., the total for a department) placed in the margin will cause a frequency error when you run your report.

3. On the **Fields** page, verify that the fields and totals you want to include in the heading are either available or displayed in the report.
4. In the Paper Layout view, click the Margin button in the toolbar.

> **Note:** The margin area is defined by a thick black line that separates it from the body. If you create objects in the body portion of a report while displaying the margin area, you can only edit those objects when the margin is displayed.

5. For any values that require that you specify a format mask, create a hidden field object in the margin area (see Section 3.9.1.1, "Creating a field object"):
   - In the Property Inspector, under the **Field** node, set the Source property to the source column for the value, set the Visible property to *No*, and set the Format Mask property as desired.
   - Under the **General Layout** node, set the Horizontal Elasticity property to *Variable*.

> **Note:** If you do not need to specify a format mask, you can simply reference the value directly in Step 6, and you do not need to create this hidden field.

6. Create a boilerplate text object in the margin area, and reference the field and/or report-level totals you want to include in the heading. See Section 3.9.2.1, "Creating a boilerplate object for text" and Section 3.9.2.4, "Referencing a field in boilerplate text".

See also

The example report in Chapter 17, "Building a Header with Database Values Report".

3.9.9.3 Creating a group header or footer

To create a header or footer above or below each group of records:

1. In the Paper Layout view, click the Confine Off button and the Flex On button in the toolbar.
2. Click frame that encloses the repeating frame for the group to which you want to add a header or footer, then drag and resize the frame to allow enough room to type the header or footer text.

> **Tip:** In the Object Navigator, under the **Paper Layout** node, expand the **Body** node to select the frame by name.

3. Create a boilerplate text object for the header or footer text (see Section 3.9.2.1, "Creating a boilerplate object for text").

3.9.10 Margin, Header Page, or Trailer Page Objects

This section provides procedures for the following tasks that you may perform as you work with margin, header page, or trailer page objects:

- Creating a margin object
- Creating a header page or trailer page object

3.9.10.1 Creating a margin object

To create an object or heading in the margin of a report:

1. In the Paper Layout view, click the Edit Margin button in the toolbar.

> **Note:** The margin area is defined by a thick black line that separates it from the body. If you create objects in the body portion of a report while displaying the margin area, you can only edit those objects when the margin is displayed.

2. To adjust the margin, click the margin border, then drag a handle to the desired position. You can adjust the margin on all four sides of a report.
3. Create required objects in the margin area. They will appear on all pages of the report.

4. Click the Header Section, Main Section, or the Trailer Section buttons in the toolbar to reactivate the appropriate section of the body area of the report.

3.9.10.2 Creating a header page or trailer page object

With report sectioning, Header and Trailer pages are identical to Body pages. In effect, this means that the Header, Trailer, and Body are three sections of a report. The names of the sections are exposed under the **Paper Design** node in the Object Navigator as **Header Section**, **Main Section**, and **Trailer Section**. You can use the margin and body of the Header and Trailer sections to create a Header and Trailer "page" as in earlier releases.

1. In the Paper Layout view, click the Confine Off button and the Flex On button in the toolbar.
2. Click the repeating frame for the page, then drag and resize the frame to allow enough room to type the header or footer text.

> **Tip:** In the Object Navigator, under the **Paper Layout** node, expand the **Body** node to select the repeating frame by name.

3. Create a boilerplate text object for the header or footer text.

3.9.11 Move Objects

This section provides procedures for the following tasks that you may perform as you move your report objects:

- Moving multiple objects
- Moving an object outside its parent
- Adjusting parent borders automatically
- Moving a column in report output
- Offsetting detail objects in a group report
- Aligning objects
- Changing object layering
- Rotating a boilerplate object

3.9.11.1 Moving multiple objects

To move multiple objects:

1. Click or drag a box around the objects you want to move.

> **Tip:** To select multiple objects, click one object, then Shift-click all other objects. To select all objects, choose **Edit > Select All**.

2. Use the arrow keys on your keyboard to move the objects in the desired direction.

See also

Section 2.4.7, "About moving and layering objects in the Paper Layout view"

3.9.11.2 Moving an object outside its parent

To move a child object outside its enclosing parent object:

1. In the Paper Layout view or Paper Design view, click the Confine Off button in the toolbar.
2. Click and drag the child object(s) as desired.

> **Tip:** To select multiple objects, click one object, then Shift-click all other objects. To select all objects, choose **Edit > Select All**.

See also

Section 2.4.7, "About moving and layering objects in the Paper Layout view"

3.9.11.3 Adjusting parent borders automatically

To adjust parent borders as you move child objects:

1. In the Paper Layout view, click the Flex On button in the toolbar.
2. Click and drag the child object(s) as desired.

> **Tip:** To select multiple objects, click one object, then Shift-click all other objects. To select all objects, choose **Edit > Select All**.

3. To move an object that aligns with another object in the horizontal or vertical direction (for example, a field and its label), hold down the Ctrl key as you drag the first object. To move both objects simultaneously, do not use the Ctrl key.

See also

Section 2.4.6, "About resizing objects"

3.9.11.4 Moving a column in report output

To move a column in report output:

1. In the Paper Layout view or Paper Design view, click the column.
2. Click the Flex On button in the toolbar
3. Drag the column to the desired position.

See also

Section 2.4.7, "About moving and layering objects in the Paper Layout view"

Chapter 3.9.13.1, "Changing columns labels or widths"

Chapter 3.9.13.3, "Changing the default layout spacing"

3.9.11.5 Offsetting detail objects in a group report

To offset the detail fields in a group above or group left report:

1. In the Paper Design view, click the Flex On button in the toolbar.
2. Select the detail objects and move them to the right.
3. Click the Flex Off button in the toolbar.

See also

Section 1.3.2, "About group above reports"

Section 1.3.3, "About group left reports"

3.9.11.6 Aligning objects

To align objects:

1. Click the objects you want to align.

> **Tip:** To select multiple objects, click one object, then Shift-click all other objects. To select all objects, choose **Edit > Select All**.

2. Choose **Layout > Alignment**.
3. In the Align Objects dialog box, specify the desired alignment.
4. Click **OK**.

See also

Section 2.4.7, "About moving and layering objects in the Paper Layout view"

3.9.11.7 Changing object layering

To change the order in which objects are layered on top of each other:

1. In the Paper Layout view, click the Confine Off button in the toolbar.
2. Click the object you want to move.
3. Choose one of the following items from the **Arrange** menu:
   - **Bring to Front** to move the object in front of all other objects.
   - **Send to Back** to move the object behind all other objects.
   - **Move Forward** to move the object in front of the object directly on top of it.
   - **Move Backward** to move the object behind the object directly underneath it.

See also

Section 2.4.7, "About moving and layering objects in the Paper Layout view"

3.9.11.8 Rotating a boilerplate object

You can only rotate boilerplate text and graphics. You cannot rotate other layout objects (repeating frames or fields).

To rotate a boilerplate object:

1. Click the object(s) that you want to rotate.

   Tip: To select multiple objects, click one object, then Shift-click all other objects. To select all objects, choose **Edit > Select All**.

2. Click the Rotate tool in the tool palette.
3. Drag a handle to rotate the object or group.

See also

Section 1.8.5, "About boilerplate objects"

3.9.12 Resize Objects

This section provides procedures for the following tasks that you may perform as you resize your report objects:

- Resizing objects
- Making multiple objects the same size
- Resizing object borders

3.9.12.1 Resizing objects

To resize one or more objects:

1. Click the object(s) that you want to resize.

   Tip: To select multiple objects, click one object, then Shift-click all other objects. To select all objects, choose **Edit > Select All**.

2. Set the size:
   - For a fixed size, drag the handle of one of the selected objects. All selected objects will change size accordingly.

- Or, choose **Tools > Property Inspector**. In the Property Inspector, under the **General Layout** node, set the Vertical Elasticity and Horizontal Elasticity properties to *Contract*, *Expand*, *Fixed*, or *Variable*.

See also

Section 2.4.6, "About resizing objects"

Section 1.8.3, "About frame and repeating frame sizing"

3.9.12.2 Making multiple objects the same size

To make multiple objects the same size:

1. Click the objects you want to size.

   Tip: To select multiple objects, click one object, then Shift-click all other objects. To select all objects, choose **Edit > Select All**.

2. Choose **Layout > Size Objects**.
3. In the Size Objects dialog box, specify the desired settings. Click **OK**.

3.9.12.3 Resizing object borders

See Section 3.9.11.3, "Adjusting parent borders automatically".

3.9.13 Change Spacing

This section provides procedures for the following tasks that you may perform as you change spacing between your report objects:

- Changing columns labels or widths
- Changing spacing within a text object
- Changing the default layout spacing
- Changing the spacing between all rows
- Adding blank lines between groups of rows

3.9.13.1 Changing columns labels or widths

To change column labels or widths:

1. Choose **Tools > Report Wizard**.
2. In the Report Wizard, click the **Labels** tab.
3. Change the values in the **Labels** and **Width** columns, as desired.
4. Click **Apply**.

See also

Chapter 3.9.11.4, "Moving a column in report output"

3.9.13.2 Changing spacing within a text object

To change spacing within a text object:

1. In the Paper Design view, click the object(s).

> **Tip:** To select multiple objects, click one object, then Shift-click all other objects. To select all objects, choose **Edit > Select All**.

2. Choose **Format > Text Spacing** and select the desired spacing for the text object.

3.9.13.3 Changing the default layout spacing

To change the default layout spacing used by the Report Wizard when defaulting the report layout:

1. Choose **Tools > Preferences**.
2. In the Preferences dialog box, click the **Wizards** tab.
3. To increase the space between objects and the objects they enclose, type larger values for **Horizontal Gap** and **Vertical Gap**.
4. To increase the space between fields, type larger values in **Horizontal Interfield** and **Vertical Interfield**.
5. Click **OK** to close the Preferences dialog box.
6. For smaller spacings, choose **View > Snap to Grid** to toggle this setting off (no checkmark). When **Snap to Grid** is set on, defaulting the layout honors the layout spacing values, but then snaps to the closest gridpoint; the result may be that smaller changes to these values are not evident in the report output.

See also

Chapter 3.9.11.4, "Moving a column in report output"

3.9.13.4 Changing the spacing between all rows

To change the spacing between all rows using the Paper Design view:

1. In the Paper Design view, click the second record.
2. Drag the second record down to create the desired spacing between all rows in the report output.

To change the spacing between all rows using the Property Inspector:

1. In the Paper Design view, click any field in the body of the report.
2. Click the Select Parent Frame button in the toolbar.
3. Choose **Tools > Property Inspector**.
4. Under the **Repeating Frame** node, set the Vert. Space Between Frames (inches) property to the amount of blank space to leave between row in the report output (e.g., 0.25).

3.9.13.5 Adding blank lines between groups of rows

To add blank lines between groups of rows:

1. Create a summary column (see Section 3.8.8, "Creating a summary column"). On the **Totals** page of the Data Wizard (or Report Wizard), select any field that appears in each row of output, and Count as the calculation. On the **Fields** page, remove the summary from the displayed fields column.
2. Create a user parameter named SPACE (see Section 3.11.2, "Creating a user parameter"). In the Property Inspector, under the **Parameter** node, set the Datatype property to Number and the Initial Value property to the number of records you want in each group (e.g., 5).
3. In the Paper Layout view, click the Flex On and Confine Off buttons in the toolbar.
4. Click the repeating frame for the group, then drag the handle on the bottom of the frame to create a space slightly larger than the space you want to add between groups of rows in the report output.
5. Choose **Tools > Property Inspector** to display the Property Inspector for the repeating frame.
6. Under the **General Layout** node, set the Vertical Elasticity property to *Variable*.
7. Click the Rectangle tool in the tool palette.
8. Draw a rectangle in the space under the record.
9. Double-click the rectangle object.
10. In the Property Inspector, under the **Advanced Layout** node, double-click the Format Trigger property field (labeled**...**) to display the PL/SQL Editor.
11. In the PL/SQL Editor, define the PL/SQL for the format trigger. For example, the following PL/SQL code inserts blank space between groups of rows (displays a boilerplate rectangle when the row count divided by the value of SPACE leaves no remainder):

```
function spacing return BOOLEAN is
begin
  if :CountENAMEPerReport MOD :SPACE = 0 then
    return (true);
  else
    return (false);
  end if;
end;
```

12. Hide the rectangle object:
    - on the Fill Color palette, click **No Fill**
    - on the Line Color palette, click **No Line**

3.9.14 Modify the Page Layout

This section provides procedures for the following tasks that you may perform as you modify the page layout of your report:

- Adjusting margins
- Adding a page break

3.9.14.1 Adjusting margins

To adjust margins:

1. In the Paper Layout view, click the Edit Margin button in the toolbar.

   Note: The margin area is defined by a thick black line that separates it from the body. If you create objects in the body portion of a report while displaying the margin area, you can only edit those objects when the margin is displayed.

2. Click the margin border, then drag a handle to the desired position. You can adjust the margin on all four sides of a report.

3.9.14.2 Adding a page break

To add a page break:

1. In the Paper Layout view, click any field that is part of the frame at which you want to insert a page break.
2. Click the Parent Frame in the toolbar.
3. Choose **Tools > Property Inspector**.
4. To set a page break...
   - before or after the selected frame: under the **General Layout** node, set the Page Break Before property to *Yes* for the object you want placed on a separate page.

- after a certain number of rows of output on each page: under the **Repeating Frame** node, set the Maximum Records per Page property to the number of rows you want to display on each page.

5. To display icons that identify where page breaks are set, choose **View > Page Breaks** in the Paper Layout view.

Usage notes

If you want to set a page break at a specific row of report output, you can create a hidden object (for example, a rectangle with No Fill and No Line) that formats only at the specific row. If you set the Page Break Before property to *Yes* for the rectangle object, you will get a page break after the row.

3.10 Work with Report Sections

This section provides procedures for the following tasks that you may perform as you work with report sections:

- Displaying a section layout view
- Creating a default layout for a section

See also

Section 2.1.2, "About report sectioning and sections"

3.10.1 Displaying a section layout view

To display the layout view for a report section:

- In the Paper Layout view, do either of the following:
  - Choose **View > Layout Section**, then select the section you want to view and check whether or not you want to view or edit the margin area.
  - In the toolbar, click the Header Section, the Main Section, or Trailer Section buttons to view the Header Section, Main Section, or Trailer Section, respectively; click the Edit Margin button to view or edit the margin area.

3.10.2 Creating a default layout for a section

To create a default layout for a report section:

1. In the Paper Layout view, display the layout view for the section.
2. Choose **Tools > Report Wizard**.
3. Follow the wizard to create a default layout for your report.
4. To create a layout or objects in the margin area of a section, click the Edit Margin button in the toolbar to view/edit the margin area.
5. To add another layout section to the current layout, create an additional report layout (see Section 3.5.5, "Creating an additional report layout").

6. Make further modifications to the default layout manually in the Paper Layout view.

   Caution: If you re-enter the Report Wizard after making manual adjustments to your layout in the Paper Layout or Paper Design view, you will lose these layout changes when you click **Finish** in the Report Wizard, which redefaults the layout.

7. Set properties for the section, as desired.

See also

Section 2.4.2, "About layout defaulting"

Section 3.5.4, "Creating a default layout for a report"

Chapter 3.9.13.3, "Changing the default layout spacing"

3.11 Work with Parameters and the Parameter Form

This section provides procedures for the following tasks that you may perform as you work with parameters and the Parameter Form:

- Using a pre-defined system parameter
- Creating a user parameter
- Creating a list of values (LOV) for a parameter
- Validating a parameter value at runtime
- Creating a default Parameter Form
- Selecting parameters to include in the Runtime Parameter Form
- Displaying the Parameter Form at runtime
- Adding more pages to the Runtime Parameter Form
- Passing parameters to reports running in batch mode
- Creating an HTML Parameter Form header using PL/SQL
- Creating an HTML Parameter Form footer using PL/SQL
- Creating HTML Parameter Form input or select events
- Changing HTML Parameter Form input to uppercase

See also

Section 1.9.1, "About parameters"

Section 2.3.4, "About referencing columns and parameters"

Section 1.6.5, "About the Paper Parameter Form view"

Section 1.11.1, "About the Runtime Parameter Form"

Section 1.9.4, "About Parameter Forms for Web reports"

3.11.1 Using a pre-defined system parameter

To use a system parameter:

1. In the Object Navigator, expand the Data Model node, then expand the System Parameters node.
2. Double-click the properties icon for the desired parameter to display the Property Inspector.
3. Under the Parameter node, set the Initial Value property, if required.
4. To validate the parameter's value at runtime, set the Validation Trigger property by clicking the**...** button to display the PL/SQL Editor and define the PL/SQL to be triggered at runtime.

3.11.2 Creating a user parameter

> **Note:** Reports Builder automatically creates a user parameter when you use a bind parameter reference in a query.

To create a user parameter:

1. In the Object Navigator, expand the **Data Model** node, then click the **User Parameters** node.
2. Click the Create button in the toolbar.
3. Double-click the properties icon for the new parameter to display the Property Inspector.
4. Under the **General Information** node, replace the Name property with the desired parameter name.
5. Under the **Parameter** node, set the Initial Value and List of Values properties, if required.
6. To validate the parameter's value at runtime, set the Validation Trigger property by clicking property field to display the PL/SQL Editor and define the PL/SQL to be triggered at runtime.

3.11.3 Creating a list of values (LOV) for a parameter

> **Tip:** If you define a parameter in a template, you must apply the template to a report in order to select that parameter for the Runtime Parameter Form.

To create a LOV for a parameter:

1. In the Object Navigator, expand the **Data Model** node, then the **User Parameters** node.
2. Double-click the properties icon for the parameter for which you want to create a LOV to display the Property Inspector.
3. Under the **Parameter** node, double-click the List of Values property field to display the Parameter List of Values dialog box.
4. Select the type of list that you want to create:
   - For **Static Values**, type a value in the **Value** text box and click **Add**. Repeat for each value you want to add. (Click **Remove** to delete items from the list)
   - For **SELECT Statement**, type a query to populate the list of values. You can select more than one column to display in the LOV, where the first column contains the value to be assigned to the parameter. The LOV displays columns in the order specified in the query.
5. If you want the parameter value to be restricted to only those in the LOV, select the **Restrict List to Predetermined Values** check box. To display a combo box that allows users to edit values or type a different value in the Runtime Parameter Form, clear the check box.
6. If you do not want the first column (which contains the parameter value) of your query displayed in the LOV, select the **Hide First Column** check box. If there is no need to preserve the confidentiality of the first column, clear the check box.

Caution: If you send the report output to an HTML file, either from Reports Builder or running it in your Web browser, the value of the first column will be visible in the HTML source, even if **Hide First Column** is selected.

If you run the report from a Web browser and the list of values is unrestricted, the HTML Parameter Form will display a text field instead of a combo box, and a list of static values that you can copy and paste into the text field. In this case, the first column will always be shown in the Parameter Form, even if **Hide First Column** is selected.

7. Click **OK**.

3.11.4 Validating a parameter value at runtime

To validate a parameter value at runtime:

1. In the Object Navigator, expand the **Data Model** node, then the **System Parameters** or **User Parameters** node.
2. Double-click the PL/SQL icon for the parameter for which you want to add a PL/SQL validation trigger.
3. In the PL/SQL Editor, define the PL/SQL to be triggered at runtime.

See also

Section 2.6.12.3, "About validation triggers"

3.11.5 Creating a default Parameter Form

To create a default Parameter Form:

1. Choose **Tools > Parameter Form Builder**.
2. Click **OK** to display the Paper Parameter Form view with the default Parameter Form.

3.11.6 Selecting parameters to include in the Runtime Parameter Form

To select parameters to include in the Runtime Parameter Form:

1. Choose **Tools > Parameter Form Builder**.
2. In the Parameter Form Builder, click the parameters you want to include in the Runtime Parameter Form.
3. Modify the parameter labels as desired.
4. Click **OK** to display the Paper Parameter Form view.

3.11.7 Displaying the Parameter Form at runtime

To display the Runtime Parameter Form when you run your report:

1. Choose **Edit > Preferences** to display the Preferences dialog box.
2. On the Runtime Settings page, make sure that the Parameter Form check box is selected.

Note: The Parameter Form can be used only for paper reports. If you display your paper-based report on the Web, you can create an HTML Parameter Form by adding HTML header and footer tags (using either the Before Form Value property and After Form Value property or the SRW.SET_BEFORE_FORM_HTML and SRW.SET_AFTER_FORM_HTML procedures). However, if you design a JSP-based Web report with a Parameter Form in Reports Builder, be aware that Web reports that use JSPs do not support the display of the Runtime Parameter Form at runtime. For more information, see Section 1.9.4, "About Parameter Forms for Web reports".

3.11.8 Adding more pages to the Runtime Parameter Form

To add more pages to the Runtime Parameter Form:

1. In the Object Navigator, double-click the properties icon next to the report name.
2. In the Property Inspector, under the **Parameter Form Window** node, set the Number of Pages property as desired.

3.11.9 Passing parameters to reports running in batch mode

To pass parameters (e.g., data ranges) to reports running in batch mode:

- Use bind variables in your query to restrict the query, and use the command line parameters to pass the values to the query. For example:

Query:

```
SELECT * FROM EMP WHERE HIREDATE BETWEEN
:FROM_DATE AND :END_DATE
```

Runtime:

```
RWRUN REP1 SCOTT/TIGER FROM_DATE='12-JUN-92'
END_DATE='24-JUN-92'
```

3.11.10 Creating an HTML Parameter Form header using PL/SQL

See Section 3.6.7.2.5, "Creating an HTML Parameter Form header using PL/SQL"

3.11.11 Creating an HTML Parameter Form footer using PL/SQL

See Section 3.6.7.2.6, "Creating an HTML Parameter Form footer using PL/SQL"

3.11.12 Creating HTML Parameter Form input or select events

To create an HTML Parameter Form field with input or select events:

1. In the Object Navigator, double click the view icon next to the **Paper Parameter Form** node to display the Paper Parameter Form view.
2. Create or edit a Parameter Form field (see Section 3.9.1.1, "Creating a field object").
3. Double-click the field object to display the Property Inspector.
4. Under **Web Settings**, set the Additional Attributes (HTML) property to a valid JavaScript event handler.

Note: In some cases, for example, when raising messages it may be necessary to type JavaScript code in the Before Form trigger.

To insert the JavaScript code in the Before Form trigger:

1. In the Object Navigator, double-click the properties icon next to the report name to display the Property Inspector.
2. Under **Report Escapes**, set the Before Form Type property to *Text* (if you will type the Javascript) or *File* (if you will import the JavaScript from a file).
3. Set the Before Form Value property by clicking the **...** button to either type JavaScript in the dialog box or select an HTML file with the JavaScript to import.

Example 1: Data input validation

This example shows how to set Parameter Form fields for input validation when the report is run via the Web. Doing so will raise a message whenever an end user enters invalid data in the Parameter Form field.

1. In the Paper Parameter Form view, create a Parameter Form field called PF_DEPTNO.
2. Double-click the field object to display the Property Inspector, and set properties:
   - Under **Parameter Form Field**, set the Source property to DEPTNO.
   - Under **Web Settings**, set the Additional Attributes (HTML) property to the following JavaScript event handler:

     ```
     onChange="checkIt(this.form)"
     ```

3. In the Object Navigator, click (the properties icon) next to your report name to display the Property Inspector, and set properties:
   - Under **Report Escapes**, set the Before Form Type property to Text.
   - Set the Before Form Value property the following JavaScript code:

```
<SCRIPT LANGUAGE = "JavaScript">
function isNumber(inputStr){
   for (var i = 0; i < inputStr.length; i++) {
      var oneChar = inputStr.charAt(i)
      if (oneChar < "0" || oneChar > "9") {
         alert("Please enter a numeric value.")
         return false
         }
     }
    return true
```

```
   }
function checkIt(form) {
   inputStr = form.DEPTNO.value
   if (isNumber(inputStr)) {
      // statements if true
      }
      else {
         form.numeric.focus()
         form.numeric.select()
      }
   }
</SCRIPT>
```

At runtime, if the end user enters the department name in the Runtime Parameter Form rather than the department number when running the report via the Web, the following message is raised:

```
Please enter a numeric value.
```

Example 2: Select validation

This example shows you how to set Parameter Form fields for select validation when the report is run via the Web. Doing so will raise a message whenever an end user selects Printer from the DESTYPE list of values in the Runtime Parameter Form.

1. In the Paper Parameter Form view, create a Parameter Form field called PF_DESTYPE.
2. Double-click the field object to display the Property Inspector, and set properties:
   - Under **Parameter Form Field**, set the Source property to DESTYPE.
   - Under **Web Settings**, set the Additional Attributes (HTML) property to the following JavaScript event handler:

     ```
     onChange="isPrinter(this.form)"
     ```

3. In the Object Navigator, click (the properties icon) next to your report name to display the Property Inspector, and set properties:
   - Under **Report Escapes**, set the Before Form Type property to Text.

- Set the Before Form Value property the following JavaScript code:

```
<SCRIPT LANGUAGE = "JavaScript">
function isPrinter(form) {
  if( form.DESTYPE.options[form.DESTYPE.selectedIndex].value
       == 'Printer')
    alert("Please be sure that your print is installed and running.")
    return true}
   }
</SCRIPT>
```

At runtime, if the end user selects PRINTER from a list of values in the DESTYPE field, the following message is raised:

```
Please be sure that your print is installed and running.
```

See also

Section 2.5.1, "About Parameter Form HTML extensions"

3.11.13 Changing HTML Parameter Form input to uppercase

To change data input values default to uppercase upon entry in a Parameter Form field:

1. Choose **Tools > Parameter Form Builder**.
2. In the Parameter Form Builder, create a Parameter Form field called PF_DESFORMAT, with a source of DESFORMAT.
3. Choose **Tools > Property Inspector**.
4. In the Property Inspector, under Web Settings, set the Additional Attributes (HTML) property to:

   ```
   onChange="this.value=this.value.toUpperCase()"
   ```

Example: Default input to uppercase

This example specifies that data input values default to uppercase upon entry in a Parameter Form field.

1. In the Paper Parameter Form view, create a Parameter Form field called PF_DESTYPE.
2. Double-click the field object to display the Property Inspector, and set properties:

- Under **Parameter Form Field**, set the Source property to *DESFORMAT*.
- Under **Web Settings**, set the Additional Attributes (HTML) property to the following JavaScript event handler:

```
onChange="this.value=this.value.toUpperCase()"
```

At runtime, if the end user enters `pdf` as the destination format, the value will change to uppercase (i.e., PDF) in the Runtime Parameter Form when running the report via the Web.

3.12 Define a Template

This section provides procedures for the following tasks that you may perform as you define a report template:

- Creating a template
- Defining default template attributes
- Defining override template attributes
- Applying a template to a report
- Formatting the report title in a template
- Modifying the color, pattern, or border of body objects in a template
- Adding a template to the pre-defined templates list

3.12.1 Creating a template

To create a template:

1. In the Object Navigator, click the **Templates** node, then click the Create button in the toolbar.
2. Define default template attributes for all report styles.
3. Optionally, define override template attributes for a selected report style.
4. To define system parameters, user parameters, report triggers, program units, and attached libraries for the template, double-click the associated nodes in the Object Navigator.

See also

Section 2.7.5, "About the Template Editor"

Section 2.7.1, "About templates"

Section 2.7.2, "About template attributes"

Section 2.7.3, "About applying templates"

Section 2.7.4, "About inheritance in templates"

3.12.2 Defining default template attributes

To define default attributes for report objects in a template that applies to all report styles:

1. In the Object Navigator, expand the **Templates** node, then the **Paper Layout** node, then the **Section** node, then the **Body** node, then the **Default** node.
2. To change default properties for the overall layout and spacing for report styles, double-click the properties icon next to the **Default** node to display the Property Inspector.
3. To change default properties for frames, field labels/headings, fields, summary labels, and summaries (totals), fully expand the **Default** node, then double-click the properties icons to display the Property Inspector.

> **Note:** The Paper Layout view is synchronized with the Object Navigator; when you select a layout object, the associated node is highlighted in the Object Navigator. To select the parent frame for a layout object, click the object, then click the Select Parent Frame button in the toolbar to select the parent frame.

4. In the Paper Layout view, select from the **Report Style** drop-down list to view the layout for individual report styles.
5. To make further changes for the currently displayed report style, either return to the Object Navigator, or double-click an object to display the Property Inspector.

> **Note:** Any changes you make for an individual report style will override the attributes defined under the **Default** node.

See also

Section 2.7.1, "About templates"

Section 2.7.2, "About template attributes"

3.12.3 Defining override template attributes

To define override attributes for a selected report style:

1. In the Object Navigator, expand the **Templates** node, then the **Paper Layout** node, then the Section node, then the **Body** node, then the **Override** node, then the node for the report style for which you want to define overrides (e.g., **Tabular, Group Left**).
2. To change default properties for the overall layout and spacing for a section of the selected report style, double-click the properties icon next to the **Section (Level n)** node to display the Property Inspector.
3. To create more than the default number of sections for a particular report style, click a Section (Level n) node, then click the Create button in the toolbar.

   Note: Sections are mapped to groups in a report. For more information, see About template attributes.

4. To delete a section for a particular report, click a Section (Level n) node, then click the Delete button in the toolbar.

   Note: You can only delete nodes down to the minimum requirement for the current report style.

5. For each section for which you want to define overrides, fully expand the **Section (Level n)** node, then double-click the properties icons to display the Property Inspectors for the frames, field labels/headings, fields, summary labels, and summaries (totals).

   Note: The Paper Layout view is synchronized with the Object Navigator; when you select a layout object, the associated node is highlighted in the Object Navigator. To select the parent frame for a layout object, click the object, then click the Select Parent Frame button in the toolbar to select the parent frame.

6. In the Paper Layout view, select from the **Report Style** drop-down list to view the layout for individual report styles.

7. To make further changes for the currently displayed report style, either return to the Object Navigator, or double-click an object to display the Property Inspector.

> **Note:** Any changes you make for an individual report style will override the attributes defined under the **Default** node.

See also

Section 2.7.1, "About templates"

Section 2.7.2, "About template attributes"

3.12.4 Applying a template to a report

To apply a template to a report:

1. In the Object Navigator, click the report or report section to which you want to apply a template.

> **Note:** If you select the node for the entire report in the Object Navigator, the template will be applied to the Main section of the report by default. To apply the template to a specific section of your report, select the node for that section in the Object Navigator.

2. Choose **Tools > Report Wizard**.
3. On the Template page, select a pre-defined template, or click **Template file**, and click **Browse** to open the desired template.
4. Click **Finish** to apply the template to the current report section.

See also

Section 2.7.1, "About templates"

Section 2.7.3, "About applying templates"

3.12.5 Formatting the report title in a template

To set default attributes (such as font and color) for the title in a selected template:

1. In the Object Navigator, expand the **Templates** node, then the **Paper Layout** node, then the **Section** node, then the **Body** node, then the **Default** node.
2. Double-click the properties icon next to the **Default** node to display the Property Inspector.
3. Under **Title**, modify the properties as desired.

To set the placement and override attributes (such as font and color) for the title in a selected template:

4. In the Paper Layout view of the template, create a boilerplate text object for the title in the margin.
5. Size and position the title, and modify the color and text attributes as desired.
6. Double-click the title object to display its Property Inspector.
7. Under **General Information**, set the Name property to B_OR$REPORT_TITLE.

See also

Section 2.1.1, "About report titles"

Section 2.7.1, "About templates"

3.12.6 Adding items and objects to a template

To add items and objects to a template:

1. In the Paper Layout view of the Template Editor, click the margin button in the toolbar to display the margin area.
2. Create objects in the margin of your template. You cannot create objects in the body of a template.

See also

Section 2.7.5, "About the Template Editor"

Section 2.7.1, "About templates"

Section 2.7.2, "About template attributes"

Section 2.7.4, "About inheritance in templates"

3.12.7 Modifying objects in the template margin

To modify objects in the margin of a template:

1. In the Paper Layout view of the Template Editor, click the Edit Margin button in the toolbar.
2. Modify objects in the margin of your template.

See also

Section 2.7.5, "About the Template Editor"

Section 2.7.1, "About templates"

Section 2.7.2, "About template attributes"

Section 2.7.4, "About inheritance in templates"

3.12.8 Modifying the color, pattern, or border of body objects in a template

Template objects can be modified using the same techniques as report objects. To change the color or pattern of objects in a template, however, you use the template Property Inspector, rather than the Color tools in the tool palette:

- The Fill Pattern property defines the pattern to use for the space enclosed by the objects. You can define the background and foreground colors of the fill pattern using the Foreground Color and Background Color properties.
- The Edge Pattern property defines the pattern to use for the borders of the objects. You can define the background and foreground colors of the edge pattern using the Edge Foreground Color and Edge Background Color properties.
- The Text Color property specifies the text color to use for the object(s).

See also

Section 2.4.5, "About changing colors and patterns"

Section 3.9.6.2, "Changing colors"

Section 3.9.6.3, "Changing patterns"

Section 2.7.1, "About templates"

3.12.9 Adding a template to the pre-defined templates list

To add your own template to the list of predefined templates in the Report Wizard:

1. In a text editor (e.g., Wordpad), open the preferences file. (On Windows, open *ORACLE_HOME*\CAUPREFS.ORA (user preferences) if it exists; otherwise, open *ORACLE_HOME*\CAGPREFS.ORA (global preferences). On UNIX, open *your_home_directory*/prefs.ora (user preferences) if it exists; otherwise, open *ORACLE_HOME*/tools/admin/prefs.ora (global preferences).)
2. Scroll down to the template descriptions identified by Reports.*xxx*_Template_Desc (where *xxx* specifies a report style: Tabular, BreakAbove, BreakLeft, FormLetter, Formlike, MailingLabel, Matrix, MatrixBreak).
3. For each report style for which the template is defined:
   - To the Reports.*xxx*_Template_Desc list, add the description that you want to appear on the Template page of the Report Wizard.
   - To the corresponding Reports.*xxx*_Template_File list, add the file name of your template in the same position as the addition you made to the description list.
4. Copy the template file (*filename*.tdf) to ORACLE_HOME/REPORTS/TEMPLATES.

To include a sample image of your template in the Report Wizard:

1. Copy or create the .bmp file you want to use as the image for the template and name it *yyy*.bmp, where *yyy* is the file name of your template.
2. Copy each .bmp file to *ORACLE_HOME*/REPORTS/TEMPLATES.

See also

Section 2.7.1, "About templates"

Section 2.7.3, "About applying templates"

3.13 Use PL/SQL in a Report or Template

This section provides procedures for the following tasks that you may perform as you use PL/SQL in a report or template:

- Using a built-in package
- Working with the PL/SQL Editor
- Creating or editing report-level or template-level PL/SQL
- Creating or editing an external PL/SQL library
- Compiling and running program units

3.13.1 Using a built-in package

To use a built-in package:

1. Choose **Tools > PL/SQL Editor** to display the PL/SQL Editor.
2. In the Object Navigator, expand the **Built-in Packages** node, then the package you want to use.
3. Right-click a procedure, function, or exception and choose **Paste Name** or **Paste Arguments** to copy a call to the package into your PL/SQL code.

See also

Section 2.6.11, "About built-in packages"

3.13.2 Working with the PL/SQL Editor

This section provides procedures for the following tasks that you may perform as you work with the PL/SQL Editor:

- Defining PL/SQL
- Searching and replacing text in a program unit
- Editing a program unit
- Inserting syntax into the PL/SQL Editor

3.13.2.1 Defining PL/SQL

To define PL/SQL:

1. In the PL/SQL Editor, type or edit the PL/SQL code for the program unit.
2. Click **Compile**.
3. If necessary, click an error to navigate to its location in the source code.

> **Tip:** Check for missing semicolons at the end of statements, or misspelled syntax.

4. When the code compiles successfully, click **Close**.

See also

Section 2.6.4, "About program units"

3.13.2.2 Searching and replacing text in a program unit

To search and replace text in a program unit:

1. In the Object Navigator, expand the **Program Units** node.
2. Double-click the PL/SQL Editor view icon next to the program unit you want to edit.
3. In the PL/SQL Editor, place your cursor where you wish to begin the search.
4. Choose **Edit > Find and Replace**.
5. In the dialog box, type your search criteria, and, optionally, the replace string. You can supply either a text string or a regular expression for your search text.
6. Click **Search**.
7. Upon locating an instance of the search criteria, click **Replace** to replace a single instance, **Replace All** to replace all instances, or edit the text directly in the PL/SQL Editor.
8. Click **Search** to proceed to the next instance, or click **Cancel** to close the dialog box.
9. When you have finished replacing all text in the program unit, click **Compile** to recompile the program unit.

Tip: If you are going to be replacing text in multiple program units, you can recompile them all at once when you are finished.

10. When the code compiles successfully, click **Close**.

Tip: If your edits are extensive, you may want to click **Apply** to save your changes incrementally, without having to compile.

See also

Section 2.6.4, "About program units"

3.13.2.3 Editing a program unit

To edit a program unit:

1. In the Object Navigator, expand the **Program Units** node.
2. Double-click the PL/SQL Editor view icon for the program unit you want to edit.
3. In the PL/SQL Editor, edit the PL/SQL for the program unit.

See also

Section 2.6.4, "About program units"

3.13.2.3.1 Editing features in the PL/SQL Editor The editing features in the PL/SQL Editor (and Stored PL/SQL Editor) include:

Table 3–7 Editing features of PL/SQL Editor

Editing Feature	Description
Automatic indent	When you press the Enter key at the end of a line, the next line is automatically indented.
Color syntax highlighting	Keywords, comments, strings, and symbols such as := and || are colored differently.
Column and line selection	You can select columns of text as well as lines of source code. To select a column, press the ALTkey, then click and drag horizontally.To select a line, click on the extreme left margin of the line in the Source pane.

Table 3–7 Editing features of PL/SQL Editor

Editing Feature	Description
Drag and drop text manipulation	Selected text may be copied or moved by dragging and dropping.To copy text, press the CTRL key and drag the selected text.To move text, simply drag the selected text.
Indent/Outdent lines	The Indent/Outdent commands in the Edit menu enable you to indent or outdent selected source lines.
Multiple split views	You can create up to four separate views of the current program unit.To create horizontal views, place the cursor on the split bar at the top of the vertical scroll bar. Then click and drag the split bar down.To create vertical views, place the cursor on the split bar at the far left of the horizontal scroll bar. Then click and drag the split bar to the right.To remove a horizontal or vertical view, double-click the split bar that separates the views.
Printing	Choose File Print to print the current program unit.
Unlimited undo/redo	The Undo/Redo commands on theEdit menu enable you to undo or redo changes as far back as the last save operation.

Note: These features are available in Microsoft Windows only. In UNIX, you can print the current program unit, and you can use TAB/Shift-TAB to indent/outdent selected lines.

3.13.2.3.2 Using the keyboard in the PL/SQL Editor The following keyboard actions are supported when using the PL/SQL Editor (and Stored PL/SQL Editor) in Microsoft Windows and UNIX:

Table 3–8 Using the keyboard in the PL/SQL Editor

To do the following	Use Keystroke	On Platform
Move cursor left one character	Left arrow key	Windows and UNIX
Select character as cursor moves left	Shift + Left arrow key	Windows and UNIX
Move cursor right one character	Right arrow key	Windows and UNIX
Select character as cursor moves right	Shift + Right arrow key	Windows and UNIX

Table 3–8 Using the keyboard in the PL/SQL Editor

To do the following	Use Keystroke	On Platform
Copy selected characters or words	Ctrl + C	Windows and UNIX
Cut selected characters or words	Ctrl + X	Windows and UNIX
Paste from the clipboard	Ctrl + V	Windows and UNIX
Delete character on the right of current cursor position, or delete selected characters or words	Delete key	Windows and UNIX
Delete character on the left of current cursor position	Backspace	Windows and UNIX
Move cursor to the end of the program unit	Ctrl + End	Windows
Move cursor to the start of the program unit	Ctrl + Home	Windows
Select source lines from the current cursor position to the end of the program unit	Ctrl + Shift + End	Windows
Select source lines from the current cursor position to the start of the program unit	Ctrl + Shift + Home	Windows
Move cursor to the end of current line	End	Windows
Move cursor to the start of current line	Home	Windows
Select characters from the current cursor position to the end of the current line	Shift + End	Windows

Table 3–8 Using the keyboard in the PL/SQL Editor

To do the following	Use Keystroke	On Platform
Select characters from the current cursor position to the start of the current line	Shift + Home	Windows
Indent selected line, or indent characters on the right of current cursor position	Tab Key	Windows and UNIX
Outdent selected line	Shift + Tab key	Windows and UNIX
Move cursor down one line	Down arrow key	Windows and UNIX
Select line as cursor moves down	Shift + Down arrow key	Windows and UNIX
Move cursor up one line	Up arrow key	Windows and UNIX
Select line as cursor moves up	Shift + Up arrow key	Windows and UNIX
Scroll down the program unit by the number of lines that are shown in the Source pane	PageDown key	Windows and UNIX
Scroll down the program unit by the number of lines that are shown in the Source pane and selects the lines at the same time	Shift + PageDown key	Windows and UNIX
Scroll up the program unit by the number of lines that are shown in the Source pane	PageUp key	Windows and UNIX
Scroll up the program unit by the number of lines that are shown in the Source pane and selects the lines at the same time	Shift + PageUp key	Windows and UNIX

Table 3–8 Using the keyboard in the PL/SQL Editor

To do the following	Use Keystroke	On Platform
Undo most recent action	Ctrl + Z	Windows
Revert most recent undo action	Ctrl + Y	Windows
Move cursor left one character	Left arrow key	Windows and UNIX
Select character as cursor moves left	Shift + Left arrow key	Windows and UNIX
Move cursor right one character	Right arrow key	Windows and UNIX

3.13.2.3.3 Using the mouse in the PL/SQL Editor The following table describes the mouse actions that are supported when using the PL/SQL Editor (and Stored PL/SQL Editor) in Microsoft Windows:

Table 3–9 Using the mouse in the PL/SQL Editor

To do the following	Use mouse action
Select characters in a range	Click and drag the cursor from the first character to the last character in the range you wish to select. (also supported in UNIX) or... Click the first character, then press the **Shift** key, and click the last character in the range you wish to select.
Select word under cursor	Double-click the word. (also supported in UNIX) or... Press the **Ctrl** key, then click the word you wish to select.
Select words in a range	Press the **Ctrl** key, then click and drag the cursor from the first word to the last word in the range you wish to select.
Select a line	Place the cursor on the left margin of the line you wish to select. Click when the cursor changes to an arrow.
Select multiple lines	Place the cursor on the left margin of the first line you wish to select. When the cursor changes to an arrow, click and drag the cursor to the last line you wish to select.

Table 3–9 Using the mouse in the PL/SQL Editor

To do the following	Use mouse action
Select columns of text	Press the **Alt** key, then click and drag the cursor from the first column to the last column in the range you wish to select.
Move selected text	Select the text first. Then click and drag the selected text to its new position.
Copy selected text	Select the text first. Then press the **Ctrl** key, and drag the selected text to the location where you want to place a copy and release the mouse.
Split window into two horizontal views	Double-click the split bar at the top of the vertical scroll bar. or... Click and drag the split bar at the top of the vertical scroll bar.
Split window into two vertical views	Double-click the split bar at the extreme left of the horizontal scroll bar. or... Click and drag the split bar at the extreme left of the horizontal scroll bar.
Adjust relative size of split views	Click and drag the split bar that separates the views.
Remove split views	Double-click the split bar that separates the views. or... Click and drag the split bar to the edge of the window. To remove all four views at once, double-click the intersection where the split bars meet or drag it to any corner of the window.
Select characters in a range	Click and drag the cursor from the first character to the last character in the range you wish to select. (also supported in UNIX) or... Click the first character, then press the **Shift** key, and click the last character in the range you wish to select.
Select word under cursor	Double-click the word. (also supported in UNIX) or... Press the **Ctrl** key, then click the word you wish to select.
Select words in a range	Press the **Ctrl** key, then click and drag the cursor from the first word to the last word in the range you wish to select.

Table 3–9 Using the mouse in the PL/SQL Editor

To do the following	Use mouse action
Select a line	Place the cursor on the left margin of the line you wish to select. Click when the cursor changes to an arrow.
Select multiple lines	Place the cursor on the left margin of the first line you wish to select. When the cursor changes to an arrow, click and drag the cursor to the last line you wish to select.
Select columns of text	Press the **Alt** key, then click and drag the cursor from the first column to the last column in the range you wish to select.
Move selected text	Select the text first. Then click and drag the selected text to its new position.

3.13.2.4 Inserting syntax into the PL/SQL Editor

To copy syntax into the PL/SQL Editor:

1. Make sure the PL/SQL Editor or Stored PL/SQL Editor is the current (most recently selected) window.
2. Place the cursor in the editor where you want to insert the syntax, then choose **Tools > Syntax Palette**.
3. In the Syntax Palette, click the **PL/SQL** tab or the **Built-ins** tab.
4. Choose a PL/SQL category or a built-in package from the drop-down list.

   When you click a category or a package, the PL/SQL language elements or PL/SQL constructs that are available for selection appear in the list area below.
5. Choose a PL/SQL language element or construct in the list area.

   When you click a PL/SQL language element or construct, the syntax appears in the display area that is below the list area.
6. Click **Insert** to copy the selected syntax.

   The selected PL/SQL language element or construct is inserted into the active editor at the current cursor position.
7. Replace all lowercase items that are not comments with the appropriate values. Items within comments are optional. Reserved words are in uppercase.

Note: You can also double-click a PL/SQL language element or construct in the list area to insert the syntax into an editor.

See also

Section 2.6.3, "About the Syntax Palette"

3.13.3 Creating or editing report-level or template-level PL/SQL

This section provides procedures for the following tasks that you may perform as you create or edit report-level or template-level PL/SQL:

- Creating a local program unit
- Creating a stored program unit
- Deleting a program unit
- Moving a program unit between client and database server
- Creating a report trigger
- Deleting a report trigger
- Creating a database trigger

3.13.3.1 Creating a local program unit

To create a local (client-side) program unit:

1. In the Object Navigator, click the **Program Units** node.
2. Click the Create button in the toolbar.
3. In the dialog box, type a name for the program unit in the **Name** text box.
4. If your program unit is not a procedure (a PL/SQL subprogram that performs a specified sequence of actions), click one of the following:
   - **Function** (a PL/SQL subprogram that performs a specified sequence of actions, and then returns a value)
   - **Package Spec** (datatypes and subprograms that can be referenced by other program units)
   - **Package Body** (implementation of the package, which may include private subprograms and datatypes; optional if the package consists only of declarations)

5. Click **OK**.
6. In the PL/SQL Editor, define the PL/SQL for the program unit.

See also

Section 2.6.4, "About program units"

3.13.3.2 Creating a stored program unit

To create a stored (server-side) program unit:

1. In the Object Navigator, double-click the **Database Objects** node. If this node is disabled, the Connect dialog box displays so you can establish a database connection.
2. Expand the subnode that corresponds to the database user name you used to log in to the database to show the **PL/SQL Stored Program Units** node.
3. Click the **PL/SQL Stored Program Units** node, then click the Create button in the toolbar.
4. In the dialog box, type a name for the program unit in the **Name** text box.
5. If your program unit is not a procedure (a PL/SQL subprogram that performs a specified sequence of actions), click one of the following:
   - **Function** (a PL/SQL subprogram that performs a specified sequence of actions, and then returns a value)
   - **Package Spec** (datatypes and subprograms that can be referenced by other program units)
   - **Package Body** (implementation of the package, which may include private subprograms and datatypes; optional if the package consists only of declarations)
   - **Type Spec** (declares the name, variables (attributes) and member subprograms (methods) for an object type or collection type)
   - **Type Body** (implementation of the member methods (functions and procedures) defined in the type specification for an object type. For each method specified in an object type, there must be a corresponding method body)
6. Click **OK**.

7. In the Stored PL/SQL Editor, select a database owner name from the **Owner** drop-down list to indicate where the program unit will be stored in the database, then define the PL/SQL for the stored program unit.

See also

Section 2.6.5, "About stored program units"

3.13.3.3 Deleting a program unit

> **Note:** If you delete a PL/SQL package, function, or procedure, you must also delete all references to it in your report. Otherwise, you will get an error when you compile, generate, or run the report.

To delete a program unit:

1. In the Object Navigator, expand the **Program Units** node.
2. Click the program unit you want to delete.
3. Click the Delete button in the toolbar.
4. In the message dialog box, click **Yes** to confirm the deletion.

See also

Section 2.6.4, "About program units"

3.13.3.4 Moving a program unit between client and database server

To move a program unit *from the client to the database server*:

1. In the Object Navigator, double-click the **Database Objects** node. If this node is disabled, the Connect dialog displays so you can establish a database connection.
2. Expand the subnode that corresponds to the database user name you used to log in to the database to show the **PL/SQL Stored Program Units** node.
3. In the **Reports** section of the Object Navigator, expand the **Program Units** node.
4. Click the program unit you want to store in the database, and drag it from the **Program Units** node to the **Stored Program Units** subnode.
5. Release the mouse button to insert a *copy* of the program unit on the server.

To move a program unit *from the database server to the client*:

6. In the Object Navigator, double-click the **Database Objects** node. If this node is disabled, the Connect dialog box displays so you can establish a database connection.
7. Expand the subnode that corresponds to the database user name you used to log in to the database.
8. Expand the **Stored Program Units** node.
9. Click the stored program unit you wish to move to the client and drag it from the **Stored Program Units** node to the **Program Units** node in the **Reports** section of the Object Navigator.
10. Release the mouse button to insert a *copy* of the program unit on the client.

See also

Section 2.6.5, "About stored program units"

Section 2.6.4, "About program units"

3.13.3.5 Creating a report trigger

To create a report trigger:

1. In the Object Navigator, expand the **Report Triggers** node.
2. Double-click the PL/SQL icon for the trigger you want to create.
3. In the PL/SQL Editor, define the PL/SQL for the report trigger.

See also

Section 2.6.12.1, "About report triggers"

3.13.3.6 Deleting a report trigger

To delete a report trigger:

4. In the Object Navigator, expand the **Report Triggers** node.
5. Double-click the PL/SQL icon for the trigger you want to delete.
6. In the PL/SQL Editor, drag to select the PL/SQL code.
7. Choose **Edit > Delete**.

See also

Section 2.6.12.1, "About report triggers"

3.13.3.7 Creating a database trigger

To create a database trigger:

1. Choose **Tools > Database Trigger Editor**.
2. In the Database Trigger Editor, choose a user name (schema) from the **Table Owner** drop-down list.
3. Choose a table name from the **Table**: drop-down list.

   The **Table** drop-down list shows a list of table names owned by the user shown in the **Table Owner** field. If you select a user name other than your own in the **Table Owner** field, the **Table** drop-down list shows only the tables to which you have been granted access.
4. (*For views only*) To define an INSTEAD OF trigger for an object view, click the arrow next to **Table:** to display a drop-down list and choose **View**: Then choose a view name from the **View** drop-down list.
5. Click **New** to create a new database trigger.

   A unique trigger name (with respect to other triggers in the same schema) is automatically assigned to the new trigger in the **Name** drop-down list. You can modify the trigger name.

   The **Name** drop-down list displays a list of trigger names associated with the table (or view) shown in the **Table** (or **View**) field. The **Name** drop-down list displays only the names of the database triggers associated with the tables to which you have access.
6. After specifying the trigger options and action, click **Save** to compile the trigger.

See also

Section 2.6.12.4, "About database triggers"

3.13.4 Creating or editing object-level PL/SQL

This section provides procedures for the following tasks that you may perform as you create or edit object-level PL/SQL:

- Creating or editing a format trigger

- Creating or editing a group filter
- Creating or editing a formula column
- Creating a placeholder column
- Changing colors and patterns using PL/SQL

3.13.4.1 Creating or editing a format trigger

To create or edit a format trigger *using the Property Inspector*:

1. In the Paper Design view, double-click the object for which you want to create or edit a format trigger to display the Property Inspector.
2. Under **Advanced Layout**, set the Format Trigger property by clicking the ... button to display the PL/SQL Editor.
3. Define the PL/SQL for the format trigger.

To create or edit a format trigger *using the Object Navigator*:

1. In the Object Navigator, expand the **Paper Layout** node, then expand the node that contains the object for which you want to create or edit a format trigger.
2. Double-click the PL/SQL icon next to the object for which you want to create or edit a format trigger to display the PL/SQL Editor.
3. Define the PL/SQL for the format trigger.

3.13.4.2 Creating or editing a group filter

To create or edit a group filter:

1. In the Data Model view, double-click the title bar of the group to display the Property Inspector.
2. Scroll to the **Group** node.
3. To display the first n records for the group, set the Action Type property to *First* and set the Number of Records property to *n*.
4. To display the last n records for the group, set the Action Type property to *Last* and set the Number of Records property to *n*.
5. To create your own filter using PL/SQL, set the Action Type property to *PL/SQL* and set the PL/SQL Filter property by clicking **...** to display the PL/SQL Editor to define the PL/SQL for the filter.
6. Set other properties as desired.

See also

Section 2.6.9, "About group filters"

3.13.4.3 Creating or editing a formula column

See Section 3.8.7, "Creating or editing a formula column".

3.13.4.4 Creating a placeholder column

See Section 3.8.9, "Creating or editing a placeholder column".

3.13.4.5 Changing colors and patterns using PL/SQL

See Section 3.9.6.4, "Changing colors and patterns using PL/SQL".

3.13.5 Creating or editing an external PL/SQL library

This section provides procedures for the following tasks that you may perform as you create or edit an external PL/SQL library:

- Creating an external PL/SQL library
- Adding a program unit to an open library
- Editing a program unit in a PL/SQL library
- Removing a program unit from a PL/SQL library
- Attaching a PL/SQL library
- Converting external PL/SQL libraries

See also

Section 2.6.6, "About external PL/SQL libraries"

3.13.5.1 Creating an external PL/SQL library

To create an external PL/SQL library:

1. In the Object Navigator, click the **PL/SQL Libraries** node.
2. Click the create button in the toolbar.

The newly created library (initially named `LIB_xxx`) is automatically opened. Once a library has been created, its contents can be modified by inserting or removing program units, or attaching other libraries.

3.13.5.2 Adding a program unit to an open library

To add a program unit to an open library:

1. Perform one of the following steps in the Object Navigator, depending on your current state:
   - if the library is open, expand the library node
   - if the library is not currently open, click the **PL/SQL Libraries** node and choose **File > Open** to open the library
2. Under the **Reports** node, expand the **Program Units** node.
3. Drag the program unit you want to add to the library below the library's **Program Units** node.
4. Release the mouse button to insert a *copy* of the program unit in the library.

3.13.5.3 Editing a program unit in a PL/SQL library

To edit a program unit in a PL/SQL library:

1. In the Object Navigator, expand the **PL/SQL Libraries** node, then the library node for the program unit.
2. Under the library's **Program Units** node, double-click the PL/SQL Editor view icon for the program unit you want to edit.
3. In the PL/SQL Editor, edit the PL/SQL for the program unit.

3.13.5.4 Removing a program unit from a PL/SQL library

To remove a program unit from a PL/SQL library:

1. In the Object Navigator, expand the **PL/SQL Libraries** node, then the library node for the program unit.
2. Under the open library's **Program Units** node, click the library program unit you want to delete.
3. Click the Delete button in the toolbar.
4. In the message box, click **Yes**.

3.13.5.5 Attaching a PL/SQL library

To attach a PL/SQL library:

1. In the Object Navigator, click the **Attached Libraries** node.

2. Click the Create button in the toolbar.
3. In the Attach Library dialog box, type the name of the external PL/SQL library in the **Library** text box, or click **Browse** to search for the external PL/SQL library you want to reference.
4. Click **Attach**.

Restrictions

- If Reports Builder cannot find the specified library, a warning will be raised when you accept the dialog box, save the report, or open the report. If you try to run the report or compile the PL/SQL in it, an error will be raised.
- The Attached Libraries list is saved. The next time you open the report or library the list will have the same contents it did when you last saved the report.
- If an external library references another library, you must attach both libraries to the report even if the first library already has the second one attached.

3.13.5.6 Converting external PL/SQL libraries

To convert one or more report definitions or PL/SQL libraries from one storage format to another:

- On the command line, type `ORACLE_HOME\BIN\rwconverter`, followed by the report name and desired arguments. See the **Reference** section of the *Reports Builder online help* for more information about `rwconverter`.

3.13.6 Compiling and running program units

This section provides procedures for the following tasks that you may perform as you compile and run program units:

- Compiling a single program unit
- Compiling all program units
- Compiling all uncompiled program units

See also

Section 2.6.4, "About program units"

3.13.6.1 Compiling a single program unit

To compile a single program unit:

1. In the Object Navigator, under the **Program Units** node, click the program unit you want to compile (PL/SQL subprogram, report trigger, formula, group filter, format trigger, or validation trigger).
2. Choose **Program > Compile > Selection** to compile the selected program unit, regardless of its current compilation status.

> **Note:** An uncompiled program unit is indicated by an asterisk (*) after its name under the **Program Units** node in the Object Navigator. When you make changes to a program unit, dependent program units lose their compiled status.**:**

3. In the Compile dialog box, click any error, then click **GoTo** to navigate to the source of the error in the program unit.

> **Tip:** Check for missing semicolons at the end of statements, or misspelled syntax.

3.13.6.2 Compiling all program units

To compile all program units:

1. In the Object Navigator, click the report or library for which you want to compile all program units (including PL/SQL subprograms, report triggers, formulas, group filters, format triggers, and validation triggers).
2. Choose **Program > Compile > All** to compile all program units, regardless of their current compilation status.

> **Note:** An uncompiled program unit is indicated by an asterisk (*) after its name under the **Program Units** node in the Object Navigator. When you make changes to a program unit, dependent program units lose their compiled status.

3. In the Compile dialog box, click any error, then click **Go To** to navigate to the source of the error in the program unit.

> **Tip:** Check for missing semicolons at the end of statements, or misspelled syntax.

4. Compiling all uncompiled program units

To compile all uncompiled program units:

1. In the Object Navigator, click the report or library for which you want to compile all uncompiled program units (including PL/SQL subprograms, report triggers, formulas, group filters, format triggers, and validation triggers).
2. Choose **Program > Compile > Incremental** to compile all uncompiled program units.

> **Note:** An uncompiled program unit is indicated by an asterisk (*) after its name under the **Program Units** node in the Object Navigator. When you make changes to a program unit, dependent program units lose their compiled status.

3. In the Compile dialog box, click any error, then click **Go To** to navigate to the source of the error in the program unit.

> **Tip:** Check for missing semicolons at the end of statements, or misspelled syntax.

3.14 Debug a Report

This section provides procedures for the following tasks that you may perform as you debug a report:

- Debugging a report
- Running a report in debug mode
- Setting a breakpoint
- Setting a debug trigger
- Browsing debug actions
- Editing a debug action
- Disabling and enabling debug actions
- Deleting a debug action
- Running a program unit in the PL/SQL Interpreter
- Inserting a Navigator pane in the PL/SQL Interpreter
- Controlling program unit execution
- Stepping through the code
- Modifying code at runtime
- Displaying the current scope location
- Examining or changing local variables
- Modifying application variables
- Viewing subprogram references
- Tracing report execution
- Tracing report distribution
- Tracing using the SQL TRACE function

For conceptual information that supports these tasks, see Section 2.10, "Debugging Tools".

See also

Section 2.10.1, "About the debugging process"

Section 2.10.5, "About debug actions"

3.14.1 Debugging a report

To debug a report:

1. Run the report in debug mode (described below) to check for logical errors in the report, and displays these as warnings at runtime, before displaying the report output. Running a report in debug mode is not the same as debugging a report using the PL/SQL Interpreter.
2. After identifying a problem area, choose **Tools > PL/SQL Interpreter** to display the PL/SQL Interpreter, and create the desired debug actions (see Section 3.14.3, "Setting a breakpoint" and Section 3.14.4, "Setting a debug trigger") to isolate the failing code to a specific region of number of statement.
3. Close the PL/SQL Interpreter to run your report.
4. Browse your debug actions and evaluate application information.
5. After narrowing the failing code to a specific region, use the PL/SQL Interpreter to implement and test possible code fixes.

3.14.2 Running a report in debug mode

To compile and run a report in debug mode:

1. Choose **Edit > Preferences**.
2. In the Preferences dialog box, on the Runtime Settings page, select **Run Debug**. Click **OK**.
3. Click the Run Paper Layout button in the toolbar to run the report.

To compile and run a report in debug mode from the command line:

- On the `rwbuilder` or `rwrun` command line, specify RUNDEBUG=YES. For information about RUNDEBUG, see the **Reference > Command Line** section of the *Reports Builder online help*.

Usage notes

Running a report in debug mode specifies that you want extra runtime checking for logical errors in the report. It checks for things that are not errors but might result in undesirable output, and displays these as warnings at runtime, before displaying the report output. Running a report in debug mode is not the same as debugging a report using the PL/SQL Interpreter.

3.14.3 Setting a breakpoint

To set a breakpoint in the execution of your report:

1. If the PL/SQL Interpreter is not already displayed, choose **Tools > PL/SQL Interpreter**.
2. In the Object Navigator, single-click a compiled program unit node to display the program unit in the Interpreter Source pane.

   Note: Uncompiled program units are indicated by an asterisk (*) after their name.

3. Double-click an executable statement (a PL/SQL construct used for conditional, iterating, and sequential control, and for error handling. A semi-colon (;) must terminate every PL/SQL statement) where you wish to create the break point.

   Tip: You cannot place a breakpoint on a BEGIN, END, or NULL, statement, or on a comment.

   The breakpoint is inserted and is indicated by B00*n*, where *n* is the number of the breakpoint. When you run the program unit, execution is suspended at the line just prior to the breakpoint.

3.14.4 Setting a debug trigger

To set a debug trigger:

1. If the PL/SQL Interpreter is not already displayed, choose **Tools > PL/SQL Interpreter**.
2. In the Object Navigator, single-click a compiled program unit node to display the program unit in the Interpreter source pane.

   Note: Uncompiled program units are indicated by an asterisk (*) after their name.

3. In the Source pane, select the line where you want to create the debug trigger, then choose **Program > Debugging Triggers** (or right-click in the Source pane and choose **Debug Trigger**).

Tip: You cannot place a breakpoint on a BEGIN, END, or NULL, statement, or on a comment.

4. In the Trigger dialog box, define the trigger:
   - If you want the trigger to fire at different location than the current program unit, select a location from the **Location** drop-down list.
   - Type the debug trigger in the **Trigger Body** text box.

     For example, to create a debug trigger that interrupts program execution if the local NUMBER variable *my_sal* exceeds 5000, enter the following as the trigger body:

     ```
     IF Debug.Getn('my_sal') > 5000 THEN
      raise Debug.Suspend;
     END IF;
     ```

     Note: To create a debug trigger that contains multiple lines of text, include a BEGIN and an END statement around the code.

     You must raise the DEBUG.SUSPEND exception from the DEBUG package if you want the PL/SQL Interpreter to appear when this line is executed. Otherwise, Reports Builder executes the code silently and the PL/SQL Interpreter does not appear.
5. Click **OK** to create a debug trigger for the selected line.

Tip: You can also create a debug trigger by entering commands in the PL/SQL Interpreter pane.

3.14.5 Browsing debug actions

To browse debug actions:

- In the Object Navigator, expand the **Debug Actions** node. The Object Navigator displays a list of the debug actions associated with the current report.

  or

- In the PL/SQL Interpreter, use the DESCRIBE command to display information about a specific debug action.

For example, entering .DESCRIBE BREAK 1 in the Interpreter pane reveals the following about Breakpoint 1:

```
Breakpoint: 1
Program Unit: Procedure Body <programunit_name>
Line: 3
Enabled: YES
```

3.14.6 Editing a debug action

To edit a debug action:

1. In the Object Navigator, expand the **Debug Actions** node, then double-click the debug action icon to display the appropriate dialog box.
2. Edit the content or properties of the debug action in the dialog box.
3. Click **OK**.

3.14.7 Disabling and enabling debug actions

To disable/enable a debug action:

- In the Object Navigator, under the **Debug Actions** node, right-click the desired debug action and choose **Enable** or **Disable**.

3.14.8 Deleting a debug action

To delete a debug action:

1. In the Object Navigator, expand the **Debug Actions** node, then click the action you want to delete.
2. Click the Delete button in the toolbar.

3.14.9 Running a program unit in the PL/SQL Interpreter

To run a program unit in the PL/SQL Interpreter:

1. If the PL/SQL Interpreter is not already displayed, choose **Tools > PL/SQL Interpreter** to display it.
2. At the Interpreter's **`PL/SQL>`** prompt, type the name of the program unit followed by a terminating semi-colon (;). If the program unit requires any arguments, be sure to supply them in parentheses. For example:
   `getdata(SCOTT);`

3. Press the Enter or Return key to produce one of the following reactions:
   - Any output generated by the program unit is displayed at the command line, and the **`PL/SQL>`** prompt returns to indicate successful execution.
   - The secondary prompt appears (+>) indicating you have not finished entering an executable statement. If you forgot the terminating semicolon, enter it now and press Enter or Return. Otherwise, right-click and choose **New Prompt**.
   - Runtime errors are displayed at the command line, then the **`PL/SQL>`** prompt appears. You need to edit or debug your program unit.
   - If you have set a breakpoint or debug trigger in the program unit, execution is suspended and a new prompt is displayed as: **`(debug n)PL/SQL>`**.

Usage notes

Running a program unit as described above only works for procedures (or packaged procedures), not for functions (since there's no variable for a return value to be returned to).

3.14.10 Inserting a Navigator pane in the PL/SQL Interpreter

To insert an Object Navigator pane in the PL/SQL Interpreter:

1. If the Interpreter is not already displayed, choose **Program > PL/SQL Interpreter**.
2. Choose **View > Navigator Pane** to insert the Object Navigator pane in the middle of the PL/SQL Interpreter.

   Notice that the button bar is updated with new Object Navigator buttons.
3. Optionally, use the split bars to resize the proportions of the three panes.

3.14.11 Controlling program unit execution

Once you have inspected and modified the program state, you can resume or terminate execution using the following features:

Table 3–10 Program unit execution

Execution Feature	Description
STEP	You can use the STEP command to temporarily resume execution of an interrupted program. Control returns to the PL/SQL Interpreter after the specified set of statements have been executed. STEP Into or Over allows you to: execute the next statement (optionally descending into subprogram calls) resume execution until the current subprogram has returned continue execution until the specified source location is reached
GO	Use the GO command to resume program execution indefinitely--that is, until either the currently executing thread of execution terminates or it is interrupted again due to a debug action.
RESET	Use the RESET command to return control to an outer debug level without continuing execution in the current debug level. Thus, RESET effectively aborts execution at the current (and possibly higher) debug levels. You can explicitly reset execution to any previous debug level, or you can simply reset to top level, which is the default.

Execute these commands from either the PL/SQL Interpreter toolbar or by typing the command in the PL/SQL Interpreter pane.

3.14.12 Stepping through the code

To step through the code:

Before proceeding, you must already have set a debug action such as a breakpoint, and run your program unit at the PL/SQL Interpreter **`PL/SQL>`** prompt to suspend execution.

To step to the next line of the suspended program unit:

- Click the Step Into button in the PL/SQL Interpreter toolbar to execute the next line of executable code in the current program unit.

If the next executable line is a call to a nested subprogram (a program unit that is called from within another program unit), Step Into halts execution at the first line of the nested subprogram.

To step over a nested subprogram call in the suspended program unit:

- Click the Step Over button in the PL/SQL Interpreter toolbar.

  Step Over executes any calls to nested subprograms and then halts execution at the next executable line of the current program unit.

To step out of a nested subprogram and return to the outer program unit:

- Click the Step Out button in the PL/SQL Interpreter toolbar.

  If you previously used Step In to descend into a nested subprogram, Step Out completes execution of the nested subprogram and returns to the next line of the original program unit.

To resume program unit execution:

- Click the Go button in the PL/SQL Interpreter toolbar.

  Execution of the program unit continues until the program unit execution has finished, or until interrupted again by another debug action.

Note: If your debug action is located in a PL/SQL LOOP, using Go will cycle through the loop.

To exit suspended execution at the current debug level:

- Click the Reset button in the PL/SQL Interpreter toolbar.

  Control is returned to the Interpreter, or to an outer debug level (if any exist).

3.14.13 Modifying code at runtime

To modify your code at runtime:

1. In the Object Navigator, double-click the desired program unit, menu item command, or trigger to display the PL/SQL Editor.
2. In the PL/SQL Editor, make the desired modifications.
3. Click **Compile** then **Close** to dismiss the PL/SQL Editor.

4. In the PL/SQL Interpreter toolbar, choose **Go** or **Step Into**, **Over**, or **Out** to resume program execution.

See also

Section 2.10.9, "About modifying code at runtime"

3.14.14 Displaying the current scope location

To display the current scope location:

1. In the Object Navigator, expand the **Stack** node.
2. Expand the desired frame in the stack to reveal information about local variables and parameters.

   Or

   In the PL/SQL Interpreter, display the Source pane to view the current scope location.

See also

Section 2.10.7, "About the current scope location"

3.14.15 Examining or changing local variables

You must currently have a suspended program unit in the PL/SQL Interpreter to examine or change local variables values.

To examine local variable values:

1. In the Object Navigator, expand the **Stack** node to show the call stack frames.
2. Under the **Stack** node, double-click the node for the program unit whose variables you wish to examine or modify.

   Each local variable is displayed with its current value.

To edit local variable values:

3. In the Object Navigator, expand the **Stack** node to show the call stack frames.
4. Under the **Stack** node, double-click the node for the program unit whose variables you wish to examine or modify.

   Each local variable is displayed with its current value.
5. Click the variable value in the Object Navigator.

6. Click the variable value again to enter edit mode.
7. Change the value of the variable.
8. Click a blank area in the Object Navigator to exit edit mode and accept the changed value.
9. Resume program unit execution in the PL/SQL Interpreter to test the effect of the new value.

3.14.16 Modifying application variables

To modify a variable:

1. In the Object Navigator, expand the **Global Variables** node or the **Stack** node for local variables.
2. Select and expand the desired entry. The Object Navigator displays any variables associated with the entry.
3. Click the existing value after the "=" then edit the value by entering the desired value.

3.14.17 Viewing subprogram references

To view subprogram references:

1. In the Object Navigator, select and expand the desired subprogram.
2. Select and expand either the **References** or **Referenced By** node.

   The Object Navigator displays any subprogram references.

Note: **Referenced By** shows the program units that call the current program unit. **References** shows the program units that are called by the current program unit.

3.14.18 Tracing report execution

To set tracing options, do one of the following:

- Choose **Program > Tracing** and fill out the Runtime Trace Settings dialog box as desired.
- On the command line, specify tracing options using `rwbuilder` or `rwrun` with the TRACEFILE, TRACEMODE, and TRACEOPTS keywords.

- Specify tracing options via the Reports Builder built-in package using SRW.TRACE_START, SRW.TRACE_END, SRW.TRACE_ADD_OPTIONS, SRW.TRACE_REM_OPTIONS.

3.14.19 Tracing report distribution

To trace report distribution:

1. Choose **Program > Tracing**.
2. In the Runtime Trace Settings dialog box, specify a name for the trace file in the **Trace File** field.
3. Select the **Distribution** check box.
4. Click **OK**.
5. Run the report.
6. Use a text editor to open and view the trace file. If the trace file is empty, the distribution was successful. Otherwise, the trace file identifies the distribution error.

See also

Section 2.8.3, "About report distribution"

Section 3.7.11, "Distributing a report to multiple destinations"

3.14.20 Tracing using the SQL TRACE function

The TRACE function provides you with the exact statements that are being parsed. Once you have them, you can time them in SQL*Plus, and multiply these times with the expected number of rows to retrieve from the database. (Always compare apples to apples, i.e., send output to a file--not to the screen. Do not change anything in the SQL statements when moving it to SQL*Plus; even the slightest change in the WHERE clause can make a big difference in the performance.)

There are two ways to trace your reports:

- user level
- system level

We recommend the user level because you can more easily find the information you need. For more information on SQL TRACE, see the *Oracle9i Server SQL Language Reference* Manual.

3.14.20.1 Performing a user-level trace

1. Open the report for which you want the performance data.
2. Create a report-level formula column named SQL-TRACE that has the following formula:

```
SRW.DO_SQL ('ALTER SESSION SET SQL_TRACE=TRUE');
return(1); --Formulas must return a value.
```

> **Note:** You could also call SQL TRACE from the Before Form trigger.

3. Run the report. A new file, <some_number>.trc will be created in either ORACLE_HOME/rdbms/log, or the destination indicated by the init.ora parameter USER_DUMP_DEST. (The date stamp on the file can help you determine which .trc file is yours.)
4. Use the TKPROF command to format the trace output file.

If you issue more than one trace during the rwbuilder session, the trace outputs are concatenated into one file.

3.14.20.2 Performing a system-level trace

5. Insert these statements into your init.ora file: SQL_TRACE=TRUE and TIMED_STATISTICS=TRUE.
6. Shut down, then restart your database.

> **Note:** Every interaction with the database will be traced, and the `ORACLE_HOME/rdbms/log` is likely to grow very large.

3.15 Integrate with Other Products

This section provides procedures for the following tasks that you may perform as you integrate your reports with other products:

- Accessing non-Oracle data sources
- Publishing a report as a portlet in OracleAS Portal

3.15.1 Accessing non-Oracle data sources

Oracle Reports allows you to access any data source. The new pluggable data source (PDS) architecture replaces Oracle Open Client Adapter (OCA), and the Open Database Connectivity (ODBC) drivers are no longer supported in Oracle Reports. However, Java Database Connectivity (JDBC) is one of the pluggable data sources available that can utilize the JDBC-ODBC bridge, allowing access to other data sources.

Use the following steps to determine the easiest method for accessing your data source:

1. Does your data source have Java Database Connectivity (JDBC)?

   A pluggable data source (PDS) has already been set up. All you have to do is select JDBC as your data type.

2. Does your data source have ODBC?

   If so, make sure that you have downloaded the JDBC-ODBC bridge from Sun Microsystems. This lets you use the JDBC connection mentioned in Step 1. If you do not know if there is an ODBC driver for your data source, check the company Web site. For example, Microsoft has a downloadable ODBC driver for Excel.

3. Is the file in XML?

   Oracle Reports includes an XML PDS. If the DTD file is available, all you have to do is select XML as your data type. If the DTD file is not available, you will have to create one.

4. Is the file comma-delimited?

   You can use the Text PDS included with Oracle Reports to query the data source.

5. Has someone in your company built a PDS for your data type?

If the PDS exists, it will probably be on the Reports Server. Ask your administrator if such a file exists. If the PDS exists, add it to the classpath for your copy of Reports Builder. If the PDS does not exist, use the API Reference to help you build one.

Examples

Oracle Reports ships with XML data source sample files.

See also

The **Pluggable Data Sources** section of the *Reports Builder online help*, including the topics:

- About pluggable data sources
- Adding a pluggable data source
- Connecting to a pluggable data source
- Adding Online Help to a pluggable data source
- Pluggable data source interface definition
- Troubleshooting PDS problems

3.15.2 Publishing a report as a portlet in OracleAS Portal

OracleAS provides a portal building application called OracleAS Portal, which is integrated with Oracle Reports. This integration enables you to quickly publish a report as a portlet (i.e., a small, dynamic region) or an item on a page, so that your users can easily access dynamic reports

For more information, refer to the section titled "Integration with OracleAS Portal" in Getting Started with Oracle Reports, available on the Oracle Technology Network Oracle Reports Documentation page (`http://otn.oracle.com/docs/products/reports/content.html`).

3.16 Administer Reports Builder

This section provides procedures for the following tasks that you may perform as you administer Reports Builder:

- Setting a database role
- Converting from one format to another
- Improving performance using SQL statements
- Improving performance using WHERE clauses

3.16.1 Setting a database role

Before beginning this procedure, verify that the database administrator has created the role, granted privileges to the role, and granted the role to approved end users. Refer to your *Oracle9i Application Developer's Guide* for more information.

To set a database role for a report:

1. In the Object Navigator, double-click the properties icon for the report to display the Property Inspector.
2. Under the **Report** node, set the Role Name property as defined by the database administrator in the database.
3. Optionally, to set a Role Password, double-click the button in the Role Name value field to display the Set Role dialog box.

Usage notes

- The role settings in the report are overridden if you specify a role using the command line with Reports Runtime (`rwrun`).
- You can set only one role for a report. To set multiple roles, you need to set a role to which the other roles have been granted.

Caution: Do not attempt to set the role in a PL/SQL trigger. The PL/SQL will not compile.

See also

Section 2.9.1, "About database roles"

3.16.2 Converting from one format to another

To convert one or more report definitions or PL/SQL libraries from one storage format to another:

- In Reports Builder, choose **Tools > File Conversion** to display the Convert dialog box.
- On the command line, type %ORACLE_HOME%\bin\rwconverter, followed by the report name and desired arguments.

3.16.3 Improving performance using SQL statements

Performing operations in SQL is faster than performing them in Reports Builder or PL/SQL. The following are the most common cases where using SQL would improve performance:

- use a WHERE clause instead of a group filter or format trigger to exclude records
- use the SUBSTR function to truncate character strings instead of truncating in Reports Builder
- perform calculations directly in your query rather than in a formula or summary

  Rationale: SQL can perform calculations more quickly than a summary or formula. WHERE and SUBSTR can reduce unnecessary fetching because they operate on the data during rather than after data retrieval. Improvements in performance become more noticeable when retrieving thousands of records versus a few records.

3.16.4 Improving performance using WHERE clauses

Consider adding a WHERE clause for the matrix cell query in a multiquery data model. If you are using a multiquery data model and your dimension queries are restricted by a WHERE clause, adding a WHERE clause to the matrix cell query ensures that you do not retrieve more records than are necessary. For example, suppose that you had the following queries for your dimensions:

Q_Dept

```
SELECT DEPTNO FROM DEPT
WHERE DEPTNO < 100
```

Q_Job

```
SELECT DISTINCT JOB FROM EMP
WHERE DEPTNO < 100
```

Q_Filler

To ensure that your cell query only retrieves the records that are necessary, you would write the following SELECT statement:

```
SELECT DEPTNO, JOB, SUM(SAL) FROM EMP
WHERE DEPTNO IN (SELECT DEPTNO FROM DEPT
WHERE DEPTNO < 100) AND JOB IN (SELECT
DISTINCT JOB FROM EMP WHERE DEPTNO < 100)
GROUP BY DEPTNO, JOB
```

If you did not add the WHERE clause to this query, all rows would be retrieved from the database, regardless of what you selected in Q_Dept and Q_Job.

Note: If you added a WHERE clause that did not use the subqueries (e.g., WHERE EMP.DEPTNO = DEPT.DEPTNO), the query would be executed once for each combination of values in the cross-product. This can lead to excessive execution of the filler query, if the cross-product has a lot of combinations.

4

Visual Index

This chapter provides an overview of the example reports in this manual. For information on accessing the demo reports and the data sources, refer to "Accessing the example reports" and "Accessing the data sources".

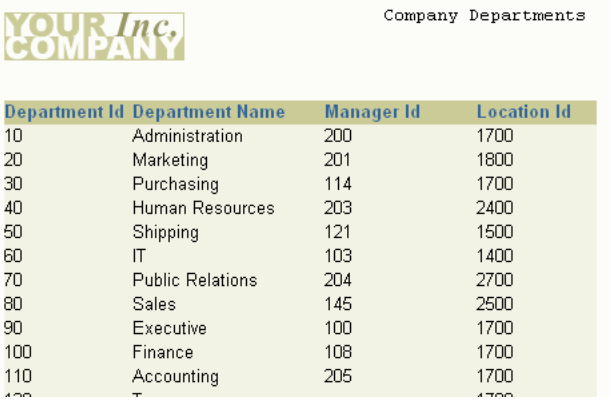
The chapter lists the example reports contained in this manual, a brief description of each report, a pointer to the location of the steps to build the example, and an image of the sample output.

4.1 Part 1: Building Basic Reports

4.1.1 Building a tabular report

In this example, you will use the Report Wizard to build a simple tabular report.

Figure 4–1 Final output of the tabular report example



For more information on building this example, refer to Chapter 5, "Building a Tabular Report".

4.1.2 Building a mailing label report

In this example, you will use the Report Wizard to build a simple mailing label report. The steps will show you how to fill out the provided template to format your mailing labels.

Figure 4–2 Final output of the mailing label report example

For more information on building this example, refer to Chapter 6, "Building a Mailing Label Report".

4.1.3 Building a form letter report

In this example, you will use the Report Wizard to build a simple form letter report. The steps will show you how to fill out the provided template to design your form letter.

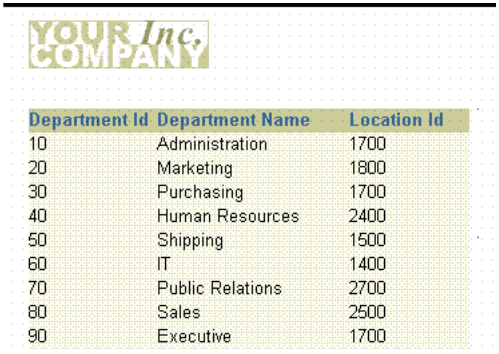
Figure 4–3 Final output of the form letter example

For more information on building this example, refer to Chapter 7, "Building a Form Letter Report".

4.1.4 Building a master/master report

In this example, you will learn how to build a master/master report, which displays at least two sets of data which are not directly related. That is, the records constituting the data are fetched using at least two separate queries. A master/master report (also called a parent/parent report) contains two or more queries with no links (parent/child relationships).

Figure 4–4 Final output of the first master/master report example



Department Id	Department Name	Location Id
10	Administration	1700
20	Marketing	1800
30	Purchasing	1700
40	Human Resources	2400
50	Shipping	1500
60	IT	1400
70	Public Relations	2700
80	Sales	2500
90	Executive	1700

Figure 4–5 Final output of the second master/master report example

Last Name	First Name	Job Id	Salary
Abel	Ellen	SA_REP	$11000.00
Ande	Sundar	SA_REP	$6400.00
Atkinson	Mozhe	ST_CLERK	$2800.00
Austin	David	IT_PROG	$4800.00
Baer	Hermann	PR_REP	$10000.00
Baida	Shelli	PU_CLERK	$2900.00
Banda	Amit	SA_REP	$6200.00

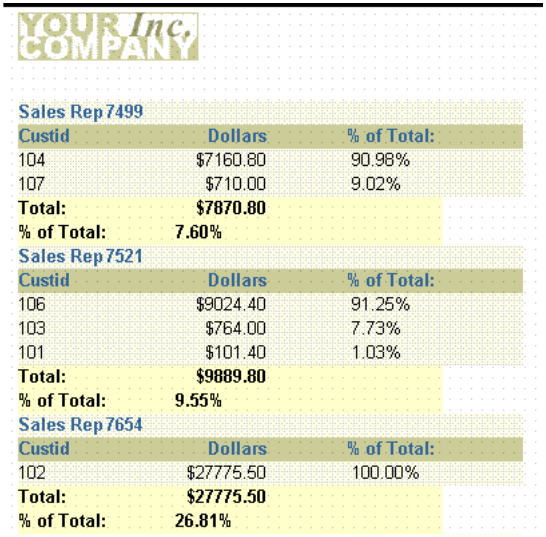
For more information on building this example, refer to Chapter 8, "Building a Master/Master Report".

4.1.5 Building a summary report

In this example, you will learn how to use the Summary Column tool to create a summary report.

A summary report contains at least one column whose value or values consist of a summary of other data. A column that totals sales, a column that averages a list of commissions, and a column that shows the maximum amounts found in a series of purchase orders are all examples of summary columns.

Figure 4–6 Final output of the summary report example



YOUR Inc. COMPANY

Sales Rep 7499		
Custid	Dollars	% of Total:
104	$7160.80	90.98%
107	$710.00	9.02%
Total:	**$7870.80**	
% of Total:	**7.60%**	
Sales Rep 7521		
Custid	Dollars	% of Total:
106	$9024.40	91.25%
103	$764.00	7.73%
101	$101.40	1.03%
Total:	**$9889.80**	
% of Total:	**9.55%**	
Sales Rep 7654		
Custid	Dollars	% of Total:
102	$27775.50	100.00%
Total:	**$27775.50**	
% of Total:	**26.81%**	

For more information on building this example, refer to Chapter 9, "Building a Summary Report".

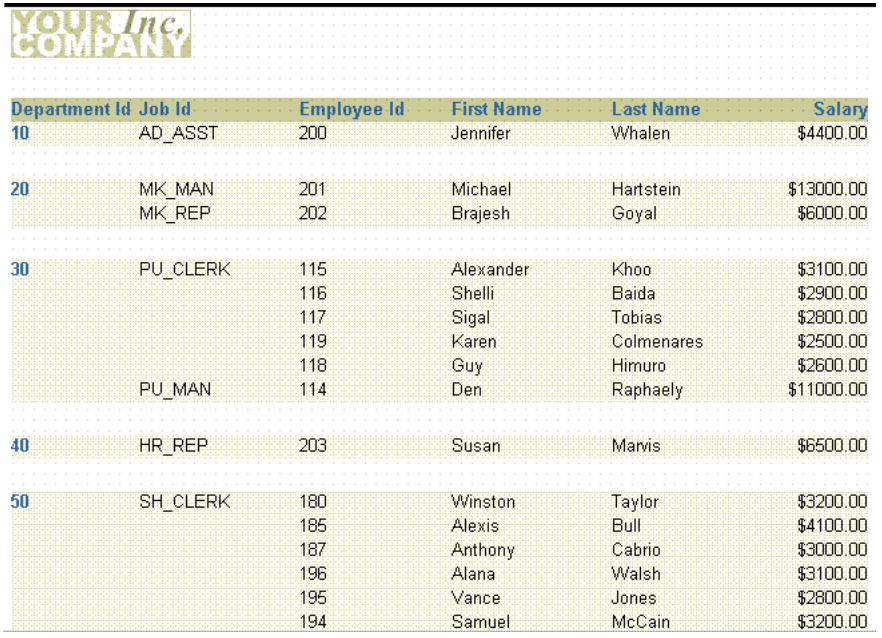
4.2 Part 2: Building Group Reports

4.2.1 Building a single-query group report

In this example, you will build a simple group left report using one query.

Group left and group above reports divide the rows of a report into ”sets,” based on common values in one or more of the columns, such as the department number in the example above. In the sample output below, notice that each department number prints only once. If the report above was not a group report, the department number would print once for each employee in the department rather than just once for the whole department.

Figure 4–7 Final output of the single-query group report example



Department Id	Job Id	Employee Id	First Name	Last Name	Salary
10	AD_ASST	200	Jennifer	Whalen	$4400.00
20	MK_MAN	201	Michael	Hartstein	$13000.00
	MK_REP	202	Brajesh	Goyal	$6000.00
30	PU_CLERK	115	Alexander	Khoo	$3100.00
		116	Shelli	Baida	$2900.00
		117	Sigal	Tobias	$2800.00
		119	Karen	Colmenares	$2500.00
		118	Guy	Himuro	$2600.00
	PU_MAN	114	Den	Raphaely	$11000.00
40	HR_REP	203	Susan	Marvis	$6500.00
50	SH_CLERK	180	Winston	Taylor	$3200.00
		185	Alexis	Bull	$4100.00
		187	Anthony	Cabrio	$3000.00
		196	Alana	Walsh	$3100.00
		195	Vance	Jones	$2800.00
		194	Samuel	McCain	$3200.00

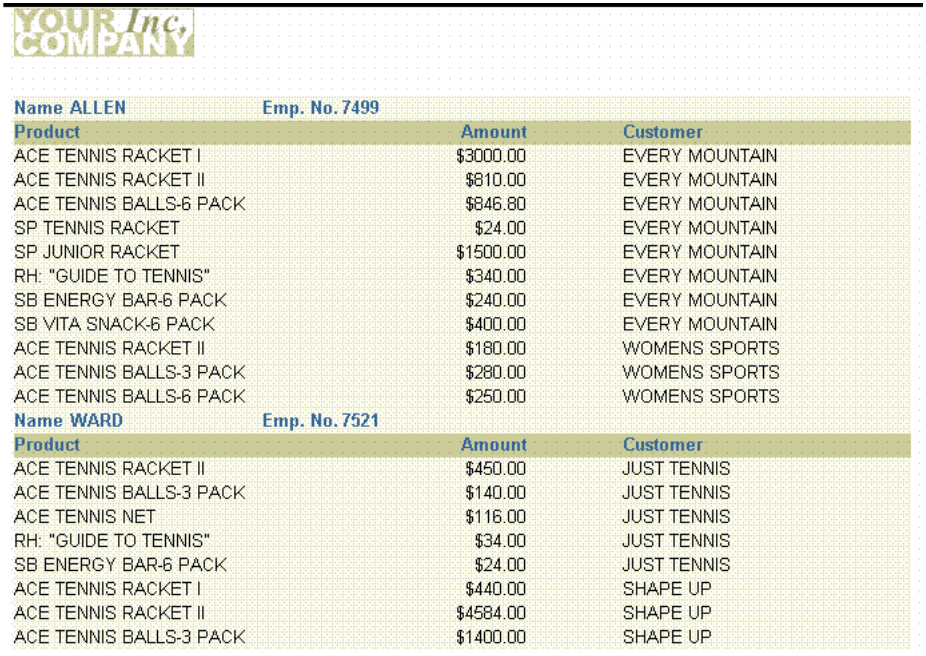
For more information on building this example, refer to Chapter 10, "Building a Single-Query Group Report".

4.2.2 Building a two-query group report

In this example, you will build a group above report using two queries.

A two-query group report appears much the same as a single-query group report. Performance is the key issue when contrasting single-query and multiple-query group reports. In most cases, single-query reports will run faster than multiple-query reports. Multiple-query reports are, however, sometimes easier to understand conceptually and easier to maintain. For example, if you are in a situation where only a few users run the report and the report returns a relatively small number of records, you might want to use multiple queries to simplify maintenance and make the data model easier to understand. If you have many users and the report is quite large, then you should try to use a single-query report.

Figure 4–8 Final output of the two-query group report example



YOUR Inc. COMPANY

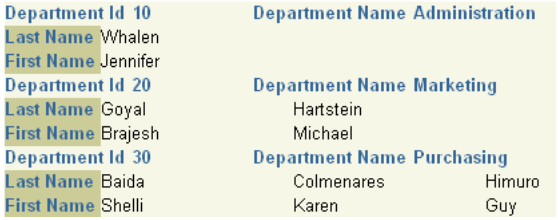
Name ALLEN	Emp. No. 7499		
Product		**Amount**	**Customer**
ACE TENNIS RACKET I		$3000.00	EVERY MOUNTAIN
ACE TENNIS RACKET II		$810.00	EVERY MOUNTAIN
ACE TENNIS BALLS-6 PACK		$846.80	EVERY MOUNTAIN
SP TENNIS RACKET		$24.00	EVERY MOUNTAIN
SP JUNIOR RACKET		$1500.00	EVERY MOUNTAIN
RH: "GUIDE TO TENNIS"		$340.00	EVERY MOUNTAIN
SB ENERGY BAR-6 PACK		$240.00	EVERY MOUNTAIN
SB VITA SNACK-6 PACK		$400.00	EVERY MOUNTAIN
ACE TENNIS RACKET II		$180.00	WOMENS SPORTS
ACE TENNIS BALLS-3 PACK		$280.00	WOMENS SPORTS
ACE TENNIS BALLS-6 PACK		$250.00	WOMENS SPORTS
Name WARD	**Emp. No. 7521**		
Product		**Amount**	**Customer**
ACE TENNIS RACKET II		$450.00	JUST TENNIS
ACE TENNIS BALLS-3 PACK		$140.00	JUST TENNIS
ACE TENNIS NET		$116.00	JUST TENNIS
RH: "GUIDE TO TENNIS"		$34.00	JUST TENNIS
SB ENERGY BAR-6 PACK		$24.00	JUST TENNIS
ACE TENNIS RACKET I		$440.00	SHAPE UP
ACE TENNIS RACKET II		$4584.00	SHAPE UP
ACE TENNIS BALLS-3 PACK		$1400.00	SHAPE UP

For more information on building this example, refer to Chapter 11, "Building a Two-Query Group Report".

4.2.3 Building an across group report

In this example, you will build an across group report that prints the values of a database column across the page instead of down.

Figure 4–9 Final output of the across group report example



For more information on building this example, refer to Chapter 12, "Building an Across Group Report".

4.2.4 Building a group left summary report

In this example, you will build a summary report that groups the summaries on the left in the report output.

This report consists of master records (Name, at the upper left of the figure below), detail records (Product, Itemtot, and Orderdate, to the upper right), and summary records (Product, and Sum Total). The summary calculates totals for the details under each master record. Notice that the column Product appears twice. With Reports Builder, you can display columns any number of times.

Figure 4–10 Final output of the group left summary report example

YOUR Inc. COMPANY

Name	Descrip	Itemtot	Orderdate
EVERY MOUNTAIN	ACE TENNIS BALLS-6 PACK	$5.60	18-JUL-86
		$11.20	25-JUL-86
		$550.00	15-JAN-87
		$280.00	22-FEB-87
	ACE TENNIS RACKET I	$3000.00	15-JAN-87
	ACE TENNIS RACKET II	$810.00	15-JAN-87
	RH: "GUIDE TO TENNIS"	$340.00	22-FEB-87
	SB ENERGY BAR-6 PACK	$240.00	22-FEB-87
	SB VITA SNACK-6 PACK	$400.00	22-FEB-87
	SP JUNIOR RACKET	$1500.00	15-JAN-87
	SP TENNIS RACKET	$24.00	25-JUL-86

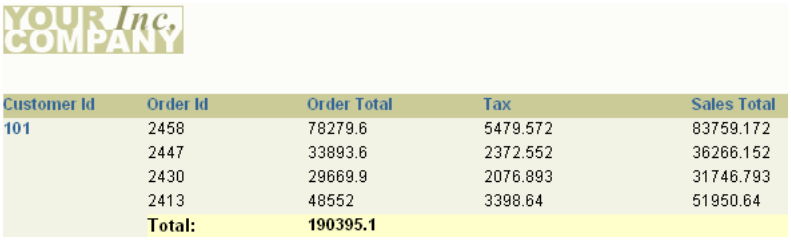
Product	Sum Total
ACE TENNIS BALLS-6 PACK	$846.80
ACE TENNIS RACKET I	$3000.00
ACE TENNIS RACKET II	$810.00
RH: "GUIDE TO TENNIS"	$340.00
SB ENERGY BAR-6 PACK	$240.00
SB VITA SNACK-6 PACK	$400.00
SP JUNIOR RACKET	$1500.00
SP TENNIS RACKET	$24.00

For more information on building this example, refer to Chapter 13, "Building a Group Left Summary Report".

4.2.5 Building a group left formula report

In this example, you will use the Report Wizard to set up your report and write the one query that selects all the necessary database columns. You will then manually create the two formula columns to calculate tax and order totals for each customer, then add the formula columns to your report. You will use a Group Left style report to make the data in the report easy to read.

Figure 4–11 Final output of the group left formula report example



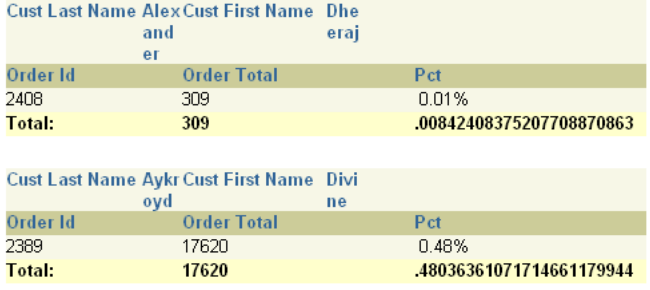
For more information on building this example, refer to Chapter 14, "Building a Group Left Formula Report".

4.3 Part 3: Building Reports with Special Formatting

4.3.1 Building a wrapped field report

In this example, you will build a break report where the line wraps on word boundaries if it is too long to fit on one line.

Figure 4–12 Final output of the wrapped field report example

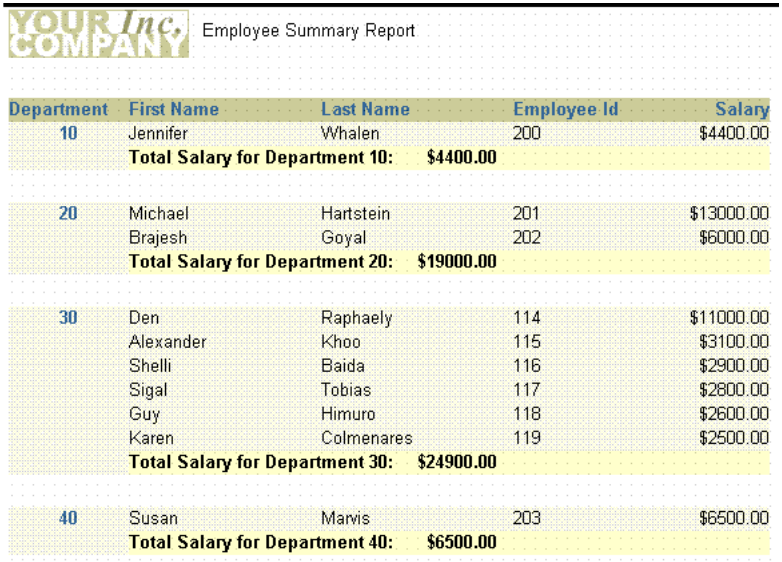


For more information on building this example, refer to Chapter 15, "Building a Wrapped Field Report".

4.3.2 Building a header and footer report

In this example, you will build a report that has a page header printed in the upper margin area of every page of the report, and a footer printed at the end of the list of employee information for each department.

Figure 4–13 Final output of the header and footer report example



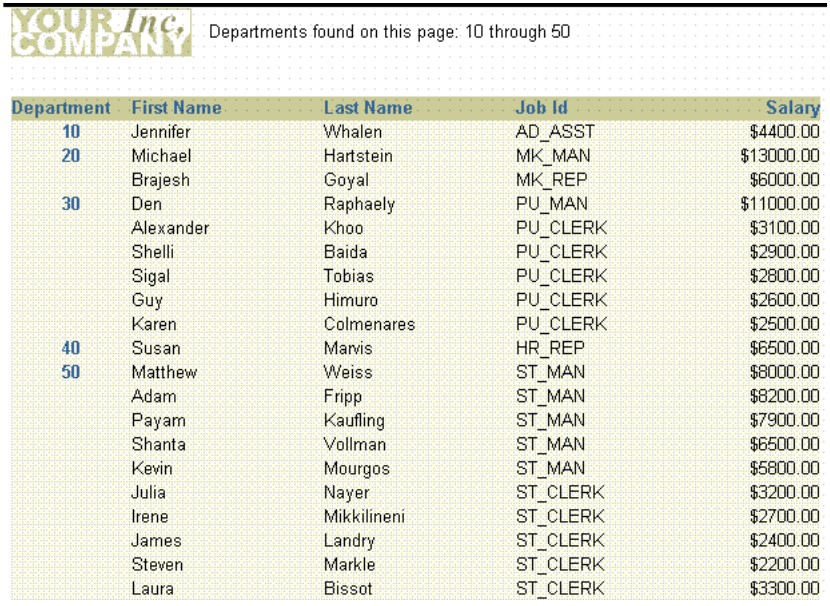
Department	First Name	Last Name	Employee Id	Salary
10	Jennifer	Whalen	200	$4400.00
	Total Salary for Department 10:	**$4400.00**		
20	Michael	Hartstein	201	$13000.00
	Brajesh	Goyal	202	$6000.00
	Total Salary for Department 20:	**$19000.00**		
30	Den	Raphaely	114	$11000.00
	Alexander	Khoo	115	$3100.00
	Shelli	Baida	116	$2900.00
	Sigal	Tobias	117	$2800.00
	Guy	Himuro	118	$2600.00
	Karen	Colmenares	119	$2500.00
	Total Salary for Department 30:	**$24900.00**		
40	Susan	Marvis	203	$6500.00
	Total Salary for Department 40:	**$6500.00**		

For more information on building this example, refer to Chapter 16, "Building a Header and Footer Report".

4.3.3 Building a header with database values report

In this example, both the first and last department numbers found on each page are displayed in the page header.

Figure 4–14 Final output of the header with database values report example



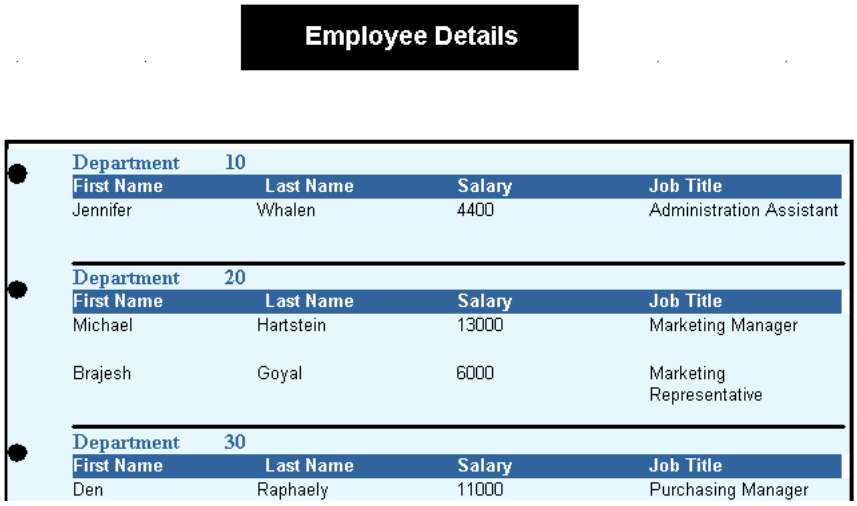
Department	First Name	Last Name	Job Id	Salary
10	Jennifer	Whalen	AD_ASST	$4400.00
20	Michael	Hartstein	MK_MAN	$13000.00
	Brajesh	Goyal	MK_REP	$6000.00
30	Den	Raphaely	PU_MAN	$11000.00
	Alexander	Khoo	PU_CLERK	$3100.00
	Shelli	Baida	PU_CLERK	$2900.00
	Sigal	Tobias	PU_CLERK	$2800.00
	Guy	Himuro	PU_CLERK	$2600.00
	Karen	Colmenares	PU_CLERK	$2500.00
40	Susan	Marvis	HR_REP	$6500.00
50	Matthew	Weiss	ST_MAN	$8000.00
	Adam	Fripp	ST_MAN	$8200.00
	Payam	Kaufling	ST_MAN	$7900.00
	Shanta	Vollman	ST_MAN	$6500.00
	Kevin	Mourgos	ST_MAN	$5800.00
	Julia	Nayer	ST_CLERK	$3200.00
	Irene	Mikkilineni	ST_CLERK	$2700.00
	James	Landry	ST_CLERK	$2400.00
	Steven	Markle	ST_CLERK	$2200.00
	Laura	Bissot	ST_CLERK	$3300.00

For more information on building this example, refer to Chapter 17, "Building a Header with Database Values Report".

4.3.4 Building a report with graphics, text, and color

In this example, you will build a report and enhance it by adding an image to the margin, a title, and a border. You will also change the look of the report by applying different fonts and text styles.

Figure 4–15 Final output of the graphics, text, and color example

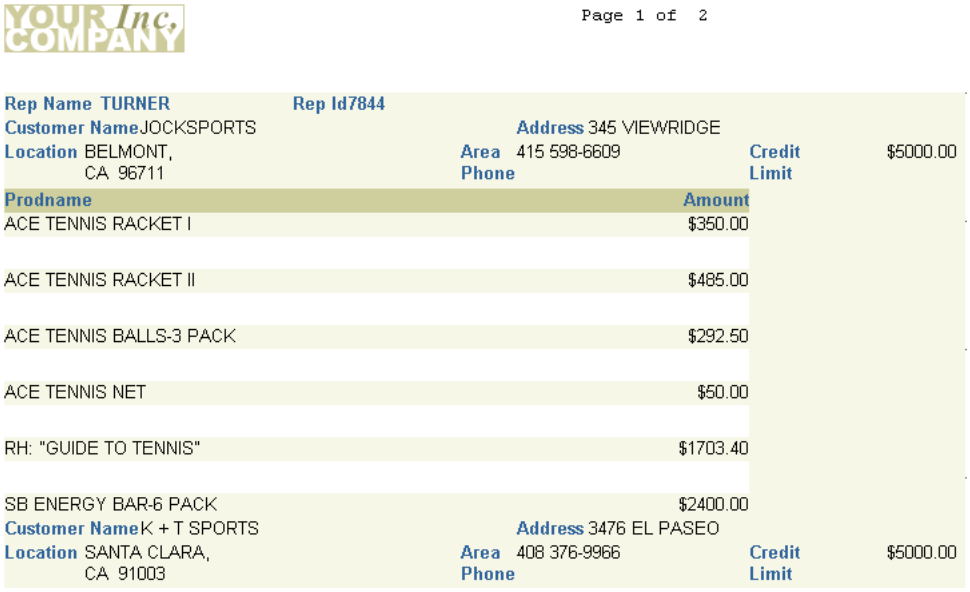


For more information on building this example, refer to Chapter 18, "Building a Report with Graphics, Text, and Color".

4.3.5 Building a report that renumbers pages by repeating frame

In this example, you will build a report that numbers pages using the format "Page *X* of *Y* Pages". The first number (*X*) corresponds to the current page for each parent record (i.e., each sales representative). This page number is reset to ”1” for each sales representative, thus tracking the statistics of each representative separately.

Figure 4–16 Final output of the renumbering pages by repeating frame example

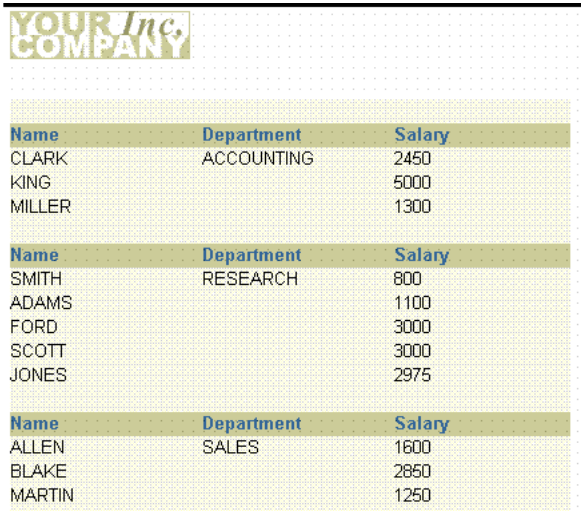


For more information on building this example, refer to Chapter 19, "Building a Report that Renumbers Pages by Repeating Frame".

4.3.6 Building an intermixed fields report

In this example, you will build a report where the group field appears between its related fields. Normally, a group (break) field appears to the left of (in group left report) or above (in group above report) its related fields.

Figure 4–17 Final output of the intermixed fields report example



Name	Department	Salary
CLARK	ACCOUNTING	2450
KING		5000
MILLER		1300

Name	Department	Salary
SMITH	RESEARCH	800
ADAMS		1100
FORD		3000
SCOTT		3000
JONES		2975

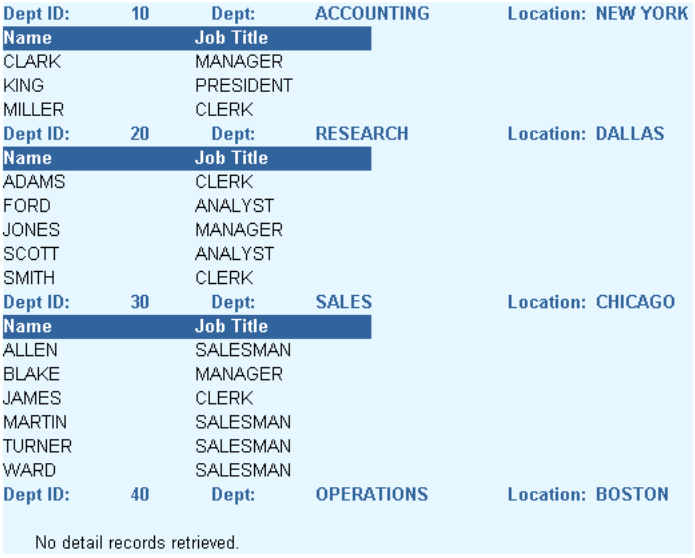
Name	Department	Salary
ALLEN	SALES	1600
BLAKE		2850
MARTIN		1250

For more information on building this example, refer to Chapter 20, "Building an Intermixed Fields Report".

4.3.7 Building a report that suppresses labels

In this example, you will build a master/detail report that fetches a master record with no associated details. In the sample output below, notice how the field labels for Department 40 do not display because no detail records were found. In this chapter, you will learn how to suppress the detail information for a single record, but allow the other master/detail records to display.

Figure 4–18 Final output for the suppressing labels example



Dept ID:	**10**	**Dept:**	**ACCOUNTING**	**Location:**	**NEW YORK**
Name		**Job Title**			
CLARK		MANAGER			
KING		PRESIDENT			
MILLER		CLERK			
Dept ID:	**20**	**Dept:**	**RESEARCH**	**Location:**	**DALLAS**
Name		**Job Title**			
ADAMS		CLERK			
FORD		ANALYST			
JONES		MANAGER			
SCOTT		ANALYST			
SMITH		CLERK			
Dept ID:	**30**	**Dept:**	**SALES**	**Location:**	**CHICAGO**
Name		**Job Title**			
ALLEN		SALESMAN			
BLAKE		MANAGER			
JAMES		CLERK			
MARTIN		SALESMAN			
TURNER		SALESMAN			
WARD		SALESMAN			
Dept ID:	**40**	**Dept:**	**OPERATIONS**	**Location:**	**BOSTON**

No detail records retrieved.

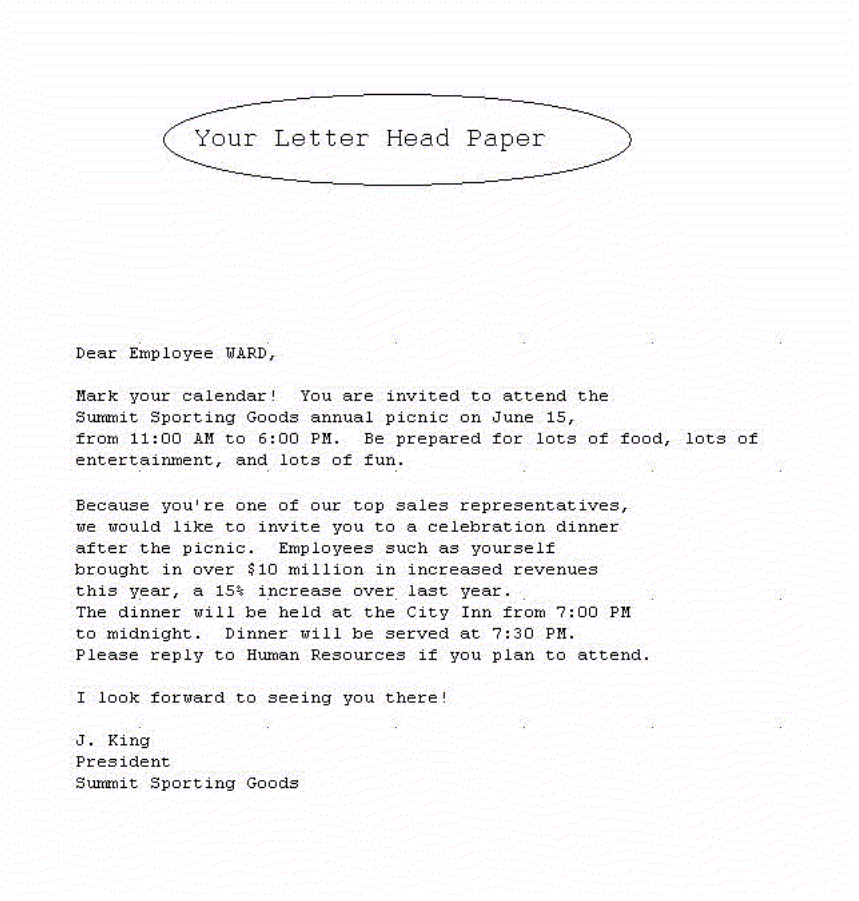
For more information on building this example, refer to Chapter 21, "Building a Report that Suppresses Labels".

4.3.8 Building a conditional form letter report

In this example, you will build two form letters from the same report, as shown in the sample output below. As you can see, the two letters share a number of features. Hence, it is more convenient to create a base form letter and then apply conditions to certain parts to determine whether they should be displayed for the current record, in this case, employees.

Figure 4–19 Final output for the first conditional form letter report example

Figure 4–20 FInal output for the second conditional form letter report example



For more information on building this example, refer to Chapter 22, "Building a Conditional Form Letter Report".

4.3.9 Building a report with conditional highlighting

In this example, you will learn how to highlight data in your report. In this report that shows employee salaries, salaries that are greater than or equal to 10,000 are displayed in bold and in red color, and values that are between 4,999 and 10,000 are displayed in bold. Using the Conditional Formatting dialog box in Reports Builder,

you can create format triggers that will change the appearance of retrieved data depending on factors you define.

Figure 4–21 Final output of the conditional highlighting report example

Employee Id	First Name	Last Name	Salary
100	Steven	Kings	$24,000.00
101	Neena	Kochhar	$17,000.00
102	Lex	De Haan	$17,000.00
103	Alexander	Hunold	$9,000.00
104	Bruce	Ernst	$6,000.00
105	David	Austin	$4,800.00
106	Valli	Pataballa	$4,800.00
107	Diana	Lorentz	$4,200.00
108	Nancy	Greenberg	$12,000.00
109	Daniel	Faviet	$9,000.00
110	John	Chen	$8,200.00
111	Ismael	Sciarra	$7,700.00
112	Jose Manuel	Urman	$7,800.00
113	Luis	Popp	$6,900.00
114	Den	Raphaely	$11,000.00
115	Alexander	Khoo	$3,100.00

For more information on building this example, refer to Chapter 23, "Building a Report with Conditional Highlighting".

4.3.10 Building a report with dynamic graphics

In this example, you will build an employee report with a bar graph that displays monthly revenues for the company. Since the data in it changes monthly, you cannot import or draw the graph.

Figure 4–22 Final output of the dynamic graphics report example

For more information on building this example, refer to Chapter 24, "Building a Report with Dynamic Graphics".

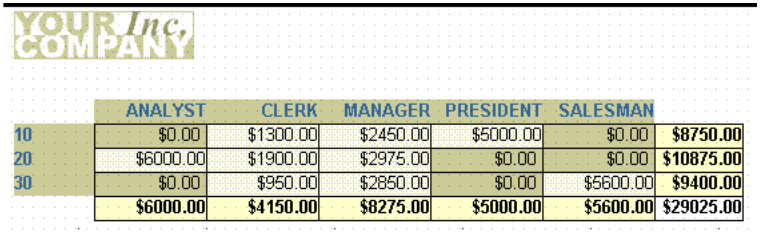
4.4 Part 4: Building Matrix Reports

4.4.1 Building a matrix report

A matrix report looks like a grid. As shown in the sample output below, it contains one row of labels, one column of labels, and information in a grid format that is related to both the row and column labels. (Matrix reports are also sometimes referred to as "crosstab" reports.)

In this example, you will build a matrix report that contains three additions to the basic matrix: summaries have been added, zeroes replace non-existent values in the cells, and the cells themselves are surrounded by grid lines. Of the summaries, one sums the salaries by department, one sums them by job, and one sums them for the whole report.

Figure 4–23 Final output of the matrix report example



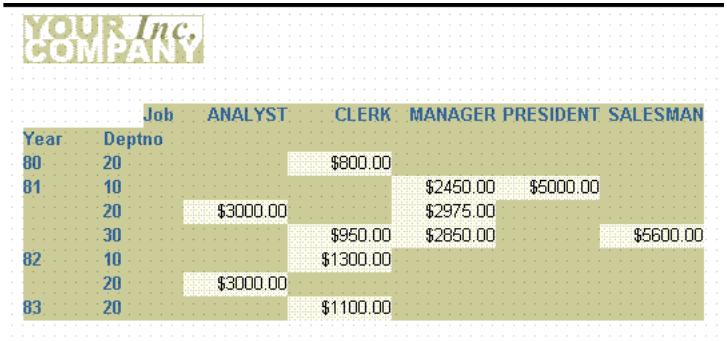
	ANALYST	CLERK	MANAGER	PRESIDENT	SALESMAN	
10	$0.00	$1300.00	$2450.00	$5000.00	$0.00	**$8750.00**
20	$6000.00	$1900.00	$2975.00	$0.00	$0.00	**$10875.00**
30	$0.00	$950.00	$2850.00	$0.00	$5600.00	**$9400.00**
	$6000.00	**$4150.00**	**$8275.00**	**$5000.00**	**$5600.00**	**$29025.00**

For more information on building this example, refer to Chapter 25, "Building a Matrix Report".

4.4.2 Building a nested matrix report

In this example, you will build the nested matrix report shown below. The cross product is capable of displaying every possible value for three dimensions: two down (YEAR and DEPTNO) and one across (JOB). This method does not include rows that have null values.

Figure 4–24 Final output of the nested matrix report example



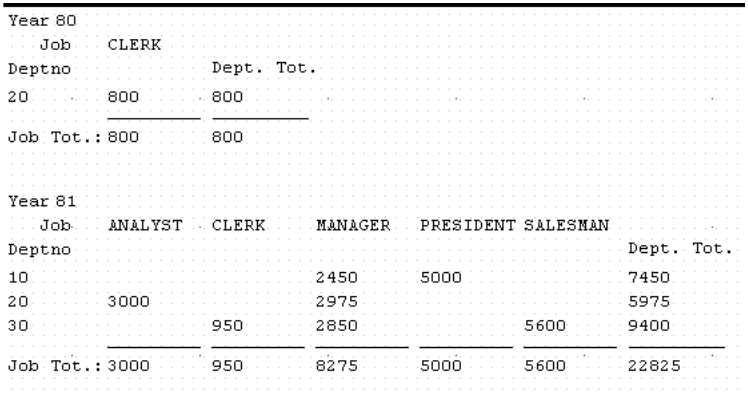
	Job	ANALYST	CLERK	MANAGER	PRESIDENT	SALESMAN
Year	Deptno					
80	20		$800.00			
81	10			$2450.00	$5000.00	
	20	$3000.00		$2975.00		
	30		$950.00	$2850.00		$5600.00
82	10		$1300.00			
	20	$3000.00				
83	20		$1100.00			

For more information on building this example, refer to Chapter 26, "Building a Nested Matrix Report".

4.4.3 Building a matrix with group above report

In this example, you will build a report that shows department, job, and salary information for each employee by the year they were hired using a matrix break format.

Figure 4–25 Final output of the matrix with group above report example



Year 80

Job	CLERK	
Deptno		Dept. Tot.
20	800	800
Job Tot.:	800	800

Year 81

Job	ANALYST	CLERK	MANAGER	PRESIDENT	SALESMAN	
Deptno						Dept. Tot.
10			2450	5000		7450
20	3000		2975			5975
30		950	2850		5600	9400
Job Tot.:	3000	950	8275	5000	5600	22825

For more information on building this example, refer to Chapter 27, "Building a Matrix with Group Above Report".

4.5 Part 5: Building Reports for Business Cases

4.5.1 Building a time series calculations report

In this example, you will build a report that calculates and displays the four-month average of purchases for each customer. You will use the Report Wizard to create a simple time series calculations report for both paper and the Web. For the JSP-based Web report, you will modify the Web source to change labels and add format masks.

Figure 4–26 Final output of the time series calculations report example

Custid	Shipdate	Total	4-Month Moving Average
100	30-JUL-86	$3.40	$3.40
	15-AUG-86	$97.50	$50.45
	01-JAN-87	$730.00	$730.00
	12-MAR-87	$4,450.00	$2,590.00
101	08-JAN-87	$101.40	$101.40
102	05-JUN-86	$224.00	$224.00
	20-JUN-86	$56.00	$140.00
	11-JAN-87	$45.00	$45.00
	05-FEB-87	$23,940.00	$11,992.50
	06-MAR-87	$3,510.50	$9,165.17
103	10-FEB-87	$764.00	$764.00
104	18-JUL-86	$5.60	$5.60
	25-JUL-86	$35.20	$20.40

For more information on building this example, refer to Chapter 28, "Building a Time Series Calculations Report".

4.5.2 Building a report with aggregate data

In this example, you will build a report that collects and displays names of all employees whose salaries fall within the range of 0 to 999, then collects and displays all employees whose salaries fall within the range of 1000 to 1999, etc. You will be able to modify this report to display any aggregate range you need.

Figure 4–27 Final output of the aggregate data report example

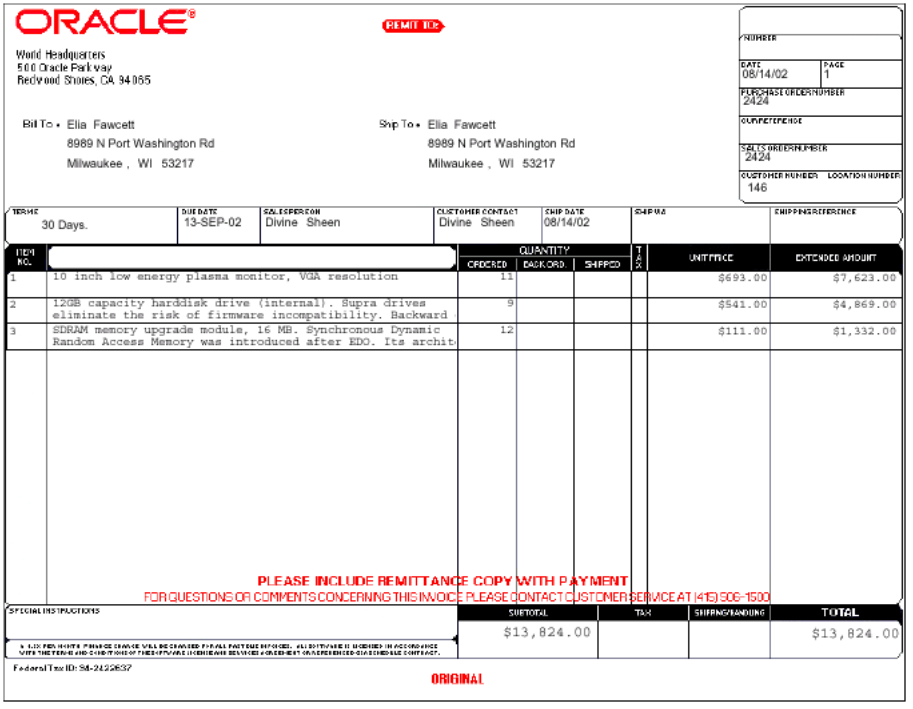
Salary Range			Name	Dept
0	-	1000	SMITH	20
			JAMES	30
1000	-	2000	ADAMS	20
			WARD	30
			MARTIN	30
			MILLER	10
			TURNER	30
			ALLEN	30
2000	-	3000	CLARK	10
			BLAKE	30
			JONES	20
3000	-	4000	SCOTT	20
			FORD	20
5000	-	6000	KING	10

For more information on building this example, refer to Chapter 29, "Building a Report with Aggregate Data".

4.5.3 Building a check printing report with spelled-out cash amounts

In this chapter, you will build a check printing report with a stub and spelled-out cash amounts. The steps described in this chapter will help you create a PL/SQL function that returns spelled-out numerical values. You will also learn how to import an image of a pre-printed form (in this case, a blank check image) and use the tools in the Paper Layout and Paper Design views to print your report on such a form. Although we use a check as the example in this report, you can use the steps to use any pre-printed form with Oracle Reports.

Figure 4–28 Final output of the check printing report with spelled-out cash amounts

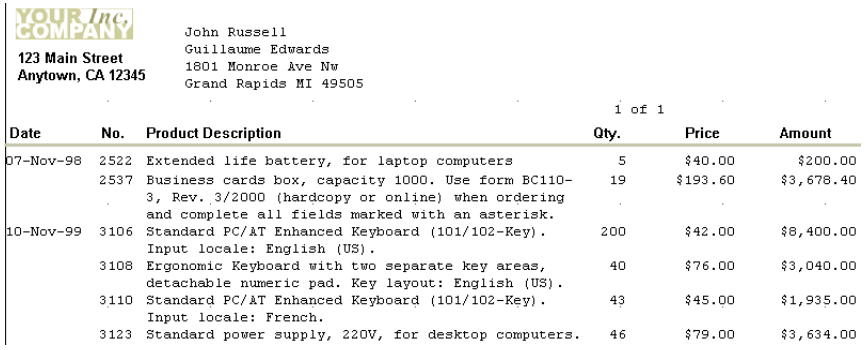


For more information on building this example, refer to Chapter 30, "Building a Check Printing Report with Spelled-Out Cash Amounts".

4.5.4 Building a report using a pre-printed form

In this chapter, you will learn formatting techniques for printing reports on pre-printed forms when you do not have access to a computer readable version of the forms. Such reports must be designed so that the data prints in exact positions on the form.

Figure 4–29 Final output of the preprinted forms example



YOUR Inc. COMPANY

123 Main Street
Anytown, CA 12345

John Russell
Guillaume Edwards
1801 Monroe Ave Nw
Grand Rapids MI 49505

1 of 1

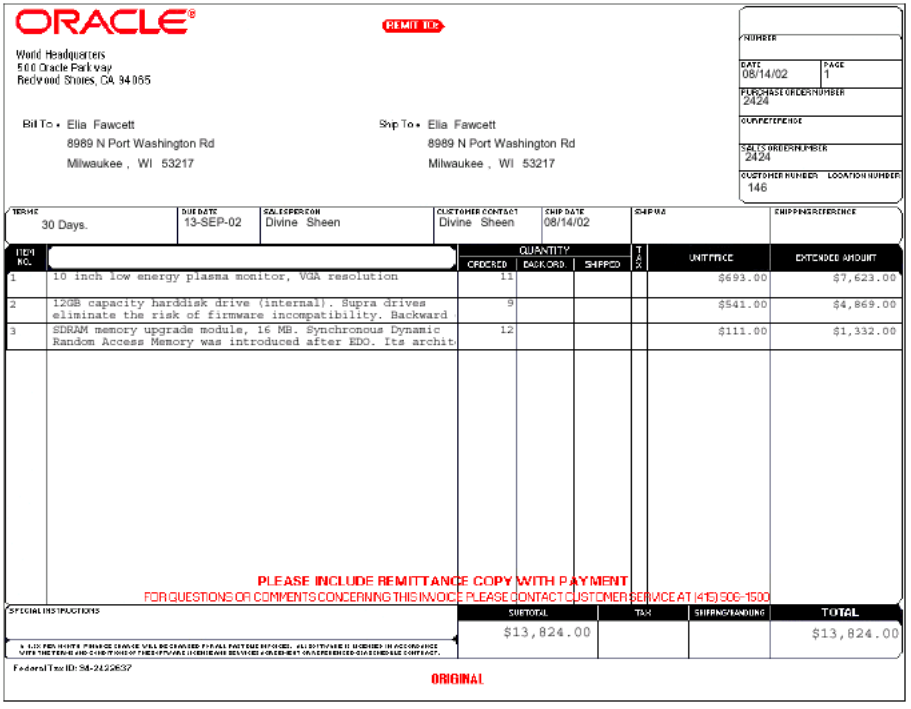
Date	No.	Product Description	Qty.	Price	Amount
07-Nov-98	2522	Extended life battery, for laptop computers	5	$40.00	$200.00
	2537	Business cards box, capacity 1000. Use form BC110-3, Rev. 3/2000 (hardcopy or online) when ordering and complete all fields marked with an asterisk.	19	$193.60	$3,678.40
10-Nov-99	3106	Standard PC/AT Enhanced Keyboard (101/102-Key). Input locale: English (US).	200	$42.00	$8,400.00
	3108	Ergonomic Keyboard with two separate key areas, detachable numeric pad. Key layout: English (US).	40	$76.00	$3,040.00
	3110	Standard PC/AT Enhanced Keyboard (101/102-Key). Input locale: French.	43	$45.00	$1,935.00
	3123	Standard power supply, 220V, for desktop computers.	46	$79.00	$3,634.00

For more information on building this example, refer to Chapter 31, "Building a Report Using a Pre-Printed Form".

4.5.5 Building an invoice report

In this example, you will build a report that displays several distinguishing characteristics of a typical invoice, such as customer name and address, sales order number, billing information, and billing totals.

Figure 4–30 Final output of the invoice report example

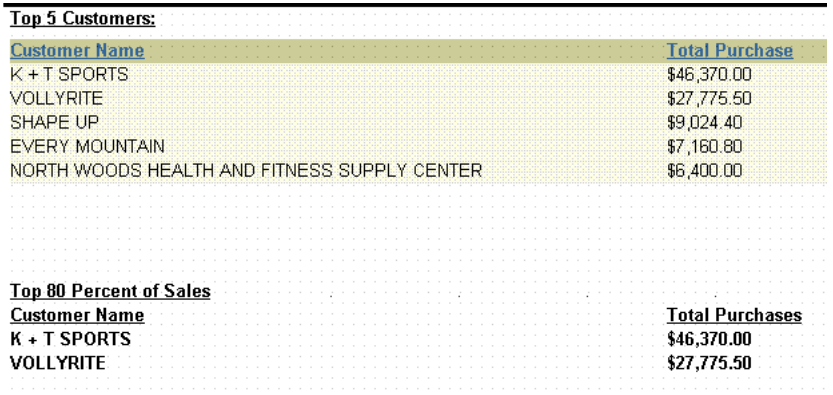


For more information on building this example, refer to Chapter 32, "Building an Invoice Report".

4.5.6 Building a ranking report

In this example, you will build a report that ranks data in two different ways: by count and by percentage. The upper portion displays the names and the total purchases of the top three customers; the lower portion displays the names and total purchases of those customers who constitute 75% of all sales. You can set the ranking criteria at runtime, or let them default to previously specified values.

Figure 4–31 Final output of the ranking report example



Top 5 Customers:

Customer Name	Total Purchase
K + T SPORTS	$46,370.00
VOLLYRITE	$27,775.50
SHAPE UP	$9,024.40
EVERY MOUNTAIN	$7,160.80
NORTH WOODS HEALTH AND FITNESS SUPPLY CENTER	$6,400.00

Top 80 Percent of Sales

Customer Name	Total Purchases
K + T SPORTS	$46,370.00
VOLLYRITE	$27,775.50

For more information on building this example, refer to Chapter 33, "Building a Ranking Report".

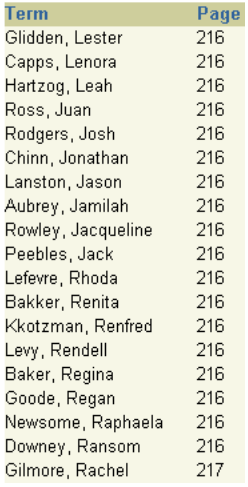
4.5.7 Building a report with a simple table of contents and index

This example is designed to teach you how to add navigational items to a large paper report. You will learn how to create a group above report, then add a simple table of contents to the beginning of your report, that will enable users to find an item by its category. You will also learn how to create an index so users can directly find a specific piece of information.

Figure 4–32 Simple table of contents

Topic	Pages
Argentina	1-7
Australia	7-26
Brazil	26-45
Denmark	45-54
France	54-147
Germany	147-345
India	345-362
Ireland	362-411
Japan	411-426
Malaysia	426-440
New Zealand	440-446
Poland	446-462

Figure 4–33 Index



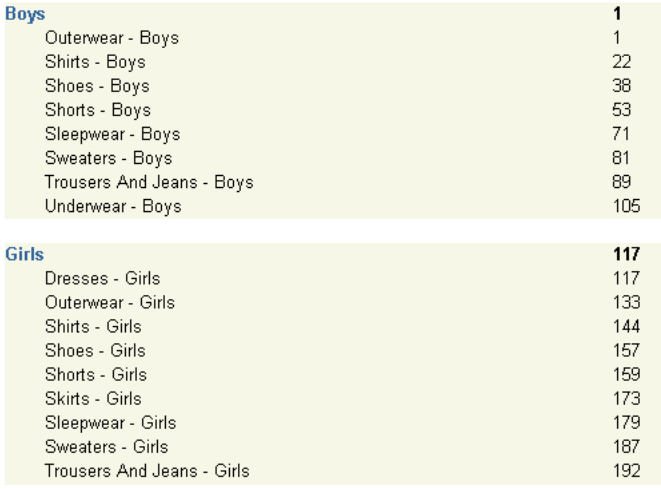
Term	Page
Glidden, Lester	216
Capps, Lenora	216
Hartzog, Leah	216
Ross, Juan	216
Rodgers, Josh	216
Chinn, Jonathan	216
Lanston, Jason	216
Aubrey, Jamilah	216
Rowley, Jacqueline	216
Peebles, Jack	216
Lefevre, Rhoda	216
Bakker, Renita	216
Kkotzman, Renfred	216
Levy, Rendell	216
Baker, Regina	216
Goode, Regan	216
Newsome, Raphaela	216
Downey, Ransom	216
Gilmore, Rachel	217

For more details on building this example, refer to Chapter 34, "Building a Paper Report with a Simple Table of Contents and Index".

4.5.8 Building a report with a multilevel table of contents

This example is designed to teach you how to add a multilevel table of contents to a large paper report. You will add a table of contents based on a category and sub-category to an existing paper report.

Figure 4–34 Multilevel table of contents



For more details on building this example, refer to Chapter 35, "Building a Paper Report with a Multilevel Table of Contents".

4.5.9 Bursting and distributing a report

In this example, you will modify a simple report we've provided to burst each section to a separate report. You will then modify a sample distribution XML file to send an e-mail to each destination with an attachment based on the separate reports. You will also send multiple e-mails to the same e-mail address with a single attachment (the entire report).

For more information on building this example, refer to Chapter 36, "Bursting and Distributing a Report".

Note: No sample output is available for this report.

4.6 Part 6: Building Reports with PL/SQL and Java

4.6.1 Building a PL/SQL report

In this example, you will learn how to use an external PL/SQL library and PL/SQL within a report to modify formatting and calculate the total compensation for each employee. In the sample output below, notice the spacing between records (e.g., between the record for Alexander Khoo and the record for Alexis Bull). This space is not due to a break; it is the result of using a PL/SQL procedure in a format trigger.

Figure 4–35 Final output of the PL/SQL report example

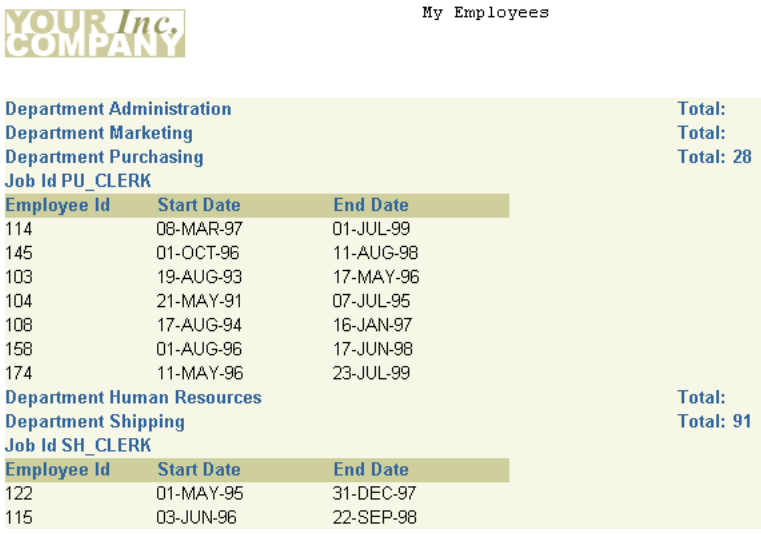
Name	Job Id	Commission Percent	Salary	Bonus	Total Compensation
Adam Fripp	ST_MAN		$8,200.0	$1,230.0	$9,430.0
Alana Walsh	SH_CLERK		$3,100.0	$465.0	$3,565.0
Alberto Errazuriz	SA_MAN	.3	$12,000.0	$1,800.0	$13,800.0
Alexander Hunold	IT_PROG		$9,000.0	$1,350.0	$10,350.0
Alexander Khoo	PU_CLERK		$3,100.0	$465.0	$3,565.0
Alexis Bull	SH_CLERK		$4,100.0	$615.0	$4,715.0
Allan McEwen	SA_REP	.35	$9,000.0	$1,350.0	$13,500.0
Alyssa Hutton	SA_REP	.25	$8,800.0	$1,320.0	$12,320.0
Amit Banda	SA_REP	.1	$6,200.0	$930.0	$7,750.0
Anthony Cabrio	SH_CLERK		$3,000.0	$450.0	$3,450.0

For more information on building this example, refer to Chapter 37, "Building a PL/SQL Report".

4.6.2 Building a paper report with ref cursors

In this example, you will learn how to use Reports Builder's features for using ref cursors. To build this paper report, you will use the Data Model view to create a multiquery data model, and then use the Report Wizard to create the report layout. You will make fairly extensive manual refinements in the Data Model view.

Figure 4–36 Final output for the ref cursors example



For more information on building this example, refer to Chapter 38, "Building a Paper Report with Ref Cursors".

4.6.3 Building a Simple Parameter Form for a JSP-based Web report

This example is designed to teach you how to build a simple JSP Parameter Form for a Web report. You will build a simple form in HTML, then create a JSP Parameter Form based on a data model in Reports Builder. You will then learn how to link the Parameter Form with an existing JSP-based Web report, then deploy and test both the JSP Parameter Form and the Web report via your browser.

Figure 4–37 JSP Parameter Form

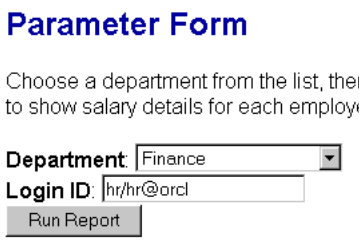
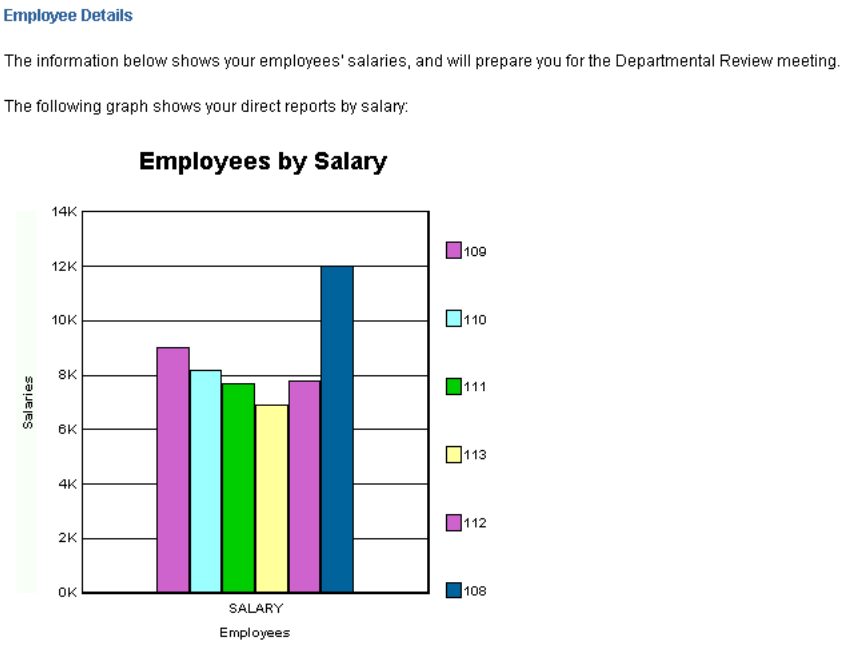


Figure 4–38 JSP-based Web report based on a user parameter



For more details on building this example, refer to Chapter 39, "Building a Simple Parameter Form for a JSP-based Web Report".

4.6.4 Building a report with a barcode

In this example, you will build a report using the barcode Java bean in Oracle Reports. You will build two reports: one for paper and one for the Web. The paper report shows an invoice for a single customer who has ordered multiple items from a company. The barcode indicates the tracking information for the order.

Figure 4–39 Final output of the barcode report with paper layout

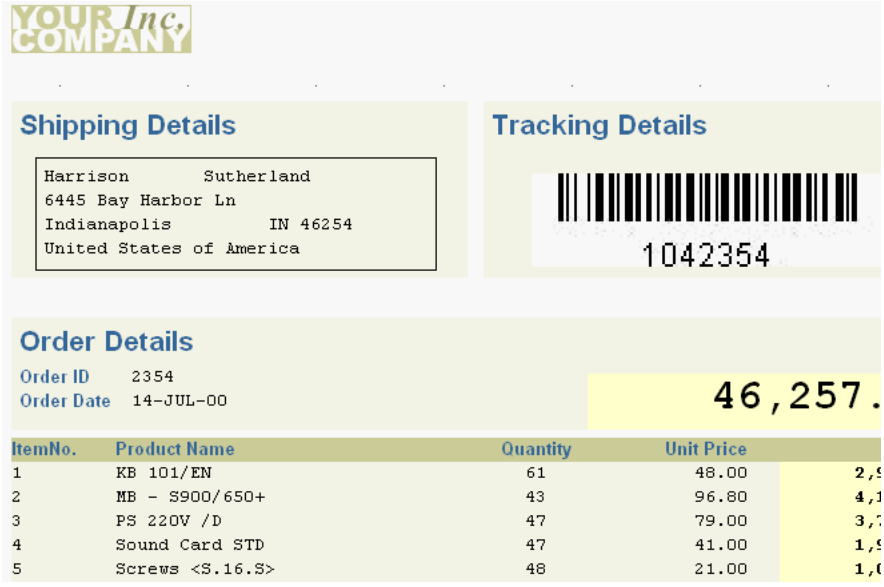


Figure 4–40 Final output of the barcode report with Web layout

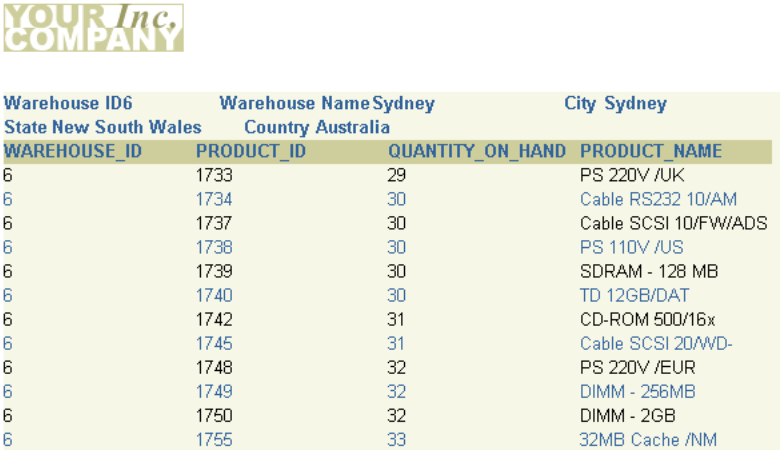
For more details on building this example, refer to Chapter 40, "Building a Report with a Barcode".

4.7 Part 7: Building Reports with Pluggable Data Sources

4.7.1 Building a report with an XML pluggable data source

Reports Builder enables you to use any data source you wish. In this example, you will learn how to use the XML pluggable data source that is provided with Oracle Reports.

Figure 4–41 Final output of the XML PDS example



For more information on building this example, refer to Chapter 41, "Building a Report with an XML Pluggable Data Source".

4.7.2 Building a report with a text pluggable data source

Reports Builder enables you to use any data source you wish. In this example, you will learn how to use character-delimited text as a data source.

Figure 4–42 Final output for the text PDS example

Category HISPANIC OR LATINO AND RACE		
Subject	**Value**	**Percentage**
Total population	33,871,648	100%
Hispanic or Latino (of any race)	10,966,556	32%
Mexican	8,455,926	25%
Puerto Rican	140,570	0%
Cuban	72,286	0%
Other Hispanic or Latino	2,297,774	7%
Not Hispanic or Latino	22,905,092	68%
White alone	15,816,790	47%

Category HOUSEHOLDS BY TYPE		
Subject	**Value**	**Percentage**
Total households	11,502,870	100%
Family households (families)	7,920,049	69%
With own children under 18 years	4,117,036	36%
Married-couple family	5,877,084	51%
With own children under 18 years	2,989,974	26%
Female householder, no husband present	1,448,510	13%
With own children under 18 years	834,716	7%

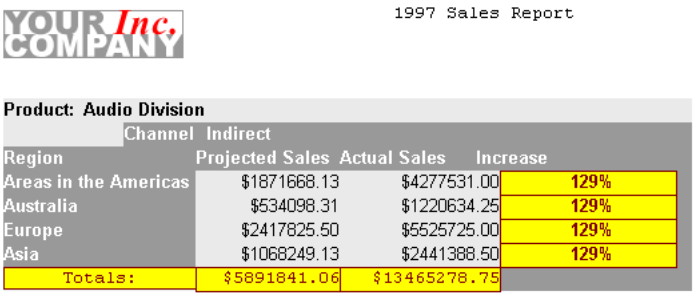
For more information on building this example, refer to Chapter 42, "Building a Report with a Text Pluggable Data Source".

4.7.3 Building a report using Oracle Express data

This example is designed to help you learn more about the Reports Builder features for Express data. You will build an Express report that summarizes the yearly projected and actual sales for each region and sales channel in a product division.

To build this report, you will use the Report Wizard to create the initial data model and report layout. You will make refinements to the data model and to the Express query. Finally, you will enhance the look of the report in the Paper Layout view and in the Paper Design view.

Figure 4–43 Final output for the Express data example



For more information on building this example, refer to Chapter 43, "Building a Report Using Oracle Express Data".

4.8 Summary

In this chapter, each of the example reports contained in this manual are described. For more details on each of the examples, refer the location provided. For access to the latest versions of the documentation and example files, visit the Oracle Reports area on the Oracle Technology Network (`http://otn.oracle.com/products/reports/`), then navigate to the Getting Started with Oracle Reports Web site.

Part I

Building Basic Reports

The chapters in this Part provide steps to build simple reports.

This part contains the following chapters:

- Chapter 5, "Building a Tabular Report"

  A tabular report is the most basic type of report. The output of this report is organized in a multicolumn, multirow format (with rows and columns corresponding to those in the database table).

- Chapter 6, "Building a Mailing Label Report"

  A mailing label report is displayed in a format suitable for use as address labels on envelopes. You write custom text and embed database values, or text from files, to create the labels. You can print the labels in one or many columns and at any position.

- Chapter 7, "Building a Form Letter Report"

  A form letter report is displayed in a format suitable for use in printed forms or letters. Similar to the mailing label report, you can write custom text and embed to it, database values or text from files.

- Chapter 8, "Building a Master/Master Report"

  A master-master report (also called a parent-parent report) consists of at least two sets of data that are not directly related. This report contains two or more queries with no links (i.e., no parent-child relationships).

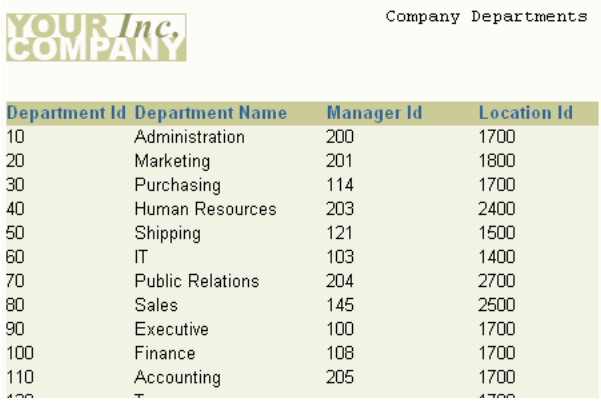
- Chapter 9, "Building a Summary Report"

  A summary report contains a field that summarizes at least one column of the report. This summary can be: total sales, average of a list of commissions, maximum amount found in a series of purchase orders, etc.

5

Building a Tabular Report

Figure 5–1 Tabular report output



Reports Builder enables you to build many types of reports, such as tabular, form letter, and group above reports. You can learn more about these different types of reports in the *Reports Builder online help*. This chapter describes the basic elements of a tabular report and how to build a simple tabular report.

About tabular reports

A tabular report is the most basic type of report you can build. The report output is organized in a multicolumn, multirow format, with each column corresponding to a column in the database table. In this example, the column labels "Deptno," "Dname," and "Loc," are derived from the columns in your SQL SELECT statement, but you can modify column labels as you wish.

Concepts

When you start Oracle Reports, it automatically creates a new report definition, which consists of default settings stored in a report buffer. You create the report you want by customizing this report definition.

Data Relationships

- Generally, the first step to define the data Oracle Reports should fetch from the database by creating a query that contains a SELECT statement. Although the report in this section uses only one query, reports can contain any number of queries.
- For each query you create, Oracle Reports automatically creates one default group. The default name of the group is the query name with "G_" prefixed to it. As shown by the data model in the next figure, we've named our query "Q_ Department." Therefore, the group created by Oracle Reports for this report is "G_Department." If your query was named "Accounts," the default group name would be "G_Accounts."
- For each database column you specify in the query's SELECT statement, Oracle Reports creates a report column and assigns it to the group. By default, the columns appear in the order in which you enter them in your SELECT statement. The default column names are generated from the database column names or replaced by any aliases specified in your SELECT statement.
- If you select two or more columns with the same name, the first column is given the default name, the second column is given the default name with a "1" appended to it (e.g., "Deptno," "Deptno1"), etc.

Layout

- Before you can run your report, you'll need to specify the layout (i.e., the format) of the report output. If you don't, Oracle Reports will fetch the data, but will not display it; you may receive an error message, depending on whether you choose paper or Web layout.
- For this report, you will use the tabular layout style, which is one of eight styles provided by Oracle Reports. The tabular layout displays data in a series of columns running down the page, with the column headings displayed directly above the columns.

Example Scenario

In this example, you will use the Report Wizard to build one query to select all of the columns displayed in this report. Oracle Reports will create all other necessary objects (e.g., groups and columns) by default.

To see a sample tabular report, open the examples folder named `tabular`, then open the Oracle Reports example named `tabular.rdf`. For details on how to open it, see "Accessing the example reports" in the Preface.

Table 5–1 Features demonstrated in this example

Feature	Location
Use the Report Wizard to create a tabular report with a paper layout.	Section 5.2, "Use the Report Wizard to create a report"

5.1 Prerequisites for this example

To build the example in this chapter, you must have access to the sample schema provided with the Oracle9*i* database. If you don't know if you have access to this sample schema, contact your database administrator. You should have access to the "Human Resources" portion of the schema to complete this example.

5.2 Use the Report Wizard to create a report

When you create a report, you can either use the Report Wizard to assist you or create the report yourself. To build the simple report in this example, you can use the Report Wizard. Using the wizard enables you to define the layout for the report, as well as set the data definition.

To create a simple report:

1. Launch Reports Builder (or, if already open, choose **File > New > Report**)
2. In the Welcome or New Report dialog box, select **Use the Report Wizard**, then click **OK**.
3. If the Welcome page displays, click **Next**.
4. On the Report Type page, select **Create Paper Layout Only**, then click **Next**.
5. On the Style page, type a **Title** for your report, select **Tabular**, then click **Next**.
6. On the Data Source page, click **SQL Query**, then click **Next**.

7. On the Data page, enter the following SELECT statement in the **Data Source definition** field:

```
SELECT ALL DEPARTMENTS.DEPARTMENT_ID, DEPARTMENTS.DEPARTMENT_NAME,
DEPARTMENTS.MANAGER_ID, DEPARTMENTS.LOCATION_ID
FROM DEPARTMENTS
ORDER BY DEPARTMENTS.DEPARTMENT_ID
```

Tip: This query selects all the department IDs, the department names, the manager IDs, and the location IDs, then sorts the data by the department IDs.

Note: You can enter this query in any of the following ways:

- Copy and paste the code from the provided text file called `tabular_code.txt` into the **Data Source definition** field.
- Click **Query Builder** to build the query without entering any code manually.
- Type the code in the **Data Source definition** field.

---

8. Click **Next**.

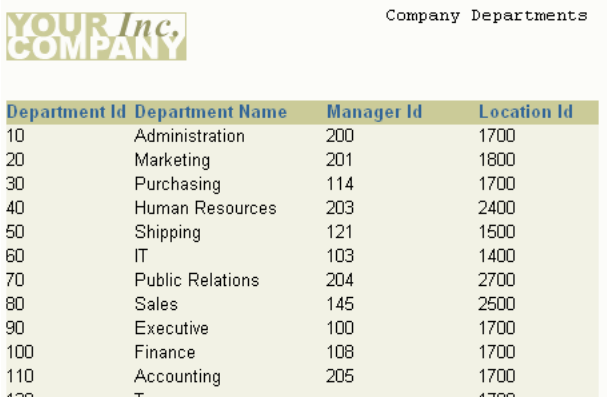
Note: If you are not already connected to a database, you will be prompted to connect to the database when you click **Next**. Ensure that you connect to a database that has the appropriate schema for this example. Section 5.1, "Prerequisites for this example" describes the sample schema requirements for this example.

---

9. On the Fields page, click the double right arrows (**>>**) to move all of the fields to the **Displayed Fields** list, then click **Next**.
10. On the Totals page, click **Next**.
11. On the Labels page, click **Next**.

12. On the Template page, select **Predefined Template** and click **Beige**, then click **Finish** to display your report output in the Paper Design view. It should look something like this:

Figure 5–2 Paper Design view for the tabular report



Note: In the Paper Design view, you can see how the tabular report displays the data like a table, in order of department ID number.

13. Save the report as `tabularreport_<your initials>.rdf`.

5.3 Summary

Congratulations! You have successfully created a tabular paper report. You now know how to:

- define a tabular report layout using the Report Wizard.

For more information on any of the wizards, views, or properties used in this example, refer to the *Reports Builder online help*, which you can access in two ways:

- From the Oracle Technology Network (`http://otn.oracle.com/products/reports/`), click **Documentation**

and navigate to the *Reports Builder Online Help* for the most recent, hosted online help.

- From Reports Builder, choose **Help > Help Contents**.

> **Note:** You can replace your Reports Builder online help with the most recent update by downloading the latest online help set available on the Oracle Technology Network (http://otn.oracle.com/docs/products/reports/). Instructions for replacing your help file are included in the readme.txt in the download file.

6

Building a Mailing Label Report

Figure 6–1 Mailing label report output

Reports Builder provides many different layouts for your reports to suit your needs. In this example, you will learn how to set up a mailing label report so you can print out addresses from your database and print them out as mailing labels.

About mailing label reports

A mailing label report consists of data displayed in a format suitable for use as address labels on envelopes. The labels can be printed in one or many columns, and can begin at any position.

Concepts

Mailing labels can be created using simple, one-query reports with a special layout style.

Data Relationships

- To fetch the data for a mailing label report, all you need to do is create a query to select it.

Layout

- Oracle Reports provides a default mailing label layout in which the fields are positioned so that each field is directly below the preceding field. Field labels are not printed.
- You will use the Report Wizard to create a simple mailing label report. The Report Wizard provides you with a formatting page, where you can choose how each mailing label will look.
- You'll use the Vertical Spacing field, located in the Property Inspector for the repeating frame, to control the amount of blank space between each mailing label. The default spacing between repeating frames may not be sufficient, and you may want to specify that space be inserted between mailing labels in the layout so that they are correctly positioned when printing them onto labels.

Example Scenario

In this example, you will use the Report Wizard to define a mailing label layout for your report, and use the Query Builder to write a single query that selects all of the columns for this report. You don't need to create any other data objects; Oracle Reports will create all other necessary data objects by default. You will then create another mailing label report, also using a single query, this time using format triggers to suppress blank lines that may appear in the labels.

To see a sample mailing label report, open the examples folder named `mailinglabel`, then open the Oracle Reports example report named `mailinglabel.rdf`. For details on how to open it, see "Accessing the example reports" in the Preface.

Table 6–1 Features demonstrated in this example

Feature	Location
Use the Report Wizard to create a introductory mailing label report with a paper layout.	Section 6.2, "Use the Report Wizard to create a mailing label report"

6.1 Prerequisites for this example

To build the example in this chapter, you must have access to the sample schema provided with the Oracle9*i* database. If you don't know if you have access to this sample schema, contact your database administrator. You should have access to the "Human Resources" portion of the schema to complete this example.

6.2 Use the Report Wizard to create a mailing label report

When you create a report, you can either use the Report Wizard to assist you or create the report yourself. To build the simple report in this example, you can use the Report Wizard. Using the wizard enables you to define the layout for the report, as well as set the data definition.

To create a simple mailing label report:

1. Launch Reports Builder (or, if already open, choose **File > New > Report**)
2. In the Welcome or New Report dialog box, select **Use the Report Wizard**, then click **OK**.
3. If the Welcome page displays, click **Next**.
4. On the Report Type page, select **Create Paper Layout Only**, then click **Next**.
5. On the Style page, type a **Title** for your report, select **Mailing Label**, then click **Next**.
6. On the Data Source page, click **SQL Query**, then click **Next**.
7. On the Data page, click the **Query Builder**.
8. In the Select Data Tables dialog box, click the **EMPLOYEES** table, then click **Include**.
9. Click the **DEPARTMENTS** table, then click **Include**.

   Note: In this case, you must include the DEPARTMENTS table since the EMPLOYEES and LOCATIONS tables are not directly related in the schema.

10. Click the **LOCATIONS** table, then click **Include**.
11. Click **Close**.

    The three tables display in the Query Builder.
12. In the EMPLOYEES table, select the check boxes next to the following column names:
    - **FIRST NAME**
    - **LAST NAME**

13. In the LOCATIONS table, select the check boxes next to the following column names:
    - **STREET ADDRESS**
    - **POSTAL CODE**
    - **CITY**
    - **STATE PROVINCE**
14. Click **OK**.
15. In the **Data Source definition** field, your query should look something like this:

```
SELECT ALL EMPLOYEES.FIRST_NAME, EMPLOYEES.LAST_NAME,
LOCATIONS.STREET_ADDRESS, LOCATIONS.POSTAL_CODE, LOCATIONS.CITY,
LOCATIONS.STATE_PROVINCE,
EMPLOYEES.EMPLOYEE_ID, LOCATIONS.LOCATION_ID
FROM DEPARTMENTS, EMPLOYEES, LOCATIONS
WHERE ((DEPARTMENTS.MANAGER_ID = EMPLOYEES.EMPLOYEE_ID)
 AND (EMPLOYEES.DEPARTMENT_ID = DEPARTMENTS.DEPARTMENT_ID)
 AND (DEPARTMENTS.LOCATION_ID = LOCATIONS.LOCATION_ID))
ORDER BY EMPLOYEES.LAST_NAME
```

> **Note:** You can enter this query in any of the following ways:
>
> - Copy and paste the code from the provided text file called mailinglabel_code.txt into the Data Source definition field.
> - Click Query Builder to build the query without entering any code manually, as described in the steps above.
> - Type the code in the Data Source definition field.

16. Click **Next**.

> **Note:** If you are not already connected to a database, you will be prompted to connect to the database when you click **Next**. Ensure that you connect to a database that has the appropriate schema for this example. Section 6.1, "Prerequisites for this example" describes the sample schema requirements for this example.

17. On the Text page, format the way you want the mailing labels to display. Steps 21 through 30 will show you how to display your labels in the following format:

    ```
    John Smith
    55 Main Street
    Springfield, ME 00000
    ```

18. In the **Available Fields** list, click **FIRST_NAME**, then click the right arrow (>) to move this field to the **Mailing Label** list.
19. In the **Available Fields** list, click **LAST_NAME**, then click the right arrow (>).
20. Click **New Line**.
21. In the **Available Fields** list, click **STREET_ADDRESS**, then click the right arrow (>).
22. Click **New Line**.
23. In the **Available Fields** list, click **CITY**, then click the right arrow (>).
24. Press **Backspace** on your keyboard to remove the extra space.
25. Click **Comma**.
26. In the **Available Fields** list, click **STATE_PROVINCE**, then click the right arrow (>).
27. In the **Available Fields** list, click **POSTAL_CODE**, then click the right arrow (>).

    The code in the Mailing Label Text box should look like this:

    ```
    &<FIRST_NAME>  &<LAST_NAME>
    &<STREET_ADDRESS>
    &<CITY>,  &<STATE> &<POSTAL_CODE>
    ```

    Tip: For more information on formatting your mailing labels, click **Help** on this page of the Report Wizard.

28. Click **Next**.
29. On the Template page, select **No Template** and click **Finish** to display your report output in the Paper Design view. It should look something like this:

Figure 6–2 Paper Design view for the mailing label report

30. Save your report as `mailinglabel_<your initials>.rdf`.

6.3 Add Vertical Spacing

In this section, you will use the Vertical Spacing property to add space between each record in your mailing label report. You can adjust this spacing according to the size of the mailing labels where the records will be printed.

To add vertical spacing:

1. In the Object Navigator, under your report name, expand the Paper Layout node.
2. Under Paper Layout, expand the Main Section node.
3. Under Body, find the repeating frame called R_G_POSTAL_CODE.
4. Press F4 to open the Property Inspector for this repeating frame.
5. Under Repeating Frame, next to **Vert. Spacing Between Frames**, type `0.25`.
6. Press Enter to add your changes.
7. In the toolbar, click Run Paper Layout to display your report in the Paper Design view.

Your report should look something like this:

Figure 6–3 Final mailing label report with vertical spacing

6.4 Summary

Congratulations! You have successfully created a mailing label paper report. You now know how to:

- define a mailing label report using the Report Wizard.
- adjust the vertical spacing between labels.

For more information on any of the wizards, views, or properties used in this example, refer to the *Reports Builder online help*, which you can access in two ways:

- From the Oracle Technology Network (`http://otn.oracle.com/products/reports/`), click **Documentation** and navigate to the *Reports Builder Online Help* for the most recent, hosted online help.
- From Reports Builder, choose **Help > Help Contents**.

> **Note:** You can replace your Reports Builder online help with the most recent update by downloading the latest online help set available on the Oracle Technology Network (http://otn.oracle.com/docs/products/reports/). Instructions for replacing your help file are included in the readme.txt in the download file.

7

Building a Form Letter Report

Figure 7–1 Form letter report output

Reports Builder enables you to build many types of reports, such as form letter, mailing label, tabular, and group above reports. You can learn more about these different types of reports by referring to the *Reports Builder online help*. This chapter describes the basic elements of a form letter report and how to build a simple form letter report.

About form letter reports

Form letter reports contain database values embedded in boilerplate text (boilerplate text can be defined as any text that appears each time the report is run). It can be text generated by Oracle Reports, text you create, or text you import from a file.

Concepts

A form letter report is similar to a mailing label report, in that it is a simple report with a special layout style.

Data Relationships

- There are no special restrictions on data relationships for a form letter report.

Layout

- A default form letter layout style is provided by Oracle Reports. It consists of the field names for the columns you've selected, prefixed by ampersands (&s). The ampersands indicate that they are hidden fields, and are contained within boilerplate and repeating frame objects. You then add the text of the letter to the layout. Oracle Reports prints one record (i.e., one letter) per page.
- Hidden fields, which are the default for a form letter layout, appear in the layout but not the output until referenced. You can do this in the Paper Layout view by typing its name, prefixed by an ampersand. Oracle Reports treats the reference as a normal field.
- In general, if you are embedding a field within boilerplate text (as in a form letter), it is best to hide the field and reference it where desired. The field values will then flow with the text. A field can appear in a report both where placed by default and where referenced, and can be referenced even when not initially hidden. Fields can also be referenced more than once in the same piece of boilerplate text.

Example Scenario

In this example, you will use the Report Wizard to build one query to select all of the columns displayed in this report. Oracle Reports will create all other necessary objects, e.g., groups and columns, by default.

To see a sample form letter report, open the examples folder called `formletter`, then open the Oracle Reports example report called `formletter.rdf`. For details on how to open it, see "Accessing the example reports" in the Preface.

Table 7–1 Features demonstrated in this example

Feature	Location
Use the Report Wizard to create a form letter report with a paper layout.	Section 7.2, "Use the Report Wizard to create a form letter report"

7.1 Prerequisites for this example

To build the example in this chapter, you must have access to the sample schema provided with the Oracle9*i* database. If you don't know if you have access to this

sample schema, contact your database administrator. You should have access to the "Human Resources" portion of the schema to complete this example.

7.2 Use the Report Wizard to create a form letter report

When you create a report, you can either use the Report Wizard to assist you or create the report yourself. To build the simple report in this example, you can use the Report Wizard. Using the wizard enables you to define the layout for the report, as well as set the data definition.

In the Report Wizard, on the Text page, you will be able to set up your form letter report exactly the way you want it to appear. On this page of the wizard, you can set up your boilerplate text (e.g., the body of the letter), and use the fields from your data tables to fill in the variable data (e.g., the addressee's name).

To create a simple report:

1. Launch Reports Builder (or, if already open, choose **File > New > Report**)
2. In the Welcome or New Report dialog box, select **Use the Report Wizard**, then click **OK**.
3. If the Welcome page displays, click **Next**.
4. On the Report Type page, select **Create Paper Layout Only**, then click **Next**.
5. On the Style page, type a **Title** for your report, select **Form Letter**, then click **Next**.
6. On the Data Source page, click **SQL Query**, then click **Next**.
7. On the Data page, enter the following SELECT statement in the **Data Source definition** field:

```
SELECT ALL EMPLOYEES.LAST_NAME, EMPLOYEES.FIRST_NAME, JOBS.JOB_ID,
EMPLOYEES.EMPLOYEE_ID, JOBS.JOB_TITLE
FROM EMPLOYEES, JOBS
WHERE (EMPLOYEES.JOB_ID = JOBS.JOB_ID)
ORDER BY EMPLOYEES.EMPLOYEE_ID
```

Note: You can enter this query in any of the following ways:

- Copy and paste the code from the provided text file called `formletter_code.txt` into the **Data Source definition** field.
- Click **Query Builder** to build the query without entering any code manually.
- Type the code in the **Data Source definition** field.

8. Click **Next**.

Note: If you are not already connected to a database, you will be prompted to connect to the database when you click **Next**. Ensure that you connect to a database that has the appropriate schema for this example. Section 7.1, "Prerequisites for this example" describes the sample schema requirements for this example.

9. On the Text page, format the letter the way you want it to appear. The steps that follow will show you how to make your form letter report look like this:

   Employee: &<FIRST_NAME> &<LAST_NAME>

   Emp. #: &<EMPLOYEE_ID>

   Dear &<FIRST_NAME> &<LAST_NAME>:

   The Human Resources department is updating its records of the company's employees. Currently, our records show your employee number as &<EMPLOYEE_ID>, and that you hold the position of &<JOB_TITLE>. If any of this information is incorrect, please contact the Human Resources department.

   Thank you,

   Human Resources

10. In the Form Letter Text box, type `Employee:`.

11. Click **Space** four times to enter four spaces.
12. In the **Available Fields** list, click **FIRST_NAME**, then click the right arrow (**>**) to move this field to the **Form Letter Text** field.
13. Click **Space**.
14. In the **Available Fields** list, click **LAST_NAME**, then click the right arrow (**>**).
15. Click **New Line**.
16. In the **Form Letter Text** field, type `Emp. #:`.
17. In the **Available Fields** list, click **EMPLOYEE_ID**, then click the right arrow (**>**).
18. Click **New Line** twice.
19. In the **Form Letter Text** field, type `Dear`.
20. Click **Space**.
21. In the **Available Fields** list, click **FIRST_NAME**, then click the right arrow (**>**).
22. Click **Space**.
23. In the **Available Fields** list, click **LAST_NAME**, then click the right arrow (**>**).
24. In the **Form Letter Text** field, type a colon (:) next to LAST_NAME, then click **New Line** twice.
25. Type the body of the letter. For the field names, use the **Available Fields** list to select the appropriate name, then click the right arrow (**>**) to insert it into the **Form Letter Text** field. The result should look like this:

    The Human Resources department is updating its records of the company's employees. Currently, our records show your employee number as &<EMPLOYEE_ID>, and that you hold the position of &<JOB_TITLE>. If any of this information is incorrect, please contact the Human Resources department.

    Thank you,

    Human Resources

26. Click **Next**.
27. On the Template page, select **Predefined Template** and click **Beige**, then click **Finish** to display your report output in the Paper Design view. It should look something like this:

Figure 7–2 Paper Design view for the form letter report

28. Save the report as `formletterreport_<your initials>.rdf`.

7.3 Summary

Congratulations! You have successfully created a form letter paper report. You now know how to:

- define a form letter report layout using the Report Wizard.

For more information on any of the wizards, views, or properties used in this example, refer to the *Reports Builder online help*, which you can access in two ways:

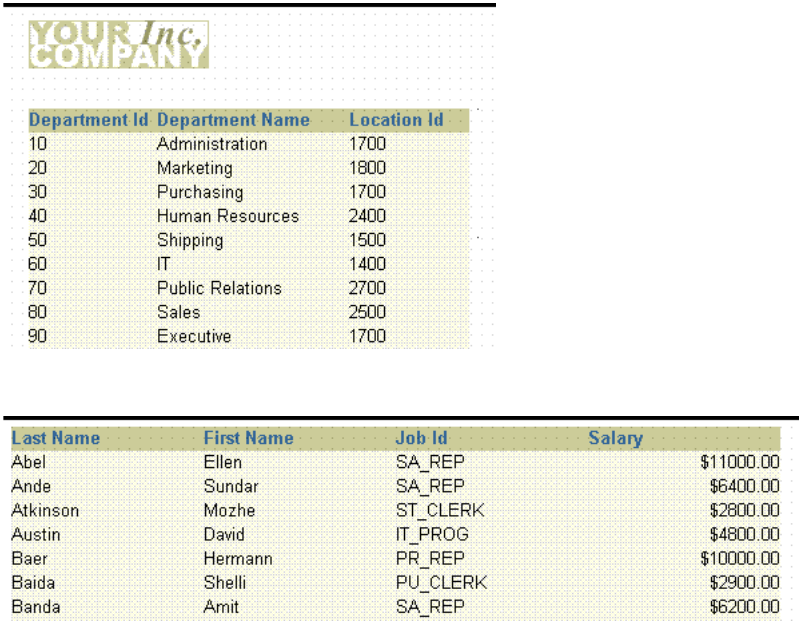
- From the Oracle Technology Network (`http://otn.oracle.com/products/reports/`), click **Documentation** and navigate to the *Reports Builder Online Help* for the most recent, hosted online help.
- From Reports Builder, choose **Help > Help Contents**.

> **Note:** You can replace your Reports Builder online help with the most recent update by downloading the latest online help set available on the Oracle Technology Network (http://otn.oracle.com/docs/products/reports/). Instructions for replacing your help file are included in the readme.txt in the download file.

8

Building a Master/Master Report

Figure 8–1 Master/Master report output



Department Id	Department Name	Location Id
10	Administration	1700
20	Marketing	1800
30	Purchasing	1700
40	Human Resources	2400
50	Shipping	1500
60	IT	1400
70	Public Relations	2700
80	Sales	2500
90	Executive	1700

Last Name	First Name	Job Id	Salary
Abel	Ellen	SA_REP	$11000.00
Ande	Sundar	SA_REP	$6400.00
Atkinson	Mozhe	ST_CLERK	$2800.00
Austin	David	IT_PROG	$4800.00
Baer	Hermann	PR_REP	$10000.00
Baida	Shelli	PU_CLERK	$2900.00
Banda	Amit	SA_REP	$6200.00

A master/master report displays at least two sets of data which are not directly related—that is, the records constituting the data are fetched using at least two separate queries. A master/master report (also called a parent/parent report) contains two or more queries with no links (parent/child relationships).

Concepts

- Although the data may be conceptually related, the queries make no attempt to relate it by the manner in which it is fetched. To select the data for a master/master report, create two or more unrelated queries. Reports Builder will create all other necessary data objects by default.
- This report uses the default tabular layout style. You'll move the layout objects associated with the second query to insert more space between the information fetched by the two queries and make your report easier to read.
- For this report you'll create a format mask and change some of the default column labels to clarify their meanings.

Example Scenario

Suppose that you want to create a master layout that lists department names, numbers, and locations followed by another master layout that lists all of your employees with their job, titles, and salaries.

To see a sample master/master report, open the examples folder named `masterb`, then open the Oracle Reports example named `masterb.rdf`. For details on how to access it, see "Accessing the example reports" in the Preface.

Table 8–1 Features demonstrated in this example

Feature	Location
Create a new, empty report.	Section 8.2, "Create a new report manually"
Create queries	Section 8.3, "Use the Data Wizard to create two queries"
Layout the data	Section 8.4, "Use the Report Wizard to layout the data"
Add white space	Section 8.5, "Use the Paper Layout view to add white space"
Format monetary values	Section 8.6, "Format a field"

8.1 Prerequisites for this example

To build the example in this chapter, you must have access to the sample schema provided with the Oracle9*i* database. If you don't know if you have access to this sample schema, contact your database administrator. You should have access to the "Human Resources" portion of the schema to complete this example.

8.2 Create a new report manually

In this case, it is easier to create the data model and layout separately. Hence, we will create an empty report first, then add the queries, and then create the layouts.

To create a blank report:

1. Launch Reports Builder (or, if already open, choose **File > New > Report**)
2. In the Welcome or New Report dialog box, select **Build a new report manually**, then click **OK**.

8.3 Use the Data Wizard to create two queries

When you create a report with multiple queries, it is typically easier to create all of the queries with the Data Wizard first and then create the layouts with the Reports Wizard.

To create the queries:

1. In the Data Model view, choose **Insert > Query** to display the Data Wizard.
2. If the Welcome page displays, click **Next**.
3. On the Query page, type `Q_Dept` for the Query name and click **Next**.
4. On the Data Source page, click **SQL Query**, then click **Next**.
5. On the Data page, in the **Data Source definition** field, enter the following SELECT statement:

```
SELECT DEPARTMENT_ID, DEPARTMENT_NAME, LOCATION_ID
FROM DEPARTMENTS
```

Note: You can enter this query in any of the following ways:

- Copy and paste the code from the provided text file called `masterb_code.txt` into the **Data Source definition** field.
- Click **Query Builder** to build the query without entering any code manually.
- Type the code in the **Data Source definition** field.

---

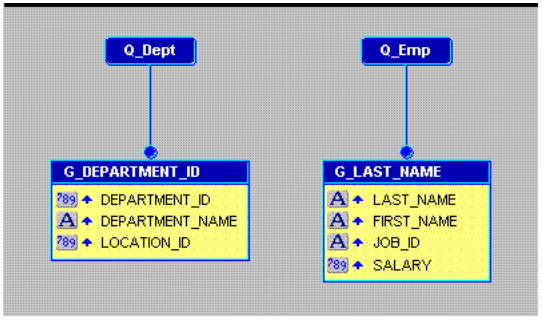
6. Click **Next**.

> **Note:** If you are not already connected to a database, you will be prompted to connect to the database when you click **Next**. Ensure that you connect to a database that has the appropriate schema for this example. Section 8.1, "Prerequisites for this example" describes the sample schema requirements for this example.

7. On the Groups page, click **Next**.
8. Click **Finish** to display the data model for your report in the Data Model view.
9. Repeat the steps above for a second query, but this time name your query `Q_Emp` and use the following SELECT statement:

```
SELECT LAST_NAME, FIRST_NAME, JOB_ID, SALARY
FROM EMPLOYEES
ORDER BY LAST_NAME, FIRST_NAME
```

Figure 8–2 Data model with two unrelated queries



8.4 Use the Report Wizard to layout the data

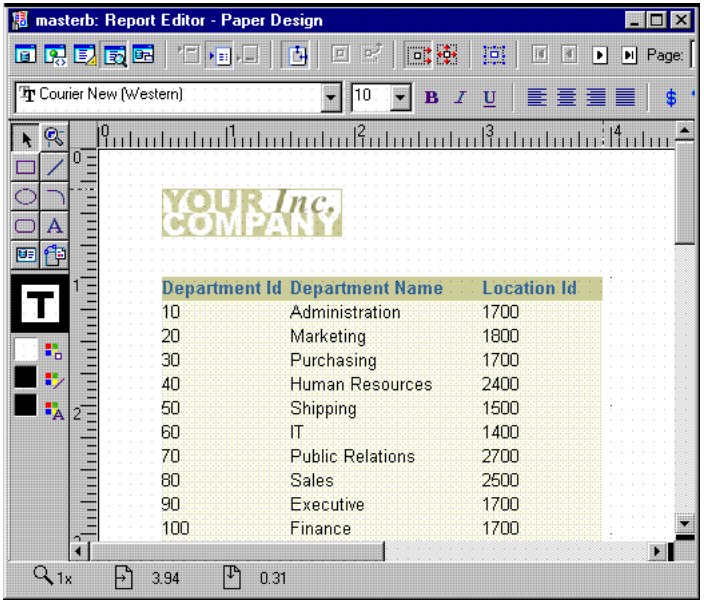
Once your data model is complete, you need to create a layout for the data objects to display in the report output. The Report Wizard enables you to create layouts for your data model.

Tip: When you have multiple queries in your data model, make sure that you check the names of the groups associated with each query prior to entering the Report Wizard. The Report Wizard requires you to choose data for the layout by group name.

To create the layout:

1. In the Data Model view, right-click on the canvas, then choose **Report Wizard**.
2. In the Report Wizard, on the **Report Type** page, select **Create Paper Layout only**.
3. On the **Style** page, select **Tabular**.
4. On the **Groups** page, ensure that both groups from your data model appear in the **Displayed Groups** list.
5. On the **Fields** page, click the double right arrows (**>>**) to move all of the fields to the **Displayed Fields** list.
6. On the **Template** page, select **Predefined Template** and click **Beige**, then click **Finish** to display your report output in the Paper Design view. It should look something like this:

Figure 8–3 Paper Design view for the Master/Master report



8.5 Use the Paper Layout view to add white space

If you scroll down in the Paper Design view, you will notice that the field labels from the second query, Q_Emp, are located directly below the fields from the first query, Q_Dept. In fact, they are almost touching each other. For this report, you'll move all of the layout objects belonging to Q_Emp down the page to create some white space between the two layouts.

To create white space between the layouts:

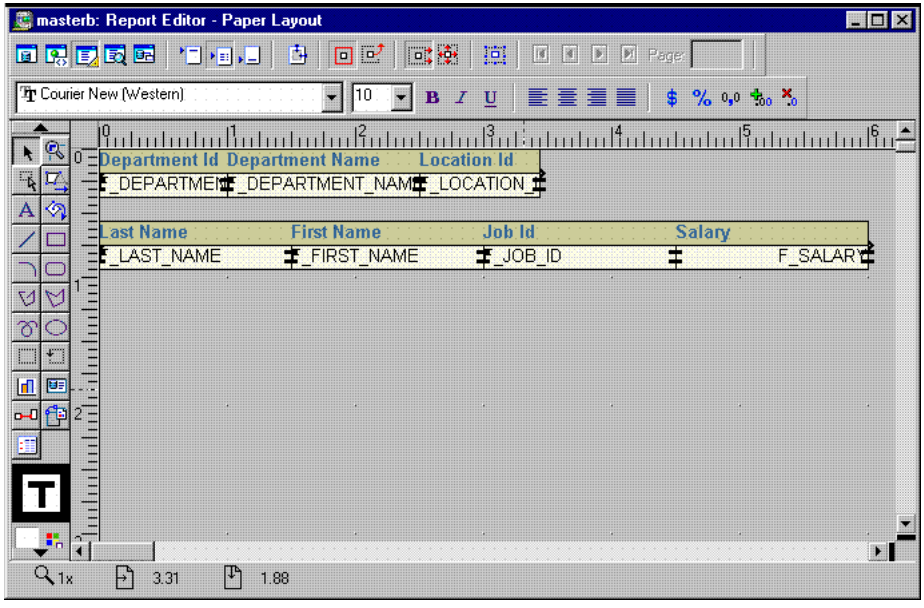
1. Move and resize the Report Editor window such that you can easily view it and the Object Navigator simultaneously.
2. Click the Paper Layout button in the toolbar of the Report Editor window to display the Paper Layout view.
3. In the Object Navigator, expand the **Paper Layout** node, then the **Main Section** node, then the **Body** node.
4. Click the frame named **M_G_LAST_NAME_GRPFR**. Note that it is also now selected in the Paper Layout view.

5. Click the title bar of the Report Editor to make it the active window.
6. Press the down arrow key a few times to move the frame and all of its objects down the page about a quarter of an inch.

   Notice that you have moved not only M_G_LAST_NAME_GRPFR but also all the layout objects within it. Here you take advantage of Confine mode, the default mode for the Paper Layout view. With Confine mode on, an object cannot be moved outside of or placed behind its parent. Such a situation renders the layout invalid and no output is produced.

   Sometimes moving an object outside of its parent is a valid action. If you need to do so, you can turn off Confine mode by clicking the Confine Off button in the toolbar.

Figure 8–4 Paper Layout view with two layouts and white space added



7. Save your report.
8. Click the Paper Design button in the toolbar to preview your output again. Notice the additional space between the two layouts now.

8.6 Format a field

In the Paper Design view, notice the Salary field. The values are neither aligned nor displayed as monetary amounts. You can quickly rectify this in the Paper Design view.

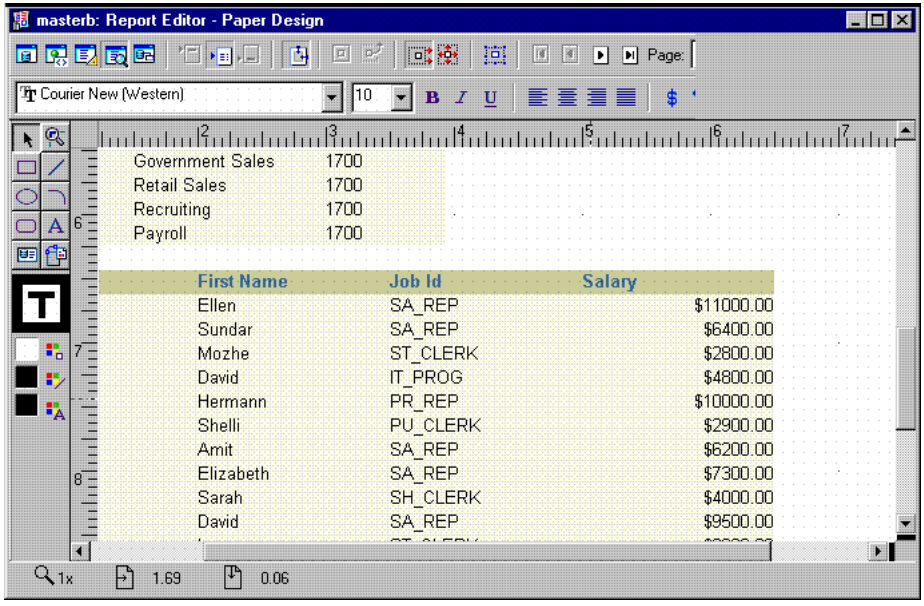
To assign a format mask to monetary values:

1. In the Paper Design view, select the first number value underneath the Salary label in the second layout. Notice that all of the values are immediately selected, indicating that you can change their properties simultaneously.

   Tip: If you are familiar with format mask syntax, you could now right-click the field values, choose Property Inspector, and choose or manually type a value for the Format Mask property.

2. Click the Currency button in the toolbar. A currency symbol immediately appears next to all of the values.
3. Click the Add Decimal Place tool twice. Two decimal places are added to the right of the decimal point.
4. Click the Align Right tool. All of the values are immediately right aligned.
5. Save your report.

Figure 8–5 Paper Design view with monetary values formatted



8.7 Summary

Congratulations! You have successfully created a master/master report. You now know how to:

- create queries with the Data Wizard.
- layout the data with the Report Wizard.
- create white space in the Paper Layout view.
- format fields in the Paper Design view.

For more information on any of the wizards, views, or properties used in this example, refer to the *Reports Builder online help*, which you can access in two ways:

- From the Oracle Technology Network (`http://otn.oracle.com/products/reports/`), click **Documentation** and navigate to the *Reports Builder Online Help* for the most recent, hosted online help.

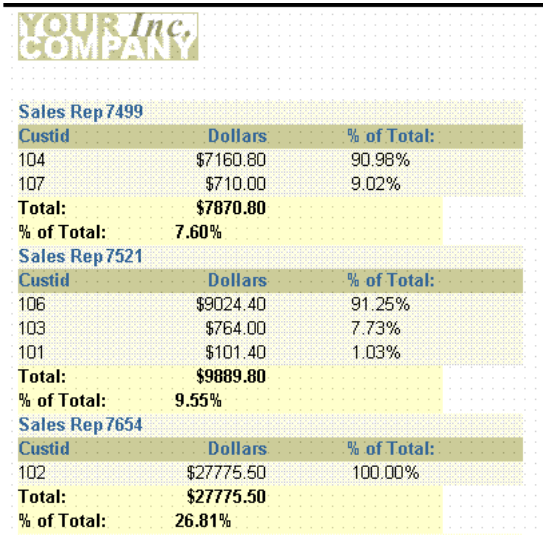
- From Reports Builder, choose **Help > Help Contents**.

> **Note:** You can replace your Reports Builder online help with the most recent update by downloading the latest online help set available on the Oracle Technology Network (http://otn.oracle.com/docs/products/reports/). Instructions for replacing your help file are included in the readme.txt in the download file.

9

Building a Summary Report

Figure 9–1 Summary report output



A summary report contains at least one column whose value or values consist of a summary of other data. A column that totals sales, a column that averages a list of commissions, and a column that shows the maximum amounts found in a series of purchase orders are all examples of summary columns.

Concepts

- The value or values in a summary column are calculated by summarizing data retrieved from another column. Reports Builder provides packaged summary functions you can use to compute the values of summary columns.

- By default, the values of a column summarizing data on a record-by-record basis appear in a default report column, as in the example above, where % of Total displays as a default tabular column.
- The values of a column summarizing data once per set of information (one summary for each break group) appear under the column of values it summarizes.
- The result of a column calculating one final result appears once at the bottom of the report.
- The difference between a summary column and the columns in previous introductory reports is that you do not select summary columns from the database. You create the summary columns and add them to groups in your report.
- To create a summary column, you need to define at least three properties:
  - *Source* is the name of the column that contains the data on which the summary column performs its computations. The source column remains unchanged.
  - *Function* is the type of summary to be performed. The function tells Reports Builder how to compute summary column values. The functions provided with Reports Builder are Average, Count, First, Last, Maximum, Minimum, % of Total, Standard Deviation, Sum, and Variance. (If none of the Reports Builder summary functions performs the computation you need for your report, you can create your own functions using PL/SQL.)
  - *Reset At* is the level or frequency at which the summary columns value returns to zero. The reset level, also known as the reset group, determines when to reset the value of the summary column to zero--in other words, how much of the source column to summarize. You can specify that the summary column summarize all values of the source column for the entire report, you can summarize column values in a break group, or you can summarize column values on a record-by-record basis.
- This report uses a master/detail layout style. You'll modify a repeating frame to add space between instances of its occurrence, shorten some field widths, move a label and a field in the Paper Layout view, edit a field label, assign some format masks.

Example Scenario

Suppose that you want to create a report that displays and summarizes sales data by sales representative. This report would include the following summaries:

- the total of all orders from all customers for each sales representative
- the percentage value of each customer's orders in relation to each sales representative's total orders
- the percentage value of each sales representative's orders in relation to the total orders
- the grand total of all orders in the report

To see a sample summary report, open the examples folder named `summary`, then open the Oracle Reports example called `summaryb.rdf`. For details on how to access it, see "Accessing the example reports" in the Preface.

Table 9–1 Features demonstrated in this example

Feature	Location
Create a data model with summaries and a layout	Section 9.2, "Create a data model and a group above layout"
Format fields for monetary values	Section 9.3, "Format fields"
Review the properties of summaries created by the Report Wizard	Section 9.4, "Examine the summary column properties (optional)"

9.1 Prerequisites for this example

To build the example in this chapter, you must have access to the Order Entry schema, which is provided by default with the Oracle9*i* database. If you don't know if you have access to the sample schema provided with the Oracle9*i* database, contact your database administrator.

9.2 Create a data model and a group above layout

When you are creating a single-query report, such as this one, you can use the Report Wizard to create the data model and layout simultaneously.

To create a data model and layout:

1. Launch Reports Builder (or, if already open, choose **File > New > Report**).
2. In the Welcome or New Report dialog box, select **Use the Report Wizard**, then click **OK**.
3. If the Welcome page displays, click **Next**.

4. On the Report Type page, select **Create Paper Layout Only**, then click **Next**.
5. On the Style page, type a **Title** for your report, select **Group Above**, then click **Next**.
6. On the Data Source page, click **SQL Query**, then click **Next**.
7. On the Data page, enter the following SELECT statement in the **Data Source definition** field:

```
SELECT SALES_REP_ID, CUSTOMER_ID, SUM(ORDER_TOTAL) TOTAL
FROM ORDERS
GROUP BY SALES_REP_ID, CUSTOMER_ID
```

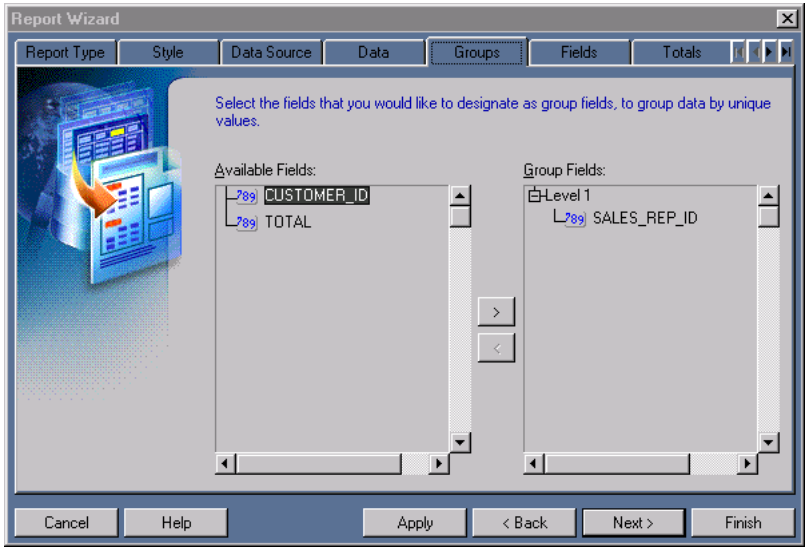
> **Note:** You can enter this query in any of the following ways:
>
> - Copy and paste the code from the provided text file called `summary_code.txt` into the **Data Source definition** field.
> - Click **Query Builder** to build the query without entering any code manually.
> - Type the code in the **Data Source definition** field.

8. Click **Next**.

> **Note:** If you are not already connected to a database, you will be prompted to connect to the database when you click **Next**. Ensure that you connect to a database that has the appropriate schema for this example. Section 9.1, "Prerequisites for this example" describes the sample schema requirements for this example.

9. On the Groups page, click **SALES_REP_ID** and click the right arrow (**>**) to move this field to the **Group Fields** list, then click **Next**.

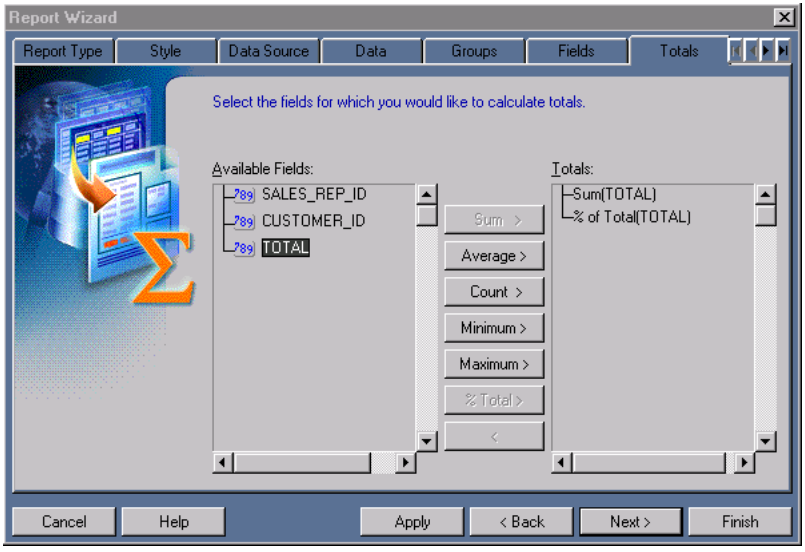
Figure 9–2 Groups page of the Report Wizard



10. On the Fields page, click the double right arrows (>>) to move all of the fields to the **Displayed Fields** list, then click **Next**.
11. On the Totals page, click **TOTAL** in the **Available Fields** list, then click **Sum**. Given the data model you are using, this step will create two summary columns for you:
    - SumTOTALPerSALES_REP_ID sums the total of TOTAL for each value of SALES_REP_ID (i.e., for each sales representative).
    - SumTOTALPerReport sums the total of TOTAL for the entire report (i.e., for all sales representatives).
12. Still on the Totals page, click **% of Total**. Given the data model you are using, this step will create two columns for you:
    - TotalTOTALPerCUSTOMER_ID calculates dollars for each customer (CUSTOMER_ID) as a percentage of the total dollars for each sales representative (SALES_REP_ID).

- TotalTOTALPerSALES_REP_ID calculates dollars for each sales representative (SALES_REP_ID) as a percentage of the total dollars in the entire report.

Figure 9–3 Totals page of Report Wizard



13. Click **Next**.
14. On the Labels page, change the labels as follows, then click **Next**:

Fields	Labels
SALES_REP_ID	`Sales Rep`
CUSTOMER_ID	`Customer`
TOTAL	`Dollars`

15. On the **Template** page, make sure **Beige** is selected under **Predefined Template**.

16. Click **Finish** to display your report output in the Paper Design view.

9.3 Format fields

In the Paper Design view, notice the Total field. The values are neither aligned nor displayed as monetary amounts. You can quickly rectify this in the Paper Design view.

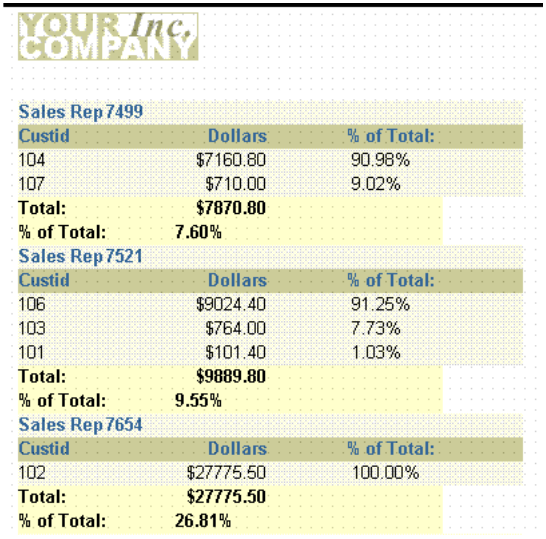
To assign a format mask to monetary values:

1. In the Paper Design view, select the first number value underneath the Dollars label. Notice that all of the values are immediately selected, indicating that you can change their properties simultaneously.

   Tip: If you are familiar with format mask syntax, you could now right click on the field values, choose Property Inspector, and choose or manually enter a value for the Format Mask property.

2. Click the Currency button in the toolbar. A currency symbol immediately appears next to all of the values.
3. Click the Add Decimal Place button twice. Two decimal places are added to the right of the decimal point.
4. Resize the field by clicking and dragging the rightmost handle of the field approximately 0.5 inches to the left.
5. Click the Align Right button. All of the values are immediately right aligned.
6. Select the Dollars label itself.
7. Click the Align Right button.
8. Select the number value to the right of the **Total:** label.
9. Click the Currency button. A currency symbol immediately appears next to all of the values.
10. Click the Add Decimal Place button twice. Two decimal places are added to the right of the decimal point.
11. Click the Align Right button. All of the values are immediately right aligned.

Figure 9–4 Summary report output with monetary values formatted

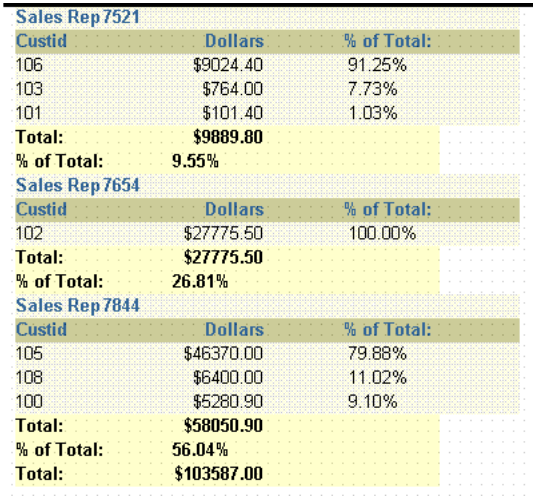


YOUR Inc. COMPANY

Sales Rep 7499		
Custid	Dollars	% of Total:
104	$7160.80	90.98%
107	$710.00	9.02%
Total:	**$7870.80**	
% of Total:	**7.60%**	
Sales Rep 7521		
Custid	Dollars	% of Total:
106	$9024.40	91.25%
103	$764.00	7.73%
101	$101.40	1.03%
Total:	**$9889.80**	
% of Total:	**9.55%**	
Sales Rep 7654		
Custid	Dollars	% of Total:
102	$27775.50	100.00%
Total:	**$27775.50**	
% of Total:	**26.81%**	

12. Go to the end of the report. The last value on the last page should be a summary labelled Total:, which sums the values of TOTAL across the entire report.
13. Select the unformatted number to the right of the Total: label.
14. Click and drag this field until its right edge is aligned with the right edge of the values in the Total column.
15. Click the Currency button in the toolbar. A currency symbol immediately appears next to all of the values.
16. Click the Add Decimal Place button twice. Two decimal places are added to the right of the decimal point.
17. Click the Align Right button. All of the values are immediately right aligned.
18. Save your report.

Figure 9–5 Summary report with report total formatted



Sales Rep 7521		
Custid	Dollars	% of Total:
106	$9024.40	91.25%
103	$764.00	7.73%
101	$101.40	1.03%
Total:	$9889.80	
% of Total:	9.55%	
Sales Rep 7654		
Custid	Dollars	% of Total:
102	$27775.50	100.00%
Total:	$27775.50	
% of Total:	26.81%	
Sales Rep 7844		
Custid	Dollars	% of Total:
105	$46370.00	79.88%
108	$6400.00	11.02%
100	$5280.90	9.10%
Total:	$58050.90	
% of Total:	56.04%	
Total:	$103587.00	

9.4 Examine the summary column properties (optional)

In this case, the Report Wizard created the summaries according to the requirements of the project. However, in some cases, you may need to manually adjust the settings of summaries to get the exact calculations you wish. By carefully reviewing the summaries created by the Report Wizard, you can gain a better understanding of how summaries work.

In this section, you will not be making any changes to the report, you will simply be checking the summary settings to better understand summaries.

Reviewing summary settings

1. In the Paper Design view, select the percentage value to the right of the **% of Total** label.
2. Double-click the values to display the Property Inspector.
3. In the Property Inspector, notice the Source property, which indicates the column that is the source of the field.
4. Repeat steps 1 through 3 for all of the summaries in the report. Note down the name of the columns that are the sources for each of these summary fields.

5. In the Object Navigator, double-click the view icon next to the **Data Model** node to display the Data Model view.
6. In the Data Model view, double-click **TotalTOTALPerSALES_REP_ID** in group **G_SALES_REP_ID** to display the Property Inspector.
7. In the Property Inspector, notice the values of the properties under **Summary**:
   - Function is % of Total. Change % of Total to Sum. Note that the Compute At property disappears because it is unnecessary for a Sum calculation. Change Function back to % of Total and Compute At reappears.
   - Source is the TOTAL column, which means that TOTAL is used to compute the summary.
   - Reset At is G_SALES_REP_ID. The value of TotalTOTALPerREPID will reset to zero after each record in G_SALES_REP_ID (i.e., for each sales representative).
   - Compute At is Report. A compute level of Report means that the TotalTOTALPerSALES_REP_ID column will base its percentages on the sum of all of the TOTAL in the entire report.
8. Repeat steps 6 and 7 for each of the summaries in the data model. Notice the differences in the properties for each of the summaries.
9. It can also be a useful exercise to return to the Paper Design view and see where fields that correspond to the summaries are placed in the layout. For example, the field that corresponds to TotalTOTALPerSALES_REP_ID is placed inside the master repeating frame, R_G_SALES_REP_ID, but outside of the detail repeating frame, R_G_CUSTOMER_ID.

9.5 Summary

Congratulations! You have successfully created a summary report. You now know how to:

- create a data model with summaries and lay out the data with the Report Wizard.
- format fields in the Paper Design view.
- examine the summaries and their properties.

For more information on any of the wizards, views, or properties used in this example, refer to the *Reports Builder online help*, which you can access in two ways:

- From the Oracle Technology Network (`http://otn.oracle.com/products/reports/`), click **Documentation** and navigate to the *Reports Builder Online Help* for the most recent, hosted online help.
- From Reports Builder online help, choose **Help > Help Contents**.

> **Note:** You can replace your Reports Builder online help with the most recent update by downloading the latest online help set available on the Oracle Technology Network (http://otn.oracle.com/docs/products/reports/). Instructions for replacing your help file are included in the readme.txt in the download file.

Part II

Building Group Reports

The chapters in this Part provide steps to build group reports. In these reports, the rows of output are displayed as groups based on a single query or on multiple queries. Group reports can contain formula columns and the values of the report can be printed from top to bottom or across a page.

- Chapter 10, "Building a Single-Query Group Report"

  A single-query group report contains a single query and divides the rows of a report into groups based on common values in one or more column(s). For example, employees can be grouped based on department numbers, which means each department number prints only once for every group. Otherwise, the department number would print once for every employee in the department. These reports include the group left and group above reports and were earlier known as break reports or master/detail reports.

- Chapter 11, "Building a Two-Query Group Report"

  A two-query group report appears much the same as a single-query group report except that the former report uses multiple queries. Multiple queries enable you to simplify maintenance or make the data model easier to understand.

- Chapter 12, "Building an Across Group Report"

  An across group report prints the values of a database column across the page instead of down. In these reports, the values in the master group print from top to bottom, and the values in the detail group print across the page, from left to right.

- Chapter 13, "Building a Group Left Summary Report"

A group left summary report consists of master record values, detail records, and summaries. Summaries are totals calculated for the details under each master record.

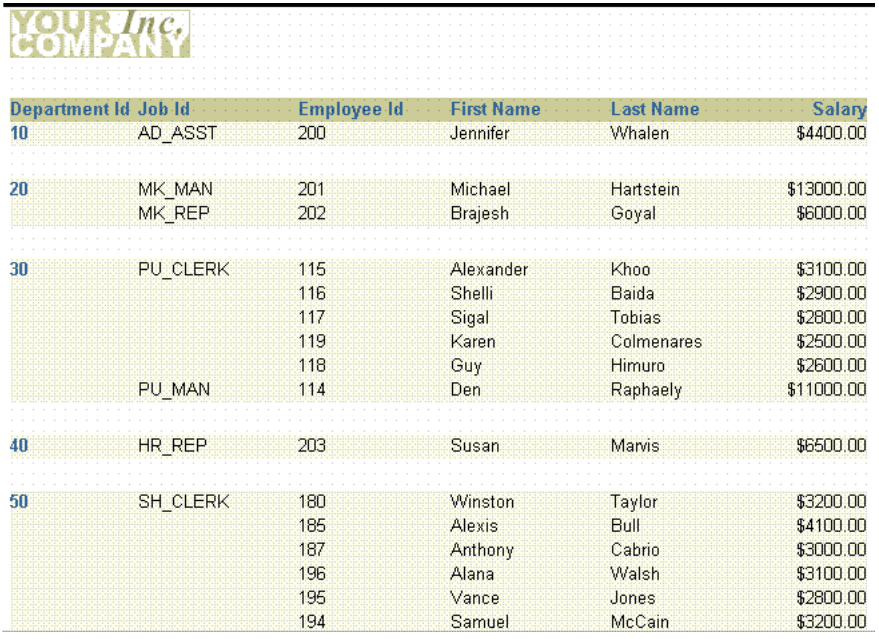
- Chapter 14, "Building a Group Left Formula Report"

  A group left formula report contains a formula column, for which values are calculated based on a PL/SQL formula. A formula column, like a summary column, is a computational column that you create. This column performs user-defined calculations on other column(s) data including placeholder columns.

10

Building a Single-Query Group Report

Figure 10–1 Group report output



Department Id	Job Id	Employee Id	First Name	Last Name	Salary
10	AD_ASST	200	Jennifer	Whalen	$4400.00
20	MK_MAN	201	Michael	Hartstein	$13000.00
	MK_REP	202	Brajesh	Goyal	$6000.00
30	PU_CLERK	115	Alexander	Khoo	$3100.00
		116	Shelli	Baida	$2900.00
		117	Sigal	Tobias	$2800.00
		119	Karen	Colmenares	$2500.00
		118	Guy	Himuro	$2600.00
	PU_MAN	114	Den	Raphaely	$11000.00
40	HR_REP	203	Susan	Marvis	$6500.00
50	SH_CLERK	180	Winston	Taylor	$3200.00
		185	Alexis	Bull	$4100.00
		187	Anthony	Cabrio	$3000.00
		196	Alana	Walsh	$3100.00
		195	Vance	Jones	$2800.00
		194	Samuel	McCain	$3200.00

Group left and group above reports divide the rows of a report into ”sets,” based on common values in one or more of the columns, such as the department number in the example above. Notice that each department number prints only once. If the report above was not a group report, the department number would print once for each employee in the department rather than just once for the whole department.

Note: In earlier releases, this type of report was commonly referred to as a break report or a master/detail report.

Example Scenario

Suppose that you want to create a group left report that lists employees with their jobs and salaries by department.

Table 10–1 Features demonstrated in this example

Feature	Location
Create a data model with a break group and default the layout	Section 10.2.1, "Create a data model with a break group and group left layout"
Format a field	Section 10.2.2, "Format a field"
Add white space between records	Section 10.2.3, "Use the Property Inspector to add white space"
Add another column to the break group	Section 10.3, "Group report with two break columns"
Add a second break group	Section 10.4, "Group report with two break groups"

10.1 Prerequisites for this example

To build the example in this chapter, you must have access to the sample schema provided with the Oracle9*i* database. If you don't know if you have access to this sample schema, contact your database administrator. You should have access to the "Human Resources" portion of the schema to complete this example.

To see a sample across report with control breaks, open the examples folder called `break`, then open the Oracle Reports example report named `grp_lft2.rdf`. For details on how to open it, see "Accessing the example reports" in the Preface.

10.2 Group report with one break column

The most basic case of a group left or above report is to have one break group with one break column in it. This means that a single master field value will print once for many detail field values.

To see a sample group left report with one break column, open the examples folder named `break`, then open the Oracle Reports example named `grp1col_lft1.rdf`. For details on how to access it, see "Accessing the example reports" in the Preface.

Concepts

- Create a group left (or above) report when you want to prevent a column from repeatedly displaying the same value while other column values in the rows change.
- You can build a group left or above report with either a single-query or multiple queries. A single-query report typically runs faster than a multiquery report.
- You'll need to make sure your SELECT statement includes a column, called a break column, containing at least one value which repeats over multiple records (e.g., department number in the example report above). With the break column, you'll create a second group, called a break group, to create the breaks in the data.
- This report uses a default tabular format. You'll improve the readability of the report by changing the vertical spacing for a repeating frame to insert a space between each set of department information (e.g., between the last record of Department 10 and the first record of Department 20, as shown in the example report above). You'll also add a format mask.

10.2.1 Create a data model with a break group and group left layout

When you are creating a single-query report, such as this one, you can use the Report Wizard to create the data model and layout simultaneously.

To create a data model and layout:

1. Launch Reports Builder (or, if already open, choose **File > New > Report**)
2. In the Welcome or New Report dialog box, select **Use the Report Wizard**, then click **OK**.
3. If the Welcome page displays, click **Next**.
4. On the Report Type page, select **Create Paper Layout Only**, then click **Next**.
5. On the Style page, type a **Title** for your report, select **Group Left**, then click **Next**.
6. On the Data Source page, click **SQL Query**, then click **Next**.

7. On the Data page, enter the following SELECT statement in the **Data Source definition** field:

```
SELECT EMPLOYEE_ID, FIRST_NAME, LAST_NAME,
JOB_ID, SALARY, DEPARTMENT_ID
FROM EMPLOYEES
```

Note: You can enter this query in any of the following ways:

- Copy and paste the code from the provided text file called `across_code.txt` into the **Data Source definition** field.
- Click **Query Builder** to build the query without entering any code manually.
- Type the code in the **Data Source definition** field.

---

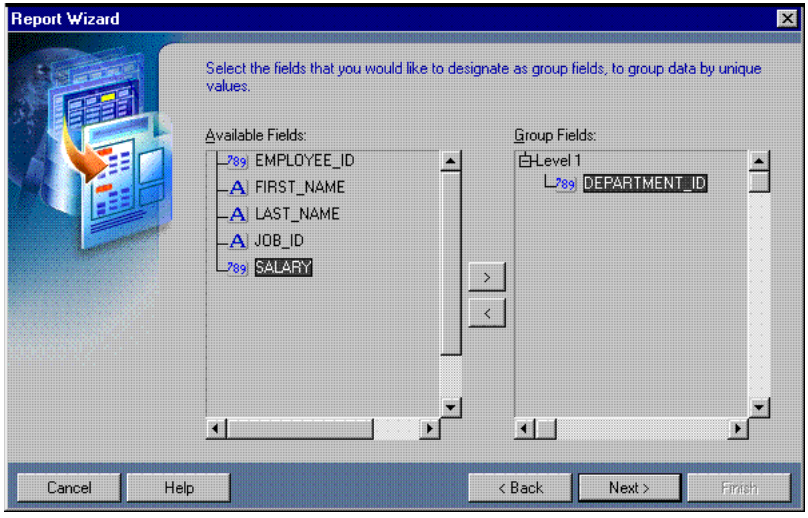
8. Click **Next**.

Note: If you are not already connected to a database, you will be prompted to connect to the database when you click **Next**. Ensure that you connect to a database that has the appropriate schema for this example. Section 10.1, "Prerequisites for this example" describes the sample schema requirements for this example.

---

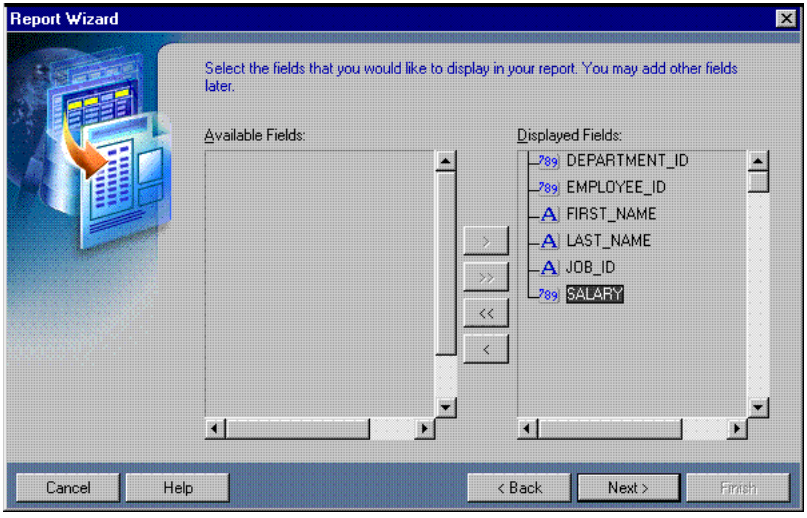
9. On the Groups page, click **DEPARTMENT_ID** and click the right arrow (>) to move this field to the **Group Fields** list, then click **Next**.

Figure 10–2 Groups page of the Report Wizard



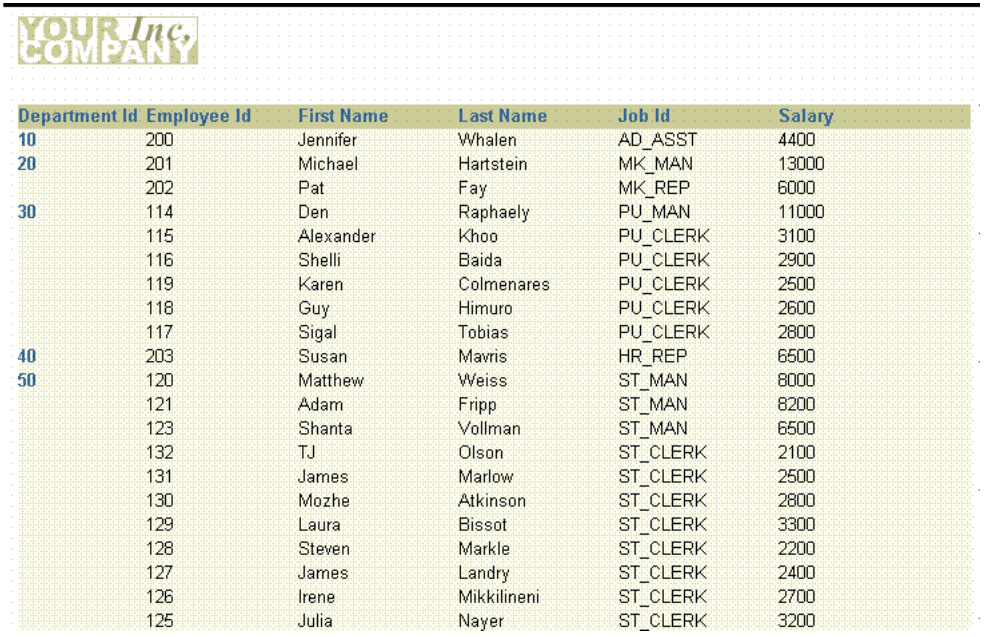
10. On the Fields page, click the double right arrows (>>) to move all of the fields to the **Displayed Fields** list.

Figure 10–3 Fields page of the Report Wizard



11. Click **Next** until you reach the Template page of the Report Wizard.
12. On the Template page, select **Predefined Template** and click **Beige**, then click **Finish** to display your report output in the Paper Design view. It should look something like this:

Figure 10–4 Paper Design view for the single-query group left report



Department Id	Employee Id	First Name	Last Name	Job Id	Salary
10	200	Jennifer	Whalen	AD_ASST	4400
20	201	Michael	Hartstein	MK_MAN	13000
	202	Pat	Fay	MK_REP	6000
30	114	Den	Raphaely	PU_MAN	11000
	115	Alexander	Khoo	PU_CLERK	3100
	116	Shelli	Baida	PU_CLERK	2900
	119	Karen	Colmenares	PU_CLERK	2500
	118	Guy	Himuro	PU_CLERK	2600
	117	Sigal	Tobias	PU_CLERK	2800
40	203	Susan	Mavris	HR_REP	6500
50	120	Matthew	Weiss	ST_MAN	8000
	121	Adam	Fripp	ST_MAN	8200
	123	Shanta	Vollman	ST_MAN	6500
	132	TJ	Olson	ST_CLERK	2100
	131	James	Marlow	ST_CLERK	2500
	130	Mozhe	Atkinson	ST_CLERK	2800
	129	Laura	Bissot	ST_CLERK	3300
	128	Steven	Markle	ST_CLERK	2200
	127	James	Landry	ST_CLERK	2400
	126	Irene	Mikkilineni	ST_CLERK	2700
	125	Julia	Nayer	ST_CLERK	3200

10.2.2 Format a field

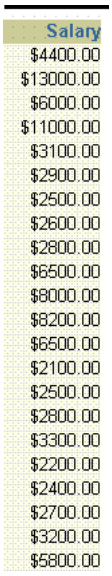
In the Paper Design view, notice the Salary field. The values are neither aligned nor displayed as monetary amounts. You can quickly rectify this in the Paper Design view.

To assign a format mask to monetary values:

1. In the Paper Design view, click the first number value underneath the **Salary** label. Notice that all of the values are immediately selected, indicating that you can change their properties simultaneously.
2. Click the Currency button in the toolbar. A currency symbol immediately appears next to all of the values.
3. Click the Add Decimal Place tool twice. Two decimal places are added to the right of the decimal point.
4. Resize the fields. Click and drag the rightmost handle of the cell value under the Salary label about 0.5 inches to the left.

5. Shift-click on the **Salary** label.
6. Click the Align Right tool in the tool bar.
7. Click in an open area of the Paper Design view to deselect all of the objects.
8. Save your report.

Figure 10–5 Salaries formatted



10.2.3 Use the Property Inspector to add white space

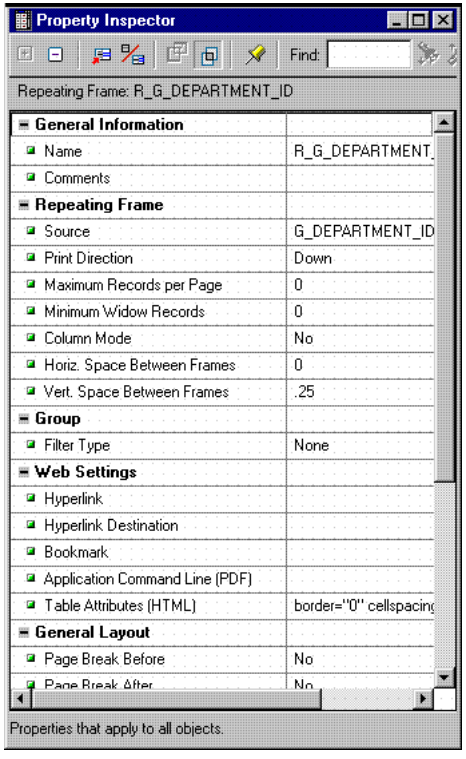
If you examine your report output in the Paper Design view, you will notice that it can be difficult to distinguish where one department's data ends and the next department's data begins. To make the report more readable, you want to add some white space between the departments, but you want to retain the same spacing between employee rows. For this report, you'll change a repeating frame property for the master repeating frame to create some white space between the department records.

To create white space between the departments:

1. In the Paper Design view, click the first number value underneath the **Department Id** label. Notice that all of the department numbers are immediately selected.
2. Click the Select Parent Frame button in the toolbar. Notice how the border of the repeating frame that contains the Department Id field is now highlighted.
3. Choose **Tools > Property Inspector** to display the Property Inspector, and set properties:

- Under **Repeating Frame**, set the Vert. Space Between Frames property to 0.25.

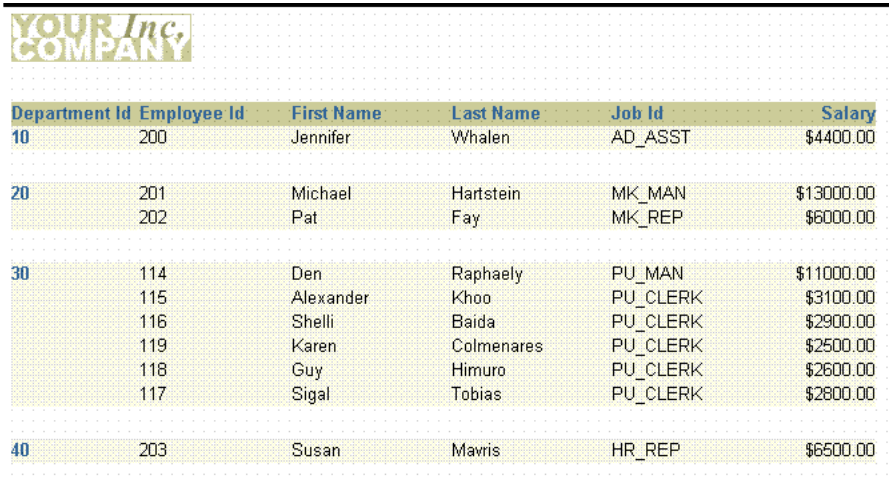
Figure 10–6 Property Inspector for master repeating frame



4. Click the title bar of the Report Editor to make it the active window again.

5. Save your report.

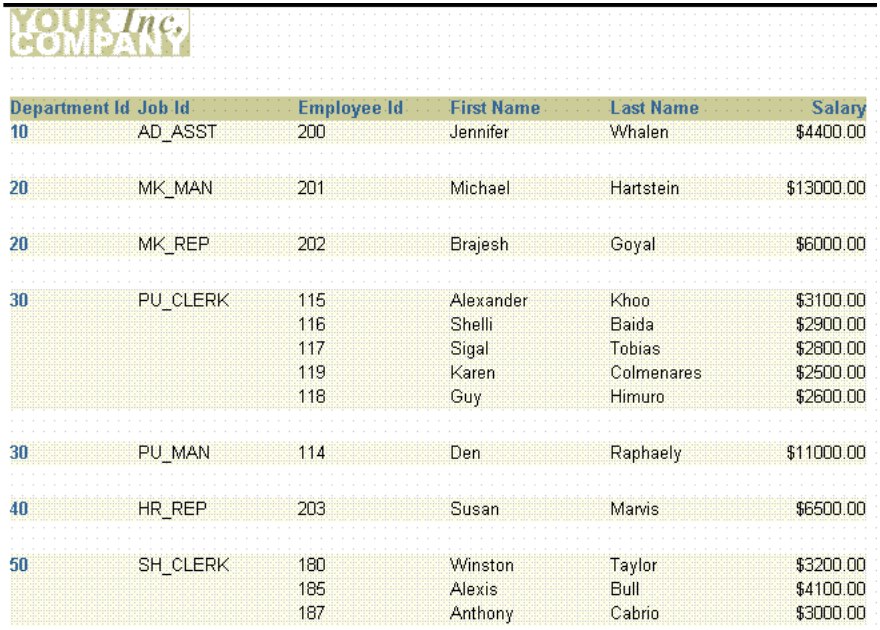
Figure 10–7 Paper Design view after white space is added



Department Id	Employee Id	First Name	Last Name	Job Id	Salary
10	200	Jennifer	Whalen	AD_ASST	$4400.00
20	201	Michael	Hartstein	MK_MAN	$13000.00
	202	Pat	Fay	MK_REP	$6000.00
30	114	Den	Raphaely	PU_MAN	$11000.00
	115	Alexander	Khoo	PU_CLERK	$3100.00
	116	Shelli	Baida	PU_CLERK	$2900.00
	119	Karen	Colmenares	PU_CLERK	$2500.00
	118	Guy	Himuro	PU_CLERK	$2600.00
	117	Sigal	Tobias	PU_CLERK	$2800.00
40	203	Susan	Mavris	HR_REP	$6500.00

10.3 Group report with two break columns

Figure 10–8 Group left report output with two break columns



Department Id	Job Id	Employee Id	First Name	Last Name	Salary
10	AD_ASST	200	Jennifer	Whalen	$4400.00
20	MK_MAN	201	Michael	Hartstein	$13000.00
20	MK_REP	202	Brajesh	Goyal	$6000.00
30	PU_CLERK	115	Alexander	Khoo	$3100.00
		116	Shelli	Baida	$2900.00
		117	Sigal	Tobias	$2800.00
		119	Karen	Colmenares	$2500.00
		118	Guy	Himuro	$2600.00
30	PU_MAN	114	Den	Raphaely	$11000.00
40	HR_REP	203	Susan	Marvis	$6500.00
50	SH_CLERK	180	Winston	Taylor	$3200.00
		185	Alexis	Bull	$4100.00
		187	Anthony	Cabrio	$3000.00

The report above looks similar to the group left report you built in Section 10.2, "Group report with one break column". However, notice that DEPARTMENT_ID values sometimes print more frequently than they did in the previous report. The DEPARTMENT_ID value repeats for each unique value of JOB_ID within the department. This behavior occurs because DEPARTMENT_ID is now grouped with JOB_ID and must print with JOB_ID. While DEPARTMENT_ID values may repeat several times, not until the position of PU_CLERK in department 30 does a job repeat, and not until that point can the break group actually break.

Concepts

- You can specify that your report break on certain combinations of information by varying the columns you include in the break group.
- You can modify your previous report by moving JOB_ID into the break group, so that your report has two break columns instead of one.

- After changing the data model, redefault the layout to incorporate your changes, then specify the format mask again.

To see a sample group left report with two break columns, open the examples folder named `break`, then open the Oracle Reports example named `grp2col_lft1.rdf`. For details on how to access it, see "Accessing the example reports" in the Preface.

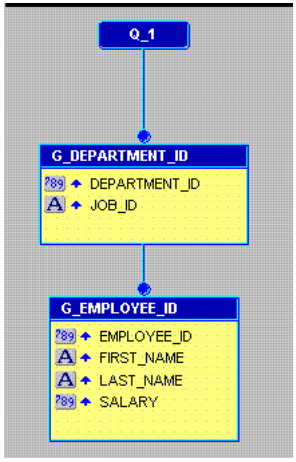
10.3.1 Modify the data model

The first task in changing your previous report is to modify the data model by placing an additional column in the break group.

To add a column to the break group:

1. Open the report you created in the previous section.
2. In the Object Navigator, double-click the view icon next to the **Data Model** node to display the Data Model view.
3. In the Data Model view, click the break group, **G_DEPARTMENT_ID**, then click and drag the handle on the bottom center of the **G_DEPARTMENT_ID** group box down about a quarter of an inch to resize it.
4. Click and drag on the **JOB_ID** column in the **G_EMPLOYEE_ID** group and move it into the **G_DEPARTMENT_ID** group, underneath the **DEPARTMENT_ID** column.

Figure 10–9 Data model with two break columns



10.3.2 Redefault the layout

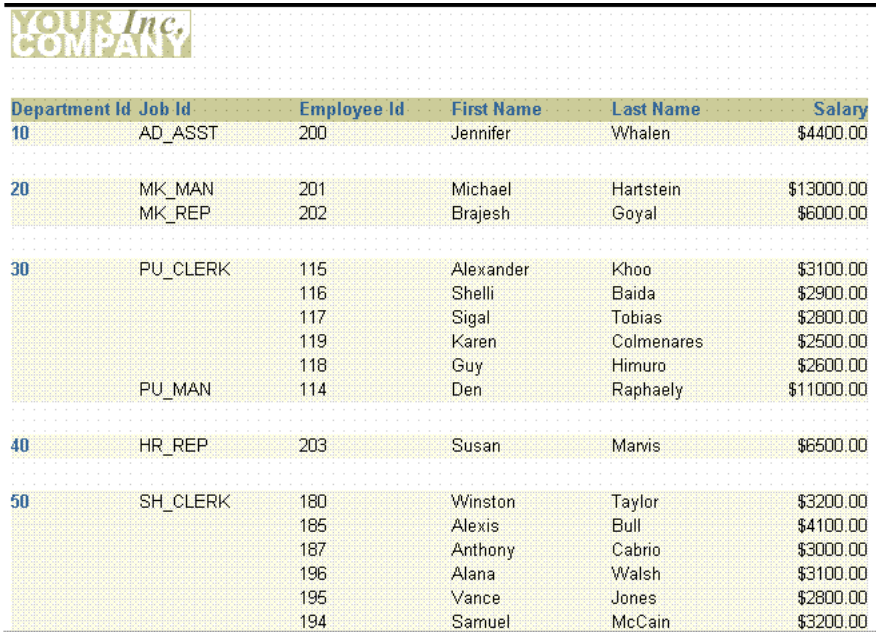
In order for your data model change to be reflected in your output, you need to redefault the layout for your report using the Report Wizard.

To redefault the layout with the Report Wizard:

1. Click the title bar of the Report Editor to make it the active window. The Report Editor must be the active window in order for you to access the Report Wizard.
2. Choose **Tools > Report Wizard**.
3. Select **Create Paper Layout only**.
4. Click **Finish**. Notice the changes to the output in the Paper Design view. Also note how the formatting of the Salary field and the additional spacing between records is retained. When possible, Reports Builder will retain your manual modifications between uses of the Report Wizard.
5. Save your report.

10.4 Group report with two break groups

Figure 10–10 Group left report output with two break groups



Department Id	Job Id	Employee Id	First Name	Last Name	Salary
10	AD_ASST	200	Jennifer	Whalen	$4400.00
20	MK_MAN	201	Michael	Hartstein	$13000.00
	MK_REP	202	Brajesh	Goyal	$6000.00
30	PU_CLERK	115	Alexander	Khoo	$3100.00
		116	Shelli	Baida	$2900.00
		117	Sigal	Tobias	$2800.00
		119	Karen	Colmenares	$2500.00
		118	Guy	Himuro	$2600.00
	PU_MAN	114	Den	Raphaely	$11000.00
40	HR_REP	203	Susan	Marvis	$6500.00
50	SH_CLERK	180	Winston	Taylor	$3200.00
		185	Alexis	Bull	$4100.00
		187	Anthony	Cabrio	$3000.00
		196	Alana	Walsh	$3100.00
		195	Vance	Jones	$2800.00
		194	Samuel	McCain	$3200.00

The figure above shows a further modification of your group left report. In this version, DEPARTMENT_ID prints only once for each department. JOB_ID also prints only once when more than one employee in a department has the same job. Both columns are now in separate break groups and DEPARTMENT_ID is in the higher group. Hence, it does not repeat for each unique value of JOB_ID within a department any more.

Concepts

- You can also create reports that include multiple break groups and assign one or more columns to each break group.
- You'll further modify the data model to include two break groups: G_DEPARTMENT_ID and G_JOB_ID.
- After changing the data model, redefault the layout to incorporate your changes, then specify the format mask again.

To see a sample group left report with two break groups, open the examples folder named `break`, then open the Oracle Reports example named `grp_lft2.rdf`. For details on how to access it, see "Accessing the example reports" in the Preface.

10.4.1 Modify the data model

The first task in changing your previous report is to modify the data model by creating another break group.

To add another break group:

1. Open the report you created in the previous section.
2. In the Object Navigator, double-click the view icon next to the **Data Model** node to display the Data Model view.
3. In the Data Model view, click and drag the group, **G_EMPLOYEE_ID**, down about a quarter of an inch.
4. Click the **JOB_ID** column in **G_DEPARTMENT_ID** and drag it to a spot somewhere in between **G_DEPARTMENT_ID** and **G_EMPLOYEE_ID**. Another break called **G_JOB_ID** is created.

10.4.2 Redefault the layout

In order for your data model change to be reflected in your output, you need to redefault the layout for your report using the Report Wizard.

To redefault the layout with the Report Wizard:

1. Click the title bar of the Report Editor to make it the active window. The Report Editor must be the active window in order for you to access the Report Wizard.
2. Choose **Tools > Report Wizard**.
3. Select **Create Paper Layout only**.
4. Click **Finish**. Notice the changes to the output in the Paper Design view.
5. Save your report.

10.5 Summary

Congratulations! You have successfully created a single-query group report. You now know how to:

- create a data model and layout with the Report Wizard.
- format fields in the Paper Design view.
- add white space between records with the Property Inspector.
- add another column to the break group and redefault the layout.
- add another break group and redefault the layout.

For more information on any of the wizards, views, or properties used in this example, refer to the *Reports Builder online help*, which you can access in two ways:

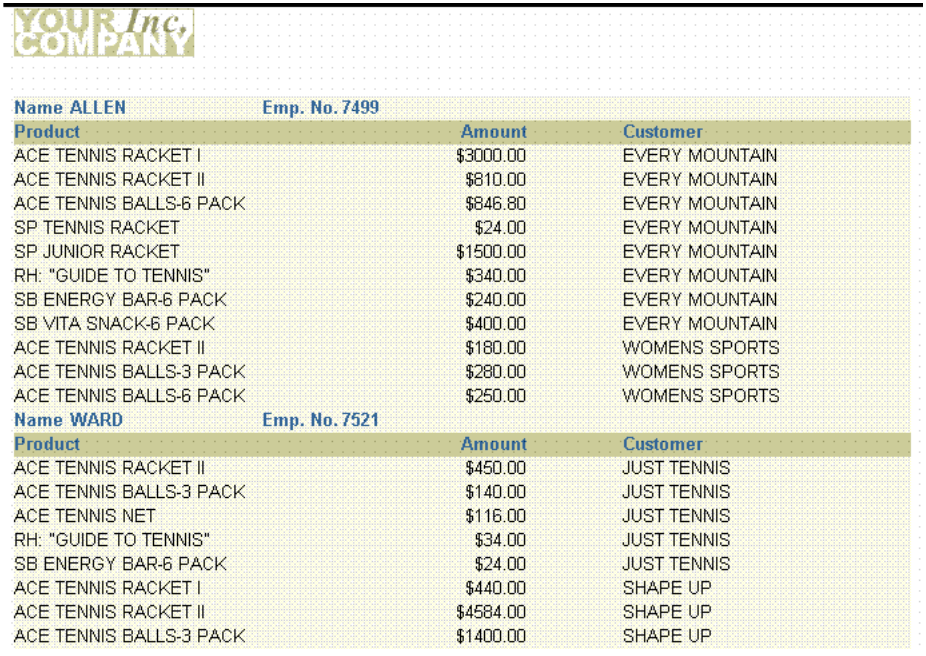
- From the Oracle Technology Network (`http://otn.oracle.com/products/reports/`), click **Documentation** and navigate to the *Reports Builder Online Help* for the most recent, hosted online help.
- From Reports Builder, choose **Help > Help Contents**.

Note: You can replace your Reports Builder online help with the most recent update by downloading the latest online help set available on the Oracle Technology Network (http://otn.oracle.com/docs/products/reports/). Instructions for replacing your help file are included in the readme.txt in the download file.

11

Building a Two-Query Group Report

Figure 11–1 Group above report output, two queries



YOUR Inc. COMPANY

Name ALLEN	Emp. No. 7499	
Product	**Amount**	**Customer**
ACE TENNIS RACKET I	$3000.00	EVERY MOUNTAIN
ACE TENNIS RACKET II	$810.00	EVERY MOUNTAIN
ACE TENNIS BALLS-6 PACK	$846.80	EVERY MOUNTAIN
SP TENNIS RACKET	$24.00	EVERY MOUNTAIN
SP JUNIOR RACKET	$1500.00	EVERY MOUNTAIN
RH: "GUIDE TO TENNIS"	$340.00	EVERY MOUNTAIN
SB ENERGY BAR-6 PACK	$240.00	EVERY MOUNTAIN
SB VITA SNACK-6 PACK	$400.00	EVERY MOUNTAIN
ACE TENNIS RACKET II	$180.00	WOMENS SPORTS
ACE TENNIS BALLS-3 PACK	$280.00	WOMENS SPORTS
ACE TENNIS BALLS-6 PACK	$250.00	WOMENS SPORTS
Name WARD	**Emp. No. 7521**	
Product	**Amount**	**Customer**
ACE TENNIS RACKET II	$450.00	JUST TENNIS
ACE TENNIS BALLS-3 PACK	$140.00	JUST TENNIS
ACE TENNIS NET	$116.00	JUST TENNIS
RH: "GUIDE TO TENNIS"	$34.00	JUST TENNIS
SB ENERGY BAR-6 PACK	$24.00	JUST TENNIS
ACE TENNIS RACKET I	$440.00	SHAPE UP
ACE TENNIS RACKET II	$4584.00	SHAPE UP
ACE TENNIS BALLS-3 PACK	$1400.00	SHAPE UP

As you can see above, a two-query group report appears much the same as a single-query group report. Performance is the key issue when contrasting single-query and multiple-query group reports. In most cases, single-query reports will run faster than multiple-query reports. The multiple-query reports are,

however, sometimes easier to understand conceptually and easier to maintain. For example, if you are in a situation where only a few users run the report and the report returns a relatively small number of records, you might want to use multiple queries to simplify maintenance and make the data model easier to understand. If you have many users and the report is quite large, then you should try to use a single-query report.

Concepts

- The single and multiple-query group reports look the same but have different data models. A single-query report has multiple groups underneath one query, while a multiple-query report uses one group underneath each of several queries. In the single-query case, the relationship is established by the hierarchy of groups underneath the query. In the multiple-query case, the relationship is established by data links between the groups of different queries.
- A data link is a data model object that enables you to relate multiple queries. For a simple group report, you relate the two queries using the primary and foreign keys of the tables from which you are selecting data.
- A primary key is a column for which each value uniquely identifies the record in which it is found. A foreign key is a column which contains the same values as the primary key for another table, and is used to reference records in that table.
- Linking two tables via primary and foreign keys is similar to specifying a join condition. In fact, the data link causes Reports Builder to create a SQL clause defining a join, based on information specified when creating the link. This clause is added to the child query's SELECT statement at run time.
- The join defined by a default data link is an outer join—in addition to returning all rows that satisfy the link's condition, an outer join returns all rows from one table that do not match a row from the second table.
- This report uses a default group above layout style in which master records display across the page with the labels to the left of their fields and the detail records appear below the master records in standard tabular format.
- You'll deselect one default column to keep it from appearing in the report, change some field widths to ensure the fields fit across the page, and format one of the fields.
- You will also specify a maximum of 1 master record per page to ensure that only one master record and its associated detail records are displayed per page of report output.

Example Scenario

Suppose that you want to create a group above report that lists employees with their jobs and salaries by department.

To see a sample group above report with two queries, open the examples folder named `masterdetail`, then open the Oracle Reports example named `grp_abv2.rdf`. For details on how to access it, see "Accessing the example reports" in the Preface.

Table 11–1 Features demonstrated in this example

Feature	Location
Create a new, empty report	Section 11.2, "Create a new report manually"
Create two queries with a data link between them	Section 11.3, "Create a data model with a data link"
Layout the data	Section 11.4, "Use the Report Wizard to layout the data"
Format monetary values	Section 11.5, "Format a field"

11.1 Prerequisites for this example

To build the example in this chapter, you must have access to the Summit Sporting Goods schema, which we've provided on the Oracle Technology Network (`http://otn.oracle.com/products/reports/`). To download the SQL scripts that install the schema, go to the Documentation page on OTN and follow the instructions provided on the Web page.

11.2 Create a new report manually

In this case, it is easier to create the data model and layout separately. Hence, we will create an empty report first, then add the queries, and then create the layouts.

To create a blank report:

1. Launch Reports Builder (or, if already open, choose **File > New > Report**)
2. In the Welcome or New Report dialog box, select **Build a new report manually**, then click **OK**.

11.3 Create a data model with a data link

When you create a report with multiple queries, it is typically easier to create all of the queries with the Data Wizard first and then create the layouts with the Report Wizard.

To create the queries:

1. In the Data Model view, choose **Insert > Query** to display the Data Wizard.
2. If the Welcome page displays, click **Next**.
3. On the Query page, type `Q_Salesrep` for the Query name and click **Next**.
4. On the Data Source page, click **SQL Query**, then click **Next**.
5. On the Data page, in the **Data Source definition** field, enter the following SELECT statement:

```
SELECT ENAME, EMPNO
FROM EMP
WHERE JOB = 'SALESMAN'
```

> **Note:** You can enter this query in any of the following ways:
>
> - Copy and paste the code from the provided text file called `grp_abv2_code.txt` into the **Data Source definition** field.
> - Click **Query Builder** to build the query without entering any code manually.
> - Type the code in the **Data Source definition** field.

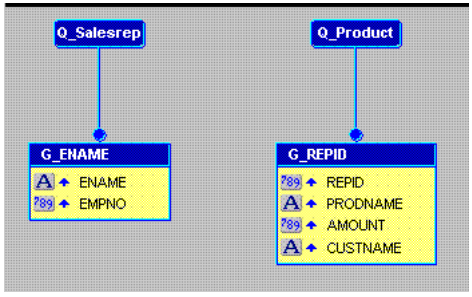
6. Click **Next**.

> **Note:** If you are not already connected to a database, you will be prompted to connect to the database when you click **Next**. Ensure that you connect to a database that has the appropriate schema for this example. Section 11.1, "Prerequisites for this example" describes the sample schema requirements for this example.

7. On the Groups page, click **Next**.
8. Click **Finish** to display the data model for your report in the Data Model view.
9. Repeat the steps above for a second query, but this time name your query `Q_Product` and use the following SELECT statement:

```
SELECT REPID, PRODNAME, AMOUNT, CUSTNAME
FROM SALES
ORDER BY REPID, CUSTNAME
```

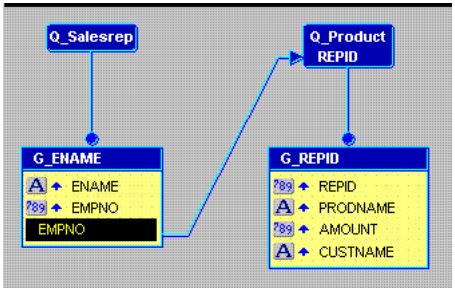
Figure 11–2 Two-query data model without a link



To add the data link

1. In the Data Model view, click the Data Link tool in the tool palette.
2. Click and drag from the **EMPNO** column in the **G_ENAME** group to the **REPID** column in the **G_REPID** group. Notice that a line is drawn from the bottom of the **G_ENAME** group to the **Q_Product** query. Labels for **EMPNO** and **REPID** are created at each end of the line to indicate they are the columns linking **G_ENAME** to **Q_Product**.

Figure 11–3 Two-query data model with a data link



3. Double-click the new data link line to display the Property Inspector and examine the property settings:

   - G_ENAME is identified as the parent, while Q_Product is listed as the child. In terms of the data, the sales rep's name and employee number make up the master record and should print once for the associated product information retrieved by the Q_Product query.
   - Notice that WHERE already appears in the SQL Clause property. WHERE is the default clause used in master/detail relationships. You can replace WHERE with other SQL clauses such as HAVING and START WITH, but for this report the default is correct.
   - The other point to notice is that an equal sign (=) appears in the Condition property. An equality (i.e., table1.columnname = table2.columnname) is the default condition for master/detail relationships defined via a data link. You can replace the equal sign with any other supported conditional operator (to see what's supported, click on the field), but for this report the default is the proper condition.

   Linking the group G_ENAME and the query Q_Product via the EMPNO and REPID columns is analogous to writing both queries as the single-query shown below:

```
SELECT ENAME, EMPNO, REPID,
PRODNAME, AMOUNT, CUSTNAME
FROM EMP, SALES
WHERE JOB = 'SALESMAN'
AND EMPNO = REPID (+)
ORDER BY REPID, CUSTNAME
```

11.4 Use the Report Wizard to layout the data

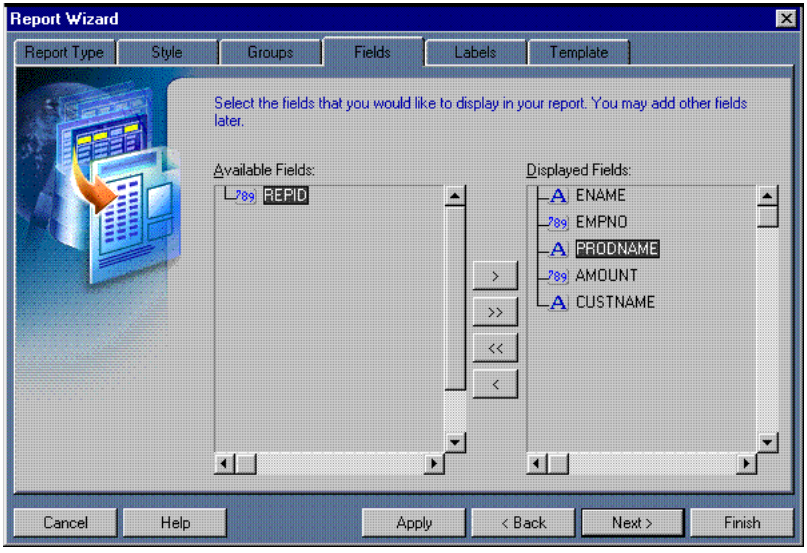
Once your data model is complete, you need to create a layout for the data objects to display in the report output. The Report Wizard enables you to create layouts for your data model.

> **Tip:** When you have multiple queries in your data model, make sure that you check the names of the groups associated with each query prior to entering the Report Wizard. The Report Wizard requires you to choose data for the layout by group name.

To create the layout:

1. In the Data Model view, right-click on the canvas, then choose **Report Wizard**.
2. In the Report Wizard, on the **Report Type** page, select **Create Paper Layout only**.
3. On the **Style** page, select **Group Above**.
4. On the **Groups** page, ensure that both groups from your data model appear in the **Displayed Groups** list.
5. On the **Fields** page:
   - Click the double right arrows (>>) to move all of the fields to the **Displayed Fields** list.
   - Click **REPID** in the **Displayed Fields** list and click the left arrow (<) to move it back to the **Available Fields** list. Since REPID and EMPNO represent the same value, you only need to display one of them. EMPNO is part of the master group, which is the level where we want to see its values in the report. REPID is part of the detail group. Hence, you remove REPID from the **Displayed Fields** list to prevent it from appearing in the output.

Figure 11–4 Fields page of the Report Wizard



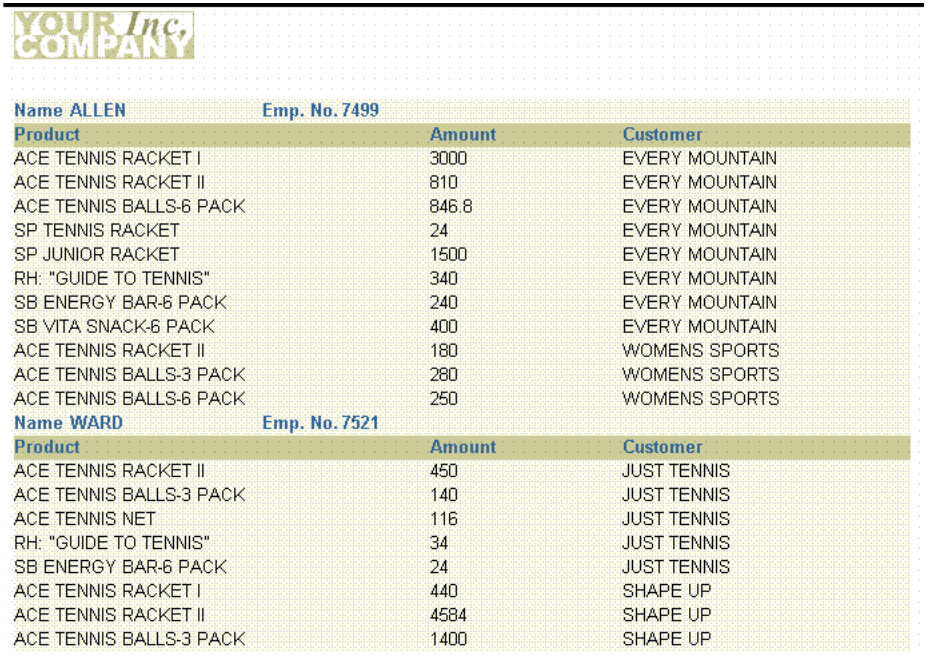
6. On the **Labels** page, change the labels and field widths as follows:

Table 11–2 Field description of Lables page

Fields	Labels	Width
PRODNAME	Product	22
AMOUNT	(no change)	10
CUSTNAME	Customer	15
ENAME	Name	(no change)
EMPNO	Emp. No.	(no change)

7. On the **Template** page, make sure **Beige** is selected under **Predefined Template**, then click Finish to display your report output in the Paper Design view. It should look like this:

Figure 11–5 Paper Design view for the two-query group report



Name ALLEN	Emp. No. 7499		
Product		Amount	Customer
ACE TENNIS RACKET I		3000	EVERY MOUNTAIN
ACE TENNIS RACKET II		810	EVERY MOUNTAIN
ACE TENNIS BALLS-6 PACK		846.8	EVERY MOUNTAIN
SP TENNIS RACKET		24	EVERY MOUNTAIN
SP JUNIOR RACKET		1500	EVERY MOUNTAIN
RH: "GUIDE TO TENNIS"		340	EVERY MOUNTAIN
SB ENERGY BAR-6 PACK		240	EVERY MOUNTAIN
SB VITA SNACK-6 PACK		400	EVERY MOUNTAIN
ACE TENNIS RACKET II		180	WOMENS SPORTS
ACE TENNIS BALLS-3 PACK		280	WOMENS SPORTS
ACE TENNIS BALLS-6 PACK		250	WOMENS SPORTS
Name WARD	Emp. No. 7521		
Product		Amount	Customer
ACE TENNIS RACKET II		450	JUST TENNIS
ACE TENNIS BALLS-3 PACK		140	JUST TENNIS
ACE TENNIS NET		116	JUST TENNIS
RH: "GUIDE TO TENNIS"		34	JUST TENNIS
SB ENERGY BAR-6 PACK		24	JUST TENNIS
ACE TENNIS RACKET I		440	SHAPE UP
ACE TENNIS RACKET II		4584	SHAPE UP
ACE TENNIS BALLS-3 PACK		1400	SHAPE UP

11.5 Format a field

In the Paper Design view, notice the Amount field. The values are neither aligned nor displayed as monetary amounts. You can quickly rectify this in the Paper Design view.

To assign a format mask to monetary values:

1. In the Paper Design view, select the first number value underneath the Amount label. Notice that all of the values are immediately selected, indicating that you can change their properties simultaneously.

   Tip: If you are familiar with format mask syntax, you could now right click on the field values, choose Property Inspector, and choose or manually enter a value for the Format Mask property.

2. Click the Currency button in the toolbar. A currency symbol immediately appears next to all of the values.
3. Click the Add Decimal Place tool twice. Two decimal places are added to the right of the decimal point.
4. Resize the field by clicking and dragging the rightmost handle of the field approximately a 0.5 inches to the left.
5. Click the Align Right tool. All of the values are immediately right aligned.
6. Select the Amount label.
7. Click the Align Right tool.
8. Save your report.

The final report output should look something like this:

Figure 11–6 Group above report output with monetary values formatted

Name ALLEN	**Emp. No. 7499**		
Product		**Amount**	**Customer**
ACE TENNIS RACKET I		$3000.00	EVERY MOUNTAIN
ACE TENNIS RACKET II		$810.00	EVERY MOUNTAIN
ACE TENNIS BALLS-6 PACK		$846.80	EVERY MOUNTAIN
SP TENNIS RACKET		$24.00	EVERY MOUNTAIN
SP JUNIOR RACKET		$1500.00	EVERY MOUNTAIN
RH: "GUIDE TO TENNIS"		$340.00	EVERY MOUNTAIN
SB ENERGY BAR-6 PACK		$240.00	EVERY MOUNTAIN
SB VITA SNACK-6 PACK		$400.00	EVERY MOUNTAIN
ACE TENNIS RACKET II		$180.00	WOMENS SPORTS
ACE TENNIS BALLS-3 PACK		$280.00	WOMENS SPORTS
ACE TENNIS BALLS-6 PACK		$250.00	WOMENS SPORTS
Name WARD	**Emp. No. 7521**		
Product		**Amount**	**Customer**
ACE TENNIS RACKET II		$450.00	JUST TENNIS
ACE TENNIS BALLS-3 PACK		$140.00	JUST TENNIS
ACE TENNIS NET		$116.00	JUST TENNIS
RH: "GUIDE TO TENNIS"		$34.00	JUST TENNIS
SB ENERGY BAR-6 PACK		$24.00	JUST TENNIS
ACE TENNIS RACKET I		$440.00	SHAPE UP
ACE TENNIS RACKET II		$4584.00	SHAPE UP
ACE TENNIS BALLS-3 PACK		$1400.00	SHAPE UP

11.6 Summary

Congratulations! You have successfully created the two-query, group report. You now know how to:

- create queries with a data link between them in the Data Model view.
- layout the data with the Report Wizard.
- format a field in the Paper Design view.

For more information on any of the wizards, views, or properties used in this example, refer to the *Reports Builder Online help*, which you can access in two ways:

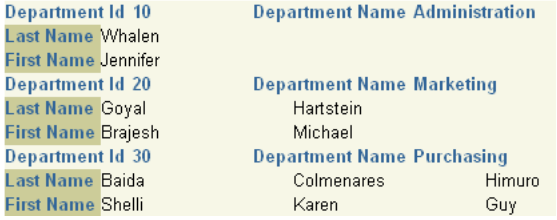
- From the Oracle Technology Network (`http://otn.oracle.com/products/reports/`), click **Documentation** and navigate to the *Reports Builder Online Help* for the most recent, hosted online help.
- From Reports Builder, choose **Help > Help Contents**.

> **Note:** You can replace your Reports Builder online help with the most recent update by downloading the latest online help set available on the Oracle Technology Network (http://otn.oracle.com/docs/products/reports/). Instructions for replacing your help file are included in the readme.txt in the download file.

12

Building an Across Group Report

Figure 12–1 Across group report output



Reports Builder enables you to modify the look of your report in multiple ways. In this example, you will build an across group report that prints the values of a database column across the page instead of down.

Concepts

In across reports with breaks, the master (or break) group prints "top to bottom"; i.e., as it would in other master/detail reports. However, the values in the detail group print across the page, from left to right. When there are more values than will fit on a line, Oracle Reports wraps the line and prints the remaining values across the page on the next line.

For more information on break reports, refer to the *Reports Builder online help* (choose **Index**, then type "break report" in the box).

Data Relationships

- The break in this report is created via a data link between a master group and a detail query.

Layout

- To create the layout used in this report, you'll select the master/detail style, then modify the Print Direction setting for one of the groups to ensure it prints across the page instead of down the page.

Example Scenario

This example uses two queries that select all of the columns displayed. You'll link them to establish a master/detail relationship. Oracle Reports creates all other necessary data objects by default.

To see a sample across report with control breaks, open the examples folder called `acrossbreak`, then open the Oracle Reports example report named `acrossbreak.rdf`. For details on how to open it, see "Accessing the example reports" in the Preface.

Table 12–1 Features demonstrated in this example

Feature	Location
Manually create a data model with two queries.	Section 12.2, "Create two queries"
Create the default layout using the Report Block Wizard.	Section 12.3, "Create the default layout"
Run your report to paper.	Section 12.4, "Run your report to paper"

12.1 Prerequisites for this example

To build the example in this chapter, you must have access to the sample schema provided with the Oracle9*i* database. If you don't know if you have access to this sample schema, contact your database administrator. You should have access to the "Human Resources" portion of the schema to complete this example.

12.2 Create two queries

When you create a report, you can either use the Report Wizard to assist you or create the report yourself. In this example, you will use the Data Model view to create your two queries, then use the tool palette to create a data link between the two queries to relate the data tables.

To create two queries:

1. Launch Reports Builder (or, if already open, choose **File > New > Report**)
2. In the Welcome or New Report dialog box, select **Build a new report manually**, then click **OK**.
3. In the Data Model view that displays, click the SQL Query tool in the tool palette.
4. In the SQL Query Statement box, type the following code:

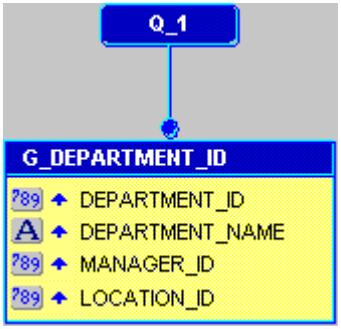
```
SELECT ALL DEPARTMENTS.DEPARTMENT_ID, DEPARTMENTS.DEPARTMENT_NAME,
DEPARTMENTS.MANAGER_ID, DEPARTMENTS.LOCATION_ID
FROM DEPARTMENTS
```

> **Note:** You can enter this query in any of the following ways:
>
> - Copy and paste the code from the provided text file called `acrossbreak_code.txt` into the **SQL Query Statement** field.
> - Click **Query Builder** to build the query without entering any code manually.
> - Type the code in the **SQL Query Statement** field.

5. Click **OK.**

   The data model for your new query displays:

Figure 12–2 Data Model for Query 1



6. Create another query, this time using the following code:

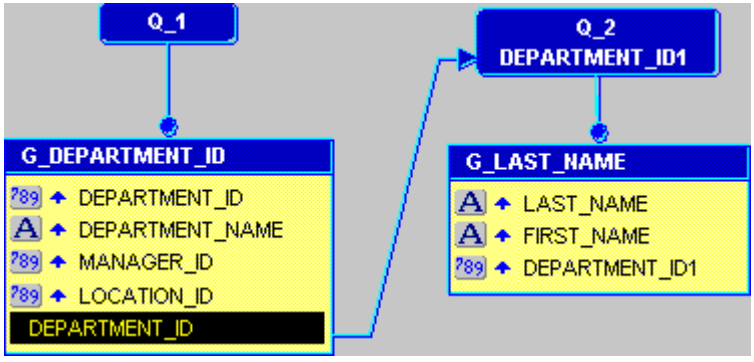
```
SELECT ALL EMPLOYEES.LAST_NAME, EMPLOYEES.FIRST_NAME, EMPLOYEES.DEPARTMENT_
ID
FROM EMPLOYEES
ORDER BY EMPLOYEES.LAST_NAME
```

> **Note:** You can also copy and paste the code from the text file provided in the **acrossbreak** folder, called `acrossbreak_code.txt`.

7. Click **OK**.
8. In the Data Model view, click the Data Link tool in the tool palette.
9. Click and drag your mouse from **DEPARTMENT_ID** column in Q_1, to **DEPARTMENT_ID1** in Q_2 to create a data link between the two queries.

   Your data model should now look like this:

Figure 12–3 Data Model of the two linked queries



Note: You can right-click the data link, then choose **Property Inspector** from the pop-up menu to ensure that the data link was created properly.

12.3 Create the default layout

The steps in this section show you how to use the Report Block Wizard to create the layout and choose how your data will display in your report. Here, you will choose the style of report you want to create, and choose to display the data across the report (hence creating the across group report).

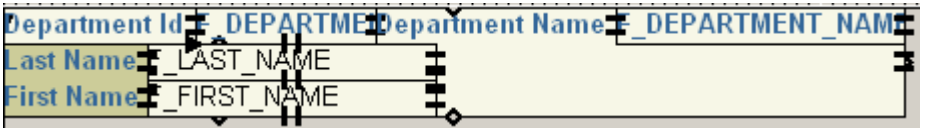
To create the default layout:

1. In the Object Navigator, double-click the Paper Layout node to display the Paper Layout view.
2. In the Paper Layout view, choose **Insert > Report Block** to display the Report Block Wizard.
3. In the Report Block Wizard, on the Style page, select **Group Above**, then click **Next**.
4. On the Groups page:
   - Click **G_LAST_NAME** in the **Available Groups** list, then click **Across** to specify the Print Direction and move this field to the **Displayed Groups** list.

- Click **G_DEPARTMENT_ID**, then click **Down**.
- Click **Next**.

5. On the Fields page, click the right arrow (>) to move the following fields from the **Available Fields** list to the **Displayed Fields** list, then click **Next**:
   - **DEPARTMENT_ID**
   - **DEPARTMENT_NAME**
   - **LAST_NAME**
   - **FIRST_NAME**
6. On the Labels page, click **Next**.
7. On the **Template** page, make sure **Beige** is selected under **Predefined Template**, then click Finish to display your report output in the Paper Design view. It should look like this:

Figure 12–4 Paper Layout view for the across group report

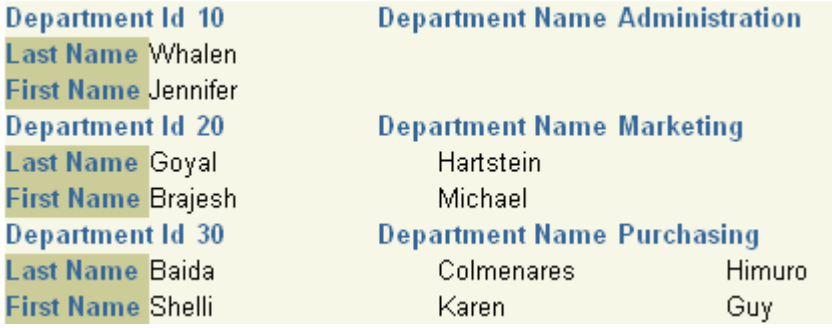


12.4 Run your report to paper

In this section, you will run your report to the Paper Design view so you can see how your report displays.

- Click the Paper Design button in the toolbar. Your report runs, then displays in the Paper Design view. It should look like the following:

Figure 12–5 Paper Design View of the Across Group Report



Save your report as `acrossbreak_<your initials>.rdf`.

12.5 Summary

Congratulations! You have successfully created a across group paper report. You now know how to:

- manually create a data model with two queries.
- create a master/detail report using the Report Block wizard.

For more information on any of the wizards, views, or properties used in this example, refer to the *Reports Builder online help*, which you can access in two ways:

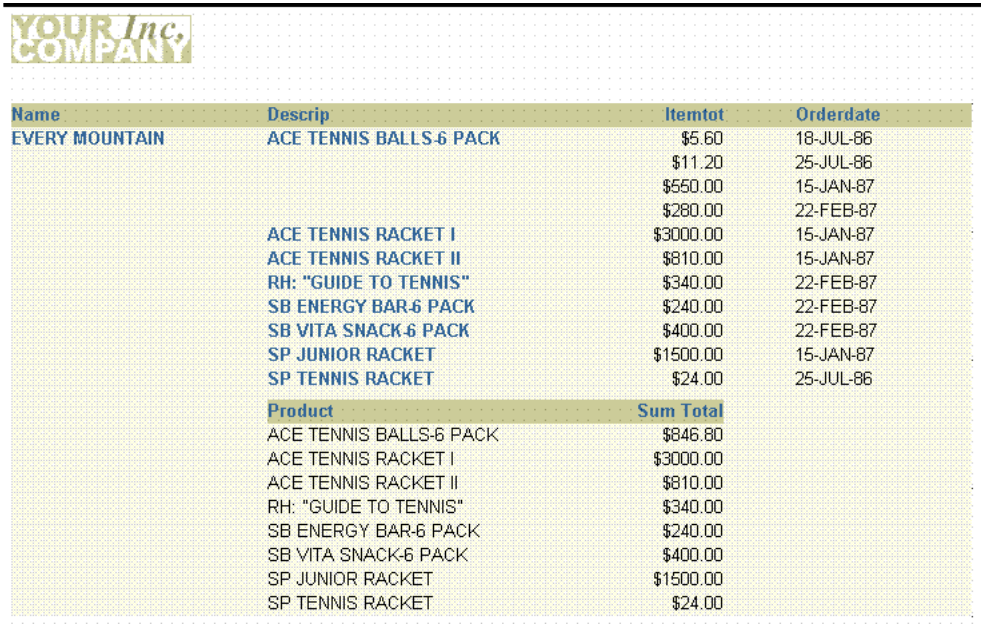
- From the Oracle Technology Network (`http://otn.oracle.com/products/reports/`), click **Documentation** and navigate to the *Reports Builder Online Help* for the most recent, hosted online help.
- From Reports Builder, choose **Help > Help Contents**.

> **Note:** You can replace your Reports Builder online help with the most recent update by downloading the latest online help set available on the Oracle Technology Network (http://otn.oracle.com/docs/products/reports/). Instructions for replacing your help file are included in the readme.txt in the download file.

13

Building a Group Left Summary Report

Figure 13–1 Group left summary report output



YOUR Inc. COMPANY

Name	Descrip	Itemtot	Orderdate
EVERY MOUNTAIN	ACE TENNIS BALLS-6 PACK	$5.60	18-JUL-86
		$11.20	25-JUL-86
		$550.00	15-JAN-87
		$280.00	22-FEB-87
	ACE TENNIS RACKET I	$3000.00	15-JAN-87
	ACE TENNIS RACKET II	$810.00	15-JAN-87
	RH: "GUIDE TO TENNIS"	$340.00	22-FEB-87
	SB ENERGY BAR-6 PACK	$240.00	22-FEB-87
	SB VITA SNACK-6 PACK	$400.00	22-FEB-87
	SP JUNIOR RACKET	$1500.00	15-JAN-87
	SP TENNIS RACKET	$24.00	25-JUL-86

Product	Sum Total
ACE TENNIS BALLS-6 PACK	$846.80
ACE TENNIS RACKET I	$3000.00
ACE TENNIS RACKET II	$810.00
RH: "GUIDE TO TENNIS"	$340.00
SB ENERGY BAR-6 PACK	$240.00
SB VITA SNACK-6 PACK	$400.00
SP JUNIOR RACKET	$1500.00
SP TENNIS RACKET	$24.00

This report consists of master records (Name, at the upper left of the figure above), detail records (Product, Itemtot, and Orderdate, to the upper right), and summary records (Product, and Sum Total). The summary calculates totals for the details under each master record. Notice that the column Product appears twice. With Reports Builder, you can display columns any number of times.

Concepts

- A master/detail summary report is a master/detail report that also contains one or more summaries.
- This report will use two queries to select data from four tables. The master query will select the customer name, while the detail query will select the information associated with the products ordered by each customer.
- Because the detail query will select data from several tables, you'll need to specify joins to link the information in the tables together.
- You'll drag a column out of the detail group to further group the data. You'll also create a summary column to calculate the item totals.
- The layout for this report is constructed in two parts. The top portion contains the master and detail information, and is created using the Report Wizard. The second portion is constructed by hand in the Paper Layout view and formats the summary.
- You'll also resize two groups in the Layout editor to ensure that the new, user-created bottom portion of the layout is integrated into the top portion, omit some columns from the layout that were queried only to join the tables, and include more space between instances of a repeating frame.

Example Scenario

Suppose that you want to create a report that displays and summarizes sales data by customer. This report would include the following for each customer:

- a list of the products they purchased by order date and how much they spent on each product
- a summary for each customer that shows how much they spent in total on each product over time

To see a sample master/detail summary report, open the examples folder named `masterdetailsummary`, then open the Oracle Reports example called `masdtmb.rdf`. For details on how to access it, see "Accessing the example reports" in the Preface.

Table 13–1 Features demonstrated in this example

Feature	Location
Create a new, empty report	Section 13.2, "Create a new report manually"

Table 13–1 Features demonstrated in this example

Feature	Location
Create two queries with a data link between them	Section 13.3, "Create a data model with a data link"
Create two separate layouts	Section 13.4, "Use the Paper Layout view to create two layouts"
Combine the separate layouts into one	Section 13.5, "Merge the two layouts"
Format monetary values	Section 13.6, "Format fields"

13.1 Prerequisites for this example

To build the example in this chapter, you must have access to the Summit Sporting Goods schema, which we've provided on the Oracle Technology Network (`http://otn.oracle.com/products/reports/`). To download the SQL scripts that install the schema, go to the Documentation page on OTN and follow the instructions provided on the Web page.

13.2 Create a new report manually

In this case, it is easier to create the data model and layout separately. Hence, we will create an empty report first, then add the queries, and then create the layouts.

To create a blank report:

1. Launch Reports Builder (or, if already open, choose **File > New > Report**)
2. In the Welcome or New Report dialog box, select **Build a new report manually**, then click **OK**.

13.3 Create a data model with a data link

When you create a report with multiple queries, it is typically easier to create all of the queries with the Data Wizard first and then create the layouts with the Report Wizard.

To create the queries:

1. In the Data Model view, choose **Insert > Query** to display the Data Wizard.
2. If the Welcome page displays, click **Next**.

3. On the Query page, type `Q_Customer` for the **Query name**, then click **Next**.
4. On the Data Source page, select **SQL Query**, then click **Next**.
5. On the Data page, in the **Data Source definition** field, enter the following SELECT statement:

```
SELECT CUSTID, NAME
FROM CUSTOMER
ORDER BY NAME
```

> **Note:** You can enter this query in any of the following ways:
>
> - Copy and paste the code from the provided text file called `masdtmb_code.txt` into the **Data Source definition** field.
> - Click **Query Builder** to build the query without entering any code manually.
> - Type the code in the **Data Source definition** field.

6. Click **Next**.

> **Note:** If you are not already connected to a database, you will be prompted to connect to the database when you click **Next**. Ensure that you connect to a database that has the appropriate schema for this example. Section 13.1, "Prerequisites for this example" describes the sample schema requirements for this example.

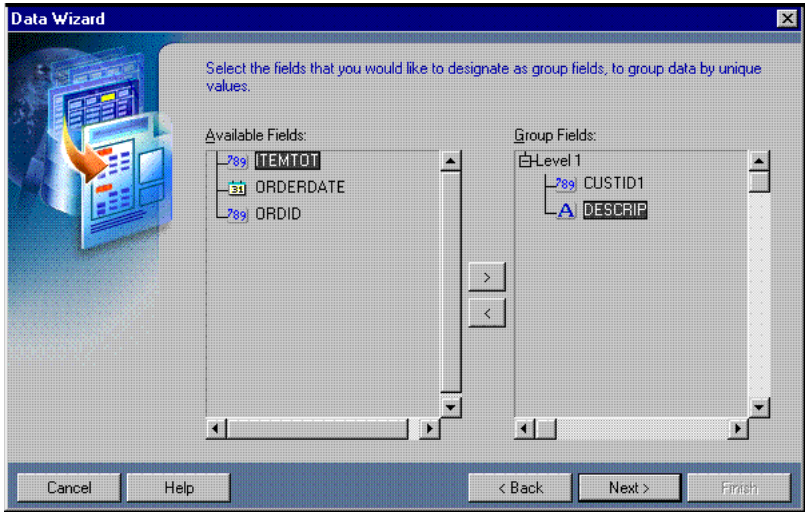
7. On the Groups page, click **Next**.
8. Click **Finish** to display your first query in the Data Model view.
9. Repeat the steps above for a second query, but this time name your query `Q_Item` and use the following SELECT statement:

```
SELECT CUSTID, DESCRIP, ITEMTOT, ORDERDATE,
ITEM.ORDID
FROM ORD, PRODUCT, ITEM
WHERE ITEM.ORDID = ORD.ORDID
```

```
AND ITEM.PRODID = PRODUCT.PRODID
ORDER BY CUSTID, DESCRIP, ORDERDATE
```

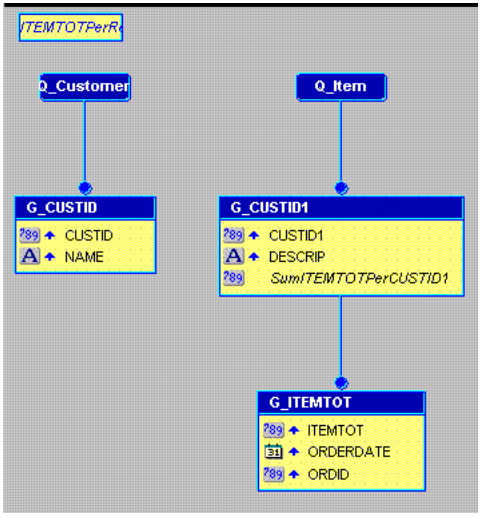
10. On the Groups page of the Data Wizard:
    - Click **CUSTID1** and click the right arrow (>) to move this field to the **Group Fields** list.
    - Do the same for **DESCRIP**.

Figure 13–2 Groups page of the Data Wizard



11. Click **Next**.
12. On the Totals page, click **ITEMTOT** and click **Sum**.
13. Click **Finish** to display the data model for your report in the Data Model view. It should look something like this:

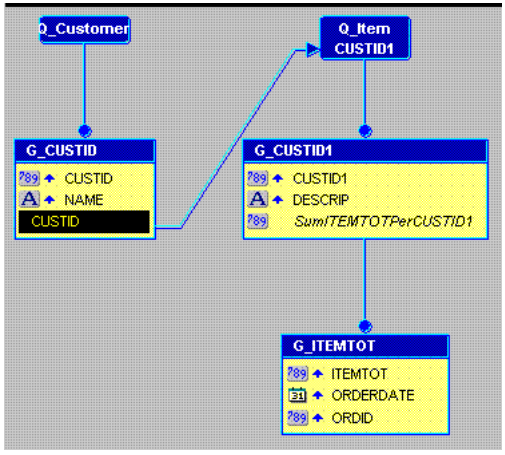
Figure 13–3 Two-query data model with summaries



To add the data link:

1. In the Data Model view, click the Data Link tool in the tool palette.
2. Click and drag from the **CUSTID** column in the **G_CUSTID** group to the **CUSTID1** column in the **G_CUSTID1** group. Notice that a line is drawn from the bottom of the **G_CUSTID** group to the **Q_Item** query. Labels for **CUSTID** and **CUSTID1** are created at each end of the line to indicate they are the columns linking **G_CUSTID** to **Q_Item**.

Figure 13–4 Two-query data model with a data link



3. Double-click the new data link line to display the Property Inspector and examine the property settings:

   - G_CUSTID is identified as the parent, while Q_Item is listed as the child. In terms of the data, the customer's identifier and name make up the master record and should print once for the associated item order information retrieved by the Q_Item query.
   - Notice that WHERE already appears in the SQL Clause property. WHERE is the default clause used in master/detail relationships. You can replace WHERE with other SQL clauses such as HAVING and START WITH, but for this report the default is correct.
   - The other point to notice is that an equal sign (=) appears in the Condition property. An equality (i.e., table1.columnname = table2.columnname) is the default condition for master/detail relationships defined via a data link. You can replace the equal sign with any other supported conditional operator (to see what's supported, click on the field), but for this report the default is the proper condition.

13.4 Use the Paper Layout view to create two layouts

Once your data model is complete, you need to create a layout for the data objects to display in the report output. This particular report consists of two separate layouts:

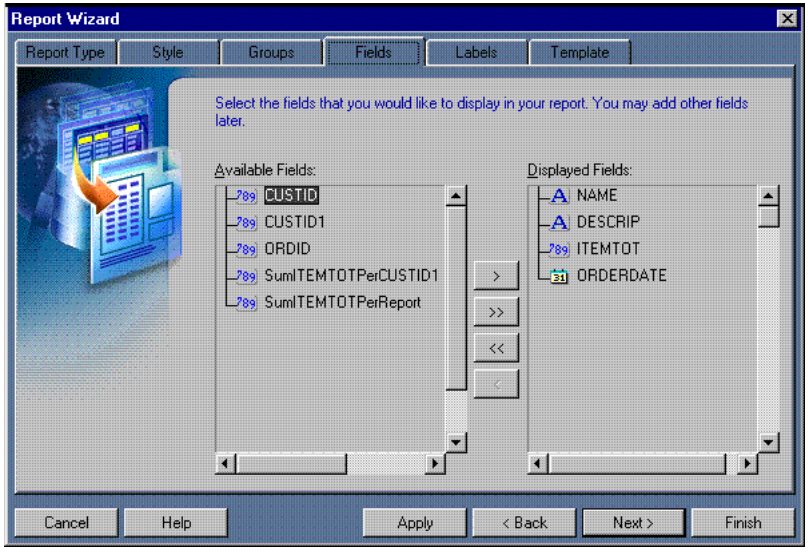
- a group left layout for listing customer purchases individually
- a tabular layout that shows a summary of the customer's purchases by product

Given that two layouts are required, you need to create the first layout through the Report Wizard and the second by inserting a report block. The reason for taking this approach is that the Report Wizard overwrites everything in the layout. Hence, you can only create the first layout through the Report Wizard. Additional layouts must be created by inserting a report block.

To create the first layout:

1. Choose **Tools > Report Wizard**.
2. On the **Report Type** page, select **Create Paper Layout only**.
3. On the **Style** page, select **Group Left**.
4. On the **Groups** page, ensure that all of the groups from your data model appear in the **Displayed Groups** list.
5. On the **Fields** page, ensure that only the following columns appear in the **Displayed Fields** list:
   - **NAME**
   - **DESCRIP**
   - **ITEMTOT**
   - **ORDERDATE**

Figure 13–5 Fields page of Report Wizard



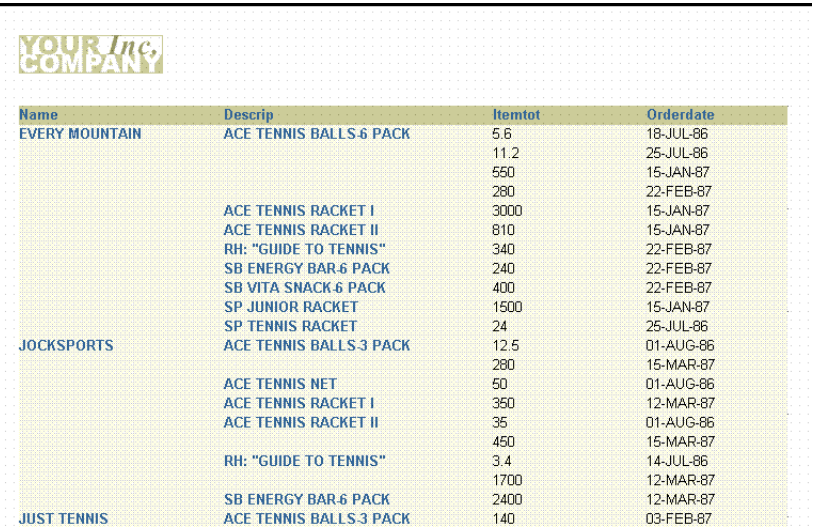
6. On the **Labels** page, change the labels and field widths as follows:

Table 13–2 Field Description Labels Page

Fields	Labels	Width
DESCRIP	Product	20
NAME	(no change)	15

7. On the **Template** page, make sure **Beige** is selected under **Predefined Template**, then click Finish to display your report output in the Paper Design view. It should look like this:

Figure 13–6 Paper Design view for the group left layout with two group columns



To create the second layout:

1. Click the Paper Layout button in the toolbar to display the Paper Layout view.
2. In the Paper Layout view, click the Report Block tool in the tool palette.
3. Starting about 0.5 inches below the existing layout, click and drag a box about 2 inches tall and 4 inches wide. Release your mouse button to display the Report Block Wizard.
4. In the Report Block Wizard, on the Style page, select **Tabular**, then click **Next**.
5. On the Groups page, click **G_CUSTID1** and click **Down** to specify the Print Direction and move this group to the **Displayed Groups** list, then click **Next**. (**G_CUSTID1** should be the only group in the **Displayed Groups** list when you are done.)
6. On the Fields page, click the following fields and click the right arrow (**>**) to move them to the **Displayed Fields** list, then click **Next**:
   - **DESCRIP**
   - **SumITEMTOTPerCUSTID1**.

   DESCRIP and **SumITEMTOTPerCUSTID1** should be the only fields in the **Displayed Fields** list when you are done.

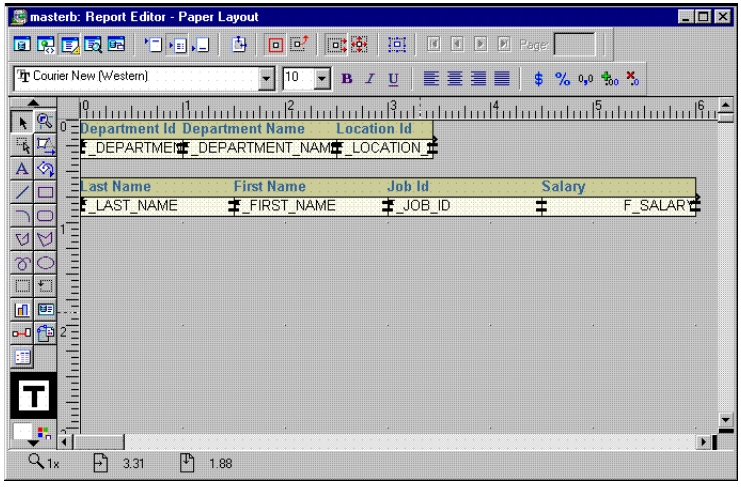
7. On the Labels page, change the labels as follows, then click **Next**:

Table 13–3 Field Description

Fields	Labels
DESCRIP	`Product`
SumITEMTOTPerCUSTID1	`Sum Total`

8. On the **Template** page, make sure **Beige** is selected under **Predefined Template**, then click Finish to display your report output in the Paper Design view. It should look like this:

Figure 13–7 Paper Layout view with two layouts



13.5 Merge the two layouts

After you have created the two layouts, you have only achieved part of your desired result. The summary of purchases by product is outside of the master/detail layout. Hence it summarizes the product purchases for the entire report rather than the product purchases for each customer. To show the summary for each customer, you must move the second layout inside of the first one.

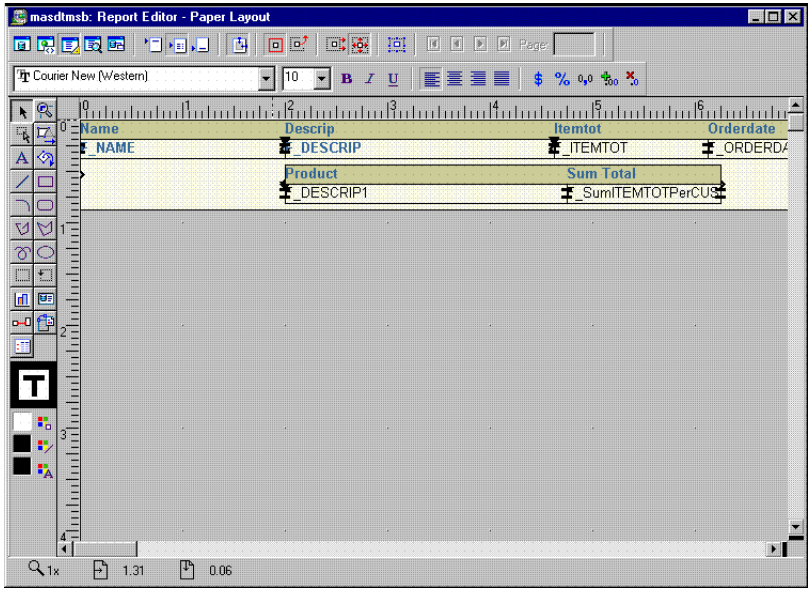
To merge the second layout with the first:

1. In the Object Navigator, select **M_G_CUSTID1_GRPFR1**.

   Tip: To make finding this frame easier, just type in the name in the Find field at the top of the Object Navigator.

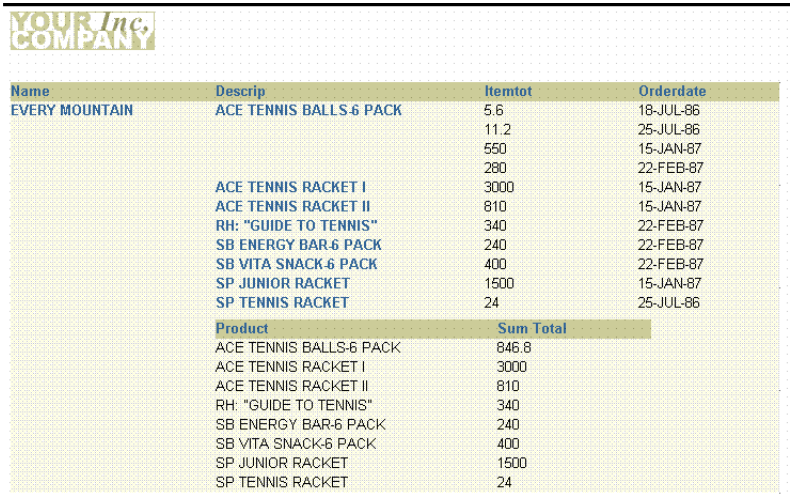
2. Click the title bar of the Report Editor to return to the Paper Layout view. Notice that the frame around the second layout is now selected for you.
3. Use the arrow keys to position the second layout so that the DESCRIP1 field lines up with the DESCRIP field in the first layout.
4. Click the Confine Off button in the toolbar. Ensure that Flex mode is on (it is on by default).
5. In the Object Navigator, select R_G_CUSTID.
6. Click the title bar of the Report Editor to return to the Paper Layout view. Notice that the master repeating frame in the first layout is now selected for you.
7. Click on the handle at the bottom center of R_G_CUSTID and drag it down about a half inch. Because Flex Mode is on, the frame surrounding it grows as you drag. Similarly, the second layout is moved down to avoid being overwritten.
8. With R_G_CUSTID still selected, choose **Tools > Property Inspector**.
9. Change the Vertical Spacing Between Frames property to 0.25.
10. In the Object Navigator, select M_G_CUSTID1_GRPFR1.
11. Click the title bar of the Report Editor to return to the Paper Layout view.
12. Using the tool bar along the top of the Paper Layout view, turn Flex Mode off.
13. Using the arrow keys, move M_G_CUSTID1_GRPFR1 and its contents inside of the first layout.

Figure 13–8 Paper Layout view with two layouts merged into one



14. Click the Paper Design button in the toolbar of the Report Editor to display the Paper Layout view. Notice how the summary table now repeats for each customer.

Figure 13–9 Paper Design view with two layouts merged



13.6 Format fields

In the Paper Design view, notice the monetary values are neither aligned nor displayed as monetary amounts. You can quickly rectify this in the Paper Design view.

To assign a format mask to monetary values:

1. In the Paper Design view, Shift-click on the values underneath the Itemtot and Sum Total. Notice that all of the values are immediately selected, indicating that you can change their properties simultaneously.
2. Click the Currency button in the toolbar. A currency symbol immediately appears next to all of the values.
3. Click the Add Decimal Place button twice. Two decimal places are added to the right of the decimal point.
4. Resize the fields. Click and drag the rightmost handle of the Itemtot field approximately 0.5 inches to the left. Repeat for the Sum Total field. Try to have the right boundaries of the two fields align with each other.
5. Click the Align Right button. All of the values are immediately right aligned.
6. Shift-click on the Itemtot label and the Sum Total label.

7. Click the Align Right button.
8. Save your report.

Figure 13–10 Combined group left and tabular report output

13.7 Summary

Congratulations! You have successfully created a group left summary report. You now know how to:

- create two queries with a data link between them.
- create two separate layouts.
- combine the separate layouts into one.
- format monetary values.

For more information on any of the wizards, views, or properties used in this example, refer to the *Reports Builder online help*, which you can access in two ways:

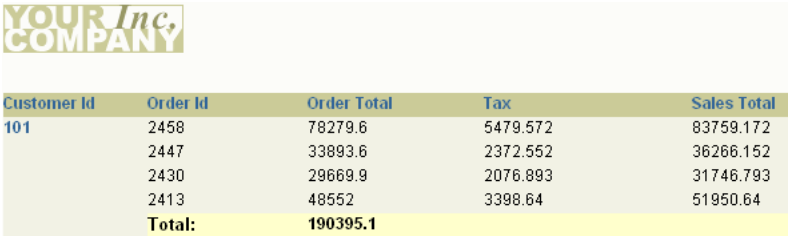
- From the Oracle Technology Network (`http://otn.oracle.com/products/reports/`), click **Documentation** and navigate to the *Reports Builder Online Help* for the most recent, hosted online help.
- From Reports Builder, choose **Help > Help Contents**.

Note: You can replace your Reports Builder online help with the most recent update by downloading the latest online help set available on the Oracle Technology Network (http://otn.oracle.com/docs/products/reports/). Instructions for replacing your help file are included in the readme.txt in the download file.

14

Building a Group Left Formula Report

Figure 14–1 Formula report output



Customer Id	Order Id	Order Total	Tax	Sales Total
101	2458	78279.6	5479.572	83759.172
	2447	33893.6	2372.552	36266.152
	2430	29669.9	2076.893	31746.793
	2413	48552	3398.64	51950.64
	Total:	**190395.1**		

A formula column, like a summary column, is a computational column you create yourself. Unlike a summary, its values are calculated based on a PL/SQL formula you provide. The formula may use data from another column in the report, but is not required to do so.

About formula columns

A formula column performs a user-defined computation on another column(s) data, including placeholder columns. Formula columns should not be used to set values for parameters.

Concepts

A formula column contains at least one column whose value or values are computed using a PL/SQL formula. Formula columns are similar in usage to summary columns.

For more information on formula columns, refer to the *Reports Builder online help* (choose **Index**, then type "formula column" in the box).

Data Relationships

- To create a formula report, create a query and select your data. Next, create additional columns and add them to groups in your report. Specify their formulas in the column property sheets.
- As with summaries, you do not select formulas from the database. unlike summaries, which use packaged computations shipped with Oracle Reports, formula columns use formulas you provide by referencing PL/SQL functions. These formulas can be any legal PL/SQL constructs, which allows a great deal of flexibility in the formulas you use.
- A formula performs computations using data from a single record which can span multiple columns. This is in contrast to a summary, which summarizes the data from multiple records in a single column.

Layout

- This report uses the default group left format with no modifications.

Example Scenario

In this example, you will use the Report Wizard to set up your report and write the one query that selects all the necessary database columns. You will then manually create the two formula columns to calculate tax and order totals for each customer, then add the formula columns to your report. You will use a Group Left style report to make the data in the report easy to read.

To see a sample formula report, open the examples folder called `formula`, then open the Oracle Reports example report named `formula.rdf`. For details on how to open it, see "Accessing the example reports" in the Preface.

Table 14–1 Features demonstrated in this example

Feature	Location
Use the Report Wizard to create a report with a paper layout.	Section 14.2, "Use the Report Wizard to create a simple report"
Create two formula columns for the report.	Section 14.3, "Create two formula columns"

14.1 Prerequisites for this example

To build the example in this chapter, you must have access to the sample schema provided with the Oracle9*i* database. If you don't have access to this sample

schema, contact your database administrator. You should have access to the "Order Entry" portion of the schema to complete this example.

14.2 Use the Report Wizard to create a simple report

When you create a report, you can either use the Report Wizard to assist you or create the report yourself. To build the simple report in this example, you can use the Report Wizard. Using the wizard enables you to define the layout for the report, as well as set the data definition.

To create a simple report:

1. Launch Reports Builder (or, if already open, choose **File > New > Report**)
2. In the Welcome or New Report dialog box, select **Use the Report Wizard**, then click **OK**.
3. If the Welcome page displays, click **Next**.
4. On the Report Type page, select **Create Paper Layout Only**, then click **Next**.
5. On the Title page, select **Group Left**, then click **Next**.
6. On the Data Source page, click **SQL Query**, then click **Next**.
7. On the Data page, click **Query Builder**.

> **Note:** Make sure you're connected to the Order Entry portion of the sample schema shipped with the Oracle9*i* database. If you are already connected to another database, click **Connect** to connect to the sample schema. When the Connection dialog box displays, enter the connection string for the Order Entry portion of the sample schema provided by your database administrator.

8. In the Select Data Tables dialog box, click **CUSTOMERS**, then click **Include**.
9. Click **ORDERS**, then click **Include**.
10. Click **Close**.
11. In Query Builder, in the **CUSTOMERS A1** table, select the checkboxes next to the following column:
    - **CUSTOMER ID**

12. In the **ORDERS** table, select the checkboxes next to the following columns:
    - **ORDER ID**
    - **ORDER TOTAL**
13. Click **OK**.
14. In the Report Wizard, you should see the following code in the **Data Source definition** field:

```
SELECT ALL CUSTOMERS_A1.CUSTOMER_ID, ORDERS.ORDER_ID, ORDERS.ORDER_TOTAL
FROM CUSTOMERS CUSTOMERS_A1, ORDERS
WHERE (ORDERS.CUSTOMER_ID = CUSTOMERS_A1.CUSTOMER_ID)
```

15. At the end of the code, type the following line:

```
ORDER BY CUSTOMERS_A1.CUSTOMER_ID
```

16. Now, your code should look like this:

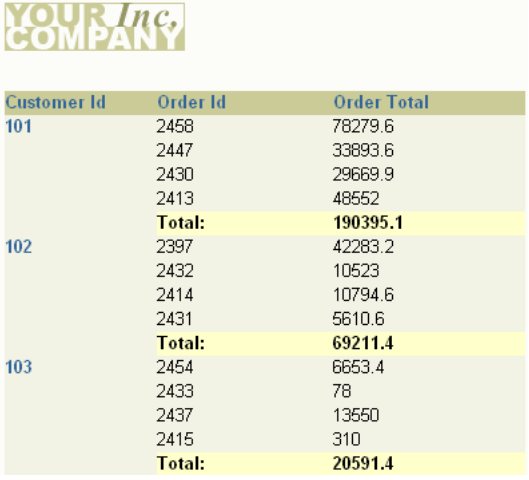
```
SELECT ALL CUSTOMERS_A1.CUSTOMER_ID, ORDERS.ORDER_ID, ORDERS.ORDER_TOTAL
FROM CUSTOMERS CUSTOMERS_A1, ORDERS
WHERE (ORDERS.CUSTOMER_ID = CUSTOMERS_A1.CUSTOMER_ID)
ORDER BY CUSTOMERS_A1.CUSTOMER_ID
```

> **Note:** Although we've shown you how to use the Query Builder to build this query, you can enter this query in any of the following ways:
>
> - Copy and paste the code from the provided text file called `formula_code.txt` into the **Data Source definition** field.
> - Click **Query Builder** to build the query without entering any code manually, as described in the steps above.
> - Type the code in the **Data Source definition** field.

17. Click **Next**.
18. On the Groups page, select **CUSTOMER_ID** and click the right arrow (>) to move this field to the **Groups Fields** list, then click **Next**.

19. On the Fields page, click the double right arrow button (>>) to move all of the fields to the **Displayed Fields** list, then click **Next**.
20. On the Totals page, click **ORDER_TOTAL** and click **Sum**, then click **Next**.
21. On the Template page, select **Predefined Template** and click **Beige**, then click **Finish** to display your report output in the Paper Design view. It should look something like this:

Figure 14–2 Paper Design view for the formula report



22. Save the report as `formulareport_<your initials>.rdf`.

14.3 Create two formula columns

Frequently, you want to base calculations on values in your data source. One way you can do this is by using formula columns. The steps in this section will show you how to create two formula columns that calculate the following values:

- the tax on each order
- the grand total for each customer, including tax

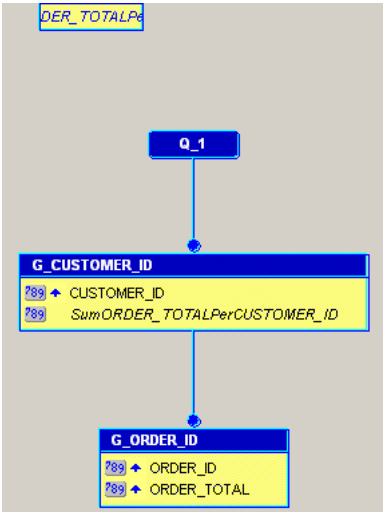
14.3.1 Create a formula column to calculate the tax

To create a formula column that calculates the tax:

1. In Reports Builder, click the Data Model button in the toolbar to display the Data Model view of your report.

   The data model should look something like this:

Figure 14–3 Data Model of the Formula Report



2. Click the Formula Column tool in the tool palette, then click in the **G_ORDER_ID** group under **ORDER_TOTAL** to create a formula column.

   Tip: To view the names of the tools in the tool palette and the toolbar, drag and hold your mouse over each icon and hint text will display describing the icon.

3. Double-click the new formula column object (**CF_1**) to display the Property Inspector, and set properties:

   - Under **General Information**, set the Name property to Tax.

- Under **Placeholder/Formula**, click the PL/SQL Formula property field to display the PL/SQL Editor.

4. In the PL/SQL Editor, use the template to enter the following PL/SQL code:

```
function TAXFormula return Number is
tax number;
begin
  tax := :ORDER_TOTAL * .07;
  return (tax);
end;
```

5. Click **Compile**.
6. If no errors display, click **Close**. If errors display, verify that your code looks exactly like the above code, paying close attention to the column names.
7. Close the Property Inspector.

You have created a formula column that calculates the tax (7%) of each order.

14.3.2 Create a formula column that calculates customer order totals

To create a formula column that calculates customer order totals:

1. Repeat Steps 2 and 3 from Section 14.3.1, "Create a formula column to calculate the tax": click the Formula Column tool in the tool palette, then click in the G_ORDER_ID group under TAX. Then, double-click CF_1 to display the Property Inspector.
2. Change the name of the column to `SALES_TOTAL`.
3. Open the Program Unit Editor and modify your formula so that it looks like this:

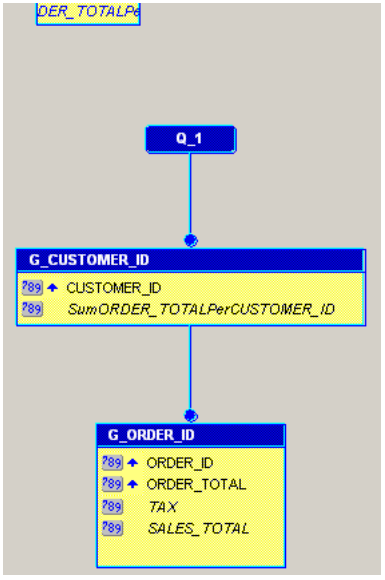
```
function SALES_TOTALFormula return Number is
sales_total number;
begin
  sales_total := :ORDER_TOTAL + :TAX;
  return (sales_total);
end;
```

4. Click **Compile**.

5. If no errors display, click **Close**. If errors display, verify that your code looks exactly like the above code, paying close attention to the column names.
6. Close the Property Inspector.

You have created a formula column that calculates the total orders of each customer. Your data model should now look something like this:

Figure 14–4 Data Model with Formula Columns



14.3.3 Add the formula columns to the report layout

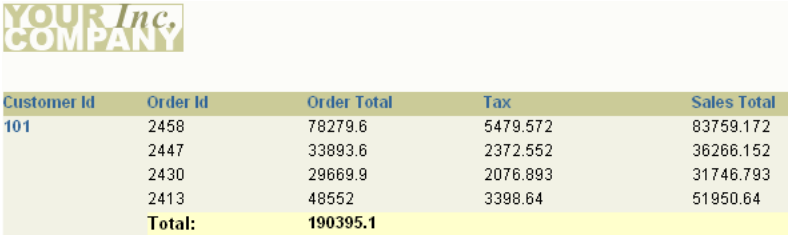
Now that you've created your formula columns, you must add them to your report layout. The easiest way to do this is to return to the Report Wizard.

To add formula columns to your report layout:

1. In the Data Model view, right-click on the canvas, then choose **Report Wizard**.
2. In the Report Wizard, click the **Fields** tab. In the **Available Fields** list, you should now see your two new formula columns. Click each one, then click the right arrow (>) to move them to the **Displayed Fields** list.

3. Click **Finish** to display your report output in the Paper Design view. It should look something like this:

Figure 14–5 Final formula report output



14.4 Summary

Congratulations! You have successfully created a formula paper report. You now know how to:

- define a report layout using the Report Wizard.
- create two formula columns and add them to your report.

For more information on any of the wizards, views, or properties used in this example, refer to the *Reports Builder online help*, which you can access in two ways:

- From the Oracle Technology Network (`http://otn.oracle.com/products/reports/`), click **Documentation** and navigate to the *Reports Builder Online Help* for the most recent, hosted online help.
- From Reports Builder, choose **Help > Help Contents**.

> **Note:** You can replace your Reports Builder online help with the most recent update by downloading the latest online help set available on the Oracle Technology Network (http://otn.oracle.com/docs/products/reports/). Instructions for replacing your help file are included in the readme.txt in the download file.

Part III

Building Reports with Special Formatting

The chapters in this Part provide steps to build reports that include special formatting. The example reports show you how to wrap text data, add headers and footers, format headers with database values that change at runtime, renumber pages, include intermixed fields, suppress labels, use conditional text, and add text, color, or graphics to your report.

This part contains the following chapters:

- Chapter 15, "Building a Wrapped Field Report"

  A wrapped field report wraps the data on word boundaries if it is too long to fit in one line.The report contains a field that is of a fixed horizontal width but can expand vertically if the contents of the break field are longer than the specified width. The contents of the fields are not truncated.

- Chapter 16, "Building a Header and Footer Report"

  A header and footer report contains boilerplate or fields in the header or footer section. This report has a page header printed in the upper margin area of every page of the report and a footer printed at the end.

- Chapter 17, "Building a Header with Database Values Report"

  A header with database values report can include dynamic values in the page header of the report. For example, you can print both the first and last department numbers found on each page.

- Chapter 18, "Building a Report with Graphics, Text, and Color"

  A report with graphics, text, and color enables you to modify the look of your report. You can enhance a report by adding an image to the margin, adding a title or a border, or applying different fonts and text styles.

- Chapter 19, "Building a Report that Renumbers Pages by Repeating Frame"

  A report that renumbers pages by repeating frame displays page numbers using the format "Page X of Y Pages". The first number (X) corresponds to the current page of each parent record. The second number (Y) corresponds to the total number of pages of each parent record.

- Chapter 20, "Building an Intermixed Fields Report"

  An intermixed fields report includes the group field between its related fields. This is in contrast to a group report. Normally in the group report, the break field appears to the left of its related fields in the group left report or above in the group above report.

- Chapter 21, "Building a Report that Suppresses Labels"

  A report that suppresses labels is a master/detail report that fetches a master record with no associated details. Therefore, the report suppresses the detail information for a single record but allows the other master/detail records to display.

- Chapter 22, "Building a Conditional Form Letter Report"

  A conditional form letter report generates two different form letters from one report. Since the two letters share common features, it is more convenient to create a base form letter and then apply conditions to certain parts. The conditions will determine whether the part should display for the current record.

- Chapter 23, "Building a Report with Conditional Highlighting"

  Conditional highlighting draws attention to specific data in a report by using visual formatting. You can use Reports Builder or the PL/SQL Editor to create a format trigger that changes the appearance of retrieved data depending on factors you define. For example, you can display amounts greater than 1000 in red.

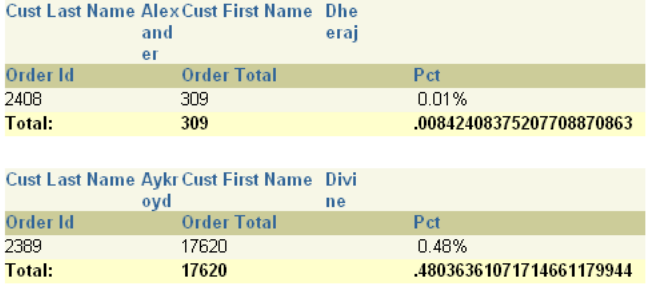
- Chapter 24, "Building a Report with Dynamic Graphics"

  A report with dynamic graphics contains drawings and images that are "dynamic." Any changes that you make to the graphics will be reflected in your report output at runtime.

15

Building a Wrapped Field Report

Figure 15–1 Wrapped field report output



Cust Last Name	Alexander	Cust First Name	Dheeraj	
Order Id		Order Total		Pct
2408		309		0.01%
Total:		**309**		**.008424083752077088708863**

Cust Last Name	Aykroyd	Cust First Name	Divine	
Order Id		Order Total		Pct
2389		17620		0.48%
Total:		**17620**		**.48036361071714661179944**

Reports Builder enables you to modify the look of your report in multiple ways. In this example, you will build a break report where the line wraps on word boundaries if it is too long to fit on one line.

Concepts

This report contains a field which is of a fixed horizontal width, but can expand vertically if the contents of the break field are longer than the specified width. The field's contents are not truncated; rather, the contents will wrap within the specified width of the field, and the field will expand vertically.

For more information on break reports, refer to the *Reports Builder online help* (choose **Index**, then type "break report" in the box).

Data Relationships

- This is a simple, one-query break report. In addition, you'll add three summary columns.

Layout

- This report is formatted as a master/detail report. You'll modify the layout to add space between sets of information, then change the width of a field so that the break field will wrap.
- Optional: You'll change the format masks of three fields. You'll also use Page Protect in this report. Page Protect causes all objects within a frame or repeating frame to remain together. By specifying Page Protect for the master repeating frame, the customer name and all of its related records will always appear on the same page.

Example Scenario

This example is organized like a simple break report: one query and two groups, one of which is user-created. In addition, you'll create three summary columns.

To see a sample wrapped field report, open the examples folder called `wrappedbreak`, then open the Oracle Reports example report named `wrappedbreak.rdf`. For details on how to open it, see "Accessing the example reports" in the Preface.

Table 15–1 Features demonstrated in this example

Feature	Location
Manually create a query.	Section 15.2, "Create a query in the Data Model view"
Create three summary columns to calculate various totals and percentages for your report.	Section 15.3, "Create three summary columns"
Create the layout for your paper report.	Section 15.4, "Create the default layout using the Report Wizard"

15.1 Prequisites for this example

To build the example in this chapter, you must have access to the sample schema provided with the Oracle9*i* database. If you don't know if you have access to this sample schema, contact your database administrator. You should have access to the "Order Entry" portion of the schema to complete this example.

15.2 Create a query in the Data Model view

When you create a report, you can either use the Report Wizard to assist you or create the report yourself. In this example, you will use the Data Model view to create your two queries, then use the tool palette to create a data link between the two queries to relate the data tables.

To create the query:

1. Launch Reports Builder (or, if already open, choose **File > New > Report**)
2. In the Welcome or New Report dialog box, select **Build a new report manually**, then click **OK**.
3. In the Data Model view that displays, click the SQL Query tool in the tool palette then click in an open area of the Data Model view to display the SQL Query Statement dialog box.
4. In the SQL Query Statement dialog box, enter the following SELECT statement:

```
SELECT ALL CUSTOMERS_A1.CUST_FIRST_NAME,
CUSTOMERS_A1.CUST_LAST_NAME, ORDERS.ORDER_ID, ORDERS.ORDER_TOTAL
FROM CUSTOMERS CUSTOMERS_A1, ORDERS
WHERE (ORDERS.CUSTOMER_ID = CUSTOMERS_A1.CUSTOMER_ID)
ORDER BY CUSTOMERS_A1.CUST_LAST_NAME
```

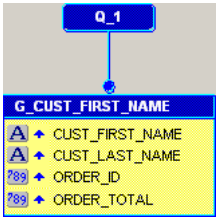
Note: You can enter this query in any of the following ways:

- Copy and paste the code from the provided text file called `wrappedbreak_code.txt` into the **SQL Query Statement** field.
- Click **Query Builder** to build the query without entering any code manually.
- Type the code in the **SQL Query Statement** field.

---

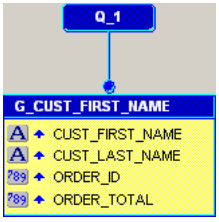
5. Click **OK** to display the data model for your new query in the Data Model view. It should look like this:

Figure 15–2 Data Model for the query



6. In the **G_CUST_FIRST_NAME** group, click **CUST_FIRST_NAME** and drag it above the group to create another group.
7. Click **CUST_LAST_NAME**, then drag it into the new group, so that the data model now looks like this:

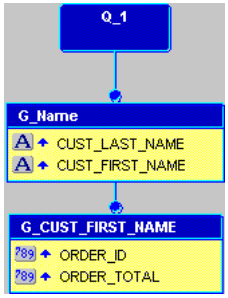
Figure 15–3 Data model with groups



8. Double-click the new group (**G_1**) the display the Property Inspector, and set properties:
   - Under **General Information**, set the Name property to G_Name.

   Your data model should now look like this:

Figure 15–4 Data model with new G_Name group



9. Save your report as `wrappedbreak_<your initials>.rdf`.

15.3 Create three summary columns

The steps in this section show you how to use the Summary Column tool in the Data Model view to create three summary columns. These columns will calculate the percentage of each order total that the customer has purchased, the total purchases the customer has made, and the percentage of the total sales of all customers.

To create the summary columns:

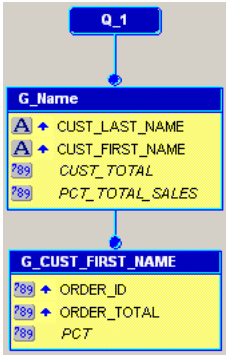
1. In the Data Model view, click the Summary Column tool in the tool palette, then click in the **G_CUST_FIRST_NAME** group to create a summary column.
2. Double-click the new summary column object (**CS_1**) to display the Property Inspector, and set properties:
   - Under **General Information**, set the Name property to PCT.
   - Under **Summary**, set the Function property to % of Total, set the Source property to ORDER_TOTAL, set the Reset At property to G_CUST_FIRST_NAME, and set the Compute At property to Report.
3. Create a second summary column in group **G_NAME** with the following properties:
   - Under **General Information**, set the Name property to CUST_TOTAL.

- Under **Summary**, set the Function property to Sum, set the Source property to ORDER_TOTAL, and set the Reset At property to G_NAME.

4. Create a third summary column in group **G_NAME** with the following properties:
   - Under **General Information**, set the Name property to PCT_TOTAL_SALES.
   - Under **Summary**, set the Function property to Sum, set the Source property to PCT, and set the Reset At property to G_NAME.

Your data model should now look something like this:

Figure 15–5 Data Model with Three Summary Columns



5. Save your report as `wrappedbreak_<your initials>.rdf`.

15.4 Create the default layout using the Report Wizard

The steps in this section show you how to use the Report Wizard to create the layout and choose how your data will display in your report. Here, you will choose the style of report you want to create, and choose to display the data across the report (hence creating the across group report).

You can create a default layout using the Report Wizard, which deletes any existing layouts in your report and replaces it with the new one.

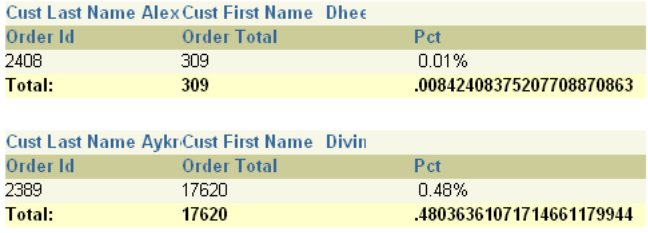
To create the default layout:

1. In the Data Model view, right-click on the canvas, then choose **Report Wizard**.
2. In the Report Wizard, on the **Report Type** page, select **Create Paper Layout only**.
3. On the **Style** page, select **Group Above**.
4. On the **Fields** page, click the double right arrows (>>) to move all of the fields to the **Displayed Fields** list.
5. On the **Labels** page, change the field widths as follows:

Fields	Width
CUST_FIRST_NAME	2
CUST_LAST_NAME	2

6. Click **Finish** to display your report output in the Paper Design view. It should look something like this:

Figure 15–6 Paper Design view for the wrapped field report



Notice how the complete names do not display. The steps in the next section will show you how to quickly and easily correct this issue.

15.5 Modify the layout of the report

In this section, you will change the field to expand vertically if the contents of the break field are longer than the specified width. As you can see in Figure 16-3, the width of the fields are fixed, but the names are incomplete. You will also add space to display between each record in the report.

To modify the layout:

1. In the Object Navigator, click the **R_G_NAME** node under Paper Layout > Main Section > Body, then press F4 to display the Property Inspector.
2. Under Repeating Frame, change the Vert. Spacing Between Frames to `0.25`, then press **Enter**.
3. Close the Property Inspector.
4. In the Object Navigator, under the Paper Layout node, navigate to **Main Section > Body > M_G_NAME_GRPFR > R_G_NAME**.
5. Click the **F_CUST_LAST_NAME** field, then press F4 to display the Property Inspector for that field.
6. Under General Layout, make sure the Vertical Elasticity property is set to **Expand**, then close the Property Inspector to accept your changes.
7. Follow steps 5 and 6 for the **F_CUST_FIRST_NAME** field.

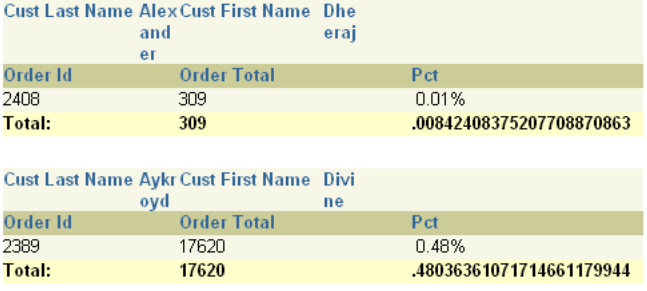
You have now modified the layout of your report to display all the text in the wrapped fields, and added space between the records.

15.6 Run your report to paper

In this section, you will run your report to the Paper Design view so you can see how your report displays.

Click the Paper Design button in the toolbar to display the Paper Design view. Your report runs, then displays in the Paper Design view. It should look like the following:

Figure 15–7 Paper Design view for the wrapped field report



Cust Last Name	Alexander	Cust First Name	Dheeraj	
Order Id		Order Total		Pct
2408		309		0.01%
Total:		**309**		**.008424083752077088708863**

Cust Last Name	Aykroyd	Cust First Name	Divine	
Order Id		Order Total		Pct
2389		17620		0.48%
Total:		**17620**		**.48036361071714661179944**

Save your report as `wrappedbreak_<your initials>.rdf`.

15.7 Summary

Congratulations! You have successfully created a wrapped field paper report. You now know how to:

- manually create a data model.
- create a master/detail (or group above) report using the Report Wizard.
- modify the layout of your report to wrap the text in a field.
- add space between records.

For more information on any of the wizards, views, or properties used in this example, refer to the *Reports Builder online help*, which you can access in two ways:

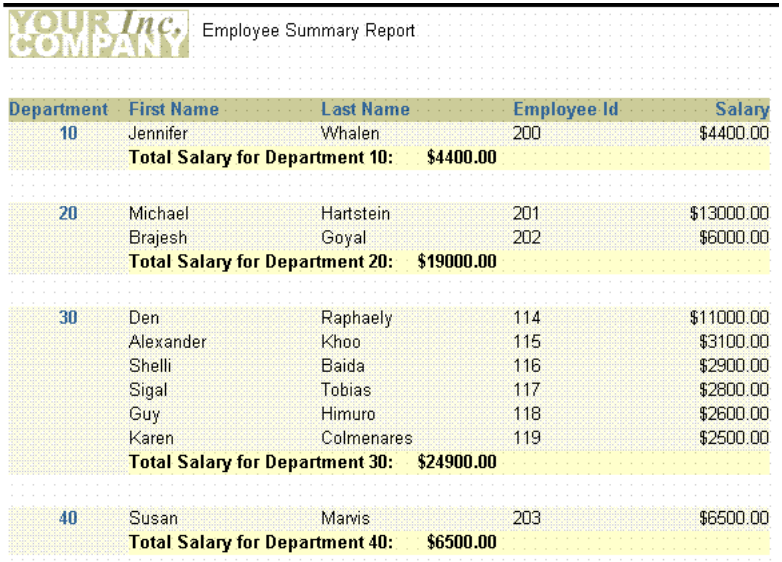
- From the Oracle Technology Network (`http://otn.oracle.com/products/reports/`), click **Documentation** and navigate to the *Reports Builder Online Help* for the most recent, hosted online help.
- From Reports Builder, choose **Help > Help Contents**.

Note: You can replace your Reports Builder online help with the most recent update by downloading the latest online help set available on the Oracle Technology Network (http://otn.oracle.com/docs/products/reports/). Instructions for replacing your help file are included in the readme.txt in the download file.

16

Building a Header and Footer Report

Figure 16–1 Group left report output with header and footer



YOUR Inc. COMPANY Employee Summary Report

Department	First Name	Last Name	Employee Id	Salary
10	Jennifer	Whalen	200	$4400.00
	Total Salary for Department 10:	**$4400.00**		
20	Michael	Hartstein	201	$13000.00
	Brajesh	Goyal	202	$6000.00
	Total Salary for Department 20:	**$19000.00**		
30	Den	Raphaely	114	$11000.00
	Alexander	Khoo	115	$3100.00
	Shelli	Baida	116	$2900.00
	Sigal	Tobias	117	$2800.00
	Guy	Himuro	118	$2600.00
	Karen	Colmenares	119	$2500.00
	Total Salary for Department 30:	**$24900.00**		
40	Susan	Marvis	203	$6500.00
	Total Salary for Department 40:	**$6500.00**		

This report has a page header, `Employee Summary Report`, printed in the upper margin area of every page of the report, and a footer, `Total Salary for Department <number>: <sum_sal>`, printed at the end of the list of employee information for each department.

Concepts

- A header and footer report contains boilerplate or fields in its header or footer. To create headers and footers in your report, add the boilerplate or field in the

Paper Layout view of the Report Editor. If you create a field, make sure there is only one value for it. Otherwise, Reports Builder will not know which of the different values you want to print.

- How often the boilerplate and fields appear and whether they are headers or footers depends on where you position them in the layout. Any boilerplate that is positioned within a repeating frame will print once every time the repeating frame prints. Boilerplate that is outside of all repeating frames will print once for the entire report.
- This report uses one query to select all necessary columns. You'll add a break by assigning a column to a second group.
- The report uses a group left layout with modifications. You'll increase the size of a repeating frame and the frame enclosing it to ensure they are large enough to contain a footer, then add the footer. Then you'll modify the vertical spacing of a repeating frame to add space between each instance of it.

Example Scenario

Suppose that you want to create a report that displays and summarizes employee data by department. To make the report more readable, you decide to add a header to each page and to add footer for each master record.

To see a sample report with a header and footer, open the examples folder named `headerfooter`, then open the Oracle Reports example named `headfootb.rdf`. For details on how to access it, see "Accessing the example reports" in the Preface.

Table 16–1 Features demonstrated in this example

Feature	Location
Create a data model and a layout	Section 16.2, "Create a data model and a group left layout"
Move departmental summary closer to its label	Section 16.3, "Move a summary"
Add a heading that repeats on all pages	Section 16.4, "Add a page heading"
Add white space and format monetary values	Section 16.5, "Adding white space and formatting values"

16.1 Prerequisites for this example

To build the example in this chapter, you must have access to the sample schema provided with the Oracle9*i* database. If you don't know if you have access to this

sample schema, contact your database administrator. You should have access to the "Human Resources" portion of the schema to complete this example.

16.2 Create a data model and a group left layout

When you are creating a single-query report, such as this one, you can use the Report Wizard to create the data model and layout simultaneously.

To create a data model and layout:

1. Launch Reports Builder (or, if already open, choose **File > New > Report**)
2. In the Welcome or New Report dialog box, select **Use the Report Wizard**, then click **OK**.
3. If the Welcome page displays, click **Next**.
4. On the Report Type page, select **Create Paper Layout Only**, then click **Next**.
5. On the Style page, type a **Title** for your report, select **Group Left**, then click **Next**.
6. On the Data Source page, click **SQL Query**, then click **Next**.
7. On the Data page, enter the following SELECT statement in the **Data Source definition** field:

```
SELECT FIRST_NAME, LAST_NAME, EMPLOYEE_ID, SALARY, DEPARTMENT_ID
FROM EMPLOYEES
ORDER BY DEPARTMENT_ID, EMPLOYEE_ID
```

Note: You can enter this query in any of the following ways:

- Copy and paste the code from the provided text file called `headfoodtb_code.txt` into the **Data Source definition** field.
- Click **Query Builder** to build the query without entering any code manually.
- Type the code in the **Data Source definition** field

---

8. Click **Next**.

> **Note:** If you are not already connected to a database, you will be prompted to connect to the database when you click **Next**. Ensure that you connect to a database that has the appropriate schema for this example. Section 16.1, "Prerequisites for this example" describes the sample schema requirements for this example.

9. On the Groups page, click **DEPARTMENT_ID** and click the right arrow (>) to move this field to the **Group Fields** list, then click **Next**.
10. On the Fields page, click the double right arrows (>>) to move all of the fields to the **Displayed Fields** list, then click **Next**.
11. On the Totals page, click **SALARY**, then click Sum.
12. Click **Next**.
13. On the Labels page, change the labels as follows, then click **Next**:

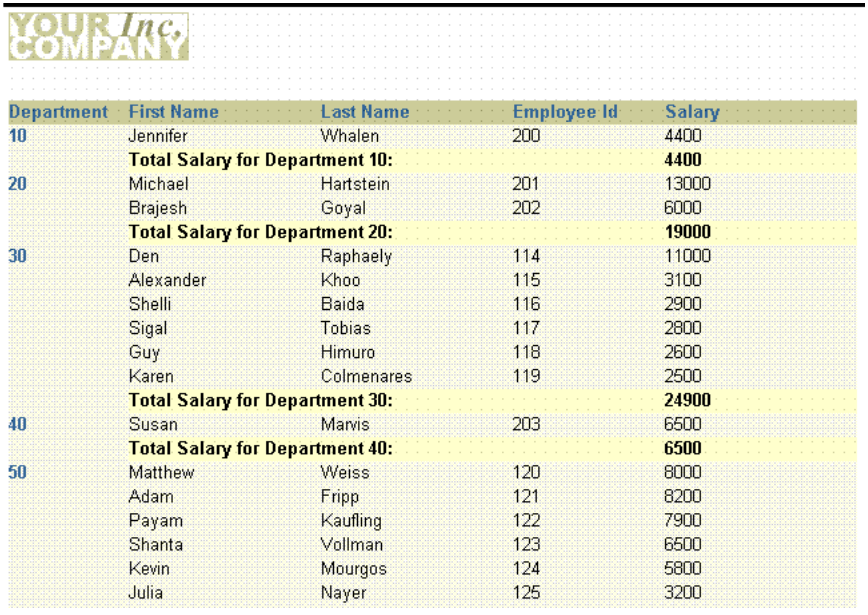
Fields	Labels
DEPARTMENT_ID	`Department`
SumSALARYPerDEPARTMENT_ID	`Total Salary for Department &DEPARTMENT_ID:`

This step will add a footer to each instance of the master repeating frame (departments).

> **Note:** The DEPARTMENT_ID column is part of the data model so you need to reference DEPARTMENT_ID in the footer label using &DEPARTMENT_ID.

14. On the Template page, click Finish to display your report output in the Paper Design view. It should look something like this:

Figure 16–2 Group left report output with footer for master repeating frame



Department	First Name	Last Name	Employee Id	Salary
10	Jennifer	Whalen	200	4400
	Total Salary for Department 10:			**4400**
20	Michael	Hartstein	201	13000
	Brajesh	Goyal	202	6000
	Total Salary for Department 20:			**19000**
30	Den	Raphaely	114	11000
	Alexander	Khoo	115	3100
	Shelli	Baida	116	2900
	Sigal	Tobias	117	2800
	Guy	Himuro	118	2600
	Karen	Colmenares	119	2500
	Total Salary for Department 30:			**24900**
40	Susan	Marvis	203	6500
	Total Salary for Department 40:			**6500**
50	Matthew	Weiss	120	8000
	Adam	Fripp	121	8200
	Payam	Kaufling	122	7900
	Shanta	Vollman	123	6500
	Kevin	Mourgos	124	5800
	Julia	Nayer	125	3200

16.3 Move a summary

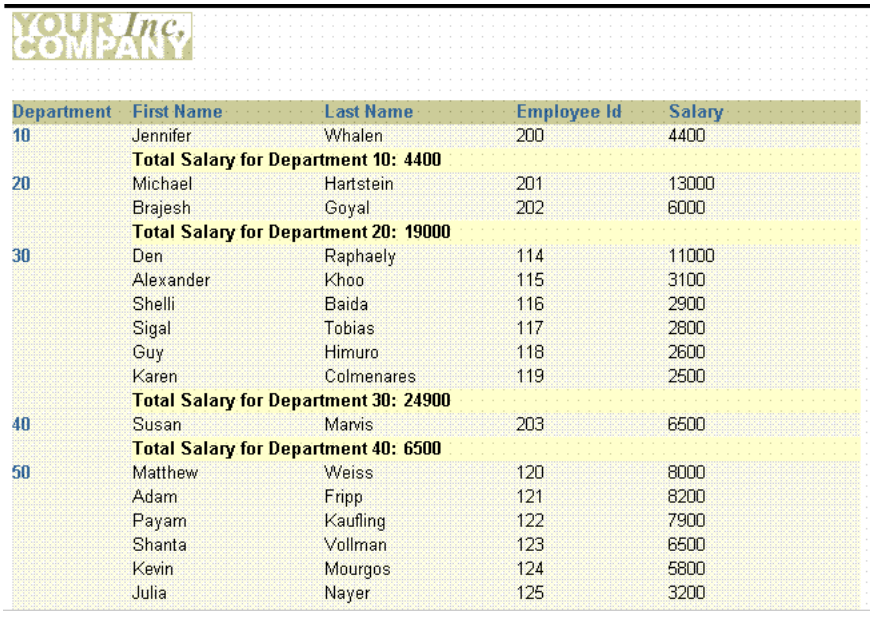
In the Paper Design view, notice how the department summary of salaries is very far to the right of the footer label (**Total Salary for Department &DEPARTMENT_ID**). For this report, you will move the salary summary closer to its label.

To move the summary closer to its label:

1. In the Paper Design view, click the Flex Off button in the toolbar to turn Flex mode off.
2. Select the label, **Total Salary for Department &DEPARTMENT_ID**.
3. Click and drag the rightmost handle of the label about 0.75 inches to the left to resize the object.
4. Select the first summary value to the right of the label **Total Salary for Department &DEPARTMENT_ID**.
5. Click and drag the summary value to the left until it is fairly close to its label.

6. Click and drag the rightmost handle of the field about 1 inch to the left.
7. Click the Flex On button in the toolbar to turn Flex mode back on.

Figure 16–3 Group left report output with summary closer to label



Department	First Name	Last Name	Employee Id	Salary
10	Jennifer	Whalen	200	4400
	Total Salary for Department 10: 4400			
20	Michael	Hartstein	201	13000
	Brajesh	Goyal	202	6000
	Total Salary for Department 20: 19000			
30	Den	Raphaely	114	11000
	Alexander	Khoo	115	3100
	Shelli	Baida	116	2900
	Sigal	Tobias	117	2800
	Guy	Himuro	118	2600
	Karen	Colmenares	119	2500
	Total Salary for Department 30: 24900			
40	Susan	Marvis	203	6500
	Total Salary for Department 40: 6500			
50	Matthew	Weiss	120	8000
	Adam	Fripp	121	8200
	Payam	Kaufling	122	7900
	Shanta	Vollman	123	6500
	Kevin	Mourgos	124	5800
	Julia	Nayer	125	3200

16.4 Add a page heading

Now that you have added the repeating frame footer, it is time to add a page header. You perform this task from the Paper Layout view.

1. In the Paper Layout view, click the Edit Margin button in the toolbar.
2. From the font lists in the toolbar, choose Arial (Western), point size 10.
3. Click the Text tool in the tool palette.
4. Click somewhere to the right of the logo image and type the following text:

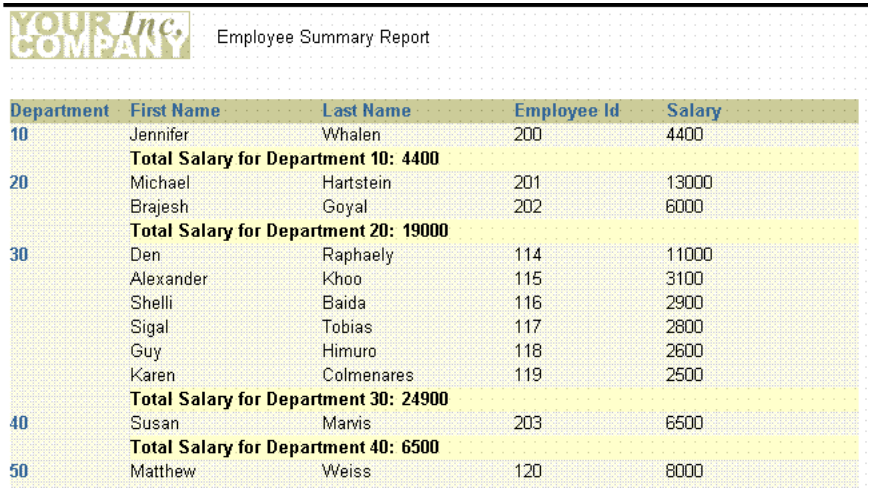
   ```
   Employee Summary Report
   ```

5. Move to an open area of the Paper Layout view and click the mouse button to exit text mode. Notice that the text object you just created is still selected, you

can now adjust its positioning with the arrow keys. If you click in an open area a second time, the object is deselected.

6. Save your report.

Figure 16–4 Group left report output with page heading



YOUR Inc. COMPANY

Employee Summary Report

Department	First Name	Last Name	Employee Id	Salary
10	Jennifer	Whalen	200	4400
	Total Salary for Department 10: 4400			
20	Michael	Hartstein	201	13000
	Brajesh	Goyal	202	6000
	Total Salary for Department 20: 19000			
30	Den	Raphaely	114	11000
	Alexander	Khoo	115	3100
	Shelli	Baida	116	2900
	Sigal	Tobias	117	2800
	Guy	Himuro	118	2600
	Karen	Colmenares	119	2500
	Total Salary for Department 30: 24900			
40	Susan	Marvis	203	6500
	Total Salary for Department 40: 6500			
50	Matthew	Weiss	120	8000

16.5 Adding white space and formatting values

The only task that remains now is to make your report a little more readable by adding some white space between records and formatting values properly.

1. In the Paper Design view, select the first department number value, which should be 10. All of the department numbers are immediately selected indicating that you can change their properties simultaneously.
2. Click the Align Center button in the toolbar.
3. Click the Select Parent Frame button in the toolbar.
4. Choose **Tools > Property Inspector** to display the Property Inspector, and set properties:
   - Under **Repeating Frame**, set the Vertical Space Between Frames property to 0.25.
5. Click the title bar of the Paper Design view.

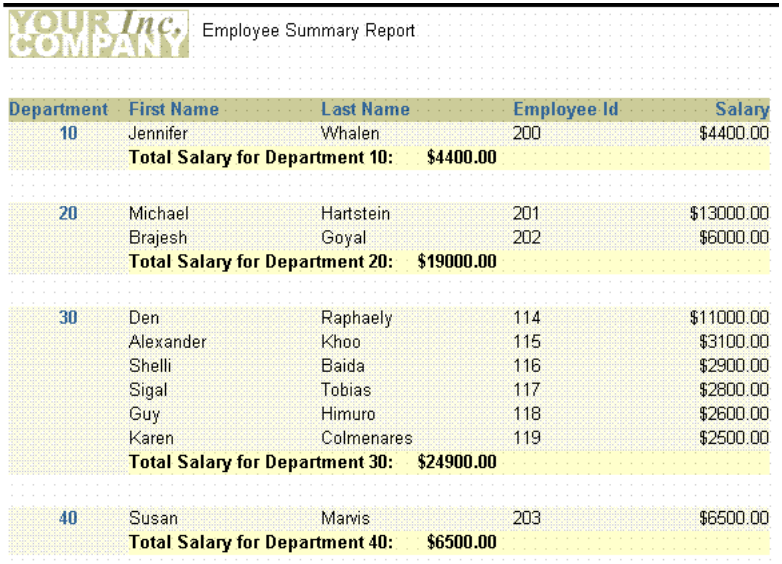
6. Select the first number value underneath the **Salary** label.
7. Shift-click on the department summary value for the first department.

   Tip: If you are familiar with format mask syntax, you could now right click on the field values, choose Property Inspector, and choose or manually enter a value for the Format Mask property.

8. Click the Currency button in the toolbar. A currency symbol immediately appears next to all of the values.
9. Click the Add Decimal Place button twice. Two decimal places are added to the right of the decimal point.
10. Resize the salary field (**F_SALARY**) by clicking and dragging its rightmost handle approximately 0.5 inches to the left. Notice that even though the summary field (**F_SumSALARYPerDEPARTMENT_ID**) is also selected, only **F_SALARY** is resized.
11. Shift-click the **Salary** label itself.
12. Click the Align Right button in the toolbar. All of the values are immediately right aligned.

    Tip: The steps above have formatted all of the salaries and the department summary of your report. The report summary (the summary of all salaries in the report), however, is not yet formatted. You can find the report summary on the last page of the report and format it using the same techniques as above.

Figure 16–5 Group left report output with values formatted



Employee Summary Report

Department	First Name	Last Name	Employee Id	Salary
10	Jennifer	Whalen	200	$4400.00
	Total Salary for Department 10:	**$4400.00**		
20	Michael	Hartstein	201	$13000.00
	Brajesh	Goyal	202	$6000.00
	Total Salary for Department 20:	**$19000.00**		
30	Den	Raphaely	114	$11000.00
	Alexander	Khoo	115	$3100.00
	Shelli	Baida	116	$2900.00
	Sigal	Tobias	117	$2800.00
	Guy	Himuro	118	$2600.00
	Karen	Colmenares	119	$2500.00
	Total Salary for Department 30:	**$24900.00**		
40	Susan	Marvis	203	$6500.00
	Total Salary for Department 40:	**$6500.00**		

16.6 Summary

Congratulations! You have successfully created a header and footer report. You now know how to:

- create a data model and a layout.
- move a summary field.
- add a heading that repeats on all pages.
- add white space and format values.

For more information on any of the wizards, views, or properties used in this example, refer to the *Reports Builder online help*, which you can access in two ways:

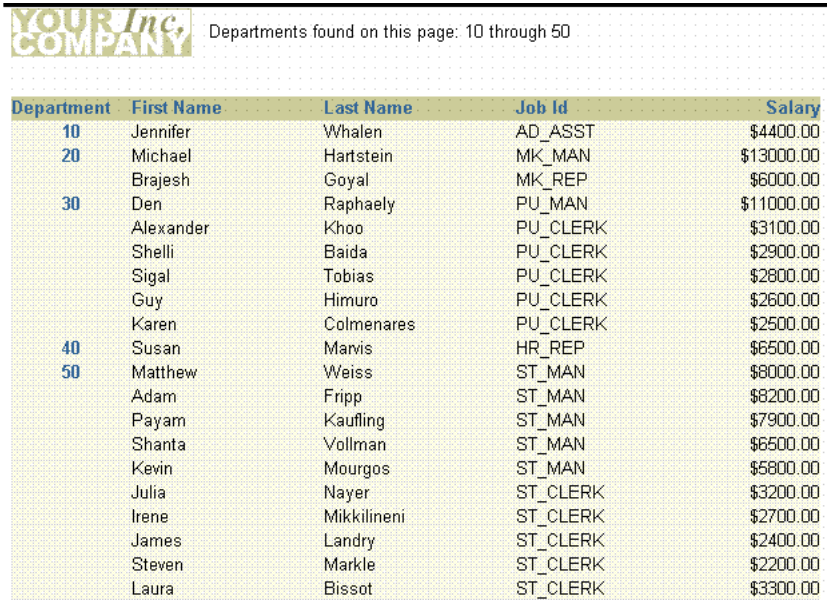
- From the Oracle Technology Network (`http://otn.oracle.com/products/reports/`), click **Documentation** and navigate to the *Reports Builder Online Help* for the most recent, hosted online help.
- From Reports Builder, choose **Help > Help Contents**.

Note: You can replace your Reports Builder online help with the most recent update by downloading the latest online help set available on the Oracle Technology Network (http://otn.oracle.com/docs/products/reports/). Instructions for replacing your help file are included in the readme.txt in the download file.

17

Building a Header with Database Values Report

Figure 17–1 Group left report output with database values in header



Department	First Name	Last Name	Job Id	Salary
10	Jennifer	Whalen	AD_ASST	$4400.00
20	Michael	Hartstein	MK_MAN	$13000.00
	Brajesh	Goyal	MK_REP	$6000.00
30	Den	Raphaely	PU_MAN	$11000.00
	Alexander	Khoo	PU_CLERK	$3100.00
	Shelli	Baida	PU_CLERK	$2900.00
	Sigal	Tobias	PU_CLERK	$2800.00
	Guy	Himuro	PU_CLERK	$2600.00
	Karen	Colmenares	PU_CLERK	$2500.00
40	Susan	Marvis	HR_REP	$6500.00
50	Matthew	Weiss	ST_MAN	$8000.00
	Adam	Fripp	ST_MAN	$8200.00
	Payam	Kaufling	ST_MAN	$7900.00
	Shanta	Vollman	ST_MAN	$6500.00
	Kevin	Mourgos	ST_MAN	$5800.00
	Julia	Nayer	ST_CLERK	$3200.00
	Irene	Mikkilineni	ST_CLERK	$2700.00
	James	Landry	ST_CLERK	$2400.00
	Steven	Markle	ST_CLERK	$2200.00
	Laura	Bissot	ST_CLERK	$3300.00

In this example report, both the first and last department numbers found on each page are displayed in the page header.

Concepts

- This report uses one query and two groups. You'll also create two summary columns to provide the values for the header fields. To ensure unique field values for each page, you'll compute the values using the First and Last functions. The First function will return the first database value selected for a group, page, or report, and the Last function will return the last database value selected for a group, page, or report.

Example Scenario

Suppose that you want to create a report that displays and summarizes employee data by department. To make the report more readable, you decide to add a header to each page that indicates which departments appear on the page.

To see a sample header with database values report, open the examples folder named `headingdb`, then open the Oracle Reports example called `headingdb.rdf`. For details on how to access it, see "Accessing the example reports" in the Preface.

Table 17–1 Features demonstrated in this example

Feature	Location
Create a data model and a layout	Section 17.2, "Create a data model and a group left layout"
Add summary columns to populate a header	Section 17.3, "Add summary columns for the header data"
Add a heading that repeats on all pages with database values	Section 17.4, "Add a page heading"

17.1 Prerequisites for this example

To build the example in this chapter, you must have access to the sample schema provided with the Oracle9*i* database. If you don't know if you have access to this sample schema, contact your database administrator. You should have access to the "Human Resources" portion of the schema to complete this example.

17.2 Create a data model and a group left layout

When you are creating a single-query report, such as this one, you can use the Report Wizard to create the data model and layout simultaneously.

To create a data model and layout:

1. Launch Reports Builder (or, if already open, choose **File > New > Report**)
2. In the Welcome or New Report dialog box, select **Use the Report Wizard**, then click **OK**.
3. If the Welcome page displays, click **Next**.
4. On the Report Type page, select **Create Paper Layout Only**, then click **Next**.
5. On the Style page, type a **Title** for your report, select **Group Left**, then click **Next**.
6. On the Data Source page, click **SQL Query**, then click **Next**.
7. On the Data page, enter the following SELECT statement in the **Data Source definition** field:

```
SELECT DEPARTMENT_ID, FIRST_NAME, LAST_NAME, JOB_ID, SALARY
FROM EMPLOYEES
ORDER BY DEPARTMENT_ID, EMPLOYEE_ID
```

> **Note:** You can enter this query in any of the following ways:
>
> - Copy and paste the code from the provided text file called `heading_code.txt` into the **Data Source definition** field.
> - Click **Query Builder** to build the query without entering any code manually.
> - Type the code in the **Data Source definition** field.

8. Click **Next**.

> **Note:** If you are not already connected to a database, you will be prompted to connect to the database when you click **Next**. Ensure that you connect to a database that has the appropriate schema for this example. Section 17.1, "Prerequisites for this example" describes the sample schema requirements for this example.

9. On the Groups page, click **DEPARTMENT_ID** and click the right arrow (>) to move this field to the **Group Fields** list, then click **Next**.
10. On the Fields page, click the double right arrows (>>) to move all of the fields to the **Displayed Fields** list, then click **Next**.
11. On the Totals page, click **Next**.
12. On the Labels page, change the labels as follows, then click **Next**:

Fields	Labels
DEPARTMENT_ID	`Department`
JOB_ID	`Job`

13. On the Template page, click **Finish** to display your report output in the Paper Design view.
14. In the Paper Design view, click the first number value underneath the **Salary** label.

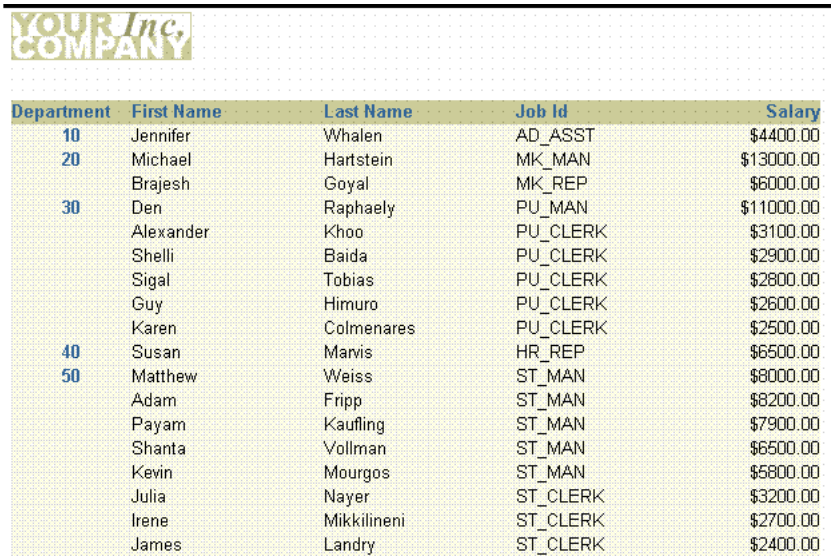
    Tip: If you are familiar with format mask syntax, you could now right click on the field values, choose Property Inspector, and choose or manually enter a value for the Format Mask property.

15. Click the Currency button in the toolbar. A currency symbol immediately appears next to all of the values.
16. Click the Add Decimal Place button twice. Two decimal places are added to the right of the decimal point.
17. Resize the field by clicking and dragging the rightmost handle of the field approximately 0.5 inches to the left.
18. Shift-click the **Salary** label itself.
19. Click the Align Right button in the toolbar. All of the values and the Salary label are immediately right aligned.
20. Click the first department number value under the **Department** label to select it.

21. Click the Align Center button in the toolbar.

    Your report should look something like this:

Figure 17–2 Group left report output with values formatted



Department	First Name	Last Name	Job Id	Salary
10	Jennifer	Whalen	AD_ASST	$4400.00
20	Michael	Hartstein	MK_MAN	$13000.00
	Brajesh	Goyal	MK_REP	$6000.00
30	Den	Raphaely	PU_MAN	$11000.00
	Alexander	Khoo	PU_CLERK	$3100.00
	Shelli	Baida	PU_CLERK	$2900.00
	Sigal	Tobias	PU_CLERK	$2800.00
	Guy	Himuro	PU_CLERK	$2600.00
	Karen	Colmenares	PU_CLERK	$2500.00
40	Susan	Marvis	HR_REP	$6500.00
50	Matthew	Weiss	ST_MAN	$8000.00
	Adam	Fripp	ST_MAN	$8200.00
	Payam	Kaufling	ST_MAN	$7900.00
	Shanta	Vollman	ST_MAN	$6500.00
	Kevin	Mourgos	ST_MAN	$5800.00
	Julia	Nayer	ST_CLERK	$3200.00
	Irene	Mikkilineni	ST_CLERK	$2700.00
	James	Landry	ST_CLERK	$2400.00

22. Save your report as `headingb_<your initials>.rdf`.

17.3 Add summary columns for the header data

For the header that you want to create (i.e., departments found on this page), you need to create two summary columns that compute the values you need in order to populate the header.

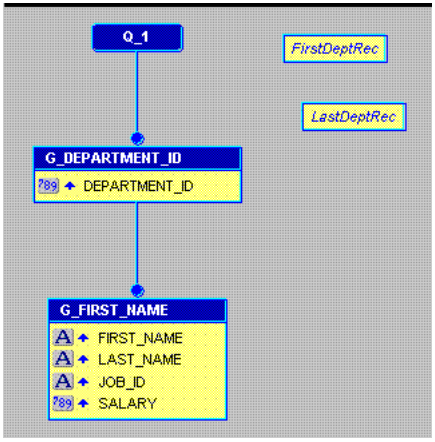
1. In the Report Editor, click the Data Model button in the toolbar to display the Data Model view.
2. Click the Summary Column tool in the tool palette, then click in an open area of the Data Model view to create a summary column.
3. Double-click the new summary column object (**CS_1**) to display the Property Inspector, and set properties:

- Under **General Information**, set the Name property to FirstDeptRec.
- Under **Summary**, set the Function property to First, set the Source property to DEPARTMENT_ID, set the Reset At property to Page.

4. Create a second summary column, and set its properties as follows:
   - Under **General Information**, set the Name property to LastDeptRec.
   - Under **Summary**, set the Function property to Last, set the Source property to DEPARTMENT_ID, set the Reset At property to Page.

   Your data model should now look like this:

Figure 17–3 Data Model with Summary Columns for Header



17.4 Add a page heading

Now that you have the data for the header, you need to add the corresponding layout objects to the margin area of the report.

1. In the Paper Design view, from the font lists in the toolbar, choose Arial (Western), point size 10.
2. In the Paper Layout view, click the Edit Margin button in the toolbar.
3. Click the Text tool in the tool palette.
4. Draw an area somewhere to the right of the logo image and type the following text:

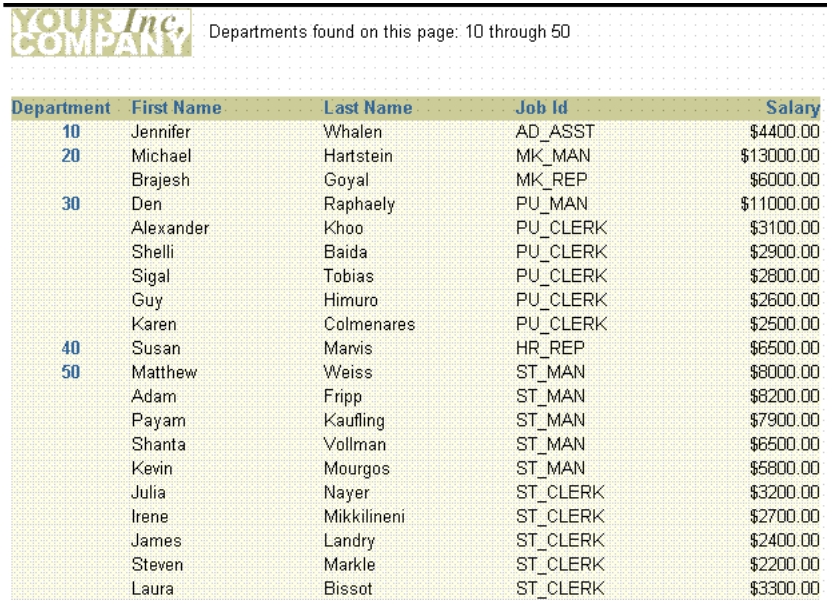
```
Departments found on this page: &FirstDeptRec through &LastDeptRec
```

5. Move to an open area of the Paper Layout view and click the mouse button to exit text mode. Notice that the text object you just created is still selected, you can now adjust its positioning with the arrow keys. If you click in an open area a second time, the object is deselected.

   Tip: If the header values on the first page appear incorrectly after you exit text mode (e.g., 50 through 50), choose **Program > Run Paper Layout**.

Your report should now look like this:

Figure 17–4 Group left report output with database values in the header



Department	First Name	Last Name	Job Id	Salary
10	Jennifer	Whalen	AD_ASST	$4400.00
20	Michael	Hartstein	MK_MAN	$13000.00
	Brajesh	Goyal	MK_REP	$6000.00
30	Den	Raphaely	PU_MAN	$11000.00
	Alexander	Khoo	PU_CLERK	$3100.00
	Shelli	Baida	PU_CLERK	$2900.00
	Sigal	Tobias	PU_CLERK	$2800.00
	Guy	Himuro	PU_CLERK	$2600.00
	Karen	Colmenares	PU_CLERK	$2500.00
40	Susan	Marvis	HR_REP	$6500.00
50	Matthew	Weiss	ST_MAN	$8000.00
	Adam	Fripp	ST_MAN	$8200.00
	Payam	Kaufling	ST_MAN	$7900.00
	Shanta	Vollman	ST_MAN	$6500.00
	Kevin	Mourgos	ST_MAN	$5800.00
	Julia	Nayer	ST_CLERK	$3200.00
	Irene	Mikkilineni	ST_CLERK	$2700.00
	James	Landry	ST_CLERK	$2400.00
	Steven	Markle	ST_CLERK	$2200.00
	Laura	Bissot	ST_CLERK	$3300.00

6. Save your report.

17.5 Summary

Congratulations! You have successfully created a header with database values report. You now know how to:

- create a data model and a layout.
- add summary columns to populate a header.
- add a heading that repeats on all pages with database values.

For more information on any of the wizards, views, or properties used in this example, refer to the *Reports Builder online help*, which you can access in two ways:

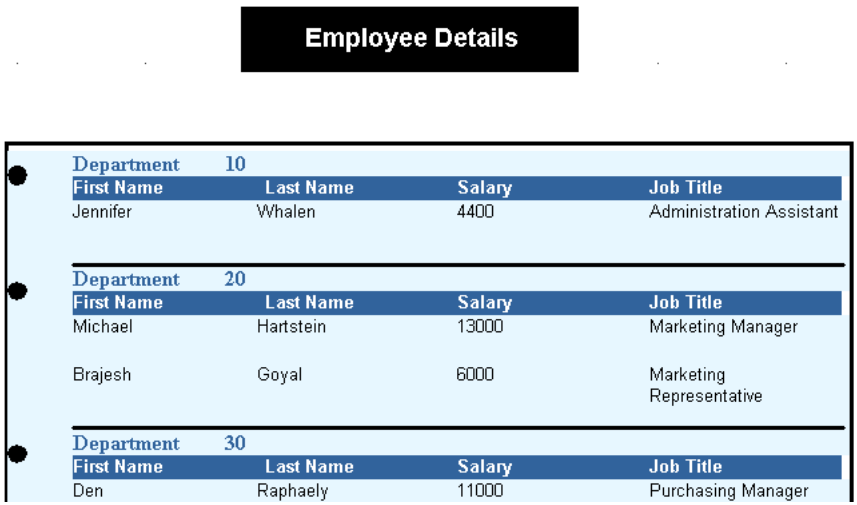
- From the Oracle Technology Network (`http://otn.oracle.com/products/reports/`), click **Documentation** and navigate to the *Reports Builder Online Help* for the most recent, hosted online help.
- From Reports Builder, choose **Help > Help Contents**.

> **Note:** You can replace your Reports Builder online help with the most recent update by downloading the latest online help set available on the Oracle Technology Network (http://otn.oracle.com/docs/products/reports/). Instructions for replacing your help file are included in the readme.txt in the download file.

18

Building a Report with Graphics, Text, and Color

Figure 18–1 Graphics, text, and color report output



Reports Builder enables you to modify the look of your report in multiple ways. In this example, you will build a report and enhance it by adding an image to the margin, a title, and a border. You will also change the look of the report by applying different fonts and text styles.

Concepts

This report shows you how to enhance your reports with graphics by:

- using the Paper Layout view's drawing tools.
- highlighting boilerplate text appearing in the output.
- using different colors (if available) and patterns for boilerplate text.

For more information on enhancing the appearance of your reports, refer to the *Reports Builder online help*.

Reports Builder enables to you to manipulate the appearance of your report on the fly. If you intend to reuse the look and feel of your report, you can always create a template and add it to your template library. You can find out more information about using templates in Getting Started with Oracle Reports (`http://otn.oracle.com/products/reports/`).

Data Relationships

- This is a one-query group left report.

Layout

- This report uses a Group Above layout, which you'll modify in the Paper Layout view to make room for the title and company logo. Then, you'll import the logo, create the report title, and the rest of the graphics shown in the image above.

Example Scenario

In this example, you'll create one query that selects all the columns for this report. You'll also assign a column to a new break group.

To see a sample report with graphics and highlighted text, open the examples folder named `graphics`, then open the Oracle Reports example named `graphics.rdf`. For details on how to open it, see "Accessing the example reports" in the Preface.

Table 18–1 Features demonstrated in this example

Feature	Location
Use the Report Wizard to create a simple report definition.	Section 18.2, "Create a simple report definition"
Modify the appearance of your report in the Paper Layout view.	Section 18.3, "Modify the report in the Paper Layout view"
Run your report to paper.	Section 18.3.1, "Adding a border around the report"

Table 18–1 Features demonstrated in this example

Feature	Location
Use the Font dialog box to change the font of your text.	Section 18.3.2, "Change the font size and style"
Use the tool palette to create bullets in your report.	Section 18.3.3, "Add bullets your report"
Use the Line tool to create a horizontal separator between records.	Section 18.3.4, "Display a line between each record"
Use the Fill Color and other layout tools to add a title to your report.	Section 18.3.5, "Add a title to your report"

18.1 Prequisites for this example

To build the example in this chapter, you must have access to the sample schema provided with the Oracle9*i* database. If you don't know if you have access to this sample schema, contact your database administrator. You should have access to the "Human Resources" portion of the schema to complete this example.

18.2 Create a simple report definition

When you create a report, you can either use the Report Wizard to assist you or create the report yourself. In this example, you will use the Report Wizard to create your query and basic report layout.

To create the report definition:

1. Launch Reports Builder (or, if already open, choose **File > New > Report**)
2. In the Welcome or New Report dialog box, select **Use the Report Wizard**, then click **OK**.
3. If the Welcome page displays, click **Next**.
4. On the Report Type page, select **Create Paper Layout Only**, then click **Next**.
5. On the Style page, type a **Title** for your report, select **Group Above**, then click **Next**.
6. On the Data Source page, click **SQL Query**, then click **Next**.
7. On the Data page, enter the following SELECT statement in the **Data Source definition** field:

```
SELECT ALL DEPARTMENTS.DEPARTMENT_ID, EMPLOYEES.FIRST_NAME, EMPLOYEES.LAST_
NAME, EMPLOYEES.JOB_ID, EMPLOYEES.SALARY, JOBS.JOB_TITLE
FROM DEPARTMENTS, EMPLOYEES, JOBS
WHERE ((DEPARTMENTS.DEPARTMENT_ID = EMPLOYEES.DEPARTMENT_ID)
AND (EMPLOYEES.JOB_ID = JOBS.JOB_ID))
ORDER BY DEPARTMENTS.DEPARTMENT_ID
```

Note: You can enter this query in any of the following ways:

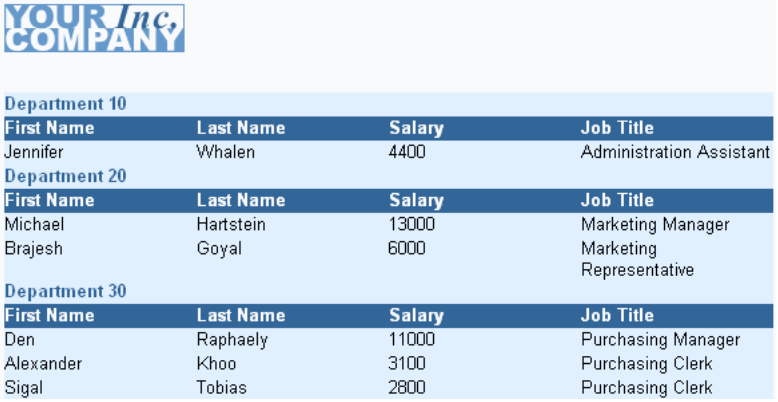
- Copy and paste the code from the provided text file called `graphics_code.txt` into the **Data Source definition** field.
- Click **Query Builder** to build the query without entering any code manually.
- Type the code in the **Data Source definition** field.

8. Click **Next**.

Note: If you are not already connected to a database, you will be prompted to connect to the database when you click **Next**. Ensure that you connect to a database that has the appropriate schema for this example. Section 18.1, "Prequisites for this example" describes the sample schema requirements for this example.

9. On the Groups page, click **DEPARTMENT_ID** and click the right arrow (**>**) to move this field to the **Group Fields** list, then click **Next**.
10. On the Fields page, move all fields *except* **JOB_ID** to the **Displayed Fields** list, then click **Next**.
11. On the Totals page, click **Next**.
12. On the Labels page, change the label **Department Id** to `Department`, then click **Next**.
13. On the Template page, select **Predefined Template** and click **Blue**, then click **Finish** to display your report output in the Paper Design view. It should look something like this:

Figure 18–2 Paper Design view for the graphics report with one query



18.3 Modify the report in the Paper Layout view

The steps in this section show you how to use the various tools in the Paper Layout view to modify the appearance of your report. First, you will move the existing frames to accommodate the changes you will make. You will then add a title to your report, borders around each record, and change the text color.

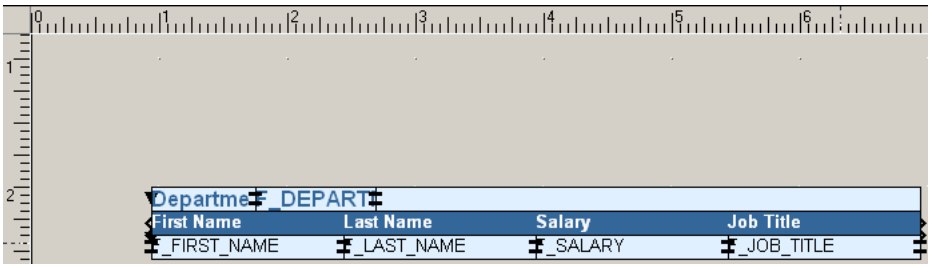
To modify your report:

1. Click the Paper Layout view button in the toolbar to display the Paper Layout view.
2. Make sure the Confine On button in the toolbar is set to Confine mode on. Doing so allows you to move the objects in the Paper Layout view around the canvas.
3. In the Object Navigator, under **Paper Layout > Body**, click the item **M_G_DEPARTMENT_ID_GRPFR**.

   Notice that, in the Paper Layout view, the outermost frame is selected.
4. Click on the frame in the Paper Layout view and drag it about two inches down and one inch to the right.

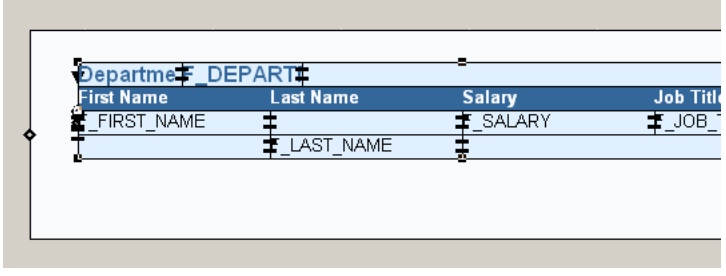
   You can use the rulers along the sides of the Paper Layout view as guidance. Your Paper Layout should now look something like this:

Figure 18–3 Paper Layout View of Adjusted Report Layout



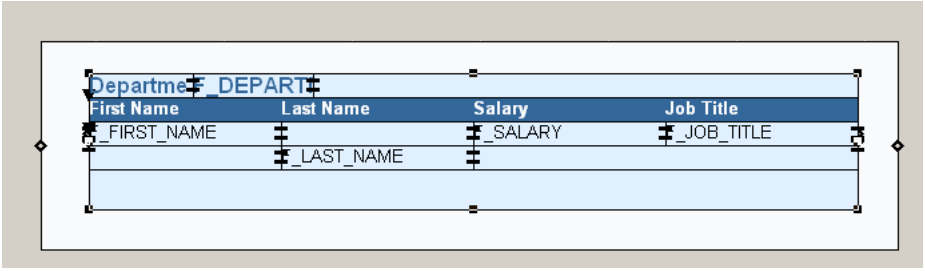
5. Now, make sure Flex mode is on and click the Confine Off button in the toolbar to set Confine mode off.
6. Make space for the border you will create that will surround each record. In the Object Navigator, select **M_G_DEPARTMENT_ID_GRPFR** again.
7. Drag each corner to make space around the inner repeating frame, so that the layout now looks like the following image. Notice the white space around the groups.

Figure 18–4 Paper Layout View with Expanded Repeating Frame



8. Now, let's make space between each record so that we can later put a horizontal line between the records. In the Object Navigator, click **R_G_DEPARTMENT_ID**.
9. In the Paper Layout view, click the bottom right corner and drag it down about 0.5 inches so that your layout now looks like this:

Figure 18–5 Paper Layout View with Expanded R_G_DEPARTMENT_ID Repeating Frame



10. Save your report as `graphics_<your initials>.rdf`.

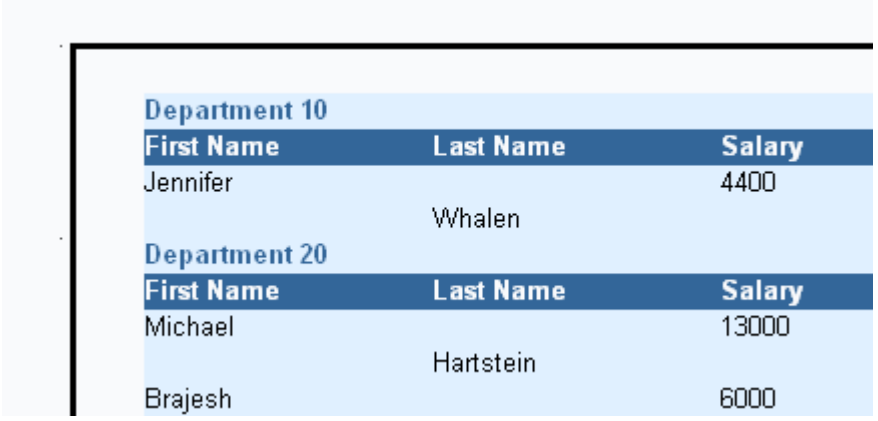
18.3.1 Adding a border around the report

Now that you've set up your layout, you can start adding new items. In this section, you will add a border around the entire report.

To add a border:

1. In the Object Navigator, select the M_G_DEPARTMENT_ID_GRPFR frame.
2. In the Paper Layout view, while the M_G_DEPARTMENT_ID_GRPFR frame is selected, click the Line Color tool. The tool can be found at the very bottom of the tool palette.
3. In the color palette that displays, choose the black square to make the border black.
4. While the line is still selected, choose **Format > Line > Line Width > 2 pt**.
5. Now, preview the layout. In the toolbar, click the Paper Design button in the toolbar to run the report to paper. It should look something like this:

Figure 18–6 Preview of the Report with a Border



6. Save your report as `graphics_<your initials>.rdf`.

18.3.2 Change the font size and style

In the Paper Layout view, you can also change the font and style of the text. The steps in this section will show you how to modify the Department Number text.

To change the font:

1. In the Paper Layout view, click the Paper Layout button in the toolbar.
2. In the Object Navigator, select the **B_DEPARTMENT_ID** and **F_DEPARTMENT_ID** fields so that you can change the font of the Department ID label and text.

> **Note:** B_DEPARTMENT_ID refers to the boilerplate text "Department ID", which serves as the label for the F_DEPARTMENT_ID field.

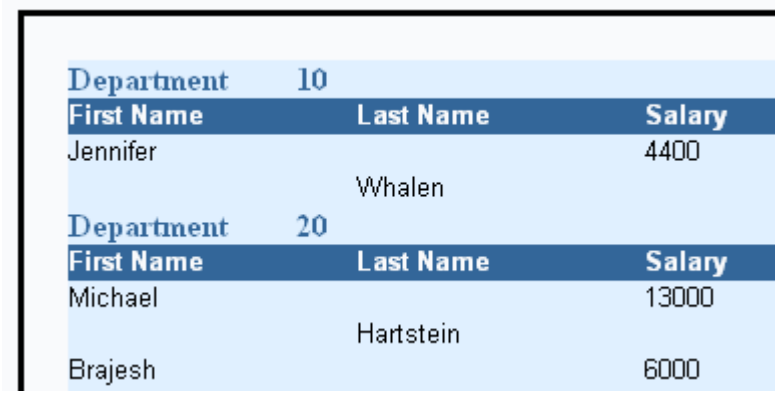
3. Click on the title bar of Paper Layout view to make it the active window.
4. While the two fields are selected, choose **Format > Font**.

5. In the Font dialog box, choose a different font and size. In the example, we chose Times New Roman and 12 pt.

> **Note:** Depending on the font and font size you choose, you may have to adjust the size of the fields to accommodate the text. In the example, since the font was increased, we selected the boilerplate text field and enlarged it.

6. When you've finished your changes, click **OK**.
7. Display your report in the Paper Design view to see what it looks like. Here's an example:

Figure 18–7 Paper Design View of the Report with New Fonts



8. Save your report as `graphics_<your initials>.rdf`.

18.3.3 Add bullets your report

The steps in this section will show you how to add bullets so that a bullet displays next to each department ID.

To add bullets to your report:

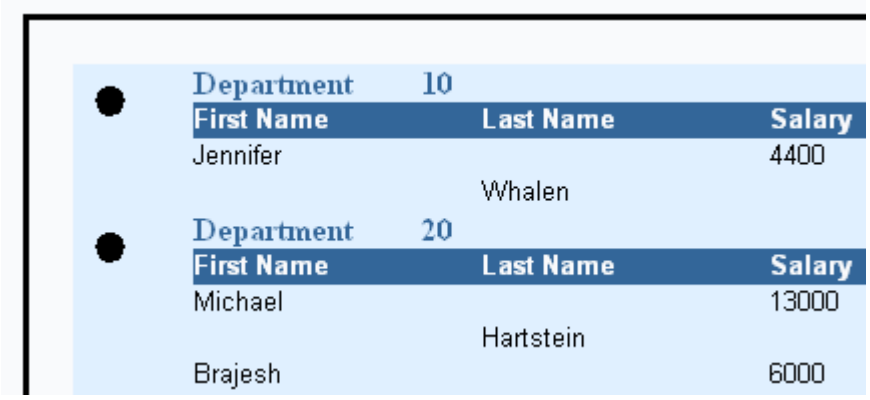
1. In the Paper Layout view, make sure you have enough space next to the B_DEPARTMENT_ID field where you can add a bullet. The bullet needs to be within the R_G_DEPARTMENT_ID repeating frame so that it displays for each record.
2. Click the Ellipse tool in the tool palette.
3. Press the SHIFT key, then draw an ellipse next to the boilerplate text "Department."

> **Note:** Pressing the SHIFT key while you draw the ellipse puts the tool in a "constrained" mode. When you create an ellipse in this mode, you can create a circle. Other tools are similar; for example, constraining the Rectangle tool creates a square. For more information on using these tools, refer to the *Reports Builder online help*.

4. You should now see a circle in the repeating frame. To make the bullet a solid color, select the circle, then click the Fill Color tool in the tool palette.
5. In the color palette that displays, choose **black**.
6. Run your report to the Paper Design view to see what it looks like.

> **Note:** When you're checking your layout, make sure that the ellipse displays within the R_G_DEPARTMENT_ID repeating frame.

Figure 18–8 Paper Design View of the Report with Bullets



7. Save your report as `graphics_<your initials>.rdf`.

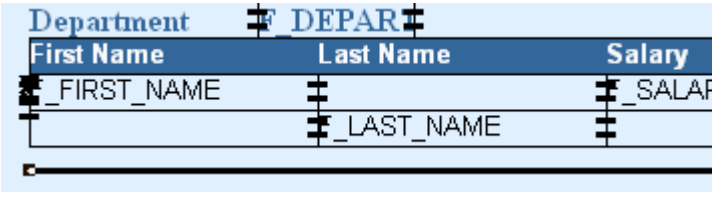
18.3.4 Display a line between each record

When you have a lot of data, you sometimes want to further distinguish each record, so that it's clear how the information is related. In this section, you will add a horizontal line to the layout that will display between each department record.

To add a line separator:

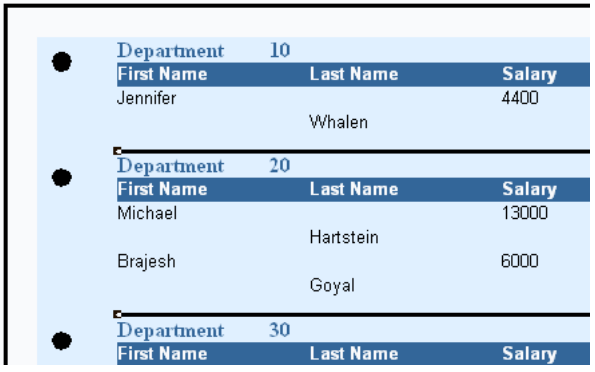
1. In the Paper Layout view, click the Line tool in the tool palette.
2. Press the SHIFT key, then draw a line in the R_G_DEPARTMENT_ID frame beneath the fields, like this:

Figure 18–9 Paper Layout View with Line Separator



3. Format the line the way you want to appear. In our example, we chose Format > Line Width > 2 pt. Then, we changed the color to black.
4. Run your report to the Paper Design view. It should now looks something like this:

Figure 18–10 Paper Design View of Report with Line Separator



5. Save your report as `graphics_<your initials>.rdf`.

18.3.5 Add a title to your report

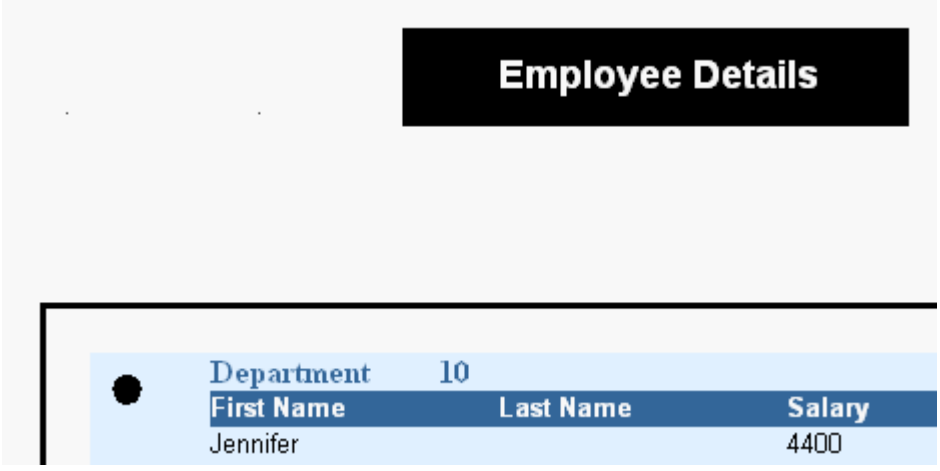
The steps in this section will show you how to use the tools in the tool palette to create a title for your report. You will use the Fill Color to create a black background and white text, as well as arrange your items so they display properly.

To add a title to the report:

1. In the Paper Layout view, click the Text tool in the tool palette.
2. Draw a rectangle where you want the title to display. We drew the rectangle above the repeating frames, so that the title would display once, above all the data.
3. In the new rectangle object, type a title, such as "`Employee Details`."
4. Click the Fill Color tool in the tool palette, and choose **black**.

5. Click the Text Color tool and choose **white**.
6. Click the Rectangle tool, and draw a rectangle around the text.
7. When you are done, click the Fill Color tool and choose **black**. Notice how you don't see the text anymore.
8. In the Object Navigator, find the text object you created under **Body**. You can tell which objects are text objects by the **A** icon next to the name. In our case, the text object is **B_3**.
9. Once you've selected the object, click the title bar of the Paper Design view to select the Paper Design view.
10. Choose **Layout > Bring to Front**.
11. Run your report to the Paper Design view. It should now look something like this:

Figure 18–11 Paper Design View of Report with Title



12. Save your report as `graphics_<your initials>.rdf`.

18.4 Summary

Congratulations! You have successfully created a paper report and modified the text color and fill colors, as well as added graphics. You now know how to:

- use the Report Wizard to create a simple report definition.
- modify the layout of the objects in your report.
- add a border to the report.
- add bullets to each record.
- add a horizontal separator between each record.
- add a title to your report using fill colors and layout features.

For more information on any of the wizards, views, or properties used in this example, refer to the *Reports Builder online help*, which you can access in two ways:

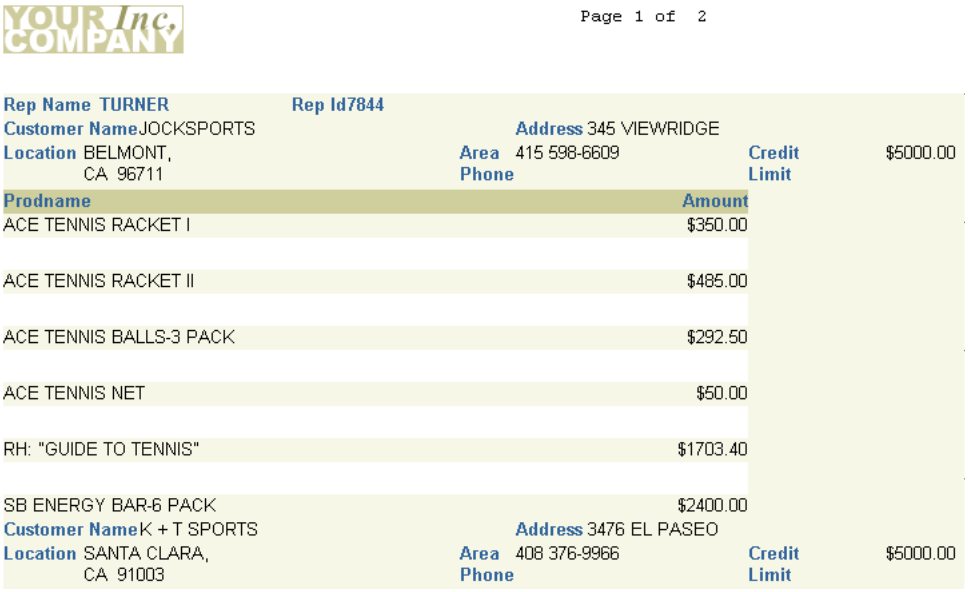
- From the Oracle Technology Network (`http://otn.oracle.com/products/reports/`), click **Documentation** and navigate to the *Reports Builder Online Help* for the most recent, hosted online help.
- From Reports Builder, choose **Help > Help Contents**.

> **Note:** You can replace your Reports Builder online help with the most recent update by downloading the latest online help set available on the Oracle Technology Network (http://otn.oracle.com/docs/products/reports/). Instructions for replacing your help file are included in the readme.txt in the download file.

19

Building a Report that Renumbers Pages by Repeating Frame

Figure 19–1 Pages renumbered by repeating frame report output



YOUR Inc. COMPANY

Page 1 of 2

Rep Name TURNER **Rep Id** 7844
Customer Name JOCKSPORTS **Address** 345 VIEWRIDGE
Location BELMONT, CA 96711 **Area Phone** 415 598-6609 **Credit Limit** $5000.00

Prodname	Amount
ACE TENNIS RACKET I	$350.00
ACE TENNIS RACKET II	$485.00
ACE TENNIS BALLS-3 PACK	$292.50
ACE TENNIS NET	$50.00
RH: "GUIDE TO TENNIS"	$1703.40
SB ENERGY BAR-6 PACK	$2400.00

Customer Name K + T SPORTS **Address** 3476 EL PASEO
Location SANTA CLARA, CA 91003 **Area Phone** 408 376-9966 **Credit Limit** $5000.00

This report numbers pages using the format "Page *X* of *Y* Pages". The first number (*X*) corresponds to the current page for each parent record (i.e., each sales representative). This page number is reset to ”1” for each sales representative, thus tracking the statistics of each representative separately.

The second number (*Y*) is the total pages required to display the customer information for each sales representative. This field is also reset with each new parent record.

Important Note: The steps described here explain how to produce the "Page *X* of *Y* Pages" for the paper layout and have no effect on the Web source, because the Web source does not have pagination.

Concepts

- The Page Numbering dialog box enables you to specify that the page number either reset at a particular repeating frame or increment over the entire report.
- However, if you decide to use a page total that repeats on several pages, as with the second number in the example report, you need to be aware of certain rules that apply when formatting a field that contains a count of other objects. To be able to display such a value, called a page–dependent reference, on the first page—or any page prior to the final page—of the report, Reports Builder must format the field that will contain the value before it knows the value. As a result, any field displaying a value of this type must be fixed, so that Reports Builder knows how much space to reserve for it when formatting the report. Even if you set the Horizontal Elasticity property or Vertical Elasticity property to *Expand*, *Contract*, or *Variable* for such a field, Reports Builder considers it fixed. When you create such a field, you must ensure that it is large enough to contain the value.
- An alternative is to hide the field and reference it from a boilerplate object. When using this method, the field can be variable, but the boilerplate remains fixed, so be sure to create a boilerplate object of sufficient size to hold the field.
- Some values that are treated in this fashion by Reports Builder are &TOTAL LOGICAL PAGES, &TOTAL PANELS, and &TOTAL PHYSICAL PAGES, and any summaries or formulas that reference them. Summaries with either a Compute At property or Reset At property setting of *Page* and formulas that reference them are also considered page–dependent references. For a complete list of such values and a comprehensive discussion of Reports Builder's treatment of them, see the topics "Vertical Elasticity property" or "Horizontal Elasticity property" in the *Reports Builder online help*.

Data Relationships

- As illustrated in the screenshot at the beginning of this chapter, this report contains both a master/detail relationship and a break. This results in two

levels of data differentiation. The top level (the break) will determine when page numbers reset.

Layout

- This report uses the master/detail layout style, with modifications. In addition, you'll create two hidden fields for the page numbers, then reference them in a boilerplate object. This will ensure that the field displaying the total number of pages is the proper size. Then you'll check the Maximum Records Per Page property setting for the break group's repeating frame to ensure that your pages are numbered correctly.

Example Scenario

Suppose that you want to create a page number in the margin of a report in the format "Page *X* of *Y* Pages". The page number *X* is incremented for each page produced by customer sales for one sales representative at a time. *Y* represents the total number of pages for the sales representative of that sales grouping and is reset for the next sales representative.

To see a sample report that renumbers pages by repeating frame, open the examples folder named `pagenum`, then open the Oracle Reports example called `pagenum.rdf`. For details on how to open it, see "Accessing the example reports" in the Preface.

Table 19–1 Features demonstrated in this example

Feature	Location
Create a data model and a group above layout	Section 19.2, "Create a data model and a group above layout"
Add a second query to group the output for each sales representative by customer	Section 19.3, "Add a second query"
Redefault the layout to use the additional query data	Section 19.4, "Redefault the layout"
Set properties and format the fields of your report	Section 19.5, "Set properties and format fields"
Create new fields for the page numbers of the report	Section 19.6, "Create new fields"
Reference the page number fields	Section 19.7, "Reference fields"

19.1 Prerequisites for this example

- To build the example in this chapter, you must have access to the Summit Sporting Goods schema, which we've provided on the Oracle Technology Network (`http://otn.oracle.com/products/reports/`). To download the SQL scripts that install the schema, go to the Documentation page on OTN and follow the instructions provided on the Web page. Only the EMP, CUSTOMER, and SALES tables are required.
- The HTML view of the report that displays the page number technique must be deployed via a report server using the report in JSP or RDF format and HTML as the destination format. For example:

```
http://machine_host:port/reports/rwservlet?report=pgnum.rdf
&userid=scott/tiger@DB9i&destype=cache&desformat=html&server=rep9i_server
```

19.2 Create a data model and a group above layout

When you are creating a single-query report, such as this one, you can use the Report Wizard to create the data model and layout simultaneously.

To create a data model and group above layout:

1. Launch Reports Builder (or, if already open, choose **File > New > Report**).
2. In the Welcome or New Report dialog box, select **Use the Report Wizard**, then click **OK**.
3. If the Welcome page displays, click **Next**.
4. On the Report Type page, select **Create both Web and Paper Layout**, then click **Next**.
5. On the Style page, type a **Title** for your report, select **Group Above**, then click **Next**.
6. On the Data Source page, click **SQL Query**, then click **Next**.
7. On the Data page, enter the following SELECT statement in the **Data Source definition** field:

```
SELECT ENAME, REPID, CUSTID, NAME, ADDRESS, TRIM(CITY)||',
'||STATE||'  '||ZIP LOCATION,  AREA||' '||PHONE, CREDITLIMIT
FROM EMP, CUSTOMER
WHERE EMPNO = REPID
ORDER BY REPID
```

Note: You can enter this query in any of the following ways:

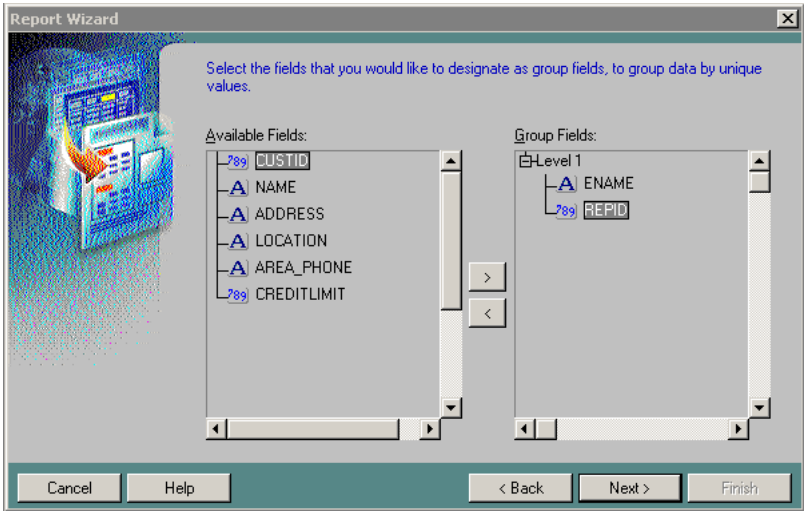
- Copy and paste the code from the provided text file called `pagenum_code.txt` into the **Data Source definition** field.
- Click **Query Builder** to build the query without entering any code manually.
- Type the code in the **Data Source definition** field.

8. Click **Next**.

Note: If you are not already connected to a database, you will be prompted to connect to the database when you click **Next**. Ensure that you connect to a database that has the appropriate schema for this example. Section 19.1, "Prerequisites for this example" describes the sample schema requirements for this example.

9. On the Groups page, click **ENAME** and **REPID** in the **Available Fields** list and click the right arrow (**>**) to move these fields to the **Group Fields** list, then click **Next**.

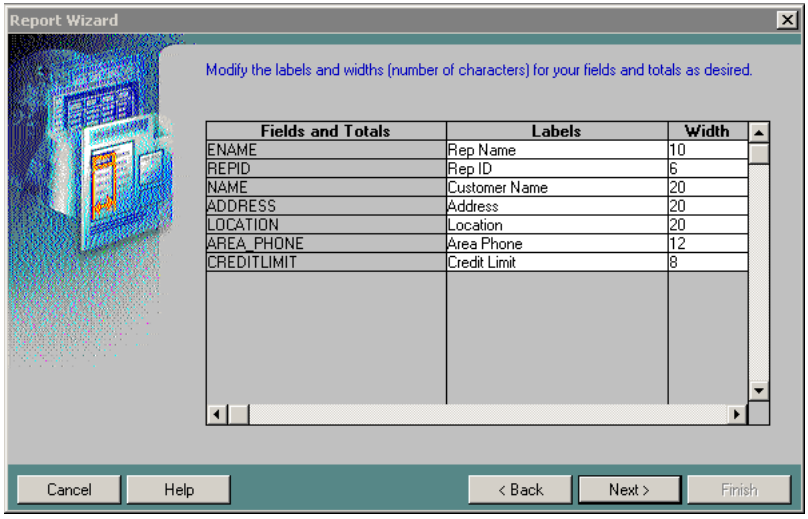
Figure 19–2 Selecting group fields in the Report Wizard



10. On the Fields page, click the double right arrows (>>) to move all of the fields to the **Displayed Fields** list, then click **CUSTID** and click the left arrow (<) to move it back to the **Available Fields** list, then click **Next**.
11. On the Totals page, click **Next**.
12. On the Labels page, change the labels and field widths as follows, then click **Next**:

Fields	Labels	Width
ENAME	`Rep Name`	`10`
REPID	`Rep ID`	`6`
NAME	`Customer Name`	`20`
ADDRESS	`Address`	`20`
LOCATION	`Location`	`20`
AREA_PHONE	`Area Phone`	`12`
CREDITLIMIT	`Credit Limit`	`8`

Figure 19–3 Specifying labels and widths in the Report Wizard

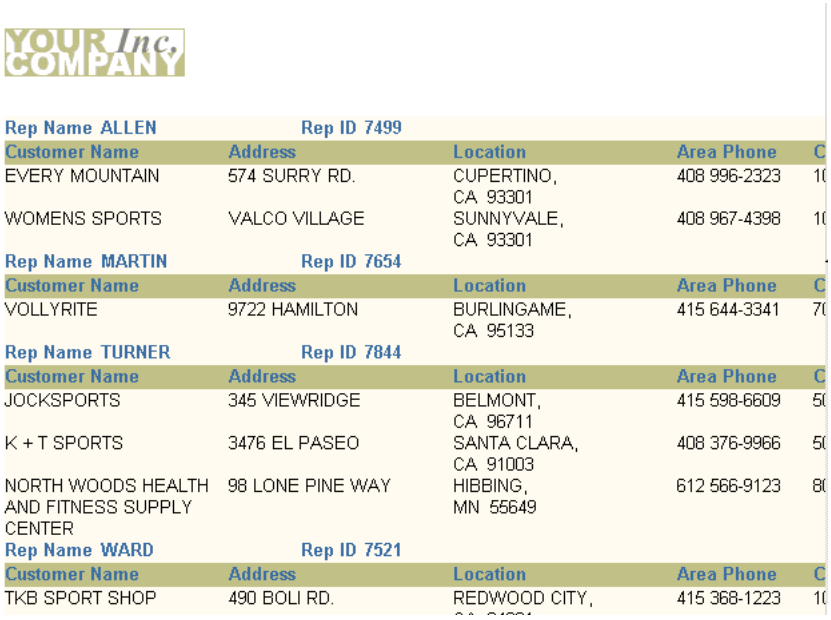


13. On the Template page, click **Finish** to display your report output in the Paper Design view.

19.3 Add a second query

In the Paper Design view, the data is grouped by the Sales Representative with customer data listed below each Sales Representative.

Figure 19–4 Report layout in the Paper Design view



All the data will display on one page and the individual sales transactions for each customer is not yet available. You will need an additional query in the Data Model to accomplish this.

To add a new query to the report:

1. Click the Data Model button in the toolbar.
2. In the Data Model view, click the SQL Query tool in the tool palette, then click in the main area (canvas region) of the window to the right of query **Q_1**.
3. In the SQL Query Statement dialog box, type the SELECT statement for the query:

```
SELECT CUSTID,PRODNAME,AMOUNT
FROM SALES
ORDER BY CUSTID
```

4. Click **OK** to display the new query in the Data Model view.
5. In the Data Model view, click the Data Link tool in the tool palette.

6. Using the cross hairs of the cursor, click **CUSTID** in the **G_NAME** group and drag the cursor to **custid1** in the **G_custid1** group, then release the mouse button to link the **CUSTID** and **custid1** fields between the two groups.
7. Position the Report Editor window to display it alongside the Object Navigator so that you can view both windows simultaneously.
8. In the Object Navigator, under the **Data Model** node, expand the **Groups** node. Then, under the **Paper Layout** node, expand the **Main Section** node. Compare these two structures and note that even though additional data is made available in the Data Model by adding query **Q_2**, the Paper Layout will not use this additional data unless modified to do so. We will do this next.

19.4 Redefault the layout

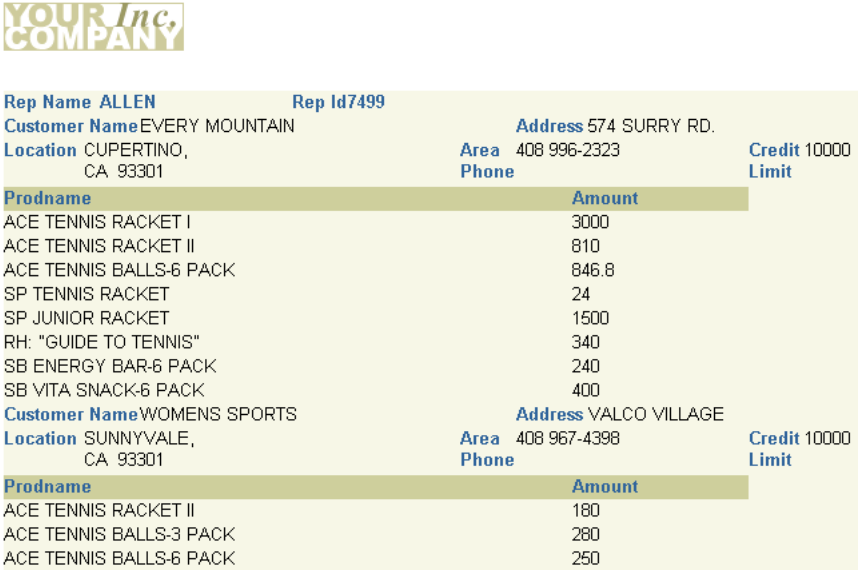
To redefault the layout to use the additional query data:

1. In the Object Navigator, click the report name.
2. Choose **Tools > Report Wizard** to re-enter the wizard.
3. In the Report Wizard, on the **Groups** page, click **G_custid1** in the **Available Groups** list and click **Down** specify the Print Direction and move this group to the **Displayed Groups** list.
4. On the **Fields** page, click **prodname** and **amount** in the **Available Fields** list and click the right arrow (>) to move these fields to the **Displayed Fields** list.
5. On the **Labels** page, change the labels and field widths as follows:

Fields	Labels	Width
prodname	`Product Name`	`30`
amount	`Amount`	`9`

6. On the **Template** page, click **Finish** to display your report output in the Paper Design view. It should look something like this:

Figure 19–5 Revised report layout in Paper Design view



Note that the data is still grouped by the Sales Representative with customer data listed below each Sales Representative. But now, the individual products sold to the customers are listed below each customer. The report now has multiple pages.

19.5 Set properties and format fields

To set properties and format the fields of your report:

1. In the Object Navigator, click the **Paper Layout** node and click the Expand All button in the toolbar.
2. Double-click the repeating frame icon next to the **R_G_ENAME** node to display the Property Inspector, and set properties:
   - Under **Repeating Frame**, set the Maximum Records per Page property to 1.
3. In the Object Navigator, double-click the repeating frame icon next to the **R_G_CUSTID1** node to display the Property Inspector, and set properties:

- Under **Repeating Frame**, set the Vert. Space Between Frames property to 0.25 to specify 1/4 inch of vertical space between instances of the repeating frame; that is, between each customer record.
- Under **General Layout**, set the Page Protect property to Yes to specify that the first instance of the repeating frame (that is, the first customer record) must all fit on the same logical page.

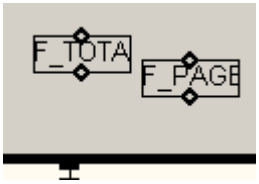
4. In the Object Navigator, click the **F_CREDITLIMIT** node, then Ctrl-click **F_AMOUNT** to select both nodes.
5. Click the Paper Design button in the toolbar to make the Paper Design view the active window, with the **F_CREDITLIMIT** and **F_AMOUNT** fields selected, and make the format changes (you can also click the corresponding buttons in the toolbar):
   - choose **Format > Number > Currency**
   - choose **Format > Number > Add Decimal Place**
   - choose **Format > Number > Add Decimal Place** (again, for two decimal places)
   - choose **Format > Justify > Right**
6. In the Paper Design view, click the Amount label (column heading), and choose **Format > Justify > Right**.

19.6 Create new fields

To create new fields for the page numbers of the report:

1. Click the Paper Layout button in the toolbar to display the Paper Layout view.
2. In the Paper Layout view, click the Edit Margin button in the toolbar to display the margin areas of your report. You will be creating two page number fields in the top margin, as shown here:

Figure 19–6 Page number source fields in top margin of report



3. Click the Field tool in the tool palette and draw a field about two characters long in the center of your margin.
4. Double-click the new field object (**F_1**) to display the Property Inspector, and set properties:
   - Under **General Information**, set the Name property to F_TOTAL_PAGENO.
   - Under **Field**, set the Source property to Total Pages, and set the Visible property to No.
   - Under **General Layout**, set the Horizontal Elasticity property to Variable.

     Hiding both this field and the field you'll create below ensures that they'll appear only where you reference them. Changing the horizontal sizing to Variable ensures that the fields can expand to accommodate large page ranges.
   - Under **Field**, click the Page Numbering property field to display the Page Numbering dialog box:

     in the **Reset at** list, click **R_G_ENAME**, and verify that the **Start at** and **Increment by** values are both *1*.
5. Close the Property Inspector.
6. In the Paper Layout view, click the F_TOTAL_PAGENO field.
7. Choose **Edit > Copy**, then choose **Edit > Paste** to create a new field with the same dimensions as F_TOTAL_PAGENO next to F_TOTAL_PAGENO.
8. Drag the new field object (F_TOTAL_PAGENO1) to an open area of the margin.
9. Double-click the new field object (F_TOTAL_PAGENO1) to display its Property Inspector, and make the following changes:

- Under **General Information**, set the Name property to F_PAGENO.
- Under **Field**, set the Source property to Page Number.

  Because it is a duplicate of F_TOTAL_PAGENO, the Horizontal Elasticity and Visible properties are already set as desired.
- Under **Field**, click the Page Numbering property field to display the Page Numbering dialog box.

  In the **Reset at** list, click **R_G_ENAME**, and verify that the **Start at** and **Increment by** values are both *1*.

10. Close the Property Inspector.

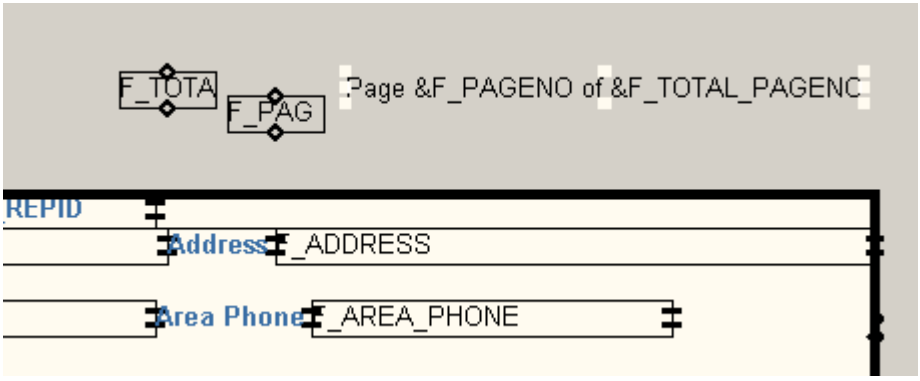
Next, you will reference these fields in a text box to display page numbers in the format "Page X of Y Pages".

19.7 Reference fields

To reference the page number fields:

1. In the Paper Layout view, click the Line Color tool in the tool palette.
2. In the color palette, click **No Line**.
3. Click the Text tool in the tool palette.
4. Position the cursor in the top margin where you would like the page number to appear, and click to create a boilerplate object.
5. In the new boilerplate object, type `Page &F_PAGENO of &F_TOTAL_PAGENO`.

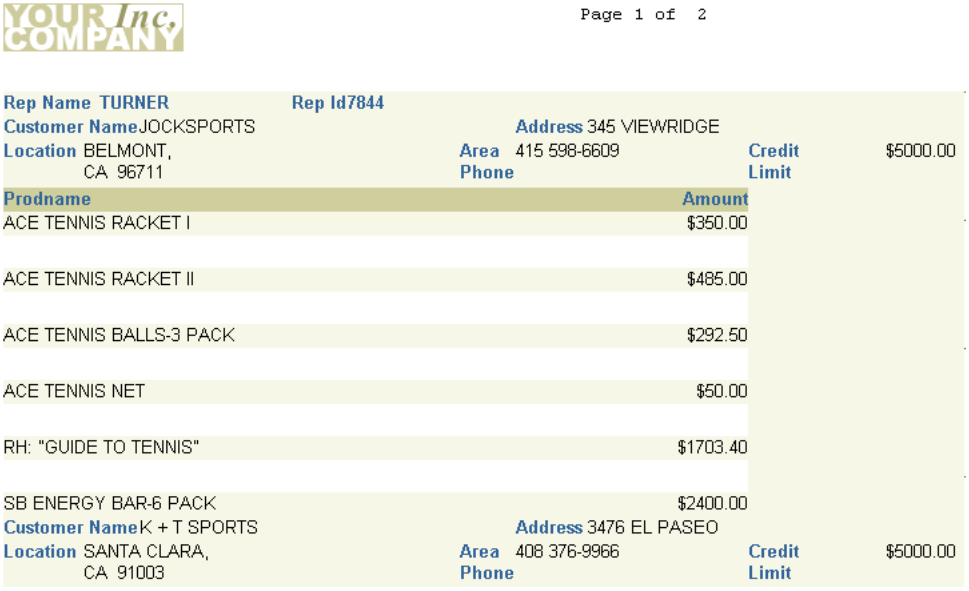
Figure 19–7 Page number source and reference fields in top margin of report



In the report output, the current page number will appear where you reference the field F_PAGENO, while the total number of pages for each master will appear where you reference the field F_TOTAL_PAGENO. If you wish, you can resize the page number boilerplate text object to lengthen it to ensure that no values for F_TOTAL_PAGENO get truncated.

6. Save and run your report. Your final report output should look something like this:

Figure 19–8 Final pages renumbered by repeating frame report output



19.8 Summary

Congratulations! You have successfully created a report that renumbers pages by repeating frame. You now know how to:

- create a data model with a group above layout.
- add a second query to create a multiquery report, and redefault the layout.
- set properties and format fields.
- create and reference a page number boilerplate object in your layout.

For more information on any of the wizards, views, or properties used in this example, refer to the *Reports Builder online help*, which you can access in two ways:

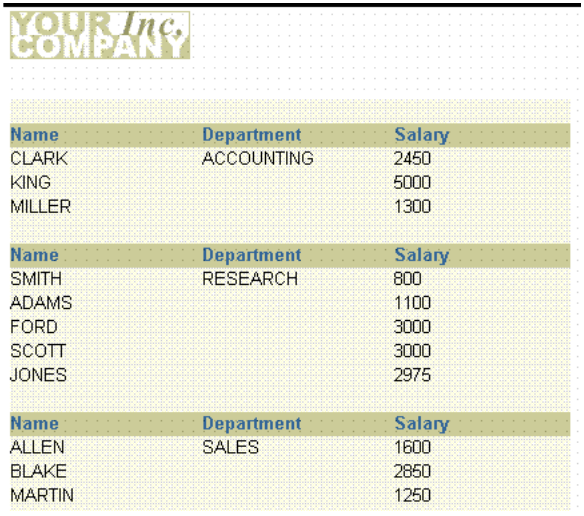
- From the Oracle Technology Network (`http://otn.oracle.com/products/reports/`), click **Documentation** and navigate to the *Reports Builder Online Help* for the most recent, hosted online help.
- From Reports Builder, choose **Help > Help Contents**.

Note: You can replace your Reports Builder online help with the most recent update by downloading the latest online help set available on the Oracle Technology Network (http://otn.oracle.com/docs/products/reports/). Instructions for replacing your help file are included in the readme.txt in the download file.

20

Building an Intermixed Fields Report

Figure 20–1 Group with master record in the middle report output



Normally, a group (break) field appears to the left of (in group left report) or above (in group above report) its related fields. In this example, the group field appears between its related fields.

Concepts

- To create this type of report you need to build a data model with two groups.
- A formula column in the detail group returns the value of the group column. The formula column is moved to the middle position of its group.

- The Report Wizard is used to create a group above layout. In the Paper Layout view, the break column is deleted from its defaulted position.
- A format trigger is created to suppress redundant printing of the formula column group value in the detail group.

Example Scenario

Suppose that you want to create a report that displays salary data by employee and department. Your users have indicated to you that they prefer the group value (i.e., department name) to appear in the middle of the layout with employee names to the left and employee salaries to the right.

To see a sample report that intermixes fields, open the examples folder named `intermix`, then open the Oracle Reports example named `intermixb.rdf`. For details on how to access it, see "Accessing the example reports" in the Preface.

Table 20–1 Features demonstrated in this example

Feature	Location
Create a data model and layout in the Report Wizard	Section 20.2, "Create a data model and a layout"
Add a formula column to the detail group	Section 20.3, "Add a formula column"
Add a field in the Report Wizard	Section 20.4, "Add a field"
Remove a redundant field in the Paper Design view	Section 20.5, "Remove a redundant field"
Suppress values in a Format Trigger	Section 20.6, "Suppress redundant values"

20.1 Prerequisites for this example

To build the example in this chapter, you must have access to the EMP and DEPT schema, which is provided by default with the Oracle database. The userid and password for accessing this schema is scott/tiger.

20.2 Create a data model and a layout

When you are creating a single-query report, such as this one, you can use the Report Wizard to create the data model and layout simultaneously.

To create a data model and group above layout:

1. Launch Reports Builder (or, if already open, choose **File > New > Report**).
2. In the Welcome or New Report dialog box, select **Use the Report Wizard**, then click **OK**.
3. If the Welcome page displays, click **Next**.
4. On the Report Type page, select **Create Paper Layout Only**, then click **Next**.
5. On the Style page, type a **Title** for your report, select **Group Above**, then click **Next**.
6. On the Data Source page, click **SQL Query**, then click **Next**.
7. On the Data page, enter the following SELECT statement in the **Data Source definition** field:

```
SELECT DEPT.DNAME, EMP.ENAME, EMP.SAL
FROM DEPT, EMP
WHERE (EMP.DEPTNO = DEPT.DEPTNO)
```

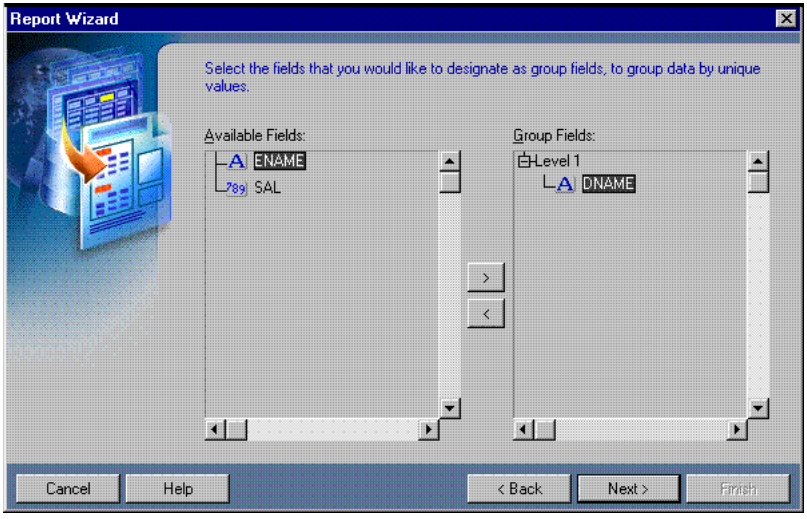
> **Note:** You can enter this query in any of the following ways:
>
> - Copy and paste the code from the provided text file called `intermxb_code.txt` into the **Data Source definition** field.
> - Click **Query Builder** to build the query without entering any code manually.
> - Type the code in the **Data Source definition** field.

8. Click **Next**.

> **Note:** If you are not already connected to a database, you will be prompted to connect to the database when you click **Next**. Ensure that you connect to a database that has the appropriate schema for this example. Section 20.1, "Prerequisites for this example" describes the sample schema requirements for this example.

9. On the Groups page, click **DNAME** in the **Available Fields** list and click the right arrow (>) to move this field to the **Group Fields** list.

Figure 20–2 Group page of Report Wizard

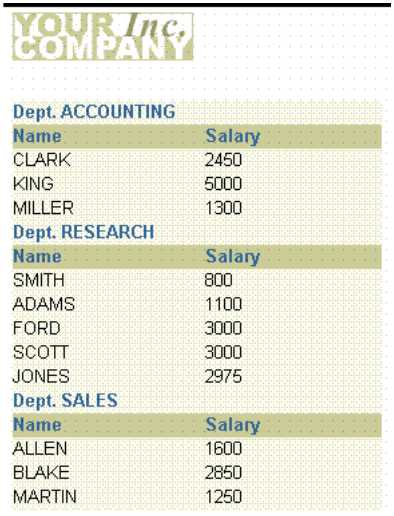


10. Click **Next**.
11. On the Fields page, click the double right arrows (>>) to move all of the fields to the **Displayed Fields** list, then click **Next**.
12. On the Totals page, click **Next**.
13. On the Labels page, change the labels as follows, then click **Next**:

Fields	Labels
DNAME	`Dept.`
ENAME	`Name`
SAL	`Salary`

14. On the Template page, click **Finish** to display your report output in the Paper Design view. It should look something like this:

Figure 20–3 Paper Design view for intermixing fields report



20.3 Add a formula column

In order to have the department names appear in the center of this group report, it is simplest to have a column in the detail group for the department names. To achieve this result, you must create a formula column in the detail group that will display the department names.

Add a formula column

1. Click the Data Model tool in the toolbar to display the Data Model view.
2. In the Data Model view, resize the **G_ENAME** group to be large enough to contain a third column.
3. Click the Formula Column tool in the tool palette.
4. Click in the **G_ENAME** group to create a formula column.
5. Double-click the new formula column object (**CF_1)** to display the Property Inspector, and set properties:
   - Under **General Information**, set the Name property to DEPARTMENT.
   - Under **Column**, set the Datatype property to Character, and set the Width property to 14.

- Under **Placeholder/Formula**, double-click the PL/SQL Formula property field to display the PL/SQL Editor.

6. In the PL/SQL Editor, at the flashing cursor (after the word `begin`), type the following:

```
return(:dname);
```

7. Click **Compile**.
8. Click **Close**.
9. Click the title bar of the Report Editor to make it the active window again.

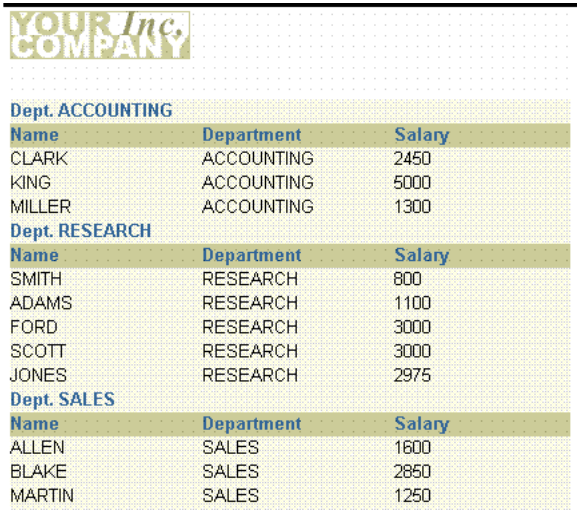
20.4 Add a field

Now that you have added the formula column to the data model, you need to add a corresponding field to display it. You can easily do this by invoking the re-entrant Report Wizard.

To add a field:

1. Choose **Tools > Report Wizard**.
2. On the **Report Type** page, select **Create Paper Layout Only**.
3. On the **Fields** page:
   - Click **DEPARTMENT** in the **Available Fields** list and click the right arrow (**>**) to move it to the **Displayed Fields** list.
   - Click and drag **DEPARTMENT** in the **Displayed Fields** list until it is located between **ENAME** and **SAL**.
4. Click **Finish** to preview your report output in the Paper Design view. It should look something like this:

Figure 20–4 Paper Design view of the intermixing fields report



Dept. ACCOUNTING		
Name	**Department**	**Salary**
CLARK	ACCOUNTING	2450
KING	ACCOUNTING	5000
MILLER	ACCOUNTING	1300
Dept. RESEARCH		
Name	**Department**	**Salary**
SMITH	RESEARCH	800
ADAMS	RESEARCH	1100
FORD	RESEARCH	3000
SCOTT	RESEARCH	3000
JONES	RESEARCH	2975
Dept. SALES		
Name	**Department**	**Salary**
ALLEN	SALES	1600
BLAKE	SALES	2850
MARTIN	SALES	1250

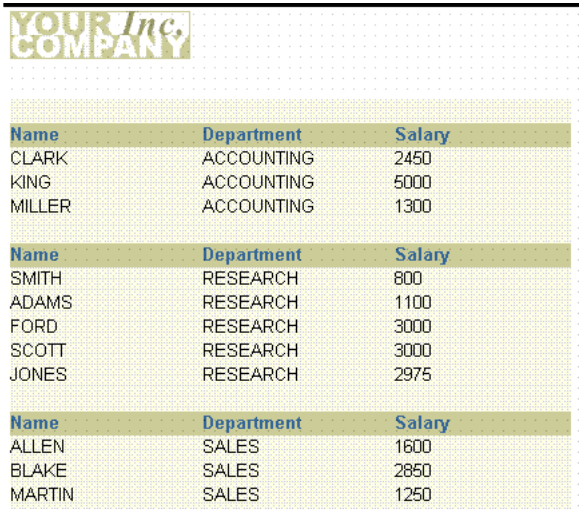
20.5 Remove a redundant field

After you have added a field for the formula column, you will notice that the department values appear in two places. To eliminate this redundancy, you need to remove the master field and its heading.

To remove a field and its label:

1. In the Paper Design view, Shift-click on the first instance of the label **Dept.** and the value next to it.
2. Choose **Edit > Delete**.

Figure 20–5 Output displayed in Paper Design view



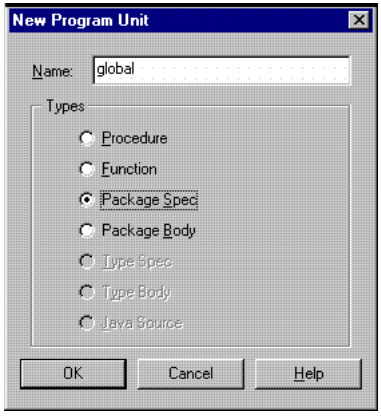
20.6 Suppress redundant values

Notice in your output that the department values are properly positioned, but they repeat for every record in the department. What you really want is for the department values to appear once for each department. To accomplish this task, you will first create a global variable to be used in comparing the current department value to the previous one. You will then write a Format Trigger to determine which values to suppress based upon the comparison within each department's records.

To create a global variable:

1. In the Object Navigator, click the **Program Units** node.
2. Click the Create button in the toolbar. The New Program Unit dialog box appears.
3. Type `global` in the **Name** field and select **Package Spec**.

Figure 20–6 New Program Unit dialog box



4. Click **OK**.
5. In the PL/SQ Editor, type the following PL/SQL:

```
PACKAGE global IS
  prev_val varchar2(14);
END;
```

6. Click **Compile**
7. Click **Close**.

To add the format trigger:

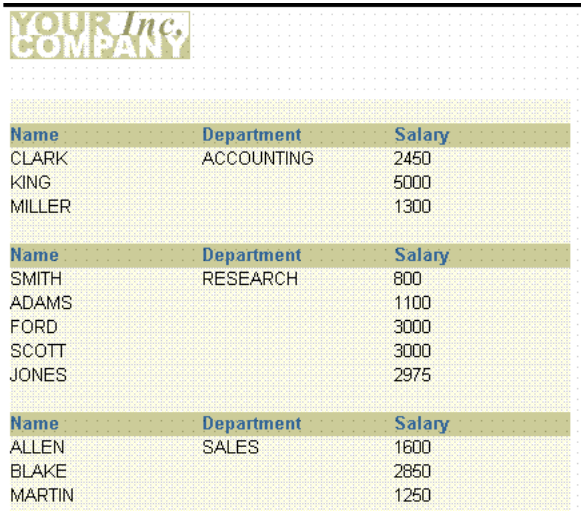
1. In the Object Navigator, type `F_DEPARTMENT` in the **Find** field to select it.
2. Double-click the properties icon to the left of **F_DEPARTMENT** to display the Property Inspector, and set properties:
   - Under **Advanced Layout**, click the Format Trigger property field to display the PL/SQL Editor.
3. In the PL/SQL Editor, use the template to enter the following PL/SQL code:

```
function F_DEPARTMENTFormatTrigger return boolean is
begin
```

```
If global.prev_val = :department then
  return(false);
  Else
    global.prev_val := :department;
    return(true);
  END IF;
end;
```

4. Click **Compile**.
5. Click **Close**.
6. Click the title bar of the Report Editor to make it the active window. Return to the Paper Design view if you are not already there. Notice the change in your report output.

Figure 20–7 Final report output displayed in the Paper Design view



20.7 Summary

Congratulations! You have successfully created an intermixed fields report. You now know how to:

- create a data model and layout in the Report Wizard.

- add a formula column to the detail group.
- add a field in the Report Wizard.
- remove a redundant field in the Paper Design view.
- suppress values in a Format Trigger.

For more information on any of the wizards, views, or properties used in this example, refer to the *Reports Builder online help*, which you can access in two ways:

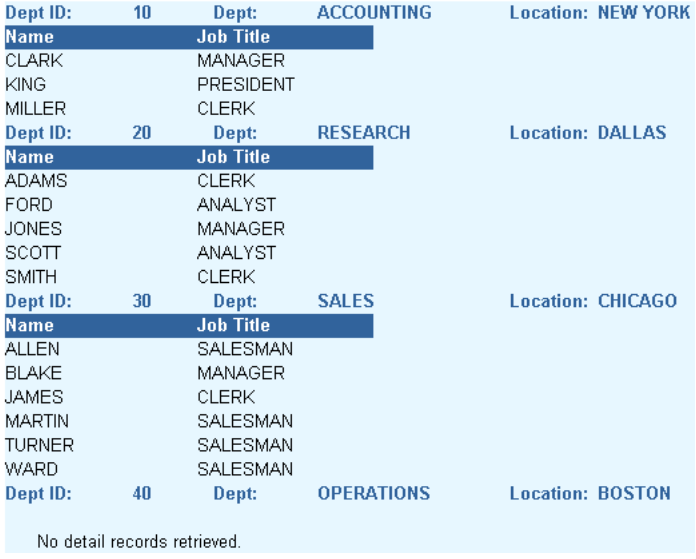
- From the Oracle Technology Network (`http://otn.oracle.com/products/reports/`), click **Documentation** and navigate to the *Reports Builder Online Help* for the most recent, hosted online help.
- From Reports Builder, choose **Help > Help Contents**.

> **Note:** You can replace your Reports Builder online help with the most recent update by downloading the latest online help set available on the Oracle Technology Network (http://otn.oracle.com/docs/products/reports/). Instructions for replacing your help file are included in the readme.txt in the download file.

21

Building a Report that Suppresses Labels

Figure 21–1 Suppressed labels report output



Dept ID:	10	Dept:	ACCOUNTING	Location:	NEW YORK
Name		Job Title			
CLARK		MANAGER			
KING		PRESIDENT			
MILLER		CLERK			
Dept ID:	20	Dept:	RESEARCH	Location:	DALLAS
Name		Job Title			
ADAMS		CLERK			
FORD		ANALYST			
JONES		MANAGER			
SCOTT		ANALYST			
SMITH		CLERK			
Dept ID:	30	Dept:	SALES	Location:	CHICAGO
Name		Job Title			
ALLEN		SALESMAN			
BLAKE		MANAGER			
JAMES		CLERK			
MARTIN		SALESMAN			
TURNER		SALESMAN			
WARD		SALESMAN			
Dept ID:	40	Dept:	OPERATIONS	Location:	BOSTON

No detail records retrieved.

The report you will build in this chapter is a master/detail report that fetches a master record that has no associated details. In the sample output above, notice how the field labels for Department 40 do not display because no detail records were found. In this chapter, you will learn how to suppress the detail information for a single record, but allow the other master/detail records to display.

Concepts

- A default master/detail report must print a detail label or field, even if there are neither fetched detail records nor values for user-created columns. This chapter demonstrates how to prevent these orphan labels from displaying in your report.

Data Relationships

- This report uses a master/detail data model. You'll also create a formula column in the detail group whose sole function is to return a value; this ensures the detail group contains at least one column that will always return a value each time a detail record is returned (as opposed to, for example, a null value).
- You'll create a summary to count the number of times this formula column returns a value. In the layout, you'll create a format trigger that references the summary to determine if the detail labels should be displayed. Doing so provides a reliable method for determining the existence of detail records.

Layout

- This report uses a Group Above layout style with modifications.

Example Scenario

This chapter will show you how to build a report that does not display field labels when there are no detail records. Instead, text will display that tells the user that no details records were retrieved.

To see a sample report that suppresses labels, open the examples folder named `suppresslabels`, then open the Oracle Reports example named `suppressinglabels.rdf`. For details on how to open it, see "Accessing the example reports" in the Preface.

Table 21–1 Features demonstrated in this example

Feature	Location
Manually create two queries and link them in the Data Model view.	Section 21.2, "Create the data model with two linked queries"
Add a formula column and a summary column in the Data Model view.	Section 21.3, "Create a formula column and a summary column"
Use the Report Wizard to create a simple layout.	Section 21.4, "Create the report layout"

Table 21–1 Features demonstrated in this example

Feature	Location
Add a format trigger that suppresses labels when no detail records are retrieved.	Section 21.5, "Add a format trigger to suppress labels"
Create boilerplate text.that displays when no records are displayed.	Section 21.6, "Add text to display when no records display"

21.1 Prerequisites for this example

To build the example in this chapter, you must have access to the EMP and DEPT schema, which is provided by default with the Oracle9*i* database. The user ID and password for accessing this schema is scott/tiger.

21.2 Create the data model with two linked queries

The steps in this section will show you how to build a simple data model with two queries in a master/detail relationship.

To create a data model

1. Launch Reports Builder (or, if already open, choose **File > New > Report**)
2. In the Welcome or New Report dialog box, select **Build a new report manually**, then click **OK**.
3. In the Data Model view that displays, click the SQL tool in the tool palette, then click in an open area of the Data Model view to display the SQL Query Statement dialog box.
4. In the **SQL Query Statement** field, type the first SELECT statement:

```
SELECT * FROM DEPT
ORDER BY DEPTNO
```

Note: You can enter this query in any of the following ways:

- Copy and paste the code from the provided text file called `suppresslabels_code.txt` into the **SQL Query Statement** field.
- Click **Query Builder** to build the query without entering any code manually.
- Type the code in the **SQL Query Statement** field.

Also note that if you have not installed the Pictures table into the sample schema, you will not be able to create this query.

---

5. Click **OK**.

Note: If you are not already connected to a database, you will be prompted to connect to the database when you click **Next**. Ensure that you connect to a database that has the appropriate schema for this example. Section 21.1, "Prerequisites for this example" describes the sample schema requirements for this example.

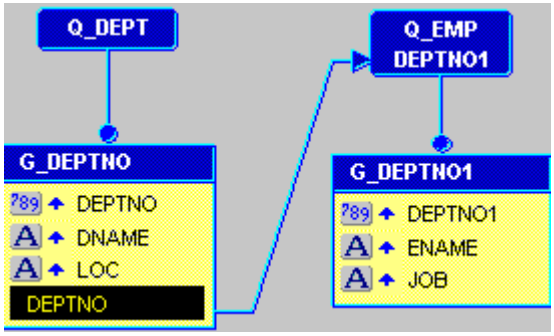
---

6. When the query displays in the Data Model view, right-click the query name (**Q_1**), then choose **Property Inspector** from the pop-up menu to set the following property:
   - Under **General Information**, set the Name property to Q_DEPT.
7. Follow the steps above to create another query named Q_EMP, with the following code:

```
SELECT DEPTNO, ENAME, JOB FROM EMP
ORDER BY ENAME
```

8. In the Data Model view, click the Data Link tool in the tool palette.
9. Drag a link between **DEPTNO** in Q_DEPT and **DEPTNO1** in Q_EMP. Your data model should now look like this:

Figure 21–2 Data Model view of the Suppress Labels report



10. Save your report as `suppresslabels_<your initials>.rdf`.

21.3 Create a formula column and a summary column

The steps in this section will show you how to add a formula column to the Q_EMP (or detail) query you built in the previous section that will return a value every time a detail record is returned. You will then add a summary column to the Q_DEPT (or master) query that will count the number of times this formula column returns a value.

Before you create either of these columns, you may want to expand the size of your groups for better visibility. To do so, select the yellow group box. Click the bottom frame, then drag it down about 0.25 inches. If you do this for both groups, you will have enough room to add your new columns.

21.3.1 Create a formula column in the detail query

This section will show you how to create a formula column that will return a single value.

To create a formula column:

1. In Data Model view, click the Formula Column tool in the tool palette.
2. Click in the **Q_EMP** query, under the **JOB** column to create a formula column.
3. Double-click the new formula column object (**CF_1***)* to display the Property Inspector, and set properties:

- Under **General Information**, set the Name property to DETAIL_VAL.
- Under **Placeholder/Formula**, click the PL/SQL Formula property field to display the PL/SQL Editor.

4. In the PL/SQL Editor, use the template to enter the following PL/SQL code:

```
function DETAIL_VALFormula return Number is
begin
     return(1);
end;
```

5. Click **Compile**.

   Tip: If you receive errors when compiling, compare your code against the code provided. You can also simply copy and paste the code from `suppresslabels_code.txt`.

6. When the code is compiled, click **Close**.

21.3.2 Create a summary column in the master query

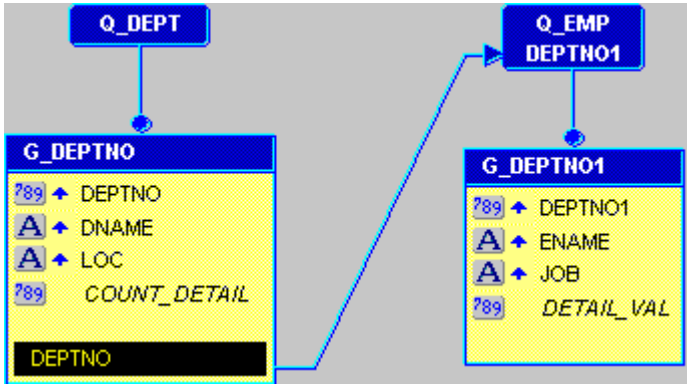
The steps in this section will show you how to create and customize a summary column in the master query that will depend on the information returned by the formula column.

To create a summary column:

1. In the Data Model view, click the Summary Column tool in the tool palette.
2. Click in the **Q_DEPT** query, under the **LOC** column to create a summary column.
3. Double-click the new summary column object (**CS_1**) to display the Property Inspector, and set properties:
   - Under **General Information**, set the Name property to COUNT_DETAIL.
   - Under **Summary**, set the Function property to Count, set the Source property to DETAIL_VAL, and set the Reset At property to G_DEPTNO.

Your data model now contains both the formula and summary columns, and should look like this:

Figure 21–3 Data Model view with formula and summary columns



4. Save your report as `suppresslabels_<your initials>.rdf`.

21.4 Create the report layout

In this section, you will create a default layout for your report. You will then add all the necessary layout objects for your checks.

21.4.1 Create the initial layout of your report

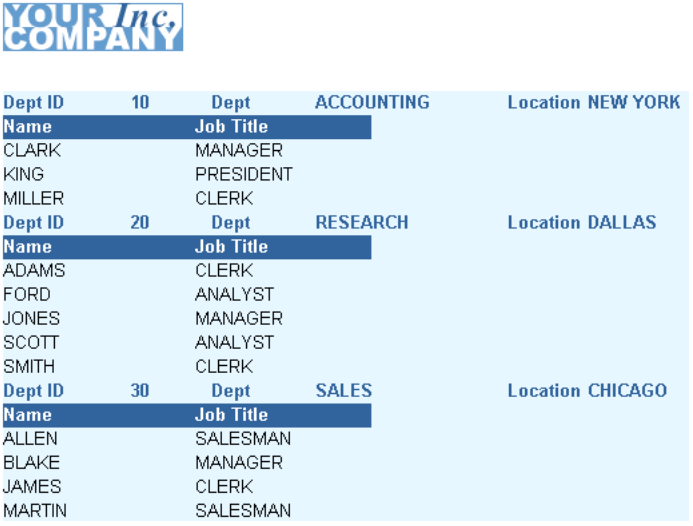
To create the initial layout:

1. In the Data Model view, right-click on the canvas, then choose **Report Wizard**.
2. In the Report Wizard, on the **Report Type** page, select **Create Paper Layout only**.
3. On the **Style** page, select **Group Above**.
4. On the **Groups** page, click the following groups in the **Available Groups** list and click **Down** to specify the Print Direction and move them to the **Displayed Groups** list:
   - **G_DEPTNO**
   - **G_DEPTNO1**
5. On the **Fields** page:

- Click the double right arrows (>>) to move all of the fields to the **Displayed Fields** list.
- In the **Displayed Fields** list, click **DEPTNO1** then click the left arrow (<) to move this field back to the **Available Fields** list.
- Do the same for the **COUNT_DETAIL** and **DETAIL_VAL** fields.

6. On the **Labels** page, make any desired changes to the labels.
7. On the **Template** page, select **Predefined Template** and click **Blue**, then click **Finish** to display your report output in the Paper Design view. It should look something like this:

Figure 21–4 Paper Design view for the suppress labels report



8. Save your report as `suppresslabels_<your initials>.rdf`.

21.5 Add a format trigger to suppress labels

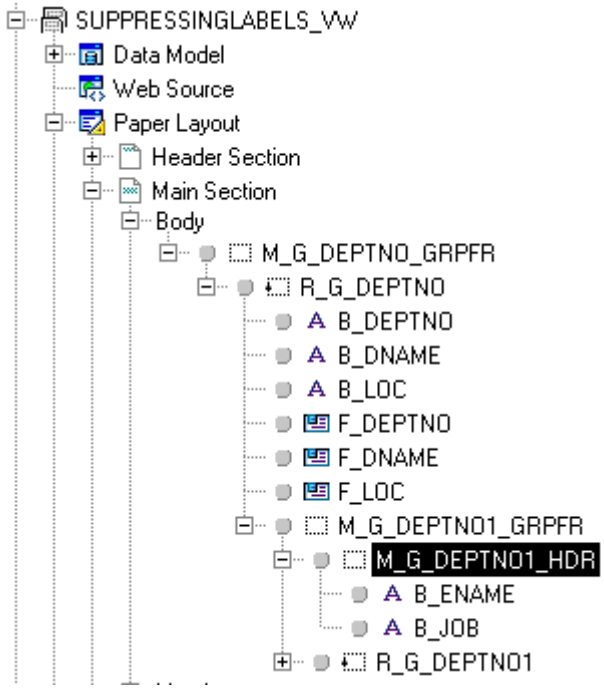
The steps in this section will show you how to add a format trigger to your report that will prevent labels from being displayed for records will null values.

To create a format trigger:

1. In the Object Navigator, under the **Paper Layout** node, navigate to **Main Section > Body > M_G_DEPTNO_GRPFR > R_G_DEPTNO > M_G_DEPTNO1_GRPFR > M_G_DEPTNO1_HDR**. Or, use the Find field in the Object Navigator to find M_G_DEPTNO1_HDR.

   Tip: See the image below for an example of where the frame is located in your Object Navigator. When you select the frame name in the Object Navigator, the corresponding frame will be selected in the Paper Layout view, as well.

Figure 21–5 Navigating to M_G_DEPTNO1_HDR



2. Double-click the properties icon next to **M_G_DEPTNO1_HDR** to display the Property Inspector, and set properties:
   - Under **Advanced Layout**, click the Format Trigger property field to display the PL/SQL Editor.
3. In the PL/SQL Editor, use the template to enter the following PL/SQL code:

```
function M_G_DEPTNO1_HDRFormatTrigger return boolean is
begin
  if :count_detail=0 then
     return (FALSE);
  else
     return (TRUE);
  end if;
end;
```

4. Click **Compile**.

   Tip: If you receive errors when compiling, compare your code against the code provided. You can also simply copy and paste the code from `suppresslabels_code.txt`.

5. When there are no compiling errors, click **Close**.
6. Save your report as `suppresslabels_<your initials>.rdf`.

21.6 Add text to display when no records display

The steps in this section will show you how to add boilerplate text to your report layout that will display when no records are retrieved.

To add boilerplate text:

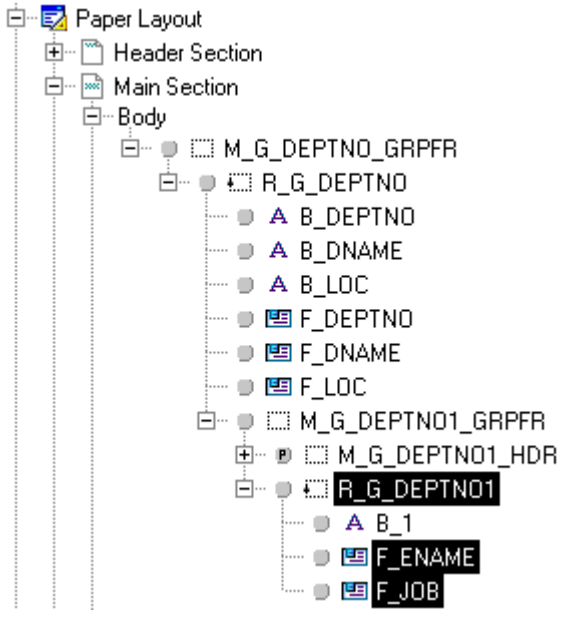
1. In the Paper Layout view, click the Confine Off and the Flex Off buttons in the toolbar.

   Note: To adjust the way the boilerplate objects display in your resulting report, you must turn off Confine and Flex modes before you create the objects.

2. Click the Text tool in the tool palette.
3. Draw a rectangle above the two fields **F_ENAME** and **F_JOB** to create a new boilerplate text object.
4. Click in the boilerplate text object, and type "`No detail records retrieved.`"
5. Select the text, then choose **Format > Font**.
6. In the Font dialog box, choose **Arial**, then click **OK**.
7. While the Paper Layout view still displays, click the Object Navigator and position the two windows so that you can see them side-by-side.
8. In the Object Navigator, navigate to **M_G_DEPTNO1_GRPFR**, and select these objects using CTRL-click:
   - **R_G_DEPTNO1**
   - **F_ENAME**
   - **F_JOB**

   Your Object Navigator should look like this:

Figure 21–6 Selected objects in the Object Navigator



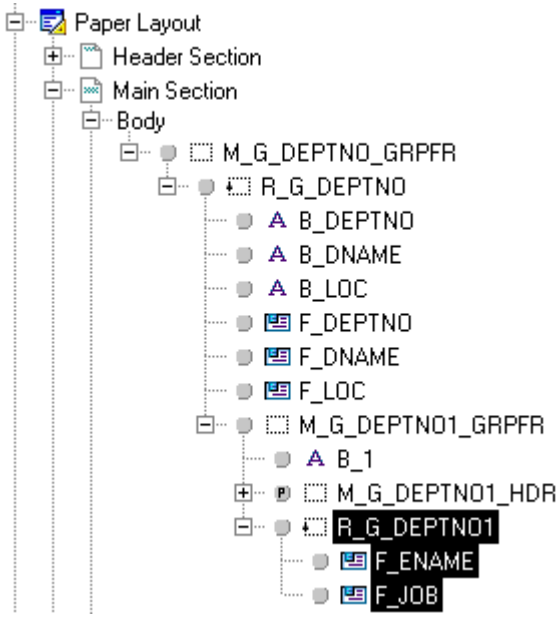
9. Choose **Layout > Bring to Front**.

> **Note:** If this menu option is greyed out, click the title bar of the Paper Layout view, but do not click on the canvas itself. This menu option is only available when the Paper Layout view is active.

By choosing this menu option, Reports Builder will display the records in front of the boilerplate text you just created. If there are no records, the boilerplate text will display.

The Object Navigator should now look like this:

Figure 21–7 Selected objects brought forward in the Object Navigator



Note: You'll notice that the boilerplate text, B_1, is now located directly under the parent frame, M_G_DEPTNO1_GRPFR. This placement means that the records displayed by the objects in the repeating frame, R_G_DEPTNO1, will display in front of the boilerplate text. The boilerplate text, which says that no detail records were retrieved, only displays when no records are present.

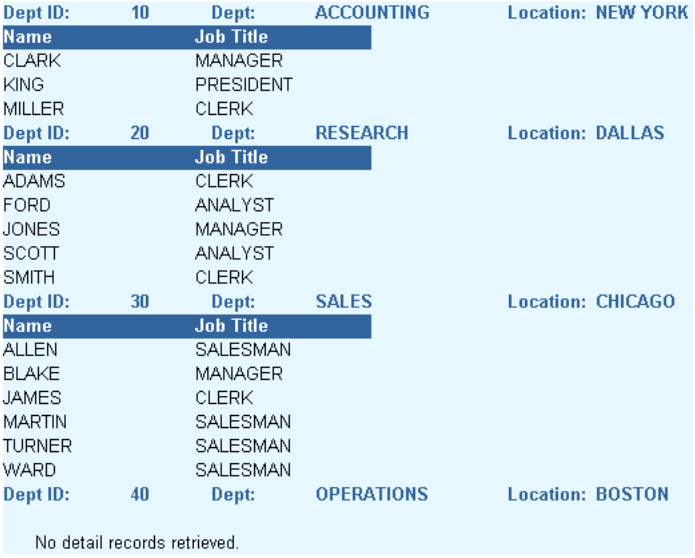
Now, we need to change the fill color of the parent frame, so that you cannot see the boilerplate text behind the detail records.

10. In the Object Navigator, click **M_G_DEPTNO1_GRPFR** so that it is the only object selected. In the Paper Layout view, you should see this same frame selected.

11. Click the Fill Color tool in the tool palette, and change the fill color to light blue, so that it matches the template.

12. Click the Paper Design button in the toolbar to run and display your report in the Paper Design view. Your report should look something like this:

Figure 21–8 Final Paper Output of the Suppressing Labels Report



Dept ID:	10	Dept:	ACCOUNTING	Location:	NEW YORK
Name		Job Title			
CLARK		MANAGER			
KING		PRESIDENT			
MILLER		CLERK			
Dept ID:	20	Dept:	RESEARCH	Location:	DALLAS
Name		Job Title			
ADAMS		CLERK			
FORD		ANALYST			
JONES		MANAGER			
SCOTT		ANALYST			
SMITH		CLERK			
Dept ID:	30	Dept:	SALES	Location:	CHICAGO
Name		Job Title			
ALLEN		SALESMAN			
BLAKE		MANAGER			
JAMES		CLERK			
MARTIN		SALESMAN			
TURNER		SALESMAN			
WARD		SALESMAN			
Dept ID:	40	Dept:	OPERATIONS	Location:	BOSTON

No detail records retrieved.

Note: Notice how the record for Department 40 shows the boilerplate text you added, and the field names for the record are suppressed.

13. Save your report as `suppresslabels_<your initials>.rdf`.

21.7 Summary

Congratulations! You have successfully built a report that suppresses labels when no data is retrieved. You now know how to:

- manually create and link two queries.
- create a formula column and a summary column.

- create a default layout in the Report Wizard.
- create a format trigger that suppresses the field labels when no detail records are displayed.
- added boilerplate text and manipulated the layout.

For more information on any of the wizards, views, or properties used in this example, refer to the *Reports Builder Online Help*, which you can access in two ways:

- From the Oracle Technology Network (`http://otn.oracle.com/products/reports/`), click **Documentation** and navigate to the *Reports Builder Online Help* for the most recent, hosted online help.
- From Reports Builder, choose **Help > Help Contents**.

Note: You can replace your Reports Builder online help with the most recent update by downloading the latest online help set available on the Oracle Technology Network (`http://otn.oracle.com/docs/products/reports/`). Instructions for replacing your help file are included in the readme.txt in the download file.

22

Building a Conditional Form Letter Report

Figure 22–1 Conditional form letter report output, base version

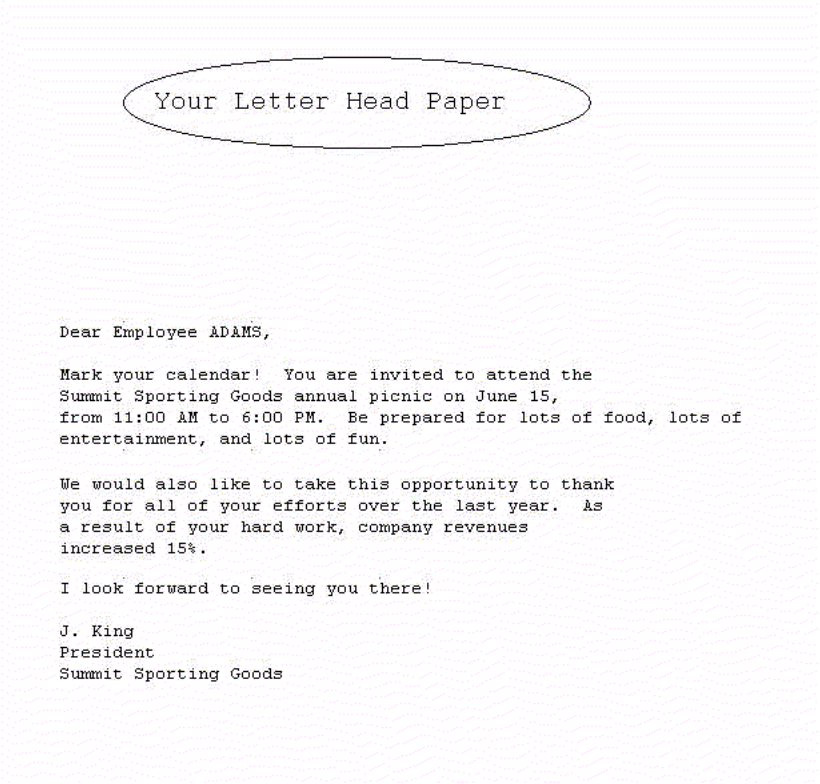
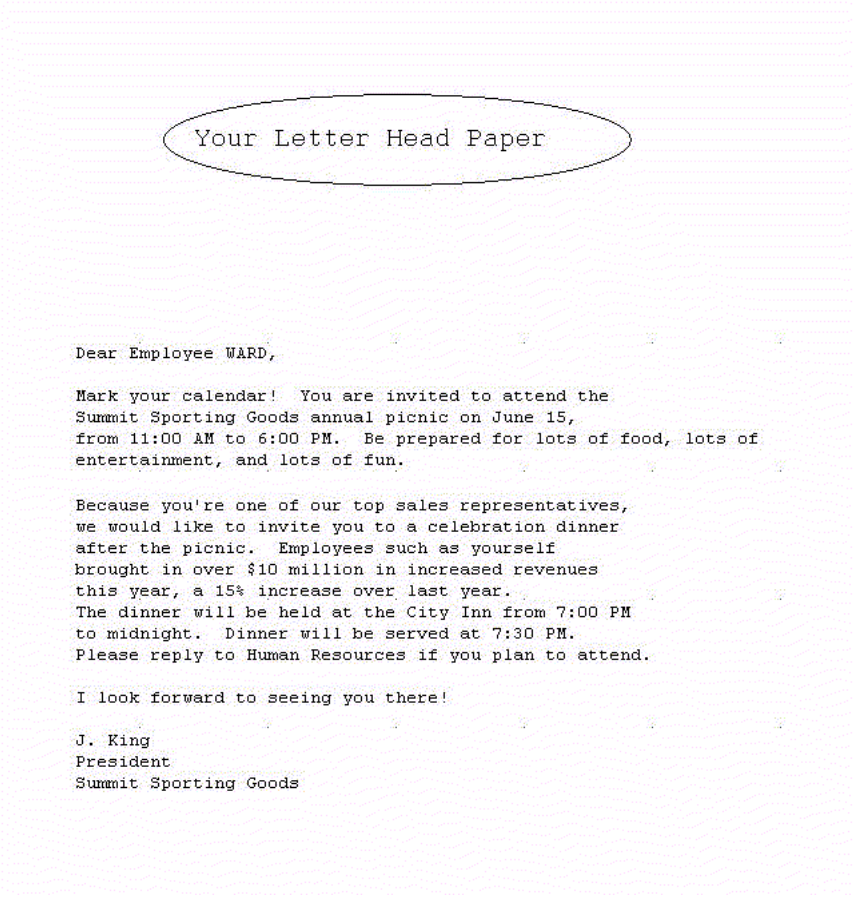


Figure 22–2 Conditional form letter report output, alternate version



The two form letters above were generated from the same report. As you can see, the two letters share a number of features. Hence, it is more convenient to create a base form letter and then apply conditions to certain parts to determine whether they should be displayed for the current record, in this case, employees.

Concepts

- Conditional printing is useful when you want to display a section of a report only if certain conditions are met at run time. Conversely, you may wish to use conditional printing to suppress certain information for those who don't need it.

Data Relationships

- This report uses one query to select all data.

Layout

- This report uses the Form Letter layout style. You'll also create the various pieces of boilerplate that will comprise the letter. To govern the printing of these boilerplate objects, you'll use vertically collapsing anchors and PL/SQL format triggers to conditionally produce different form letters for employees who meet the specified criteria.

Example Scenario

Suppose that you want to create a form letter to invite all of your employees to the company picnic. For your top sales representatives, though, you also want to include a special invitation to a dinner party in the form letter. For all other employees, you want to include a thank you without the dinner invitation.

To see a sample conditional form letter report, open the examples folder named `condform`, then open the Oracle Reports example named `condform1.rdf`. For details on how to access it, see "Accessing the example reports" in the Preface.

Table 22–1 Features demonstrated in this example

Feature	Location
Create a data model and a layout.	Section 22.2, "Create the data model and layout"
Add text.	Section 22.3, "Add additional text"
Anchor objects together and create format triggers to control whether the text displays.	Section 22.4, "Add logic for text"

22.1 Prerequisites for this example

To build the example in this chapter, you must have access to the EMP and DEPT schema, which is provided by default with the Oracle9*i* database. The default user ID and password for accessing this schema is scott/tiger.

22.2 Create the data model and layout

When you are creating a single-query report, such as this one, you can use the Report Wizard to create the data model and layout simultaneously.

To create a data model and layout:

1. Launch Reports Builder (or, if already open, choose **File > New > Report**)
2. In the Welcome or New Report dialog box, select **Use the Report Wizard**, then click **OK**.
3. If the Welcome page displays, click **Next**.
4. On the Report Type page, select **Create Paper Layout Only**, then click **Next**.
5. On the Style page, type a **Title** for your report, select **Form Letter**, then click **Next**.
6. On the Data Source page, click **SQL Query**, then click **Next**.
7. On the Data page, enter the following SELECT statement in the **Data Source definition** field:

```
SELECT ENAME, COMM FROM EMP
ORDER BY ENAME
```

Note: You can enter this query in any of the following ways:

- Copy and paste the code from the provided text file called `condform1_code.txt` into the **Data Source definition** field.
- Click **Query Builder** to build the query without entering any code manually.
- Type the code in the **Data Source definition** field.

---

8. Click **Next**.

Note: If you are not already connected to a database, you will be prompted to connect to the database when you click **Next**. Ensure that you connect to a database that has the appropriate schema for this example. Section 22.1, "Prerequisites for this example" describes the sample schema requirements for this example.

---

9. Type the first paragraph of your letter in the Form Letter Text window:

```
Dear Employee  &<ENAME>,

Mark your calendar! You are invited to
attend the Summit Sporting Goods annual
picnic on June 15, from 11:00 AM to
6:00 PM.  Be prepared for lots of food,
lots of entertainment, and lots of fun.
```

Tip: Be sure to include hard returns at the end of each line.

10. Click **Next**.
11. On the Template page, select **No template**, then click **Finish** to preview your report output in the Paper Design view.

22.3 Add additional text

After you add the basic text in the Report Wizard, you need to create some additional text. The text must reside inside of the repeating frame, so before you add text, you will resize the repeating frame.

To resize the repeating frame:

1. Click the Paper Layout button in the toolbar to display the Paper Layout view.
2. In the Object Navigator, type `R_G_ENAME` in the **Find** field to locate it.
3. Click **R_G_ENAME** to select it. It is simultaneously selected in the Paper Layout view.
4. Click the title bar of the Report Editor to make it the active window.
5. Click the bottom handle of R_G_ENAME and drag it down about 4 inches.

To add more text:

1. Click the Text tool in the tool palette.
2. Click about 0.25 inches below the first text object.
3. Type the following text:

```
Because you're one of our top sales
representatives, we would like to
```

```
invite you to a celebration dinner
after the picnic. Employees such as
yourself brought in over $10 million
in increased revenues this year, a
15% increase over last year. The
dinner will be held at the City Inn
from 7:00 PM to midnight. Dinner will
be served at 7:30 PM. Please reply to
Human Resources if you plan to
attend.
```

4. Click once in an open area of the Paper Layout view.
5. Choose **Tools > Property Inspector** to display the Property Inspector for the boilerplate text object:
   - Under **General Information**, set the Name property to B_SALESREP.
6. Click on the title bar of the Report Editor to make it the active window again.
7. Repeat the steps above, but this time use the following text and name it `B_ALL2`:

```
We would also like to take this
opportunity to thank you for all of
your efforts over the last year.  As
a result of your hard work, company
revenues increased 15%.
```

8. Again, repeat the steps above, but this time use the following text and name it `B_ALL3`:

```
I look forward to seeing you there!

J. King
President
Summit Sporting Goods
```

9. Rename the original text object to `B_ALL` by selecting it and choosing **Tools > Property Inspector** to set the Name property.

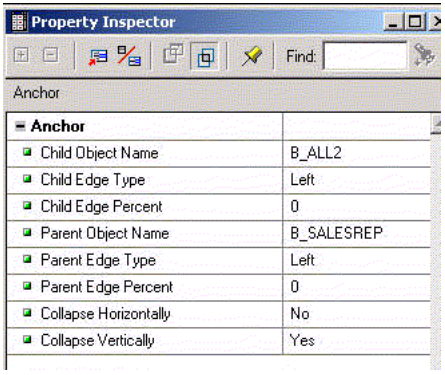
22.4 Add logic for text

The text in B_SALESREP and B_ALL2 applies only to some employees. Hence, you need to apply a condition to each one to determine when it should appear. You also need to ensure that objects (e.g., B_ALL2 and B_ALL3) move up in the layout whenever objects above them (e.g., B_SALESREP and B_ALL2) do not print. Otherwise, you will have large gaps in your report output for employees who do not get all of the objects. To achieve this, you must anchor the text objects together.

To create collapsing anchors:

1. In the Paper Layout view, click the Anchor tool in the tool palette.
2. Click and drag a line from the top left corner of B_SALESREP (the second text object) to the top left corner of B_ALL (the first text object). Double-click to create the new anchor object.
3. Choose **Tools > Property Inspector** to display the Property Inspector for the anchor object:
   - Set the Child Edge Percent and Parent Edge Percent properties are set to 0.
   - Set the Collapse Vertically property to Yes.
4. Click the title bar of the Report Editor to make it the active window again.
5. Repeat the steps above, but this time drag from top left corner of B_ALL2 to the top left corner of B_SALESREP.
6. Again, repeat the steps above, but this time drag from top left corner of B_ALL3 to the top left corner of B_ALL2.

Figure 22–3 Property Inspector with anchor properties displayed



To create format triggers:

1. In the Paper Layout view, double-click **B_SALESREP** to display the Property Inspector, and set properties:
   - Under **Advanced Layout**, click the Format Trigger property field to display the PL/SQL Editor.
2. In the PL/SQL Editor, use the template to enter the following PL/SQL code:

```
function B_SalesrepFormatTrigger return boolean is
begin
  if :comm >= 500 then return(TRUE);
   else return(FALSE);
  end if;
end;
```

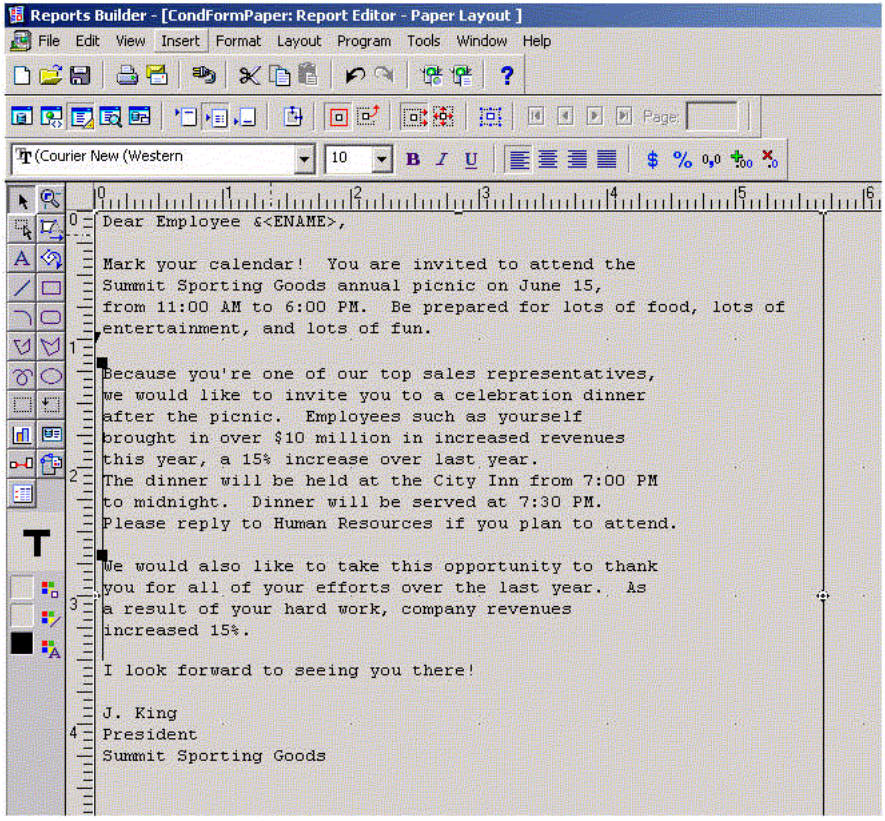
   This boilerplate will print only in the letters to employees whose commissions are greater than $500.

3. Repeat the steps above, but this time enter the following PL/SQL for B_ALL2:

```
function B_ALL2FormatTrigger return boolean is
begin
   if ((:comm < 500) or (:comm is null)) then
     return(TRUE);
   else return(FALSE);
   end if;
end;
```

This logic causes B_ALL2 to print only in the letters where B_SALESREP does not print.

Figure 22–4 Paper Layout view of modified report



4. Compile, save, and run your report. Page through the form letters to see that some have the dinner invitation while others have the "thank you" message.

Tip: When viewing this report on your screen, you may wish to specify a Destination Type of Preview. Your report output will display as though using Postscript fonts instead of screen display fonts, and you can get a clearer indication of the appearance (e.g., line wrapping) of the printed report.

In addition, when you print a conditional form letter report on stationery as illustrated, be sure to take into account the position of the letterhead, etc., when creating the layout so the text of the letter fits into the overall design of the stationery.

22.5 Summary

Congratulations! You have successfully created a conditional form letter report. You now know how to:

- create a data model and a layout.
- add text.
- anchor objects together and create format triggers to control whether the text displays.

For more information on any of the wizards, views, or properties used in this example, refer to the *Reports Builder Online Help*, which you can access in two ways:

- From the Oracle Technology Network (`http://otn.oracle.com/products/reports/`), click **Documentation** and navigate to the *Reports Builder Online Help* for the most recent, hosted online help.
- From Reports Builder, choose **Help > Help Contents**.

Note: You can replace your Reports Builder online help with the most recent update by downloading the latest online help set available on the Oracle Technology Network (`http://otn.oracle.com/docs/products/reports/`). Instructions for replacing your help file are included in the readme.txt in the download file.

23

Building a Report with Conditional Highlighting

Figure 23–1 Conditional highlighting report output

Employee Id	First Name	Last Name	Salary
100	Steven	Kings	$24,000.00
101	Neena	Kochhar	$17,000.00
102	Lex	De Haan	$17,000.00
103	Alexander	Hunold	$9,000.00
104	Bruce	Ernst	$6,000.00
105	David	Austin	$4,800.00
106	Valli	Pataballa	$4,800.00
107	Diana	Lorentz	$4,200.00
108	Nancy	Greenberg	$12,000.00
109	Daniel	Faviet	$9,000.00
110	John	Chen	$8,200.00
111	Ismael	Sciarra	$7,700.00
112	Jose Manuel	Urman	$7,800.00
113	Luis	Popp	$6,900.00
114	Den	Raphaely	$11,000.00
115	Alexander	Khoo	$3,100.00

The report you will build in this chapter demonstrates how to highlight data in your report. In this report that shows employee salaries, salaries that are greater than or equal to 10,000 are displayed in bold and in red color, and values that are between 4,999 and 10,000 are displayed in bold. Using the Conditional Formatting dialog box in Reports Builder, you can create a format trigger that will change the appearance of retrieved data depending on factors you define.

You can use the Conditional Formatting dialog box to create this format trigger, or you can manually create them using the PL/SQL Editor. The steps in this example will show you how to use the dialog box, then display the code in the PL/SQL Editor to see how the format trigger was automatically generated by Reports Builder.

Concepts

- With conditional highlighting, you can format specified portions of a report's output when certain criteria are met.

Data Relationships

- This report uses one query to fetch all data.

Layout

- This report uses a tabular layout style. To add conditional highlighting, you will use the Conditional Formatting dialog box to determine which names and salaries will be highlighted in the report output.

Example Scenario

In this example, suppose you want to create a report for managers that shows a complete list of employees in the company, but also highlights employee salaries that are greater than 10,000. You also need to indicated which employees' salaries are between 4,999 and 10,000. In this example, you will use conditional formatting to highlight these figures in bold and red text.

To see a sample report that uses conditional highlighting, open the examples folder named `condhigh`, then open the Oracle Reports example report named `condhigh.rdf`. For details on how to open it, see "Accessing the example reports" in the Preface.

Table 23–1 Features demonstrated in this example

Feature	Location
Use the Report Wizard to create a simple tabular report based on a single query.	Section 23.2, "Create a basic tabular report"
Use the Conditional Formatting dialog box to highlight specific data in your report.	Section 23.3, "Add conditional formatting to the report"
Open the PL/SQL Editor for the format trigger that was generated by Reports Builder.	Section 23.4, "Examine the conditional format trigger code"

23.1 Prerequisites for this example

To build the example in this chapter, you must have access to the Human Resources portion of the sample schema, which is provided by default with the Oracle9*i* database. See your database administrator for more information.

23.2 Create a basic tabular report

The steps in this section will show you how to use the Report Wizard to build a simple tabular report.

To create a tabular report:

1. Launch Reports Builder (or, if already open, choose **File > New > Report**)
2. In the Welcome or New Report dialog box, select **Use the Report Wizard**, then click **OK**.
3. If the Welcome page displays, click **Next**.
4. On the Report Type page, select **Create Paper Layout Only**, then click **Next**.
5. On the Style page, type a **Title** for your report, select **Tabular**, then click **Next**.
6. On the Data Source page, click **SQL Query**, then click **Next**.
7. On the Data page, enter the following SELECT statement in the **Data Source definition** field:

```
SELECT ALL EMPLOYEES.FIRST_NAME, EMPLOYEES.LAST_NAME,
EMPLOYEES.EMPLOYEE_ID, EMPLOYEES.SALARY
FROM HR.EMPLOYEES
```

Note: You can enter this query in any of the following ways:

- Copy and paste the code from the provided text file called `condhigh_code.txt` into the **Data Source definition** field.
- Click **Query Builder** to build the query without entering any code manually.
- Type the code in the **Data Source definition** field.

---

8. Click **Next**.

Note: If you are not already connected to a database, you will be prompted to connect to the database when you click **Next**. Ensure that you connect to a database that has the appropriate schema for this example. Section 23.1, "Prerequisites for this example" describes the sample schema requirements for this example.

---

9. On the Fields page, click the right arrow (>) to move the following fields to the **Displayed Fields** list, then click **Next**. Make sure you move them in the following order:
   - **EMPLOYEE_ID**
   - **FIRST_NAME**
   - **LAST_NAME**
   - **SALARY**
10. On the Totals page, click **Next**.
11. On the Labels page, click **Next**.
12. On the Template page, select **Predefined Template** and click **Beige**, then click **Finish** to preview your report output in the Paper Design view. It should look something like this:

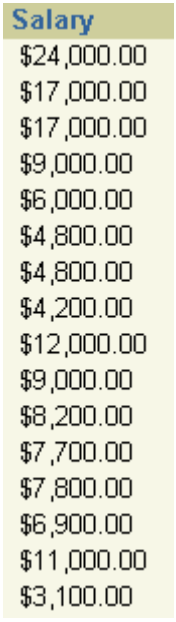
Figure 23–2 Paper Design view for the initial tabular report

Employee Id	First Name	Last Name	Salary
100	Steven	Kings	24000
101	Neena	Kochhar	17000
102	Lex	De Haan	17000
103	Alexander	Hunold	9000
104	Bruce	Ernst	6000
105	David	Austin	4800
106	Valli	Pataballa	4800
107	Diana	Lorentz	4200
108	Nancy	Greenberg	12000
109	Daniel	Faviet	9000
110	John	Chen	8200
111	Ismael	Sciarra	7700
112	Jose Manuel	Urman	7800
113	Luis	Popp	6900
114	Den	Raphaely	11000
115	Alexander	Khoo	3100

13. Now, let's format the data to make it more meaningful. In the Paper Design view, select the **Salary** column by clicking once on the column of data.
14. In the toolbar, click the Currency button, then the Commas button, and finally click the Add Decimal button twice.

    The Salary column of your report should now look like this:

Figure 23–3 Formatted salary column



15. Change the alignment of your columns by doing the following:
    1. Click the **Salary** column heading, then click the Align Center button in the toolbar.
    2. Click the **Salary** column once, then click the Align Right button in the toolbar.
    3. While the **Salary** column is selected, click and drag one of the right black squares to size the column.
    4. Click the **Employee Id** column heading, then click the Align Center button in the toolbar.
    5. Click the **Employee Id** column, then click the Align Center button in the toolbar.
16. The Paper Design view of your report should now look like this:

Figure 23–4 Paper Design view of the tabular report

Employee Id	First Name	Last Name	Salary
100	Steven	Kings	$24,000.00
101	Neena	Kochhar	$17,000.00
102	Lex	De Haan	$17,000.00
103	Alexander	Hunold	$9,000.00
104	Bruce	Ernst	$6,000.00
105	David	Austin	$4,800.00
106	Valli	Pataballa	$4,800.00
107	Diana	Lorentz	$4,200.00
108	Nancy	Greenberg	$12,000.00
109	Daniel	Faviet	$9,000.00
110	John	Chen	$8,200.00
111	Ismael	Sciarra	$7,700.00
112	Jose Manuel	Urman	$7,800.00
113	Luis	Popp	$6,900.00
114	Den	Raphaely	$11,000.00
115	Alexander	Khoo	$3,100.00

17. Save your report as `condhigh_<your initials>.rdf`.

23.3 Add conditional formatting to the report

The steps in this section will show you how to add conditional formatting so that salaries higher than 10,000 will be highlighted in bold, red text, and salaries between 4,999 and 10,000 will be highlighted in bold text.

Since the data retrieved can not be both higher than 10,000 and between 4,999 and 10,000, you will need to create two separate format exceptions. In this section, you will see how to create each distinct format exception.

To add conditional formatting:

1. In the Paper Design view, right-click the **Salary** column of data (not the Salary column heading), then choose **Conditional Formatting**.
2. In the Conditional Formatting dialog box, click **New** to create a new Format Exception.
3. In the Format Exception dialog box, make sure **SALARY** is selected.

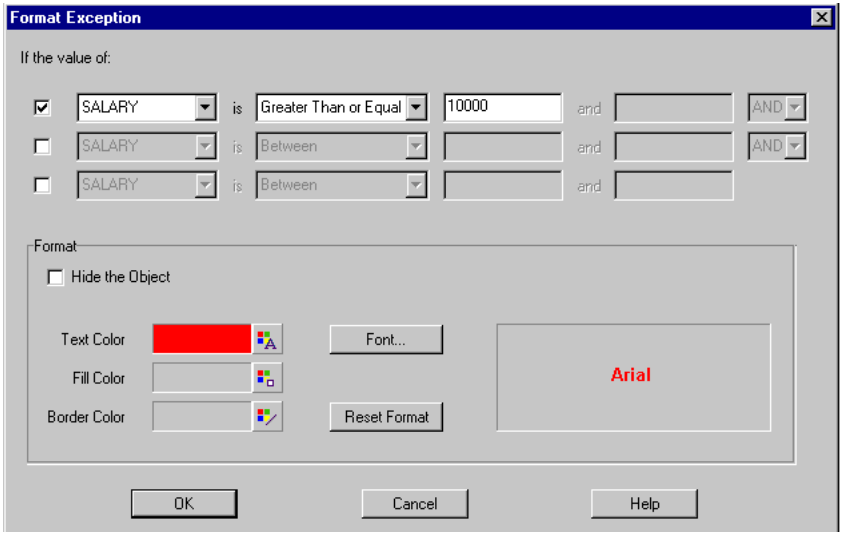
4. Next to **SALARY**, choose **Greater Than or Equal** from the drop-down list.
5. In the next box, type `10000`.
6. Under **Format**, click the icon next to **Text Color t**o display the color palette.

> **Note:** You can choose as any options as you want in the Format Exception dialog box, such as text color, style, and font.

7. Click **Red**.
8. Click **Font**, then choose **Bold**.
9. Click **OK** to accept the new font style.

   You should now see the following options selected in the Format Exception dialog box:

Figure 23–5 Format Exception dialog box



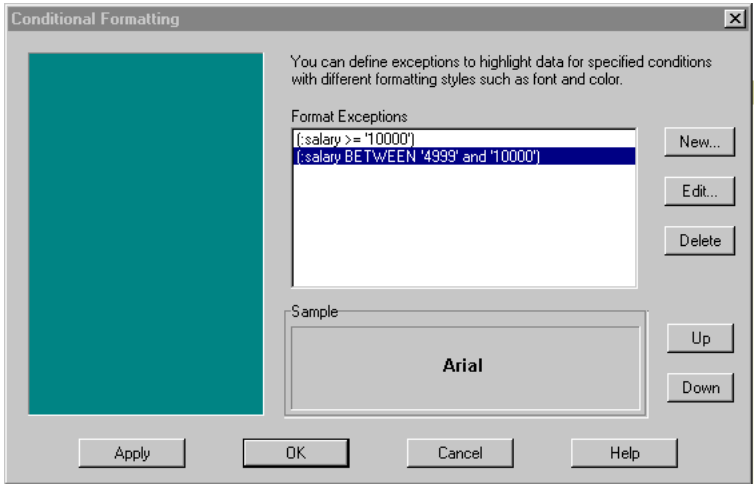
10. Click **OK**.

11. In the Conditional Formatting dialog box, click **New** to create your second format exception.
12. Create a format exception where the values of the SALARY column between 4999 and 10000 are highlighted in bold.

> **Note:** Make sure you type the values in the order described, so that Reports Builder knows to highlight the data between 4999 and 10000, and *not* 10000 and 4999.

13. When you're done, click **OK**. The Conditional Formatting dialog box should now look like the following:

Figure 23–6 Conditional Formatting dialog box



14. Click **Apply**, then click **OK**. Your report displays in the Paper Design view, and should now look something like this:

Figure 23–7 FInal Conditional Formatting report

Employee Id	First Name	Last Name	Salary
100	Steven	Kings	$24,000.00
101	Neena	Kochhar	$17,000.00
102	Lex	De Haan	$17,000.00
103	Alexander	Hunold	$9,000.00
104	Bruce	Ernst	$6,000.00
105	David	Austin	$4,800.00
106	Valli	Pataballa	$4,800.00
107	Diana	Lorentz	$4,200.00
108	Nancy	Greenberg	$12,000.00
109	Daniel	Faviet	$9,000.00
110	John	Chen	$8,200.00
111	Ismael	Sciarra	$7,700.00
112	Jose Manuel	Urman	$7,800.00
113	Luis	Popp	$6,900.00
114	Den	Raphaely	$11,000.00
115	Alexander	Khoo	$3,100.00

15. Save your report as `condhigh_<your initials>.rdf`.

23.4 Examine the conditional format trigger code

The steps in this section will show you the PL/SQL code that was automatically generated by Reports Builder when you used the Conditional Formatting dialog box to set up your format exceptions.

> **Note:** You can also edit the conditional format trigger code in the PL/SQL Editor, but if you attempt to modify the code again in the Conditional Formatting dialog box, your edits will be overwritten by the selections in the dialog box. If you do modify the code in the PL/SQL Editor, you will see a warning note when you try to open the Conditional Formatting dialog box.

To examine the automatically generated code:

1. In the Paper Design view, right-click the **Salary** column (on which you just applied formatting), then choose **PL/SQL Editor**.

2. In the PL/SQL Editor, you will see the following code that was automatically generated by Reports Builder.

Figure 23–8 PL/SQL code for the new format triggers

```
function F_SALARYFormatTrigger return boolean is
begin

  -- Automatically Generated from Report Builder.
  if (:SALARY >= '10000')
  then
    srw.set_text_color('red');
    srw.set_font_face('Arial');
    srw.set_font_size(10);
    srw.set_font_weight(srw.bold_weight);
    srw.set_font_style(srw.plain_style);
  end if;

  -- Automatically Generated from Report Builder.
  if (:SALARY BETWEEN '4999' and '10000')
  then
    srw.set_font_face('Arial');
    srw.set_font_size(10);
    srw.set_font_weight(srw.bold_weight);
    srw.set_font_style(srw.plain_style);
  end if;
```

Note: In this code, you can see that the two format exceptions you created comprise two parts of a format trigger. You needed to create two separate format exceptions in the Conditional Formatting dialog box to achieve this effect. If you had tried to create both exceptions simultaneously in the same Format Exception dialog box, your data would not have satisfied both exceptions, and thus would not have been highlighted.

23.5 Summary

Congratulations! You have successfully built a report that highlights specified data in the report output. You now know how to:

- use the Report Wizard to create a simple tabular report
- format the appearance of your report using tools in the Paper Design view
- use the Conditional Formatting dialog box and the Format Exception dialog box to create format triggers that highlight certain data in your report output
- examine the code automatically generated by Reports Builder

For more information on any of the wizards, views, or properties used in this example, refer to the *Reports Builder online help*, which you can access in two ways:

- From the Oracle Technology Network (`http://otn.oracle.com/products/reports/`), click **Documentation** and navigate to the *Reports Builder Online Help* for the most recent, hosted online help.
- From Reports Builder, choose **Help > Help Contents**.

Note: You can replace your Reports Builder online help with the most recent update by downloading the latest online help set available on the Oracle Technology Network (`http://otn.oracle.com/docs/products/reports/`). Instructions for replacing your help file are included in the readme.txt in the download file.

24

Building a Report with Dynamic Graphics

Figure 24–1 Dynamic graphics report output

In this chapter, you will build an employee report that displays a different graphic depending on the location of the employee's department.

Concepts

- This chapter discusses how Oracle Reports enables you to link to and display drawings and images that are "dynamic." That is, any changes made to the graphics will be reflected in your report output at runtime.

Data Relationships

- One way to display dynamic graphics in a report involves creating a database column that stores the names of the graphics files you want to display. You do this in the Data Model view.
- In the Data Model view, Oracle Reports provides a property called Read from File that enables you to pull the latest versions of the graphics into your report. When you set the Read from File property for a column listing file names, Oracle Reports displays the graphic contained within the named file, rather than the file name itself. Doing so enables Oracle Reports to pull into the report the latest version of the graphic at runtime.

- Another way to include dynamic graphics in your report is to use the **File Link** tool in the Paper Layout view. This tool enables you to create a boilerplate object used to contain an external file, such as graphics and text. This example does not show you how to use this method, but you can read more about this tool in the *Reports Builder Online Help.*

Layout

- This report uses a simple Group Above layout.

Example Scenario

This chapter will show you how to create a simple report using the Group Above layout to display data about an employee and a department, and display an image for the relevant region.

To see a sample report containing dynamic graphics, open the examples folder called `dynamicgraphics`, then open the Oracle Reports example report named `dynamicgraphics.rdf`. For details on how to open it, see "Accessing the example reports" in the Preface.

Table 24–1 Features demonstrated in this example

Feature	Location
Manually create two queries and link them in the Data Model view.	Section 24.2, "Create the data model with two linked queries"
Make complex modifications to your report in the Paper Layout view.	Section 24.3, "Create the layout of the report using the Report Wizard"

24.1 Prerequisites for this example

To build this example report, you will need access to the data source and the graphics we've provided. You will also need to install an extra table into the database, which contains the pictures for this report.

24.1.1 Access to the data source and installing the Pictures table

To build the example in this chapter, you must have access to the EMP and DEPT schema, which is provided by default with the Oracle9*i* database. The user ID and password for accessing this schema is scott/tiger.

You will also need to ask your database administrator to install the Pictures table by using the provided file called `pictures.dmp`.

24.1.2 Graphics for the report

Before you build this report, make sure the following images are in the `dynamicgraphics` directory where the sample report file is located:

- `NEW_YORK.TIF`
- `BOSTON.TIF`
- `DALLAS.TIF`
- `CHICAGO.TIF`

Although we've chosen to use TIF files in this example, you can use other graphic file formats, such as GIF, JPEG, and BMP.

Then, update your REPORTS_PATH in the Windows registry to include the directory where these images are located. The REPORTS_PATH is located under the Oracle Home entry in the Windows entry. When you edit the entry, add the full directory path of the location of your images.

On UNIX, you can set REPORTS_PATH from the command line or in a shell script.

24.2 Create the data model with two linked queries

The steps in this section will show you how to build a simple data model with two queries in a master/detail relationship.

To create a data model

1. After you've updated the REPORTS_PATH with the images directory path, launch Reports Builder.
2. Choose **File > New > Report**.
3. Select **Build a new report manually**, then click **OK**.
4. In the Data Model view that displays, click the SQL tool in the tool palette, then click an open area of the Data Model view to display the SQL Query Statement dialog box.
5. In the **SQL Query Statement** field, enter the following SELECT statement:

```
SELECT ALL DEPT.DEPTNO, DEPT.DNAME, PICTURES.PICTURE
FROM DEPT, PICTURES
WHERE (DEPT.DEPTNO = PICTURES.DEPTNO)
```

```
ORDER BY DEPT.DEPTNO
```

Note: You can enter this query in any of the following ways:

- Copy and paste the code from the provided text file called `dynamicgraphics_code.txt` into the **SQL Query Statement** field.
- Click **Query Builder** to build the query without entering any code manually.
- Type the code in the **SQL Query Statement** field.

Also note that if you have not installed the Pictures table into the sample schema, you will not be able to create this query.

---

6. Click **OK**.

Note: If you are not already connected to a database, you will be prompted to connect to the database when you click **Next**. Ensure that you connect to a database that has the appropriate schema for this example. Section 24.1, "Prerequisites for this example" describes the sample schema requirements for this example.

---

7. When the query displays in the Data Model view, right-click the query name (**Q_1**), then choose **Property Inspector** from the pop-up menu to display the Property Inspector, and set properties:
   - Under **General Information**, set the Name property to Q_PICTURES.
8. In the Data Model view, double-click the **pictures** column in the **Q_PICTURES** query to display the Property Inspector, and set properties:
   - Under **Column**, set the Read from File property to Yes, and set the File Format property to Image.
9. In the Data Model view, follow the steps above to create another query named Q_EMPLOYEES with the following code:

```
SELECT ALL EMP.DEPTNO, EMP.ENAME, EMP.JOB, EMP.HIREDATE
FROM EMP
ORDER BY DEPTNO, ENAME
```

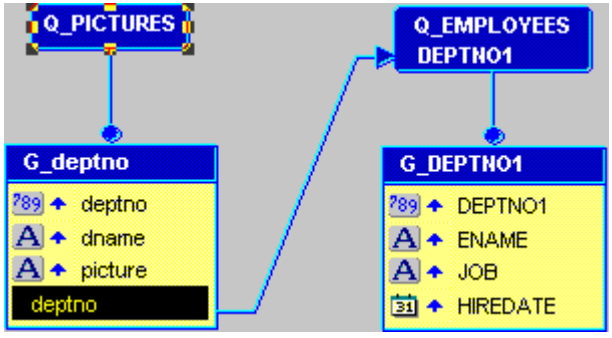
Note: You can enter this query in any of the following ways:

- Copy and paste the code from the provided text file called `dynamicgraphics_code.txt` into the **SQL Query Statement** field.
- Click **Query Builder** to build the query without entering any code manually.
- Type the code in the **SQL Query Statement** field.

---

10. In the Data Model view, click the Data Link tool in the tool palette.
11. Drag a link between **DEPTNO** in **Q_PICTURES** and **DEPTNO1** in **Q_EMPLOYEES**. Your data model should now look like this:

Figure 24–2 Data Model view for the dynamic graphics report



12. Save your report as `dynamicgraphics_<your initials>.rdf`.

24.3 Create the layout of the report using the Report Wizard

In this section, you will create a default layout for your report using the Report Wizard. You will then modify all the necessary layout objects for your graphics.

24.3.1 Create the initial layout of your report using the Report Wizard

To create the initial layout:

1. In the Paper Layout view, choose **Tools > Report Wizard** to display the Report Wizard.
2. In the Report Wizard, on the **Report Type** page, select **Create Paper Layout only**.
3. On the **Style** page, select **Group Above**.
4. On the **Groups** page, click the following groups in the **Available Groups** list and click **Down** to specify the Print Direction and move them to the **Displayed Groups** list:
   - **G_deptno**
   - **G_DEPTNO1**
5. On the **Fields** page:
   - Click the double right arrows (>>) to move all of the fields to the **Displayed Fields** list.
   - In the **Displayed Fields** list, click **deptno**, then click the left arrow (<) to move this field back to the **Available Fields** list.
   - Do the same for **DEPTNO1**.
6. On the **Labels** page, delete the label for the **Picture** field.
7. On the **Template** page, select **No template**, then click **Finish** to display your report layout in the Paper Layout view. It should look something like this:

Figure 24–3 Initial Paper Layout view for the dynamic graphics report

8. Save your report as `dynamicgraphics_<your initials>.rdf`.

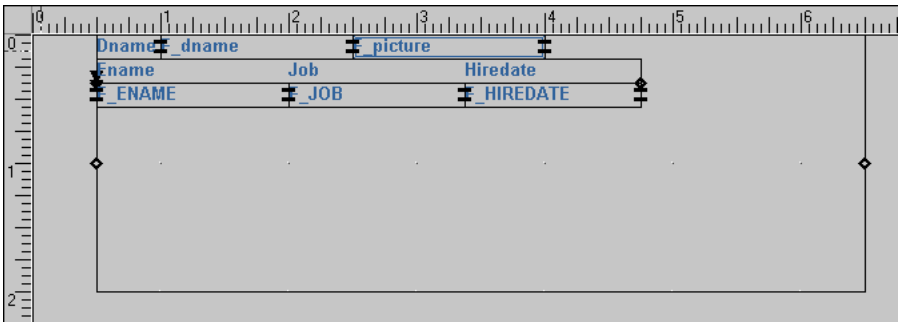
24.3.2 Modify the layout of your report

Now that you've created an initial layout for your report, you can rearrange your layout objects and add dynamic graphics.

To modify your layout:

1. In the Paper Layout view, select all the objects by pressing Ctrl+A on your keyboard.
2. Use the arrow keys on your keyboard to move all the objects 0.5 inches to the right.
3. In the Paper Layout view, expand the screen so that you can see down to 7 inches on the ruler.
4. In the Object Navigator, find your report name, then navigate to **Paper Layout > Main Section > Body > M_G_DEPTNO_GRPFR**.
5. Click **R_G_DEPTNO**. In the Paper Layout view, the **R_G_DEPTNO** repeating frame should also be selected.
6. While the R_G_DEPTNO repeating frame is selected, click on the right part of the frame, and drag it to 6.5 inches (you can see where you are by looking at the top ruler). The repeating frame should now expand from the 0.0 inch mark to the 6.5 inch mark on the top ruler.
7. Now, click the bottom part of the frame, and drag it to 2.0 inches (you can see where you are by looking at the left-hand ruler). The repeating frame should now expand from the 0.0 inch mark to the 6.0 inch mark on the left-hand ruler.

Figure 24–4 Paper Layout view of the modified layout



8. Click the Flex Off button in the toolbar.
9. Move the **Picture** field to the right, and enlarge it to about 1.5 by 1.5 inches.
10. In the Object Navigator, navigate to **Paper Layout > Main Section > Body**, then select the **M_G_DEPTNO_GRPFR** frame.
11. In the Paper Layout view, click the right edge of the frame and drag it to the right to include the **Picture** field.
12. Do the same for the **R_G_DEPTNO** repeating frame.
13. While R_G_DEPTNO is selected, press F4 to open its Property Inspector
14. Under **Repeating Frame**, set the Maximum Records per Page property to `1`.

> **Note:** By setting the maximum records per page to 1, only one state will display per page in your report.

15. Save your report.

24.4 Run your report to paper

To run your report:

1. Click the Run Paper Layout button in the toolbar.

Your report displays in the Paper Design view and should look something like this:

Figure 24–5 Final Dynamic Graphics Report

2. Click the Next Page button in the toolbar to see how the other pages of the report display different graphics depending on the location of the department.

> **Note:** The image displayed by the **Picture** field is updated for each region.

24.5 Summary

Congratulations! You have successfully built a report that displays graphics depending on the data in the report. You now know how to:

- manually create and link two queries.
- use the Read from File property to make graphics dynamic.
- create a default layout in the Report Wizard.
- modify a paper layout to display desired information.

For more information on any of the wizards, views, or properties used in this example, refer to the *Reports Builder Online Help*, which you can access in two ways:

- From the Oracle Technology Network (`http://otn.oracle.com/products/reports/`), click **Documentation** and navigate to the *Reports Builder Online Help* for the most recent, hosted online help.
- From Reports Builder, choose **Help > Help Contents**.

Note: You can replace your Reports Builder online help with the most recent update by downloading the latest online help set available on the Oracle Technology Network (`http://otn.oracle.com/docs/products/reports/`). Instructions for replacing your help file are included in the readme.txt in the download file.

Part IV

Building Matrix Reports

The chapters in this Part provide steps to build matrix reports. Matrix reports display information in a grid format and include variations, such as information displayed as nested grids or grids displayed for each group in the report.This part of the manual introduces matrix reports in Oracle Reports Developer. Matrix reports display information in a grid format and include variations, such as information displayed as nested grids or grids displayed for each group in the report.

- Chapter 25, "Building a Matrix Report"

  A matrix report looks like a grid that contains a row of labels, a column of labels, and information in a grid format related to both the row and column labels. These reports are sometimes referred to as "crosstab" reports.

- Chapter 26, "Building a Nested Matrix Report"

  A nested matrix report contains the cross product, that displays every possible value for three dimensions - two down (such as YEAR and DEPTNO) and one across (such as JOB).

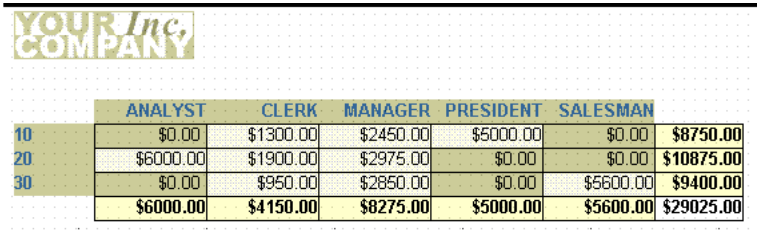
- Chapter 27, "Building a Matrix with Group Above Report"

  A matrix with group above report is a combination of a matrix layout and a group above layout. For each group in this report, the break group (e.g., the department number) displays across the top of the record, and the details of the group (e.g., the employee information) displays in a matrix format.

25

Building a Matrix Report

Figure 25–1 Matrix report output



	ANALYST	CLERK	MANAGER	PRESIDENT	SALESMAN	
10	$0.00	$1300.00	$2450.00	$5000.00	$0.00	**$8750.00**
20	$6000.00	$1900.00	$2975.00	$0.00	$0.00	**$10875.00**
30	$0.00	$950.00	$2850.00	$0.00	$5600.00	**$9400.00**
	$6000.00	**$4150.00**	**$8275.00**	**$5000.00**	**$5600.00**	**$29025.00**

A matrix report looks like a grid. As shown by the example report above, it contains one row of labels, one column of labels, and information in a grid format that is related to both the row and column labels. (Matrix reports are also sometimes referred to as "crosstab" reports.)

Our sample matrix also contains three additions to the basic matrix: summaries have been added, zeroes replace non-existent values in the cells, and the cells themselves are surrounded by grid lines. Of the summaries, one sums the salaries by department, one sums them by job, and one sums them for the whole report.

Concepts

- Certain requirements exist for building matrix reports:
  - You must have at least four groups in your data model.
  - One group must be a cross product group.
  - At least two of the groups must be within the cross product group. These groups furnish the "labels" of the matrix report.

- At least one group must be a "cell" group; i.e., it must provide the information related to the labels. The values from this group fill the cells created by the matrix.

- These requirements can be seen in the example above. It contains four groups--one group supplies the vertical labels (department numbers) and one group supplies the horizontal labels (job identifiers). These two groups are the children of the third group, called the cross product group, which creates the grid. The fourth group provides the values that fill in the grid.
- Matrix reports are different from tabular reports because the number of columns is not known in advance; i.e., the number of columns in your report is not determined by the number of columns you specify in your SELECT statement plus the columns you create yourself. The number of columns in your report depends on the number of values contained in the columns providing the horizontal and vertical labels. Thus, the report would automatically be extended if a new job function, called RECEPTIONIST, was added to the underlying data tables.
- The queries used to select data for these sample matrix reports are not intended as definitive examples of matrix queries. If you are concerned with performance issues, for example, there are alternate methods of querying data that can improve the performance of a matrix report.
- You can create matrix reports with any number of queries. Section 25.2, "Create a single-query matrix" explains how to create the matrix report using one query. Section 25.3, "Create a multiple-query matrix" explains how to create the same report using three queries. These two methods yield the same results. They are presented as options; feel free to try both methods and settle on a favorite.
- This report uses the matrix layout style. You'll modify some default settings to ensure that the column and row labels display correctly. You'll also modify some field widths to ensure that the fields fit across the page.

Example Scenario

Suppose you want to create report that cross tabulates salaries by job function and department. The result would be a matrix with job functions listed across the top, departments down the side, and sums of salaries in the cells. Thus, you could quickly determine the sum of all of the salaries for clerks in department 20 and compare that value to the one for all clerks in some other department.

To see a sample matrix report, open the examples folder named `matrix`, then open the Oracle Reports example named `matrix1qb.rdf`. For details on how to access it, see "Accessing the example reports" in the Preface.

Table 25–1 Features demonstrated in this example

Feature	Location
Create a matrix report with a single-query data model	Section 25.2, "Create a single-query matrix"
Create a matrix report with a multiple-query data model	Section 25.3, "Create a multiple-query matrix"
Add summaries for rows and columns	Section 25.4, "Add summaries to the single-query matrix"
Substitute zeroes for blanks	Section 25.6, "Add zeroes in place of blanks"
Add grid lines around cells	Section 25.7, "Add a grid"

25.1 Prerequisites for this example

To build the example in this chapter, you must have access to the EMP and DEPT schema, which is provided by default with the Oracle9*i* database. The default userid and password for accessing this schema is scott/tiger.

25.2 Create a single-query matrix

You can build a matrix report with a single query in the data model. A single-query data model typically performs better than a multiple-query data model.

When you are creating a single-query report, such as this one, you can use the Report Wizard to create the data model and layout simultaneously.

To create a data model and layout:

1. Launch Reports Builder (or, if already open, choose **File > New > Report**).
2. In the Welcome or New Report dialog box, select **Use the Report Wizard**, then click **OK**.
3. If the Welcome page displays, click **Next**.
4. On the Report Type page, select **Create Paper Layout Only**, then click **Next**.
5. On the Style page, type a **Title** for your report, select **Matrix**, then click **Next**.
6. On the Data Source page, click **SQL Query**, then click **Next**.

7. On the Data page, enter the following SELECT statement in the **Data Source definition** field:

```
SELECT DEPTNO, JOB, SUM(SAL)
FROM EMP
GROUP BY DEPTNO, JOB
ORDER BY DEPTNO, JOB
```

Note: You can enter this query in any of the following ways:

- Copy and paste the code from the provided text file called `matrix1qb_code.txt` into the **Data Source definition** field.
- Click **Query Builder** to build the query without entering any code manually.
- Type the code in the **Data Source definition** field.

---

8. Click **Next**.

Note: If you are not already connected to a database, you will be prompted to connect to the database when you click **Next**. Ensure that you connect to a database that has the appropriate schema for this example. Section 25.1, "Prerequisites for this example" describes the sample schema requirements for this example.

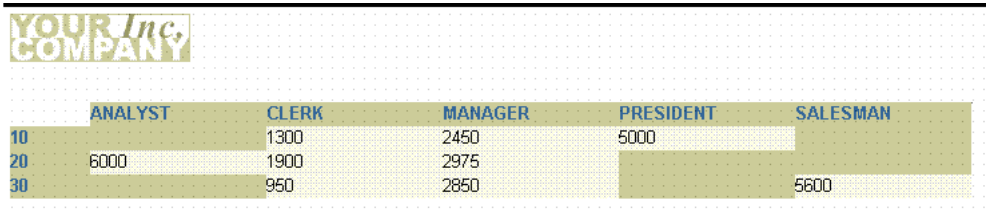
---

9. On the Rows page, click **DEPTNO** and click the right arrow (>) to move this field to the **Matrix Row Fields** list, then click **Next**.
10. On the Columns page, click **JOB** and click the right arrow (>) to move this field to the **Matrix Column Fields** list, then click **Next**.
11. On the Cell page, click **SUM_SAL** and click the right arrow (>) to move this field to the **Matrix Cell Fields** list, then click **Next**.

Note: In this case, the query itself performs the summary via the SUM function. Hence, you should not use the SUM button in this instance.

12. On the Totals page, click **Next**.
13. On the Labels page, delete the labels for all of the fields, then click **Next**.
14. On the **Template** page, make sure **Beige** is selected under **Predefined Template**, then click Finish to display your report output in the Paper Design view. It should look like this:

Figure 25–2 Paper Design view for the matrix report



25.3 Create a multiple-query matrix

You can build a matrix report with multiple queries in the data model. A multiple-query data model is typically easier to conceptualize and code than a single-query, but the single-query data model typically performs better.

25.3.1 Create a new report manually

In this case, it is easier to create the data model and layout separately. Hence, we will create an empty report first, then add the queries, and then create the layouts.

To create a blank report:

1. Choose **File > New > Report**.
2. Select **Build a new report manually**, then click **OK**.

25.3.2 Create a data model with a cross product and a data link

When you create a report with multiple queries, it is typically easier to create all of the queries with the Data Wizard first and then create the layouts with the Report Wizard.

To create the queries:

1. In the Data Model view, choose **Insert > Query** to display the Data Wizard.
2. If the Welcome page displays, click **Next**.
3. On the Query page, type `Q_Dept` for the Query name, then click **Next**.
4. On the Data Source page, click **SQL Query**, then click **Next**.
5. On the Data page, enter the following SELECT statement:

```
SELECT DISTINCT DEPTNO
FROM EMP
```

> **Tip:** If you click **Query Builder**, you can build the query without entering any code manually.

6. Click **Next**.

> **Note:** If you are not already connected to a database, you will be prompted to connect to the database when you click **Next**. Ensure that you connect to a database that has the appropriate schema for this example. Section 25.1, "Prerequisites for this example" describes the sample schema requirements for this example.

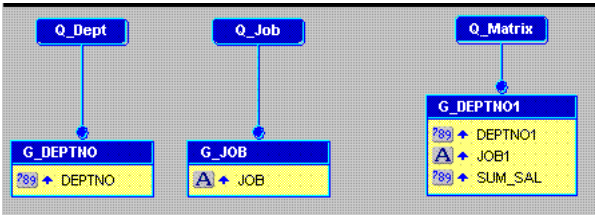
7. On the Groups page, click **Next**.
8. Click **Finish** to display your first query in the Data Model view.
9. Repeat the steps above for a second query, but this time name your query `Q_Job` and use the following SELECT statement:

```
SELECT DISTINCT JOB
FROM EMP
```

10. Again, repeat the steps above for a third query, but this time name your query `Q_Matrix` and use the following SELECT statement:

```
SELECT DEPTNO, JOB, SUM(SAL)
FROM EMP
GROUP BY DEPTNO, JOB
ORDER BY DEPTNO, JOB
```

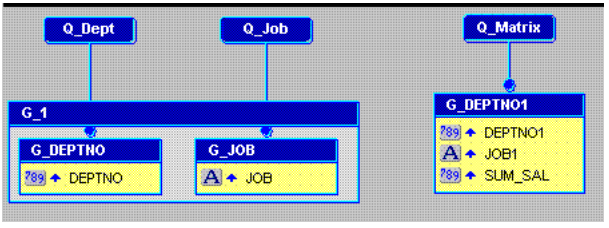
Figure 25–3 Three query data model, queries unrelated



To create the cross product group:

1. In the Data Model view, click the Cross Product tool in the tool palette.
2. Drag a box around G_DEPTNO and G_JOB. When you release the mouse button, the cross product group is created. Ensure that it completely surrounds both groups.

Figure 25–4 Three query data model with cross product group

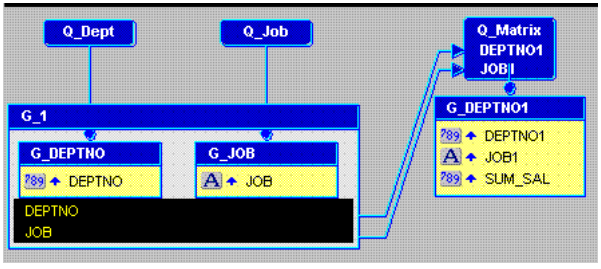


To add the data link:

1. In the Data Model view, click the Data Link tool in the tool palette.

2. Click and drag from the **DEPTNO** column in the **G_DEPTNO** group to the **DEPTNO1** column in the **G_DEPTNO1** group.
3. Repeat steps 1 and 2, but this time drag the link between the **JOB** column in **G_JOB** and **JOB1** in **G_DEPTNO1**.

Figure 25–5 Three query data model with cross product group and data links



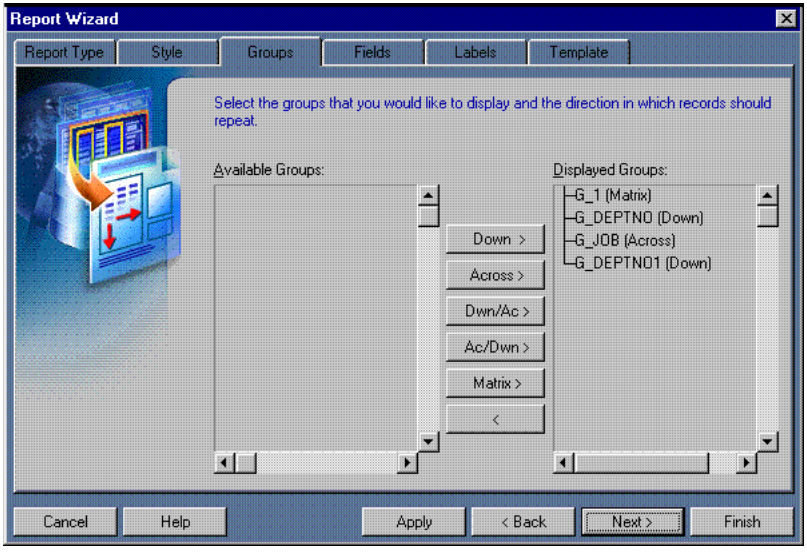
25.3.3 Create the layout with the Report Wizard

Once your data model is complete, you need to create a layout for the data objects to display in the report output.

To create the layout:

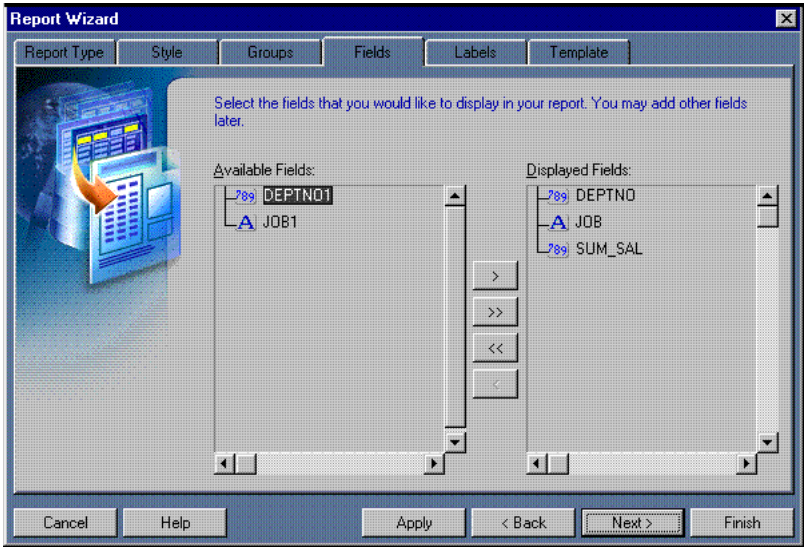
1. In the Data Model view, right-click on the canvas, then choose **Report Wizard**.
2. In the Report Wizard, on the **Report Type** page, select **Create Paper Layout only**.
3. On the **Style** page, select **Matrix**.
4. On the **Groups** page, ensure that all of the groups from your data model appear in the **Displayed Groups** list. **G_1** should be **Matrix**, **G_DEPTNO** should be **Down**, **G_JOB** should be **Across**, and **G_DEPTNO1** should be **Down**.

Figure 25–6 Groups page of the Report Wizard



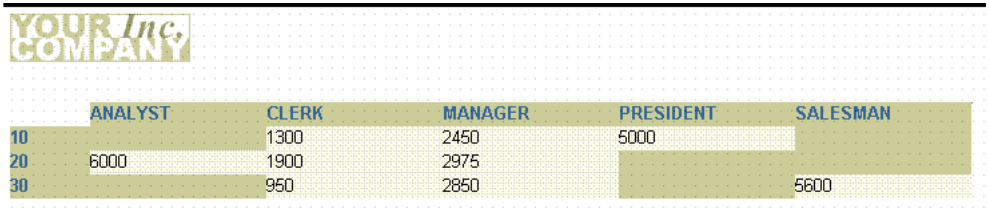
5. On the **Fields** page, ensure that only the following columns appear in the **Displayed Fields** list:
   - **DEPTNO**
   - **JOB**
   - **SUM_SAL**

Figure 25–7 Fields page of the Report Wizard



6. On the **Labels** page, delete the labels for all of the fields.
7. On the **Template** page, make sure **Beige** is selected under **Predefined Template**, then click Finish to display your report output in the Paper Design view. It should look like this:

Figure 25–8 Paper Design view for the matrix report



	ANALYST	CLERK	MANAGER	PRESIDENT	SALESMAN
10		1300	2450	5000	
20	6000	1900	2975		
30		950	2850		5600

25.4 Add summaries to the single-query matrix

To make your matrix report more useful, you should add summaries of each row and column in the matrix, and the whole report. In the single-query case, you can

add these summaries very easily with the Report Wizard. In the multiquery case, you would need to add the summaries manually and then use the Report Wizard to create fields for them.

The steps below describe the procedure for adding summaries to the single-query matrix.

To add summaries to a single-query matrix:

1. Return to the Report Wizard by choosing **Tools** > **Report Wizard**.

   Note: Although you can use the Report Wizard to add summaries to a single-query matrix, you cannot use this method for a multiquery matrix.

2. On the **Report Type** page, select **Create Paper Layout only**.
3. On the **Totals** page, click **SUM_SAL** in the **Available Fields** list, then click **Sum**.

   Tip: You may have to use the arrows to make the Totals tab visible.

4. Click **Finish** to preview your report output in the Paper Design view.

   Note: When your new layout is created, you should notice a couple of things. First, at the bottom of each column of the matrix, you should now see a summary of that column's values. The report is also probably more than one page long now. Because of the width of the layout, the summaries for the departments cannot fit on the page with the rest of the matrix. Hence, the department summaries and the report summary overflow to the next page. Go to the second page of the report and you will see the department summaries and the report summary.

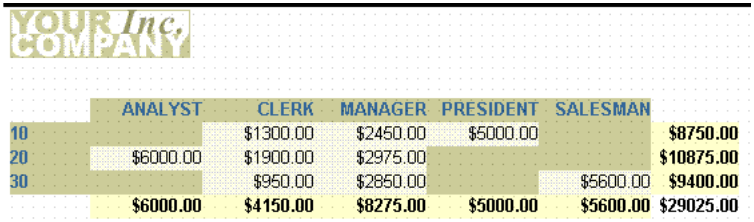
25.5 Format monetary values

To make your report easier to read, you can add formatting to the monetary values.

To format monetary values:

1. On the first page of the report, click one of the cell values. Notice that all of the values are immediately selected, indicating that you can change their properties simultaneously.
2. Shift click on one of the summary values at the bottom of a column of the matrix.
3. Click the Currency button in the toolbar. A currency symbol immediately appears next to all of the values.
4. Click the Add Decimal Place button in the toolbar twice. Two decimal places are added to the right of the decimal point.
5. Resize the fields. Click and drag the rightmost handle of the cell value under the **SALESMAN** label about 0.5 inches to the left. After you complete this operation, the department summaries from the second page should move onto the first page.
6. Shift-click the **SALESMAN** label.
7. Click the Align Right button in the toolbar.
8. Click in an open area of the Paper Design view to deselect all of the objects.
9. Click one of the department summaries at the end of a row in the matrix. All of the department summaries are immediately selected.
10. Shift-click the report summary underneath the department summaries.
11. Use the left arrow key to move these summaries to the left until they are approximately flush with the **SALESMAN** column in the matrix.
12. Click the Currency button in the toolbar. A currency symbol immediately appears next to all of the values.
13. Click the Add Decimal Place button in the toolbar twice. Two decimal places are added to the right of the decimal point.
14. Resize the fields. Click and drag the rightmost handle of one of the selected fields about 0.5 inches to the left.
15. Click the Align Right button in the toolbar.

Figure 25–9 Matrix report in Paper Design view with monetary values formatted



	ANALYST	CLERK	MANAGER	PRESIDENT	SALESMAN	
10		$1300.00	$2450.00	$5000.00		$8750.00
20	$6000.00	$1900.00	$2975.00			$10875.00
30		$950.00	$2850.00		$5600.00	$9400.00
	$6000.00	$4150.00	$8275.00	$5000.00	$5600.00	$29025.00

25.6 Add zeroes in place of blanks

A matrix report displays a juxtaposition of data--in other words, the values held in common by two different categories of data. These categories are indicated by the row and column labels.

The matrix displays this juxtaposition of values using a grid-like format. If the two categories have nothing in common, the grid at that point is empty. The matrix appears to be full of "holes". You can fill the holes using boilerplate text.

Note: Do not confuse empty spaces in the grid with null values. A null value is an actual value fetched from the database. The spaces in a matrix report are empty because nothing has been fetched to fill them.

To replace blanks with zeroes:

1. Open the matrix report to which you previously added the summaries.
2. In the Object Navigator, double-click the view icon next to the **Paper Layout** node to display the Paper Layout view.

   Tip: The steps that follow require some precision in the placement of objects. Hence, you may want to magnify the view to make the process easier. Click the Magnify tool and then click somewhere in the Paper Layout view. Repeat as many times as necessary.

3. In the Paper Layout view, click the Confine On and Flex On buttons in the toolbar to turn both modes on.

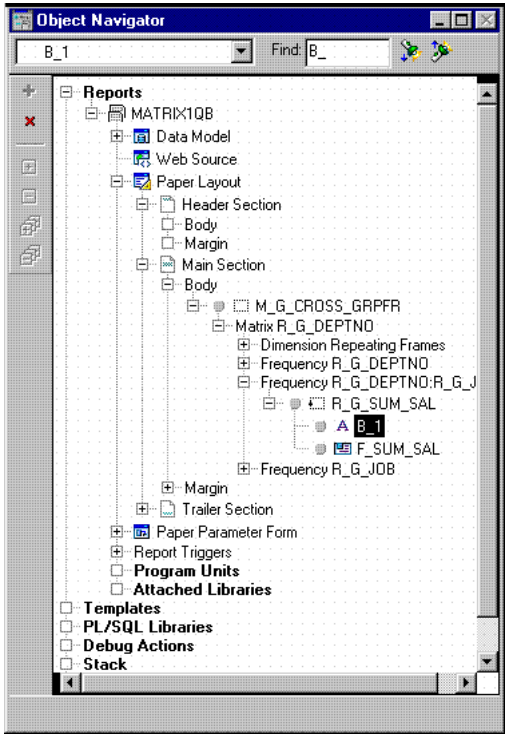
4. From the font lists in the tool bar, choose **Arial** (Western), point size **10**.
5. Click the Align Right button in the toolbar.
6. Click the Text tool in the tool palette.
7. Click on top of the **F_SUM_SAL** field about 0.75 inches from its right edge. Your objective is to create an object right on top of **F_SUM_SAL**.
8. Type the following:

   ```
   $0.00
   ```

9. Click in an open area of the layout.
10. In the Object Navigator, type `B_1` in the **Find** field. You will be taken to the object you just created. If you are viewing the Object Navigator in Ownership View (**View > Change View > Ownership View**), you should see **B_1** underneath **R_G_SUM_SAL** and on the same level as **F_SUM_SAL**.

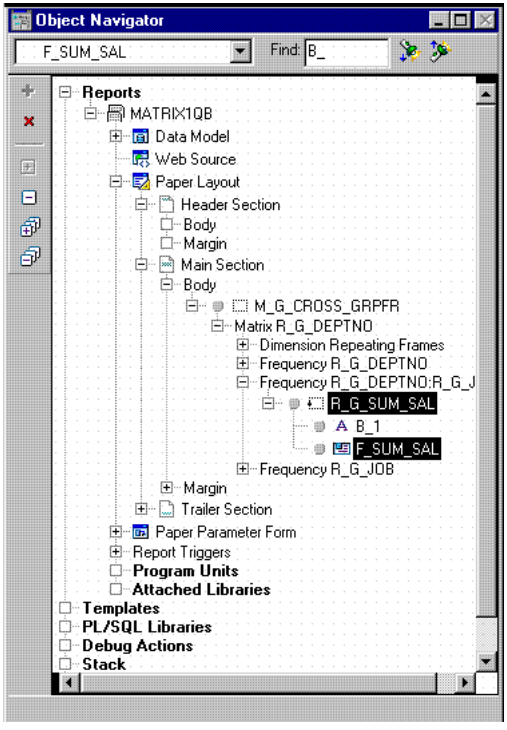
    Tip: If B_1 does not appear underneath R_G_SUM_SAL, return to the Paper Layout view, delete B_1 and try again.

Figure 25–10 Object Navigator with new object selected



11. Click **R_G_SUM_SAL** and then Ctrl-click on **F_SUM_SAL** so that they are both selected and **B_1** is deselected.

Figure 25–11 Object Navigator with repeating frame and field selected



12. Click on the title bar of the Report Editor to make it the active window.
13. Click the Confine Off button in the toolbar to turn Confine mode off.
14. Choose **Layout > Move Forward**.

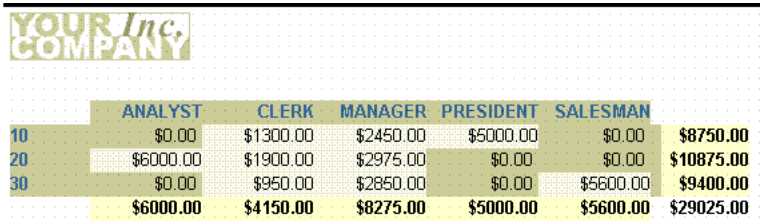
    Tip: After this operation, B_1 should appear just above R_G_SUM_SAL, as a peer rather than a child of R_G_SUM_SAL. If B_1 is still appearing as a child under R_G_SUM_SAL, repeat steps 11 through 14 until it is no longer appearing as a child of R_G_SUM_SAL.

15. In the Paper Layout view, click the Confine On button in the toolbar to turn Confine mode back on again.

16. Click the Paper Design button in the toolbar to display the Paper Design view.

> **Tip:** If the $0.00 is not quite aligning with the other monetary values around it, select it and use the arrow keys to move it to the desired location.

Figure 25–12 Matrix report output with zeroes replacing blanks



	ANALYST	CLERK	MANAGER	PRESIDENT	SALESMAN	
10	$0.00	$1300.00	$2450.00	$5000.00	$0.00	**$8750.00**
20	$6000.00	$1900.00	$2975.00	$0.00	$0.00	**$10875.00**
30	$0.00	$950.00	$2850.00	$0.00	$5600.00	**$9400.00**
	$6000.00	**$4150.00**	**$8275.00**	**$5000.00**	**$5600.00**	**$29025.00**

25.7 Add a grid

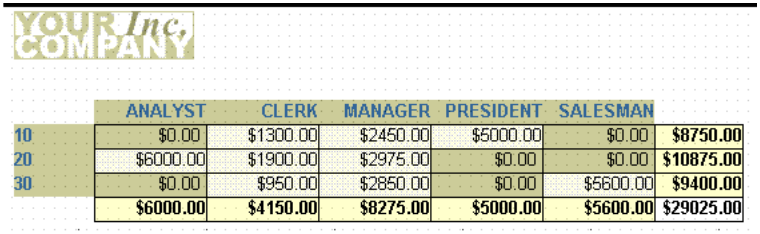
Sometimes matrix reports are easier to read when they have grid lines that divide the cells from each other. For most objects, No Line is the default setting. To add grid lines, all you need to do is select the appropriate objects and give them a line color.

To add grid lines

1. Open the matrix report which you previously modified to show zeroes instead of blanks.
2. In the Object Navigator, double-click the view icon next to the **Paper Layout** node to display the Paper Layout view.
3. In the Object Navigator, choose **View > Change Views > Object Type View**.
4. Select all of the following objects in the Object Navigator using Ctrl-click:
   - **F_SumSUM_SALPerDEPTNO**
   - **F_SumSUM_SALPerJOB**
   - **F_SumSUM_SALPerReport**
   - **F_SUM_SAL**
   - **B_1**

5. Click on the title bar of Paper Layout view to make it the active window.
6. In the Paper Layout view, click the Line Color tool in the tool palette, and choose **black**.
7. Click the Paper Design button in the toolbar to display the Paper Design view. You should now see a grid around all of the cells in your matrix.

Figure 25–13 Matrix report output with a grid



25.8 Summary

Congratulations! You have successfully created a matrix report. You now know how to:

- create a matrix report with a single-query data model.
- create a matrix report with a multiple-query data model.
- add summaries and format monetary values.
- substitute zeroes for blanks in a matrix layout.
- add a grid.

For more information on any of the wizards, views, or properties used in this example, refer to the *Reports Builder Online Help*, which you can access in two ways:

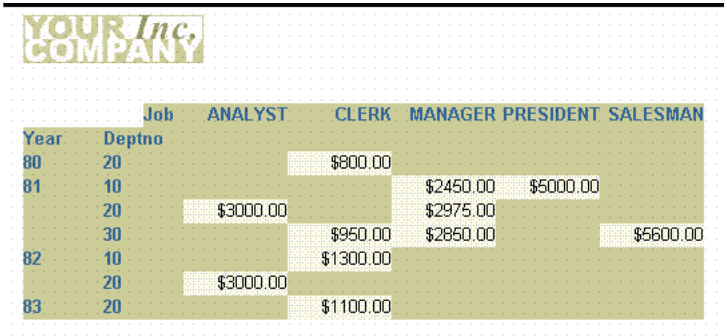
- From the Oracle Technology Network (`http://otn.oracle.com/products/reports/`), click **Documentation** and navigate to the *Reports Builder Online Help* for the most recent, hosted online help.
- From Reports Builder, choose **Help > Help Contents**.

Note: You can replace your Reports Builder online help with the most recent update by downloading the latest online help set available on the Oracle Technology Network (`http://otn.oracle.com/docs/products/reports/`). Instructions for replacing your help file are included in the readme.txt in the download file.

26

Building a Nested Matrix Report

Figure 26–1 Nested matrix report output



In the example nested matrix report shown above, the cross product is capable of displaying every possible value for three dimensions - two down (YEAR and DEPTNO) and one across (JOB). This method does not include rows that have null values because there is a break group within the cross product group in the data model. This chapter describes how to create a nested matrix with and without a break group inside of the cross product group so that you can see the difference in the output.

Concepts

- A nested matrix report is a matrix report which contains more than the usual two dimensions (across and down) of a simple matrix.
- You can create nested matrix reports with any number of queries. In Section 26.2, "Create a single-query matrix", you'll create the example nested matrix report using one query. In Section 26.3, "Create a multiple-query matrix",

you'll use multiple queries. In the Section 26.4, "Create a multiple-query matrix with a break", you will use not only group hierarchy to nest one dimension within another, but you'll also create a true parent/child relationship between the relevant groups by explicitly creating a new group and placing it within the cross product and above the other group. This will restrict the records displayed to only those for which data exists.

Note: The queries used to select data for those sample matrix reports are not intended as definitive examples of matrix queries. If you are concerned with performance issues, for example, there are alternate methods of querying data that can improve the performance of a matrix report. Refer to the *Reports Builder online help* for details.

- This report uses a matrix with three dimensions, created by including three columns in the cross product. The order in which the two down dimensions will be displayed is based upon the order in which you position their corresponding groups within the cross product; i.e., of YEAR and DEPTNO, the two groups printing down, you'll position YEAR to the left of DEPTNO. This "nests" DEPTNO inside of YEAR and causes all records for DEPTNO to appear for each value of YEAR, regardless of whether that department has any data for that year.

Example Scenario

Suppose that you want to create a report that cross tabulates salaries by year and department, and by job function. The result would be a matrix with job functions listed across the top, years and departments down the side, and sums of salaries in the cells. Thus, you could quickly determine the sum of all of the salaries for clerks in department 20 and compare that value to the one for all clerks in some other department.

To see a sample nested matrix report, open the examples folder named `nestedmatrix`, then open any of the Oracle Reports examples named `nested1b.rdf`, `nested3b.rdf`, or `nested4b_brk.rdf`. For details on how to access them, see "Accessing the example reports" in the Preface.

Table 26–1 Features demonstrated in this example

Feature	Location
Create a nested matrix report with a single-query data model	Section 26.2, "Create a single-query matrix"
Create a nested matrix report with a multiple-query data model	Section 26.3, "Create a multiple-query matrix"
Create a nested matrix report with multiple queries and a break within the cross product group	Section 26.4, "Create a multiple-query matrix with a break"
Format monetary values	Section 26.5, "Format monetary values"

26.1 Prerequisites for this example

To build the example in this chapter, you must have access to the EMP and DEPT schema, which is provided by default with the Oracle9*i* database. The userid and password for accessing this schema is scott/tiger.

26.2 Create a single-query matrix

You can build a matrix report with a single-query in the data model. A single-query data model typically performs better than a multiple-query data model.

When you are creating a single-query report, such as this one, you can use the Report Wizard to create the data model and layout simultaneously.

To create a data model and layout:

1. Launch Reports Builder (or, if already open, choose **File > New > Report**).
2. In the Welcome or New Report dialog box, select **Use the Report Wizard**, then click **OK**.
3. If the Welcome page displays, click **Next**.
4. On the Report Type page, select **Create Paper Layout Only**, then click **Next**.
5. On the Style page, type a **Title** for your report, select **Matrix**, then click **Next**.
6. On the Data Source page, click **SQL Query**, then click **Next**.
7. On the Data page, enter the following SELECT statement in the **Data Source definition** field:

```
SELECT TO_CHAR (HIREDATE, 'YY') YEAR, DEPTNO, JOB,
SUM (SAL) FROM EMP
GROUP BY TO_CHAR(HIREDATE, 'YY'), DEPTNO, JOB
ORDER BY TO_CHAR(HIREDATE, 'YY'), DEPTNO, JOB
```

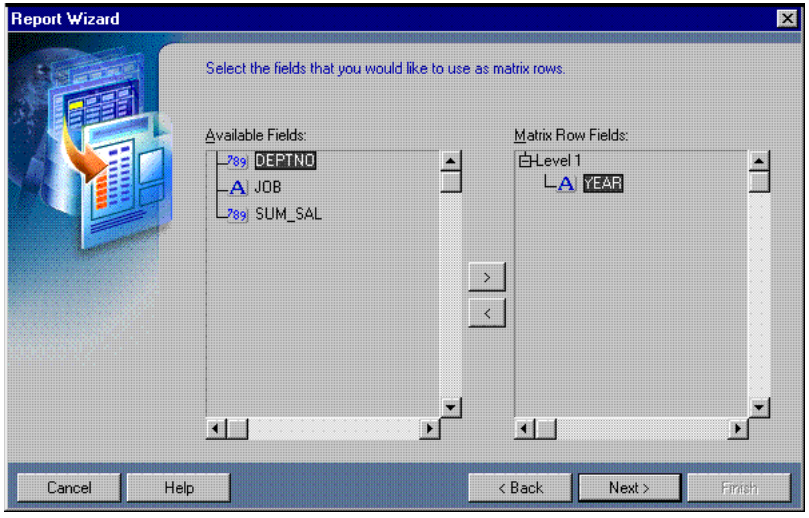
Note: You can enter this query in any of the following ways:

- Copy and paste the code from the provided text file called `nested1b_code.txt` into the **Data Source definition** field.
- Click **Query Builder** to build the query without entering any code manually.
- Type the code in the **Data Source definition** field.

Note: If you are not already connected to a database, you will be prompted to connect to the database when you click **Next**. Ensure that you connect to a database that has the appropriate schema for this example. Section 26.1, "Prerequisites for this example" describes the sample schema requirements for this example.

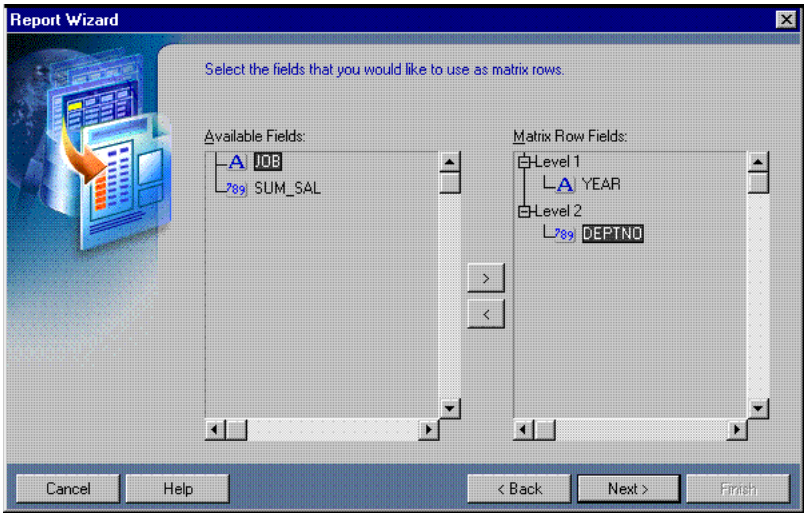
8. On the Rows page:
   - Click **YEAR** in the **Available Fields** list and click the right arrow (>) to move this field to the **Matrix Row Fields** list.**YEAR** should appear under **Level 1**.

Figure 26–2 First matrix row in the Report Wizard



- Click **Level 1** in the **Matrix Row Fields** list. This step ensures that the next column added will be at Level 2 rather than Level 1.
- Click **DEPTNO** in the **Available Fields** list and click the right arrow (>) to move this field to the **Matrix Row Fields** list. Notice that it is added under Level 2 rather than Level 1.

Figure 26–3 Second matrix row in Report Wizard



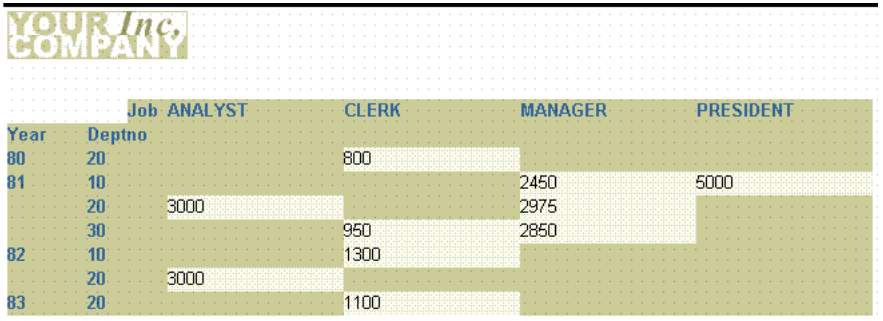
- Click **Next**.

9. On the Columns page, click **JOB** and click the right arrow (>) to move this field to the **Matrix Column Fields** list. JOB should appear under Level 1, then click **Next**.
10. On the Cell page, click **SUM_SAL** and click the right arrow (>) to move this field to the **Matrix Cell Fields** list, then click **Next**.
11. On the Totals page, click **Next**.
12. On the Labels page, change the labels and field widths as follows, then click **Next**:

Fields	Labels	Width
SUM_SAL	<none>	10
YEAR	(no change)	4
JOB	(no change)	10

13. On the **Template** page, make sure **Beige** is selected under **Predefined Template**, then click **Finish** to display your report output in the Paper Design view. It should look like this:

Figure 26–4 Paper Design view for the nested matrix report



Note: You can set alignment and format monetary values directly in the Paper Design view. Simply click the item, then click the appropriate toolbar button (if you run the mouse over these buttons, hint text displays). If the Paper Design view and the Object Navigator are displayed side-by-side, notice that when you select an item in one, the selection is reflected in the other.

26.3 Create a multiple-query matrix

You can build a matrix report with multiple queries in the data model. A multiple-query data model is typically easier to conceptualize and code than a single-query, but the single-query data model typically performs better.

26.3.1 Create a new report manually

In this case, it is easier to create the data model and layout separately. Hence, we will create an empty report first, then add the queries, and then create the layouts.

To create a blank report:

1. Choose **File > New > Report**.
2. Select **Build a new report manually**, then click **OK**.

26.3.2 Create a data model with a cross product and data links

When you create a matrix report with multiple queries, it is typically easier to create all of the queries with the Data Wizard first and then create the cross product group and the necessary links in the Data Model view.

To create the queries:

1. In the Data Model view, choose **Insert > Query** to display the Data Wizard.
2. If the Welcome page displays, click **Next**.
3. On the Query page, type Q_Year for the Query name, then click **Next**.
4. On the Data Source page, select **SQL Query**, then click **Next**.
5. On the Data page, enter the following SELECT statement:

```
SELECT DISTINCT TO_CHAR (HIREDATE, 'YY') YEAR
FROM EMP
```

> **Note:** You can enter this query in any of the following ways:
>
> - Copy and paste the code from the provided text file called `nested4b_code.txt` into the **Data Source definition** field.
> - Click **Query Builder** to build the query without entering any code manually.
> - Type the code in the **Data Source definition** field.

6. Click **Next**.

> **Note:** If you are not already connected to a database, you will be prompted to connect to the database when you click **Next**. Ensure that you connect to a database that has the appropriate schema for this example. Section 26.1, "Prerequisites for this example" describes the sample schema requirements for this example.

7. On the Groups page, click **Next**.

8. Click **Finish** to display the data model for your report in the Data Model view.
9. Repeat the steps above for a second query, but this time name your query `Q_Dept` and use the following SELECT statement:

```
SELECT DISTINCT DEPTNO
FROM EMP
```

> **Note:** You can enter these queries in any of the following ways:
>
> - Copy and paste the code from the provided text file called `nested4b_code.txt` into the **Data Source definition** field.
> - Click **Query Builder** to build the query without entering any code manually.
> - Type the code in the **Data Source definition** field.

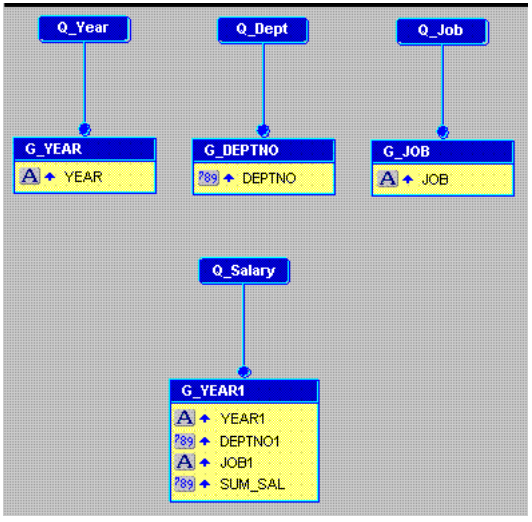
10. Again, repeat the steps above for a third query, but this time name your query `Q_Job` and use the following SELECT statement:

```
SELECT DISTINCT JOB
FROM EMP
```

11. Again, repeat the steps above for a fourth query, but this time name your query `Q_Salary` and use the following SELECT statement:

```
SELECT TO_CHAR(HIREDATE, 'YY') YEAR,  DEPTNO,  JOB,
SUM(SAL) FROM EMP
GROUP BY TO_CHAR(HIREDATE, 'YY'),  DEPTNO,  JOB
ORDER BY TO_CHAR(HIREDATE, 'YY'),  DEPTNO,  JOB
```

Figure 26–5 Data model with four queries

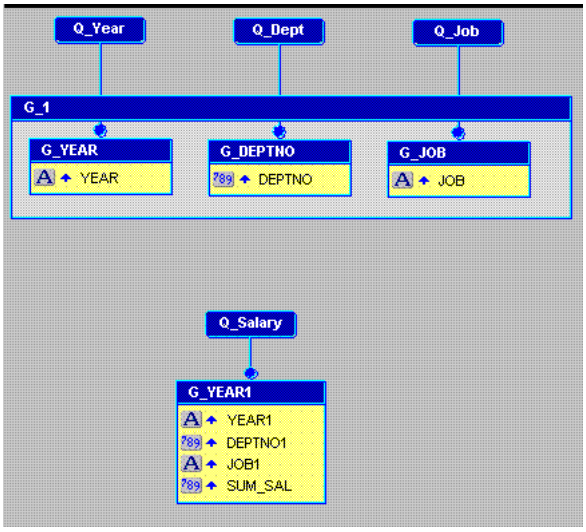


Tip: The order of the queries is significant in this case. Q_Dept must appear to the right of or below Q_Year in order for the values of DEPTNO to be nested inside of the values of YEAR in the output.

To create the cross product group:

1. In the Data Model view, click the Cross Product tool in the tool palette.
2. Drag a box around **G_YEAR**, **G_DEPTNO**, and **G_JOB**. When you release the mouse button, the cross product group is created. Ensure that it completely surrounds all three groups.

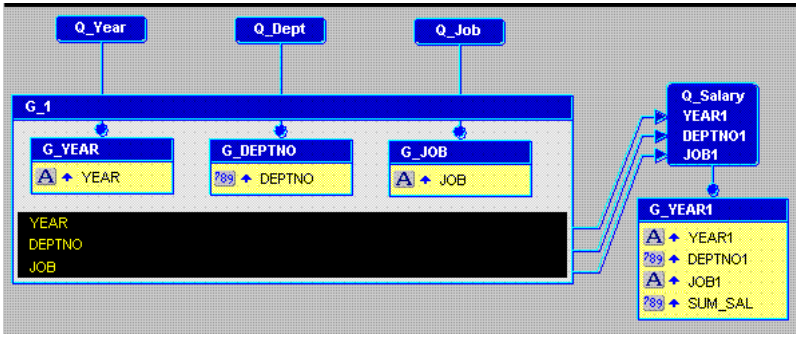
Figure 26–6 Data Model with four queries and a cross product group



To add the data links:

1. In the Data Model view, click the Data Link tool in the tool palette.
2. Click and drag from the **YEAR** column in the **G_YEAR** group to the **YEAR1** column in the **G_YEAR1** group.
3. Repeat steps 1 and 2, but this time drag the link between the **DEPTNO** column in **G_DEPTNO** and **DEPTNO1** in **G_YEAR1**.
4. Again, repeat steps 1 and 2, but this time drag the link between the **JOB** column in **G_JOB** and **JOB1** in **G_YEAR1**.

Figure 26–7 Nested matrix data model



5. Optionally, move the data model objects around to appear like the above figure.

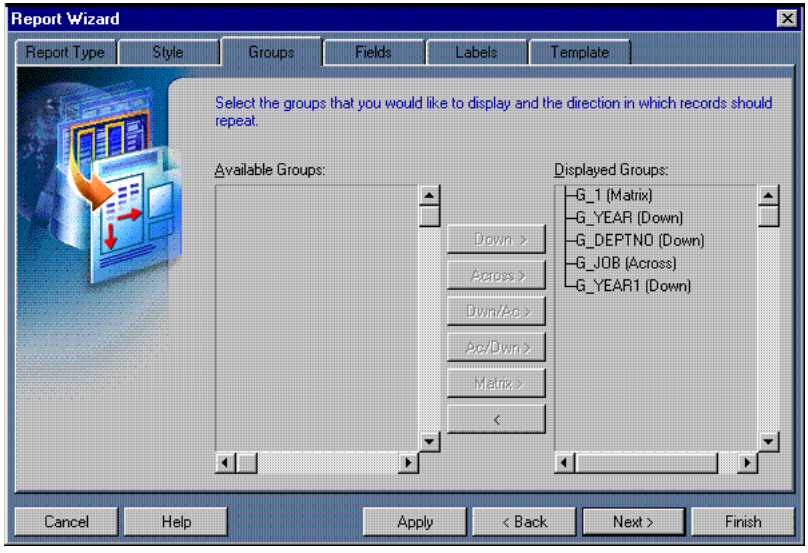
26.3.3 Create the layout with the Report Wizard

Once your data model is complete, you need to create a layout for the data objects to display in the report output.

To create the layout:

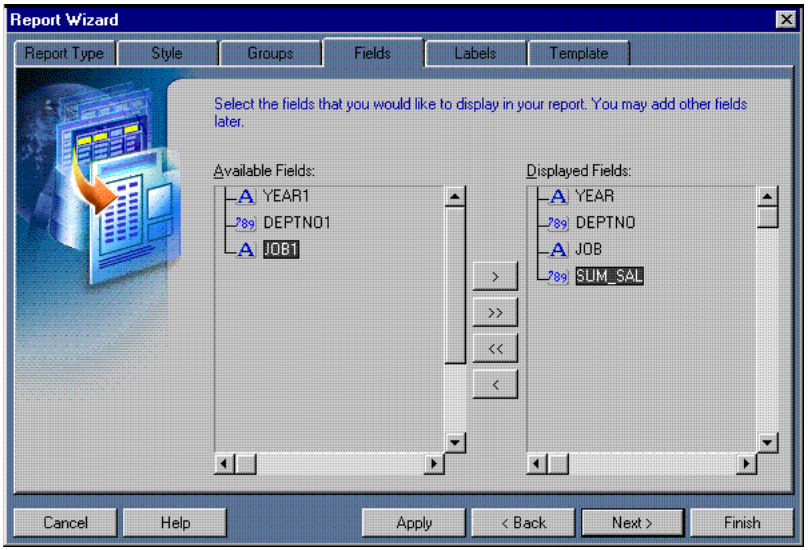
1. In the Data Model view, right-click on the canvas, then choose **Report Wizard**.
2. In the Report Wizard, on the **Report Type** page, select **Create Paper Layout only**.
3. On the **Style** page, select **Matrix**.
4. On the **Groups** page, ensure that all of the groups from your data model appear in the **Displayed Groups** list. **G_1** should be **Matrix**, **G_YEAR** and **G_DEPTNO** should be **Down**, **G_JOB** should be **Across**, and **G_YEAR1** should be **Down**.

Figure 26–8 Groups page of the Report Wizard



5. On the **Fields** page, ensure that only the following columns appear in the **Displayed Fields** list:
   - **YEAR**
   - **DEPTNO**
   - **JOB**
   - **SUM_SAL**

Figure 26–9 Fields page of the Report Wizard

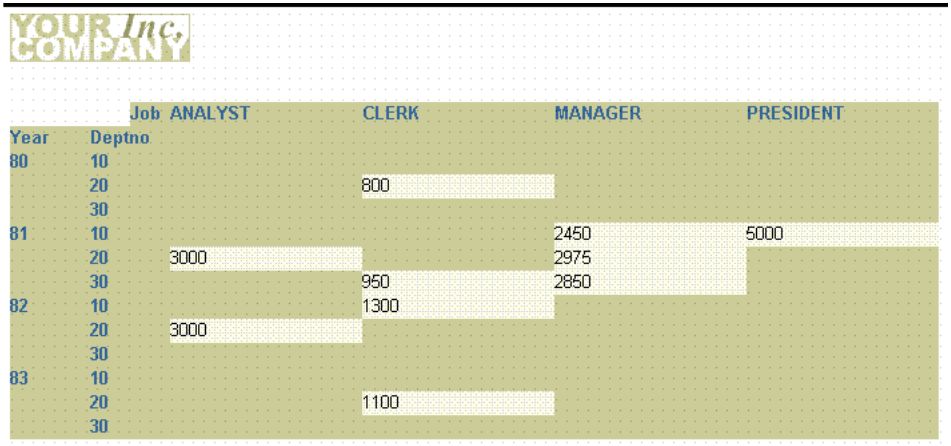


6. On the **Labels** page, change the labels and field widths as follows:

Fields	Labels	Width
SUM_SAL	<none>	10
YEAR		4
JOB		10

7. On the **Template** page, make sure **Beige** is selected under **Predefined Template**, then click **Finish** to display your report output in the Paper Design view. It should look like this:

Figure 26–10 Paper Design view for the nested matrix output



26.4 Create a multiple-query matrix with a break

If you compare the single-query nested matrix output in Figure 26–4, "Paper Design view for the nested matrix report" to the multiple-query output in Figure 26–10, "Paper Design view for the nested matrix output", you notice that the multiple-query case displays all of the departments for every year while the single-query case does not. In the single-query case, only those departments that actually have values in their matrix cells are displayed in the output. To achieve a similar result with multiple queries, you need to have a parent/child relationship between the groups containing YEAR and DEPTNO.

26.4.1 Create a new report manually

In this case, it is easier to create the data model and layout separately. Hence, we will create an empty report first, then add the queries, and then create the layouts.

To create a blank report:

1. Choose **File > New > Report**.
2. Select **Build a new report manually**, then click **OK**.

26.4.2 Create a data model with a cross product and data links

When you create a matrix report with multiple queries, it is typically easier to create all of the queries with the Data Wizard first and then create the cross product group and the necessary links in the Data Model view.

To create the queries:

1. In the Data Model view, choose **Insert > Query** to display the Data Wizard.
2. If the Welcome page displays, click **Next**.
3. On the Query page, type `Q_Dept` for the Query name, then click **Next**.
4. On the Data Source page, select **SQL Query**, then click **Next**.
5. On the Data page, enter the following SELECT statement:

```
SELECT TO_CHAR(HIREDATE, 'YY') YEAR, DEPTNO
FROM EMP
GROUP BY TO_CHAR(HIREDATE, 'YY'), DEPTNO
ORDER BY TO_CHAR(HIREDATE, 'YY'), DEPTNO
```

> **Note:** You can enter this query in any of the following ways:
>
> - Copy and paste the code from the provided text file called `nested3b_brk_code.txt` into the **Data Source definition** field.
> - Click **Query Builder** to build the query without entering any code manually.
> - Type the code in the **Data Source definition** field.

6. Click **Next**.

> **Note:** If you are not already connected to a database, you will be prompted to connect to the database when you click **Next**. Ensure that you connect to a database that has the appropriate schema for this example. Section 26.1, "Prerequisites for this example" describes the sample schema requirements for this example.

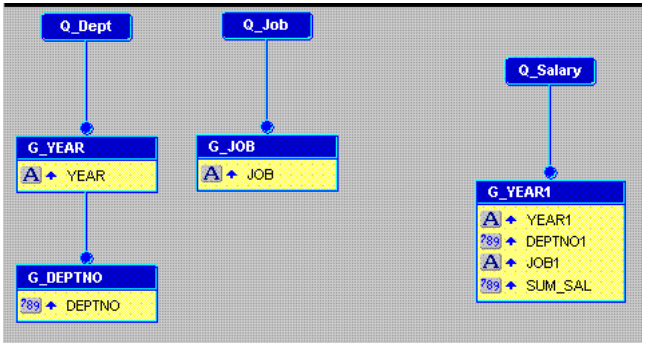
7. On the Groups page, click **YEAR** in the **Available Fields** list and click the right arrow (**>**) to move this field to the **Group Fields** list, then click **Next**.
8. On the Totals page, click **Next**.
9. Click **Finish** to display the data model for your report output in the Data Model view.
10. Choose **Insert > Query** and follow the steps above to create another query named `Q_Job` and use the following SELECT statement:

```
SELECT DISTINCT JOB
FROM EMP
```

11. Again, choose **Insert > Query** and follow the steps above to create a third query named `Q_Salary` and use the following SELECT statement:

```
SELECT TO_CHAR(HIREDATE, 'YY') YEAR,  DEPTNO,  JOB,
SUM(SAL) FROM EMP
GROUP BY TO_CHAR(HIREDATE, 'YY'),  DEPTNO,  JOB
ORDER BY TO_CHAR(HIREDATE, 'YY'),  DEPTNO,  JOB
```

Figure 26–11 Data model with three queries

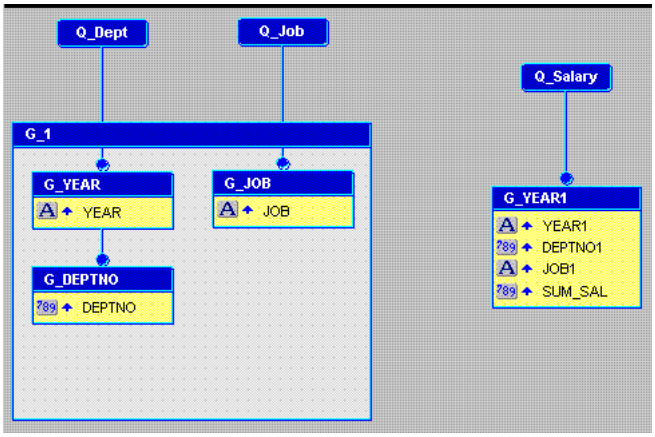


To create the cross product group:

1. In the Data Model view, click the Cross Product tool in the tool palette.

2. Drag a box around **G_YEAR**, **G_DEPTNO**, and **G_JOB**. When you release the mouse button, the cross product group is created. Ensure that it completely surrounds all three groups.

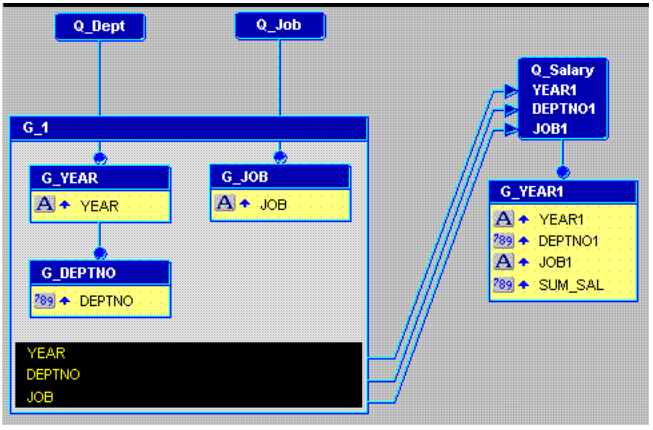
Figure 26–12 Data Model with three queries and a cross product group



To add the data links:

1. In the Data Model view, click the Data Link tool in the tool palette.
2. Click and drag from the **YEAR** column in the **G_YEAR** group to the **YEAR1** column in the **G_YEAR1** group.
3. Repeat steps 1 and 2, but this time drag the link between the **DEPTNO** column in **G_DEPTNO** and **DEPTNO1** in **G_YEAR1**.
4. Again, repeat steps 1 and 2, but this time drag the link between the **JOB** column in **G_JOB** and **JOB1** in **G_YEAR1**.

Figure 26–13 Nested matrix data model



5. Optionally, move the data model objects around to appear like the above figure.

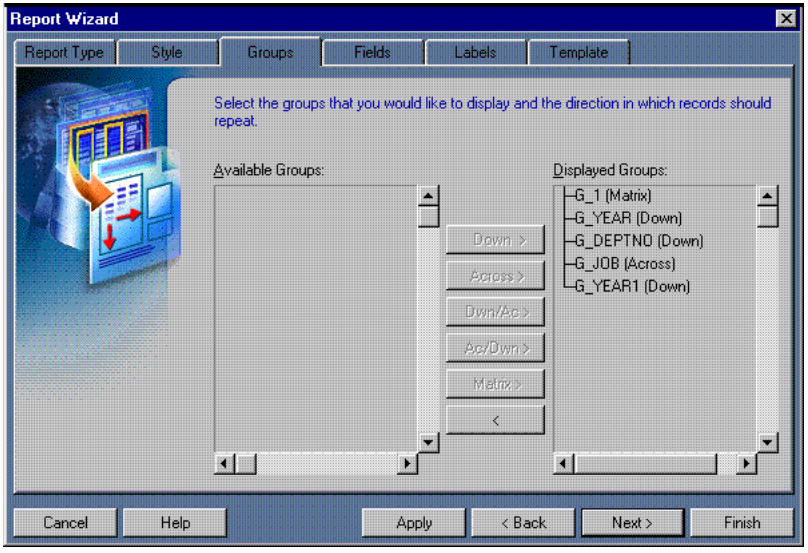
26.4.3 Create the layout with the Report Wizard

Once your data model is complete, you need to create a layout for the data objects to display in the report output.

To create the layout:

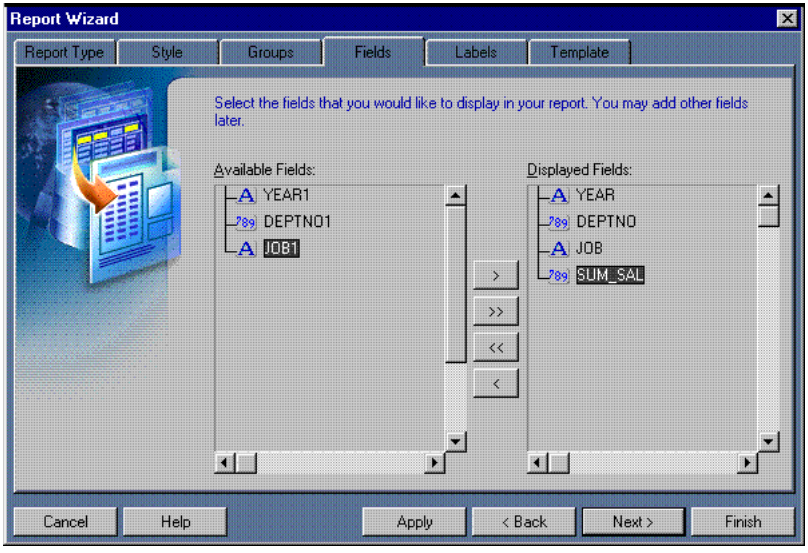
1. In the Data Model view, right-click on the canvas, then choose **Report Wizard**.
2. In the Report Wizard, on the **Report Type** page, select **Create Paper Layout only**.
3. On the **Style** page, select **Matrix**.
4. On the **Groups** page, ensure that all of the groups from your data model appear in the **Displayed Groups** list. **G_1** should be **Matrix**, **G_YEAR** and **G_DEPTNO** should be **Down**, **G_JOB** should be **Across**, and **G_YEAR1** should be **Down**.

Figure 26–14 Groups page of the Report Wizard



5. On the **Fields** page, ensure that only the following columns appear in the **Displayed Fields** list:
   - **YEAR**
   - **DEPTNO**
   - **JOB**
   - **SUM_SAL**

Figure 26–15 Fields page of the Report Wizard

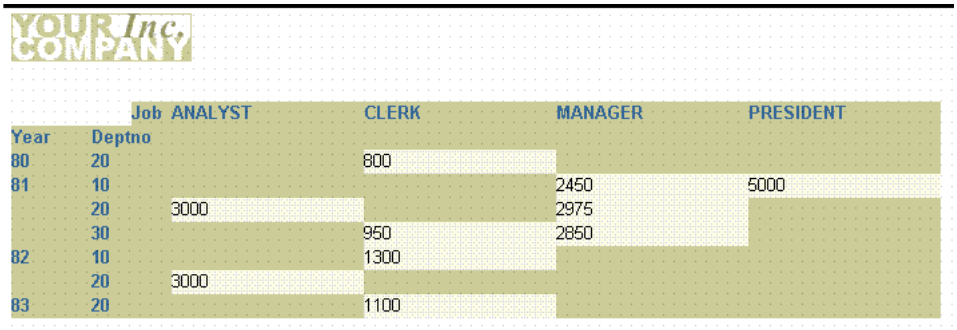


6. On the **Labels** page, change the labels and field widths as follows:

Fields	Labels	Width
SUM_SAL	<none>	10
YEAR		4
JOB		10

7. On the **Template** page, make sure **Beige** is selected under **Predefined Template**, then click **Finish** to display your report output in the Paper Design view. It should look like this:

Figure 26–16 Paper Design view for the final nested matrix report output



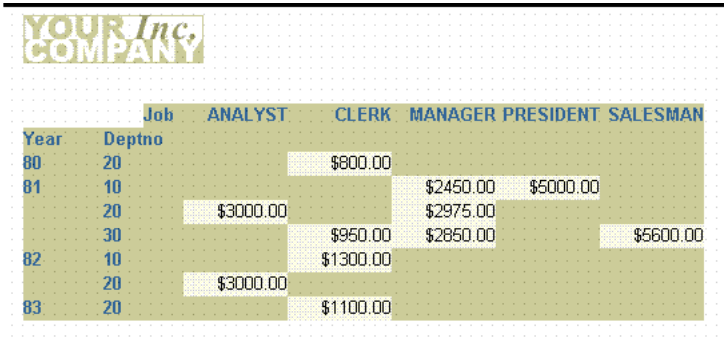
26.5 Format monetary values

To make your matrix report more readable, you should format the monetary values.

To format monetary values:

1. On the first page of the report, click one of the cell values. Notice that all of the values are immediately selected, indicating that you can change their properties simultaneously.
2. Click the Currency button in the toolbar. A currency symbol immediately appears next to all of the values.
3. Click the Add Decimal Place button in the toolbar twice. Two decimal places are added to the right of the decimal point.
4. Resize the fields. Click and drag the rightmost handle of the cell value under the **SALESMAN** label about 0.5 inches to the left. After you complete this operation, the department summaries from the second page should move onto the first page.
5. Shift-click on the **SALESMAN** label.
6. Click the Align Right button in the toolbar.
7. Click in an open area of the Paper Design view to deselect all of the objects.

Figure 26–17 Nested matrix report output with formatted monetary values



	Job	ANALYST	CLERK	MANAGER	PRESIDENT	SALESMAN
Year	Deptno					
80	20		$800.00			
81	10			$2450.00	$5000.00	
	20	$3000.00		$2975.00		
	30		$950.00	$2850.00		$5600.00
82	10		$1300.00			
	20	$3000.00				
83	20		$1100.00			

26.6 Summary

Congratulations! You have successfully created three nested matrix reports. You now know how to:

- create a nested matrix report with a single-query data model.
- create a nested matrix report with a multiple-query data model.
- create a nested matrix report with multiple queries and a break within the cross product group.
- format monetary values.

For more information on any of the wizards, views, or properties used in this example, refer to the *Reports Builder Online Help*, which you can access in two ways:

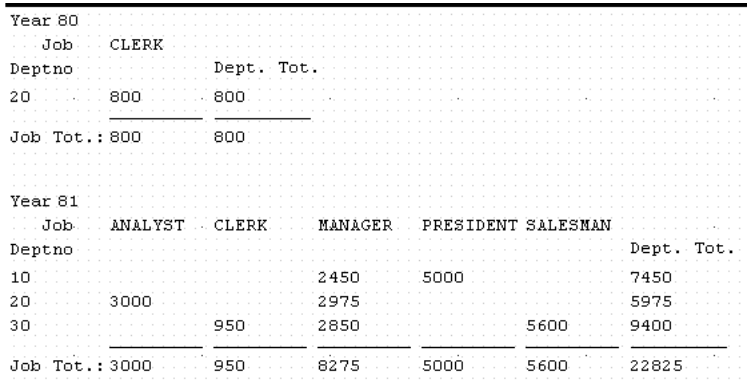
- From the Oracle Technology Network (`http://otn.oracle.com/products/reports/`), click **Documentation** and navigate to the *Reports Builder Online Help* for the most recent, hosted online help.
- From Reports Builder, choose **Help > Help Contents**.

Note: You can replace your Reports Builder online help with the most recent update by downloading the latest online help set available on the Oracle Technology Network (`http://otn.oracle.com/docs/products/reports/`). Instructions for replacing your help file are included in the readme.txt in the download file.

27

Building a Matrix with Group Above Report

Figure 27–1 Matrix with group above report output



```
Year 80
   Job    CLERK
Deptno              Dept. Tot.
20        800       800
          ________  ________
Job Tot.: 800       800


Year 81
   Job    ANALYST   CLERK     MANAGER   PRESIDENT SALESMAN
Deptno                                                    Dept. Tot.
10                             2450      5000              7450
20        3000                 2975                        5975
30                   950       2850                5600    9400
          ________  ________  ________  ________  ________  ________
Job Tot.: 3000      950       8275      5000      5600      22825
```

This report shows department, job, and salary information for each employee by the year they were hired using a matrix break format.

Concepts

A matrix with group above report is a combination of a matrix and a group above report layout. Essentially, a matrix report is printed for each master group record. A matrix with group is similar to the multiquery nested matrix, except that in a matrix with group report, the parent exists above the cross product. In a multiquery nested matrix report with groups, the parent/child relationship exists within the across or down dimension of the cross product.

Data Relationships

- This example of a matrix with group above report uses one query and at least five groups. At least one group is placed above the cross product to serve as the Master group.

Layout

- This report uses the Matrix with Group layout style.

Example Scenario

Suppose that you want to create report that cross tabulates salaries by department and by job for each year. The result would be a group report where year is the master and the detail is a matrix with job functions listed across the top, departments down the side, and sums of salaries in the cells. Thus, you could quickly determine the sum of all of the salaries for clerks in department 20 for any particular year.

To see a sample matrix with group above report, open the examples folder named `matrixgroup`, then open the Oracle Reports example named `matrixgroup.rdf`. For details on how to access it, see "Accessing the example reports" in the Preface.

Table 27–1 Features demonstrated in this example

Feature	Location
Create a matrix group above report with a single-query data model	Section 27.2, "Create a matrix group data model and layout"
Add labels and line for summaries	Section 27.3, "Add labels and lines for summaries"
Add some white space between the master records	Section 27.4, "Add space between groups"
Create a Web layout without changing the paper layout	Section 27.5, "Create a Web layout"

27.1 Prerequisites for this example

To build the example in this chapter, you must have access to the EMP and DEPT schema, which is provided by default with the Oracle9*i* database. The userid and password for accessing this schema is scott/tiger.

27.2 Create a matrix group data model and layout

Since this report is a single-query report, it is easiest to build it with the Report Wizard.

To create the data model and layout with the Report Wizard:

1. Launch Reports Builder (or, if already open, choose **File > New > Report**).
2. In the Welcome or New Report dialog box, select **Use the Report Wizard**, then click **OK**.
3. If the Welcome page displays, click **Next**.
4. On the Report Type page, select **Create Paper Layout Only**, then click **Next**.
5. On the Style page, type a **Title** for your report, select **Matrix with Group**, then click **Next**.
6. On the Data Source page, click **SQL Query**, then click **Next**.
7. On the Data page, enter the following SELECT statement in the **Data Source definition** field:

```
SELECT TO_CHAR(HIREDATE, 'YY') YEAR, DEPTNO, JOB, SUM(SAL)
FROM EMP
GROUP BY TO_CHAR(HIREDATE, 'YY'), DEPTNO, JOB
```

Note: You can enter this query in any of the following ways:

- Copy and paste the code from the provided text file called `matrixgroup_code.txt` into the **Data Source definition** field.
- Click **Query Builder** to build the query without entering any code manually.
- Type the code in the **Data Source definition** field.

8. Click **Next**.

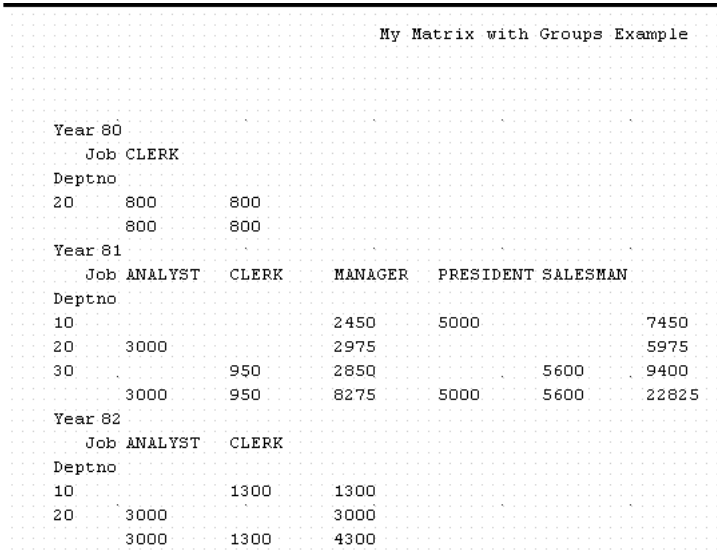
> **Note:** If you are not already connected to a database, you will be prompted to connect to the database when you click **Next**. Ensure that you connect to a database that has the appropriate schema for this example. Section 27.1, "Prerequisites for this example" describes the sample schema requirements for this example.

9. On the Groups page, click **YEAR** in the **Available Fields** list and click the right arrow (>) to move this field to the **Matrix Group Fields** list, then click **Next**.
10. On the Rows page, click **DEPTNO** in the **Available Fields** list and click the right arrow (>) to move this field to the **Matrix Row Fields** list, then click **Next**.
11. On the Columns page, click **JOB** in the **Available Fields** list and click the right arrow (>) to move this field to the **Matrix Column Fields** list, then click **Next**.
12. On the Cell page, click **SUM_SAL** in the **Available Fields** list and click the right arrow (>) to move this field to the **Matrix Cell Fields** list, then click **Next.**

> **Note:** In this case, the query itself performs the summary via the SUM function. Hence, you should not use the **Sum** button in this instance.

13. On the Totals page, click **SUM_SAL** in the **Available Fields** list and click **Sum** to move this field to the **Matrix Totals** list, then click **Next**.
14. On the Labels page, remove the label for **SUM_SAL**, then click **Next**.
15. On the Template page, select **No Template**, then click **Finish** to display your report output in the Paper Design view. It should look something like this:

Figure 27–2 Paper Design view for matrix with group above report



My Matrix with Groups Example

Year 80						
Job	CLERK					
Deptno						
20	800	800				
	800	800				
Year 81						
Job	ANALYST	CLERK	MANAGER	PRESIDENT	SALESMAN	
Deptno						
10			2450	5000		7450
20	3000		2975			5975
30		950	2850		5600	9400
	3000	950	8275	5000	5600	22825
Year 82						
Job	ANALYST	CLERK				
Deptno						
10		1300	1300			
20	3000		3000			
	3000	1300	4300			

27.3 Add labels and lines for summaries

To make your report more readable, it would be useful to add labels for the row and column summaries. It would also be nice to have lines above the column summaries to better distinguish them from the cell values.

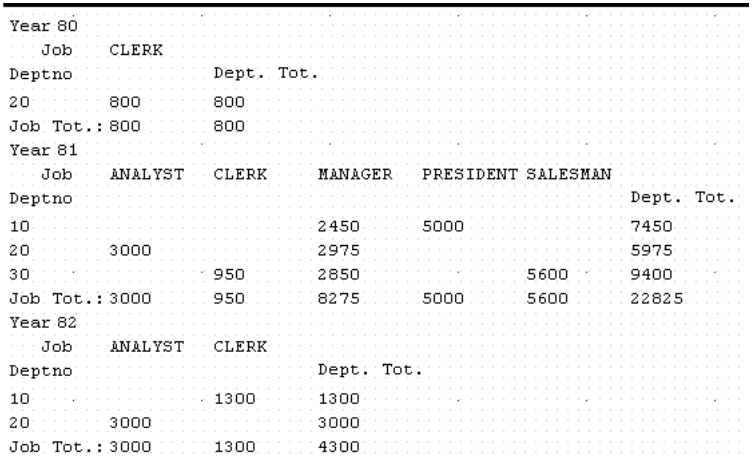
To add text:

1. In the Paper Design view, click on the **Text** tool from the toolbar on the left.
2. Click in the open space just below the first department number and to the left of the first total.
3. Enter `Job Tot.:`, then click in a blank area of the Paper Design view. If you perform this step correctly, the label should now appear for all of the column summaries in the report.
4. If necessary, use the arrow keys to better align the label with the summaries.

> **Tip:** If you want more precise movements, turn off **Snap to Grid** in the **View** menu. It may also be helpful for this step to turn off Flex mode by clicking the Flex Off button in the toolbar.

5. Click on the **Text** tool from the toolbar on the left.
6. Click in the open space just above the row summaries.
7. Enter `Dept. Tot.`, then click in a blank area of the Paper Design view. If you perform this step correctly, the label should now appear for all of the column summaries in the report.
8. If necessary, use the arrow keys to better align the label with the summaries.

Figure 27–3 Matrix with group above report output with summary labels



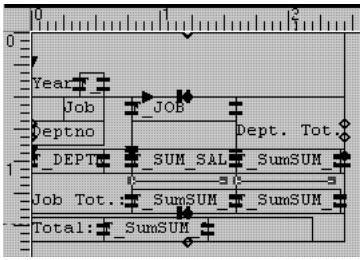
Year 80						
Job	CLERK					
Deptno		Dept. Tot.				
20	800	800				
Job Tot.:	800	800				
Year 81						
Job	ANALYST	CLERK	MANAGER	PRESIDENT	SALESMAN	
Deptno						Dept. Tot.
10			2450	5000		7450
20	3000		2975			5975
30		950	2850		5600	9400
Job Tot.:	3000	950	8275	5000	5600	22825
Year 82						
Job	ANALYST	CLERK				
Deptno			Dept. Tot.			
10		1300	1300			
20	3000		3000			
Job Tot.:	3000	1300	4300			

To add lines above column summaries

1. In the Paper Design view, click the Job Tot. label that you created in the previous section.
2. Shift-click the summaries to the right of Job Tot.
3. Click the Flex On button in the toolbar to set Flex mode on.
4. Click and drag Job Tot. and the summary fields down about 0.25 inches.
5. Click the Paper Layout button in the toolbar to display the Paper Layout view.
6. In the Paper Layout view, note the two spaces that now appear just above the **F_SumSUM_SALPerJOB** and **F_SumSUM_SALPerYEAR** fields.
7. Click the Line tool in the tool palette.

8. Click and drag a line in the space above **F_SumSUM_SALPerJOB**.
9. Repeat steps 7 and 8 to create a line above **F_SumSUM_SALPerYEAR**.

Figure 27–4 Layout model with lines above column summaries



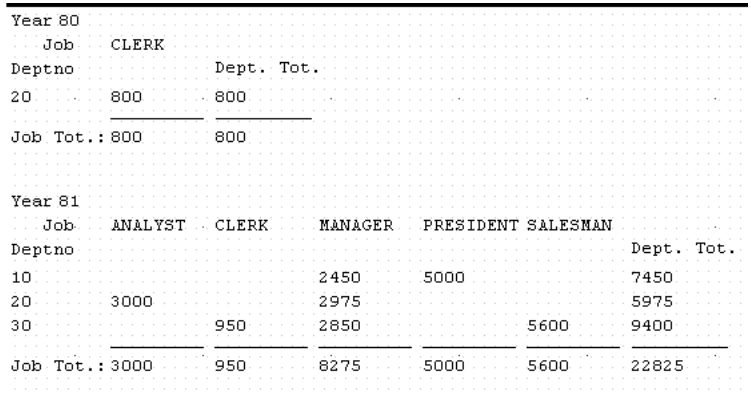
27.4 Add space between groups

Note that the output has no space between the groups.

To add more space:

1. In the Paper Layout view, click the Confine Off and Flex On buttons in the toolbar to set Confine mode off and Flex mode on.
2. Click the **Year** label to select it, then Shift-click the **F_YEAR** field to the right to select it, too.
3. Click and drag the **F_YEAR** field down about 0.25 inches.
4. Click the Paper Design button in the toolbar to preview your report output in the Paper Design view.

Figure 27–5 Matrix group above report output with added space



Year 80

Job	CLERK	
Deptno		Dept. Tot.
20	800	800
Job Tot.:	800	800

Year 81

Job	ANALYST	CLERK	MANAGER	PRESIDENT	SALESMAN	
Deptno						Dept. Tot.
10			2450	5000		7450
20	3000		2975			5975
30		950	2850		5600	9400
Job Tot.:	3000	950	8275	5000	5600	22825

27.5 Create a Web layout

Now that you have created a paper layout, suppose that you decide you would also like to have a Web layout for this report. You can quickly create a Web layout in the Report Wizard without changing your paper layout.

1. Choose **Tools > Report Wizard**.
2. Select **Create Web Layout only**.
3. Click **Finish**.
4. Click the Web Source view button in the toolbar and review the Web source for your Web layout.
5. Choose **Program > Run Web Layout** to preview your Web layout.

> **Note:** If Netscape 7.0 is your default browser, the browser may not display. You can work around this bug by making a copy of the Netscape 7.0 executable, naming it netscape.exe; with this name, the browser will display as expected.

27.6 Summary

Congratulations! You have successfully created a matrix group above report. You now know how to:

- create a matrix group above report with a single-query data model.
- add labels and line for summaries.
- add some white space between the master records.
- create a Web layout without changing the paper layout.

For more information on any of the wizards, views, or properties used in this example, refer to the *Reports Builder Online Help*, which you can access in two ways:

- From the Oracle Technology Network (`http://otn.oracle.com/products/reports/`), click **Documentation** and navigate to the *Reports Builder Online Help* for the most recent, hosted online help.
- From Reports Builder, choose **Help > Help Contents**.

> **Note:** You can replace your Reports Builder online help with the most recent update by downloading the latest online help set available on the Oracle Technology Network (`http://otn.oracle.com/docs/products/reports/`). Instructions for replacing your help file are included in the readme.txt in the download file.

Part V

Building Reports for Business Cases

The chapters in this Part provide steps to build reports that are intended for specific business purposes. The examples provided include computing values over specified periods of time, collecting data over a specified range, and grading data. Following the steps, you will learn how to print reports on pre-printed forms (for checks or invoices), split a report into sections and distribute each section to a different destination, and include a simple table or contents and index, or a multilevel table of contents.

- Chapter 28, "Building a Time Series Calculations Report"

  A time series calculations report computes values over a specified period of time. You can use the techniques used for this report to produce other formats of time series calculations. For example, you can build a report that calculates and displays the quarterly average of purchases for each customer.

- Chapter 29, "Building a Report with Aggregate Data"

  A report with aggregate data collects data within a range, retrieves values from the database, and formats them based on an aggregate range defined. You can use parameters to specify the range over which the data should be collected.

- Chapter 30, "Building a Check Printing Report with Spelled-Out Cash Amounts"

  A check printing report contains a stub and spelled-out cash amounts. You can create a PL/SQL function that returns spelled-out numerical values, imports an image of a pre-printed form (such as a blank check image), and prints your report on the form.

- Chapter 31, "Building a Report Using a Pre-Printed Form"

A report using a pre-printed form enables you to use formatting techniques to print reports on pre-printed forms when you do not have access to a computer readable version of the forms. Such reports must be designed so that the data prints in the exact positions on the form.

- Chapter 32, "Building an Invoice Report"

  An invoice report displays several distinguishing characteristics of a typical invoice, such as customer name and address, sales order number, billing information, and billing totals. You can import an image and print your report on a pre-printed form.

- Chapter 33, "Building a Ranking Report"

  A ranking report grades data in two different ways: by count and by percentage. You can set the ranking criteria at runtime or let the criteria default to previously specified values.

- Chapter 34, "Building a Paper Report with a Simple Table of Contents and Index"

  Navigational tools, such as a simple table of contents and an index, can be added to a paper report. First, the page numbering must be determined for the entire report, then Reports Builder can generate the table of contents and index based on the established page numbering.

- Chapter 35, "Building a Paper Report with a Multilevel Table of Contents"

  A multilevel table of contents can be added to a paper report to facilitate navigation. This table of contents is created similarly to that in Chapter 36, "Building a Paper Report with a Simple Table of Contents and Index", however with a hierarchy that categorizes the information in the table of contents.

- Chapter 36, "Bursting and Distributing a Report"

  A report built for bursting and distributing a report enables you to simultaneously deliver a single report to multiple destinations. You can create a single report, burst each section to a separate report, and then send each section in any format (such as, PDF or HTML) to multiple destinations.

28

Building a Time Series Calculations Report

Figure 28–1 Time series calculations report output

Custid	Shipdate	Total	4-Month Moving Average
100	30-JUL-86	$3.40	$3.40
	15-AUG-86	$97.50	$50.45
	01-JAN-87	$730.00	$730.00
	12-MAR-87	$4,450.00	$2,590.00
101	08-JAN-87	$101.40	$101.40
102	05-JUN-86	$224.00	$224.00
	20-JUN-86	$56.00	$140.00
	11-JAN-87	$45.00	$45.00
	05-FEB-87	$23,940.00	$11,992.50
	06-MAR-87	$3,510.50	$9,165.17
103	10-FEB-87	$764.00	$764.00
104	18-JUL-86	$5.60	$5.60
	25-JUL-86	$35.20	$20.40

In this example, you will build a report that calculates and displays the four-month average of purchases for each customer.

Concepts

- Reports with time series calculations calculate values over a specified period of time. The techniques described for this report can be used to produce other formats of time series calculations, as well.

Data Relationships

- This time series calculations report uses a query that will compute four-month moving averages of customer purchases. The SELECT statement will sum the current purchase (TOTAL) with purchases made by that customer in the previous four months, then average that sum through use of a self-join. For example, if the data queried is 30-JUL-00, Oracle Reports will average all purchases the customer made between 30 MAR-00 and 30-JUL-00.

Example Scenario

This chapter will show you how to use the Report Wizard to create a simple time series calculations report for both paper and the Web. For the JSP-based Web report, you will modify the Web source to change labels and add format masks.

To see a sample time series calculations report, open the examples folder named `timeseries`, then open the Oracle Reports example called `timeseries.rdf`. For details on how to open it, see "Accessing the example reports" in the Preface.

Table 28–1 Features demonstrated in this example

Feature	Location
Use the Report Wizard to create a data model and layout for both the paper and Web reports.	Section 28.2, "Create a query and the layout"
Modify the JSP to generate the JSP-based Web report.	Section 28.3, "Modify the Web source of your JSP report"

28.1 Prerequisites for this example

To build the example in this chapter, you must have access to the SUMMIT schema, which you can download from the Oracle Reports Documentation page on the Oracle Technology Network (`http://otn.oracle.com/products/reports/`).

28.2 Create a query and the layout

The steps in this section will show you how to build a simple data model and report layout in the Report Wizard, which you can then use to generate either a JSP-based Web report or a paper report. In the next section, you will modify the JSP so that the appropriate information displays in your Web report.

When you are creating a single-query report, such as this one, you can use the Report Wizard to create the data model and layout simultaneously.

To create a data model and layout:

1. Launch Reports Builder (or, if already open, choose **File > New > Report**).
2. In the Welcome or New Report dialog box, select **Use the Report Wizard**, then click **OK**.
3. If the Welcome page displays, click **Next**.
4. On the Report Type page, select **Create both Web and Paper Layout**, then click **Next**.
5. On the Style page, type a **Title** for your report, select **Group Left**, then click **Next**.
6. On the Data Source page, click **SQL Query**, then click **Next**.
7. On the Data page, enter the following SELECT statement in the **Data Source definition** field:

```
SELECT O.CUSTID, O.SHIPDATE, O.TOTAL,
AVG(A.TOTAL) MAVG
FROM ORD O, ORD A
WHERE A.CUSTID = O.CUSTID
AND A.SHIPDATE BETWEEN O.SHIPDATE -123 AND O.SHIPDATE
GROUP BY O.CUSTID, O.SHIPDATE, O.TOTAL
```

> **Note:** You can enter this query in any of the following ways:
>
> - Copy and paste the code from the provided text file called `timeseries_code.txt` into the **Data Source definition** field.
> - Click **Query Builder** to build the query without entering any code manually.
> - Type the code in the **Data Source definition** field.

8. Click **Next**.

> **Note:** If you are not already connected to a database, you will be prompted to connect to the database when you click **Next**. Ensure that you connect to a database that has the appropriate schema for this example. Section 28.1, "Prerequisites for this example" describes the sample schema requirements for this example.

9. On the Groups page, click **CUSTID** and click the right arrow (**>**) to move this field to the **Group Fields** list, then click **Next**.
10. On the Fields page, click the double right arrows (**>>**) to move all of the fields to the **Displayed Fields** list, then click **Next**.
11. On the Totals page, click **Next**.
12. On the Labels page, change the label for **MAVG** to `4-Month Moving Average`, then click **Next**
13. On the Template page, select **Predefined Template** and click **Beige**, then click **Finish** to display your report output in the Paper Design view.
14. In the Paper Design view, double-click field **F_MAVG** to display the Property Inspector, and set properties:
    - Under **Field**, set the Format Mask property to LNNNGNN0D00.
15. Do the same for field **F_TOTAL**.
16. The report in the Paper Design view should now look something like this:

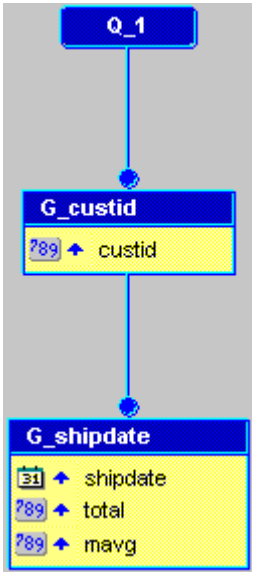
Figure 28–2 Paper Design view of the Time Series Calculations report

Shipdate	Total	4-Month moving Average
30-JUL-86	$3.40	$3.40
15-AUG-86	$97.50	$50.45
01-JAN-87	$730.00	$730.00
12-MAR-87	$4,450.00	$2,590.00
08-JAN-87	$101.40	$101.40
05-JUN-86	$224.00	$224.00
20-JUN-86	$56.00	$140.00
11-JAN-87	$45.00	$45.00
05-FEB-87	$23,940.00	$11,992.50

17. To view the data model you just created using the Report Wizard, click the Data Model button in the toolbar. The same data model can be used for both your paper and your JSP-based Web reports.

Your data model should look something like this:

Figure 28–3 Data Model view of the Time series Calculations report



18. Save your report as `timeseries_<your initials>.rdf`.

28.3 Modify the Web source of your JSP report

Now that you've created your paper report layout, you can take the same report and generate a JSP-based Web report that looks the same as your paper report.

To modify your JSP-based Web report:

1. Save your report, `timeseries_<your initials>.rdf` as a JSP under the same name (`timeseries_<your initials>.jsp`).
2. Click the Web Source button in the toolbar to display the Web Source view.

   In the Web Source, you need to change the format mask to match that of the paper report.
3. In the Web Source view, find the text:

```
<td <rw:headers id="HFtotal" src="HBtotal, HBcustid, HFcustid"/>
class="OraCellNumber"><rw:field id="F_total" src="total" nullValue=" ">
```

```
F_total </rw:field></td>
```

4. Add the tag `formatMask="LNNNGNN0D00"` to the line, so that the line looks like this:

```
<td <rw:headers id="HFtotal" src="HBtotal, HBcustid, HFcustid"/>
class="OraCellNumber"><rw:field id="F_total" src="total"
formatMask="LNNNGNN0D00" nullValue=" "> F_total </rw:field></td>
```

> **Note:** The bold text is the new format mask tag that we've added to the Total field.

5. Change the format mask for the **MAVG** field. The resulting line in the Web Source should look like this:

```
<td <rw:headers id="HFmavg" src="HBmavg, HBcustid, HFcustid"/>
class="OraCellNumber"><rw:field id="F_mavg" src="mavg"
formatMask="LNNNGNN0D00" nullValue=" "> F_mavg </rw:field></td>
```

6. Save your report as a JSP.
7. Click the Run Web Layout button in the toolbar to display your new JSP-based Web report in your browser. The report should look something like this:

> **Note:** If Netscape 7.0 is your default browser, the browser may not display. You can work around this bug by making a copy of the Netscape 7.0 executable, naming it netscape.exe; with this name, the browser will display as expected.

Figure 28–4 Time Series Calculations JSP-based Web Report

Custid	Shipdate	Total	4-Month Moving Average
100	30-JUL-86	$3.40	$3.40
	15-AUG-86	$97.50	$50.45
	01-JAN-87	$730.00	$730.00
	12-MAR-87	$4,450.00	$2,590.00
101	08-JAN-87	$101.40	$101.40
102	05-JUN-86	$224.00	$224.00
	20-JUN-86	$56.00	$140.00
	11-JAN-87	$45.00	$45.00
	05-FEB-87	$23,940.00	$11,992.50
	06-MAR-87	$3,510.50	$9,165.17
103	10-FEB-87	$764.00	$764.00
104	18-JUL-86	$5.60	$5.60
	25-JUL-86	$35.20	$20.40

The report displays the total for each customer, as well as the average over the past four months.

Note: For information on creating a Parameter Form for a JSP-based Web report, refer to the documented example in Getting Started with Oracle Reports, located on the Oracle Technology Network (`http://otn.oracle.com/products/reports/`).

28.4 Summary

Congratulations! You have successfully created a time series calculations report for both paper and Web. You now know how to:

- create a time series calculations report definition.
- modify your report for the Web.

For more information on any of the wizards, views, or properties used in this example, refer to the *Reports Builder Online Help*, which you can access in two ways:

- From the Oracle Technology Network (`http://otn.oracle.com/products/reports/`), click **Documentation**

and navigate to the *Reports Builder Online Help* for the most recent, hosted online help.

- From Reports Builder, choose **Help > Help Contents**.

Note: You can replace your Reports Builder online help with the most recent update by downloading the latest online help set available on the Oracle Technology Network (`http://otn.oracle.com/docs/products/reports/`). Instructions for replacing your help file are included in the readme.txt in the download file.

29

Building a Report with Aggregate Data

Figure 29–1 Aggregate data report output

Salary Range			Name	Dept
0	-	1000	SMITH	20
			JAMES	30
1000	-	2000	ADAMS	20
			WARD	30
			MARTIN	30
			MILLER	10
			TURNER	30
			ALLEN	30
2000	-	3000	CLARK	10
			BLAKE	30
			JONES	20
3000	-	4000	SCOTT	20
			FORD	20
5000	-	6000	KING	10

In this chapter, you will build a report that collects and displays names of all employees whose salaries fall within the range of 0 to 999, then collects and displays all employees whose salaries fall within the range of 1000 to 1999, etc. You will be able to modify this report to display any aggregate range you need.

Concepts

- In a report that aggregates, or collects, data within ranges, values from the database are retrieved and formatted based on an aggregate range that you

define. You can even use parameters to specify the range over which the data should be collected.

Data Relationships

- This report uses two "functions" in its SELECT statement to specify the aggregate range. The functions are (FLOOR(SAL/1000))*1000), which calculates the lowest salary, and (CEIL*((SAL+1)/1000) * 1000), which calculates the highest salary. The columns that receive the values of these functions are placed into a break group to produce the control break format of this example report.

Layout

- This report uses the Group Left layout style.

Example Scenario

This chapter will show you how to use the Report Wizard to create the Web and paper layout and report definition. This report will fetch and display data based on an aggregate range of increments of one thousand, starting at zero. You will then modify the Web source of your report.

To see a sample report that aggregates data within ranges, open the examples folder named `aggregatingdata`, then open the Oracle Reports example named `aggregatingdata.rdf`. For details on how to open it, see "Accessing the example reports" in the Preface.

Table 29–1 Features demonstrated in this example

Feature	Location
Use the Report Wizard to create a data model and layout for both the paper and Web reports.	Section 29.2, "Create a query and the layout"
Modify the JSP to generate the JSP-based Web report.	Section 29.3, "Modify the Web source of your JSP report"

29.1 Prerequisites for this example

To build the example in this chapter, you must have access to the EMP and DEPT schema, which is provided by default with the Oracle9*i* database. The user ID and password for accessing this schema is scott/tiger.

29.2 Create a query and the layout

The steps in this section will show you how to build a simple data model and report layout in the Report Wizard, which you can then use to generate either a JSP-based Web report or a paper report. In the next section, you will modify the JSP so that the appropriate information displays in your Web report.

When you are creating a single-query report, such as this one, you can use the Report Wizard to create the data model and layout simultaneously.

To create a data model and layout:

1. Launch Reports Builder (or, if already open, choose **File > New > Report**).
2. In the Welcome or New Report dialog box, select **Use the Report Wizard**, then click **OK**.
3. If the Welcome page displays, click **Next**.
4. On the Report Type page, select **Create both Web and Paper Layout**, then click **Next**.
5. On the Style page, type a **Title** for your report, select **Group Left**, then click **Next**.
6. On the Data Source page, click **SQL Query**, then click **Next**.
7. On the Data page, enter the following SELECT statement in the **Data Source definition** field:

```
SELECT (FLOOR(SAL/1000))*1000 BOTTOM,
CEIL((SAL+1)/1000) * 1000 TOP,
ENAME,
DEPTNO
FROM EMP
ORDER BY 1,2, SAL
```

Note: You you can enter this query in any of the following ways:

- Copy and paste the code from the provided text file called `aggregatingdata_code.txt` into the **Data Source definition** field.
- Click **Query Builder** to build the query without entering any code manually.
- Type the code in the **Data Source definition** field.

8. Click **Next**.

Note: If you are not already connected to a database, you will be prompted to connect to the database when you click **Next**. Ensure that you connect to a database that has the appropriate schema for this example. Section 29.1, "Prerequisites for this example" describes the sample schema requirements for this example.

9. On the Groups page, click the following fields in the **Available Fields** list and click the right arrow (>) to move them to the **Group Fields** list, then click **Next**:
   - **BOTTOM**
   - **TOP**
10. On the Fields page, click the double right arrows (>>) to move all of the fields to the **Displayed Fields** list, then click **Next**.
11. On the Totals page, click **Next.**
12. On the Labels page, click **Next.**
13. On the Template page, select **Predefined Template** and click **Beige**, then click **Finish** to display your report output in the Paper Design view. It should look something like this:

Figure 29–2 Paper Design view for the aggregating data report

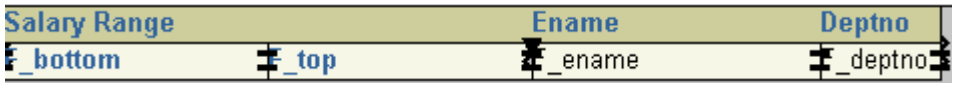
Bottom	Top	Ename	Deptno
0	1000	SMITH	20
		JAMES	30
1000	2000	ADAMS	20
		WARD	30
		MARTIN	30
		MILLER	10
		TURNER	30
		ALLEN	30
2000	3000	CLARK	10
		BLAKE	30
		JONES	20
3000	4000	SCOTT	20
		FORD	20
5000	6000	KING	10

14. In the Paper Design view, click the text **Bottom** and change the text to `Salary Range`.
15. Delete the text **Top**.

> **Note:** You can also modify the other column headers to make the text more meaningful.

16. Click the Paper Layout button in the toolbar to display the Paper Layout view. The layout currently looks like this:

Figure 29–3 Paper Layout view of the Aggregating Data report

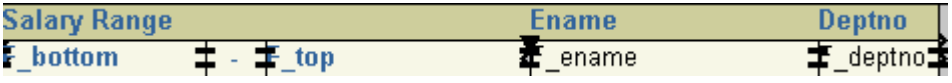


17. In the Paper Layout view, click the Flex Off button in the toolbar to set Flex mode off.
18. Click the right edge of the **f_bottom** frame and drag it to the left about 0.5 inches to make room between the **f_bottom** and **f_top** fields.

    Tip: To find objects in the Paper Layout view, you can use the Object Navigator. When you click an item name in the Object Navigator, the corresponding object is selected in the Paper Layout view.

19. Click the Text tool in the tool palette.
20. Drag a boilerplate text object between **f_bottom** and **f_top**, then type "-" in the text box. The layout should now look like this:

Figure 29–4 Paper Layout view with new boilerplate text



21. Now, click the Run Paper Layout button in the toolbar to display your report. It should look something like this.

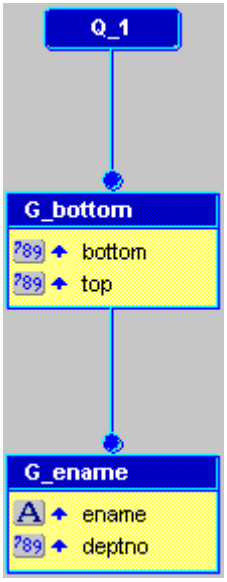
Figure 29–5 Paper Design view of the Aggregating Data report

Salary Range			Name	Dept
0	-	1000	SMITH	20
			JAMES	30
1000	-	2000	ADAMS	20
			WARD	30
			MARTIN	30
			MILLER	10
			TURNER	30
			ALLEN	30
2000	-	3000	CLARK	10
			BLAKE	30
			JONES	20
3000	-	4000	SCOTT	20
			FORD	20
5000	-	6000	KING	10

Note: Notice how the report displays the employee names per salary range, hence aggregating the data.

22. Save your report as `aggregatereport_<your initials>.rdf`.
23. Take a look at the data model of your report. You can use this data model to generate either a paper report or a JSP-based Web report. To view the data model, click the Data Model button in the toolbar. Your data model should look something like this:

Figure 29–6 Data Model for the aggregating data report



29.3 Modify the Web source of your JSP report

Now that you've create your paper report layout, you'll now learn how to take the same report and generate a JSP-based Web report that looks the same as your paper report.

To modify your JSP-based Web report:

1. Save your report, `aggregatereport_<your initials>.rdf` as a JSP under the same name (`aggregatereport_<your initials>.jsp`).
2. Click the Web Source button in the toolbar to display the Web Source view.

   In the Web Source, you need to change the column titles to match those of the paper report.
3. In the Web Source view, find the text:

```
<th <rw:id id="HBbottom" asArray="no"/> class="OraColumnHeader"> Bottom
</th>
```

Tip: Choose **Edit > Find and Replace**, then type "bottom" in the **Find what** text box.

4. Change the column header to "Salary" so that the line now looks like this:

```
<th <rw:id id="HBbottom" asArray="no"/> class="OraColumnHeader"> Salary
</th>
```

5. The next line of code indicates the header for the "Top" column. Delete the header text so that the line looks like this:

```
<th <rw:id id="HBtop" asArray="no"/> class="OraColumnHeader">  </th>
```

6. Change the other two columns for Ename and Deptno as desired. For example, we changed them to "Name" and "Dept", like so:

```
<th <rw:id id="HBename" asArray="no"/> class="OraColumnHeader"> Name </th>
<th <rw:id id="HBdeptno" asArray="no"/> class="OraColumnHeader"> Dept </th>
```

7. Save your report as a JSP.
8. Click the Run Web Layout button in the toolbar to display your new JSP-based Web Report in your browser. The report should look something like this:

Note: If Netscape 7.0 is your default browser, the browser may not display. You can work around this bug by making a copy of the Netscape 7.0 executable, naming it netscape.exe; with this name, the browser will display as expected.

Figure 29–7 Aggregating Data JSP-based Web Report

Salary		Name	Dept
0	1000	SMITH	20
		JAMES	30
1000	2000	ADAMS	20
		WARD	30
		MARTIN	30
		MILLER	10
		TURNER	30
		ALLEN	30
2000	3000	CLARK	10
		BLAKE	30
		JONES	20
3000	4000	SCOTT	20
		FORD	20
5000	6000	KING	10

The report displays, in Web format, the aggregate data you specified. Here, you can see which employees fall into the specified salary ranges.

Note: For information on creating a Parameter Form for a JSP-based Web report, refer to the documented example in Getting Started with Oracle Reports, located on the Oracle Technology Network (`http://otn.oracle.com/products/reports/`).

29.4 Summary

Congratulations! You have successfully built a report that aggregates data, for both paper and Web. You now know how to:

- create a report definition that aggregates data.
- modify your report for the Web.

For more information on any of the wizards, views, or properties used in this example, refer to the *Reports Builder Online Help*, which you can access in two ways:

- From the Oracle Technology Network (`http://otn.oracle.com/products/reports/`), click **Documentation**

and navigate to the *Reports Builder Online Help* for the most recent, hosted online help.

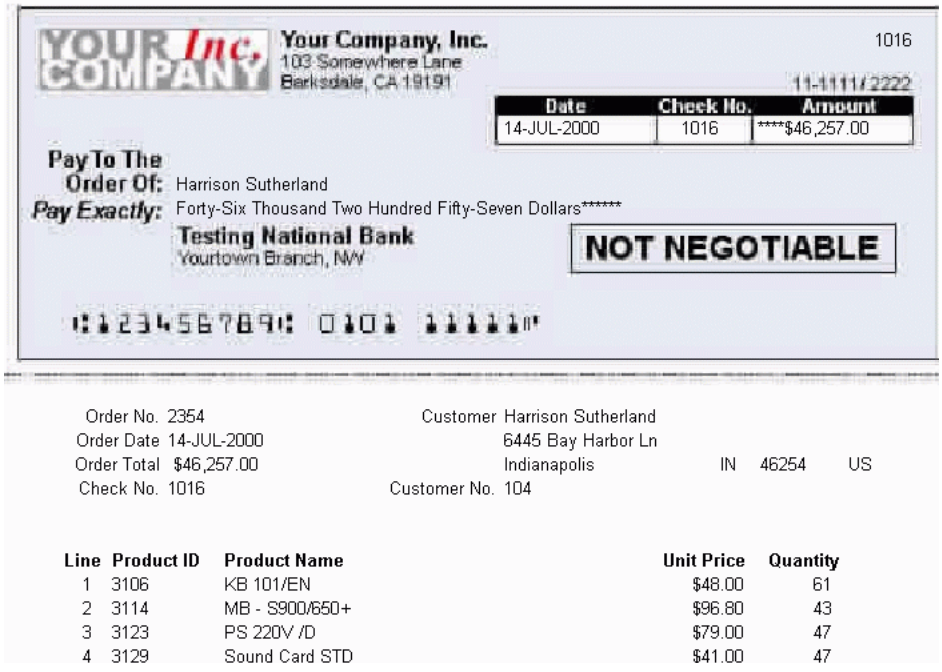
- From Reports Builder, choose **Help > Help Contents**.

> **Note:** You can replace your Reports Builder online help with the most recent update by downloading the latest online help set available on the Oracle Technology Network (`http://otn.oracle.com/docs/products/reports/`). Instructions for replacing your help file are included in the readme.txt in the download file.

30

Building a Check Printing Report with Spelled-Out Cash Amounts

Figure 30–1 Check printing report output



In this chapter, you will build a check printing report with a stub and spelled-out cash amounts. The steps described in this chapter will help you create a PL/SQL

function that returns spelled-out numerical values. You will also learn how to import an image of a pre-printed form (in this case, a blank check image) and use the tools in the Paper Layout and Paper Design views to print your report on such a form. Although we use a check as the example in this report, you can use the steps to use any pre-printed form with Oracle Reports.

Note: Many of the same concepts are used in the following three example reports

- This example, where you import an image of a check and use it as a guide to position fields in the Paper Layout view. In addition, you learn how to create a PL/SQL function that returns spelled-out numerical values.
- Chapter 31, "Building a Report Using a Pre-Printed Form", where you learn formatting techniques for printing reports on pre-printed forms when you do not have access to a computer readable version of the forms. Such reports must be designed so that the data prints in exact positions on the form.
- Chapter 32, "Building an Invoice Report", where you import an image of an invoice and use it as a guide to position fields in the Paper Layout view.

Concepts

- To spell out the value of a check, you will use a PL/SQL function to split the number into its constituent numerals, then use a formula column to combine the words into the spelled-out cash amount.

Data Relationships

- This report uses two linked queries. The first query retrieves the information necessary for the check, and the second query retrieves the order details that will be printed on the check stub.
- One of the queries you'll create assumes that the numbers to be spelled are stored in the TOTAL column of the ORDERS table. The text of the spelled-out number will be fetched from the table called Lookup, and returned in the Thousands_text, Thousands_Symbol, Ones_text, and Decimal_Text columns

Layout

- This report uses a default Form style.

Example Scenario

This chapter will show you how to use the Report Wizard to create your basic report definition. You will create a PL/SQL function that spells out the numerical value of the check amount. You will also create a formula column that will format the dollar amounts on your checks.

You will use the tools in the Paper Layout view to import a blank check image, which is provided to you in the example report directory. You will then use the tools in the Paper Layout and Paper Design views to rearrange the fields in your report according to how you want them to display on the resulting check printing report. You will also use these tools to create a stub for every check.

To see a sample report where the cash amounts are spelled out, open the examples folder named `spellcash`, then open the Oracle Reports example named `spellcash.rdf`. For details on how to open it, see "Accessing the example reports" in the Preface.

Table 30–1 Features demonstrated in this example

Feature	Location
Use the Report Wizard to create a simple form report.	Section 30.2, "Create a report using the Report Wizard"
Use the PL/SQL Editor to create a PL/SQL function and the Data Model view to create a formula column.	Section 30.3, "Create a formula column that returns the spelled-out cash amounts"
Use the Data Model view to create a second query and link the two queries together.	Section 30.4, "Create a query that will return the items in the order"
Use the Paper Layout view to import a blank check image and add fields necessary for printing the check and check stub. Use the Paper Design view to arrange the new fields on the blank check image.	Section 30.5, "Import a check image and arrange fields for printing"
Use the Paper Design view to create a check stub and the Report Block Wizard to create a simple tabular report that displays order details.	Section 30.6, "Create a check stub with payment information and order details"

30.1 Prerequisites for this example

To build the example in this chapter, you must have access to the Order Entry schema, which is provided by default with the Oracle9*i* database. If you don't know if you have access to the sample schema provided with the Oracle9*i* database, contact your database administrator.

30.2 Create a report using the Report Wizard

The steps in this section will show you how to use the Report Wizard to create a simple report definition.

To create a report definition:

1. Launch Reports Builder (or, if already open, choose **File > New > Report**)
2. In the Welcome or New Report dialog box, select **Use the Report Wizard**, then click **OK**.
3. If the Welcome page displays, click **Next**.
4. On the Report Type page, select **Create Paper Layout Only**, then click **Next**.
5. On the Style page, type a **Title** for your report, select **Form**, then click **Next**.
6. On the Data Source page, click **SQL Query**, then click **Next**.
7. On the Data page, enter the following SELECT statement in the **Data Source definition** field:

```
SELECT ALL ORDERS.ORDER_ID, TO_CHAR(ORDERS.ORDER_DATE, 'DD-MON-YYYY')
ORDER_DATE, ORDERS.CUSTOMER_ID, ORDERS.ORDER_TOTAL,
CUSTOMERS.CUST_FIRST_NAME || ' ' || CUSTOMERS.CUST_LAST_NAME CUSTOMER_NAME,
CUSTOMERS.CUST_ADDRESS, (ROWNUM + 1000) AS CHECK_NO
FROM ORDERS, CUSTOMERS
WHERE CUSTOMERS.CUSTOMER_ID = ORDERS.CUSTOMER_ID
ORDER BY ORDERS.ORDER_ID ASC
```

Note: You can enter this query in any of the following ways:

- Copy and paste the code from the provided text file called `spellcash_code.txt` into the **Data Source definition** field.
- Click **Query Builder** to build the query without entering any code manually.
- Type the code in the **Data Source definition** field.

8. Click **Next**.

Note: If you are not already connected to a database, you will be prompted to connect to the database when you click **Next**. Ensure that you connect to a database that has the appropriate schema for this example. Section 30.1, "Prerequisites for this example" describes the sample schema requirements for this example.

9. On the Fields page, click the double right arrows (>>) to move all of the fields to the **Displayed Fields** list, then click **Next**.
10. On the Totals page, click **Next**.
11. On the Labels page, change the labels and field widths as follows, then click **Next**:

Table 30–2 Field description of Labels page

Fields	Labels	Width
ORDER_ID	Order No.	5
ORDER_DATE	Order Date	12
CUSTOMER_ID	Customer No.	3
ORDER_TOTAL	Order Total	15
CUSTOMER_NAME	Customer	40
C_STREET_ADDRESS	[blank]	40

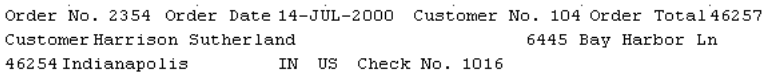
Table 30–2 Field description of Labels page

Fields	Labels	Width
C_POSTAL_CODE	[blank]	5
C_CITY	[blank]	20
C_STATE_PROVINCE	[blank]	3
C_COUNTRY_ID	[blank]	3
CHECK_NO	Check No.	5

12. On the Template page, select **No Template**, then click **Finish** to preview your report output in the Paper Design view.

Since you've only created the initial report definition, the formatting will not display like a check. It should look something like this:

Figure 30–2 Paper Design view of the simple report



13. Save your report as `spellcash_<your initials>.rdf`.

30.3 Create a formula column that returns the spelled-out cash amounts

The steps in this section will show you how to create a formula column that is based on a function. The function you create will return verbal or word value for the numerical value of the check. You will then create a formula column that will spell out the numerical value in the designated currency. In this case, we will use dollars and cents.

30.3.1 Create a PL/SQL function

In this section, you will create a function that simply returns the check amount in word format, such as "twenty-four sixty-five." The formula column you create in the

next section will use the information retrieved by this function to spell out the cash amounts on your checks.

To create a PL/SQL function:

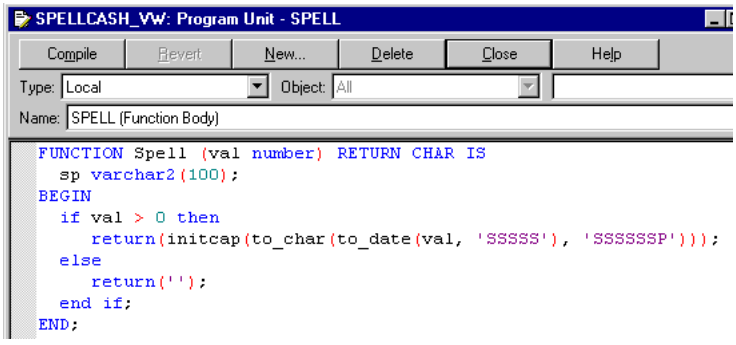
1. In the Object Navigator, under your report name, double-click **Program Units**.
2. In the New Program Unit dialog box, type "`Spell`" in the **Name** box.
3. Select **Function**, then click **OK**.
4. In the PL/SQL Editor, type the following code:

```
FUNCTION Spell (val number) RETURN CHAR IS
  sp varchar2(100);
BEGIN
  if val > 0 then
   return(initcap(to_char(to_date(val, 'SSSSS'), 'SSSSSSP')));
  else
   return('');
  end if;
END;
```

> **Note:** You can enter this code by copying and pasting it from the provided text file called `spellcash_code.txt` into the PL/SQL Editor.

5. Click **Compile**. If you see any errors, compare your code against the code we've provided, which is shown in the image below:

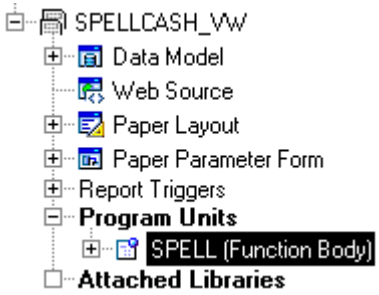
Figure 30–3 PL/SQL Editor displaying the SPELL function



6. When your code successfully compiles, click **Close**.

Your new function now displays in the Object Navigator:

Figure 30–4 Object Navigator with SPELL PL/SQL function



7. Save your report.

30.3.2 Create a formula column in your data model

In this section, you will create a formula column that uses the information retrieved by the Spell function you created in Section 30.3.1, "Create a PL/SQL function". This formula column will use the verbal values of the check amounts and combine the

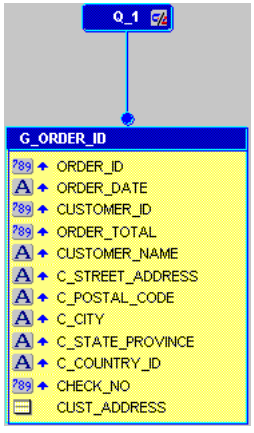
words with the correct currency. For example, the "twenty-four sixty-five" returned by the Spell function will be turned into "twenty-four dollars and sixty-five cents".

To create a formula column:

1. In the Object Navigator, under your report name, double-click the view icon next to the **Data Model** node to display the Data Model view.

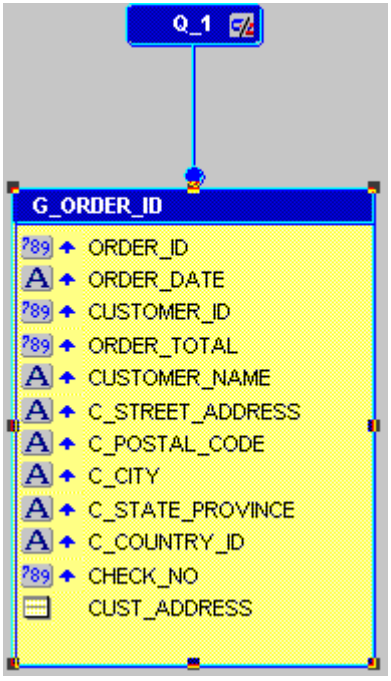
   Your data model should look like this:

Figure 30–5 Data Model view



2. Resize the **G_ORDER_ID** box by clicking at the top, then dragging the bottom center resize handle downwards. Your data model should now look something like this:

Figure 30–6 Resized Data Model view



3. Click the Formula Column tool in the tool palette.
4. Click in the **G_ORDER_ID** group, in the space you just created, to create a new formula column.
5. Double-click the new formula column object (**CF_1**) to display the Property Inspector, and set properties:
   - Under **General Information**, set the Name property to SPELLED_AMOUNT.
   - Under **Column**, set the Datatype property to CHARACTER, and the Width property to 100.
   - Under **Placeholder/Formula**, click the PL/SQL Formula property field to display the PL/SQL Editor.
6. In the PL/SQL Editor, use the template to enter the following PL/SQL code:

```
 function SPELLED_AMOUNTFormula return Char is
cents number;
c_str varchar2(80);
val number;
begin
  val := :order_total;
  cents := (val mod 1) * 100;
  if cents > 0 then --creates string for cents
  c_str := ' and ' || TO_CHAR(cents) || '/100 Dollars******';
   else
  c_str := ' Dollars******';
  end if;
  if val < 1000 and val > 1 then
  return (initcap(spell(floor(val))) || c_str);
  elsif val > 1000 then
   return(initcap(spell(floor(val/1000))) || ' Thousand ' ||
   spell(floor(val mod 1000)) || c_str);
  else
   return('Zero'||c_str);
  end if;
end;
```

> **Note:** You can enter this code by copying and pasting it from the provided text file called `spellcash_code.txt` into the PL/SQL Editor.

7. Click **Compile**. If you see any errors, compare your code against the code we've provided, which is shown in the image below:

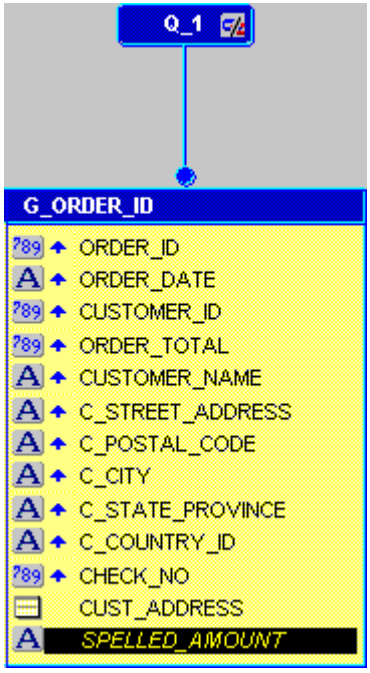
Figure 30–7 PL/SQL Code for SPELLED_AMOUNT formula column

```
function SPELLED_AMOUNTFormula return Char is
 cents number;
 c_str varchar2(80);
 val number;
begin
 val := :order_total;
 cents := (val mod 1) * 100;
 if cents > 0 then --creates string for cents
   c_str := ' and ' || TO_CHAR(cents) || '/100 Dollar******';
   else
   c_str := ' Dollars******';
 end if;
 if val < 1000 and val > 1 then
   return (initcap(spell(floor(val))) || c_str);
 elsif val > 1000 then
   return(initcap(spell(floor(val/1000))) || ' Thousand ' ||
    spell(floor(val mod 1000)) || c_str);
 else
     return('Zero'||c_str);
 end if;
end;
```

8. When your code successfully compiles, click **Close**.

Your new formula column, called SPELLED_AMOUNT, now displays in the data model.

Figure 30–8 Data Model with SPELLED_AMOUNTS formula column



9. Save your report.

30.4 Create a query that will return the items in the order

The steps in this section will show you how to manually create a query in the Data Model view that will return the items in the customer's order. This data retrieved will be used to display the order details in the check stub.

To manually create a query:

1. In the Data Model view, click the SQL Query tool in the tool palette.
2. Click an open area in the Data Model view to display the SQL Query Statement dialog box.
3. In the **SQL Query Statement** field, type the following code:

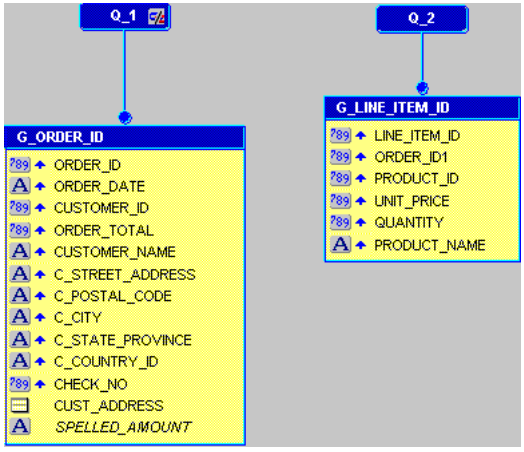
```
SELECT ALL
    ORDER_ITEMS.LINE_ITEM_ID,
    ORDER_ITEMS.ORDER_ID,
    ORDER_ITEMS.PRODUCT_ID,
    ORDER_ITEMS.UNIT_PRICE,
    ORDER_ITEMS.QUANTITY,
    PRODUCT_INFORMATION.PRODUCT_NAME
FROM ORDER_ITEMS, PRODUCT_INFORMATION
WHERE PRODUCT_INFORMATION.PRODUCT_ID=ORDER_ITEMS.PRODUCT_ID
ORDER BY ORDER_ITEMS.LINE_ITEM_ID ASC
```

Note: You can enter this query in any of the following ways:

- Copy and paste the code from the provided text file called `spellcash_code.txt` into the **SQL Query Statement** field.
- Click **Query Builder** to build the query without entering any code manually.
- Type the code in the **SQL Query Statement** field.

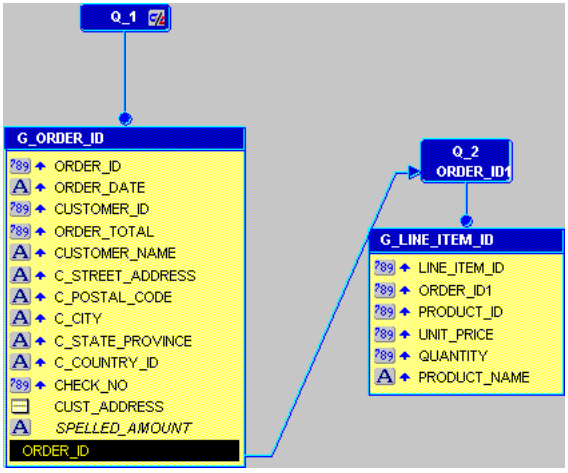
4. Click **OK**. Your query displays in the Data Model view, and should look something like this:

Figure 30–9 Data Model with two queries



5. In the Data Model view, click the Data Link tool in the tool palette.
6. In the first query, Q_1, under G_ORDER_ID, click **ORDER_ID**, then drag the line to **ORDER_ID1** in Q_2.
7. Your data model should now look like this:

Figure 30–10 Data Link between two queries



8. Save your report.

30.5 Import a check image and arrange fields for printing

The steps in this section will show you how to adjust the margins of your check printing report and align the fields with an image of a check. You can scan any check and use its image to lay out the objects of your check report. In this section, we use the image we've provided to you, called `blankcheck.jpg`. This image is located in the `spellcash` example folder.

30.5.1 Rearrange the layout objects

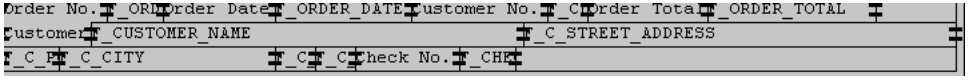
Before you can insert the check image, you must first rearrange the layout objects.

To rearrange the layout objects:

1. Click the Paper Layout button in the toolbar to display the Paper Layout view.

   The layout currently looks like this:

Figure 30–11 Paper Layout view of your report

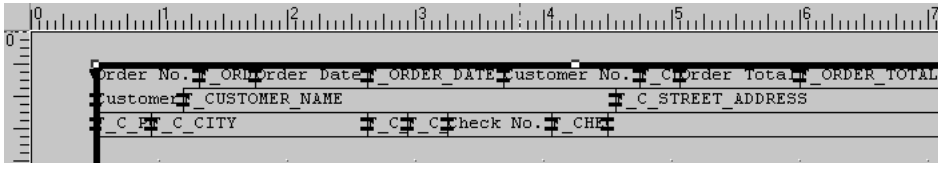


2. Click the Edit Margin button in the toolbar.
3. In the Paper Layout view, click the margin frame (the heavy black line) to select it.
4. At the top of the frame, click the center black resizing square and drag it up, so that the margin is 0.25 inches.

   Tip: When you click the center square, notice that a dotted guideline displays while your mouse button is depressed. You can use these guidelines to help place objects exactly where you want them in the Paper Layout view.

   When you're done, it should look something like this:

Figure 30–12 Paper Layout view with resized margin



5. Save your report.

30.5.2 Import the blank check image

In this section, you will import an image of a blank check. You can use any check you'd like. For this example, we've provided an image called `blankcheck.jpg` in the **spellcash** example directory.

To import the blank check image:

1. While the frame is still selected, choose **Insert > Image**.
2. In the Import Image dialog box, make sure that **File** is selected.
3. In the text box next to **File**, type or browse for the location of the image, `blankcheck.jpg`, then click **OK** to display the blank check image in the Paper Layout view.
4. Click the Edit Margin tool in the toolbar to return to edit mode. Notice that the image no longer displays.

30.5.3 Set up the check printing fields

The steps in this section will show you how to use the various tools in the Paper Layout view to modify the look and feel of your check report. Here, you will learn how to add and modify layout objects and fields according to how you want them to display on the resulting checks. You will create copies of certain fields that you will then use in Section 30.6, "Create a check stub with payment information and order details" to create the check stub.

To set up the check printing fields:

1. In the Paper Layout view, click the **Order No**. boilerplate text.
2. Click the Select Parent Frame button in the toolbar

   The repeating frame called R_G_ORDER_ID should now be selected in the Paper Layout view. You can also look in the Object Navigator, under Paper Layout, to make sure R_G_ORDER_ID is selected.
3. While R_G_ORDER_ID is selected, click and drag the bottom center resize handle to 8.75 inches (the bottom of the page).

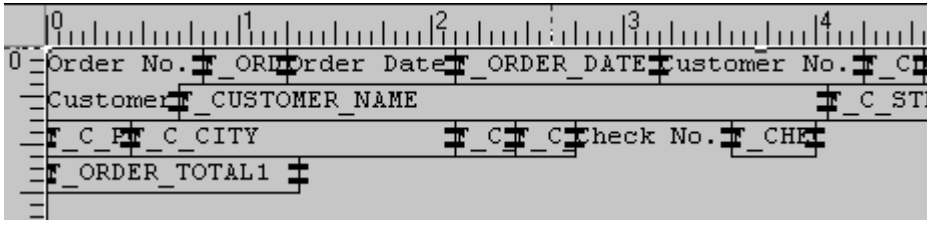
   > **Tip:** When you click and drag a resize handle in the Paper Layout view, guidelines display along the ruler to help you place your objects. In this case, drag the bottom center resize handle down the page until you the horizontal guideline reaches 8.75.

4. Click the Flex Off button in the toolbar to set Flex mode off.
5. In the Paper Layout view, click **F_ORDER_TOTAL**, then press CTRL+C on your keyboard.

6. Click in the area below your layout objects and press CTRL+V.

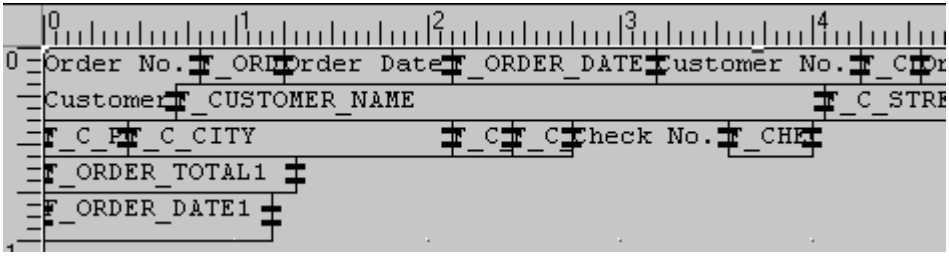
   You should see a new field called **F_ORDER_TOTAL1**. If this field displays on top of the other layout objects, click and drag it down below the other layout objects so that your Paper Layout view now looks something like this:

Figure 30–13 Partial Paper Layout view with F_ORDER_TOTAL1



7. Copy the **F_ORDER_DATE** field and paste it below the other fields, so that your layout now looks like this:

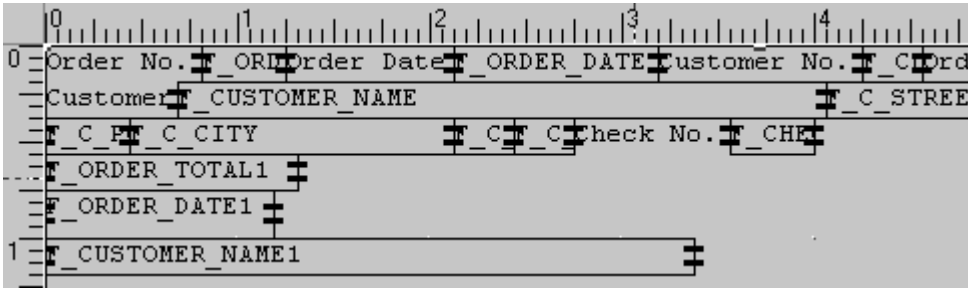
Figure 30–14 Partial Paper Layout view with F_ORDER_DATE1



Note: When you copy and paste a field, Reports Builder maintains the size of the field. So, when you copy and paste F_ORDER_DATE, you may not see the full name of the field. While the field is selected, you can click and drag the right border of the field to the right so that you can see the full name of the field, F_ORDER_DATE1.

8. Copy **F_CUSTOMER_NAME** and paste the field below the other layout objects, so that your layout looks like this:

Figure 30–15 Partial Paper Layout view with F_CUSTOMER_NAME1

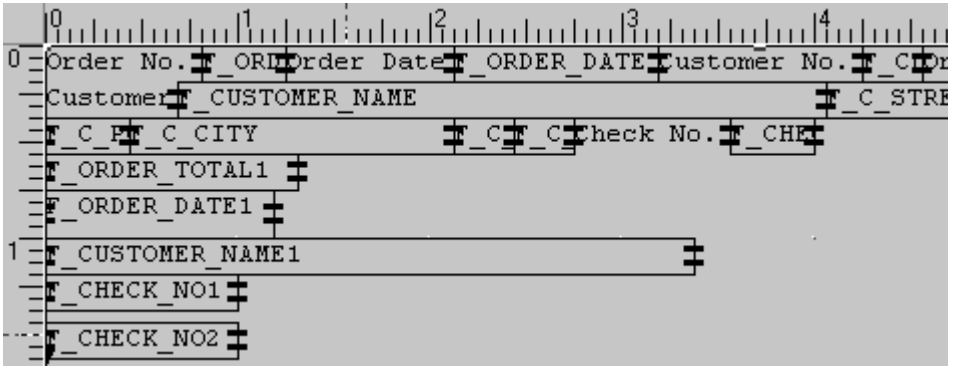


9. Make two copies of **F_CHECK_NO** by pressing CTRL+C on your keyboard once, then CTRL+V twice.

> **Note:** Since this field is small, it may be difficult to find. You can use the Object Navigator to find and select the field. When you click a field in the Object Navigator, under Paper Layout, the corresponding field is also selected in the Paper Layout view.

10. Position these two new fields below the other layout objects, so that your layout looks like this:

Figure 30–16 Partial Paper Layout view with F_CHECK_NO1 and F_CHECK_NO2

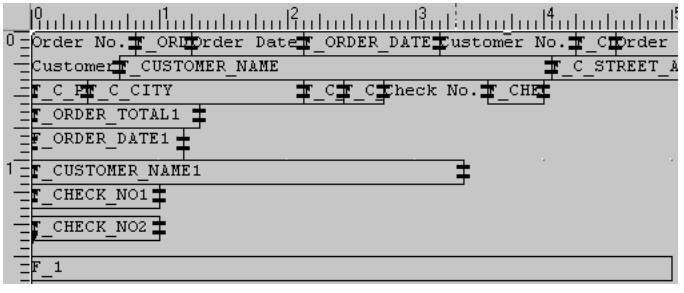


Note: We've expanded the size of the two new fields (F_CHECK_NO1 and F_CHECK_NO2) to make them easier to see, but you do not need to resize these fields:

11. Click the Field tool in the tool palette.

12. Draw a field below the other layout objects about 5 inches long, so that your layout now looks like this:

Figure 30–17 Partial Paper Layout view with F_1 field



13. While the new field, F_1, is still selected, click the Fill Color tool in the tool palette, and choose **No Fill**.
14. Click the Line Color tool in the tool palette, and choose **No Line**.
15. Click the field F_1, then press F4 on your keyboard to display the Property Inspector.
16. Set the Name property to F_SPELLED_AMOUNT.
17. Under **Field**, set the Source property to SPELLED_AMOUNT (the formula column you created in Section 30.3.2, "Create a formula column in your data model").
18. Save your report.

30.5.4 Rearrange the new fields according to the blank check image

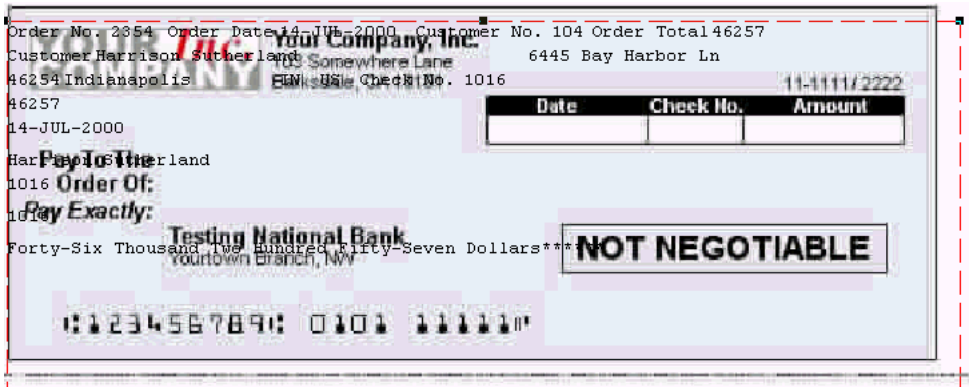
The steps in this section will show you how to use the Paper Design view to rearrange the fields you just created. You will use the blank check image we've provided as a guide.

1. Click the Paper Design button in the toolbar to display your report in the Paper Design view.

> **Note:** When you click the Paper Design button while you are in another view (i.e., Paper Layout view or Data Model view), Reports Builder runs your query and your layout. If you receive any error messages when you click the Paper Design button, go back to your original view to verify your changes. You can always compare your report against the sample report we've provided, called `spellcash.rdf`.

You will see your layout objects, as well as the blank check image:

Figure 30–18 Partial Paper Design view of the check report



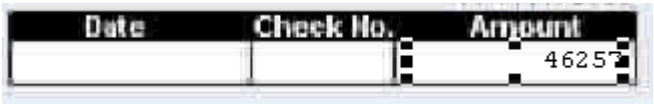
Note: You'll see that the preview doesn't quite look like a check report. The steps in this section will show you how to move your fields so that your report looks like a proper check.

2. In the Paper Design view, position the **F_ORDER_TOTAL** field in the "Amount" box in the blank check image.

Note: If you closed Reports Builder for any reason and are returning to building this report, make sure you click the Flex Off button in the toolbar so that you can move the fields around in the Paper Design view.

3. Click the Align Right button in the toolbar, so that this portion of the check looks like the following:

Figure 30–19 Amount area on blank check



4. Position the **F_CHECK_NO** field in the "Check No." box on the blank check, and center-align the text.
5. Position the **F_ORDER_DATE** field in the "Date" box on the blank check, and center-align the text.
6. Position the **F_CHECK_NO2** field in the upper right-hand corner of the check image, and right-align the text.
7. Position the **F_SPELLED_AMOUNT** field next to the "Pay Exactly:" text on the blank check image.
8. Position the **F_CUSTOMER_NAME** field next to the "Pay To The Order Of:" text on the blank check image.

   The check should now look like this:

Figure 30–20 Check image with fields positioned

9. Save your report.

30.5.5 Modify the look and feel of the check

The steps in this section will show you how to change the font, alignment, and formatting of the data on your check.

To modify the look and feel of the check:

1. In the Paper Design view, choose **Edit > Select All** (or press CTRL+A).
2. Choose **Format > Font**.
3. In the Font dialog box, choose **Arial** font, **Regular** style, Size **10**, then click **OK**.
4. In the Paper Design view, click anywhere to deselect all the objects.

   Tip: If you have trouble deselecting the objects, go to the Object Navigator and click on any item. The objects that were selected in the Object Navigator should display as deselected.

5. In the Paper Design view, click **F_CHECK_NO2** (the check number in the upper right-hand corner of the check) and change the font to Arial, Bold, 12 point.
6. Click the **F_ORDER_TOTAL** field (the number in the Amount box on the check).
7. Click the Currency button in the toolbar, then click the Commas button.
8. While **F_ORDER_TOTAL** is still selected, click the Add Decimal Place button in the toolbar twice, so that the amount now displays like this: `$46,257.00`.
9. Press F4 to display the Property Inspector for F_ORDER_TOTAL.
10. Under **Field**, next to the Format Mask property, add to the beginning of the existing format mask `"****"`, then close the Property Inspector.

    The amount should now display like this: `****$46,257.00`.
11. Select the **F_CUSTOMER_NAME** and **F_SPELLED_AMOUNT** fields by Shift-clicking them, then change the font to Times New Roman, 12 point.

    Your report should now look something like this:

Figure 30–21 Formatted Check

Note: You'll notice that the other boilerplate text is still overlaying the image. We will rearrange this text later in the chapter.

12. Save your report.

30.6 Create a check stub with payment information and order details

When you create a check printing report, you will sometimes want to create a check "stub" that contains the payment information on the check, as well as the details of the order. The steps in this section will show you how to create a check stub that contains the payee, amount, date, check number, and other payment information.

You will then use the information retrieved by the second query in your data model to print a simple tabular report on the check stub that displays the order information associated with the check.

30.6.1 Create a check stub in the Paper Design view

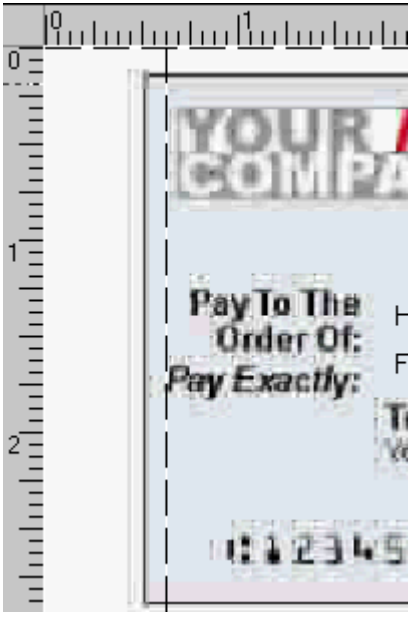
The steps in this section will show you how to use the tools in the Paper Design view to create a check stub for your check report. You will rearrange the fields you created in the previous section so that every time you run the report, a corresponding check stub will display for every check.

You will also create guidelines in the Paper Design view to help you align the check stub information with the data in the check report.

To create a check stub:

1. Click the Flex Off button in the toolbar.
2. Click in the left, vertical ruler and drag your mouse to the right, so that the line is flush with the text "Pay Exactly."
3. When you release your mouse button, you'll notice that a guideline displays, like the one in the following image:

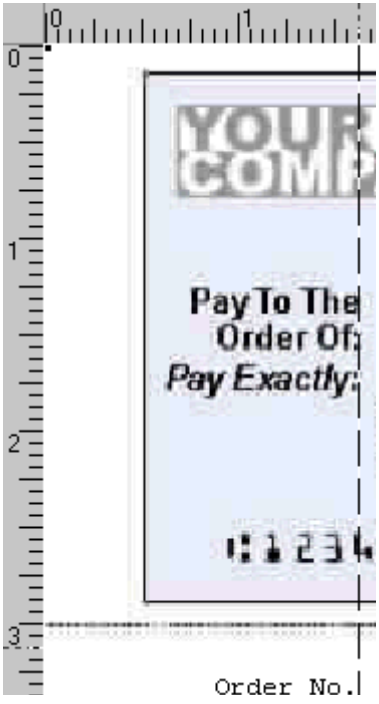
Figure 30–22 Partial Paper Design view of the check report with horizontal guideline



Note: The guideline is a broken black line spanning across this image, to the left of the "P" in "Pay Exactly."

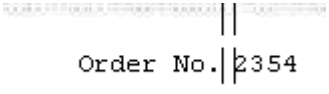
4. Click the **Order No.** boilerplate object (located in the upper left-hand corner of the check) and position it along the new guideline, and below the check image at 3.25 inches, like this:

Figure 30–23 Partial Paper Design view of the check with Order No. boilerplate text



5. Click the **F_ORDER_ID** field (in the above image, it is the field with "2354") and position it next to the Order No. field, like this:

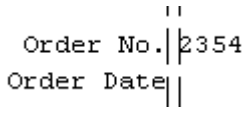
Figure 30–24 Order No. and F_ORDER_ID fields in the Paper Design view



Note: You can always select your fields in the Object Navigator, under **Paper Layout > Main Section > Body**, and these fields will appear selected in the Paper Design view.

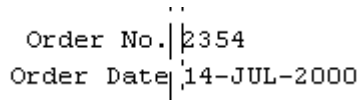
6. Now, click the **Order Date** boilerplate text and position it directly below "Order No.", like this:

Figure 30–25 Order Date field in the Paper Design view



7. Click **F_ORDER_DATE1** (the copy you made of the F_ORDER_DATE field), which you can locate using the Object Navigator, and position the field next to the Order Date boilerplate text, like this:

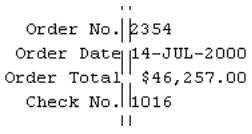
Figure 30–26 F_ORDER_DATE1 field next to Order Date text



8. Position the **Order Total** boilerplate text directly below the **Order Date** boilerplate text.
9. Position the **F_ORDER_TOTAL1** field next to the **Order Total** boilerplate text.
10. Position the **Check No.** boilerplate text directly below the **Order Total** boilerplate text.
11. Position **F_CHECK_NO1** next to the **Check No**. boilerplate text.

The check stub should now look like this:

Figure 30–27 Partial check stub in Paper Design view



12. Create another vertical guideline that is aligned with the box containing the data, amount, and check number. The guideline should appear here:

Figure 30–28 Second vertical guideline in the Paper Design view

Note: In this image, the second vertical guideline is at the 4.25 inch mark on the top ruler. To create this guideline, click in the left-hand ruler, then drag your mouse until the line is aligned with the 4.25 inch mark.

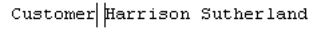
---

13. Click the **Customer** boilerplate text, and position it so that it is vertically aligned with the second guideline, and horizontally aligned with the **Order No**. text (at 3.25 inches), like this:

Figure 30–29 Customer field in the Paper Design view

14. Position the **F_CUSTOMER_NAME1** field next to the **Customer** boilerplate text so that these layout objects look like this:

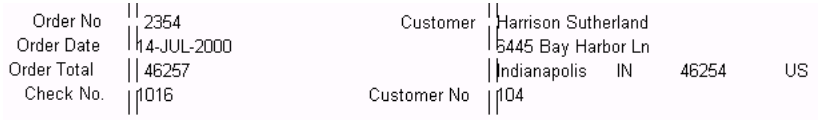
Figure 30–30 Customer and F_CUSTOMER_NAME1 fields



> **Tip:** If you have difficulty fitting the F_CUSTOMER_NAME1 field into the provided space, you may need to resize the field. To do so, select the field, then drag the right middle resize handle to the left, making sure that there is still enough space for the customer name (Harrison Sutherland).

15. Position the **F_C_STREET_ADDRESS** field directly below **F_CUSTOMER_NAME1**.
16. Position **F_C_CITY** directly below **F_C_STREET_ADDRESS**.
17. Position the **F_C_STATE_PROVINCE** field next to **F_C_CITY**.
18. Position the F**_C_POSTAL_CODE** field next to **F_C_STATE_PROVINCE**.
19. Position the **F_C_COUNTRY_ID** field next to **F_C_POSTAL_CODE**.
20. Position the **F_CUSTOMER_ID** field directly below the **F_C_CITY** field.
21. Position the **Customer No**. boilerplate text (**B_CUSTOMER_ID**) directly to the left of the **F_CUSTOMER_ID** field, so that this portion of the check stub should now look like this:

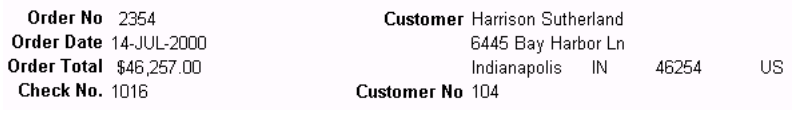
Figure 30–31 Partial check stub in the Paper Design view



22. Now, let's format the information you've just added. Select the following boilerplate text objects in the check stub:
    - Order No.
    - Order Date
    - Order Total
    - Check No.
    - Customer
    - Customer No.
23. Change the font to Bold, then right-align the text.
24. Click **F_ORDER_TOTAL1**, then click the Currency button and the Commas button in the toolbar.
25. Click the Add Decimals button in the toolbar twice.

    Your check stub should now look like this:

Figure 30–32 Formatted check stub



26. Save your report.

30.6.2 Add order details to the check stub

In the previous section, you created a check stub that displays the payment information, that is a copy of the information on the check itself. The steps in this section will show you how to add the details of the order to the check stub by creating a simple tabular report. You will use the Report Block Wizard to create the simple tabular report, then format the data in the Paper Design view.

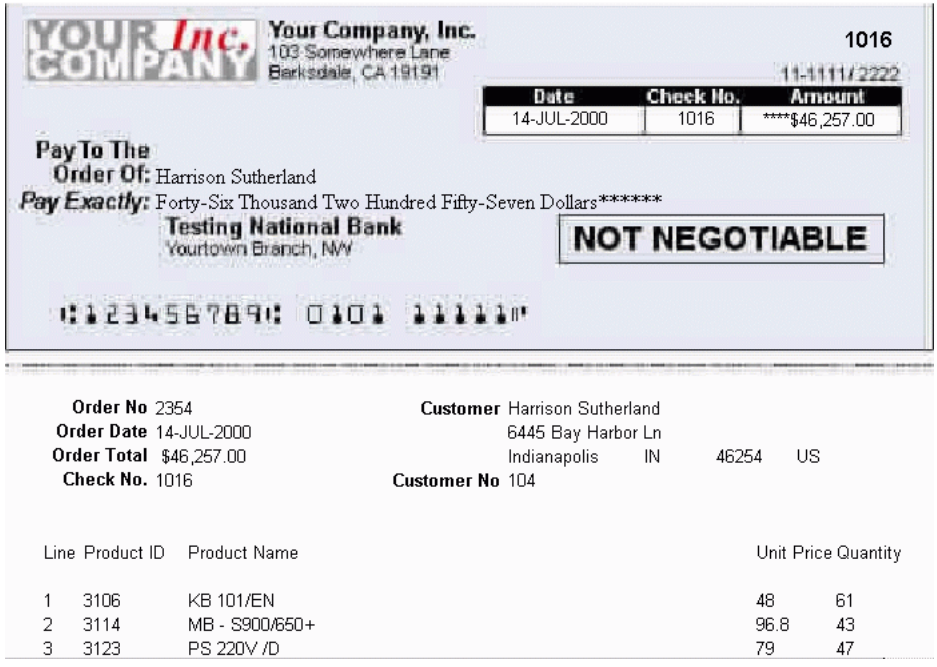
To add order details:

1. In the Paper Layout view, click the Report Block tool in the tool palette.
2. Drag a rectangular area for the new layout to display the Report Block Wizard.
3. In the Report Block Wizard, on the Style page, select **Tabular**, then click **Next**.
4. On the Groups page, click **G_LINE_ITEM_ID** and click **Down** in the **Available Groups** list to specify the Print Direction and move this group to the **Displayed Groups** list, then click **Next**.
5. On the Fields page, click each of the following fields in the **Available Fields** list, then click the right arrow (**>**) to move them to the **Displayed Fields** list, then click **Next**:
   - **LINE_ITEM_ID**
   - **PRODUCT_ID**
   - **PRODUCT_NAME**
   - **UNIT_PRICE**
   - **QUANTITY**
6. On the Labels page, change the labels and field widths as follows, then click **Next**:

Fields	Labels	Width
LINE_ITEM_ID	`Line`	2
PRODUCT_ID	`Product ID`	4
QUANTITY	Quantity	4
PRODUCT_NAME	Product Name	40
UNIT_PRICE	Unit Price	4

7. On the Template page, select **No template**, then click **Finish** to display your report layout in the Paper Layout view, with the new fields and layout objects displaying below the original layout objects.
8. In the Paper Layout view, make sure the new fields and layout objects are located within the **R_G_ORDER_ID** repeating frame, and that all the layout objects fit on a single page (within 8.75 inches).
9. Click the Paper Design button in the toolbar to run and display your report in the Paper Design view. It should look something like this:

Figure 30–33 Check printing report with non-formatted order details



Since the new report block is not formatted, let's format the data to look like the rest of the report.

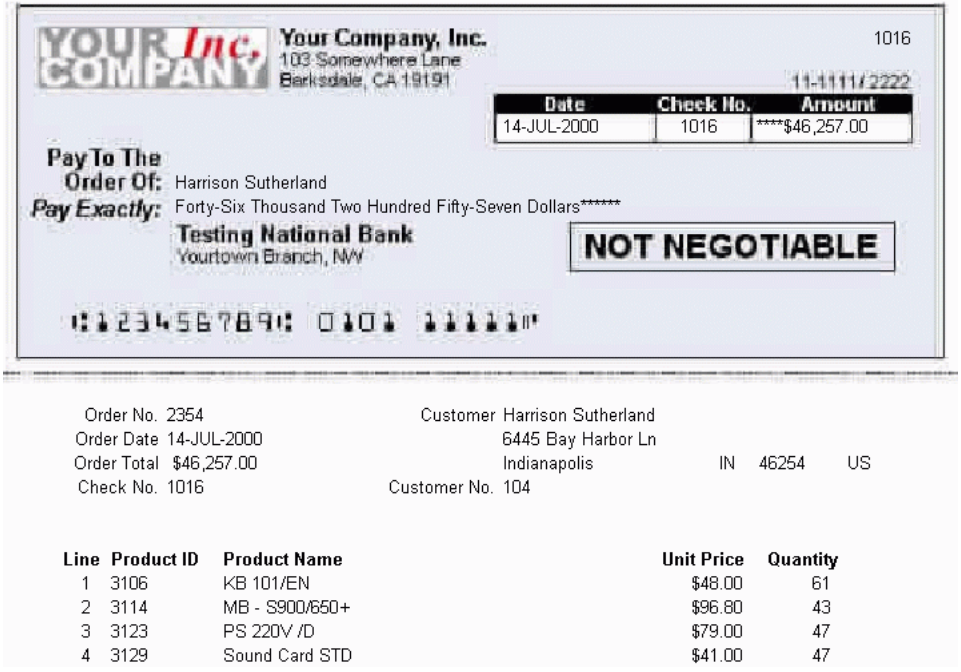
10. Shift-click the following new labels, then click the Bold button in the toolbar and change the font to **Arial**, **10 point**:

- **Line**
- **Product ID**
- **Product Name**
- **Unit Price**
- **Quantity**

11. Click the data under **Line** (F_LINE_ITEM_ID), then click the Align Center button in the toolbar.
12. Click the data under **Unit Price** (F_UNIT_PRICE), then click the Align Right button in the toolbar.
13. While the data is still selected, click the Currency button, Commas button, then click the Add Decimal button twice in the toolbar.
14. Click on the right resize handle and align the right edge of the data field with the **Unit Price** label.
15. Click the data under **Quantity** (F_QUANTITY), then click the Align Right button in the toolbar, then resize the field so that the right edge of the data field is aligned with the **Quantity** label.
16. Save your report.

Your report should now look like this:

Figure 30–34 Final check report with spelled-out cash amounts and stub



Note: You can also compare your results with the example report we've provided, called `spellcash.rdf`.

30.7 Summary

Congratulations! You have successfully built a report with spelled-out cash amounts on the check, as well as the payment and order details on the check stub. You now know how to:

- use the Report Wizard to create a simple report definition.
- use the tools in the Paper Layout view to import a blank check image and add fields to the report.
- create a PL/SQL function that returns spelled-out numbers.

- create a formula column based on a PL/SQL function.
- use the tools in the Data Model view to link two queries.
- use the tools in the Paper Design view to format your check printing report to print on a desired check.
- create a check stub that displays a duplicate of the information on the check.
- use the Report Block Wizard to create a simple tabular report that displays order details on a check stub.

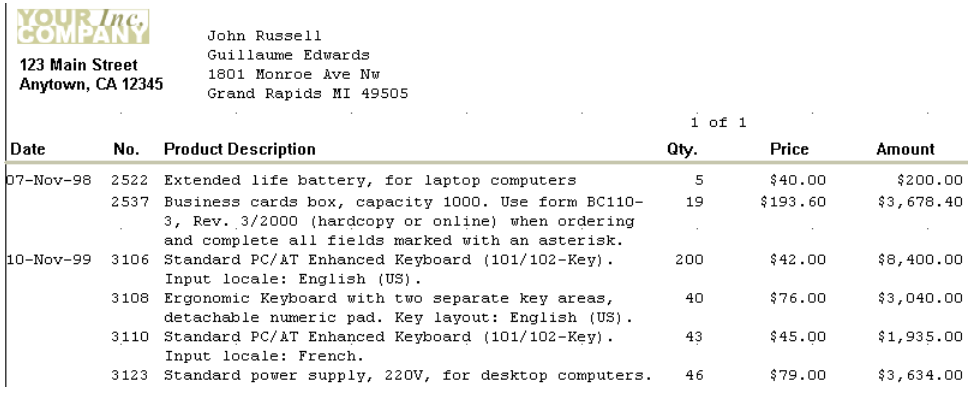
For more information on any of the wizards, views, or properties used in this example, refer to the *Reports Builder Online Help*, which you can access in two ways:

- From the Oracle Technology Network (`http://otn.oracle.com/products/reports/`), click **Documentation** and navigate to the *Reports Builder Online Help* for the most recent, hosted online help.
- From Reports Builder, choose **Help > Help Contents**.

31

Building a Report Using a Pre-Printed Form

Figure 31–1 Printing Reports on Pre-Printed Forms



YOUR Inc. COMPANY

123 Main Street
Anytown, CA 12345

John Russell
Guillaume Edwards
1801 Monroe Ave Nw
Grand Rapids MI 49505

1 of 1

Date	No.	Product Description	Qty.	Price	Amount
07-Nov-98	2522	Extended life battery, for laptop computers	5	$40.00	$200.00
	2537	Business cards box, capacity 1000. Use form BC110-3, Rev. 3/2000 (hardcopy or online) when ordering and complete all fields marked with an asterisk.	19	$193.60	$3,678.40
10-Nov-99	3106	Standard PC/AT Enhanced Keyboard (101/102-Key). Input locale: English (US).	200	$42.00	$8,400.00
	3108	Ergonomic Keyboard with two separate key areas, detachable numeric pad. Key layout: English (US).	40	$76.00	$3,040.00
	3110	Standard PC/AT Enhanced Keyboard (101/102-Key). Input locale: French.	43	$45.00	$1,935.00
	3123	Standard power supply, 220V, for desktop computers.	46	$79.00	$3,634.00

In this chapter, you will learn formatting techniques for printing reports on pre-printed forms when you do not have access to a computer readable version of the forms. Such reports must be designed so that the data prints in exact positions on the form.

The above image shows the example report you will create in this chapter, printed on a sample pre-printed form.

Note: Many of the same concepts are used in the following three example reports

- This example, where you learn formatting techniques for printing reports on pre-printed forms when you do not have access to a computer readable version of the forms. Such reports must be designed so that the data prints in exact positions on the form.
- Chapter 30, "Building a Check Printing Report with Spelled-Out Cash Amounts", where you import an image of a check and use it as a guide to position fields in the Paper Layout view. In addition, you learn how to create a PL/SQL function that returns spelled-out numerical values.
- Chapter 32, "Building an Invoice Report", where you import an image of an invoice and use it as a guide to position fields in the Paper Layout view.

Concepts

- This chapter explains how to use the various tools in the Paper Layout view and the Paper Design view to format a report for printing on pre-printed forms.

Data Relationships

- This report uses two queries linked in a master/detail relationship. You will create a break in the detail group and two summary columns in the master group.

Layout

- This report consists of two partial default layouts, one with a mailing label format and one with a tabular format. You'll move fields to the correct positions on the page and restrict the master repeating frame to appearing once per form.

Example Scenario

This report consists of data formatted such that all values fall precisely within the corresponding fields of a pre-printed sales order.

In our example, the form is on a standard sized 8.5 inches by 11 inches page. The employee name and the customer's name and address must display in the region at the top of the page. The order item details must only display 40 characters.

Anything over 40 characters will be placed on a second page of the form. At the end of our example, we add page numbers. This section is optional, as your forms may not require page numbers.

> **Note:** The above measurements are for the form we use in this example. If you do not have access to an online, or computer readable version of your pre-printed form, you will need to take these measurements yourself, then use the steps below to create your own report definition.

To see a sample report that uses pre-printed forms, open the examples folder named `preprint`, then open the Oracle Reports example named `preprint.rdf`. For details on how to open it, see "Accessing the example reports" in the Preface.

Table 31–1 Features demonstrated in this example

Feature	Location
Use the tools in the Data Model view to manually create two queries. You will also link the two queries, create a break group, and create two summary columns	Section 31.2, "Manually create the data model for your report"
Use the tools in the Paper Layout view to create the layout for your report. You will also use the Report Block Wizard to create a mailing label portion of your report and a tabular report that displays the order detail information.	Section 31.3, "Create the layout for your report"
Use the tools in the Paper Design view to add format masks and adjust the appearance of your report.	Section 31.4, "Format your report in the Paper Design view"
Use the tools in the Paper Layout view to add page numbering fields to your report.	Section 31.5, "Add page numbers (optional)"

31.1 Prerequisites for this example

To build the example in this chapter, you must have access to the Order Entry portion of the sample schema, which is provided by default with the Oracle9*i* database. Contact your database administrator for access to this schema.

31.2 Manually create the data model for your report

The steps in this section will show you how to manually create the data model for your report. Your data model will consist of two linked queries. The first query will retrieve all the customers that have orders in the database. The second query will retrieve all of the order information for each of the customers, and will contain a break group based on the date of each order.

31.2.1 Create the queries

The steps in this section will show you how to manually create two queries in the Data Model view.

To create the queries:

1. Launch Reports Builder (or, if already open, choose **File > New > Report**)
2. In the Welcome or New Report dialog box, select **Build a new report manually**, then click **OK**.
3. In the Data Model view that displays, click the SQL Query tool in the tool palette.
4. Click in an open area of the Data Model view to display the SQL Query Statement dialog box.
5. In the **SQL Query Statement** field, enter the following SELECT statement:

```
SELECT FIRST_NAME||' '||LAST_NAME EMPLOYEE_NAME
, CUST_FIRST_NAME||' '||CUST_LAST_NAME CUSTOMER_NAME
, CUST_ADDRESS
, CUSTOMER_ID
FROM EMPLOYEES, CUSTOMERS
WHERE EMPLOYEE_ID = ACCOUNT_MGR_ID
AND CUSTOMER_ID IN (SELECT UNIQUE(CUSTOMER_ID) FROM ORDERS)
ORDER BY LAST_NAME, CUST_LAST_NAME
```

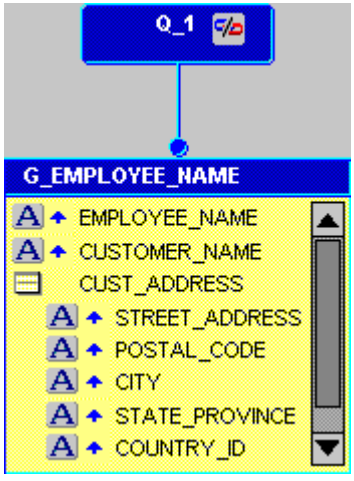
Note: You can enter this query in any of the following ways:

- Copy and paste the code from the provided text file called `preprint_code.txt` into the SQL Query Statement field.
- Click **Query Builder** to build the query without entering any code manually.
- Type the code in the SQL Query Statement field.

6. Click **OK**.
7. If the **Connect** dialog box displays, type the connection information for the database where the **Order Entry** portion of the sample schema resides, then click **Connect**.

   The query displays in the Data Model view, and should look something like this:

Figure 31–2 Data Model view of the first query



8. Right-click the query (**Q_1**), then choose **Property Inspector** from the pop-up menu.
9. In the Property Inspector, under **General Information**, set the Name property to Q_CUSTOMER.
10. Double-click the group (**G_EMPLOYEE_NAME**) to display the Property Inspector, and set the Name property to G_CUSTOMER.
11. In the Data Model view, click the SQL Query tool in the tool palette to create a second query with the following SELECT statement:

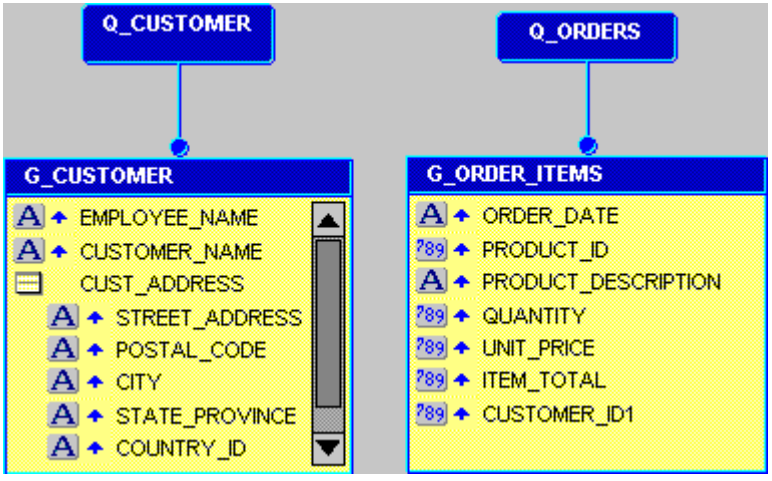
```
SELECT to_char(ORDER_DATE, 'DD-Mon-YY') ORDER_DATE
, I.PRODUCT_ID
, PRODUCT_DESCRIPTION
```

```
, QUANTITY
, UNIT_PRICE
, QUANTITY * UNIT_PRICE ITEM_TOTAL
, CUSTOMER_ID
FROM ORDERS O, ORDER_ITEMS I, PRODUCTS P
WHERE O.ORDER_ID = I.ORDER_ID
AND I.PRODUCT_ID = P.PRODUCT_ID
ORDER BY ORDER_DATE, I.PRODUCT_ID
```

12. Right-click the query object, then choose **Property Inspector** from the pop-up menu.
13. In the Property Inspector, set the Name property to Q_ORDERS.
14. Double-click the new group object to display the Property Inspector.
15. In the Property Inspector, set the Name property to `G_ORDER_ITEMS`.

    Your data model should now look like this:

Figure 31–3 Data Model view with two queries



Note: In Q_CUSTOMER, the second WHERE clause restricts the values returned to only those customers who have current orders. In Q_ORDERS, we've used "to_char" to account for a new datatype, "Datetime." Since this new datatype is not recognized by Oracle Reports, you cannot use format masks on this particular column yet.

16. Save your report as `preprint_<your initials>.rdf`.

31.2.2 Modify your data model

The steps in this section will show you how to modify your data model to link the two queries and to cause the data retrieved to break on a particular column.You will also create several summary columns to calculate information for each order.

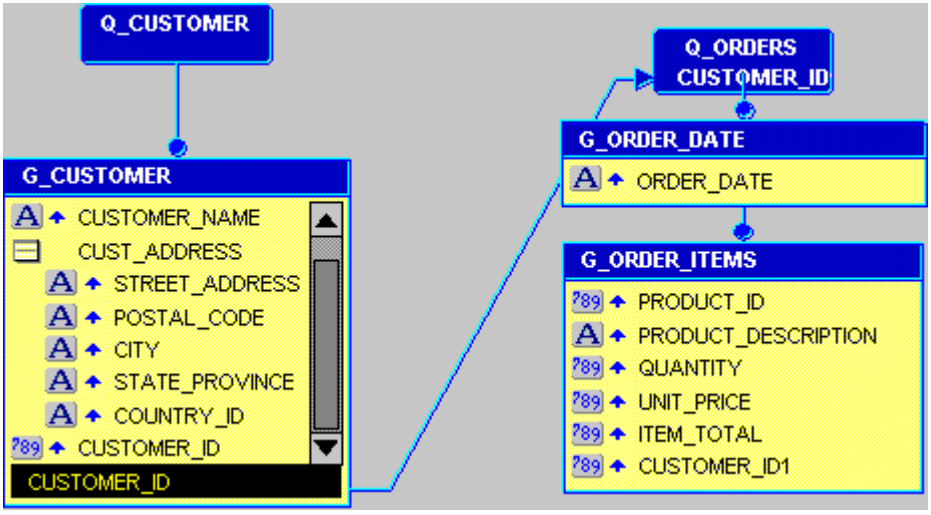
To modify your data model:

1. In the Data Model view, click the Data Link tool in the tool palette.
2. Click the **CUSTOMER_ID** column in the **G_Customer** group.
3. While holding down your mouse button, drag your cursor to the **CUSTOMER_ID1** column in the **G_ORDER_ITEMS** group and release your mouse button to create a link between the two queries.
4. Now, create a break group by clicking the **ORDER_DATE** column in the **G_ORDER_ITEMS** group and dragging it above the group.

   Tip: You can select the group or the query title and use the resize handles to modify the look of your data model.

   Your data model should now look like this:

Figure 31–4 Data Model view of the linked queries and break group

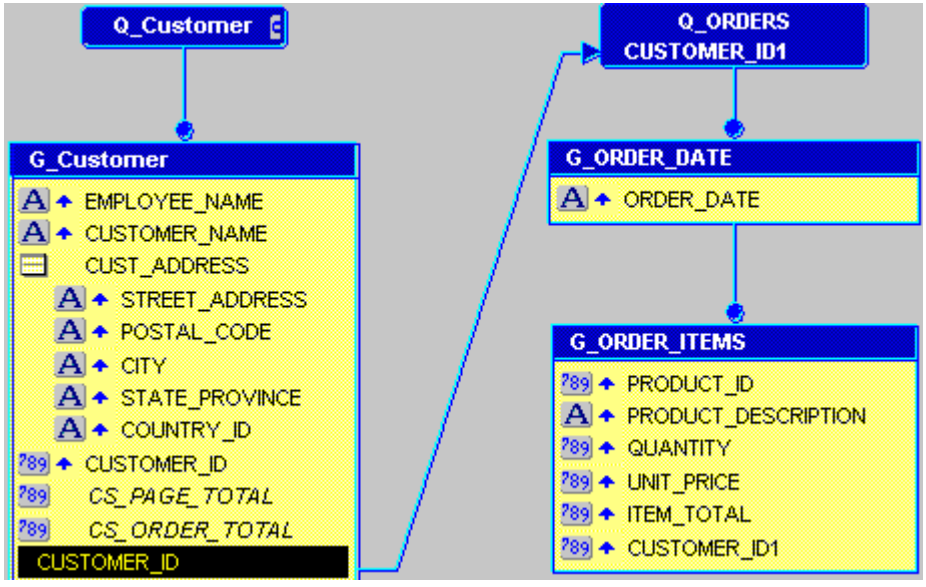


5. Click the Summary Column tool in the tool palette, then click in the **G_CUSTOMER** group to create a summary column. You may want to resize the group so that you can see your summary columns.
6. Double-click the new summary column object (**CS_1**) to display the Property Inspector, and set properties:
   - Under **General Information**, set the Name property to CS_PAGE_TOTAL.
   - Under **Summary**, set the Function property to Sum, set the Source property to ITEM_TOTAL, and set the Reset At property to Page.
7. Create another summary column in the G_CUSTOMER group, and set its properties as follows:
   - Under **General Information**, set the Name property to CS_ORDER_TOTAL.
   - Under **Summary**, set the Function property to Sum, set the Source property to ITEM_TOTAL, and set the Reset At property to G_CUSTOMER.

Note: These two summary columns calculate the total items per page and the total items in the order for the report.

Your data model is now complete, and should look like this:

Figure 31–5 Complete Data Model view of the pre-printed forms example



8. Save your report.

31.3 Create the layout for your report

The steps in this section will show you how to create a layout that displays the objects in the desired locations on your pre-printed form. You will start by adjusting the margin of your report, then creating the mailing label portion that will print at the top of each page that shows the address information for the customer. You will then create the body of the report, which contains the order information that display the date, order number, description, number, price per item, and total price.

31.3.1 Set up your report layout

Before you can start adding layout objects, you will need to adjust the margin of your layout.

To set up the layout:

1. Click the Paper Layout button in the toolbar to display the Paper Layout view of your report.
2. To make it easier for you to manipulate your layout objects, expand the size of the Paper Layout window.
3. Click the Edit Margin button in the toolbar to view the margin portion of the layout.
4. Using the resize handles, adjust the size of the margin (the solid black bar) to the dimensions of your form. In our example, our form is 8.5 inches by 11 inches, so we will expand the margin to 8.5 inches across, and 11 inches down.

   Tip: You can use the corner resize handles to adjust the size of your margin.

5. Click the Edit Margin button in the toolbar again to return to the body of the layout.

   Now, you are ready to create your report layout.

31.3.2 Create an address label for your form

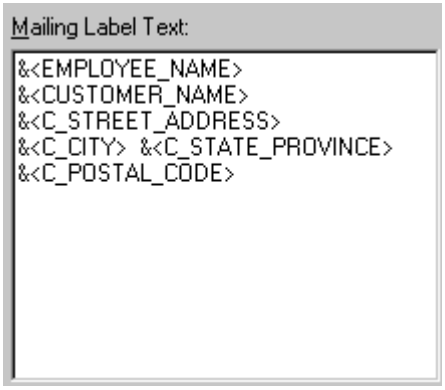
In this section, you will use the Report Block Wizard to create a mailing label for the address section of your pre-printed form.

To create an address label:

1. In the Paper Layout view, click the Report Block tool in the tool palette.
2. Using the rulers along the edges of the Paper Layout view, drag a rectangle starting at 1 inch from the left and 0.25 inches from the top (1, 0.25) to 6 inches from the left and 1.25 inches from the top (6, 1.25). Release your mouse button to display the Report Block Wizard.
3. On the Style page of the Report Block Wizard, select **Mailing Label**, then click **Next**.

4. On the Groups page, click **G_CUSTOMER** and click **Down** in the **Available Groups** list to specify the Print Direction and move this group to the **Displayed Groups** list, then click **Next**.
5. In the **Mailing Label Text** field, type the columns (or click the columns in the **Available Fields** list) so that the field looks like this:

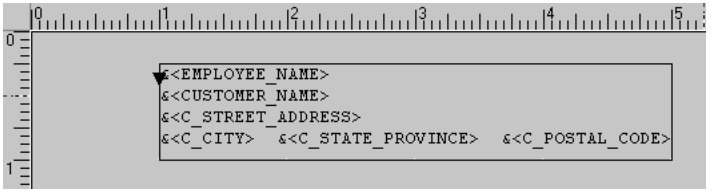
Figure 31–6 Mailing label text field



Note: For more information on building a mailing label report, see Chapter 6, "Building a Mailing Label Report".

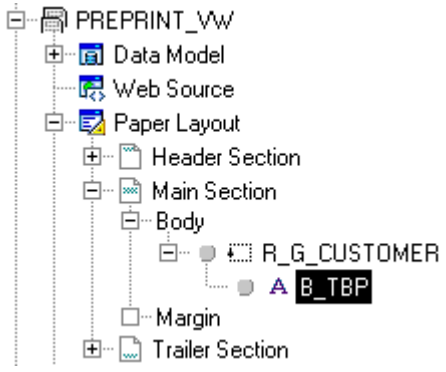
6. Once you've verified your text against the above, click **Next**.
7. On the Template page, select **No template**, then click **Finish** to display your report layout in the Paper Layout view. It should look something like this:

Figure 31–7 Paper Layout view of the initial layout



8. The region that you just created is a boilerplate object called B_TBP. While this object is selected, view the Object Navigator and notice the new boilerplate object:

Figure 31–8 Report Block boilerplate object in the Object Navigator



9. While the object is selected, press F4 on your keyboard to display the Property Inspector (or right-click the object and choose **Property Inspector**).
10. In the Property Inspector, under **General Layout**, set the Horizontal Elasticity property to Variable. This ensures that appropriate space is available for the mailing label text.
11. Save your report.

31.3.3 Add the order item details to your report

In this section, you will add a report block that displays the order items and details that correspond to the customer identified in each mailing label. These items and details are required for your pre-printed form.

To add order item details to your report:

1. In the Paper Layout view, click the repeating frame that surrounds the boilerplate object (B_TBP) you created in Section 31.3.2, "Create an address label for your form".
2. Use the corner resize handles to adjust the size of the repeating frame to the size of your pre-printed form (in our example, 8.5 inches across and 11 inches down). Make sure the frame does not extend beyond the single page boundary.
3. Click the Report Block tool in the tool palette.
4. Using the rulers as guides, drag a region 0.25 inches from the left and 1.5 inches from the top (0.25, 1.5) to 8.25 inches from left and 2 inches from the top (8.25, 10). Release your mouse button to display the Report Block Wizard.
5. On the Style page of the Report Block Wizard, select **Group Left**, then click **Next**.
6. On the Groups page, click the following groups and click **Down** in the **Available Groups** list to specify the Print Direction and move them to the **Displayed Groups** list, then click **Next**:
   - **G_ORDER_DATE**
   - **G_ORDER_ITEMS**
7. On the Fields page, click the following fields in the **Available Fields** list and click the right arrow (>) to move them to the **Displayed Fields** list, then click **Next**:
   - **ORDER_DATE**
   - **PRODUCT_ID**
   - **PRODUCT_DESCRIPTION**
   - **QUANTITY**
   - **UNIT_PRICE**
   - **ITEM_TOTAL**

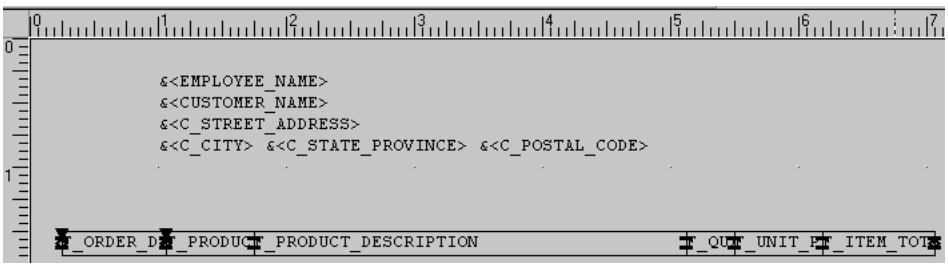
8. On the Labels page, remove the label names since they are not required for the pre-printed form, and adjust the widths as follows, then click **Next**:

Fields and Totals	Labels	Width
ORDER_DATE	<none>	9
PRODUCT_ID	<none>	8
PRODUCT_ DESCRIPTION	<none>	40
QUANTITY	<none>	4
UNIT_PRICE	<none>	8
ITEM_TOTAL	<none>	10

> **Note:** The Width values are based on the space available on the pre-printed form.

9. On the Template page, select **No Template**, then click **Finish** to display your report layout in the Paper Layout view. It should look something like this:

Figure 31–9 Paper Layout view of the pre-printed form example with mailing label and order item details



10. Save your report.

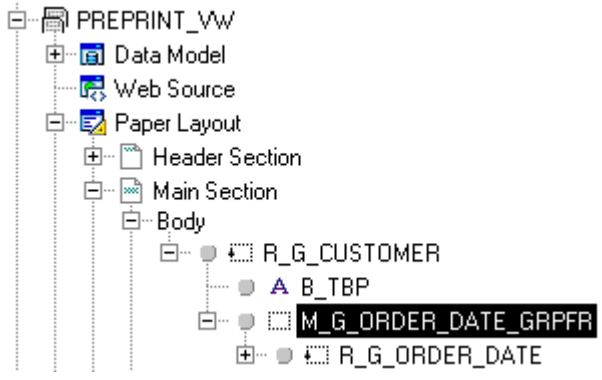
31.3.4 Adjust the layout and add summaries

Now that you've added the elements you want displayed on your report, you need to set up the layout so that only records required for the pre-printed form display on each page, and that the correct amount of information displays in the available amount of space on the form. Also, the form requires that a summary of each page displays in the same location on the page.

To adjust the layout and add page summaries:

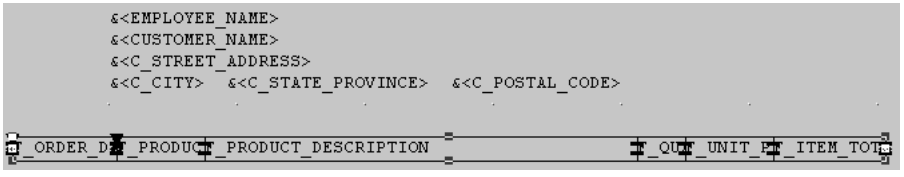
1. In the Object Navigator, click **M_G_ORDER_ITEMS_GRPFR**, as shown in the following image:

Figure 31–10 Selected repeating frame in the Object Navigator



2. In the Paper Layout view, you should see the same object selected, as shown in the following image:

Figure 31–11 Selected repeating frame in the Paper Layout view



3. In the Object Navigator, double-click the properties icon next to **M_G_ORDER_ITEMS_GRPFR** to display the Property Inspector, and set properties:
   - Under **General Layout**, set the Vertical Elasticity property to Fixed to ensure that only this area is available for the order items to display.
4. In the Object Navigator, double-click the properties icon next to **R_G_CUSTOMER** to display the Property Inspector, and set properties:
   - Under **Repeating Frame**, set the Maximum Records per Page property to 1, to ensure that only one customer's information prints per page.
5. In the Object Navigator, double-click the view icon next to the **Paper Layout** node to display the Paper Layout view.
6. In the Paper Layout view, click the Field tool in the tool palette.
7. Draw a region 7 inches from the left and 10.25 inches from the top (7, 10.25) to 8 inches from the left and 10.5 inches from the top (8, 10.5).

   Note: This field, as well as the second field you will create, should *not* be located within the repeating frame called M_G_ORDER_DATE_GRPFR. If you cannot fit these fields on the page, resize the repeating frame so that you can place the fields below the repeating frame, but still on the same page.

8. Double-click the new field object (**F_1**) to display the Property Inspector, and set properties:
   - Under **General Information**, set the Name property to F_PAGE_TOTAL.
   - Under **Field**, set the Source property to CS_PAGE_TOTAL (the summary column you created in Section 31.2, "Manually create the data model for your report").
9. Following the steps above, create another field directly below F_PAGE_TOTAL in the Paper Layout view, then set its Name property to F_ORDER_TOTAL and Source property to CS_ORDER_TOTAL.
10. The new layout objects should now look like this:

Figure 31–12 F_PAGE_TOTAL and F_ORDER_TOTAL fields in the Paper Layout view

11. Save your report.

31.4 Format your report in the Paper Design view

The steps in this section will show you how to preview your report in the Paper Design view, and make a few last-minute tweaks using the tools in the Paper Design view. It's sometimes easier to use the Paper Design view to finish your report, since you can see actual data displayed.

To format your report in the Paper Design view:

1. Click the Paper Design button in the toolbar to display the Paper Design view.

2. In the Paper Design view, click the field **F_PRODUCT_DESCRIPTION**.

> **Tip:** If you cannot find this field, you can use the Object Navigator to find the field name and click it. When you select an item in the Object Navigator, the corresponding object is selected in the Paper Design view.

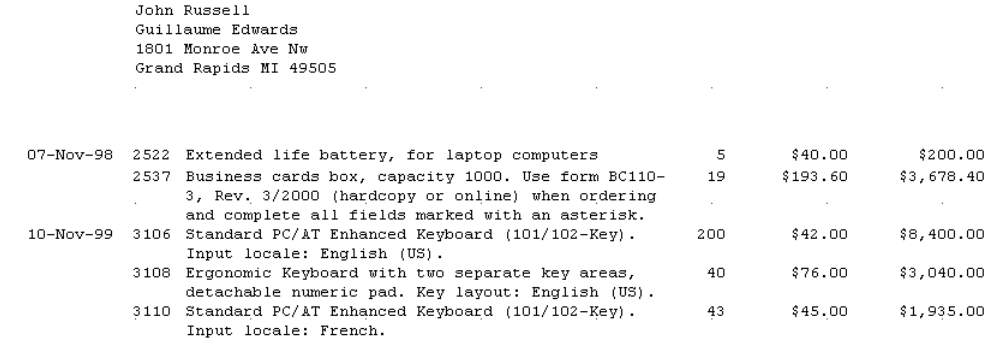
3. Increase the size of this field as much as possible, given the size of the area on your form.

4. Shift-click to select the four numbered fields, then click the Align Right button in the toolbar.

5. At the bottom of the page, Shift-click the two fields (**F_PAGE_TOTAL** and **F_ORDER_TOTAL**), then click the Line Color tool in the tool palette and choose **No Line**.
6. Shift-click the three currency number fields on the right.
7. Click the Currency button in the toolbar to add a dollar sign to the amounts.
8. Click the Commas button to add a comma to values over 999.
9. Click the Add Decimal Place button to add a decimal to the values.
10. In the Object Navigator, Shift-click the following three objects:
    - **B_TBP**
    - **F_ORDER_TOTAL**
    - **F_PAGE_TOTAL**
11. Press F4 on your keyboard to display the Property Inspector, and set properties:
    - Under **Advanced Layout**, set the Print Object On property to All Pages, and set the Base Printing On property to Enclosing Object.
12. In the Paper Design view, adjust the sizes of the fields according to the measurements on your pre-printed form.

    Your report should now look like this:

Figure 31–13 Final preview of your pre-printed form report



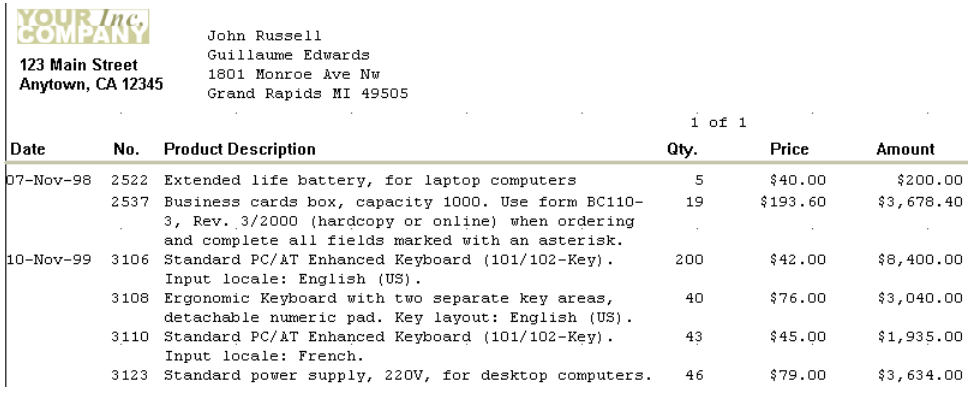
```
John Russell
Guillaume Edwards
1801 Monroe Ave Nw
Grand Rapids MI 49505

07-Nov-98  2522  Extended life battery, for laptop computers            5      $40.00      $200.00
           2537  Business cards box, capacity 1000. Use form BC110-     19     $193.60     $3,678.40
                 3, Rev. 3/2000 (hardcopy or online) when ordering
                 and complete all fields marked with an asterisk.
10-Nov-99  3106  Standard PC/AT Enhanced Keyboard (101/102-Key).       200     $42.00      $8,400.00
                 Input locale: English (US).
           3108  Ergonomic Keyboard with two separate key areas,       40     $76.00      $3,040.00
                 detachable numeric pad. Key layout: English (US).
           3110  Standard PC/AT Enhanced Keyboard (101/102-Key).       43     $45.00      $1,935.00
                 Input locale: French.
```

Note: To see a more complete view of the report, you can open the sample report we've provided, called `preprint.rdf`. To view the report, open the report and click the Run Paper Layout button in the toolbar.

13. When you print your report on to the form, it might look something like this:

Figure 31–14 Final report printed on a sample pre-printed form



YOUR Inc. COMPANY

123 Main Street
Anytown, CA 12345

John Russell
Guillaume Edwards
1801 Monroe Ave Nw
Grand Rapids MI 49505

1 of 1

Date	No.	Product Description	Qty.	Price	Amount
07-Nov-98	2522	Extended life battery, for laptop computers	5	$40.00	$200.00
	2537	Business cards box, capacity 1000. Use form BC110-3, Rev. 3/2000 (hardcopy or online) when ordering and complete all fields marked with an asterisk.	19	$193.60	$3,678.40
10-Nov-99	3106	Standard PC/AT Enhanced Keyboard (101/102-Key). Input locale: English (US).	200	$42.00	$8,400.00
	3108	Ergonomic Keyboard with two separate key areas, detachable numeric pad. Key layout: English (US).	40	$76.00	$3,040.00
	3110	Standard PC/AT Enhanced Keyboard (101/102-Key). Input locale: French.	43	$45.00	$1,935.00
	3123	Standard power supply, 220V, for desktop computers.	46	$79.00	$3,634.00

Note: The Paper Design view of your report will not look like the image above. We printed our report on a sample pre-printed form. The above image is a snapshot of that form.

14. Save your report.

31.5 Add page numbers (optional)

Some pre-printed forms may require you to print page numbers. The steps in this section will show you how to add page numbers to your output that will display when the number of order items extends beyond a single page.

To add page numbers to your report:

1. Click the Paper Layout button in the toolbar to display the Paper Layout view.
2. Click the Field tool in the tool palette.
3. Draw a small region on your layout 6 inches from the left and 0.25 inches from the top (6, 0.25) to 7 inches from the left and 0.5 inches from the top (7, 0.5).
4. Click the new field object, F_1, then press F4 on your keyboard to display the Property Inspector, and set properties:
   - Under **General Information**, set the Name property to F_PAGE_NUMBER.
   - Under **General Layout**, set the Horizontal Elasticity property to Variable.
   - Under **Field**, set the Source property to Page Number, and set the Visible property to No. Click the Page Numbering property field to display the Page Numbering dialog box.
5. In the Page Numbering dialog box:
   - Clear the **Header Section** and **Trailer Section** check boxes, and make sure that the **Main Section** check box is selected.
   - In the **Reset At** list, click **R_G_Customer**.
   - Click **OK**.
6. In the Paper Layout view, following the steps above to create another field from (7.25, 0.25) to (8.25, 0.5).
7. Double-click the new field object to display the Property Inspector, and set properties:
   - Under **General Information**, set the Name property to F_TOTAL_PAGES.
   - Under **General Layout**, set the Horizontal Elasticity property to Variable.
   - Under **Field**, set the Source property to Total Pages, and set the Visible property to No. Click the Page Numbering property field to display the Page Numbering dialog box.
8. In the Page Numbering dialog box:
   - Clear the **Header Section** and **Trailer Section** check boxes, and make sure that the **Main Section** check box is selected.
   - In the **Reset At** list, click **R_G_Customer**.
   - Click **OK**.

Now that you've created the page number fields, you need to make them visible.

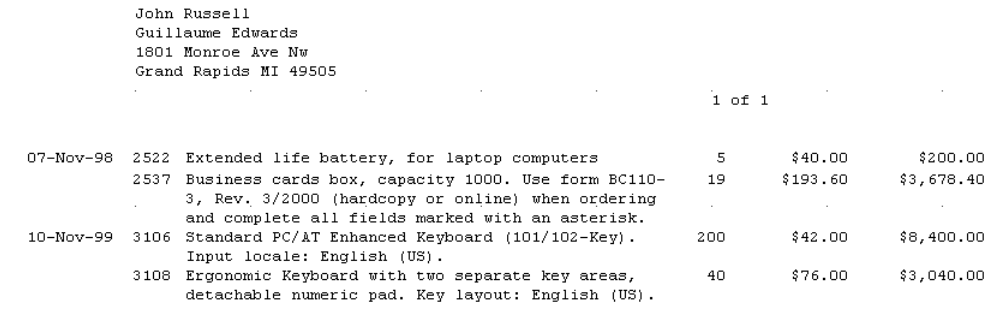
9. In the Paper Layout view, click the Text tool in the tool palette.
10. Draw a region on your layout 6 inches from the left and 1 inch from the top (6, 1) to 8 inches from the left and 1.25 inches from the top (8, 1.25).
11. Click inside the rectangle and type "`&<F_PAGE_NUMBER> of &<F_TOTAL_PAGES>`".

> **Note:** The "&<F_PAGE_NUMBER>" text will replace that text with the current value of the summary field you just created, which determines the page number of the current page. The "&<F_TOTAL_PAGES>" text will replace that text with the current total number of pages of the report, based on the value of the summary field of the same name.

12. With the text object selected, press F4 on your keyboard to display the Property Inspector, and set properties:
    - Under **General Information**, set the Name property to B_PAGE_NUMBERS.
    - Under **Advanced Layout**, set the Print Object On property to All Pages, and set the Base Printing On property to Enclosing Object.
13. In the Paper Design view, click the Run Paper Layout button in the toolbar to run your report and display it in the Paper Design view.

    Your report should now look like the following image. Notice how Reports Builder calculates that this particular report is one page, thus displays "1 of 1" above the order details.

Figure 31–15 Paper Design view of the pre-printed form report with page numbers



14. Save your report.

31.6 Summary

Congratulations! You have successfully built a report that you can print on a pre-printed form. You now know how to:

- manually create two queries and link them.
- create a summary column.
- create a break group in your data model.
- use the tools in the Paper Layout view to create your layout.
- use the Report Block Wizard to create a mailing label report.
- use the Report Block Wizard to create a tabular report.
- add page numbering using fields in the Paper Layout view.

For more information on any of the wizards, views, or properties used in this example, refer to the *Reports Builder Online Help*, which you can access in two ways:

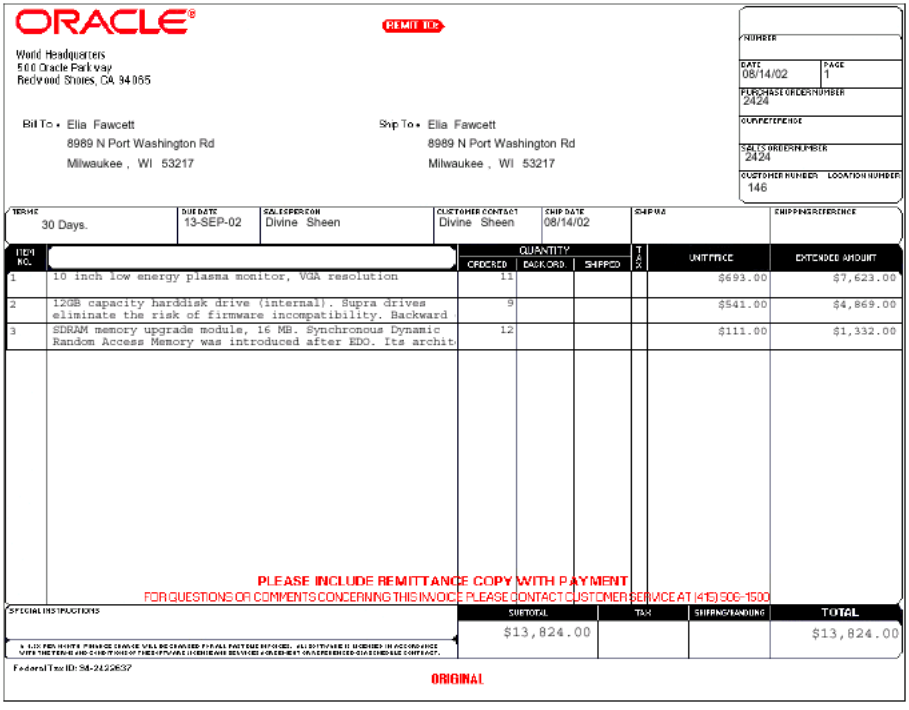
- From the Oracle Technology Network (`http://otn.oracle.com/products/reports/`), click **Documentation** and navigate to the *Reports Builder Online Help* for the most recent, hosted online help.
- From Reports Builder, choose **Help > Help Contents**.

Note: You can replace your Reports Builder online help with the most recent update by downloading the latest online help set available on the Oracle Technology Network (`http://otn.oracle.com/docs/products/reports/`). Instructions for replacing your help file are included in the readme.txt in the download file.

32

Building an Invoice Report

Figure 32–1 Invoice report output



ORACLE®

REMIT TO:

World Headquarters
500 Oracle Parkway
Redwood Shores, CA 94065

NUMBER

DATE 08/14/02 | PAGE 1

PURCHASE ORDER NUMBER 2424

OUR REFERENCE

SALES ORDER NUMBER 2424

CUSTOMER NUMBER | LOCATION NUMBER 146

Bill To: Elia Fawcett
8989 N Port Washington Rd
Milwaukee , WI 53217

Ship To: Elia Fawcett
8989 N Port Washington Rd
Milwaukee , WI 53217

TERMS	DUE DATE	SALESPERSON	CUSTOMER CONTACT	SHIP DATE	SHIP VIA	SHIPPING REFERENCE
30 Days.	13-SEP-02	Divine Sheen	Divine Sheen	08/14/02		

ITEM NO.		QUANTITY ORDERED	QUANTITY BACK ORD	QUANTITY SHIPPED	TAX	UNIT PRICE	EXTENDED AMOUNT
1	10 inch low energy plasma monitor, VGA resolution	11				$693.00	$7,623.00
2	120GB capacity harddisk drive (internal). Supra drives eliminate the risk of firmware incompatibility. Backward	9				$541.00	$4,869.00
3	SDRAM memory upgrade module, 16 MB. Synchronous Dynamic Random Access Memory was introduced after EDO. Its archit	12				$111.00	$1,332.00

PLEASE INCLUDE REMITTANCE COPY WITH PAYMENT
FOR QUESTIONS OR COMMENTS CONCERNING THIS INVOICE PLEASE CONTACT CUSTOMER SERVICE AT (415) 506-1500

SPECIAL INSTRUCTIONS	SUBTOTAL	TAX	SHIPPING/HANDLING	TOTAL
	$13,824.00			$13,824.00

Federal Tax ID: 94-2229837

ORIGINAL

This report displays several distinguishing characteristics of a typical invoice, such as customer name and address, sales order number, billing information, and billing totals.

In this chapter, you will build an invoice report where you will learn how to import an image of a pre-printed form (in this case, a blank invoice image) and use the tools in the Paper Layout view to print your report on such a form.

Note: Many of the same concepts are used in the following three example reports

- This example, where you import an image of an invoice and use it as a guide to position fields in the Paper Layout view.
- Chapter 30, "Building a Check Printing Report with Spelled-Out Cash Amounts", where you import an image of a check and use it as a guide to position fields in the Paper Layout view. In addition, you learn how to create a PL/SQL function that returns spelled-out numerical values
- Chapter 31, "Building a Report Using a Pre-Printed Form", where you learn formatting techniques for printing reports on pre-printed forms when you do not have access to a computer readable version of the forms. Such reports must be designed so that the data prints in exact positions on the form.

Concepts

- Invoice reports are master/detail reports with billing amounts that print conditionally. The customer name, address, and related information are derived from the master query (or group, if there is only one query). The line–items come from the detail query. The billing amounts are printed in the page footer.
- This information must all print on specific line and column positions on the pre–printed invoice form. To format your information correctly, you'll use many of the same techniques as you used in Chapter 31, "Building a Report Using a Pre-Printed Form".
- As an additional aid to formatting this report, you will link to a TIFF image of the invoice. This will enable you to position your fields more precisely.

Data Relationships

- The data for this report is fetched using two queries, linked together in a master/detail relationship. You'll also define a break and two summaries.

Layout

- This layout is similar to the layout for the pre-printed form report, but in this case you will also link to a TIFF image of your form to use as a guide while adjusting the positions of several fields.

Example Scenario

Suppose that you want to create a report that prints an invoice for each customer, listing items purchased, subtotals, and total cost. This report would include the following on an invoice form for each customer:

- standard billing information such as name, address, date, purchase order number, and so on
- a list of items purchased, including item number, description, and price
- the total cost of all items purchased

To see a sample invoice report, open the examples folder named `invoice`, then open the Oracle Reports example named `invoice.rdf`. For details on how to open it, see "Accessing the example reports" in the Preface.

Table 32–1 Features demonstrated in this example

Feature	Location
Create a new, empty report	Section 32.2, "Create a new report manually"
Create two queries with a data link between them	Section 32.3, "Create a data model with a data link"
Add two summary columns and a formula column to the data model to include on the invoice	Section 32.4, "Create summary and formula columns"
Prepare the layout for inserting the invoice information	Section 32.5, "Prepare the layout"
Create new fields for the invoice information, positioning them in the correct locations on the invoice form	Section 32.6, "Insert invoice information"

32.1 Prerequisites for this example

To build the example in this chapter, you must have access to the sample schema provided with the Oracle9*i* database. If you don't know if you have access to this sample schema, contact your database administrator. You should have access to the "Order Entry" portion of the schema to complete this example.

32.2 Create a new report manually

In this case, it is easier to create the data model and layout separately. Hence, we will create an empty report first, then add the queries, and then create the layout.

To create a blank report:

1. Launch Reports Builder (or, if already open, choose **File > New > Report**).
2. In the Welcome or New Report dialog box, select **Build a new report manually**, then click **OK**.

32.3 Create a data model with a data link

When you create a report with multiple queries, it is typically easier to create all of the queries with the Data Wizard in the Data Model view. In this report, you will create two queries, linked with a group-to-group data link.

To create the queries:

1. In the Data Model view, choose **Insert > Query** to display the Data Wizard.
2. If the Welcome page displays, click **Next**.
3. On the Query page, type `Q_Order` for the **Query name**, then click **Next**.
4. On the Data Source page, select **SQL Query**, then click **Next**.

5. On the Data page, in the **Data Source definition** field, enter the following SELECT statement:

```
SELECT
   ORDER_ID,
   ORDER_DATE,
   ORDER_TOTAL,
   ORDERS.CUSTOMER_ID,
   SALES_REP_ID,
   C1.CUST_FIRST_NAME,
   C1.CUST_LAST_NAME,
   C1.CUST_ADDRESS,
   C1.CUSTOMER_ID,
   C2.CUST_FIRST_NAME,
   C2.CUST_LAST_NAME,
   C2.CUST_ADDRESS,
   C2.CUSTOMER_ID
FROM ORDERS, CUSTOMERS C1, CUSTOMERS C2
WHERE ORDERS.CUSTOMER_ID = C1.CUSTOMER_ID AND
   ORDERS.SALES_REP_ID = C2.CUSTOMER_ID
```

This query joins the Orders table to the Customers table for customer information and sales representative information.

Note: You can enter this query in any of the following ways:

- Copy and paste the code from the provided text file called `invoice_code.txt` into the **Data Source definition** field.
- Click **Query Builder** to build the query without entering any code manually.
- Type the code in the **Data Source definition** field.

---

6. Click **Next**.

> **Note:** If you are not already connected to a database, you will be prompted to connect to the database when you click **Next**. Ensure that you connect to a database that has the appropriate schema for this example. Section 32.1, "Prerequisites for this example" describes the sample schema requirements for this example.

7. On the Groups page, click **Next**.
8. Click **Finish** to display your first query in the Data Model view.
9. Repeat the steps above for a second query, but this time name your query `Q_Item` and use the following SELECT statement:

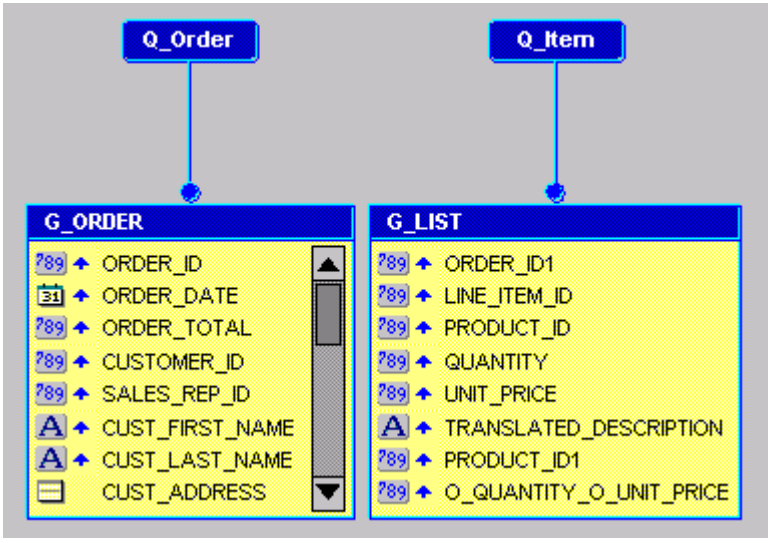
```
SELECT
   O.ORDER_ID,
   O.LINE_ITEM_ID,
   O.PRODUCT_ID,
   O.QUANTITY,
   O.UNIT_PRICE,
   TRANSLATED_DESCRIPTION,
   P.PRODUCT_ID,
   O.QUANTITY * O.UNIT_PRICE
FROM ORDER_ITEMS O, PRODUCT_DESCRIPTIONS P
WHERE  O.PRODUCT_ID = P.PRODUCT_ID
   AND P.LANGUAGE_ID = 'US'
```

   This query joins Order items to the Product table for product descriptions.

10. In the Data Model view, double-click the title bar of the **G_ORDER_ID** group (for the master query **Q_ORDER**) to display the Property Inspector, and set properties:
    - Under **General Information**, set the Name property to G_ORDER.
11. Double-click the title bar of the **G_ORDER_ID1** group (for the detail query **Q_ITEM**) to display the Property Inspector, and set properties:
    - Under **General Information**, set the Name property to G_LIST.

Your data model should now look something like this:

Figure 32–2 Queries in the Data Model view



To add the data link:

1. In the Data Model view, click the Data Link tool in the tool palette.
2. Click and drag from the **ORDER_ID** column in the **G_ORDER group** to the **ORDER_ID1** column in the **G_LIST** group. Notice that a line is drawn from the bottom of the **G_ORDER** group to the **Q_Item** query. Labels for **ORDER_ID** and **ORDER_ID1** are created at each end of the line to indicate they are the columns linking **G_ORDER** to **Q_Item**.

32.4 Create summary and formula columns

To provide the necessary information for the invoice, you need to create summary and formula columns for subtotal amounts, a total amount, and due dates.

To create the invoice summary and formula columns:

1. Click the Summary Column tool in the tool palette, then click in an open area of the Data Model view to create a summary column.

2. Double-click the new summary column object to display the Property Inspector, and set properties:
   - Under **General Information**, set the Name property to CS_SUB_TOTAL.
   - Under **Summary**, set the Source property to O_QUANTITY_O_UNIT_PRICE, and set the Reset At property to Page.

   This summary is used for subtotal amounts on each page of the invoice.
3. Click the **G_ORDER** group, and drag the bottom edge handle down to create an empty space at the bottom of the group.
4. Again, click the Summary Column tool in the tool palette, then click in the empty space in the **G_ORDER** group to create a second summary column.
5. Double-click the new summary column object to display the Property Inspector, and set properties:
   - Under **General Information**, set the Name property to CS_TOTAL.
   - Under **Summary**, set the Source property to O_QUANTITY_O_UNIT_PRICE and set the Reset At property to G_ORDER.

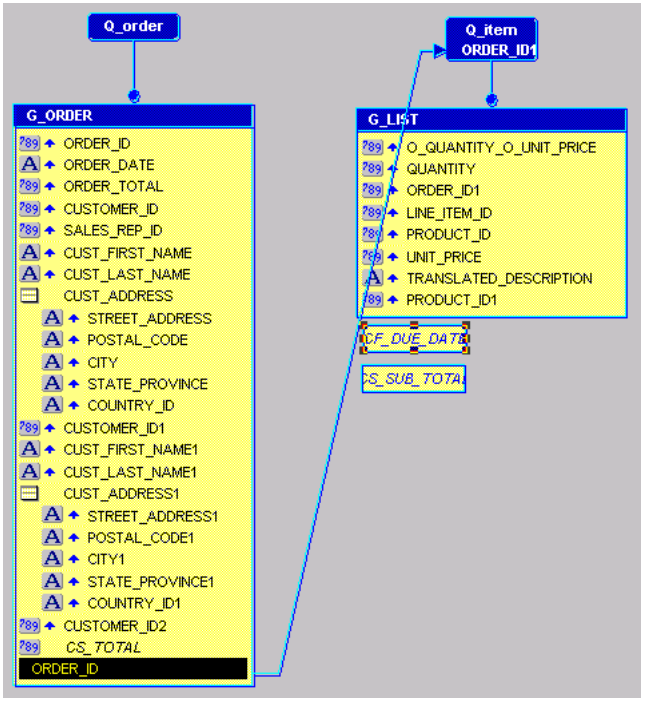
   This summary is used for a total amount of each invoice.
6. Click the Formula Column tool in the tool palette, then click in an open area of the Data Model view to create a formula column.
7. Double-click the new formula column object to display the Property Inspector, and set properties:
   - Under **General Information**, set the Name property to CF_DUE_DATE.
   - Under **Column**, set the Datatype property to Date.
   - Under **Placeholder/Formula**, click the PL/SQL Formula property field to display the PL/SQL Editor.
8. In the PL/SQL Editor, use the template to enter the following PL/SQL code:

```
function CF_DUE_DATEFormula return Date is
today date;
begin
 select sysdate into today from dual;
  return today + 30;
end;
```

   This formula is used to calculate the due date, which is 30 days after today.

Your data model should now look something like this:

Figure 32–3 Data Model view with data link, summary and formula columns



32.5 Prepare the layout

The layout for an invoice report is created by importing an image of the invoice form, then creating desired fields and frames on top of the image. First, you need to set up your layout with the invoice form.

To prepare the invoice report layout:

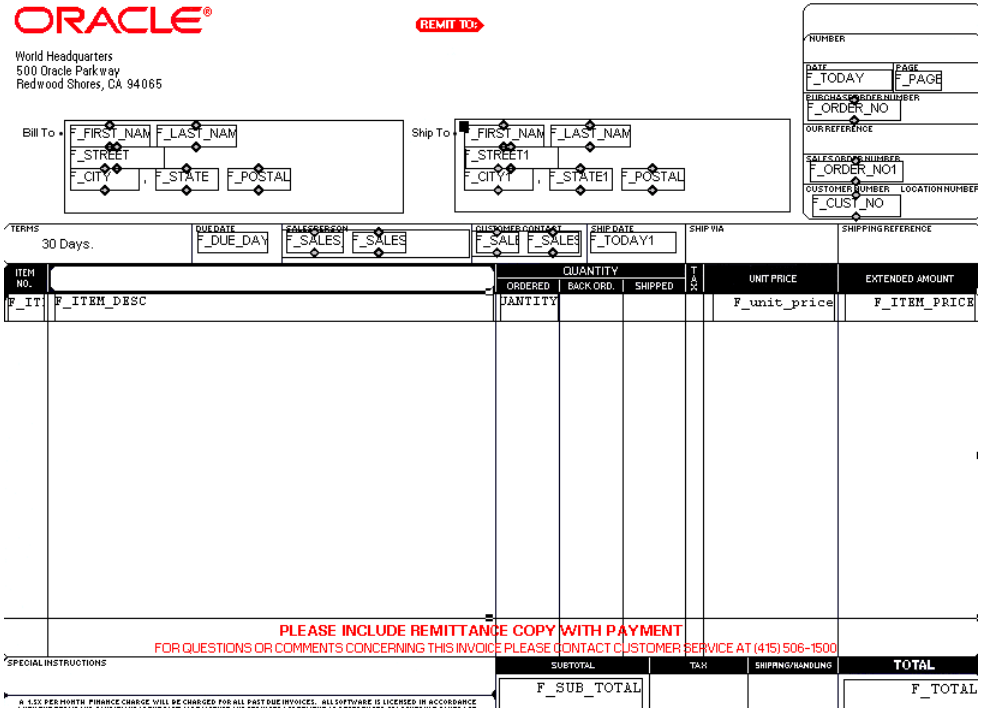
1. Click the Paper Layout button in the toolbar to display the Paper Layout view.
2. In the Paper Layout view, choose **Insert > Image**, and browse to select the provided file `invdemo.tif`. Position the image at the top left corner of the Paper Layout view
3. Double-click the new image object to display the Property Inspector.

4. In the Property Inspector, under **Advanced Layout**, set the Print Object On property to All Pages.
5. In the Object Navigator, expand the **Paper Layout** node, and double-click the properties icon next to **Main Section** to display the Property Inspector, and set properties:
   - Under **Section**, set the Width property to 11 and the Height property to 8.5 for landscape paper orientation to fit the invoice image.
6. In the Paper Layout view, click the Repeating Frame tool in the tool palette, then drag a repeating frame around the invoice image so that the image falls just inside the repeating frame. The repeating frame that you create should cover the image.
7. Double-click the repeating frame to display the Property Inspector, and set properties:
   - Under **General Information**, set the Name property to R_ORDER.
   - Under **Repeating Frame**, set the Source property to G_ORDER, the Print Direction property to Down, and the Maximum Records per Page property to 1.
8. Click the Frame tool in the tool palette, then create a frame just inside the borders of the R_ORDER repeating frame.
9. Double-click the frame object to display the Property Inspector, and set properties:
   - Under **General Information**, set the Name property to M_INVOICE.
   - Under **Advanced Layout**, set the Print Object On property to All Pages.
10. Click the repeating frame, then click the Fill Color tool in the tool palette and choose **No Fill**. Do the same for the frame. You should now see the invoice image, even though it is still layered behind the frame and repeating frame.

32.6 Insert invoice information

You are now ready to create fields on the invoice image to retrieve desired information for the invoice report. When you print the report, the field values will be correctly positioned on the invoice form. As you complete the steps below, use the figure below as a guide (you can also open the provided example report `invoice.rdf` in Reports Builder).

Figure 32–4 Invoice report layout



To insert the information for the invoice report:

1. In the Paper Layout view, click the Text tool in the tool palette, then create a boilerplate text object on top of the invoice image, in the area labeled **Terms**. In the boilerplate text object, type: `30 days`.
2. Click the Field tool in the tool palette, and create fields on top of the invoice image, in the following areas:
   - **Bill To** (6 fields, one for element of the name and address):

     Set Name property to F_FIRST_NAME, Source property to CUST_FIRST_NAME.

     Set Name property to F_LAST_NAME, Source property to CUST_LAST_NAME.

Set Name property to F_STREET, Source property to C_STREET_ ADDRESS.

Set Name property to F_CITY, Source property to C_CITY.

Set Name property to F_STATE, Source property to C_STATE_PROVINCE.

Set Name property to F_POSTAL, Source property to C_POSTAL_CODE.

- **Ship To** (6 fields, one for element of the name and address):

  Set Name property to F_FIRST_NAME1, Source property to CUST_FIRST_ NAME.

  Set Name property to F_LAST_NAME1, Source property to CUST_LAST_ NAME.

  Set Name property to F_STREET1, Source property to C_STREET_ ADDRESS.

  Set Name property to F_CITY1, Source property to C_CITY.

  Set Name property to F_STATE1, Source property to C_STATE_ PROVINCE.

  Set Name property to F_POSTAL1, Source property to C_POSTAL_CODE.

- **Date**: Set Name property to F_TODAY, Source property to Current Date, and Format Mask property to MM/DD/RR.
- **Page**: Set Name property to F_PAGE, Source property to Page Number, and Page Numbering property (in the Page Numbering dialog box) to Reset at: R_ORDER
- **Purchase Order Number**: Set Name property to F_ORDER_NO, Source property to ORDER_ID.
- **Sales Order Number**: Set Name property to F_ORDER_NO1, Source property to ORDER_ID.
- **Customer Number**: Set Name property to F_CUST_NO, Source property to CUSTOMER_ID.
- **Due Date**: Set Name property to F_DUE_DAY, Source property to CF_ DUE_DATE.
- **Salesperson** (2 fields, one for first name, one for last name):

  Set Name property to F_SALES_REP_FNAME, Source property to CUST_ FIRST_NAME.

Set Name property to F_SALES_REP_LNAME, Source property to CUST_LAST_NAME.

- **Customer Contact**: (2 fields, one for first name, one for last name):

  Set Name property to F_SALES_REP_FNAME1, Source property to CUST_FIRST_NAME.

  Set Name property to F_SALES_REP_LNAME1, Source property to CUST_LAST_NAME.

  Note: this examples assumes that the customer contact is same person as the sales representative.

- **Ship Date**: Set Name property to F_TODAY1, Source property to Current Date, and Format Mask property to MM/DD/RR.
- **Item No.**: Set Name property to F_ITEM_NO, Source property to LINE_ITEM_ID. With the field selected, click the Align Right button in the toolbar.
- **Description**: Set Name property to F_ITEM_DESC, Source property to TRANSLATED_DESCRIPTION.
- **Quantity Ordered**: Set Name property to F_QUANTITY, Source property to QUANTITY. With the field selected, click the Align Right button in the toolbar.
- **Unit Price**: Set Name property to F_UNIT_PRICE, Source property to UNIT_PRICE, and Format Mask property to $NNN,NN0.00. With the field selected, click the Align Right button in the toolbar.
- **Extended Amount**: Set Name property to F_ITEM_PRICE, Source property to O_QUANTITY_O_UNIT_PRICE, and Format Mask property to $NNN,NN0.00. With the field selected, click the Align Right button in the toolbar.
- **Subtotal**: Set Name property to F_SUB_TOTAL, Source property to CS_SUB_TOTAL.
- **Total**: Set Name property to F_TOTAL, Source property to CS_TOTAL.

3. Click the Frame tool in the tool palette, and draw a frame around the following fields to group them:

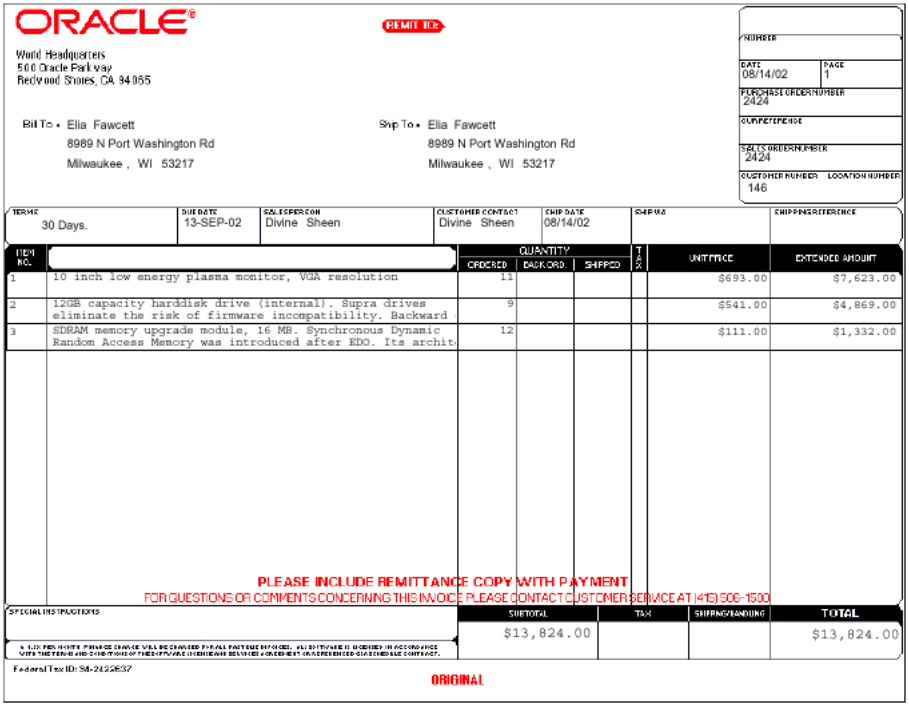
   - **Bill To**: F_FIRST_NAME, F_LAST_NAME, F_STREET, F_CITY, F_STATE, and F_POSTAL. Set the Frame's Name property to M_BILL_TO.

- **Ship To**: F_FIRST_NAME1, F_LAST_NAME1, F_STREET1, F_CITY1, F_STATE1, and F_POSTAL1. Set the Frame's Name property to M_SHIP_TO.
- **Salesperson**: F_SALES_REP_FNAME and F_SALES_REP_LNAME. Set the Frame's Name property to M_SALES_REP.
- **Customer Contact**: F_SALES_REP_FNAME1 and F_SALES_REP_LNAME1. Set the Frame's Name property to M_CUST_CONTACT.
- **Item list**: F_ITEM_NO, F_ITEM_DESC, F_QUANTITY, F_UNIT_PRICE, and F_ITEM_PRICE, including all the empty space below these fields to the bottom border of the list area. Set the Frame's Name property to M_ORDER_LIST. This frame is used to position and size the detail item list repeating frame.

Note: Using a frame ensures that other objects will not be pushed by the variable horizontal size of the name fields.

4. Click the Repeating Frame tool in the tool palette, and draw a repeating frame around the following fields to group them:
   - Item list: F_ITEM_NO, F_ITEM_DESC, F_QUANTITY, F_UNIT_PRICE, and F_ITEM_PRICE, including just the fields and *not* the empty space below the fields. Set the Repeating Frame's Name property to R_ORDER_LIST, and the Source property to G_LIST.
5. Save and run your report. Adjust the position of objects if necessary to fit to the background of the invoice image. Your final report output should look something like this:

Figure 32–5 Final invoice report output



ORACLE

REMIT TO:

World Headquarters
500 Oracle Parkway
Redwood Shores, CA 94065

NUMBER

DATE 08/14/02 | PAGE 1

PURCHASE ORDER NUMBER 2424

OUR REFERENCE

SALES ORDER NUMBER 2424

CUSTOMER NUMBER 146 | LOCATION NUMBER

Bill To: Ella Fawcett
8989 N Port Washington Rd
Milwaukee , WI 53217

Ship To: Ella Fawcett
8989 N Port Washington Rd
Milwaukee , WI 53217

TERMS	DUE DATE	SALESPERSON	CUSTOMER CONTACT	SHIP DATE	SHIP VIA	SHIPPING REFERENCE
30 Days.	13-SEP-02	Divine Sheen	Divine Sheen	08/14/02		

ITEM NO.		QUANTITY ORDERED	QUANTITY BACK ORD.	QUANTITY SHIPPED	TAX	UNIT PRICE	EXTENDED AMOUNT
1	10 inch low energy plasma monitor, VGA resolution	11				$693.00	$7,623.00
2	12GB capacity harddisk drive (internal). Supra drives eliminate the risk of firmware incompatibility. Backward	9				$541.00	$4,869.00
3	SDRAM memory upgrade module, 16 MB. Synchronous Dynamic Random Access Memory was introduced after EDO. Its archit	12				$111.00	$1,332.00

PLEASE INCLUDE REMITTANCE COPY WITH PAYMENT
FOR QUESTIONS OR COMMENTS CONCERNING THIS INVOICE PLEASE CONTACT CUSTOMER SERVICE AT (415) 506-1500

SPECIAL INSTRUCTIONS	SUBTOTAL	TAX	SHIPPING/HANDLING	TOTAL
	$13,824.00			$13,824.00

Federal Tax ID: 94-2229637

ORIGINAL

32.7 Summary

Congratulations! You have successfully created an invoice report. You now know how to:

- create two queries with a data link between them.
- add two summary columns and a formula column to the data model to include on the invoice.
- prepare a report layout for adding information onto an image of a invoice.
- create and position fields in the correct locations on an invoice form.

For more information on any of the wizards, views, or properties used in this example, refer to the *Reports Builder Online Help*, which you can access in two ways:

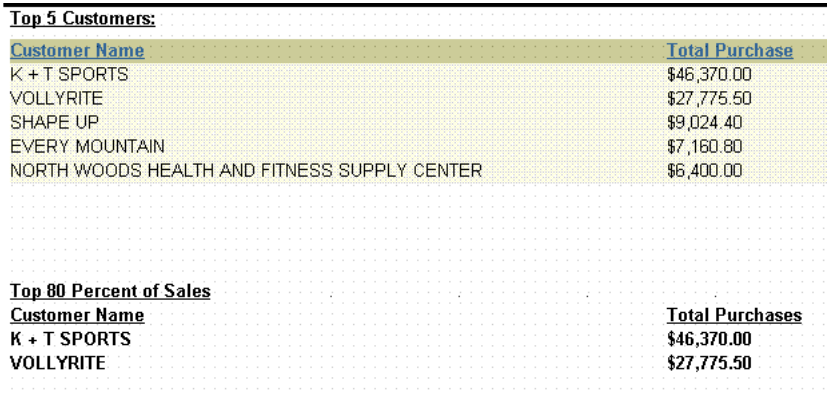
- From the Oracle Technology Network (`http://otn.oracle.com/products/reports/`), click **Documentation** and navigate to the *Reports Builder Online Help* for the most recent, hosted online help.
- From Reports Builder, choose **Help > Help Contents**.

Note: You can replace your Reports Builder online help with the most recent update by downloading the latest online help set available on the Oracle Technology Network (`http://otn.oracle.com/docs/products/reports/`). Instructions for replacing your help file are included in the readme.txt in the download file.

33

Building a Ranking Report

Figure 33–1 Tabular report output, ranked by number and percentage of customers



Top 5 Customers:

Customer Name	Total Purchase
K + T SPORTS	$46,370.00
VOLLYRITE	$27,775.50
SHAPE UP	$9,024.40
EVERY MOUNTAIN	$7,160.80
NORTH WOODS HEALTH AND FITNESS SUPPLY CENTER	$6,400.00

Top 80 Percent of Sales

Customer Name	Total Purchases
K + T SPORTS	$46,370.00
VOLLYRITE	$27,775.50

This report ranks data two different ways: by count and by percentage. The upper portion displays the names and the total purchases of the top three customers; the lower portion displays the names and total purchases of those customers who constitute 75% of all sales. You can set the ranking criteria at runtime, or let them default to previously specified values.

Concepts

This example report ranks the data by comparing it to a user-specified bind parameter. It requires one query to fetch the data. To rank the data, you'll create the following objects:

- A Bind parameter that gives you the option of setting the cutoff point at runtime.

- A Bind parameter that increments by one each time a record is fetched from the database. The value in this parameter will be compared to the cutoff Bind parameter.
- A customized group filter to include only records within the cutoff value. Using a customized filter rather than a packaged filter allows you to parameterize your cutoff values.

Layout

- This report uses a tabular layout style, with some modifications.

Example Scenario

Suppose that you want to create a report that displays the names and total purchases of your top 3 customers and the top 75% of all customers. Furthermore, suppose that you decide it would be better to let the users of the report choose the ranking criteria (i.e., the top x customers and the top x% of all customers).

To see a sample ranking report, open the examples folder named `ranking`, then open the Oracle Reports example named `rank.rdf`. For details on how to access it, see "Accessing the example reports" in the Preface.

Table 33–1 Features demonstrated in this example

Feature	Location
Create a data model and a tabular layout	Section 33.2, "Create a data model and tabular layout"
Create parameters and group filter to control the ranking criteria	Section 33.3, "Create ranking logic for top number of customers"
Add a layout object to display the parameter value in the report output	Section 33.4, "Add a layout object for a parameter"
Create a Parameter Form that only displays the parameter you want users to set	Section 33.5, "Create a Parameter Form"
Add a percentage ranking	Section 33.6, "Add a percentage ranking"

33.1 Prerequisites for this example

To build the example in this chapter, you must have access to the Summit Sporting Goods schema, which we've provided on the Oracle Technology Network (`http://otn.oracle.com/products/reports/`). To download the SQL

scripts that install the schema, go to the Documentation page on OTN and follow the instructions provided on the Web page.

33.2 Create a data model and tabular layout

When you are creating a single-query report, such as this one, you can use the Report Wizard to create the data model and layout simultaneously.

To create a data model and layout:

1. Launch Reports Builder (or, if already open, choose **File > New > Report**).
2. In the Welcome or New Report dialog box, select **Use the Report Wizard**, then click **OK**.
3. If the Welcome page displays, click **Next**.
4. On the Report Type page, select **Create Paper Layout Only**, then click **Next**.
5. On the Style page, type a **Title** for your report, select **Tabular**, then click **Next**.
6. On the Data Source page, click **SQL Query**, then click **Next**.
7. On the Data page, enter the following SELECT statement in the **Data Source definition** field:

```
SELECT CUSTNAME CNAME, SUM(AMOUNT) SUM_AMT FROM SALES
GROUP BY CUSTNAME
ORDER BY SUM(AMOUNT) DESC
```

> **Note:** You can enter this query in any of the following ways:
>
> - Copy and paste the code from the provided text file called `rank_code.txt` in the **Data Source definition** field.
> - Click **Query Builder** to build the query without entering any code manually.
> - Type the code in the **Data Source definition** field.

8. Click **Next**.

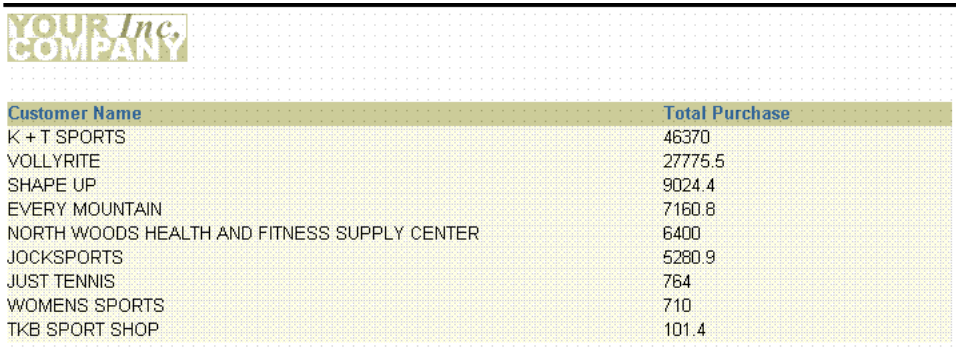
> **Note:** If you are not already connected to a database, you will be prompted to connect to the database when you click **Next**. Ensure that you connect to a database that has the appropriate schema for this example. Section 33.1, "Prerequisites for this example" describes the sample schema requirements for this example.

9. On the Fields page, click the double right arrows (>>) to move all of the fields to the **Displayed Fields** list, then click **Next**.
10. On the Totals page, click **Next**.
11. On the Labels page, set the labels and field widths as follows, then click **Next**:

Fields	Labels	Width
CNAME	`Customer Name`	`35`
SUM_AMT	`Total Purchases`	`15`

12. On the Template page, click **Finish** to preview your report output in the Paper Design view. It should look something like this:

Figure 33–2 Paper Design view for the ranking report



33.3 Create ranking logic for top number of customers

Now that your report has a data model and layout, you can add the logic to control the number of customers displayed. First, you will create two parameters that the user can enter values for at runtime. Second, you will create a group filter that uses the parameters to control which data is included.

To create parameters and add a group filter:

1. In the Object Navigator, under the **Data Model** node, click the **User Parameters** node.
2. Choose **Edit > Create** to create a new user parameter under the **User Parameters** node.
3. If the Property Inspector is not already displayed, right-click the new user parameter (**P_1**), then choose **Property Inspector** to display the Property Inspector, and set properties:
   - Under **General Information**, set the Name property to CUTOFF_CNT.
   - Under **Parameter**, set the Datatype property to Number, set the Width property to 1, and set the Initial Value property to 3.
4. Repeat the steps above to create another user parameter, using the following property settings this time:
   - Under **General Information**, set the Name property to INCR_CNT.
   - Under **Parameter**, set the Datatype property to Number, set the Width property to 3, and set the Initial Value property to 0.
5. In the Object Navigator, under the **Data Model** node, expand the **Groups** node, then double-click the properties icon next to the **G_CNAME** group to display the Property Inspector, and set properties:
   - Under **Group**, set the Filter Type property to PL/SQL, then click the PL/SQL Filter property field to display the PL/SQL Editor.
6. In the PL/SQL Editor, use the template to enter the following PL/SQL code:

```
function G_CNAMEGroupFilter return boolean is
begin
  :incr_cnt:=:incr_cnt+1;
    if :incr_cnt <= :cutoff_cnt then
      return (TRUE);
    else
      return(FALSE);
```

```
    end if;
end;
```

> **Note:** You can enter this code by copying and pasting it from the provided text file called `rank_code.txt`. This code is for the Group Filter.

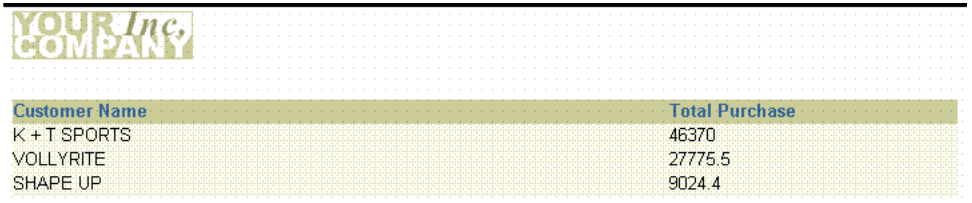
7. Click **Compile**.
8. Click **Close**.

   This filter increments the counter by 1 each time a record in G_CNAME is fetched, then compares the counter's value to the specified cutoff. When the counter exceeds the cutoff, no more records are fetched.

   > **Tip:** Notice that, if the Paper Design view is still open while you add this logic, the report now returns no records in the Paper Design view. To fix this issue, you should display one of the other views (e.g., the Data Model view) and then come back to the Paper Design view. You will be prompted by the Runtime Parameter Form to enter values for the two parameters, INCR_CNT and CUTOFF_CNT

9. Click the Run Paper Layout button in the toolbar to display the report output in the Paper Design view
10. Save your report.

Figure 33–3 Tabular report output restricted to top three customers



33.4 Add a layout object for a parameter

As a way for users to quickly tell the number of customers displayed in the list, you need to display the value of CUTOFF_CNT in the report. To perform this task, you will create a boilerplate text object that references the parameter.

1. In the Object Navigator, double-click the view icon next to the **Paper Layout** node to display the Paper Layout view.
2. Position the Paper Layout view and Object Navigator side-by-side so that they are visible simultaneously.
3. In the Object Navigator, in the **Find** field, type `M_G_CNAME_GRPFR`.
4. Click **M_G_CNAME_GRPFR** in the Object Navigator to select it in the Paper Layout view.
5. Click the title bar of the Paper Layout view to make it the active window.
6. Select **M_G_CNAME_GRPFR** and all of its contents by press CTRL-A on your keyboard.

> **Note:** You can also use the Find field in the Object Navigator to locate specific objects. When you select an item in the Object Navigator while the Paper Layout view is displayed, the corresponding object is selected in the Paper Layout view.

7. Use the down arrow key to move the items down about one inch.
8. From the font lists in the toolbar, choose **Arial (Western)**, point size **10**.
9. Click the Text tool in the tool palette.
10. Click directly above the label **Customer Name** to create a new boilerplate text object and type the following text:

    ```
    Top &CUTOFF_CNT Customers:
    ```

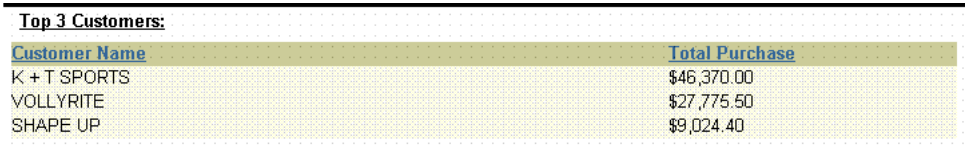
11. Move to an open area of the Paper Layout view and click the mouse button to exit text mode. Notice that the text object you just created is still selected, you can now adjust its positioning with the arrow keys.
12. Click the Bold button in the toolbar to make the new text bold.
13. Shift-click on the labels **Customer Name** and **Total Purchase** so that they are selected simultaneously with the object you just created.

14. Click the Underline button in the toolbar.
15. Click in an open area of the Paper Layout view to deselect everything.
16. Click **F_SUM_AMT**, then choose **Tools > Property Inspector** to display the Property Inspector, and set properties:
    - Under **Field**, set the Format Mask property by typing the following:

      ```
      -$NNN,NN0.00
      ```

17. Click the Paper Design button in the toolbar to display the Paper Design view. (If the Runtime Parameter Form displays, click the Run Paper Layout button in the toolbar.)

Figure 33–4 Tabular report output with parameter value displayed



Top 3 Customers:

Customer Name	Total Purchase
K + T SPORTS	$46,370.00
VOLLYRITE	$27,775.50
SHAPE UP	$9,024.40

18. Save your report.

33.5 Create a Parameter Form

By default, both of your user parameters appear in the Runtime Parameter Form. In reality, you only want users to set CUTOFF_CNT. You do not want them to be able to set INCR_CNT (the amount by which your counter is increased for each record). To prevent users from seeing INCR_CNT on the Runtime Parameter Form, you will build your own Parameter Form.

To create a Parameter Form:

1. Choose **Tools > Parameter Form Builder**.
2. In the Parameter Form Builder dialog box, scroll down the list of parameters until you find **INCR_CNT**.
3. Click **INCR_CNT** to deselect it.
4. Change the label for **CUTOFF_CNT** to:

```
# of Top Customers:
```

5. Click **OK**.
6. Click the Run Paper Layout button in the toolbar.
7. In the Runtime Parameter Form, type a value for **# of Top Customers**, then click the Run Report button in the toolbar. You should now see as many records as you asked for in the Runtime Parameter Form and they should be in order from largest total purchases to smallest total purchases.

   Tip: As an additional exercise, you could now change the Initial Value property of CUTOFF_CNT to see its effect on the Runtime Parameter Form.

8. Save your report.

33.6 Add a percentage ranking

Another way to rank customers is by percentage of total sales. The idea is the same as ranking by count, but with an important difference. Since you need to fetch all of the data in order to compute a running percent of total summary, you don't want to use a group filter to weed out data. You need to use a format trigger on the repeating frame to compare the running total to your cutoff parameter.

To fetch the data for the percentage calculation:

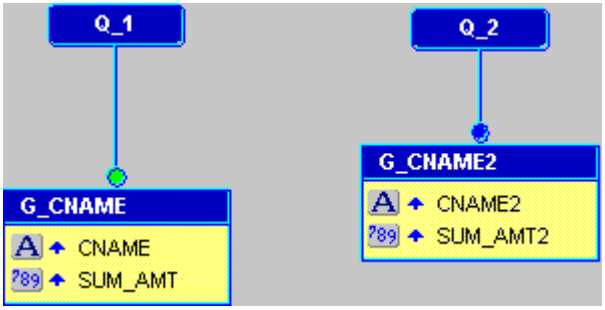
1. In the Object Navigator, double-click the view icon next to the **Data Model** node to display the Data Model view.
2. In the Data Model view, click the SQL Query tool in the tool palette.
3. Click in an open area of the Data Model view, somewhere to the right of the first query (**Q_1**) to display the SQL Query Statement dialog box.
4. In the SQL Query Statement dialog box, enter the following SELECT statement:

```
SELECT CUSTNAME CNAME2, SUM(AMOUNT) SUM_AMT2
FROM SALES
GROUP BY CUSTNAME
ORDER BY SUM(AMOUNT) DESC
```

Note: You can enter this code by copying and pasting it from the provided text file called `rank_code.txt`. This code is for the Percentage Calculation.

5. Click **OK**. Your data model should now look like the following image.

Figure 33–5 Data Model view of the Ranking Report



6. Save your report.

To create a parameter for the percentage cutoff:

1. In the Object Navigator, under the **Data Model** node, click the **User Parameters** node.
2. Choose **Edit > Create** to create a new user parameter under the **User Parameters** node.
3. If the Property Inspector is not already displayed, right-click the new user parameter (**P_1**), then choose **Property Inspector** to display the Property Inspector, and set properties:
   - Under **General Information**, set the Name property to CUTOFF_PCT.
   - Under **Parameter**, set the Datatype property to Number, set the Width property to 2, and set the Initial Value property to 75.
4. In the Data Model view, click the group object **G_CNAME2**, then click and drag the bottom handle down about 0.25 inches to make the group bigger.

5. Click the Summary Column tool in the tool palette.
6. Click in the empty space beneath **SUM_AMT2** to create a summary column.
7. Double-click the new summary column object (**CS_1**) to display the Property Inspector, and set properties:
   - Under **General Information**, set the Name property to R_PCT.
   - Under **Column**, set the Datatype property to Number, and set the Width property to 10.
   - Under **Summary**, set the Function property to % of Total, set the Source property to SUM_AMT2, set the Reset At property to Report, and set the Compute At property to Report.

To create a second layout for the list of customers by percentage:

1. Click the Paper Layout button in the toolbar to display the Paper Layout view.
2. In the Paper Layout view, click the Report Block tool in the tool palette.
3. Starting about 1 inch below the existing layout, click and drag a box about the same size as the existing layout to define the size of the second layout and display the Report Block Wizard.
4. On the Style page of the Report Block Wizard, select **Tabular**, then click **Next**.
5. On the Groups page, click **G_CNAME2** in the **Available Groups** list and click **Down** to specify its Print Direction and move this group to the **Displayed Groups** list, then click **Next**.
6. On the Fields page, click the following fields and click the right arrow (**>**) to move them to the **Displayed Fields** list, then click **Next**:
   - **CNAME2**
   - **SUM_AMT2**
7. On the Labels page, change the labels and field widths as follows, then click **Next**:

Fields	Labels	Width
CNAME2	`Customer Name`	`35`
SUM_AMT2	`Total Purchases`	`15`

8. On the Template page, select **No template**. (If you were to select a template, it would override the template of the previous layout), then click **Finish** to display your report layout in the Paper Layout view.
9. From the font lists in the toolbar, choose **Arial (Western)**, point size **10**.
10. In the Paper Layout view, click the Text tool in the tool palette.
11. Click directly above the label **Customer Name** in the new layout and type the following text in the new boilerplate text object:

    ```
    Top &CUTOFF_PCT Percent of Sales:
    ```

12. Move to an open area of the Paper Layout view and click the mouse button to exit text mode. Notice that the text object you just created is still selected, you can now adjust its positioning with the arrow keys.
13. If the text is not already bold, click the Bold button in the toolbar to make it bold.
14. Shift-click on the labels **Customer Name** and **Total Purchase** so that they are selected simultaneously with the object you just created.
15. Click the Underline tool in the toolbar.
16. Click in an open area of the Paper Layout view to deselect everything.
17. Double-click field **F_SUM_AMT2** to display the Property Inspector, and set properties:
    - Under **Field**, set the Format Mask property by typing:

      ```
      -$NNN,NN0.00
      ```

To add the logic for the percentage cutoff:

1. In the Object Navigator, type `R_G_CNAME2` in the **Find** field to locate that repeating frame.
2. Double-click the properties icon next to **R_G_CNAME2** to display the Property Inspector, and set properties:
    - Under **Advanced Layout**, click the Format Trigger property field to display the PL/SQL Editor.
3. In the PL/SQL Editor, use the template to enter the following PL/SQL code:

    ```
    function R_G_CNAME2FormatTrigger return boolean is
    ```

```
begin
if :r_pct <= :cutoff_pct then
   return(TRUE);
else
   return(FALSE);
end if;
end;
```

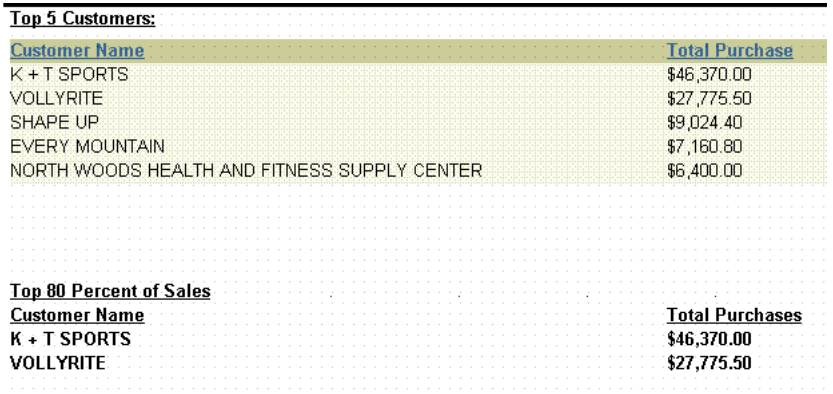
> **Note:** You can enter this code by copying and pasting it from the provided text file called `rank_code.txt`. This code is for the Percentage Cutoff.

To update the Parameter Form for the new percentage parameter:

1. Choose **Tools > Parameter Form Builder**.
2. In the Parameter Form Builder dialog box, scroll down the list of parameters until you find **CUTOFF_PCT**, and change its label to:

   `Top Percentage (%):`
3. Click **OK**.
4. Click the Run Paper Layout button in the toolbar.
5. In the Runtime Parameter Form:
   - For **# of Top Customers**, type `5`.
   - For **Top Percentage (%)**, type `80`.
6. Click the Run Report button in the toolbar.

Figure 33–6 Tabular report output restricted by number and percentage of customers



Top 5 Customers:

Customer Name	Total Purchase
K + T SPORTS	$46,370.00
VOLLYRITE	$27,775.50
SHAPE UP	$9,024.40
EVERY MOUNTAIN	$7,160.80
NORTH WOODS HEALTH AND FITNESS SUPPLY CENTER	$6,400.00

Top 80 Percent of Sales

Customer Name	Total Purchases
K + T SPORTS	$46,370.00
VOLLYRITE	$27,775.50

33.7 Summary

Congratulations! You have successfully created a ranking report. You now know how to:

- create a data model and a tabular layout.
- create parameters and group filter to control the ranking criteria.
- add a layout object to display the parameter value in the report output.
- create a Parameter Form that only displays the parameter you want users to set.
- add a percentage ranking.

For more information on any of the wizards, views, or properties used in this example, refer to the *Reports Builder Online Help*, which you can access in two ways:

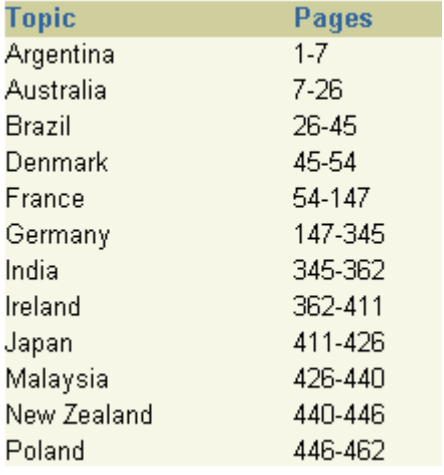
- From the Oracle Technology Network (`http://otn.oracle.com/products/reports/`), click **Documentation** and navigate to the *Reports Builder Online Help* for the most recent, hosted online help.
- From Reports Builder, choose **Help > Help Contents**.

Note: You can replace your Reports Builder online help with the most recent update by downloading the latest online help set available on the Oracle Technology Network (`http://otn.oracle.com/docs/products/reports/`). Instructions for replacing your help file are included in the readme.txt in the download file.

34

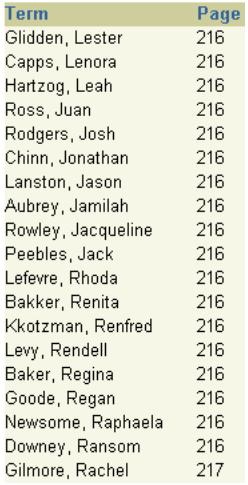
Building a Paper Report with a Simple Table of Contents and Index

Figure 34–1 Simple table of contents for a large report



Topic	Pages
Argentina	1-7
Australia	7-26
Brazil	26-45
Denmark	45-54
France	54-147
Germany	147-345
India	345-362
Ireland	362-411
Japan	411-426
Malaysia	426-440
New Zealand	440-446
Poland	446-462

Figure 34–2 Simple index for a large report



Term	Page
Glidden, Lester	216
Capps, Lenora	216
Hartzog, Leah	216
Ross, Juan	216
Rodgers, Josh	216
Chinn, Jonathan	216
Lanston, Jason	216
Aubrey, Jamilah	216
Rowley, Jacqueline	216
Peebles, Jack	216
Lefevre, Rhoda	216
Bakker, Renita	216
Kkotzman, Renfred	216
Levy, Rendell	216
Baker, Regina	216
Goode, Regan	216
Newsome, Raphaela	216
Downey, Ransom	216
Gilmore, Rachel	217

When you have large paper report, it's sometimes easier for your users to navigate through the report if you include a table of contents. Using Reports Builder, you can generate a table of contents that displays at the beginning of your paper report.

Normally, Reports Builder formats the report starting with the Header section, then the Main section, followed by the Trailer section. Now, Reports Builder can format any section first to create information that is only known at the time of formatting, such as page numbers, then use that information in the formatting of a previous section.

In this example, you will use the built-in procedure called SRW.SET_FORMAT_ORDER to generate the Main section of the report first, then the Header and Trailer sections. Doing so enables you to generate the page numbers for the report in the Main section, then display the page numbers in the table of contents in the Header section.

Note: Although you can use SRW.SET_FORMAT_ORDER to change the order in which the report is formatted, the report will still display the Header section first, then the Main section, and finally the Trailer section. You can use SRW.SET_FORMAT_ORDER to simply change the formatting order, but not the display order of the report sections.

You will use the After Parameter Form trigger and a format trigger to place the page numbers temporarily in a table you create in the database. You will then create a simple tabular report that displays at the beginning of your report, that lists the category and its page number. Finally, you will create an index that simply lists the record and the page number in the Trailer section of the report.

For information on creating a multilevel table of contents, refer to Chapter 35, "Building a Paper Report with a Multilevel Table of Contents".

Example Scenario

Suppose you want to generate a list of all the e-mail addresses for your customer database. Your customer database contains thousands of records. First, you want to categorize the customers (in this case, by country), then generate a table of contents so you can easily find the e-mail address for the desired customer.

Table 34–1 Features demonstrated in this example

Feature	Location
Use SQL*Plus to create a table in your database to hold the table of contents (TOC) information.	Section 34.2.1, "Create a table in the database to hold the information for the TOC"
Use the Report Wizard to create a simple group above report.	Section 34.2.2, "Create a group above report"
Use the Data Model view to create a second query.	Section 34.2.4, "Create a second query in the data model"
Use the Report Block Wizard to create a tabular report to display the country name and page number.	Section 34.2.5, "Create a report block to display the table of contents"
Create a format trigger to generate a page number for every customer record.	Section 34.3.2, "Create a format trigger"
Use the Data Model view to create another query that retrieves the necessary information for the index.	Section 34.3.3, "Add a query to the data model"
Use the Report Block Wizard to display the index.	Section 34.3.4, "Create a report block to display the index in the Trailer section"

34.1 Prerequisites for this example

To build the examples in this manual, you must have the example files we've provided, as well as access to the sample schema that is shipped with the Oracle9*i* database.

34.1.1 Example files

If you haven't already done so, you can download the files you'll need to complete this example from the Oracle Technology Network (`http://otn.oracle.com/products/reports/`) and install them on your machine.

To download and install the example files:

1. Go to the Oracle Technology Network Web site (`http://otn.oracle.com/products/reports/`).
2. Click **Getting Started with Oracle Reports**.
3. Click **Index**, then browse through the list of examples and find the "Building a Report with a Table of Contents and Index" example.
4. Download the file `formatorder.zip` into a temporary directory on your machine (e.g., `d:\temp`).
5. Unzip the contents of the file, maintaining the directory structure, into an examples directory on your machine (e.g., `d:\orawin90\examples`).

   This zip file contains the following files:

Table 34–2 Files necessary for building a report with a simple table of contents and an index

File	Description
`Examples\result\formatorder.rdf`	The final RDF version of the paper report.
`Examples\result\formatorder_index.rdf`	The final RDF version of the paper report with an index.
`Examples\scripts\formatorder_code.txt`	The various SQL and PL/SQL code you will use in this report.

Note: The index.html file and assets directory are used as part of the Getting Started with Oracle Reports Web site. Please do not delete or move these files

34.1.2 Access to the sample schema

If you don't know if you have access to the sample schema provided with the Oracle9*i* database, contact your database administrator. You should have access to the "Sales History" portion of the schema to complete this example. Typically, you can log into this schema by using the user ID and password "`sh/sh`", then enter the name of the database.

34.2 Create a simple table of contents for a large report

The steps in this section will show you how to create a basic table of contents for a large, multipage report. The table of contents will look like the following:

Figure 34–3 Simple table of contents

Topic	Pages
Argentina	1-7
Australia	7-26
Brazil	26-45
Denmark	45-54
France	54-147
Germany	147-345
India	345-362
Ireland	362-411
Japan	411-426
Malaysia	426-440
New Zealand	440-446
Poland	446-462

34.2.1 Create a table in the database to hold the information for the TOC

The steps in this section will show you how to create a table in the database that will hold the page numbers for the records you want to list in the table of contents. If you are not sure whether you have the privileges to create a table in the database, contact the database administrator.

To create a table in the database:

1. Start SQL*Plus.
   - In Windows, from the Start menu, choose **Programs > Oracle Application Server - oracle_home_name> Application Development > SQL Plus**.
   - On UNIX, type `sqlplus`.
2. Connect to the Sales History portion of the database (e.g., use the `sh/sh@`*`database name`* connect string).
3. At the SQL prompt, type the following line:

   ```
   create table toc_example (topic varchar2(100), page number);
   ```

4. Press Enter.

   You should see a notification that the table has been created.
5. Exit SQL*Plus.

34.2.2 Create a group above report

The steps in this section will show you how to build a simple group above report in Reports Builder. This group above report will display the country name, then the customers and customer e-mail addresses under each country name. The table of contents you create will then be based on the country name in this report.

To create a simple group above report:

1. In Reports Builder, choose **File > New > Report**, then choose to create the report manually.
2. In the Data Model view that displays, right-click on the canvas, then choose **Report Wizard** from the pop-up menu.
3. On the Report Type page, select **Create Paper Layout only**, then click **Next**.
4. On the Style page, select **Group Above**, then click **Next**.
5. On the Data Source page, click **SQL Query**, then click **Next**.
6. On the Data page, enter the following SELECT statement in the **SQL Query Statement** field:

   ```
   SELECT ALL COUNTRIES.COUNTRY_NAME,
              CUSTOMERS.CUST_LAST_NAME,
   ```

```
            CUSTOMERS.CUST_FIRST_NAME,
            CUSTOMERS.CUST_e-mail
FROM CUSTOMERS, COUNTRIES
WHERE (CUSTOMERS.COUNTRY_ID = COUNTRIES.COUNTRY_ID)
```

Note: You can enter this query in any of the following ways:

- Copy and paste the code from the provided text file called `formatorder_code.txt` into the **Data Source definition** field.
- Click **Query Builder** to build the query without entering any code manually.
- Type the code in the **Data Source definition** field.

---

7. Click **Next**.

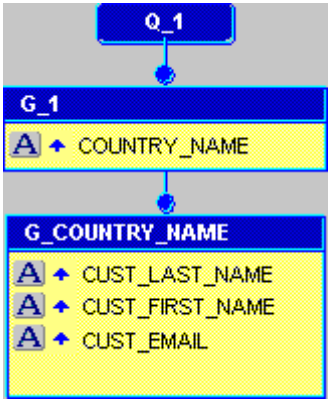
Note: If you are not already connected to a database, you will be prompted to connect to the database when you click **Next**. Ensure that you connect to a database that has the appropriate schema for this example. Section 34.1, "Prerequisites for this example" describes the sample schema requirements for this example.

---

8. On the Groups page, click **COUNTRY_NAME**, then click **>** to move the field to the Group Fields list.
9. Click **Next**.
10. On the Fields page, click **>>** to move all fields to the Displayed Fields list, then click **Next**.
11. On the Totals page, click **Next**.
12. On the Labels page, click **Next**.
13. On the Template page, click **Finish**.

14. In the toolbar, click the Data Model view button. The data model should look like this:

Figure 34–4 Data model for the group above report



15. In the toolbar, click the Paper Layout view button. The Paper Layout view should look like this:

Figure 34–5 Paper Layout view of the group above report

16. Save your report as `formatorder_<your initials>.rdf`. You have now created the data model and the basic layout for your report.

34.2.3 Create an After Parameter Form trigger and a format trigger

The steps in this section will show you how to create two triggers. The first trigger will use the SRW.SET_FORMAT_ORDER built-in procedure to format the Main section of the report first, then the Header and Trailer sections. The second trigger will fetch the number of the page on which each country name displays, and place

that information into the table you created in Section 34.2.1, "Create a table in the database to hold the information for the TOC".

Normally, Reports Builder formats a report with the Header section first, followed by the Main and Trailer sections. By changing the order so that the Main section is formatted first, you can generate the page numbers for the records, place this data into a database table, then generate the table of contents based on that data. In later steps, you will create the table of contents in the Header section of the report.

For more information on the SRW built-in package, refer to the *Reports Builder Online Help*.

34.2.3.1 Create an After Parameter Form trigger

An After Parameter Form trigger is a function that executes after the parameter form is executed. In this section, you will set the trigger to change the order of the report execution, so that the Main section runs first.

To create an After Parameter Form trigger:

1. In the Object Navigator, under your report name (**FORMATORDER_<your initials>**), expand the **Report Triggers** node.
2. Right-click **AFTER PARAMETER FORM**, then choose **PL/SQL Editor** from the pop-up menu.
3. In the PL/SQL Editor, use the template to enter the following PL/SQL code:

```
function AfterPForm return boolean is
begin
    srw.set_format_order(srw.main_section, srw.header_section,
    srw.trailer_section);
    return (TRUE);
end;
```

> **Note:** You can also copy and paste the code from the provided file, `formatorder_code.txt`.

4. Click **Compile**.
5. Once the code has compiled, click **Close**.

34.2.3.2 Create a format trigger

In this section, you will create a format trigger based on the field F_COUNTRY_NAME. This field displays the country name. This format trigger will fetch the page number for each country name, so that the table of contents will enable the user to navigate to various parts of the report based on the name of the country.

To create a format trigger:

1. In the Object Navigator, under the **Paper Layout** node for your report, navigate to **Main Section > Body > M_G_1_GRPFR > R_G_1**, then click **F_COUNTRY_NAME**.

   Tip: If you can't find this field, use the **Find** field at the top of the Object Navigator.

2. Press F4 to display the Property Inspector for this field.
3. Under **Advanced Layout**, click the Format Trigger property field to display the PL/SQL Editor.
4. In the PL/SQL Editor, type the following code:

```
function F_COUNTRY_NAMEFormatTrigger return boolean is
pageNum number;
begin
    -- get current page number
    srw.get_page_num(pageNum);
    -- insert row into table
    insert into toc_example
    values (:country_name, pageNum);
    return (TRUE);
end;
```

   Note: You can also copy and paste this code from the provided file, formatorder_code.txt.

5. Click **Compile**.
6. Once the code is compiled, click **Close**, then close the Property Inspector.
7. Save your report.

34.2.4 Create a second query in the data model

The steps in this section will show you how to create a second query in the data model with a formula column that calculates the page range for the data under each country name. This query will fetch the information from the database table you created in Section 34.2.1, "Create a table in the database to hold the information for the TOC". You will later create a tabular layout to display this information in your table of contents.

To create a second query with a formula column:

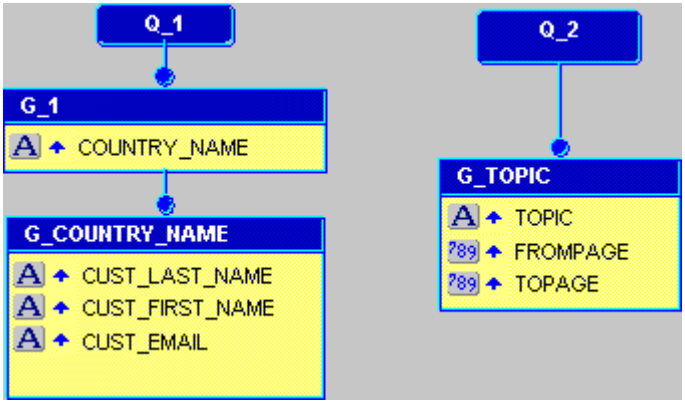
1. In the Data Model view, click the SQL Query tool in the tool palette, then click in the main area (canvas region) of the window to the right of query **Q_1**.
2. In the SQL Query Statement dialog box, type the SELECT statement for the query:

```
SELECT TOPIC, MIN(PAGE) FROMPAGE,
              MAX(PAGE) TOPAGE
FROM TOC_EXAMPLE
GROUP BY TOPIC
```

> **Note:** You can also copy and paste this code from the provided file, called `formatorder_code.txt`.

3. Click **OK**. Your data model should now look like this:

Figure 34–6 Data model with second query



4. In the Data Model view, click the Formula Column tool in the tool palette, then click in the group (**G_TOPIC**) for the second query.
5. While the formula column is selected, press F4 to display its Property Inspector.
6. Under **General Information**, set the Name property to `TOC_pages`, and the Datatype property to Character.
7. Under **Placeholder/Formula**, click the PL/SQL Formula property field to display the PL/SQL Editor.
8. In the PL/SQL Editor, use the template to enter the following code:

```
begin
  if :fromPage = :toPage then
  return (:fromPage);
  else
  return (:fromPage ||'-'|| :toPage);
  end if;
end;
```

Note: You can also copy and paste this code from the provided file, called `formatorder_code.txt`.

9. Click **Compile**.
10. Once the code is compiled, click **Close**.
11. Save the report.

34.2.5 Create a report block to display the table of contents

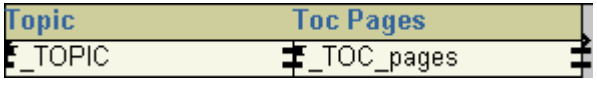
The steps in this section will show you how to create a tabular report block in the Header section of your report. This report block will display the headings in the table of contents (in this example, the country name), and the page range where the information can be found in the report.

To create a tabular report block in the Header section:

1. In the Paper Layout view, click the Header Section button in the toolbar.
2. Click the Report Block tool in the tool palette, then draw an area about 5 inches long and 1.5" inches long in the Paper Layout view. When you release the mouse button, the Report Block Wizard displays.
3. On the Style page, select **Tabular**, then click **Next**.
4. On the Groups page, click **G_TOPIC** and click **Down** to move it to the **Displayed Groups** list, then click **Next**.
5. On the Fields page, click each of the following fields in the **Available Fields** list, then click the right arrow (>) to move them to the **Displayed Fields** list, then click **Next**:
   - **TOPIC**
   - **TOC_pages**
6. On the Labels page, click **Next**.
7. On the Template page, click **Finish**.

Your report block displays in the Paper Layout view:

Figure 34–7 Paper Layout view of the Header Section



8. Save your report.

34.2.6 Run your simple table of contents report to paper

Click the Run to Paper button in the toolbar. The table of contents displays on the first page of the report, and looks like the following:

> **Note:** The report may take a few minutes to run. Also note that to generate the table of contents (TOC), you must click the Run to Paper button. If you click the Paper Design view button, the change of format order will not take effect, thus the TOC will not be generated.

Figure 34–8 Table of contents page of the report

Topic	Pages
Argentina	1-7
Australia	7-26
Brazil	26-45
Denmark	45-54
France	54-147
Germany	147-345
India	345-362
Ireland	362-411
Japan	411-426
Malaysia	426-440
New Zealand	440-446
Poland	446-462

When you navigate to, for example, page 1, you'll see the e-mail addresses for the customers in Argentina:

Figure 34–9 Results for customers in Argentina

Country Name Argentina		
Cust Last Name	**Cust First Name**	**Cust Email**
Grandy	Mandisa	Grandy@company.com
Eubank	Hattie	Eubank@company.com
Pearson	Idola	Pearson@company.com
Pardue	Bryant	Pardue@company.com
Elgin	Torrey	Elgin@company.com
Naber	Torrance	Naber@company.com
Markerman	Diana	Markerman@company.com
Burgess	Beulah	Burgess@company.com
Tanney	Woodrow	Tanney@company.com
Justice	Diamond	Justice@company.com
Gutierrez	Marcel	Gutierrez@company.com
Felton	Madelene	Felton@company.com

You can navigate to various pages in your report to view the data for each country.

Note: At this point, you can compare your report against the example file we've provided, `formatorder.rdf`. First, compile the PL/SQL by choosing **Program > Compile > All**, then run the report to paper.

For information on creating a multilevel table of contents, seeChapter 35, "Building a Paper Report with a Multilevel Table of Contents".

34.3 Create an index for your report

In this section, you will create an index that displays at the end of your report. This index will simply display the customer's name and the page number on which the

name appears. This indexing technique is useful when users want to find a specific customer's name, but are not sure what category (e.g., country) to reference.

Figure 34–10 Sample index for the paper report

Term	Page
Glidden, Lester	216
Capps, Lenora	216
Hartzog, Leah	216
Ross, Juan	216
Rodgers, Josh	216
Chinn, Jonathan	216
Lanston, Jason	216
Aubrey, Jamilah	216
Rowley, Jacqueline	216
Peebles, Jack	216
Lefevre, Rhoda	216
Bakker, Renita	216
Kkotzman, Renfred	216
Levy, Rendell	216
Baker, Regina	216
Goode, Regan	216
Newsome, Raphaela	216
Downey, Ransom	216
Gilmore, Rachel	217

34.3.1 Create a table in the database to hold the information for the index

The steps in this section will show you how to create a table in the database that will hold the page numbers for the items you want to list in the index. If you are not sure if you can create a table in the database, contact the database administrator.

To create a table in the database:

1. Follow the steps in Section 34.2.1, "Create a table in the database to hold the information for the TOC"to create a new table in the database, using the following code:

```
create table index_example (term varchar2(100), page number);
```

2. Press Enter.

   You should see a notification that the table has been created.

3. Exit SQL*Plus.

34.3.2 Create a format trigger

The steps in this section will show you how to create a format trigger based on the customer's last name. You will use the SRW.GET_PAGE_NUM built-in function to find the page number for the current customer, and insert that page number into the table you created in the previous section.

To create a format trigger:

1. In Reports Builder, open the report, `formatorder_<your initials>.rdf`.
2. Click the Paper Layout button in the toolbar to display the Paper Layout view.
3. In the Paper Layout view, right-click the field **F_CUST_LAST_NAME**, then choose **Property Inspector** from the pop-up menu.
4. Under **Advanced Layout Properties**, click the Format Trigger property field to display the PL/SQL Editor.
5. In the PL/SQL Editor, use the template to enter the following code:

```
function F_CUST_LAST_NAMEFormatTrigger return boolean is
  PageNum number;
begin
  -- get pagenumber
  srw.get_page_num(pageNum);
  -- insert into table
  insert into index_example
  values (:Cust_last_name||', '||:Cust_first_name, PageNum);
  return (TRUE);
end;
```

Note: You can enter this code by copying and pasting it from the provided text file called `formatorder_code.txt`.

6. Click **Compile**.

7. If there are compilation errors, match your code to the code we've provided (either in the example RDF file or in this chapter), then compile it again.
8. Once there are no compilation errors, click **Close**.
9. Save your report as `formatorder_index_<your initials>.rdf`.

34.3.3 Add a query to the data model

The steps in this section will show you how to add a query to your data model that will retrieve the individual customer names and the page numbers for your index.

To add the query:

1. In the Data Model view, click the SQL Query tool in the tool palette, then click in the main area (canvas region) of the window to the right of query **Q_1**.
2. In the SQL Query Statement dialog box, type the SELECT statement for the query:

```
SELECT SUBSTR(TERM,1,1) INITIAL_LETTER, TERM, PAGE FROM INDEX_EXAMPLE
ORDER BY TERM
```

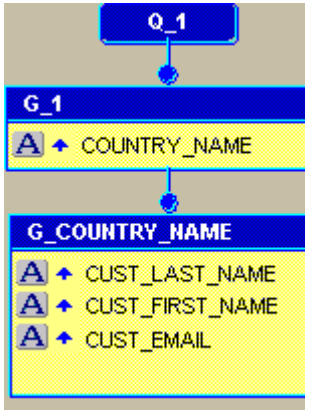
Note: You can enter this query in any of the following ways:

- Copy and paste the code from the provided text file called `formatorder_code.txt` into the **Data Source definition** field.
- Click **Query Builder** to build the query without entering any code manually.
- Type the code in the **Data Source definition** field.

---

3. Click **OK**.
4. In the new query that displays in the Data Model view, click and drag the **INITIAL_LETTER** column above the rest of the query, so that the data model now looks like this:

Figure 34–11 Data model view of the index



5. Save your report.

34.3.4 Create a report block to display the index in the Trailer section

The steps in this section will show you how to display the index information in the Trailer section of your report.

1. In the Paper Layout view, click the Trailer Section button in the toolbar.
2. Choose **Insert > Report Block**.
3. On the Style page, select the **Group Above** radio button, then click **Next**.
4. On the Groups page, click **G_2** (the name of the new group that contains the **INITIAL_LETTER** column), then click **Down** to specify the print direction.
5. In the **Available Groups** list, click **INITIAL_LETTER**, then click **Down**.
6. Click **Next**.
7. On the Fields page, click **INITIAL_LETTER**, then click > to move it to the **Displayed Fields** list.
8. Move **TERM** and **PAGE** to the Displayed Fields list.
9. Click **Next**, then **Next** again, then click **Finish**.
10. Save your report.

34.3.5 Run your report to paper

Click the Run to Paper button in the toolbar. The index displays on the last pages of the report. We chose page 2402, which (in our example) looks like the following:

> **Note:** The report may take a few minutes to run. Also note that to generate the table of contents (TOC), you must click the Run to Paper button. If you click the Paper Design view button, the change of format order will not take effect, thus the TOC will not be generated.

Figure 34–12 Index page of the report

Term	Page
Glidden, Lester	216
Capps, Lenora	216
Hartzog, Leah	216
Ross, Juan	216
Rodgers, Josh	216
Chinn, Jonathan	216
Lanston, Jason	216
Aubrey, Jamilah	216
Rowley, Jacqueline	216
Peebles, Jack	216
Lefevre, Rhoda	216
Bakker, Renita	216
Kkotzman, Renfred	216
Levy, Rendell	216
Baker, Regina	216
Goode, Regan	216
Newsome, Raphaela	216
Downey, Ransom	216
Gilmore, Rachel	217

Note: The data on page 2402 in your report may not appear the same as ours, depending on the amount and type of data you have in the schema.

34.4 Summary

Congratulations! You have successfully created a report with a table of contents. You now know how to:

- create a table of contents (TOC) and index for a large paper report.
- use the SRW.SET_FORMAT_ORDER built-in procedure to change the order in which the sections of a report are executed.
- use the SRW.GET_PAGE_NUM built-in package to retrieve the page number for a specific record.
- create a new table in your database to hold the necessary page numbers.
- create a simple group above report using the Report Wizard.
- manually a simple data model to retrieve the information for your report.
- create a format trigger to generate the necessary page numbers for an index.
- use the Report Block Wizard to create a group report that displays the index in the trailer of a report.

To learn how to add hyperlinks to your table of contents, refer to the examples in the Getting Started with Oracle Reports Web site, located on the Oracle Technology Network (`http://otn.oracle.com/products/reports/`).

For more information on any of the wizards, views, or properties used in this example, refer to the *Reports Builder Online Help*, which you can access in two ways:

- From the Oracle Technology Network (`http://otn.oracle.com/products/reports/`), click **Documentation** and navigate to the *Reports Builder Online Help* for the most recent, hosted online help.
- From Reports Builder, choose **Help > Help Contents**.

Note: You can replace your Reports Builder online help with the most recent update by downloading the latest online help set available on the Oracle Technology Network (`http://otn.oracle.com/docs/products/reports/`). Instructions for replacing your help file are included in the readme.txt in the download file.

35

Building a Paper Report with a Multilevel Table of Contents

Figure 35–1 Multilevel table of contents

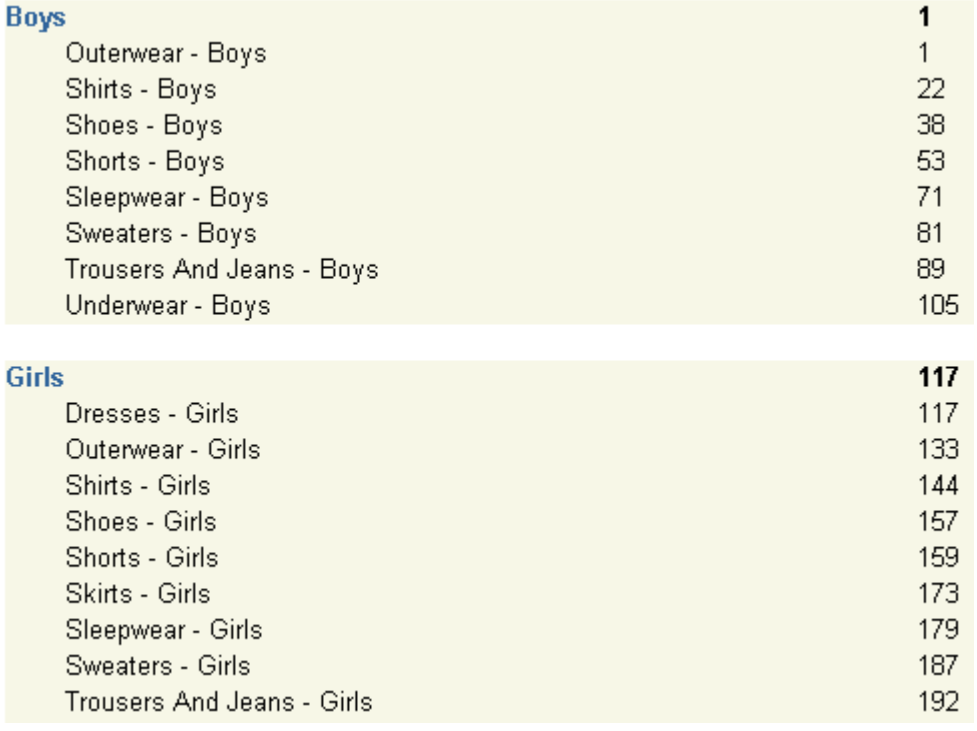
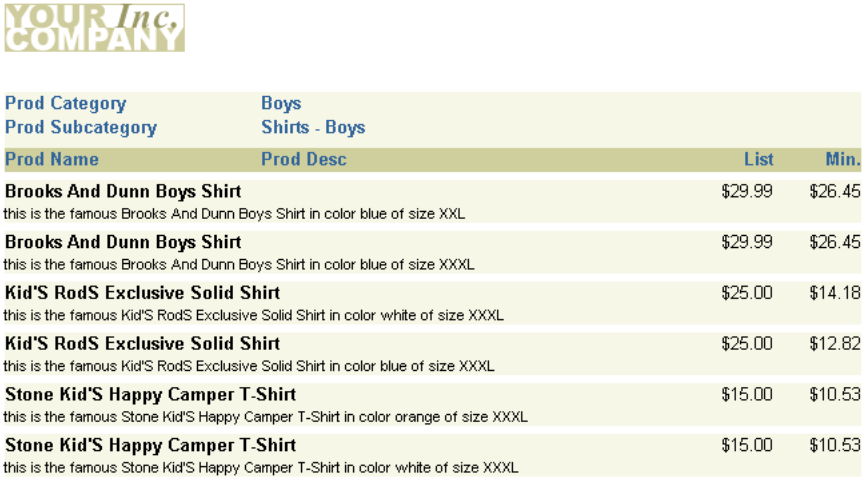


Figure 35–2 Main category with sub-category sample page in the report



When you create a large paper report, you can create a table of contents (TOC) to make it easier for users to navigate through the pages of information. In the previous chapter, you learned how to create a basic table of contents that categorized customer information by country, then listed the country and page ranges in the table of contents. You also learned how to index this information.

In this chapter, you will learn how to create a multilevel table of contents that will enable you to create a hierarchy. This hierarchy will further enable you to categorize your information, so that users can find a specific piece of information. For information on creating a simple table of contents, refer to Chapter 34, "Building a Paper Report with a Simple Table of Contents and Index".

As in the previous chapter, you will use the SRW.SET_FORMAT_ORDER built-in procedure in an After Parameter Form trigger to generate the Main section of your report before the Header and Trailer sections. Doing so enables you to generate the page numbers for the report in the Main section, then display the page numbers in the table of contents in the Header section.

Note: Although you can use SRW.SET_FORMAT_ORDER to change the order in which the report is formatted, the report will still display the Header section first, then the Main section, and finally the Trailer section. You can use SRW.SET_FORMAT_ORDER to simply change the formatting order, but not the display order of the report sections.

You will also create a format trigger based on a sub-category in your report that will use the SRW.GET_PAGE_NUM built-in package to retrieve the page number for each record and temporarily place it in a database table you create. This format trigger will also get the page numbers for each sub-category that you specify. Finally, you will use a SQL query to display the desired information in a multilevel table of contents that displays at the beginning of your report.

For more information on the SRW built-in packages in Oracle Reports, refer to the *Reports Builder Online Help*.

This chapter assumes that you know how to create a basic data model and layout for your report, and focuses on the building of the table of contents. If you are not sure how to create a data model and paper layout, refer to the *Reports Builder Online Help* or, for specific steps, refer to any other chapter in this manual.

Example Scenario

Suppose you have a large paper catalog that lists all the clothing products a company sells. This catalog simply lists all the items in a simple report. Since the product line has become much larger, the catalog report now exceeds 700 pages. You will create a multilevel table of contents that sorts the clothing by department (e.g., Boys, Girls), then lists each item under each category.

Table 35–1 Features Demonstrated in this example

Feature	Location
Use SQL*Plus to create a table in the database to hold the first and second tier categories and page numbers.	Section 35.2, "Create a table in the database to hold the TOC data"

Table 35–1 Features Demonstrated in this example

Feature	Location
Use the PL/SQL Editor to create two triggers that change the order in which the report sections are executed and fetch the page numbers for each category, sub-category, and record.	Section 35.3, "Create an After Parameter Form trigger and format trigger"
Use the Report Block Wizard to create a simple layout in the Header section of the report layout.	Section 35.4, "Create a layout for the table of contents"

35.1 Prerequisites for this example

To build the examples in this manual, you must have the example files we've provided, as well as access to the sample schema that is shipped with the Oracle9*i* database.

35.1.1 Example files

If you haven't already done so, you can download the files you'll need to complete this example from the Oracle Technology network (`http://otn.oracle.com/products/reports/`) and install them on your machine.

To download and install the example files:

1. Go to the Oracle Technology Network Web site (`http://otn.oracle.com/products/reports/`).
2. Click **Getting Started with Oracle Reports**.
3. Click **Index**, then browse through the list of examples and find the "Building a Report with a Table of Contents and Index" example.
4. Download the file `formatorder.zip` into a temporary directory on your machine (e.g., `d:\temp`).
5. Unzip the contents of the file, maintaining the directory structure, into an examples directory on your machine (e.g., `d:\orawin90\examples`).

   This zip file contains the following files:

Table 35–2 Files necessary for building a report with a table of contents

File	Description
`Examples\MultiLevelTOC\source\multilevel_source.rdf`	The source file that contains a basic paper layout and data model for your report.
`Examples\MultiLevelTOC\result\multilevel_toc.rdf`	The final RDF version of the report with a multilevel table of contents.
`Examples\MultiLevelTOC\scripts\multilevel_code.txt`	The various SQL statements you will use in this report.

Note: The `index.html` file and `assets` directory are used as part of the **Getting Started with Oracle Reports** Web site. Please do not delete or move these files

35.1.2 Access to the sample schema

If you don't know if you have access to the sample schema provided with the Oracle9*i* database, contact your database administrator. You should have access to the "Sales History" portion of the schema to complete this example. Typically, you can log into this schema with the user ID and password "`sh/sh`", then enter the name of the database.

35.2 Create a table in the database to hold the TOC data

The steps in this section will show you how to create a table in the database that will hold the page numbers for the main and sub-categories in the report. If you are not sure if you can create a table in the database, contact the database administrator.

To create a table in the database:

1. Create a new table in the Sales History database schema, using the following code:

```
create table toc_multilevel (main_topic varchar2(100), sub_topic
varchar2(100), page number);
```

Tip: To create a table, use SQL*Plus and connect to the sample schema provided with the Oracle9*i* database called "Sales History." Typically, you connect to this database using the user ID and password "`sh/sh`." If you do not have access to this schema, contact your database administrator.

2. Press Enter.
3. You should see a notification that the table has been created.
4. Exit SQL*Plus.

35.3 Create an After Parameter Form trigger and format trigger

The steps in this section will show you how to create two triggers. The first trigger will use the SRW.SET_FORMAT_ORDER built-in procedure to format the Main section of the report first, then the Header and Trailer sections. The second trigger will fetch the page number for each category, sub-category, and record, and place that information into the table you created in Section 35.2, "Create a table in the database to hold the TOC data".

35.3.0.1 Create an After Parameter Form trigger

An After Parameter Form trigger is a function that executes after the Parameter Form is executed. In this section, you will set the trigger to change the order of the report execution, so that the Main section formats first.

To create an After Parameter Form trigger:

1. In Reports Builder, open the provided file called `multilevel_source.rdf` in the `source` directory.
2. In the Object Navigator, under your report name (**MULTILEVEL_SOURCE**), expand the **Report Triggers** node.
3. Right-click **AFTER PARAMETER FORM**, then choose **PL/SQL Editor** from the pop-up menu.
4. In the PL/SQL Editor, use the template to enter the following code:

```
function AfterPForm return boolean is
begin
  srw.set_format_order(srw.main_section, srw.header_section,
```

```
srw.trailer_section);
  return (TRUE);
end;
```

Note: You can also copy and paste the code from the provided file, `multilevel_code.txt`.

5. Click **Compile**.
6. Once the code has compiled, click **Close**.
7. Save your report as `multilevel_<your initials>.rdf`.

35.3.0.2 Create a format trigger

In this section, you will create a format trigger based on the field F_PROD_SUBCATEGORY. This field displays the sub-category, which is the product name. This sub-category falls under the main category, which is the product department. This format trigger will fetch the page number for each product name, so that the table of contents will enable the user to navigate to various parts of the report based on the department name, then the product name.

To create a format trigger:

1. In the Object Navigator, under the **Paper Layout** node for your report, navigate to **Main Section > Body > M_G_PROD_CATEGORY_GRPFR > R_G_PROD_CATEGORY > M_G_PROD_SUBCATEGORY_GRPFR > R_G_PROD_SUBCATEGORY_GPRFR**, then click **F_PROD_SUBCATEGORY**.

   Tip: If you can't find this field, use the **Find** field at the top of the Object Navigator.

2. Press F4 to display the Property Inspector for this field.
3. Under **Advanced Layout**, click the Format Trigger property field to display the PL/SQL Editor.
4. In the PL/SQL Editor, use the template to enter following code:

```
function F_PROD_SUBCATEGORYFormatTrigge return boolean is
  PageNum number;
```

```
  myCount number;
begin
  -- get page number
  srw.get_page_num(pageNum);
  -- check table for duplicates
  select count(*)
  into myCount
  from toc_multilevel
  where main_topic = :PROD_CATEGORY and
  sub_topic = :PROD_SUBCATEGORY;
  -- if no duplicates, insert row
  if myCount = 0 then
  insert into toc_multilevel
  values (:PROD_CATEGORY, :PROD_SUBCATEGORY, pageNum);
  end if;
  return (TRUE);
end;
```

Note: You can also copy and paste this code from the provided file, `multilevel_code.txt`.

5. Click **Compile**.
6. Once the code is compiled, click **Close**, then close the Property Inspector.
7. Save your report.

35.3.1 Create a second query in the data model

The steps in this section will show you how to create a second query in the data model that retrieves the information stored in the table you created in Section 35.2, "Create a table in the database to hold the TOC data". This query will fetch the data necessary for generating the table of contents in the Header section of your report.

To create a query:

1. In the Data Model view, click the SQL Query tool in the tool palette, then click in the main area (canvas region) of the window to the right of query **Q_1**.
2. In the SQL Query Statement dialog box, type the SELECT statement for the query:

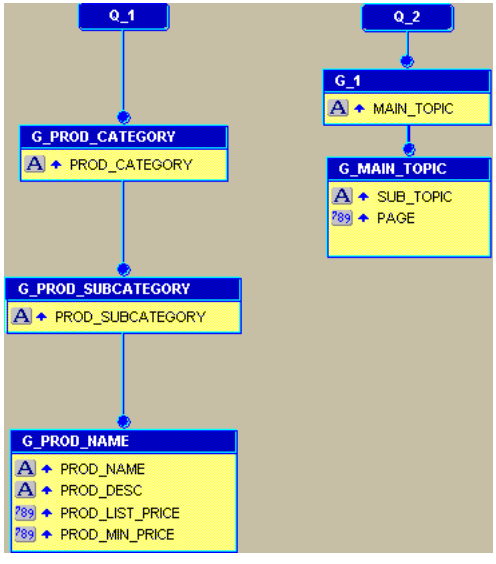
```
SELECT * FROM TOC_MULTILEVEL
```

Note: You can also copy and paste this code from the provided file, called `multilevel_code.txt`.

3. Click **OK**.
4. In the new query, click and drag the **MAIN_TOPIC** column to a separate group above the rest of the query.
5. Double-click the new summary column object to display the Property Inspector, and set properties:
   - Under **General Information**, set the Name property to `Subcategory_Page`
   - Under **Summary**, set the Function property to **Minimum** and set the Source property to **PAGE**

   Your data model should now look like this:

Figure 35–3 Data model for the multilevel TOC report



6. Save your report.

35.4 Create a layout for the table of contents

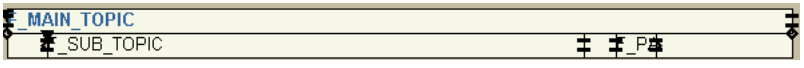
To display the TOC in your report, you will need to create a layout in the Header section of the report. In doing so, the multilevel TOC displays at the beginning of your report. You will also add a field to your layout to display the page number for each category.

To create a layout for the TOC:

1. In the Paper Layout view, click the Header Section button in the toolbar.
2. Click the Report Block tool in the tool palette, then draw an area about 6 inches long and 1.5" inches long. Release the mouse button to display the Report Block Wizard.
3. In the Report Block Wizard, on the Style page, select **Group Above**, then click **Next**.
4. On the Groups page, click **G_MAIN_TOPIC**, then click **Down** to move it to the **Displayed Groups** list.
5. In the Available Groups list, click **G_1**, then click **Down** to move it to the **Displayed Groups** list.
6. Click **Next**.
7. On the Fields page, use **>** to move the **MAIN_TOPIC**, **SUB_TOPIC**, and **PAGE** fields to the Displayed Fields list, then click **Next**.
8. On the Labels page, click **Next**.
9. On the Template page, click **Finish**.

Your report block displays in the Paper Layout view:

Figure 35–4 Paper Layout view of the Header Section



10. Click the Field tool in the tool palette and create a field above the **PAGE** field, next to the F_MAIN_TOPIC field.
11. Double-click the new field object to display the Property Inspector, and set properties:
    - Under **Field**, set the Source property to **Subcategory_Page**, then click the Page Numbering property field to display the Page Numbering dialog box:
    - In the **Reset at** list, click **G_1**, and verify that the **Start at** and **Increment by** values are both **1**.
12. Save your report.

35.4.1 Run your multilevel table of contents report to paper

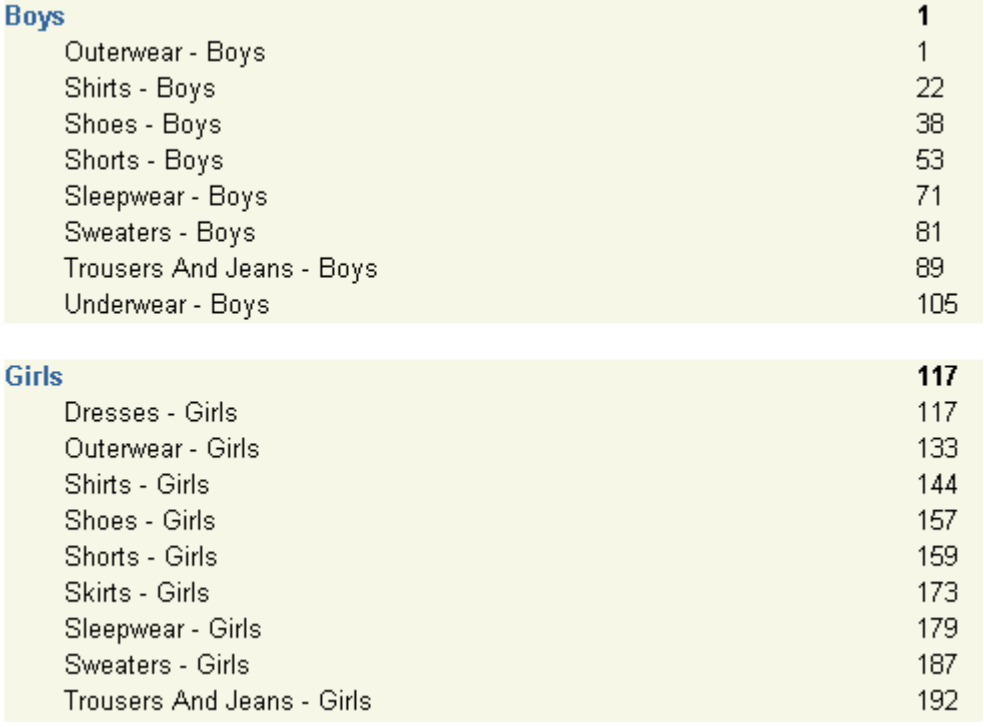
The steps in this section will show you how to run your report and make a few modifications in the Paper Design view.

1. In the Paper Layout view, click the Run to Paper button in the toolbar.
2. In the Paper Design view, change the font of the field (**F_1**) to **Arial, Bold**.
3. Change the Fill color to **Beige**, and the Line Color to **No Line Color**.

   The table of contents displays on the first page of the report, and looks like the following:

> **Note:** The report may take a few minutes to run. Also note that to generate the table of contents (TOC), you must click the Run to Paper button. If you click the Paper Design view button, the change of format order will not take effect, thus the TOC will not be generated.

Figure 35–5 Table of contents page of the report



When you move to, for example, page `30` (which is between the page range of 22 and 38), you'll see the SHIRTS sub-category under the BOYS product department:

Figure 35–6 Results for the BOYS product department and SHIRTS sub-category

Prod Category	Boys		
Prod Subcategory	Shirts - Boys		
Prod Name	**Prod Desc**	**List**	**Min.**
Brooks And Dunn Boys Shirt	this is the famous Brooks And Dunn Boys Shirt in color blue of size XXL	$29.99	$26.45
Brooks And Dunn Boys Shirt	this is the famous Brooks And Dunn Boys Shirt in color blue of size XXXL	$29.99	$26.45
Kid'S RodS Exclusive Solid Shirt	this is the famous Kid'S RodS Exclusive Solid Shirt in color white of size XXXL	$25.00	$14.18
Kid'S RodS Exclusive Solid Shirt	this is the famous Kid'S RodS Exclusive Solid Shirt in color blue of size XXXL	$25.00	$12.82
Stone Kid'S Happy Camper T-Shirt	this is the famous Stone Kid'S Happy Camper T-Shirt in color orange of size XXXL	$15.00	$10.53
Stone Kid'S Happy Camper T-Shirt	this is the famous Stone Kid'S Happy Camper T-Shirt in color white of size XXXL	$15.00	$10.53

You can navigate to various pages in your report to view the data for each department and sub-category.

Note: At this point, you can compare your report against the example file we've provided, `multilevel_toc.rdf`. First, compile the PL/SQL by choosing **Program > Compile > All**, then run the report to paper.

35.5 Summary

Congratulations! You have successfully created a report with a table of contents You now know how to:

- create a multilevel table of contents for a large paper report.
- Use the SRW.SET_FORMAT_ORDER built-in procedure to change the order in which the report sections are formatted
- create a format trigger that fetches the page numbers for categories and subcategories that you specify.
- use the Report Block Wizard to create a simple group above report layout to display your table of contents.

To learn how to add hyperlinks to your table of contents, refer to the examples in the Getting Started with Oracle Reports Web site, located on the Oracle Technology Network (`http://otn.oracle.com/products/reports/`).

For more information on any of the wizards, views, or properties used in this example, refer to the *Reports Builder Online Help*, which you can access in two ways:

- From the Oracle Technology Network (`http://otn.oracle.com/products/reports/`), click **Documentation** and navigate to the *Reports Builder Online Help* for the most recent, hosted online help.
- From Reports Builder, choose **Help > Help Contents**.

> **Note:** You can replace your Reports Builder online help with the most recent update by downloading the latest online help set available on the Oracle Technology Network (`http://otn.oracle.com/docs/products/reports/`). Instructions for replacing your help file are included in the readme.txt in the download file.

36

Bursting and Distributing a Report

Oracle Reports enables you to deliver a single report to multiple destinations simultaneously. By taking advantage of this feature, you can create a single report, then send it in any format (e.g., PDF or HTML) to multiple destinations.

In this example, you will modify a simple report we've provided to burst each section to a separate report. You will then modify a sample distribution XML file to send an e-mail to each destination with an attachment based on the separate reports. You will also send multiple e-mails to the same e-mail address with a single attachment (the entire report).

About bursting and distribution

Oracle Reports enables you to deliver a single report to multiple destinations simultaneously. Using the new, enhanced distribution feature, you can set up your report to be distributed to an e-mail destination, a portal, a printer, or anywhere else when the report is run. This feature also enables you to improve performance, since you only fetch the data once for many different formats and destinations. Using distribution also reduces your maintenance overhead because you only need one job request to publish the report to multiple destinations. You can refine this further by sending the Header section to some recipients, the main to others, and the entire report to an entirely different recipient list.

For more information on bursting and distribution, see *Oracle Application Server Reports Services Publishing Reports to the Web* manual located on the Oracle Technology Network (http://otn.oracle.com) and on the documentation CD provided with OracleAS.

Example Scenario

Suppose you are the report developer for a manufacturing company who needs to deliver monthly information to its warehouses. In this example, you will modify an

existing report to burst on sections, based on each warehouse ID, creating separate PDF reports for each warehouse. You will then edit the distribution XML file we've provided to e-mail each section as an attachment to individual warehouses.

Table 36–1 Features demonstrated in this example

Feature	Location
Modify an existing report (or one that you created with the Report Wizard) for bursting on a section.	Section 36.2, "Set up an existing report for bursting"
Modify the XML definition file for distribution.	Section 36.3, "Edit the distribution XML definition"
Run the report to PDF format so that the report is e-mailed to the destinations defined in the `distribution.xml` file.	Section 36.4, "Run the report"

36.1 Prerequisites for this example

To build the examples in this manual, you must have the example files we've provided, as well as access to the sample schema that is shipped with the Oracle9*i* database.

36.1.1 Example files

If you haven't already done so, you can download the files you'll need to complete this example from the Oracle Technology network (`http://otn.oracle.com/products/reports`) and install them on your machine.

To download and install the example files:

1. Go to the Oracle Technology Network Web site (`http://otn.oracle.com/products/reports`).
2. Click **Getting Started with Oracle Reports**.
3. Click **Index**, then find the "Bursting and Distributing a Report" example.
4. Download the file `Distribution.zip` into a temporary directory on your machine (e.g., `d:\temp`).

5. Unzip the contents of the file, maintaining the directory structure, into an examples directory on your machine (e.g., `d:\orawin90\examples`).

   This zip file contains the following files:

Table 36–2 Files necessary for building the sample report for bursting and distribution

File	Description
`Examples\Distribution\source\distribution.xml.`	The XML file that controls the distribution properties for your report.
`Examples\Distribution\source\inventory_report_dist.rdf`	The report you will distribute.
`Examples\Distribution\result\inventory_report_dist.rdf.`	The report you will burst.
`Examples\Distribution\result\REP_*.pdf`	The PDFs that are generated when you distribute and burst your report.

Note: You can save the `distribution.xml` file we provide and reuse it later, so you don't have to create another one from scratch.

36.1.2 Access to the sample schema

If you don't know if you have access to the sample schema provided with the Oracle9*i* database, contact your database administrator. You should have access to the "Order Entry" portion of the schema to complete this example.

36.2 Set up an existing report for bursting

For the purposes of this chapter, we've provided you with an RDF file you can use for bursting. When you open this report, you should also connect to the sample schema in your database by choosing **File > Connect**. If you don't know the connection information for the "Order Entry" portion of the schema, contact your database administrator.

In this section, you will set up a repeating frame so that the data bursts on each warehouse ID. This way, you can later send the data resulting from each section as a report to each individual warehouse.

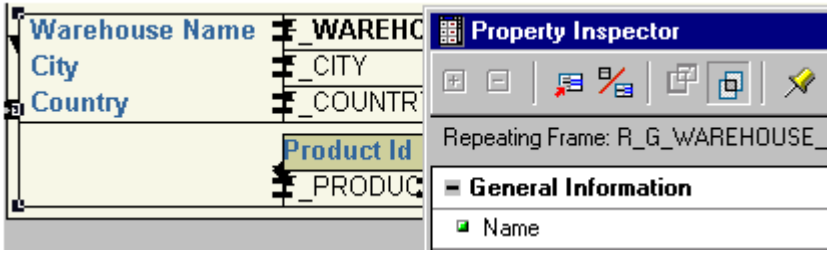
To set up a repeating frame for bursting:

1. In Reports Builder, choose **File > Open**.
2. Navigate to the directory where your examples source files are located, and open the file `inventory_report_dist.rdf`.

   The report displays in the Object Navigator.
3. In the Object Navigator, under the report name, double-click the view icon next to the **Paper Layout** node to display the Paper Layout view.
4. In the Paper Layout view, select the outermost repeating frame (R_G_WAREHOUSE_ID), then delete it.

   Tip: You can select the **R_G_WAREHOUSE_ID** in the Object Navigator to simultaneously select it in the Paper Layout view, where you can delete it. Do not delete the repeating frame in the Object Navigator, as you will delete all the objects within the frame, as well.

Figure 36–1 Deleting the repeating frame



Note: You must delete the existing repeating frame because you will modify the report to burst on each section. Later, you will distribute each section to each warehouse e-mail ID

For more information on frames and repeating frames, refer to the *Reports Builder online help.*

5. In the Object Navigator, right-click the **Main Section** node, then choose **Property Inspector** to display the Property Inspector for the Main Section:
   - Under **Section**, set the Repeat On property and to G_WAREHOUSE_ID.

6. Save the report as `inventoryreport_dist_<your initials>.rdf`.

You have set up your report to burst based on the warehouse ID.

36.3 Edit the distribution XML definition

The Oracle Reports distribution XML file enables you to specify the details of your distribution. For example, if you're distributing via e-mail, you can specify such details as the addressee, the reply to address, and the subject.

In this section, we'll show you how to modify a distribution XML file. We've indicated locations where you need to enter your own information to make the distribution work. Note that you can save this `distribution.xml` file to use later, so that you don't have to manually create another file every time you want to use distribution. Our sample file also contains comments that might come in handy later.

When you want to distribute a report, you need to either:

- Make sure your source report (e.g., `inventoryreport_dist.rdf`) and your distribution XML file (e.g., `distribution.xml`) are in the same directory.

  OR

- When you run the report from the Reports Server, set the destination to the path of the XML file. (We'll explain this in Section 36.4, "Run the report".)

We've provided both these files in a single directory: *`Examples`*`/Distribution/source/`.

For more information on distribution, see the *Oracle Application Server Reports Services Publishing Reports to the Web* manual available on the Oracle Technology Network (`http://otn.oracle.com`).

To edit the distribution XML file:

1. In a text editor, such as Notepad, open the file we've provided called `distribution.xml`.
2. Find the placeholder text we've provided: `<YourFilePath>`, and replace it with the location of where your resulting PDFs will be stored.

   Example: Replacing the placeholder text with:

   ```
   d:\temp\
   ```

   changes the path to:

```
d:\temp\Rep_%amp;&lt;city&gt;.pdf
```

Using this complete path would place the resulting PDF files in your `d:\temp` directory.

3. Find the placeholder text we've provided: `<Origine-mailAddress>`, and replace it with the sender's e-mail address.
4. Perform step 3 for all instances of the placeholder text: `<Origine-mailAddress>`.
5. Find the placeholder text we've provided: `<Destinatione-mailAddress>`, and replace it with the first recipient's e-mail address.
6. Perform step 5 for all instances of the placeholder text: `<Destinatione-mailAddress>`.

> **Note:** For this example, we do not show you how to send multiple e-mails at once, as we do not supply built-in e-mail addresses with Reports Builder. However, if you wanted to send the report to various e-mail destinations, you would need to create a recipient field in your data model. Then, in the `ex2` section of the distribution.xml, replace the placeholder text `<Destinatione-mailAddress>` with `&<recipient>`. You can then delete the first section of the `distribution.xml` file (marked `ex1`).

7. Save the XML file to the same directory where you've saved `inventoryreport_dist_<your initials>.rdf`.

> **Note:** It is not required that you save the XML file to the same directory where your RDF is located, as you can specify the destination of the XML file at runtime.

You have finished customizing the distribution XML file to send a single e-mail to corporate headquarters with all of the individual warehouse reports, and multiple e-mails with a single attachment each to the individual warehouses.

36.4 Run the report

Since this example is a Web report, you can distribute it from either the command line or using your Web browser. For both methods, the following parameters apply:

<YourPath> is where your RDF file is located and where your `distribution.xml` file is located. Your login ID and Reports server name is your login information for the sample schema you've used with the sample RDF.

From the command line

Type the following text on the command line:

```
RWRUN REPORT=<YourPath>/inventoryreport_dist_<YourInitials> USERID=<Your Login
ID> SERVER=<Your Server Name> DISTRIBUTE=YES
DESTINATION=<YourPath>/distribution.xml
```

Usage notes

- In the above command line, the parameter "SERVER=<Your Server Name>" is optional. If you do not specify a server name, Oracle Reports will use the default server.
- The SMTP mail server is set up during the installation process. If you did not specify a server during installation, you can set up the mail server manually before distributing your report to an email destination. You specify the outgoing mail server using **pluginParam** in the server config file.

  EXAMPLE:
  ```
  <pluginParam name="mailServer">smtp01.mycorp.com</pluginParam>
  ```

  For more information on setting up the mail server, refer to the chapter "Configuring Oracle Reports Services" in the *Oracle Application Server Reports Services Publishing Reports to the Web* manual, Release 10*g* (9.0.4).

Using a URL

Type the following text in the **Location** field of your browser:

```
../RWSERVLET?REPORT=<YourPath>/inventoryreport_dist_<YourInitials>&USERID=<Your
Login ID>&SERVER=<Your Server
Name>&DISTRIBUTE=YES&DESTINATION=<YourPath>/distribution.xml
```

Running the report creates a file for each warehouse based on the warehouse ID in the specified directory. When the report is distributed, a single e-mail is sent to one address with all of these files attached to the e-mail.

If you set up the `distribution.xml` file for multiple e-mail addresses, each warehouse (or each e-mail address) would be sent a single e-mail with a single attachment file that includes the report for that warehouse.

36.5 Summary

Congratulations! You have distributed a report. You now know how to:

- modify the layout of an existing report to burst on a section in your layout.
- distribute a report using e-mail by modifying the `distribution.xml` file.

For more information on any of the wizards, views, or properties used in this example, refer to the *Reports Builder Online Help*, which you can access in two ways:

- From the Oracle Technology Network (`http://otn.oracle.com/products/reports/`), click **Documentation** and navigate to the *Reports Builder Online Help* for the most recent, hosted online help.
- From Reports Builder, choose **Help > Help Contents**.

Note: You can replace your Reports Builder online help with the most recent update by downloading the latest online help set available on the Oracle Technology Network (`http://otn.oracle.com/docs/products/reports/`). Instructions for replacing your help file are included in the readme.txt in the download file.

Part VI

Building Reports with PL/SQL and Java

The chapters in this Part provide steps to build reports that use PL/SQL code or Java. You can include PL/SQL, PL/SQL libraries, ref cursors, and barcodes in your reports, and build JSP-based Web reports that include a Parameter Form.

- Chapter 37, "Building a PL/SQL Report"

  A PL/SQL report uses an external PL/SQL library and PL/SQL within a report to modify formatting and to perform calculations on column values. For example, you can use a PL/SQL procedure in a format trigger to include a space between records similar to a break report or to calculate the total compensation for each employee.

- Chapter 38, "Building a Paper Report with Ref Cursors"

  A paper report with ref cursors helps manage queries. (A ref cursor is a PL/SQL cursor datatype that you can reference from within a PL/SQL query.) For example, if you already have numerous queries built and you want to reuse those queries in your reports, you can use a ref cursor in your report data model to access those queries.

- Chapter 39, "Building a Simple Parameter Form for a JSP-based Web Report"

  A JSP-based Web report can be modified by adding a Parameter Form to accept user input at runtime that will determine what data will display in the report.

- Chapter 40, "Building a Report with a Barcode"

  A barcode can be added to a paper or Web report by using the barcode JavaBean. In Reports Builder, the barcode JavaBean automatically generates a barcode for you based on the data you specify. For example, you can use the barcode to track shipping orders by generating a barcode based on the shipping order code.

37

Building a PL/SQL Report

Figure 37–1 PL/SQL report output

Name	Job Id	Commission Percent	Salary	Bonus	Total Compensation
Adam Fripp	ST_MAN		$8,200.0	$1,230.0	$9,430.0
Alana Walsh	SH_CLERK		$3,100.0	$465.0	$3,565.0
Alberto Errazuriz	SA_MAN	.3	$12,000.0	$1,800.0	$13,800.0
Alexander Hunold	IT_PROG		$9,000.0	$1,350.0	$10,350.0
Alexander Khoo	PU_CLERK		$3,100.0	$465.0	$3,565.0
Alexis Bull	SH_CLERK		$4,100.0	$615.0	$4,715.0
Allan McEwen	SA_REP	.35	$9,000.0	$1,350.0	$13,500.0
Alyssa Hutton	SA_REP	.25	$8,800.0	$1,320.0	$12,320.0
Amit Banda	SA_REP	.1	$6,200.0	$930.0	$7,750.0
Anthony Cabrio	SH_CLERK		$3,000.0	$450.0	$3,450.0

In this chapter, you will learn how to use an external PL/SQL library and PL/SQL within a report to modify formatting and calculate the total compensation for each employee. In the figure above, notice the spacing between records (e.g., between the record for Alexander Khoo and the record for Alexis Bull). This space is not due to a break; it is the result of using a PL/SQL procedure in a format trigger.

Concepts

- There are a variety of ways to incorporate PL/SQL into your reports. You've already create formula columns that used simple PL/SQL expressions to compute their values, and format triggers that used PL/SQL to conditionally determine the formatting of mailing labels. Here, you will create external libraries and local functions and procedures.

- External PL/SQL libraries are modules that contain named PL/SQL functions and procedures. They may be stored either in the database or in a file, and can be referenced from not only any report, but from other Oracle products. External libraries eliminate the need to re-enter commonly-used PL/SQL constructs, whether in reports, forms, or graphs. This, in turn, eliminates the problem of maintaining several versions of the same PL/SQL code.
- Local PL/SQL consists of named PL/SQL functions and procedures that are saved in a report definition. Local PL/SQL may be referenced only by objects within the report (e.g., group filters, formula columns, format triggers, etc.). However, the usefulness of storing PL/SQL in a single location still applies.

Data Relationships

- This report uses one query. You will add a function stored in an external library, a report-level function, two formula columns, and a parameter governing the number of records to display before inserting a space.

Layout

- This report uses the tabular layout style, with minor modifications.

Example Scenario

This chapter will show you how to manually create a query in the Data Model view, then modify the layout of the report in the Paper Layout view. You will create formula columns, a summary column, and a format trigger that uses a summary column and a user parameter.

To see a sample PL/SQL report, open the examples folder named `plsql`, then open the Oracle Reports example named `plsql.rdf`. For details on how to open it, see "Accessing the example reports" in the Preface.

Table 37–1 Features demonstrated in this example

Feature	Location
Create a new PL/SQL library that you will use in this report.	Section 37.2, "Create a new PL/SQL library"
Manually create a query and add formula columns to your report that calculate bonuses and total compensation based on an external PL/SQL library.	Section 37.3, "Create the report definition"

Table 37–1 Features demonstrated in this example

Feature	Location
Use the Report Block Wizard to create a report layout.	Section 37.4, "Create the report layout using the Report Block Wizard"
Create a user parameter, summary column, the Paper Layout view, and a format trigger to add vertical space between a user-determined number of records.	Section 37.5, "Add vertical space between records"

37.1 Prerequisites for this example

To build the example in this chapter, you must have access to the Human Resources portion of the sample schema, which is provided with the Oracle9*i* database. If you do not have access to this schema, contact your database administrator.

37.2 Create a new PL/SQL library

The steps in this section will show you how to create a new PL/SQL library, then create a function that will live in this library.

To create the library:

1. Launch Reports Builder (or, if already open, choose **File > New > Report**)
2. In the Welcome or New Report dialog box, select **Build a new report manually**, then click **OK**.
3. In the Object Navigator, choose **File > New > PL/SQL Library**.

   A new library displays in the Object Navigator below your report name, under the **PL/SQL Libraries** node.
4. If it is not already expanded, expand the node of the new library to show the two subnodes: **Program Units** and **Attached Libraries**.
5. Click the **Program Units** node, then choose **Edit > Create**.
6. In the New Program Unit dialog box, in the **Name** field, type `BONUS_PAY`.
7. Select **Function**, then click **OK** to display the PL/SQL Editor.
8. In the PL/SQL Editor, use the template to enter the following PL/SQL code:

```
FUNCTION BONUS_PAY(JOB_ID IN CHAR, SAL IN NUMBER, COMM_PCT IN NUMBER) RETURN
```

```
NUMBER IS
BEGIN
  IF JOB_ID != 'SA_REP' THEN
    RETURN (SAL * 0.15);
  ELSE
    IF SAL * COMM_PCT >= 500 THEN
      RETURN ((SAL + SAL * COMM_PCT) * 0.15);
    ELSE
       RETURN ((SAL + SAL * COMM_PCT) * 0.10);
    END IF;
  END IF;
END;
```

Note: You can enter this code by copying and pasting it from the provided text file called `plsql_code.txt`.

9. Click **Compile**.
10. If there are compilation errors, match your code to the code we've provided (either in the example RDF file or in this chapter), then compile it again.
11. Once there are no compilation errors, click **Close**.

    Your new function displays in the Object Navigator.
12. Choose **File > Save** to save your new function.
13. In the Save Library dialog box, type `bonus.pll`, make sure **File System** is selected, then click **OK**.
14. In the Object Navigator, under the **MODULE1** report you've created, click the **Attached Libraries** node. Be sure to select this node, and not the one under the **PL/SQL Libraries** node.
15. Choose **Edit > Create**.
16. In the Attach Library dialog box, in the **Library** field, type `bonus.pll`.

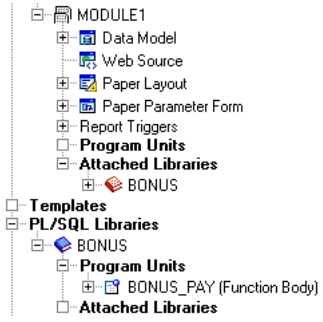
Note: If you saved bonus.pll to another directory, you can click **Browse** to find it on your file system. Just make sure you've selected **File System** before browsing.

17. When the library name displays in the **Library** field, click **OK** to attach the library.

> **Note:** If you attach a library whose name also includes a path, Reports Builder will inform you that the path names are not portable, and will give you the option of deleting the path. If you choose to continue using a path specification, Reports Builder will only look in that specific location for the library. So, if you move the library, Reports Builder will not be able to find it. If you delete the path, Reports Builder will use a standard search path to locate the library if it is moved.

The objects in your Object Navigator should now look something like this:

Figure 37–2 Object Navigator



18. Save your report as `plsqlreport_<your initials>.rdf`.

37.3 Create the report definition

The steps in this section will show you how to create the query and the formula columns that will define the report and call the code in the bonus.pll external PL/SQL library you created.

We recommend that you create the objects in the order described, as some of the formula columns depend on the functions, etc.

37.3.1 Create a query

The steps in this section will show you how to create the query that will retrieve the data necessary for this report.

To create the query:

1. In the Object Navigator, double-click the view icon next to the **Data Model** node to display the Data Model view.
2. In the Data Model view, choose **Insert > Query** to display the Data Wizard.
3. If the Welcome page displays, click **Next**.
4. On the Query page, leave the default query name, then click **Next**.
5. On the Data Source page, select **SQL Query**, then click **Next**.
6. On the Data page, in the **Data Source definition** field, enter the following SELECT statement:

```
SELECT FIRST_NAME, LAST_NAME, JOB_ID, SALARY,COMMISSION_PCT
FROM EMPLOYEES
ORDER BY LAST_NAME
```

Note: You can enter this query in any of the following ways:

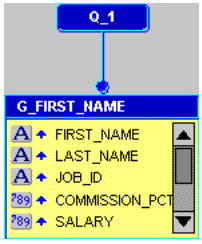
- Copy and paste the code from the provided text file called `plsql_code.txt` into the **Data Source definition** field.
- Click **Query Builder** to build the query without entering any code manually.
- Type the code in the **Data Source definition** field.

7. Click **Next**.

Note: If you are not already connected to a database, you will be prompted to connect to the database when you click **Next**. Ensure that you connect to a database that has the appropriate schema for this example. Section 37.1, "Prerequisites for this example" describes the sample schema requirements for this example.

8. On the Groups page, click **Next**.
9. Click **Finish** to display your first query in the Data Model view. It should look something like this:

Figure 37–3 Data Model view of the PL/SQL report



10. Save your report.

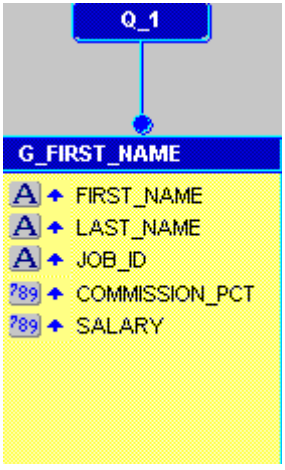
37.3.2 Create a formula column that calculates bonuses

The steps in this section will show you how to create a formula column that will calculate the salary bonus for each employee using the PL/SQL function.

To create the BONUS formula column:

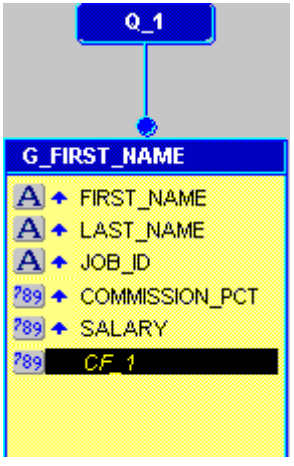
1. In the Data Model view, click group **G_FIRST_NAME**, then click the bottom resize handle and drag it down to make room in the data model for more columns. Here's an example of what it should look like now:

Figure 37–4 Data Model with expanded G_ENAME



2. Click the Formula Column tool in the tool palette, then click in the **G_FIRST_NAME** group to create a formula column.

Figure 37–5 Data Model with unnamed formula column



3. Double-click the new formula column object (**CF_1**) to display the Property Inspector, and set properties:
   - Under **General Information**, set the Name property to BONUS.
   - Under **Placeholder/Formula**, click the PL/SQL Formula property field to display the PL/SQL Editor.
4. In the PL/SQL Editor, use the template to enter the following PL/SQL code:

```
function BONUSFormula return Number is
begin
  return BONUS_PAY(:JOB_ID, :SALARY, :COMMISSION_PCT);
end;
```

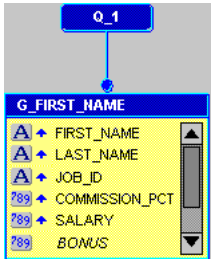
> **Note:** You can enter this code by copying and pasting it from the provided text file called `plsql_code.txt`. This code is for the Bonus Formula Column.

5. Click **Compile**.

> **Note:** If there are compilation errors, compare your code closely against the code we've provided.

6. When there are no compilation errors, click **Close** to display the data model for your report in the Data Model view. It should look something like this:

Figure 37–6 Data Model with BONUS formula column



7. Save your report.

37.3.3 Create a report-level function that calculates total compensation

The steps in this section will show you how to write a function that returns the total compensation for each sales representative (the values of columns SALARY plus COMM plus BONUS), as well as other employees (SALARY plus BONUS).

1. In the Object Navigator, click the **Program Units** node, then choose **Edit > Create**.
2. In the New Program Unit dialog box, in the **Name** field, type `FINAL_CALC`.
3. Select **Function**, then click **OK**.
4. In the PL/SQL Editor, use the template to enter the following PL/SQL code:

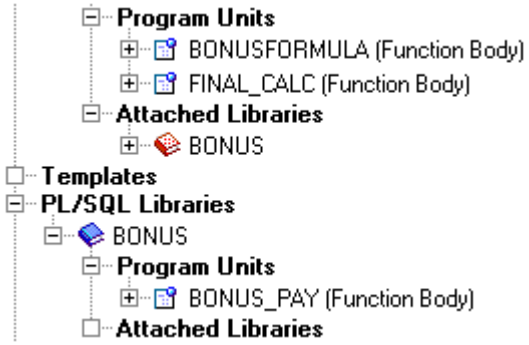
```
FUNCTION FINAL_CALC RETURN NUMBER IS
BEGIN
  IF :JOB_ID = 'SA_REP' THEN
  RETURN (:BONUS + :SALARY + :COMMISSION_PCT * :SALARY);
  ELSE
  RETURN (:BONUS + :SALARY);
  END IF;
END;
```

> **Note:** You can enter this code by copying and pasting it from the provided text file called `plsql_code.txt`. This code is for Final Calc.

5. Click **Compile**.
6. When the code is compiled without errors, click **Close**.

   The new function, **FINAL_CALC**, now displays in the Object Navigator:

Figure 37–7 Object Navigator with FINAL_CALC function



7. Save your report.

37.3.4 Create a second formula column for total compensation

The steps in this section will show you how to create another formula column that will calculate the total compensation. The value calculated by the report-level function FINAL_CALC will be assigned to the column TOTAL_COMP. If you are not sure how to create a formula column, refer to Section 37.3.2, "Create a formula column that calculates bonuses".

To create the TOTAL_COMP formula column:

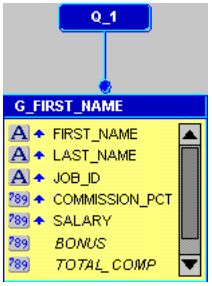
1. In the Data Model view, follow the steps above to create a second formula column below the **BONUS** formula column.
2. Double-click the new formula column object (**CF_1**) to display the Property Inspector, and set properties:
   - Under **General Information**, set the Name property to TOTAL_COMP.
   - Under **Placeholder/Formula**, click the PL/SQL Formula property field to display the PL/SQL Editor.
3. In the PL/SQL Editor, use the template to enter the following PL/SQL code:

```
function TOTAL_COMPFormula return Number is
begin
  return FINAL_CALC;
end;
```

> **Note:** You can enter this code by copying and pasting it from the provided text file called `plsql_code.txt`. This code is for the Total Comp Formula Column.

4. Click **Compile**.
5. When the code is compiled without errors, click **Close** to display the data model for your report in the Data Model view. It should look something like this:

Figure 37–8 Data Model with BONUS and TOTAL_COMP formula columns



6. Save your report.

37.4 Create the report layout using the Report Block Wizard

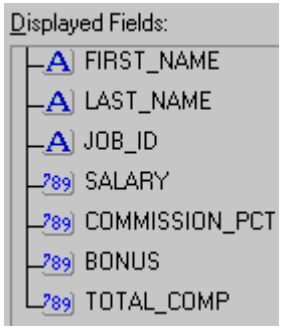
Now that you've created the necessary formula columns and functions, you can create the layout for your report.

To create the report layout:

1. Click the Paper Layout button in the toolbar to display the Paper Layout view.
2. In the Paper Layout view, choose **Insert > Report Block**.
3. In the Report Block Wizard, on the Style page, select **Tabular**, then click **Next**.
4. On the Groups page, click **G_FIRST_NAME** in the **Available Groups** list and click **Down** to specify the Print Direction and move this group to the **Displayed Groups** list, then click **Next**.

5. On the Fields page, click the double right arrows (>>) to move all of the fields to the **Displayed Fields** list. The Displayed Fields list should look like this:

Figure 37–9 Displays Fields list



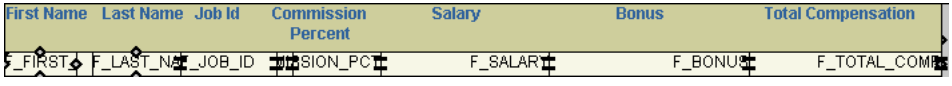
> **Note:** If the fields do not display in the correct order, simply click on the field name and drag the field to the correct position in the list.

6. Click **Next**.
7. On the Labels page, change the labels as follows, then click **Next**:

Fields	Labels
COMMISSION_PCT	`Commission`
TOTAL_COMP	`Total Compensation`

8. On the Template page, click **Finish** to display your report layout in the Paper Layout view. It should look something like this:

Figure 37–10 Paper Layout view of the PL/SQL example report

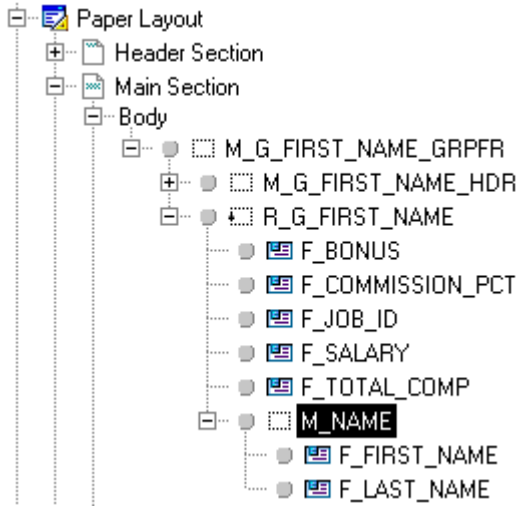


9. Click the Run Paper Layout button in the toolbar to run and display your report in the Paper Design view.
10. In the Paper Design view, click the Flex Off button in the toolbar.
11. Delete the **Last Name** label.
12. Change the text for the **First Name** label to **Name**.
13. Adjust the width of the new **Name** label to span over both the last name and first name columns
14. Adjust the sizes of the first name and last name columns so that one character displays between the columns.
15. In the Object Navigator, double-click the properties icon next to the **F_FIRST_NAME** field to display the Property Inspector, and set properties:
    - Under **General Layout**, set the Vertical Elasticity property to Fixed, and set the Horizontal Elasticity property to Variable.
16. Repeat the above step for the **F_LAST_NAME** field.
17. Click the Paper Layout button in the toolbar to display the Paper Layout view.
18. In the Paper Layout view, click the Frame tool in the tool palette.
19. Draw a frame around the two fields: **F_FIRST_NAME** and **F_LAST_NAME**.
20. With the frame selected, press F4 on your keyboard to display the Property Inspector, and set properties:
    - Under **General Information**, set the Name property to M_NAME.
21. In the Paper Layout view, make sure Flex Off is selected in the toolbar.
22. With the frame selected, choose **Layout > Move Backward** until the frame encloses both the **F_FIRST_NAME** and **F_LAST_NAME** fields.

    Tip: You can watch the fields in the Object Navigator as you choose **Layout > Move Backward**. When you see the two fields are sub-nodes of **M_NAME**, stop.

When you're done, the Object Navigator should look like this:

Figure 37–11 Object Navigator with M_NAME repeating frame



23. Click the Paper Design button in the toolbar to display the report in the Paper Design view.
24. In the Paper Design view, Shift-click to select the following columns:
    - **Salary**
    - **Commission**
    - **Bonus**
    - **Total Compensation**
25. Click the Currency button in the toolbar to add "$" to the numbers.
26. Click the Add Decimal Place button to add a decimal point to the numbers.

    Your report should now look something like this:

Figure 37–12 Paper Design view of modified report

Name	Job Id	Commission	Salary	Bonus	Total Compensation
Adam Fripp	ST_MAN		$8,200.0	$1,230.0	$9,430.0
Alana Walsh	SH_CLERK		$3,100.0	$465.0	$3,565.0
Alberto Errazuriz	SA_MAN	.3	$12,000.0	$1,800.0	$13,800.0
Alexander Hunold	IT_PROG		$9,000.0	$1,350.0	$10,350.0
Alexander Khoo	PU_CLERK		$3,100.0	$465.0	$3,565.0

27. Save your report.

37.5 Add vertical space between records

To make the report more readable, you can add space between a certain number of records. To do so, you first create the parameter that determines the number of records between which the space will display. Then, you create a summary column in the data model that counts the number of records. You will then modify the paper layout of your report so that the vertical elasticity is variable. Finally, you will create a format trigger that will display the space between the user-determined number of records.

37.5.1 Create a user parameter

The parameter you will create in this section will determine how many records are displayed before an extra space is printed. Since this parameter is a user parameter, the user can change this value at runtime.

To create a user parameter:

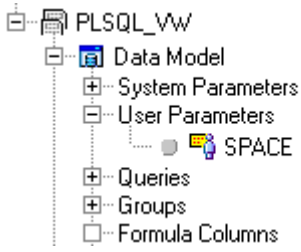
1. In the Object Navigator, under the **Data Model** node, click the **User Parameters** node.
2. Choose **Edit > Create** to create a new user parameter under the **User Parameters** node.
3. If the Property Inspector is not already displayed, right-click the new user parameter (**P_1**), then choose **Property Inspector** to display the Property Inspector, and set properties:
   - Under **General Information**, set the Name property to SPACE.

- Under **Parameter**, set the Datatype property to Number, and set the Initial Value property to 5.

> **Note:** By giving the user parameter an initial value, the user can simply run the report without changing the parameters, and a space will display between every five records.

The user parameter now displays in the Object Navigator:

Figure 37–13 User Parameter in the Object Navigator



4. Save your report.

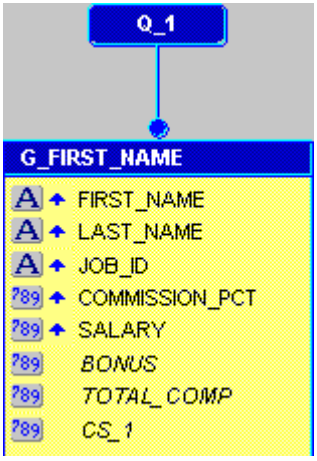
37.5.2 Create a summary column that counts the number of records

In this section, you will create a summary column in the data model that counts the number of employee records. This information will then be used by the format trigger to determine where to add extra space.

To create a summary column:

1. In the Object Navigator, double-click the view icon next to the **Data Model** node to display the Data Model view.
2. In the Data Model view, click the Summary Column tool in the tool palette, then click in the **G_FIRST_NAME** group beneath the **TOTAL_COMP** formula column to create a new summary column:

Figure 37–14 Data Model with new summary column



3. Double-click the new summary column object (**CS_1)** to display the Property Inspector, and set properties:
   - Under **General Information**, set the Name property to CNT_COLUMN.
   - Under **Summary**, set the Function property to Count, and set the Source property to FIRST_NAME.
4. Save your report.

37.5.3 Modify the layout

To allow Reports Builder to insert the vertical spacing, you must modify the layout of your report.

To add vertical elasticity:

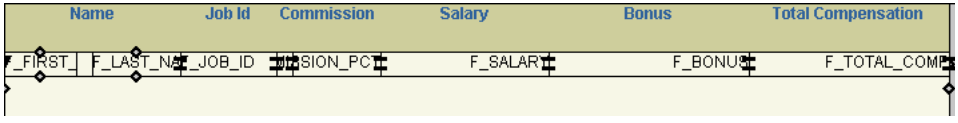
1. Click the Paper Layout button in the toolbar to display the Paper Layout view.
2. Click the Flex On button in the toolbar.
3. In the Paper Layout view, click the repeating frame associated with **G_FIRST_NAME**.

Note: If you can't find the repeating frame in the Paper Layout view, you can click **R_G_FIRST_NAME** in the Object Navigator. The associated repeating frame will be selected in the Paper Layout view.

4. Click the center handle of the frame and drag the frame downward to create additional space. This additional space should be slightly larger than what you want to see between the sets of records.
5. With the repeating frame selected, press F4 on your keyboard to display the Property Inspector, and set properties:
   - Under **General Layout**, set the Vertical Elasticity property to Variable.
6. In the Paper Layout view, click the Rectangle tool in the tool palette, and draw a rectangle below the fields in the newly created space.
7. Make sure the new rectangle has no fill and no line so that it is not visible.

   The following image shows the new layout with the invisible rectangle:

Figure 37–15 Layout with added vertical space



8. Save your report.

37.5.4 Create a format trigger

Now that you've adjusted the layout, you can create a format trigger based on the new boilerplate rectangle you created in the previous section. This format trigger will display this space after every set number of records, determined by the user parameter.

To create a format trigger on the boilerplate rectangle:

1. While the rectangle is selected in the Paper Layout view, press F11 on your keyboard (or choose **Tools > PL/SQL Editor**) to display the PL/SQL Editor.

2. In the PL/SQL Editor, use the template to enter the following PL/SQL code:

```
function B_1FormatTrigger return boolean is
begin
 If :CNT_COLUMN mod :SPACE = 0 then
   return(TRUE);
 else
   return(FALSE);
 end if;
end;
```

> **Note:** You can enter this code by copying and pasting it from the provided text file called `plsql_code.txt`. This code is for the Format Trigger.

3. Click **Compile**.
4. When the code is compiled without errors, click **Close**.
5. Save your report.

37.6 Run your report to paper

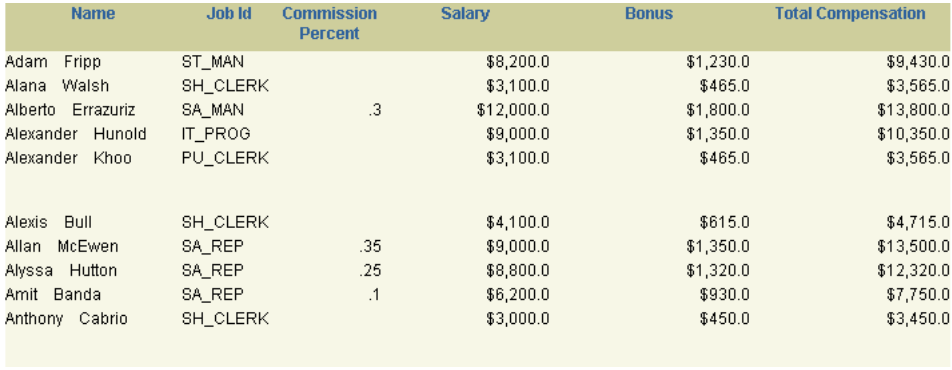
Now that you've added space and created your format trigger, your report should display with space between every five records (or whatever you specify in the Parameter Form).

To run your report:

- Click the Run Paper Layout button in the toolbar. When the Parameter Form displays, click the Run button in the toolbar.

Your report displays in the Paper Design view, and should look something like this:

Figure 37–16 Final PL/SQL example report



Name	Job Id	Commission Percent	Salary	Bonus	Total Compensation
Adam Fripp	ST_MAN		$8,200.0	$1,230.0	$9,430.0
Alana Walsh	SH_CLERK		$3,100.0	$465.0	$3,565.0
Alberto Errazuriz	SA_MAN	.3	$12,000.0	$1,800.0	$13,800.0
Alexander Hunold	IT_PROG		$9,000.0	$1,350.0	$10,350.0
Alexander Khoo	PU_CLERK		$3,100.0	$465.0	$3,565.0
Alexis Bull	SH_CLERK		$4,100.0	$615.0	$4,715.0
Allan McEwen	SA_REP	.35	$9,000.0	$1,350.0	$13,500.0
Alyssa Hutton	SA_REP	.25	$8,800.0	$1,320.0	$12,320.0
Amit Banda	SA_REP	.1	$6,200.0	$930.0	$7,750.0
Anthony Cabrio	SH_CLERK		$3,000.0	$450.0	$3,450.0

37.7 Summary

Congratulations! You have successfully built a report that uses an external PL/SQL library to calculate employee bonuses, which you can now use in other reports by simply referring to it. You now know how to:

- create and use an external PL/SQL library.
- create a default layout in the Report Wizard.
- add vertical space between a user-determined number of records.

For more information on any of the wizards, views, or properties used in this example, refer to the *Reports Builder Online Help*, which you can access in two ways:

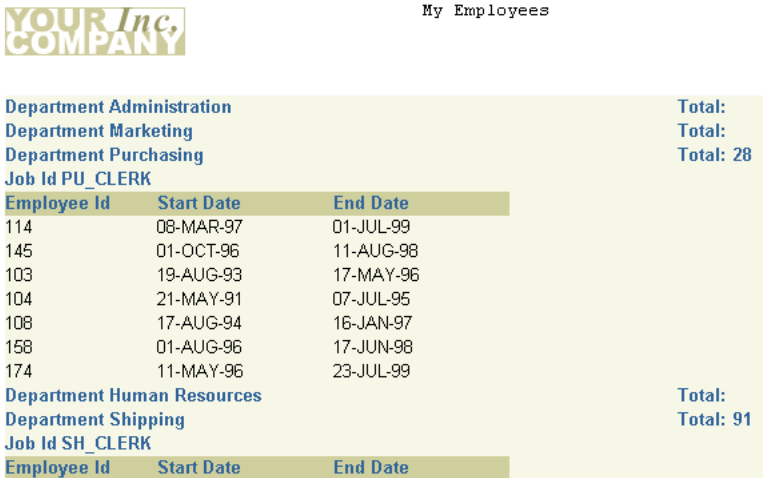
- From the Oracle Technology Network (`http://otn.oracle.com/products/reports/`), click **Documentation** and navigate to the *Reports Builder Online Help* for the most recent, hosted online help.
- From Reports Builder, choose **Help > Help Contents**.

> **Note:** You can replace your Reports Builder online help with the most recent update by downloading the latest online help set available on the Oracle Technology Network (`http://otn.oracle.com/docs/products/reports/`). Instructions for replacing your help file are included in the readme.txt in the download file.

38

Building a Paper Report with Ref Cursors

Figure 38–1 Ref cursor report output



Reports Builder enables you to easily manage your queries by use of ref cursors. By using a ref cursor, which is a PL/SQL cursor datatype, you can reference a cursor from within a PL/SQL query. For example, if you already have numerous queries built and you want to reuse those queries in your reports, you can simply use a ref cursor in your report data model to access those queries.

In this chapter, you will learn how to use Reports Builder's features for using ref cursors. To build this paper report, you will use the Data Model view to create a multiquery data model, and then use the Report Wizard to create the report layout. You will make fairly extensive manual refinements in the Data Model view.

About ref cursor queries

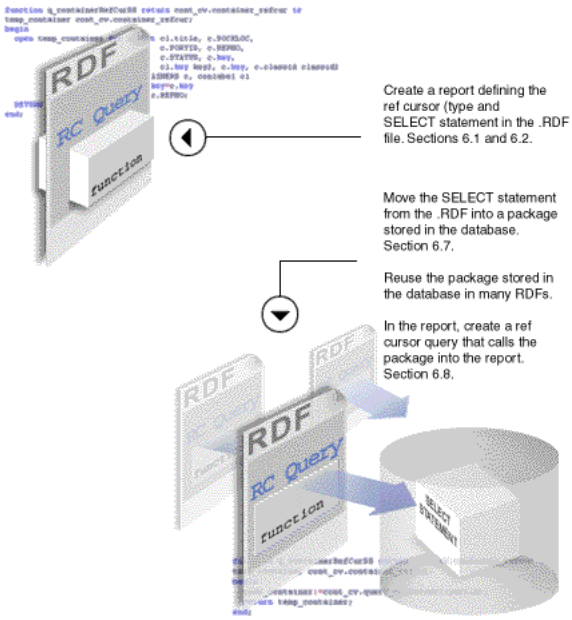
A ref cursor is a PL/SQL datatype that you can use in a query to fetch data. Each ref cursor query is associated with a PL/SQL function that returns a strongly typed ref cursor. The PL/SQL function must ensure that the ref cursor is opened and associated with a SELECT statement that has a SELECT list that matches the ref cursor type. You base a query on a ref cursor when you want to:

- more easily administer SQL.
- avoid the use of lexical parameters in your reports.
- share data sources with other applications.
- increase control and security.
- encapsulate logic within a subprogram.

Furthermore, if you use a stored program unit to implement ref cursors, you receive the added benefits that go along with storing program units in the Oracle database.

The following figure shows that you create a report with the SELECT statement in the ref cursor query of the report. It also shows that you can store the SELECT statement in a package in the database. Then, from the report, you can call the package from the database allowing you to reuse the package in many reports.

Figure 38–2 Overview of the ref cursor example



Example Scenario

In this example, you will create a detailed report showing information about employees and the job position they hold in each department.

Table 38–1 Features demonstrated in this example

Feature	Location
Create package specs that define ref cursors.	Section 38.2, "Define a ref cursor type"
Create ref cursor queries that will use the ref cursors.	Section 38.3, "Create a ref cursor query"
Rename objects in the data model so that they have more meaningful names.	Section 38.4, "Refine the data model"
Create group-to-group data links between ref cursor queries to create relationships between them.	Section 38.5, "Create links between ref cursor queries"

Table 38–1 Features demonstrated in this example

Feature	Location
Create summaries that better describe the data.	Section 38.6, "Add summary columns"
Use the Report Wizard to create a report layout.	Section 38.7, "Create a layout"
Move the SELECT statements used by the ref cursor queries from the report and into packages that define the ref cursor types.	Section 38.8, "Move the SELECT statement into a package"
Move the packages into a PL/SQL library so that other reports can share the code.	Section 38.9, "Move the packages into a library"

38.1 Prerequisites for this example

To build the examples in this manual, you must have the example files we've provided. If you haven't already done so, you can download the files you'll need to complete this example from the Oracle Technology network and install them on your machine.

To download and install the example files:

1. Go to the Oracle Technology Network Web site (`http://otn.oracle.com/product/reports/`).
2. Click **Getting Started with Oracle Reports**.
3. Click **Index**, then find the "Building a Paper Report with Ref Cursors" example.
4. Download the file refcursor.zip into a temporary directory on your machine (e.g., `d:\temp`).
5. Unzip the contents of the file, maintaining the directory structure, into an examples directory on your machine (e.g., `d:\orawin90\examples`).

   This zip file contains the following files:

Table 38–2 Files necessary for building this sample report using ref cursors

File	Description
Examples\RefCursor\result\ref_emp*.rdf	The different stages of the RDF. You can refer to these files as you complete each section of this chapter. The file ref_emp68.rdf is the final report.

Table 38–2 Files necessary for building this sample report using ref cursors

File	Description
Examples\RefCursor\scripts\refcursor_code.txt	The PL/SQL code you will write in this chapter. You can use this file as a reference point to make sure your code is accurate, or you can simply cut and paste from this file into Reports Builder.

38.1.1 Access to the sample Human Resources schema

If you don't know if you have access to the sample Human Resources schema provided with the Oracle9*i* database, contact your database administrator.

38.2 Define a ref cursor type

To create a ref cursor query, you first create a package spec that defines the ref cursor. Then you create a query that uses the ref cursor. The steps in this section will help you create package specs that define ref cursors.

To define a ref cursor type:

1. Launch Reports Builder (or, if already open, choose **File > New > Report**)
2. In the Welcome or New Report dialog box, select **Build a new report manually**, then click **OK**.
3. In the Object Navigator, click the **Program Units** node under your UNTITLED report node.
4. Click the Create button in the toolbar to display the New Program Unit dialog box.
5. In the New Program Unit dialog box, type `concl_cv` in the **Name** field.
6. Select **Package Spec,** then click **OK** to display the PL/SQL Editor.
7. In the PL/SQL Editor, use the template to enter the following PL/SQL code:

```
PACKAGE concl_cv IS
    type conclass_rec is RECORD
       (EMPLOYEE_ID NUMBER(6),
     FIRST_NAME VARCHAR2(20),
     LAST_NAME VARCHAR2(25),
     e-mail VARCHAR2(25),
     PHONE_NUMBER VARCHAR2(20),
     HIRE_DATE DATE,
```

```
     JOB_ID VARCHAR2(10),
     SALARY NUMBER(8,2),
     DEPARTMENT_ID NUMBER(4));
    type conclass_refcur is REF CURSOR return conclass_rec;
END;
```

This package spec does two things:

- defines a record (conclass_rec) that describes the data you want to select from the database.
- defines a ref cursor that returns the data in the format described by the record.

Note: You can open the file *Examples*/RefCursor/scripts/refcursor_code.txt to copy and paste the code into the PL/SQL Editor.

---

8. Click **Compile**.
9. If any compilation errors occur, check the code for syntax errors and recompile as needed.
10. Click **Close**.
11. Repeat steps 2 through 8 to create two more package specs:
    - **Package Spec Name: cont_cv**

```
PACKAGE cont_cv IS
    type container_rec is RECORD
       (EMPLOYEE_ID NUMBER(6),
        START_DATE DATE,
        END_DATE DATE,
        JOB_ID VARCHAR2(10),
        DEPARTMENT_ID NUMBER(4));
    type container_refcur is REF CURSOR return container_rec;
END;
```

    - **Package Spec Name: port_cv**

```
PACKAGE port_cv IS
  type portdesc_rec is RECORD
```

```
      (DEPARTMENT_ID NUMBER(4),
       DEPARTMENT_NAME VARCHAR2(30));
  type portdesc_refcur is REF CURSOR return portdesc_rec;
END;
```

Note: You can open the file *Examples*/RefCursor/scripts/refcursor_code.txt to copy and paste the code into the PL/SQL Editor.

12. Choose **File > Save As**. Save the report in the directory of your choice, and name the report ref61_<your initials>.rdf.

Note: It is good practice when you are designing your report to save it frequently under a different file name. If you generate an error or if you don't like some of the changes you made, you easily can go back to the previously saved file and make revisions from that point.

38.3 Create a ref cursor query

After creating package specs that define the ref cursors, you are ready to define the queries, as described in this section.

To create a ref cursor query:

1. In the Object Navigator, double-click the view icon next to the **Data Model** node to display the Data Model view.
2. In the Data Model view, click the Ref Cursor Query tool in the tool palette.
3. Click in an open area of the Data Model view to display the PL/SQL Editor for the new ref cursor query.
4. In the PL/SQL Editor, use the template to enter the following PL/SQL code:

```
function q_portdescRefCurDS return port_cv.portdesc_refcur is
temp_portdesc port_cv.portdesc_refcur;
begin
 open temp_portdesc for select department_id, department_name from depart-
```

```
ments;
  return temp_portdesc;
end;
```

Note: You can open the file *Examples*/RefCursor/scripts/refcursor_code.txt to copy and paste the code into the PL/SQL Editor.

5. Click **Compile**.
6. If any compilation errors occur, check the code for syntax errors and recompile as needed.
7. Click **Close**.
8. In the Data Model view, select the new ref cursor query object (**QR_1**), then press F4 to display the Property Inspector.
   - Under **General Information**, set the Name property to q_portdesc.

     Tip: It is usually a good idea to give objects meaningful names, particularly when building a report with many objects. Later when building the layout, it is helpful to have queries and groups with meaningful names.

9. Repeat the steps above to create two more queries:
   - **Query name: q_container**

```
function q_containerRefCurDS return cont_cv.container_refcur is
temp_container cont_cv.container_refcur;
begin
    open temp_container for
     select employee_id,
         start_date,
         end_date,
         job_id,
         department_id
      from job_history;
    return temp_container;
end;
```

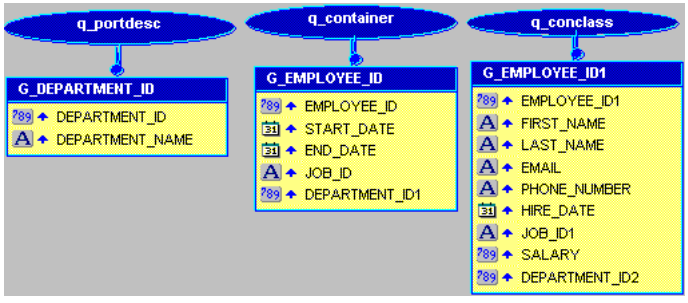
- **Query name: q_conclass**

```
function q_conclassRefCurDS return concl_cv.conclass_refcur is
temp_concl concl_cv.conclass_refcur;
begin
  open temp_concl for
   select employee_id,
        first_name,
        last_name,
        e-mail,
        phone_number,
        hire_date,
        job_id,
        salary,
        department_id
    from employees;
  return temp_concl;
end;
```

Note: You can open the file *Examples*/RefCursor/scripts/refcursor_code.txt to copy and paste the code into Reports Builder.

The Data Model should look similar to the following:

Figure 38–3 Data model with three queries



10. Save the report as `ref_62_<your initials>.rdf`.

> **Note:** You can open the provided file *Examples*`/RefCursor/result/ref_emp62.rdf` and display the Data Model to compare your results.

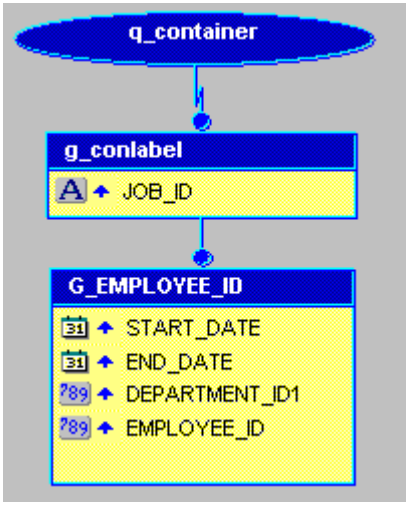
38.4 Refine the data model

In this section, you will rename some of the objects in the data model so that they have more meaningful names. You will also create a break group.

To refine the data model:

1. In the Data Model view, drag the title bar of the group **G_EMPLOYEE_ID** down a few inches to move the entire group.
2. Click and drag the column named **JOB_ID** out of and above **G_EMPLOYEE_ID** to create a new break group, as shown in the following figure:

Figure 38–4 Query with group

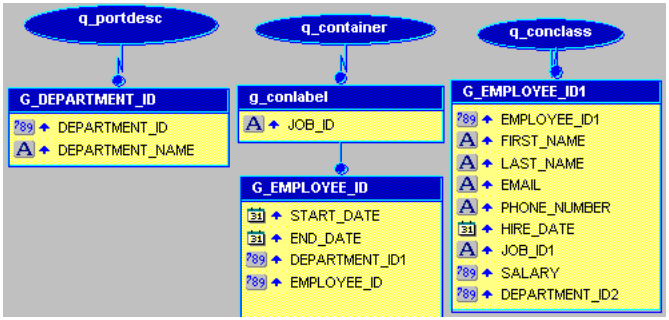


3. Double-click the title bar of the new group that contains **JOB_ID** to display the Property Inspector, and set properties:

- Under the **General Information** node, set the Name property to G_conlabel.

4. In the Data Model view, your data model should look similar to the following:

Figure 38–5 Data model with group



Note: You can open the provided file *Examples*`/RefCursor/result/ref_emp63.rdf` and display the Data Model to compare your results.

---

5. Save the report as `ref_63_<yourinitials>.rdf`.

38.5 Create links between ref cursor queries

Currently, the queries that you have created are unrelated. To create relationships between them, you need to create group-to-group data links. The steps in this section will help you create the links.

To create links between ref cursor queries:

1. In the Data Model view, click the Data Link tool in the tool palette.
2. Click the title bar of **G_DEPARTMENT_ID**, and drag a link to the title bar of **G_EMPLOYEE_ID**.
3. Double-click **q_container** to display the PL/SQL Editor.

4. In the PL/SQL Editor, append code to the WHERE clause of the SELECT statement to specify which columns are being used as primary and foreign keys:

   After `from job_history`, add the following code:

   ```
   where:department_id = department_id;
   ```

   Be sure that the semicolon `(;)` now follows the WHERE clause.

   Note that :department_id is a bind variable referring to the DEPARTMENT_ID in G_DEPARTMENT_ID.

5. Click **Compile**.
6. If any compilation errors occur, check the code for syntax errors and recompile as needed.
7. Click **Close**.
8. In the Data Model view, click the Data Link tool in the tool palette.
9. Click the title bar of G_EMPLOYEE_ID and drag a link to the title bar of G_EMPLOYEE_ID1.
10. Double-click **q_conclass** to display the PL/SQL Editor.
11. In the PL/SQL Editor, add a WHERE clause to the SELECT statement:

    Insert your cursor between `FROM EMPLOYEES` and the semicolon (;), and press ENTER or RETURN to create a new line, then add the following code:

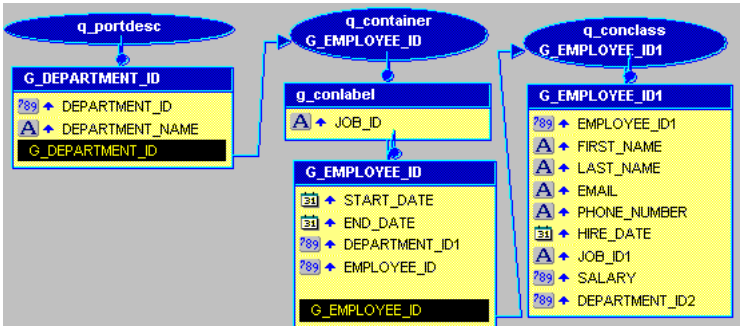
    ```
    where :employee_id = employee_id;
    ```

    Be sure that the semicolon `(;)` now follows the WHERE clause.

    Note that `:employee_id` is a bind variable referring to the EMPLOYEE_ID column in G_employee_id.

12. Click **Compile**.
13. If any compilation errors occur, check the code for syntax errors and recompile as needed.
14. Click **Close**.
15. Your data model should look similar to the following:

Figure 38–6 Data model with links



Note: You can open the provided file *Examples*/RefCursor/result/ref_emp64.rdf and display the Data Model to compare your results.

16. Save the report as ref_64_<your initials>.rdf.

38.6 Add summary columns

Now that your queries are complete and linked, the steps in this section will help you to create columns to summarize the data.

To add summary columns:

1. In the Data Model view, click the Summary Column tool in the tool palette.
2. Click inside the **G_EMPLOYEE_ID** group to create a summary column.
3. Double-click the new summary column object (**CS_1**) to display the Property Inspector, and set properties:
   - Under **General Information**, set the Name property to CS_classcount.
   - Under **Summary**, set the Function property to Count, set the Source property to employee_id, and set the Reset At property to G_department_id.

You have now created a summary that counts up the number of employees. You will not use the summary in this report's layout, but you will use it as the source for other, more interesting summaries later.

4. Repeat the steps above to create summaries with the following characteristics:

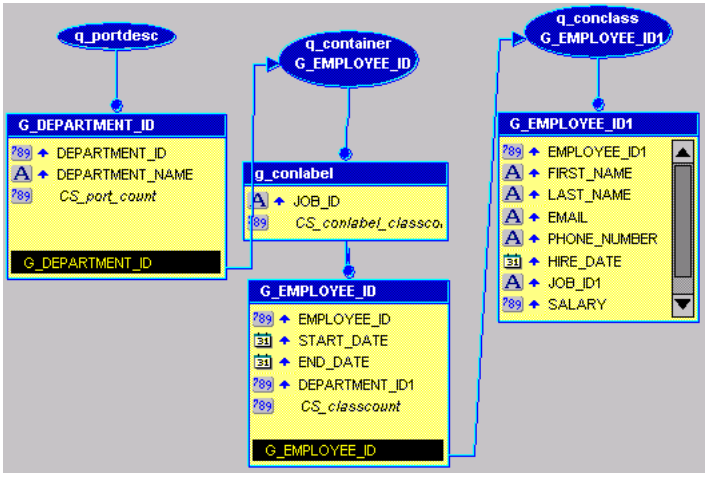
Table 38–3 Summary Characteristics

Create in Group	Name	Function	Source	Reset At
G_conlabel	CS_conlabel_ classcount	Sum	CS_classcount	G_conlabel
G_ department_ id	CS_port_count	Sum	CS_conlabel_ classcount	G_ DEPARTMENT_ ID

You may not understand these summaries now. Their purpose will become clearer when you create the report layout and preview the live data.

Your data model should look similar to the following:

Figure 38–7 Data model with summary columns



Note: You can also compare your results to the file we've provided, called `ref_emp65.rdf`.

5. Save the report as `ref_65_<your initials>.rdf`.

38.7 Create a layout

Now that you have a working data model, the steps in this section will help you to create a layout.

To create a paper layout:

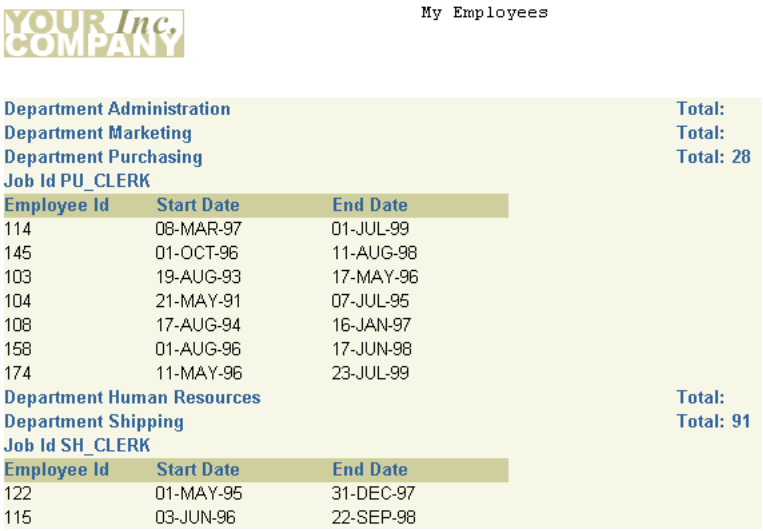
1. In the Object Navigator, right-click the report name and choose **Report Wizard**.
2. In the Report Wizard, on the **Report Type** page, select **Create Paper Layout Only**.
3. On the **Style** page, type `My Employees` in the **Title** field, select **Group Above**.
4. On the **Groups** page, click the following fields in the **Available Fields** list and click **Down** to specify the Print Direction and move them to the **Group Fields** list:
   - **G_conlabel**
   - **G_DEPARTMENT_ID**
   - **G_EMPLOYEE_ID**
5. On the **Fields** page, click the following fields and click the right arrow (>) to move them to the **Displayed Fields** list:
   - **DEPARTMENT_NAME**
   - **EMPLOYEE_ID**
   - **START_DATE**
   - **END_DATE**
   - **JOB_ID**
   - **CS_port_count**
6. On the **Labels** page, change the labels and field widths as follows:

Table 38–4 Field description of Labels page

Fields	Labels	Width
DEPARTMENT_NAME	Department	30
EMPLOYEE_ID1	Employee ID	8
START_DATE	Start Date	9
END_DATE	End Date	9
JOB_ID	Job ID	10
CS_port_count	Total:	12

7. On the **Template** page, choose **Predefined template** and click **Beige**, then click **Finish** to display your report output in the Paper Design view. It should look something like this:

Figure 38–8 Paper Design view for the ref cursor report



Note: You can open the provided file *Examples*/RefCursor/result/ref_emp66.rdf and display the Paper Design view to compare your results.

8. Save the report as `ref_66_<your initials>.rdf`.

38.8 Move the SELECT statement into a package

In your current report configuration, the SELECT statements used by the ref cursor queries reside within the report itself. In many cases, it is advantageous to have SELECT statements reside in the packages that define the ref cursor types. Then, you can simply reference the packages, rather than typing the same SELECT statement directly into every report that uses it. If you need to change the SELECT statement (for example, to modify or add clauses), you simply update it once in the package, rather than in every report that uses it.

The steps in this section will help you to move the SELECT statements to the packages that define the ref cursor types.

To move the SELECT statement into a package:

1. In the Object Navigator, click the **Program Units** node for your report.
2. Click the Create button in the toolbar to display the New Program Unit dialog box.
3. In the New Program Unit dialog box, type `cont_cv` as in the **Name** field.
4. Select **Package Body,** and click **OK** to display the PL/SQL Editor for the new program unit.
5. In the PL/SQL Editor, use the template to enter the following PL/SQL code:

```
PACKAGE BODY cont_cv IS
     function query_container (p_department_id number)
     return container_refcur is tempcv_container cont_cv.container_refcur;
 begin
    open tempcv_container for
        select employee_id,
        start_date,
        end_date,
        job_id,
        department_id
        from   job_history
        where  :department_id=department_id;
    return tempcv_container;
end;
```

```
END;
```

> **Note:** You can open the provided file *Examples*/RefCursor/scripts/refcursor_code.txt to copy and paste the code into Reports Builder.

6. Click **Compile**.
7. If any compilation errors occur, check the code for syntax errors and recompile as needed.
8. Click **Close**.
9. Now that the function is defined, you must add it to the package spec so that it can be referenced. Other program units will know about the function in the package body only if it is described in the package spec.
10. In the Object Navigator, double-click the **CONT_CV(Package Spec)** object to display the PL/SQL Editor.
11. In the PL/SQL Editor, type the following line above the `END;` statement:

```
function query_container (p_department_id number) return container_refcur;
```

12. Click **Close**.
13. Choose **Program > Compile > All**.
14. Click **OK** when done.
15. In the Object Navigator, under the **Program Units** node, double-click **Q_CONTAINERREFCURDS** to display the PL/SQL Editor.
16. In the PL/SQL Editor, edit the code to look as follows:

```
function Q_containerRefCurDS return cont_cv.container_refcur is
    temp_container cont_cv.container_refcur;
begin
     temp_container:=cont_cv.query_container (:department_id);
       return temp_container;
end;
```

When you are done, all of the query's logic will reside in the function named query_container. From now on, when you change query_container, you will change this and any other queries that reference it.

Note: You can open the file *Examples*/RefCursor/scripts/refcursor_code.txt to copy and paste the code into Reports Builder.

---

17. Click **Compile**.
18. If any compilation errors occur, check the code for syntax errors and recompile as needed.
19. Click **Close**.
20. Click the Paper Design button in the toolbar to view the report in the Paper Design view.
21. Save the report as `ref_67_<your initials>.rdf`.

Optional Exercise:

Repeat the above steps for the other two queries in the report.

38.9 Move the packages into a library

If you have many reports that use these same ref cursor types and SELECT statements, you can move the program units that you created into a PL/SQL library stored in a file or the database, so that other reports can easily share the code. The steps in this section will help you to move the program units to a PL/SQL library.

To move the packages into a library:

1. In the Object Navigator, click the **PL/SQL Libraries** node, then click the Create button in the toolbar to add a new library.
2. Choose **File > Save As**.
3. Type `DEPT_CONTAINER` as the Library.
4. Click **File System**.
5. Click **OK**.

6. Drag and drop the following program units from your report to the **Program Units** node under the newly created **DEPT_CONTAINER** library:
   - **CONCL_CV(Package Spec)**
   - **CONT_CV(Package Spec)**
   - **CONT_CV(Package Body)**
   - **PORT_CV(Package Spec)**
7. Save DEPT_CONTAINER.
8. If the Paper Design view is open, close it.
9. In the Object Navigator, under the **Program Units** node for your report, delete **CONCL_CV**(Package Spec), **CONT_CV**(Package Spec), **CONT_CV**(Package Body), and **PORT_CV**(Package Spec).

> **Note:** If the Paper Design view is open when you delete the packages from the report, you may get some errors.

10. Click the **Attached Libraries** node for your report, then click the Create button in the toolbar to add a new attached library.
11. In the Attach Library dialog box, click **File System**.
12. Click **Browse** to find the DEPT_CONTAINER library. It will have a `.PLL` file extension. After you have found and selected DEPT_CONTAINER, click **Open**.
13. Click **Attach**.
14. Choose **Program > Compile > All**.
15. Click **OK** to close the Compile window.
16. Click the Paper Design button in the toolbar to run the report and view it in the Paper Design view.

> **Note:** If you get an error when you attempt to view the report, repeat steps 16 through 18.

17. Save the report as `ref_68_<your initials>.rdf`.

Optional Exercise:

Store the PL/SQL library in the database rather than in a file. Note that you will need "create" privileges on the database to complete this optional exercise.

38.10 Summary

Congratulations! You have finished the Ref Cursor Query sample report. You now know how to:

- create package specs that define ref cursors.
- create ref cursor queries.
- create data links between ref cursor queries.
- create summaries to describe data.
- create a report layout.
- move SELECT statements into packages.
- move packages into a PL/SQL library.

For more information on any of the wizards, views, or properties used in this example, refer to the *Reports Builder Online Help*, which you can access in two ways:

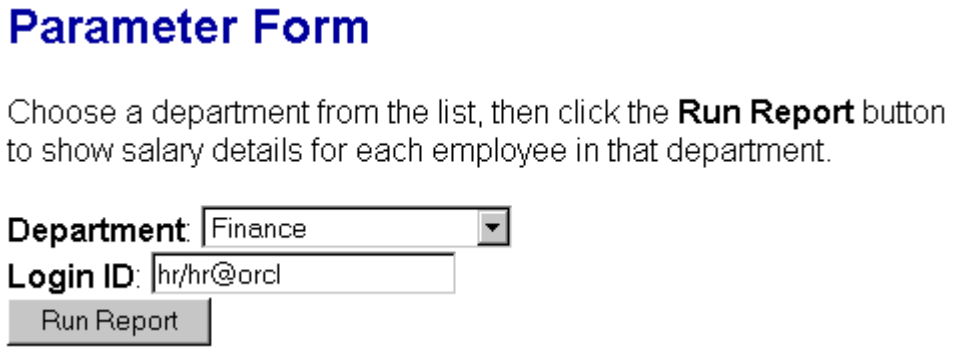
- From the Oracle Technology Network (`http://otn.oracle.com/products/reports/`), click **Documentation** and navigate to the *Reports Builder Online Help* for the most recent, hosted online help.
- From Reports Builder, choose **Help > Help Contents**.

> **Note:** You can replace your Reports Builder online help with the most recent update by downloading the latest online help set available on the Oracle Technology Network (`http://otn.oracle.com/docs/products/reports/`). Instructions for replacing your help file are included in the readme.txt in the download file.

39

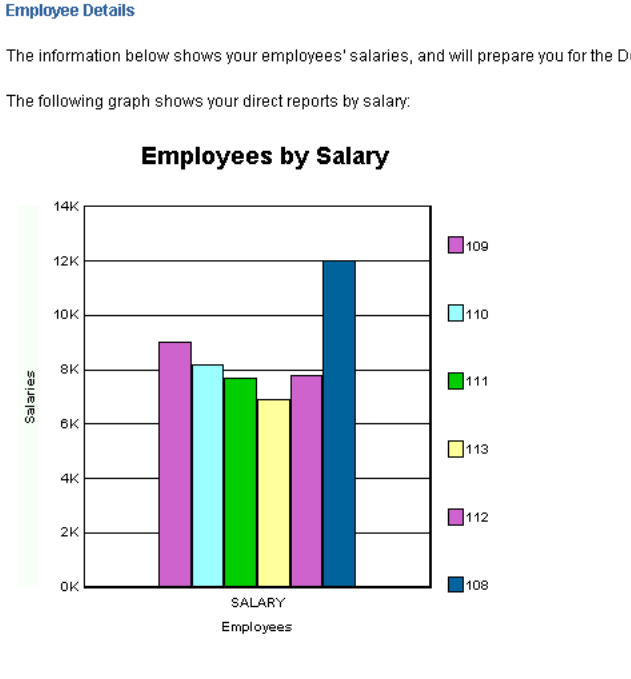
Building a Simple Parameter Form for a JSP-based Web Report

Figure 39–1 JSP Parameter Form



Note: The JSP Parameter Form in this image enables the user to choose from a list of departments, connect to a database, then run a report based on the selected parameters.

Figure 39–2 JSP-based Web report based on a user parameter



> **Note:** The JSP-based Web report in this image displays a graph and a tabular report based on the department selected in the JSP Parameter Form. To see how this report was built, refer to the *Oracle Reports Tutorial*.

Oracle Reports enables you to build reports where certain criteria are defined at runtime. To enable your users to define the criteria at runtime, you can use Parameter Forms. The steps in this chapter show you how to build an HTML Parameter Form using JavaServer Pages for a JSP-based Web report. To build a Parameter Form for a paper report, refer to the *Reports Builder Online Help*. This chapter shows you how to build a very simple Parameter Form. If this Parameter

Form does not suit your needs, refer to a more advanced example located in the Getting Started with Oracle Reports Web site on the Oracle Technology Network (`http://otn.oracle.com/products/reports/`).

For conceptual information about JSPs and Parameter Forms for Web reports, refer to Section 2.2.1, "About JavaServer Pages (JSPs) and servlets" and Section 1.9.4, "About Parameter Forms for Web reports".

Example Scenario

Suppose you have an existing JSP-based Web report that shows a bar graph of employee salaries per department, as well as a tabular report that shows the employee details. Now, your customers want to be able to specify at runtime the employee information for a specific department, so that they don't have to read the data for all departments. The steps in this report will show you how to add a JSP Parameter Form to this Web report.

This example uses the resulting report from the *Oracle Reports Tutorial*. If you'd like to learn how to build the Web report that we use in this chapter, follow the steps in the tutorial.

Table 39–1 Features demonstrated in this example

Feature	Location
Use a text or HTML editor to create a simple HTML Parameter Form.	Section 39.2, "Create a Parameter Form in HTML"
Use the Data Model and Web Source views in Reports Builder to modify the Parameter Form and save as a JSP.	Section 39.3, "Modify the HTML Parameter Form in Reports Builder"
Use the Data Model view to modify the query for the target report to accept user parameters.	Section 39.4, "Set up the target report"
Test the deployment of the JSP Parameter Form and Web report.	Section 39.5, "Deploy the JSP report and Parameter Form"

39.1 Prerequisites for this example

To build the examples in this manual, you must have the example files we've provided, as well as access to the sample schema that is shipped with the Oracle9*i* database.

39.1.1 Example files

If you haven't already done so, you can download the files you'll need to complete this example from the Oracle Technology network (`http://otn.oracle.com/products/reports`) and install them on your machine.

To download and install the example files:

1. Go to the Oracle Technology Network Web site (`http://otn.oracle.com/product/reports`).
2. Click **Getting Started with Oracle Reports**.
3. Click **Index**, then browse the list of examples to find the "Building a Simple Parameter Form for a JSP-Based Web Report" example.
4. Download the file `simplejsppf.zip` into a temporary directory on your machine (e.g., `d:\temp`).
5. Unzip the contents of the file, maintaining the directory structure, into an examples directory on your machine (e.g., `d:\orawin90\examples`).

   This zip file contains the following files:

Table 39–2 Files necessary for building a simple JSP Parameter Form

File	Description
`Examples\SimpleJSPPF\source\paramform.html`	An example HTML Parameter Form, which contains an example List of Values, a field, and a button.
`Examples\SimpleJSPPF\results\paramform.jsp`	This JSP-based Web report contains the modifications to the Parameter Form you will make in Reports Builder.
`Examples\SimpleJSPPF\results\emprev_final.jsp`	The source Web report that will become the target report for the Parameter Form.
`Examples\SimpleJSPPF\results\emprev_param.jsp`	The final JSP-based Web report with a JSP Parameter Form.

Note: If you completed the exercises in the *Oracle Reports Tutorial*, you can also use the `emprevb_<your initials>.jsp` file you finished at the end of Chapter 6.

39.1.2 Access to the sample schema

If you don't know if you have access to the sample schema provided with the Oracle9*i* database, contact your database administrator. You should have access to the "Human Resources" portion of the schema to complete this example.

39.2 Create a Parameter Form in HTML

The steps in this section will show you how to build a simple Parameter Form using plain HTML. You will then modify this HTML Parameter Form in Reports Builder so that you can call the Parameter Form from your JSP-based Web report.

If you don't want to create your own HTML file, you can open the sample HTML file we've provided in the Source directory, called `paramform.html`, then view the source code.

To create a simple Parameter Form in HTML:

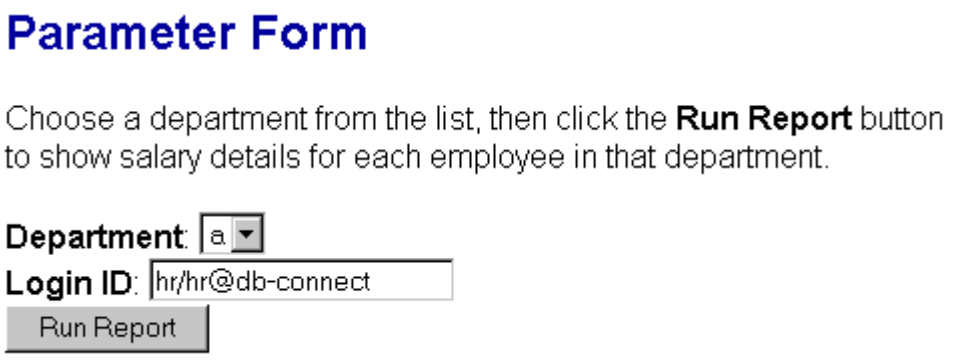
1. In a text editor or HTML editor, create an HTML page that contains a form. The form should contain a list of values, a field, and a button. The code for this form can look something like the following:

```
<form name="form1" method="post" action="">
    <select name="p_department" size="1">
      <option value="1">a</option>
    </select>
    <br>
    <input type="text" name="userid" value="hr/hr@db-connect">
<br>
    <input type="submit" name="Submit" value="Run Report">
</form>
```

> **Note:** Although you can use the above code, you will need to change the `userid` value to reflect the connection information for your data source. You can also copy and paste the HTML code from the provided file, called `simplejsppf_code.txt` in the `SimpleJSPPF/scripts` directory, then modify it in a text or HTML editor.

2. Save the HTML file as `paramform_<your initials>.html`. When you display this HTML file in a Web browser, it should look similar to the following image:

Figure 39–3 Sample HTML Parameter Form



39.3 Modify the HTML Parameter Form in Reports Builder

In this section, you will learn how to modify an HTML Parameter Form in Reports Builder to populate the list of values (LOV) you created with values from a data source. You will use JSP tags for Oracle Reports to enable the Parameter Form to access elements from a data model.

39.3.1 Create a data model manually for the Parameter Form

The steps in this section will show you how to create a simple data model for the Parameter Form.

To create a data model:

1. In Reports Builder, open the HTML file you created, `paramform_<your initials>.html`.
2. In the Object Navigator, double-click the icon next to the Data Model node to display the Data Model view.
3. In the Data Model view that displays, click the SQL Query tool in the tool palette, then click in an open area of the Data Model view to display the SQL Query Statement dialog box.

4. In the **SQL Query Statement** field, enter the following SELECT statement:

```
select department_name, department_id
from departments
order by department_name
```

Note: You you can enter this query in any of the following ways:

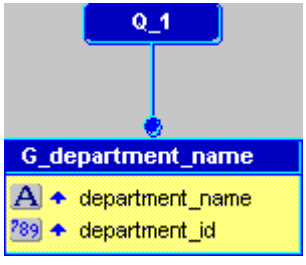
- Copy and paste the code from the provided text file called `simplejsppf_code.txt` into the **SQL Query Statement** field.
- Click **Query Builder** to build the query without entering any code manually.
- Type the code in the **SQL Query Statement** field.

Also note that if you have not installed the Pictures table into the sample schema, you will not be able to create this query.

---

5. Click **Connect**, then type the connection information for the "Human Resources" portion of the sample schema.
6. Click **OK**. Your data model should look like this:

Figure 39–4 Data Model view of the Parameter Form



7. Save the report as `paramform_<your initials>.jsp`.

39.3.2 Create a dynamic LOV in the Parameter Form

In this section, you will learn how to modify the Web source to pull data into the existing list of values (LOV) in your Parameter Form. This data will rely on the data model you created in the previous section. We will also examine the code to explain how each element operates.

To modify the LOV in the Parameter Form in Reports Builder:

1. Click the Web Source button in the toolbar to display the Web Source view.
2. In the Web Source view, look for the following code:

```
<select name="p_department" size="1">
<option value="1">a</option>
</select>
```

Note: In the above code, the LOV returns a static value. The display name is "a" and the value is "1".

Since the LOV is currently static, we need to change this HTML element to dynamically retrieve data based on our data model.

3. In the Web Source view, modify the above code so that it looks like the following:

```
<select name=" p_department">
<rw:foreach id="fn" src="G_department_name">
<option value="<rw:field id="f_deptId" src="department_id"/>"><rw:field
id="deptname1" src="department_name"/></option>
</rw:foreach>
</select>
```

Note: You can either type the code in manually or copy and paste it from the provided file called `simplejsppf_code.txt`.

4. Save your report.

Examine the JSP elements in the code:

By using JSP tags for Oracle Reports in the above code, we retrieve data into the Parameter Form's LOV by basing the parameters on fields in the data model. Let's examine each element:

- `<option>`: The display name of the LOV is replaced by the field `department_name` from the data model. When the user displays the Parameter Form, the department name will display in the list.
- `<rw:field>`: This element accesses each element of the `g_department_name` group.
- `<rw:foreach>`: This element iterates through the results based on the `g_department_name` group in the data model.
- `src`: This parameter for the `<rw:for each>` element must match the group name of the data model.
- `<rw:field id>`: This parameter can be any value, but it must be unique.

By making these modifications to the code, we've replaced the return value attribute of the LOV with the field `department_id`, based on the data model we created in the previous section. If we now choose a department name from the list of values, its related department ID is returned. Note that the return value is not displayed.

39.3.3 Run the Parameter Form report to the Web

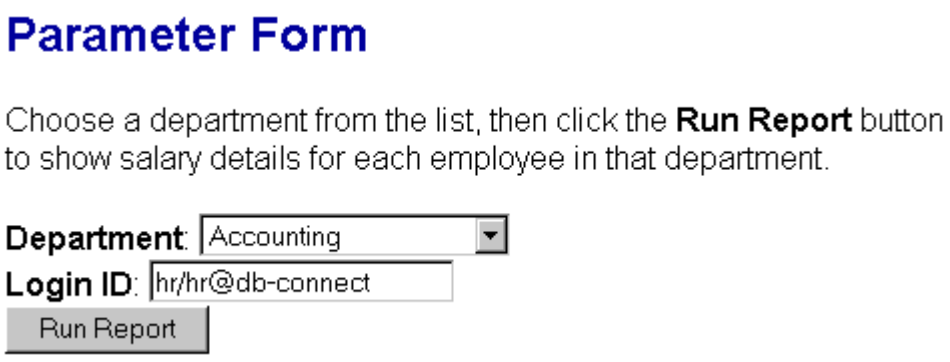
Now that we've modified and examined our Web source, let's view the Parameter Form in a Web browser.

1. Click the Run to Web button in the toolbar.

> **Note:** if Netscape 7.0 is your default browser, the browser may not display. You can work around this bug by making a copy of the Netscape 7.0 executable, naming it netscape.exe; with this name, the browser will display as expected.

2. The Parameter Form displays in your Web browser, and should look like the following:

Figure 39–5 Parameter form with values



Note: In the modified Parameter Form, notice how the list of values for the Department has changed from "a" to "Accounting."

3. Click on the **Department** drop-down list and notice how the list is now populated with department names. Although you can click the Run Report button, nothing will happen because we have not yet defined an action for it.

39.4 Set up the target report

Now that you've set up the parameters, the next step is to set up the target report to accept the parameters. Then, we will define the action for your Parameter Form. When a user clicks the Run Report button, the target report will be run based on the Department and User ID parameters.

The target report we use in this section is the sample report for the *Oracle Reports Tutorial*. If you completed the exercises in the *Oracle Reports Tutorial* and created "emprevb.jsp," you can use that report in this section. Otherwise, you can use the example file we've suppled, called `emprev_final.jsp`. We will not show the steps to build this report in this section.

For more information on building the sample JSP-based Web report, refer to the *Oracle Reports Tutorial*.

To set up the target report:

1. In Reports Builder, open the file we've provided called `emprev_final.jsp`.
2. In the Object Navigator, double-click the icon next to the Data Model node.
3. In the Data Model view that displays, double-click the query (**Q_1**) to display the SQL Query Statement dialog box.
4. Find the final line of the code:

```
WHERE (EMPLOYEES.MANAGER_ID = EMPLOYEES_A1.EMPLOYEE_ID)
AND EMPLOYEES.DEPARTMENT_ID = 100
```

5. Change the second WHERE clause so that the last line looks like this:

```
WHERE (EMPLOYEES.MANAGER_ID = EMPLOYEES_A1.EMPLOYEE_ID)
AND EMPLOYEES.DEPARTMENT_ID = :P_DEPARTMENT
```

> **Tip:** The code we've changed is in bold text.

6. Click **OK**.
7. A note displays that indicates a bind parameter has been created. Click **OK**.
8. Save your report as `emprev_param_<your initials>.jsp`.

39.5 Deploy the JSP report and Parameter Form

To deploy the JSP Parameter Form and the target JSP-based Web report, you must copy `paramform_<your initials>.jsp` and `emprev_final_<your initials>.jsp` to the deployment directory of your Application Server. For testing purposes, however, you can use the OC4J instance shipped with the Oracle Developer Suite. Once you've placed the target report in the desired directory, you can then modify the Parameter Form to point to the report location.

For more information on deploying a JSP-based Web report, refer to the *Oracle Application Server Reports Services Publishing Reports to the Web* manual.

To set up and deploy the JSP Parameter Form:

1. Copy the JSP Parameter Form and the Web report (`paramform_<your initials>.jsp` and `emprev_param_<your initials>.jsp`) into the following directory:

Oracle_Home\reports\j2ee\reports_ids\web

Note: *Oracle_Home* is the directory in which Reports Builder is installed.

2. In Reports Builder, open the file: *Oracle_Home*\reports\j2ee\reports_ids\web\paramform_<your initials>.jsp.
3. In the Web Source view, modify the action for the form so that when the user clicks the Run Report button, the emprev_param_<your initials>.jsp report executes based on the selected parameters. The line of code should look like this:

```
<form name="form1" method="post" action="/reports/emprev_param_<your
initials>.jsp">
```

Note: In the above code, the action attribute assumes that the report is located in the directory we specified in the above steps. For more information on deploying JSP-based Web reports and JSP Parameter Forms, refer to the *Oracle Application Server Reports Services Publishing Reports to the Web* manual.

4. Save your report.
5. Start your OC4J instance.
   - On Windows, you can do either of the following:
     - From the Start menu, choose **Programs>Oracle Developer Suite - *oracle_home_name* >Reports Developer>Reports Builder**
     - From the command line, execute:

       <*IDS_HOME*>\j2ee\DevSuite\startinst.bat
   - On UNIX, start the shell script:

     <*IDS_HOME*>/j2ee/DevSuite/startinst.sh

Tip: Once the containers for J2EE are initalized, the OC4J instance has been started.

6. In a Web browser, type the URL for the Parameter Form:

   ```
   http://<machine name>:8888/reports/paramform_<your
   initials>.jsp?userid=<userid>/<password>@<database name>
   ```

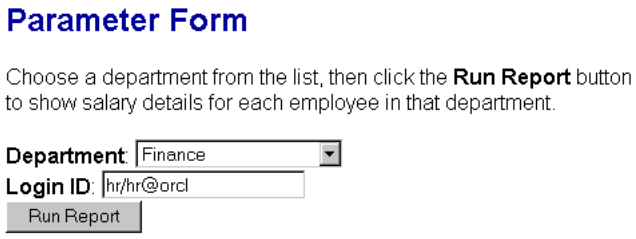
   In our example, we would use:

   ```
   http://mycomputer-pc:8888/reports/paramform.jsp?userid=hr/hr@orcl
   ```

   Note: The connect string you type in the URL is for the database you used to create the data model in Section 39.3.1, "Create a data model manually for the Parameter Form". For the purposes of this example, we've used plain text to pass the connect string. For information on using security, refer to the *Securing Oracle Reports* white paper, located on the Getting Started with Oracle Reports Web site on the Oracle Technology Network (`http://otn.oracle.com/products/reports/`). You can also find more information about security in the *Oracle Application Server Reports Services Publishing Reports to the Web* manual.

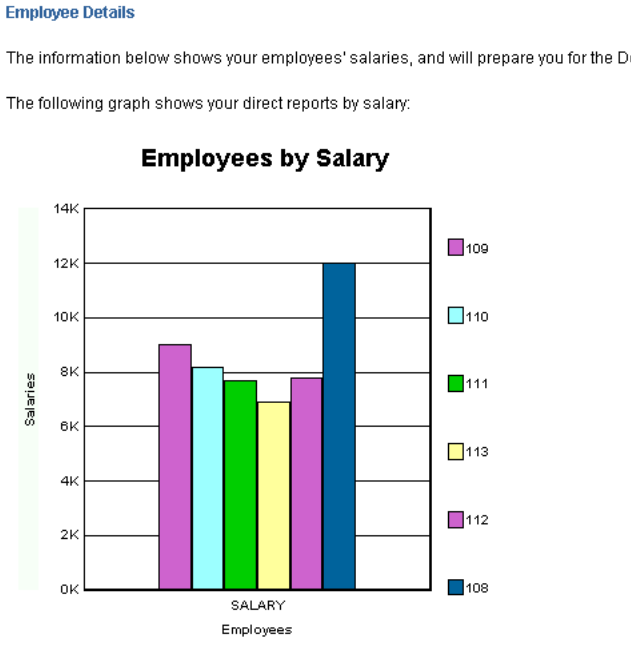
7. When the Parameter Form displays, choose a department from the drop-down list. For example, choose **Finance**.
8. In the **Login ID** field, type the connect string for the database schema that the `emprev_param_<your initials>.jsp` report uses. For example, `hr/hr@orcl`:

Figure 39–6 JSP Parameter Form with selections



9. Click the Run Report button in the toolbar.
10. The employee salary report displays with a graph at the top, and should look like this:

Figure 39–7 Sample employee report based on user parameters



Note: If you're not sure whether your report appears as it should, try using the files we've provided, `paramform.jsp` and `emprev_param.jsp` and follow all the steps in this section to deploy the Parameter Form and the Web report.

39.6 Summary

Congratulations! You have created a JSP Parameter Form for an existing Web report. You now know how to:

- create a simple Parameter Form in HTML.

- add a data model to a simple Parameter Form in Reports Builder.
- modify an existing JSP-based Web report to accept user parameters.
- test and deploy a JSP-based Web report with a JSP Parameter Form using OC4J.

For more information on deploying JSP-based Web reports, refer to the *Oracle Application Server Reports Services Publishing Reports to the Web* manual. For more information on any of the wizards, views, or properties used in this example, refer to the *Reports Builder Online Help*, which you can access in two ways:

- From the Oracle Technology Network (`http://otn.oracle.com/products/reports/`), click **Documentation** and navigate to the *Reports Builder Online Help* for the most recent, hosted online help.
- From Reports Builder, choose **Help > Help Contents**.

Note: You can replace your Reports Builder online help with the most recent update by downloading the latest online help set available on the Oracle Technology Network (`http://otn.oracle.com/docs/products/reports/`). Instructions for replacing your help file are included in the readme.txt in the download file.

40

Building a Report with a Barcode

Figure 40–1 Barcode JavaBean Web report output

Reports Builder enables you to create any type of report that displays barcodes. By using the Oracle Reports barcode JavaBean, you can build reports for the Web or for paper that display a barcode to make tasks like tracking shipping orders and employee identification numbers easier. In Reports 6*i*, you had to use a barcode font to generate the barcode. In Oracle Reports Builder, the JavaBean automatically generates the barcode for you.

To learn more about the barcode JavaBean, visit the Oracle Technology Network (`http://otn.oracle.com/products/reports/`), then click **Getting Started with Oracle Reports** and click **PL/SQL-Java Bridge** in the navigation bar.

Example Scenario

You will build two reports in this section, one for paper and one for the Web. The paper report shows an invoice for a single customer who has ordered multiple items from a company. The barcode indicates the tracking information for the order.

To build either of these reports, you must first refer to Section 40.1, "Prerequisites for this example".

Table 40–1 Features Demonstrated in this example

Feature	Location
Use the Java importer to add the barcode JavaBean for a paper report.	Section 40.2.1, "Import the Java classes into Reports Builder"
Use the Program Unit editor to create a PL/SQL package for a paper report.	Section 40.2.2, "Create a package to store your information"
Create a Before Report trigger to set up your barcode JavaBean for a paper report.	Section 40.2.3, "Create a Before Report trigger"
Use the Data Model view and toolbar to create a data model with a formula column for a paper report.	Section 40.2.4, "Create a data model with two formula columns"
Create a simple JSP-based Web report.	Section 40.3.1, "Create a query in an existing HTML file"
Create formula columns to call the barcode data for your Web report.	Section 40.3.2, "Create three formula columns in your data model"
Edit the JSP code in the Web source view.	Section 40.3.3, "Initialize the barcode JavaBean and set its properties"
View your JSP-based Web report in a browser.	Section 40.3.4, "Run your report to the Web"

40.1 Prerequisites for this example

To build the examples in this manual, you must have the example files we've provided, as well as access to the sample schema that is shipped with the Oracle9*i* database.

40.1.1 Example files

If you haven't already done so, you can download the files you'll need to complete this example from the Oracle Technology network and install them on your machine.

To download and install the example files:

1. Go to the Oracle Technology Network Web site (`http://otn.oracle.com/product/reports/`).
2. Click **Getting Started with Oracle Reports**.
3. Click **Index**, then find the "Building a Paper Report with a Barcode JavaBean" example and "Building a Web Report with a Barcode JavaBean" example. To complete this chapter, you need both sets of files. Note that this chapter covers how to build both a paper report and a Web report with a barcode bean.
4. Download the files `BarcodePaper.zip` and `BarCodeWeb.zip` into a temporary directory on your machine (e.g., `d:\temp`).
5. Unzip the contents of the file, maintaining the directory structure, into an examples directory on your machine (e.g., `d:\orawin90\examples`).

   This zip file contains the following files:

Table 40–2 Files necessary for building the barcode JavaBean sample reports

File	Description
Examples\BarCodeBeanPaper\result\ShippingManifest.pdf	The final PDF version of the paper report, containing the barcode.
Examples\BarCodeBeanPaper\scripts\oraclebarcode.jar	The barcode JavaBean.
Examples\BarCodeBeanPaper\scripts\barcode_code.txt	All the code used in this chapter, so you can copy and paste the code from this file instsead of typing it manually.
Examples\BarCodeBeanPaper\source\ShippingManifest.rdf	The source file for the sample paper report. Running this RDF in Reports Builder will display the final result of your paper report in the Paper Design view.
Examples\BarCodeBeanWeb\result\ShippingManifestWeb.jsp	The final JSP version of the Web report, containing the barcode.

Table 40–2 Files necessary for building the barcode JavaBean sample reports

File	Description
Examples\BarCodeBeanWeb\result\ShippingManifestWeb.rdf	The final RDF version of the Web report, containing the barcode.
Examples\BarCodeBeanWeb\result\assets	The images that Oracle Reports generated when the JSP was run.
Examples\BarCodeBeanWeb\scripts\SQL.txt	The SQL for the query you need to enter.
Examples\BarCodeBeanWeb\scripts\barcode_code.txt	All the code used in this chapter, so you can copy and paste the code from this file instsead of typing it manually.
Examples\BarCodeBeanWeb\source\ShippingLabel.html	The HTML page that you will use as a basis for the Web report.
Examples\BarCodeBeanWeb\source\ShippingManifestWeb.rdf	The source file for the sample Web report. Running this RDF in Reports Builder to the Web will display the final result of your Web report in your browser.
Examples\BarCodeBeanWeb\source\assets*	The images and other files that your JSP-based Web report will require to display properly on the Web.
Examples\BarCodeBeanWeb\source\assets\BLAFbeige_logo.gif	The image you will use in your JSP-based Web report.

40.1.2 Access to the sample schema

If you don't know if you have access to the sample schema provided with the Oracle9*i* database, contact your database administrator. You should have access to the "Order Entry" portion of the schema to complete this example.

40.1.3 Update the REPORTS_CLASSPATH environment variable

Before you use a Reports JavaBean (for paper or the Web), you need to perform several steps. You first need to set up your environment to use the correct classpath for the bean. For a paper report, you must then use the Java Importer to import the JavaBean into Reports Builder. For a Web report, you must call the JavaBean from your JSP-based (JavaServer Page) report.

In this section, you will update the Reports class path with the location of the JavaBean. When you launch Reports Builder, it will use this new class path to recognize the location of the barcode bean.

> **Note:** You must follow the steps in this section before you can even run the finished report we've provided, called `ShippingManifestPaper.rdf` and `ShippingManifestWeb.rdf`.

1. Find the class path:
   - In Windows, open the registry using regedit (you may want to export a backup before you modify your registry) and update the REPORTS_CLASSPATH environment variable.
   - On UNIX, update the `REPORTS_CLASSPATH` environment variable.
2. Modify the class path to reflect the location of the `oraclebarcode.jar` file, for example:

   ```
   ORACLE_HOME/Examples/BarCodeBeanPaper/Scripts/oraclebarcode.jar;
   ```

   or

   ```
   \ora9ids\reports\j2ee\reports_
   ids\web\examples\BarcodeBeanPaper\scripts\oracleb
   arcode.jar
   ```

   > **Note:** The path depends on where the files are located. If you downloaded the files from the Oracle Technology Network (`http://otn.oracle.com/products/reports/`), you may have to further edit this path. Also, when you edit the Windows registry, make sure you separate your entry with a semicolon (;), with no spaces on either side of the semicolon.

3. Save the new environment variable or exit the Windows registry.

You are now ready to begin building your report.

40.2 Create a barcode report for paper

In this section, you will create a paper-based report that shows the invoice for a particular customer. This invoice will display the address of the customer, his order, and a barcode that represents the tracking number for the order. The company can scan this barcode to find out the status of the order.

Next, you will import the JavaBean, then create a barcode report for paper (Acrobat PDF). If you want to learn how to create a barcode JSP-based report for the Web, skip to Section 40.3.

40.2.1 Import the Java classes into Reports Builder

To create a paper report using the barcode JavaBean, you must first import two Java classes into Reports Builder. When you import these Java classes, Reports Builder automatically creates the packages you need to build the report.

> **Note:** You do not need to perform this task if you're creating a Web report, as you will write JSP that calls the JavaBean.

To import the Java classes:

1. Launch Reports Builder

   > **Note:** You must launch Reports Builder now so that the new classpath is used.

2. Close the Welcome dialog box by clicking **Cancel**.
3. Choose **Program > Import Java Classes** to display the Import Java Classes dialog box.
4. Under Select Java Classes, navigate to:

   `oracle.apps.barcode.util.BarCodeConstants.`

   > **Note:** If you do not see this class listed, try exiting Reports Builder and make sure the `REPORTS_CLASSPATH` reads correctly. Then, launch Reports Builder again.

5. Select the class, then click **Import**.
6. Once the packages have been created, import the second JavaBean: `oracle.apps.barcode.BarCodeMaker`.
7. Click **Close**.
8. In the Object Navigator, under the report named **MODULE 1**, click the **Program Units** node. You'll notice that Reports Builder created two package specs and two package bodies named **BARCODECONSTANTS** and **BARCODEMAKER**.

40.2.2 Create a package to store your information

In this report, you want to create a package where the information will be stored.

To create a package for storing your information:

1. In the Object Navigator, under your new report, click the **Program Units** node.
2. Click the Create button in the toolbar to display the New Program Unit dialog box.
3. In the New Program Unit dialog box, type `globals`.
4. Select **Package Spec**, then click **OK** to display the PL/SQL Editor:
5. In the PL/SQL Editor, type the following code:

```
PACKAGE globals IS
    bcobj ora_java.jobject;
    barcode_to_use varchar2(256);
    tempdir varchar2(100);
    directory_sep varchar2(2);
END;
```

> **Note:** You can also enter this code by copying and pasting it from the provided text file called `barcode_code.txt`.

6. Click **Compile** to make sure there are no errors in your code.

Note: If your code does not compile, make sure you've typed in exactly the code we've provided.

7. Once the code is compiled, click **Close**.
8. In the Object Navigator, click your report name (e.g., **MODULE 1**).
9. Choose **File > Save**.
10. Name the file `shippingmanifest_<your initials>.rdf` (e.g., `shippingmanifest_vw`) and make sure you save it to a new directory (e.g., `My Examples`), where your new files are stored. Make sure you save the file in RDF format.
11. Click **Save**.

You have created a package that will contain the global information for your report.

40.2.3 Create a Before Report trigger

You can use the Before Report trigger to initialize specific tasks that will run before the report runs. Here, you will define the type of barcode you want to use in your report, as well as the temporary directory where your barcode images will be stored.

To create a Before Report trigger:

1. In the Object Navigator, under **SHIPPINGMANIFEST_<your initials>**, expand the **Report Triggers** node, then double-click the icon next to **BEFORE REPORT** to display the PL/SQL Editor
2. In the PL/SQL Editor, use the template to enter the following PL/SQL code:

```
function BeforeReport return boolean is
    begin
    globals.barcode_to_use := BarCodeConstants.BAR_CODE_128;
    globals.bcobj := barcodemaker.new();
return (TRUE);
end;
```

To modify the type of barcode you want to use, you can change the value `BarCodeConstants.BAR_CODE_128` to any other valid value. To determine which values are valid, check the contents of the package by opening the

BarCodeConstants package spec in the Object Navigator, under the **Program Units** node.

> **Note:** You can enter this code by copying and pasting it from the provided text file called `barcode_code.txt`.

3. Click **Compile** to make sure there aren't any errors.

> **Note:** If you have errors, make sure you've imported the necessary Java classes and that your code matches the code above. If you change the code, be sure to compile it again

4. When the code is compiled, click **Close**. Notice how the node icon next to the **BEFORE REPORT** trigger has changed.
5. Save your report.

You have created a trigger that will set up the barcode type for you when you run the report.

40.2.4 Create a data model with two formula columns

In this section, you will manually create the query that the report will use to retrieve data from the sample schema. You will also create a formula column that will communicate with the JavaBean to create the barcode, then return the file name of the generated image.

To create the query:

1. In the Object Navigator, under **SHIPPINGMANIFEST_<your initials>**, double-click the view icon next to the **Data Model** node to display the Data Model view for your report.
2. In the Data Model view, click the SQL Query tool in the tool palette, then click in an open area of the Data Model view to display the SQL Query Statement dialog box.
3. In the **SQL Query Statement** field, type (or paste) the following code:

```
SELECT ALL CUSTOMERS_A1.CUST_FIRST_NAME,
    CUSTOMERS_A1.CUSTOMER_ID, CUSTOMERS_A1.CUST_LAST_NAME,
    CUSTOMERS_A1.CUST_ADDRESS.STREET_ADDRESS,
    CUSTOMERS_A1.CUST_ADDRESS.POSTAL_CODE,
    CUSTOMERS_A1.CUST_ADDRESS.CITY,
    CUSTOMERS_A1.CUST_ADDRESS.STATE_PROVINCE,
    CUSTOMERS_A1.CUST_ADDRESS.COUNTRY_ID, ORDERS.ORDER_ID,
    ORDERS.ORDER_DATE,
    ORDERS.ORDER_TOTAL, ORDER_ITEMS.LINE_ITEM_ID,
    PRODUCTS.PRODUCT_NAME,
    ORDER_ITEMS.UNIT_PRICE, ORDER_ITEMS.QUANTITY,
    COUNTRIES.COUNTRY_NAME
    FROM CUSTOMERS CUSTOMERS_A1, ORDER_ITEMS, ORDERS,
    PRODUCTS, HR.COUNTRIES
WHERE ((ORDER_ITEMS.ORDER_ID = ORDERS.ORDER_ID)
    AND (ORDERS.CUSTOMER_ID = CUSTOMERS_A1.CUSTOMER_ID)
    AND (ORDER_ITEMS.PRODUCT_ID = PRODUCTS.PRODUCT_ID)
    AND (CUSTOMERS_A1.CUST_ADDRESS.COUNTRY_ID =
    HR.COUNTRIES.COUNTRY_ID))
    AND ORDERS.ORDER_ID = :P_ORDER_ID
ORDER BY order_ID, line_item_ID
```

Note: You can enter this query in any of the following ways:

- Copy and paste the code from the provided text file called `barcode_code.txt` into the **SQL Query Statement** field.
- Click **Query Builder** to build the query without entering any code manually.
- Type the code in the **SQL Query Statement** field.

4. Click **OK**.

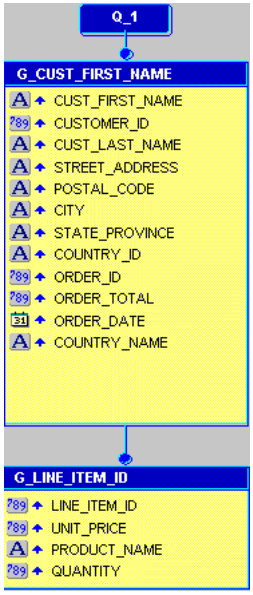
   If you are not connected to a database that contains the sample schema we've provided, you must log in now. If you're not sure what your connection string is, contact your database administrator. Note that this example uses the "Order Entry" portion of the sample schema.

5. When a message displays indicating that the bind parameter `p_order_id` was created, click **OK**.

6. In the data model you just created, select all of the following columns using Shift-click, then drag them below the current query into a detail group:
   - **LINE_ITEM_ID**
   - **PRODUCT_NAME**
   - **UNIT_PRICE**
   - **QUANTITY**

   The resulting data model should look like this:

Figure 40–2 Data Model for the query



40.2.5 Create a formula column to retrieve the barcode image

To create a formula column:

1. In the Data Model view, click the Formula Column tool in the tool palette.
2. Click in the master group (the main group that still contains most of the column names) to create a new formula column.

3. Double-click the new formula column object (**CF_1**) to display the Property Inspector, and set properties:
   - Under **Column**, set the Datatype property to Character.
   - Under **Placeholder/Formula**, click the PL/SQL Formula property field to display the PL/SQL Editor.
4. In the PL/SQL Editor, use the template to enter the following PL/SQL code:

```
function CF_1Formula return VarChar2 is
  myFileName varchar2(500);
  result varchar2(500);
  barcodeData VarChar2(50) := :customer_ID || :order_ID;
begin
myFileName := srw.create_temporary_filename;
  barcodemaker.setBarWidthInch(globals.bcobj, 0.005);
  barcodemaker.setBaseCodeData(globals.bcobj,barcodeData);
  barcodemaker.setBarCodeType(globals.bcobj,globals.barcode_to_use);
  barcodemaker.setFullPath(globals.bcobj, myFileName);
  barcodemaker.renderBarCode(globals.bcobj);
  return(myFileName);
end;
```

   Note: You can enter this code by copying and pasting it from the provided text file called `barcode_code.txt`.

5. Click **Compile** to make sure there aren't any errors.

   Note: If you have errors, make sure you've imported the necessary Java classes and that your code matches the code above. If you change the code, be sure to compile it again.

6. When the code is compiled, click **Close**.
7. Double-click the new formula column object (**CF_1**) to display the Property Inspector, and set properties:
   - Under **Column**, set the Read from File property to Yes, and set the File Format property to Image.

40.2.6 Create a formula column that returns the order total

To create the second formula column:

1. In the Data Model view, create a formula column in the detail group **G_LINE_ITEM_ID**.
2. Open the Property Inspector for the formula column.
3. Under **General Information**, set the Name property to `LineTotal`.
   - Under **Column**, make sure the Datatype property is set to Number.
   - Under **Placeholder/Formula**, click the PL/SQL Formula property field to display the PL/SQL Editor.
4. In the PL/SQL Editor, use the template to enter the following PL/SQL code:

```
function LineTotalFormula return Number is
begin
  return (:quantity * :unit_price);
end;
```

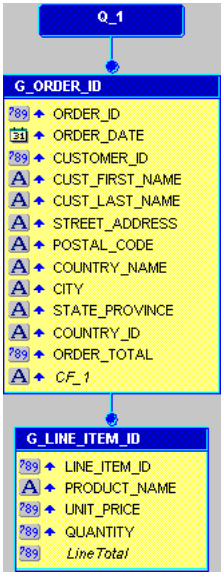
> **Note:** You can enter this code by copying and pasting it from the provided text file called `barcode_code.txt`.

5. Click **Compile** to make sure there aren't any errors.
6. When the code is compiled, click **Close**.
7. Save the report.

You have created the data model for your barcode report, which contains a formula column that retrieves the barcode information and displays the barcode image on your report, and another formula column that displays the order total.

Your data model and the PL/SQL for the formula column should look similar to this:

Figure 40–3 Data Model with two new formula columns



40.2.7 Create a layout for your report

Before you can run your report, you must create a layout.

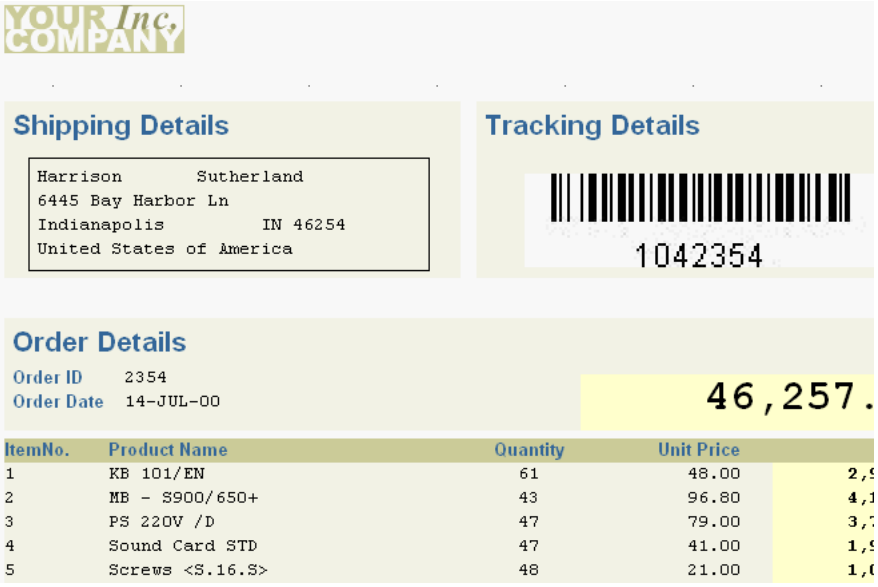
To create a paper layout:

1. Under your report's node in the Object Navigator, right-click **Paper Layout**, then choose **Report Wizard**.
2. In the Report Wizard, on the **Report Type** page, select **Create both Web and Paper Layout**, then click **Next**.
3. On the **Style** page, select **Group Above**, then click **Next**.
4. On the **Data Source** page, click **Next**.
5. On the **Data** page, click **Next**.
6. On the **Groups** page, make sure the following fields are listed in the **Group Fields** list (if not, use the arrows to move the field to the appropriate list):
   - **ORDER_ID**
   - **ORDER_DATE**

- **CUSTOMER_ID**
- **CUST_FIRST_NAME**
- **CUST_LAST_NAME**
- **STREET_ADDRESSS**
- **POSTAL_CODE**
- **COUNTRY_NAME**
- **CITY**
- **STATE_PROVINCE**
- **COUNTRY_ID**
- **ORDER_TOTAL**
- **CF_1**

7. On the **Fields** page, click the double right arrows (**>>**) to move all of the fields to the **Displayed Fields** list, then click **Finish**.
8. In the Paper Layout view, click the Run Paper Layout button in the toolbar to run your report.
9. In the Runtime Parameter Form, next to **P_ORDER_ID**, type `2354`.
10. Once your report displays in the Paper Design view, you can rearrange your layout objects in the Paper Layout view to make your report look something like this:

Figure 40–4 Paper Design view of the barcode paper report



Note: If you aren't sure whether you produced the desired results, you can always open the file we provided, called `ShippingManifest.pdf` in Acrobat Reader. Or, you can run `ShippingManifest.rdf` to paper and the report will display in the Paper Design view.

11. Save the report. You have now finished building a barcode report for paper.

40.3 Create a barcode report for the Web

The steps in this section show you how to build a Web report using JavaServer Pages (JSPs), using the barcode JavaBean you imported in Section 40.2.1, "Import the Java classes into Reports Builder". If you want to build a paper report with a barcode, see Section 40.2, "Create a barcode report for paper".

If you are not familiar with creating a JSP-based Web report and would like to learn how to create one, refer to the *Oracle Reports Tutorial*, located in the **Getting Started**

with Oracle Reports Web site on the Oracle Technology Network (`http://otn.oracle.com/reports/`).

The report you will create in this section is the same as the one you created for paper. You will create a report that displays the invoice for a particular customer. This invoice will display the address of the customer, his order, and a barcode that represents the tracking number for the order. The company can use this barcode to find out the status of the order.

You can run the final version of the JSP report we've provided to see what you'll build in these steps, but please note that you will need to update the location of the images in the source code (see Section 40.3.3, "Initialize the barcode JavaBean and set its properties") before you can run the report to the Web.

Note: Before you begin this section, make sure you have all the necessary files, and that you've imported the Java classes, and set up the class path. See Section 40.1, "Prerequisites for this example" and Section 40.2.1, "Import the Java classes into Reports Builder".

40.3.1 Create a query in an existing HTML file

When you create a JSP-based Web report, you can use an existing HTML file as a template. The steps in this section will show you how to open an HTML file in Reports Builder and add data to it.

To create a query in an existing HTML file:

1. In Reports Builder, choose **File > Open** and open the file *Examples*`\BarCodeBeanWeb\source\ShippingLabel.html`.
2. In the Object Navigator, under **SHIPPINGLABEL**, double-click the view icon next to the **Data Model** node to display the Data Model view for the report.
3. In the Data Model view, click the SQL Query tool in the tool palette, then click in an open area of the Data Model view to display the SQL Query Statement dialog box.
4. In the **SQL Query Statement** field, type (or paste) the following code:

```
SELECT ALL CUSTOMERS_A1.CUST_FIRST_NAME,
    CUSTOMERS_A1.CUSTOMER_ID, CUSTOMERS_A1.CUST_LAST_NAME,
    CUSTOMERS_A1.CUST_ADDRESS.STREET_ADDRESS,
    CUSTOMERS_A1.CUST_ADDRESS.POSTAL_CODE,
    CUSTOMERS_A1.CUST_ADDRESS.CITY,
```

```
    CUSTOMERS_A1.CUST_ADDRESS.STATE_PROVINCE,
    CUSTOMERS_A1.CUST_ADDRESS.COUNTRY_ID,
    ORDERS.ORDER_ID, ORDERS.ORDER_DATE, ORDERS.ORDER_TOTAL,
    COUNTRIES.COUNTRY_NAME FROM CUSTOMERS CUSTOMERS_A1, ORDERS,
    HR.COUNTRIES
WHERE ((ORDERS.CUSTOMER_ID = CUSTOMERS_A1.CUSTOMER_ID)
AND (CUSTOMERS_A1.CUST_ADDRESS.COUNTRY_ID = HR.COUNTRIES.COUNTRY_ID))
AND ORDERS.ORDER_ID = :P_ORDER_ID ORDER BY order_ID
```

Note: You can enter this query in any of the following ways:

- Copy and paste the code from the provided text file called `barcode_code.txt` into the **SQL Query Statement** field.
- Click **Query Builder** to build the query without entering any code manually.
- Type the code in the **SQL Query Statement** field.

5. Click **OK**.

   If you are not connected to a database that contains the sample schema we've provided, you must log in now. If you're not sure what your connection string is, contact your database administrator. Note that this example uses the "Order Entry" portion of the sample schema.

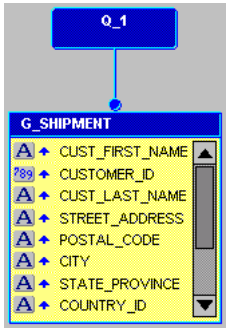
6. When a message displays indicating that the bind parameter was created, click **OK**.

7. In the Data Model view, double-click the group object to display the Property Inspector, and set properties:

   - Under **General Information**, set the Name property to G_SHIPMENT.

   Your data model should look something like this:

Figure 40–5 Data Model for the JSP-based Web report query



8. Save your report as `ShippingLabel_<your initials>.jsp` to create the JSP-based Web source for this report.

You have now created the query that will pull in the data for your report.

40.3.2 Create three formula columns in your data model

You will need to create three formula columns in your report to retrieve the tracking number for the order, the origin of the order, and the destination for the order.

To create the TrackingNumber formula column:

1. In the Data Model view, click the Formula Column tool in the tool palette.
2. Click in the **G_SHIPMENT** group to create a new formula column.
3. Double-click the new formula column object (**CF_1**) to display the Property Inspector, and set properties:
   - Under **General Information**, set the Name property to TrackingNumber.
   - Under **Column**, set the Datatype property to Character.
   - Under **Placeholder/Formula**, click the PL/SQL Formula property field to display the PL/SQL Editor.
4. In the PL/SQL Editor, use the template to enter the following PL/SQL code:

```
function TrackingNumberFormula return char is
begin
    return(:Customer_id||:Order_ID||:country_ID);
```

```
end;
```

> **Note:** You can enter this code by copying and pasting it from the provided text file called `barcode_code.txt`.

5. Click **Compile**.

> **Note:** If your code does not compile, make sure you've typed in exactly the code we've provided.

6. When the code is compiled, click **Close**.

To create the OriginScan formula column:

1. Create another formula column in the **G_SHIPMENT** group.
2. Double-click the new formula column object (**CF_1**) to display the Property Inspector, and set properties:
   - Under **General Information**, set the Name property to OriginScan.
   - Under **Column**, set the Datatype property to Character.
   - Under **Placeholder/Formula**, click the PL/SQL Formula property field to display the PL/SQL Editor.
3. In the PL/SQL Editor, use the template to enter the following PL/SQL code:

```
function OriginScanFormula return char is
begin
    return('34324-OH-US');
end;
```

> **Note:** You can enter this code by copying and pasting it from the provided text file called `barcode_code.txt`.

4. Click **Compile**.

Note: If your code does not compile, make sure you've typed in exactly the code we've provided.

5. When the code is compiled, click **Close**.

To create the DestinationScan formula column:

1. Create a third formula column in the **G_SHIPMENT** group.
2. Double-click the new formula column object (**CF_1**) to display the Property Inspector, and set properties:
   - Under **General Information**, set the Name property to DestinationScan.
   - Under **Column**, set the Datatype property to Character.
   - Under **Placeholder/Formula**, click the PL/SQL Formula property field to display the PL/SQL Editor.
3. In the PL/SQL Editor, use the template to enter the following PL/SQL code:

```
function DestinationScanFormula return char is
begin
    return(:postal_code||'-'||:state_province||'-'||:country_ID);
end;
```

Note: You can enter this code by copying and pasting it from the provided text file called `barcode_code.txt`.

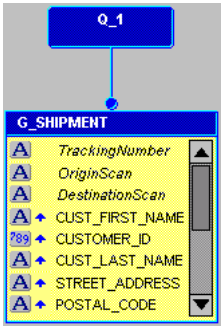
4. Click **Compile**.

Note: If your code does not compile, make sure you've typed in exactly the code we've provided.

5. When the code is compiled, click **Close**.
6. Save your report as a JSP.

You have created the three formula columns that will hold the values for the tracking number, the origin, and the destination of the order. Your data model should look something like this:

Figure 40–6 Data Model with the three formula columns



40.3.3 Initialize the barcode JavaBean and set its properties

To enable your JSP-based Web report to communicate with the JavaBean, you need to initialize it in the JSP. Unlike using the JavaBean with a paper report, you do *not* need to use the Java importer to import the Java classes.

In this section, you will also learn how to set the properties for the bean so that the correct data is being used to produce the barcode.

If you don't want to bother with typing the code in yourself, you can always open the source file (*Examples*/BarCodeBeanWeb/source/ShippingManifestWeb.rdf) and copy the appropriate pieces of the Web source into your report.

To ensure that the JavaBean references the correct barcode images, you must first update your report with the correct path.

To update the paths:

1. In the Object Navigator, double-click the view icon next to the **Web Source** node for your JSP-based Web report (**ShippingLabel_<your initials>**) to display the Web Source view.
2. In the Web Source view, look for the text: `Define Path information for your barcode images.`

3. Under this text, update the paths to make sure they point to the directories where your images will be generated (e.g., `d:\\temp\\docroot\\images\\`). Be sure to maintain the integrity of the paths we've provided.

> **Note:** The directory where these files are located must be accessible by the Web server. This directory is typically located somewhere below the directory where you keep your JSPs. Make sure that this directory exists on your computer before you run the report to the Web.

To initialize the JavaBean:

1. In the Web Source view, look for the text: `Initialize the JavaBeans.`
2. Under this text, type the following code:

```
<jsp:useBean id="BC" scope="page"
class="oracle.apps.barcode.util.BarCodeConstants" />
<jsp:useBean id="BM" scope="page" class="oracle.apps.barcode.BarCodeMaker"
/>
```

> **Note:** You can enter all the code in this section by copying and pasting it from the provided text file called `barcode_code.txt`.

To set the barcode JavaBean properties:

1. In the Web Source view, look for the text: `Setting the barcodes properties.`
2. Under this text, type the following code:

```
<jsp:setProperty name="BM" property="BarCodeType" value="<%= BC.BAR_CODE_128
%>" />
<jsp:setProperty name="BM" property="BarWidthInch" value="0.01"/>
<jsp:setProperty name="BM" property="Directory" value="<%=
BarcodePhysicalPath %>"/>
```

To define the barcode variables:

1. In the Web Source view, look for the text: `Define variables to hold the data for the three barcodes.`

2. Under this text, type the following code:

```
<%! private String BarCodeData1 = "12345-XX-XX"; %>
<%! private String BarCodeData2 = "12345-XX-XX"; %>
<%! private String BarCodeData3 = "12345-XX-XX"; %>
```

To create a For Each loop:

1. In the Web Source view, look for the text:
   `Replace this with your RW:FOREACH open tag.`
2. Replace this line with the following code:

```
<rw:foreach id="R_G_SHIPMENT" src="G_SHIPMENT">
```

3. Look for the text:
   `Replace this with your RW:FOREACH close tag.`
4. Replace this line with the following code:

```
</rw:foreach>
```

To code the formula columns to render the barcodes:

1. In the Web Source view, look for the text:
   `**BARCODEShippingTrackingNumber**.`
2. Under this text, type the following code:

```
<!-- Get the value of the TrackingNumber and assign it to the variable -->
<rw:getValue id="BarCodeData1" src="TrackingNumber"/>
<!-- Set the data for tbe barcode and the filename -->
<jsp:setProperty name="BM" property="BaseCodeData" value="<%= BarCodeData1
%>"/>
<jsp:setProperty name="BM" property="FileName" value="<%= BarCodeData1 %>"/>
<!-- Render the barcode -->
<% BM.renderBarCode(); %>
<!-- View the image in the page -->
<img src="assets/barcodes/<%= BarCodeData1 %>">
```

Note: the lines beginning with "<!--" are comments. If you do not want comments in your code, you do not have to add these lines.

3. In the Web Source view, look for the text: `**BARCODEOriginScan**`.
4. Under this text, type the following code:

```
<!-- Get the value of the OriginScan and assign it to the variable -->
<rw:getValue id="BarCodeData2" src="OriginScan"/>
<!-- Set the data for tbe barcode and the filename -->
<jsp:setProperty name="BM" property="BaseCodeData" value="<%= BarCodeData2
%>"/>
<jsp:setProperty name="BM" property="FileName" value="<%= BarCodeData2 %>"/>
<!-- Render the barcode -->
<% BM.renderBarCode(); %>
<!-- View the image in the page -->
<img src="assets/barcodes/<%= BarCodeData2 %>">
```

5. In the Web Source view, look for the text: `**BARCODEDestinationScan**`.
6. Under this text, type the following code:

```
<!-- Get the value of the DestinationScan and assign it to the variable -->
<rw:getValue id="BarCodeData3" src="DestinationScan"/>
<!-- Set the data for tbe barcode and the filename -->
<jsp:setProperty name="BM" property="BaseCodeData" value="<%= BarCodeData3
%>"/>
<jsp:setProperty name="BM" property="FileName" value="<%= BarCodeData3 %>"/>
<!-- Render the barcode -->
<% BM.renderBarCode(); %>
<!-- View the image in the page -->
<img src="assets/barcodes/<%= BarCodeData3 %>">
```

7. Save your report as a JSP.

40.3.4 Run your report to the Web

Since you've created a Web report, you must run this report to the Web to see your results.

1. In the Object Navigator, click your report name, **ShippingManifestWeb_<your initials>**.
2. Click the Run Web Layout button in the toolbar to run your report to a browser.

Note: If Netscape 7.0 is your default browser, the browser may not display. You can work around this bug by making a copy of the Netscape 7.0 executable, naming it netscape.exe; with this name, the browser will display as expected.

Your report displays in your Web browser, and should look something like this:

Figure 40–7 Snapshot of the final JSP-based Web report with barcode

Note: If you aren't sure whether you produced the desired results, you can always open the file we provided in the `results` directory, called `ShippingManifestWeb.html` in your Web browser. Or, once you've updated the images path, you can run `ShippingManifestWeb.jsp` to the Web, and the report will display in the your browser.

40.4 Summary

Congratulations! You have created a paper report and a JSP-based Web report that use the barcode JavaBean to generate barcode images.

You now know how to:

- use the Java importer to add Java classes to a paper report.
- use JavaServer Pages to call a JavaBean from within a report.

- create a PL/SQL package.
- use a Before Report trigger to tell Reports Builder what type of barcode image to use.
- manually build a data model with a SQL query and formula columns.
- edit the code in the Web Source view for a JSP-based Web report.
- set up a JavaBean in JSP.

For more information on any of the wizards, views, or properties used in this example, refer to the *Reports Builder Online Help*, which you can access in two ways:

- From the Oracle Technology Network (`http://otn.oracle.com/products/reports/`), click **Documentation** and navigate to the *Reports Builder Online Help* for the most recent, hosted online help.
- From Reports Builder, choose **Help > Help Contents**.

> **Note:** You can replace your Reports Builder online help with the most recent update by downloading the latest online help set available on the Oracle Technology Network (`http://otn.oracle.com/docs/products/reports/`). Instructions for replacing your help file are included in the readme.txt in the download file.

Part VII

Building Reports with Pluggable Data Sources

This part introduces reports that use data sources other than the database. These reports use data sources available with Oracle Reports or combine information from one or more of these data sources to publish meaningful information.

This part contains the following chapters:

- Chapter 41, "Building a Report with an XML Pluggable Data Source"

  A report with an XML pluggable data source enables you to connect to any data source that is available with Oracle Reports, for example XML data. Usage of alternate data sources is possible due to the pluggable data source (PDS) architecture of Reports Builder.

- Chapter 42, "Building a Report with a Text Pluggable Data Source"

  A report can contain a text pluggable data source-such as a report that uses character-delimited text as a data source.

- Chapter 43, "Building a Report Using Oracle Express Data"

  An Express report is a report based on an Oracle Express pluggable data source. This report enables you to deliver results of on-line analytical processing (OLAP) by using a multidimensional data model. You can refine the data model and Express query.

41

Building a Report with an XML Pluggable Data Source

Figure 41–1 Report output using an XML PDS

Warehouse ID6	Warehouse Name Sydney		City Sydney
State New South Wales	Country Australia		
WAREHOUSE_ID	PRODUCT_ID	QUANTITY_ON_HAND	PRODUCT_NAME
6	1733	29	PS 220V /UK
6	1734	30	Cable RS232 10/AM
6	1737	30	Cable SCSI 10/FW/ADS
6	1738	30	PS 110V /US
6	1739	30	SDRAM - 128 MB
6	1740	30	TD 12GB/DAT
6	1742	31	CD-ROM 500/16x
6	1745	31	Cable SCSI 20/WD-
6	1748	32	PS 220V /EUR
6	1749	32	DIMM - 256MB
6	1750	32	DIMM - 2GB
6	1755	33	32MB Cache /NM

Reports Builder enables you to use any data source you wish. In this chapter, you will learn how to use the XML pluggable data source that is provided with Oracle Reports.

About Pluggable Data Sources

The information you must publish is often derived from data in various corporate data sources. These data sources may be SQL-based (relational databases) or non-SQL-based, such as XML, OLAP, and the like. Often, you must combine data from one or more of these data sources to publish meaningful information. For

example, you may need to combine data that exists in a relational database with data from a multidimensional database to compare trends and performance.

Oracle Reports enables you to leverage capabilities, such as aggregation, summarization, formatting, and scheduling, on data from any data source. You can leverage the PDS (pluggable data source) architecture to connect to your own data source, as well as to the data sources available with Oracle Reports (XML, JDBC, text, and Express).

For more information on pluggable data sources, refer to the *Reports Builder online help* and the Javadoc documentation for the PDS APIs.

Example Scenario

Suppose you have an international business with warehouses in the United States and overseas. These warehouses are running a de-centralized management system that stores the operational data locally at each site. The inventory of the warehouses are managed by the local managers. However, for planning purposes, a team at corporate headquarters needs to access the inventory data (in SQL), including the most recent data, of every warehouse. The warehouse data is only available as an XML stream. In this example, you will learn how to combine data from a local database (i.e., the warehouse data) and data from an XML feed to create a Web report.

In this example, you will use static XML files that we've provided for you. The report will access the XML feed online using the business-to-business interface of your order entry system.

Table 41–1 Features Demonstrated in this example

Feature	Location
Manually create a SQL query.	Section 41.2.1, "Create a SQL query for your new report"
Use the Data Wizard to create an XML query.	Section 41.2.2, "Create an XML query to access your XML data source"
Create a data link between a SQL query and an XML query.	Section 41.2.3, "Create a data link between two queries"
Use the Report Wizard to create a layout for your report.	Section 41.2.4, "Create a layout for your report using the Report Wizard"
Use format triggers and procedures to apply alternating row colors.	Section 41.2.5, "Apply alternating row colors to your report"

Table 41–1 Features Demonstrated in this example

Feature	Location
Use a group filter to sort your XML data.	Section 41.2.6, "Filter your XML data using groups"

41.1 Prerequisites for this example

To build the examples in this manual, you must have the example files we've provided, as well as access to the sample schema that is shipped with the Oracle9*i* database.

41.1.1 Example files

If you haven't already done so, you can download the files you'll need to complete this example from the Oracle Technology network (`http://otn.oracle.com/products/reports/`) and install them on your machine.

To download and install the example files:

1. Go to the Oracle Technology Network Web site (`http://otn.oracle.com/products/reports/`).
2. Click **Getting Started with Oracle Reports**.
3. Click **Index**, then browse through the list of examples and find the "Building a Report using an XML Pluggable Data Source" example.
4. Download the file XML_PDS.zip into a temporary directory on your machine (e.g., `d:\temp`).
5. Unzip the contents of the file, maintaining the directory structure, into an examples directory on your machine (e.g., `d:\orawin90\examples`).

   This zip file contains the following files:

Table 41–2 Files necessary for building this sample report using XML PDS

File	Description
Examples\XML_PDS\result\inventory_report.pdf	The final PDF version of the paper report.
Examples\XML_PDS\result\inventory_report.rdf	The final RDF version of the paper report.

Table 41–2 Files necessary for building this sample report using XML PDS

File	Description
Examples\XML_PDS\scripts\XMLPDS_code.txt	The various SQL statements you will use in this report.
Examples\XML_PDS\scripts\warehouse_inventory.xml	The XML data source for the query in your report.
Examples\XML_PDS\scripts\warehouse_inventory.xsd	The XML data stream for your report.

Note: The index.html file and assets directory are used as part of the **Getting Started with Oracle Reports** Web site. Please do not delete or move these files

41.1.2 Access to the sample schema

If you don't know if you have access to the sample schema provided with the Oracle9*i* database, contact your database administrator. You should have access to the "Order Entry" portion of the schema to complete this example. Typically, you can log into this schema by using the user ID and password "`oe/oe`", then enter the name of the database.

41.2 Create a report manually with SQL and XML queries

When you create a report, you can either use the Report Wizard to assist you or create the report yourself. To build this report, you'll need to create two queries: a SQL query and an XML query.

41.2.1 Create a SQL query for your new report

When creating the SQL query, you'll need access to the Order Entry part of the sample schema provided with the Oracle9*i* database. If you don't have access, contact your database administrator. Typically, you can log in using the connection string "`oe/oe@<database name>`".

To create a SQL query:

1. Launch Reports Builder (or, if already open, choose **File > New > Report**)

2. In the Welcome or New Report dialog box, select **Build a new report manually**, then click **OK**.

   Your new report displays in the Object Navigator as something like "MODULE 2." You will also see the Data Model view of your new report.

3. In the Data Model view, click the SQL Query tool in the tool palette, then click in an open area of the Data Model view to display the SQL Query Statement dialog box.

4. In the **SQL Query Statement** field, type the following code:

```
select      W.WAREHOUSE_ID,
            W.WAREHOUSE_NAME,
            L.CITY,
            L.STATE_PROVINCE,
            C.COUNTRY_NAME
from        WAREHOUSES W,
            HR.LOCATIONS L,
            HR.COUNTRIES C
where       ( W.LOCATION_ID = L.LOCATION_ID (+)  )
  and       ( L.COUNTRY_ID = C.COUNTRY_ID (+)  )
order by  C.COUNTRY_NAME, W.WAREHOUSE_NAME
```

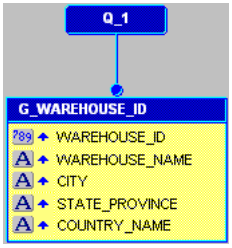
> **Note:** You can also copy and paste the code from the text file we've provided, `xmlpds_code.txt`. Open the file in a text editor, then copy the List of Warehouse query into the SQL Query Statement field.

5. Click **OK**.

> **Note:** If the Connect dialog box displays, enter the user ID, password, and name of the database that contains the sample schema.

The data model displays in the Data Model view, and should look something like this:

Figure 41–2 Data Model for the XML PDS example SQL query



6. Save your report as `inventoryreport_xml_<your initials>.rdf`.

You have created a SQL query to retrieve the data for your report.

41.2.2 Create an XML query to access your XML data source

In this section, you will create a query to access the XML data source. You can view the resulting report we've provided to make sure your query is correct. Please note, though, that you must update the paths to the Data Definition files with the location of the example files we provided to you.

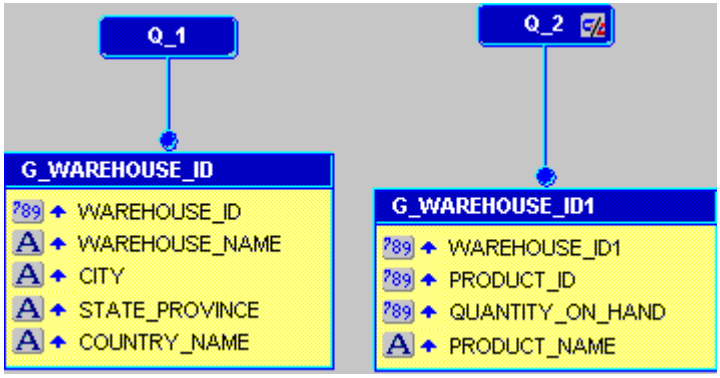
To create an XML query:

1. In the Data Model view, choose **Insert > Query** to display the Data Wizard.
2. If the Data Wizard Welcome page displays, click **Next**.
3. On the Query page, click **Next**.
4. On the Data Source page, click **XML Query**, then click **Next**.
5. On the Data page, click **Query Definition** to display the Define XML Query dialog box.
6. In the Define XML Query dialog box, under **Data Definition**, click **Browse** to locate the XSD file we've provided, `warehouse_inventory.xsd` and open it.
7. Under **Data Source**, click **Browse** to locate the XML file we've provided that contains your data, `warehouse_inventory.xml` and open it.

   If you want to compare your data definition to the one we provided, make sure that you replace the data definition locations with the locations of your files.
8. Click **OK**.

9. In the Data Wizard, still on the Data page, click **Next**.
10. Click **Finish** to display your data model in the Data Model view. It should look something like this:

Figure 41–3 Data model for the XML PDS example with XML and SQL queries



11. Save your report.

You have created an XML query to access the XML data source we've provided.

41.2.3 Create a data link between two queries

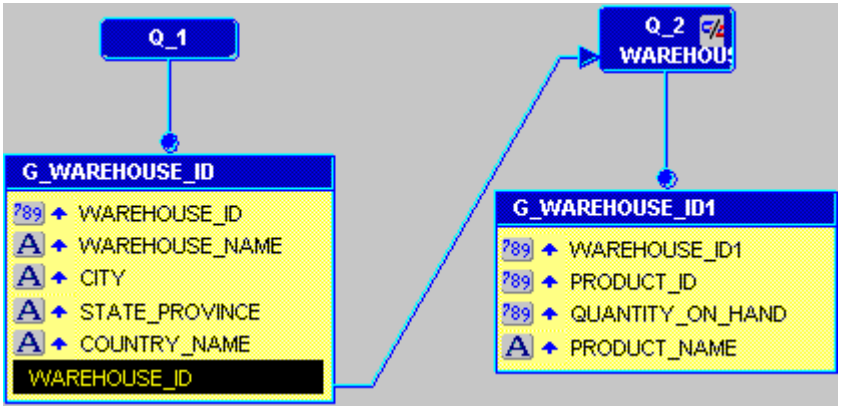
You will now need to link the SQL query and the XML query so that you can access your corporate data as well as the data for each of the local warehouses.

To create a data link:

1. In the Data Model view, click the Data Link tool in the tool palette.
2. Click the **WAREHOUSE_ID** column in your first query (**Q_1**).
3. Drag your cursor until it is over the **WAREHOUSE_ID1** column in the second query (**Q_2**).

   Your data model should now look something like this:

Figure 41–4 Data Model with a data link between a SQL query and an XML query



You'll notice that the WAREHOUSE_ID column is now highlighted at the bottom of Q_1, with a line pointing to the WAREHOUSE_ID1 column.

4. Save your report.

You have created a data link between the WAREHOUSE_ID columns in the two queries.

41.2.4 Create a layout for your report using the Report Wizard

Before you can run any report, you must define a layout. The easiest way to do this is to use the Report Wizard.

To create a paper layout:

1. In the Data Model view, right-click on the canvas, then choose **Report Wizard**.
2. In the Report Wizard, on the **Report Type** page, select **Create Paper Layout only**, then click **Next.**
3. On the **Style** page, select **Group Above**.
4. On the **Groups** page, make sure the **G_WAREHOUSE_ID** and **G_WAREHOUSE_ID1** groups are listed in the **Group Fields** list with a **Down** Print Direction.

5. On the **Fields** page, click the double right arrows (>>) to move all of the fields to the **Displayed Fields** list.
6. On the **Labels** page, adjust the labels as desired.
7. On the **Template** page, select **Predefined Template** and click **Beige**, then click **Finish** to display your report output in the Paper Design view. It should look something like this:

Figure 41–5 Paper Design view for the XML PDS report

Warehouse ID6 Warehouse Name Sydney City Sydney
State New South Wales Country Australia

WAREHOUSE_ID	PRODUCT_ID	QUANTITY_ON_HAND	PRODUCT_NAME
6	1820	69	SPNIX3.3 - NL
6	2414	59	HD 9.1GB @10000 /I
6	2417	29	Client ISO CP - V
6	2395	56	32MB Cache /M
6	2396	57	EDO - 32MB
6	2371	71	C for SPNIX4.0 - Doc
6	2492	41	SPNIX3.3 AU
6	2270	64	Modem - 56/90/I
6	1742	31	CD-ROM 500/16x
6	2596	51	SS Stock - 1mm
6	2430	65	Compact 400/LQ
6	2319	44	Screws

8. Save your report.

You have created the layout for your paper report.

> **Note:** You can also run the report we've provided in the result directory, called `inventory_report.rdf`. Before you can run the report, double-click the XML query in the Data Model view, and point the XML data source to the appropriate XSD and XML files.

41.2.5 Apply alternating row colors to your report

Now that you've created the report, you can make it more user-friendly by using a summary column to apply alternating row colors.

To create a summary column to count the rows:

1. In the Data Model view, click the Summary Column tool in the tool palette.

   If you are still in the Paper Design view, you can click the Data Model button in the toolbar to display the Data Model view.

2. Click in the XML query group (**G_WAREHOUSE_ID1**) to create a summary column.

3. Double-click the new summary column object (**CS_1**) to display the Property Inspector, and set properties:

   - Under **General Information**, set the Name property to LineNo.
   - Under **Column**, make sure the Column Type property is set to Summary, and that the Datatype property is set to Number.
   - Under **Summary**, set the Function property to Count, and set the Source property to PRODUCT_NAME.

To create a procedure that changes the row colors:

1. In the Object Navigator, click the **Program Units** node for your report.
2. Click the Create button in the toolbar to display the New Program Unit dialog box.
3. In the New Program Unit dialog box, type `linecolors` as in the **Name** field.
4. Select **Procedure,** and click **OK** to display the PL/SQL Editor for the new program unit.
5. In the PL/SQL Editor, enter the following PL/SQL code to change the text color of the alternating rows to blue:

```
PROCEDURE LineColors IS
BEGIN
  if (:LineNo mod 2=0)
  then
  srw.set_text_color('blue');
  else srw.set_text_color('black');
  end if;
END;
```

Note: You can copy and paste this code from the procedure provided in the `xmlpds_code.txt` file. Just copy the text under `Line Colors Procedure`.

6. Click **Compile** to compile the procedure.

   If any errors display, make sure the code is correct, and that you created the summary column in the steps above.

7. Click **Close**.

Note: Optionally, you can also change the fill colors of the alternating rows by following the steps in the above section, and using the following PL/SQL code. In this example code, we've changed the fill color of alternating rows to red and blue:

```
PROCEDURE LineColors IS
BEGIN
  if (:LineNo mod 2=0)
   then
     srw.set_foreground_fill_color(blue);
     srw.set_fill_pattern('solid');
  else
   srw.set_foreground_fill_color(red);
   srw.set_fill_pattern('solid');
  end if;
END;
```

To create a format trigger for each field that calls the procedure:

1. In the Object Navigator, under your report name, expand the **Paper Layout** node and navigate to **Body > M_G_WAREHOUSE_ID_GRPFR > R_G_WAREHOUSE_ID > M_G_WAREHOUSE_ID1_GRPFR > R_G_WAREHOUSEID1**.
2. While **F_PRODUCT_ID** is selected, press F4 to display the Property Inspector.

Note: If you can't find a particular field, use the **Find** field at the top of the Object Navigator.

3. In the Property Inspector, under **Advanced Layout**, click the Format Trigger property field to display the PL/SQL Editor.
4. In the PL/SQL Editor, use the template to enter the following PL/SQL code:

```
function F_PRODUCT_IDformatTrigger return Boolean is
begin
    LineColors;
return (TRUE);
end;
```

> **Note:** Make sure you do not touch the boldface text. Simply type in the code below this text to create the format trigger. You can copy and paste this code from the procedure provided in the `xmlpds_code.txt` file. Just copy the text under `Format Trigger Code.`

5. Add a format trigger for the following fields, using the same code as in the previous step. Be sure not to delete the first line of the code, where the format trigger name is defined:
   - **F_PRODUCT_NAME**
   - **F_QUANTITY_ON_HAND**
   - **F_WAREHOUSE_ID1**
6. Save your report.
7. Click the Run Paper Layout button in the toolbar to your report to paper. Your report should look something like this:

Figure 41–6 XML PDS Report with Alternating Row Colors

Warehouse ID6	Warehouse Name Sydney		City Sydney
State New South Wales	Country Australia		
WAREHOUSE_ID	PRODUCT_ID	QUANTITY_ON_HAND	PRODUCT_NAME
6	1820	69	SPNIX3.3 - NL
6	2414	59	HD 9.1GB @10000 /I
6	2417	29	Client ISO CP - V
6	2395	56	32MB Cache /M
6	2396	57	EDO - 32MB
6	2371	71	C for SPNIX4.0 - Doc
6	2492	41	SPNIX3.3 AU
6	2270	64	Modem - 56/90/I
6	1742	31	CD-ROM 500/16x
6	2596	51	SS Stock - 1mm
6	2430	65	Compact 400/LQ
6	2319	44	Screws

You have now applied alternating row colors to your report.

41.2.6 Filter your XML data using groups

If you have a lot of data in your XML file, you might want to consider sorting and filtering it. You can do so by creating a group filter and a hierarchy. The steps in this section will show you how to create a filter that will only show the inventory items for a user-defined quantity amount. The filter will be based on a a parameter that the user can enter at runtime. You will then create a hierarchy in your data model to group the data in your report.

To create a user parameter and a group filter:

1. In the Object Navigator, under the User Parameters node, create a new user parameter called `P_MAXQTY`, with a Datatype of Number, Width of 20, and Initial Value of 50.

   Tip: Once you've created a new user parameter by clicking the User Parameter node, then clicking **Create**, you can press F4 to display its Property Inspector.

2. In the Data Model view, double-click the G_WAREHOUSE_ID1 group in the XML query to display its Property Inspector.
3. In the Property Inspector, under **Group**:

- set the Filter Type property to PL/SQL.
- click the PL/SQL filter property field to display the PL/SQL Editor.

4. In the PL/SQL Editor, use the template to enter the following PL/SQL code:

```
function G_WAREHOUSE_ID1GroupFilter return boolean is
begin
  if :quantity_on_hand < :P_maxqty then
     return (TRUE);
  else
   return (false);
  end if;
end;
```

> **Note:** You can also copy and paste this code from the provided file called `xmlpds_code.txt`. Copy the code under the heading "Group Filter Code."

5. Click **Compile**, and fix any errors.

> **Note:** If you are not familiar with compiling PL/SQL, refer to a PL/SQL reference manual.

6. When the code compiles successfully, click **Close**.
7. Save your report.
8. Click the Run Paper Layout button in the toolbar to run your report to paper. Notice how a Parameter Form now displays where you can adjust the quantity of items displayed in your report.

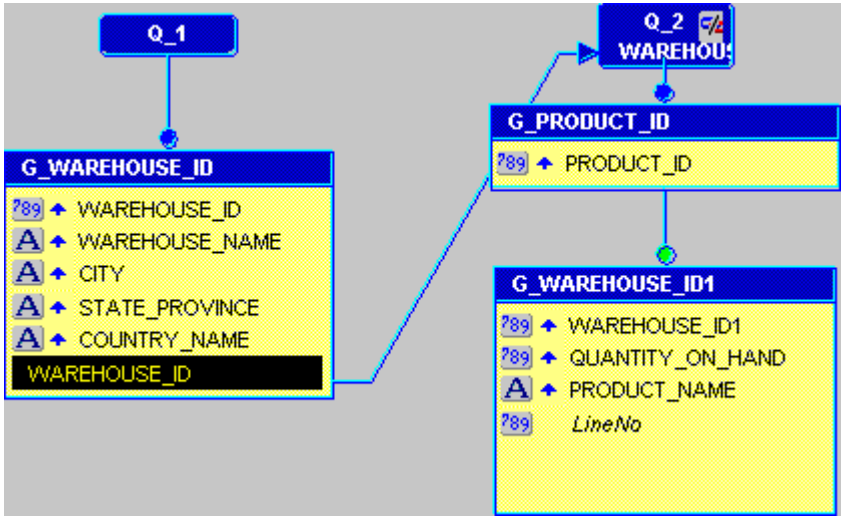
   You can also run the provided file *`Examples`*`\XML_PDS\source\inventoryreport.rdf` to view the results in Reports Builder.
9. Save your report.

To create a hierarchy for the XML query:

1. In the Data Model view, click the PRODUCT_ID column in the XML query, then drag it between the query name and the G_WAREHOUSE_ID1 group.

   Your data model should look like this:

Figure 41–7 Data Model with Group Hierarchy



Note: Noticed in the above image that a green circle now displays above G_WAREHOUSE_ID1. This circle indicates that a group filter has been created for the group.

2. Save your report. You have now created a group filter that sorts the data in your report.

41.3 Run your report to paper

To run your paper report:

1. In the Object Navigator, make sure your report (`inventoryreport_xml_<your initials>.rdf`) is selected.
2. Click the Run Paper Layout button in the toolbar to run your report to paper. Notice how the Parameter Form now displays, with the initial value of 50.
3. Your report displays in the Paper Design view, and should look something like this:

Figure 41–8 Final Paper Design view of the XML PDS example report

Warehouse ID6 Warehouse Name Sydney City Sydney
State New South Wales Country Australia

WAREHOUSE_ID	PRODUCT_ID	QUANTITY_ON_HAND	PRODUCT_NAME
6	1733	29	PS 220V /UK
6	1734	30	Cable RS232 10/AM
6	1737	30	Cable SCSI 10/FW/ADS
6	1738	30	PS 110V /US
6	1739	30	SDRAM - 128 MB
6	1740	30	TD 12GB/DAT
6	1742	31	CD-ROM 500/16x
6	1745	31	Cable SCSI 20/WD-
6	1748	32	PS 220V /EUR
6	1749	32	DIMM - 256MB
6	1750	32	DIMM - 2GB
6	1755	33	32MB Cache /NM

41.4 Summary

Congratulations! You have successfully used an XML data source for a paper report. You now know how to:

- create a SQL query from scratch.
- use the Data Wizard to create an XML query.
- create a data link between a SQL query and an XML query.
- create a layout for your report using the Report Wizard.

- apply alternating row colors to your report using format triggers and procedures.
- filter your XML data using a group filter and hierarchy.

For more information on any of the wizards, views, or properties used in this example, refer to the *Reports Builder Online Help*, which you can access in two ways:

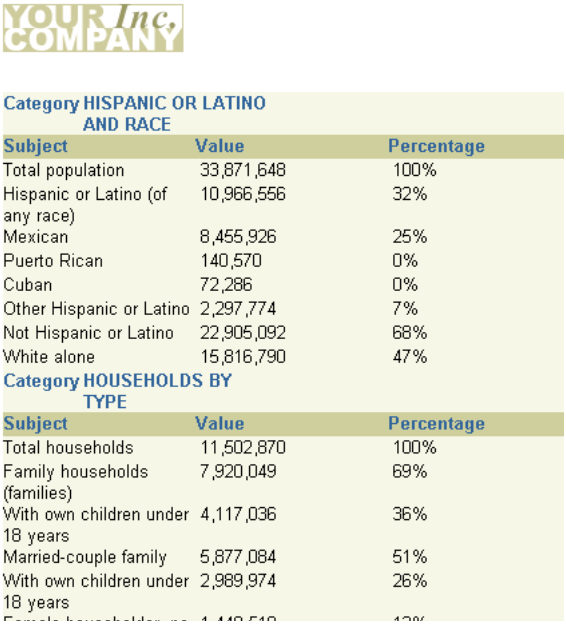
- From the Oracle Technology Network (`http://otn.oracle.com/products/reports/`), click **Documentation** and navigate to the *Reports Builder Online Help* for the most recent, hosted online help.
- From Reports Builder, choose **Help > Help Contents**.

> **Note:** You can replace your Reports Builder online help with the most recent update by downloading the latest online help set available on the Oracle Technology Network (`http://otn.oracle.com/docs/products/reports/`). Instructions for replacing your help file are included in the readme.txt in the download file.

42

Building a Report with a Text Pluggable Data Source

Figure 42–1 Report output using a text PDS



Reports Builder enables you to use any data source you wish. In this chapter, you will learn how to use character-delimited text as a data source.

About Pluggable Data Sources

The information you must publish is often derived from data in various corporate data sources. These data sources may be SQL-based (relational databases) or non-SQL-based, such as XML, OLAP, and the like. Often, you must combine data from one or more of these data sources to publish meaningful information. For example, you may need to combine data that exists in a relational database with data from a multidimensional database to compare trends and performance.

Oracle Reports enables you to leverage capabilities, such as aggregation, summarization, formatting, and scheduling, on data from any data source. You can leverage the PDS (pluggable data source) architecture to connect to your own data source, as well as to the data sources available with Oracle Reports (XML, JDBC, text, and Express).

For more information on pluggable data sources, refer to the *Reports Builder online help* and the Javadoc documentation for the PDS APIs.

Example Scenario

Suppose you downloaded the US Census Bureau data in CSV (comma-separated values) format and want to generate a report. In this example, you will create a paper report that queries this data.

Table 42–1 Features demonstrated in this example

Feature	Location
Configure Reports Builder to recognize your text file as a pluggable data source.	Section 42.2, "Set up the textpds.conf file"
Use the Report Wizard to create an paper report based on the text data source.	Section 42.3, "Use the Report Wizard to create a report"

42.1 Prerequisites for this example

To build the examples in this manual, you must have the example files we've provided. If you haven't already done so, you can download the files you'll need to complete this example from the Oracle Technology network (`http://otn.oracle.com/products/reports/`) and install them on your machine.

To download and install the example files:

1. Go to the Oracle Technology Network Web site (`http://otn.oracle.com/products/reports/`).

2. Click **Getting Started with Oracle Reports**.
3. Click **Index**, then find the "Building a Report Using a Text Pluggable Data Source" example.
4. Download the file `textpds.zip` into a temporary directory on your machine (e.g., `d:\temp`).
5. Unzip the contents of the file, maintaining the directory structure, into an examples directory on your machine (e.g., `d:\orawin90\examples`).

   This zip file contains the following files:

Table 42–2 Files necessary for building this Text PDS example

File	Description
Examples\TextPDS\result\censusreport.rdf	The finished RDF for your report.
Examples\TextPDS\result\censusreport.pdf	A PDF version of your finished report.
Examples\TextPDS\scripts\census_csv.txt.	The character-delimited data downloaded from the US Census Bureau Web site.
Examples\TextPDS\scripts\config.txt	Code for the TextPDS.conf file.

42.2 Set up the textpds.conf file

Before you can use a text file as your pluggable data source, you must set up the text PDS configuration file (`textpds.conf`) with the definition of the values in the text file. When you add this format information to the configuration file, Reports Builder can then recognize your entries as a valid format.

> **Note:** You must edit your configuration file *before* you launch Reports Builder in order for the changes to take effect.

To set up your textpds.conf:

1. In a text editor, such as UltraEdit, open the file `textpds.conf`, located in the *Oracle_Home*`/reports/conf` directory.

Note: *Oracle_Home* is where Reports Builder is installed on your computer.

2. In another text editor window, open the `config.txt` file we've provided in the *Examples*`/TextPDS/Scripts` directory.
3. Cut the text we've provided in `config.txt` and paste it into `textpds.conf` before the `</textPDS>` entry, so that the resulting `textpds.conf` file contains an entry like this:

Figure 42–2 Snapshot of the textpds.conf entry

```
<!--Data definition for Text PDS example -->
    <fileFormat  name="CensusCSV" comment="#" deli
          <columnInfo>
                <column    name="Category"
                <column    name="Subject"     type
                <column    name="value"       type
                <column    name="percentage"
            </columnInfo>
    </fileFormat>

</textPDS>
```

Note: This entry enables Reports Builder to recognize a text file as a PDS. When you choose your PDS in the Report Wizard, the text file displays as an option. Here, you also define the properties of each column in the file

4. Save the `textpds.conf` file.

You have set up your `textpds.conf` file for the character-delimited text data source.

42.3 Use the Report Wizard to create a report

When you create a report, you can either use the Report Wizard to assist you or create the report yourself. To build the simple report in this example, you can use the Report Wizard.

Before you use a text pluggable data source, you might want to examine the text file first to see what it looks like. You can open the `census_csv.conf` file in a text editor, like UltraEdit or WordPad, to see the data we'll be using in this example.

To create a simple report:

1. Now that you've updated the `textpds.conf` file, launch Reports Builder.

   Tip: If Reports Builder was already open when you modified the textpds.conf file, you should shut down Reports Builder and relaunch it.

2. In the New Report dialog box, select **Use the Report Wizard**, then click **OK**.
3. If the Welcome page displays, click **Next**.
4. On the Report Type page, select **Create Paper Layout Only**, then click **Next**.
5. On the Style page, type a title for your report and select **Group Above**, then click **Next**.
6. On the Data Source page, click **Text Query**, then click **Next**.
7. On the Data page, click **Query Definition**.

   If you get an error message and cannot display the Query Definition dialog box, check your configuration file (`textpds.conf`) to confirm the code you added.

8. Under **Data Definition**, click the **CensusCSV** format in the drop-down list.

   Note: The CensusCSV format displays in this list because you added it to the `textpds.conf` file.

9. Under **Data Source**, click **Browse** to find the `census_csv.txt` file we provided in the *`Examples`*`/TextPDS/Scripts` directory.

   If you do not see the file listed in your directory, make sure you've selected **TXT** from the drop-down list.

10. Click the file, then click **Open**.
11. When the `census_csv.txt` file displays in the **Location** field, click **OK**.

    In the Report Wizard, the data source definition displays in the **Data Source definition** field.
12. Click **Next**.
13. On the Groups page click **CATEGORY** in the **Available Fields** list and click the right arrow (**>**) to move this field to the **Group Fields** list, then click **Next**.
14. On the Fields page, click the double right arrows (**>>**) to move all available fields to the **Displayed Fields** list, then click **Next**.
15. On the Totals page, click **Next**.
16. On the Labels page, click **Next**.
17. On the Template page, select **Predefined Template** and click **Beige**, then click **Finish** to display your report output in the Paper Design view. Make whatever modifications you wish. The report should look like the following image:

Figure 42–3 Paper Design view for the text PDS report

Category HISPANIC OR LATINO AND RACE

Subject	Value	Percentage
Total population	33,871,648	100%
Hispanic or Latino (of any race)	10,966,556	32%
Mexican	8,455,926	25%
Puerto Rican	140,570	0%
Cuban	72,286	0%
Other Hispanic or Latino	2,297,774	7%
Not Hispanic or Latino	22,905,092	68%
White alone	15,816,790	47%

Category HOUSEHOLDS BY TYPE

Subject	Value	Percentage
Total households	11,502,870	100%
Family households (families)	7,920,049	69%
With own children under 18 years	4,117,036	36%
Married-couple family	5,877,084	51%
With own children under 18 years	2,989,974	26%
Female householder, no husband present	1,448,510	13%
With own children under 18 years	834,716	7%

18. Save the report.

Note: You can compare your results against the report we've provided in the examples directory. Before you can run the report, however, follow these steps:

- In the Data Model view, right-click the query and choose **Data Wizard**.
- On the Data page, click **Query Definition**.
- Update the location of **census_csv.txt** to its location on your local drive.

42.4 Summary

Congratulations! You have successfully used a text pluggable data source for a paper report. You now know how to:

- set up your `textpds.conf` file.
- use the Report Wizard to create a paper report based on the text data source.

For more information on any of the wizards, views, or properties used in this example, refer to the *Reports Builder Online Help*, which you can access in two ways:

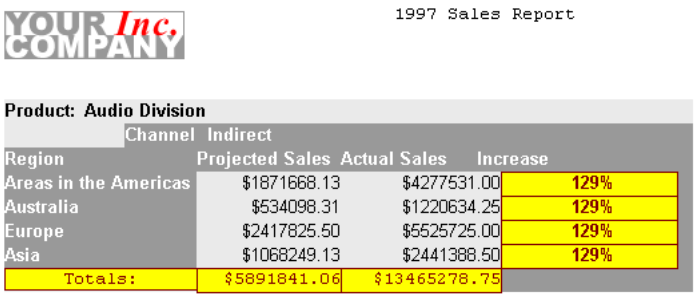
- From the Oracle Technology Network (`http://otn.oracle.com/products/reports/`), click **Documentation** and navigate to the *Reports Builder Online Help* for the most recent, hosted online help.
- From Reports Builder, choose **Help > Help Contents**.

Note: You can replace your *Reports Builder online help* with the most recent update by downloading the latest online help set available on the Oracle Technology Network (`http://otn.oracle.com/docs/products/reports/`). Instructions for replacing your help file are included in the readme.txt in the download file.

43

Building a Report Using Oracle Express Data

Figure 43–1 Express report output



The report that is described in this chapter is designed to help you learn more about the Reports Builder features for Express data. You will build an Express report that summarizes the yearly projected and actual sales for each region and sales channel in a product division.

> **Note:** The Oracle Express pluggable data source does not work on any UNIX platform in the Oracle Development Suite.

To build this report, you will use the Report Wizard to create the initial data model and report layout. You will make refinements to the data model and to the Express query. Finally, you will enhance the look of the report in the Paper Layout view and in the Paper Design view.

About Express

Express delivers on-line analytical processing (OLAP) using a multidimensional data model. This model is optimized for the analysis of trends or patterns of intersecting corporate data — such as sales, marketing, or financial variables.

Example Scenario

In this example, you will build a Sales report. Think of the data that you want to extract as being contained in the volume of a cube. Each side of the cube is a list of variable data that is contained in a category (such as Product). This category and its list of values together is called a dimension. You will select portions of each dimension and analyze them for their interaction with other dimensions. This analysis is called a measure.

An example measure for a sales analysis might select data from dimensions for time, product, geographic division, and channel. With Express, you can create a query to report on information that is as broad (for example, yearly direct and indirect sales for products sold everywhere) or as narrow (for example, monthly direct sales for all televisions sold in California) as you like.

Table 43–1 Features demonstrated in this example

Feature	Location
Use the Report Wizard to define the Express query and create a first draft of the report.	Section 43.2, "Create an Express report with the Report Wizard"
Streamline the Express query by specifying dimension values.	Section 43.3, "Refine the Express query"
Add summary and calculated totals using the Data Model view.	Section 43.4, "Add summary columns and custom measures to the data model"
Add summary and calculated totals to the report layout. Enhance the look of the report.	Section 43.5, "Enhance the report layout"

Tips for working with Express data Before you start building a report, be sure that you have reviewed the tips for working with Express data.

Note: For more information on tips for Express data, refer to "About working with Express data" in the *Reports Builder online help*.

43.1 Prerequisites for this example

To build the examples in this manual, you must have the example files we've provided, as well as access to an Oracle Express data source.

43.1.1 Example files

If you haven't already done so, you can download the files you'll need to complete this example from the Oracle Technology network (`http://otn.oracle.com/products/reports`) and install them on your machine.

To download and install the example files:

1. Go to the Oracle Technology Network Web site (`http://otn.oracle.com/product/reports/`).
2. Click **Getting Started with Oracle Reports**.
3. Click **Index**, then find the "Building a Report Using an Express Data Source" example.
4. Download the file Express.zip into a temporary directory on your machine (e.g., `d:\temp`).
5. Unzip the contents of the file, maintaining the directory structure, into an examples directory on your machine (e.g., `d:\orawin90\examples`).

   This zip file contains the following file:

Table 43–2 File(s) necessary for building the Express sample report

File	Description
Examples`\Express\result\xprs.rdf`	The final report you will have created when you finish this chapter.

43.1.2 Access to an Express Server

Before you start building this Express report, you must have already configured Reports Builder to run with Express Server.

Note: For more information on configuring for Express data, refer to "About configuring the Express data source" in the *Reports Builder online help*.

43.2 Create an Express report with the Report Wizard

You can use the Report Wizard to start building a report. The Report Wizard alone may give you an Express report that satisfies your requirements. If it does not, then you can use the Data Model view, the Paper Design view, and the Paper Layout view to further refine the report. For this report, you will start with the Report Wizard. The steps in this section will help you to create the initial report.

The report that you create in this exercise will present the monthly regional and channel projected and actual sales for each product division. The Express query will have two measures, and each measure will be dimensioned by product, time, geographic area, and channel.

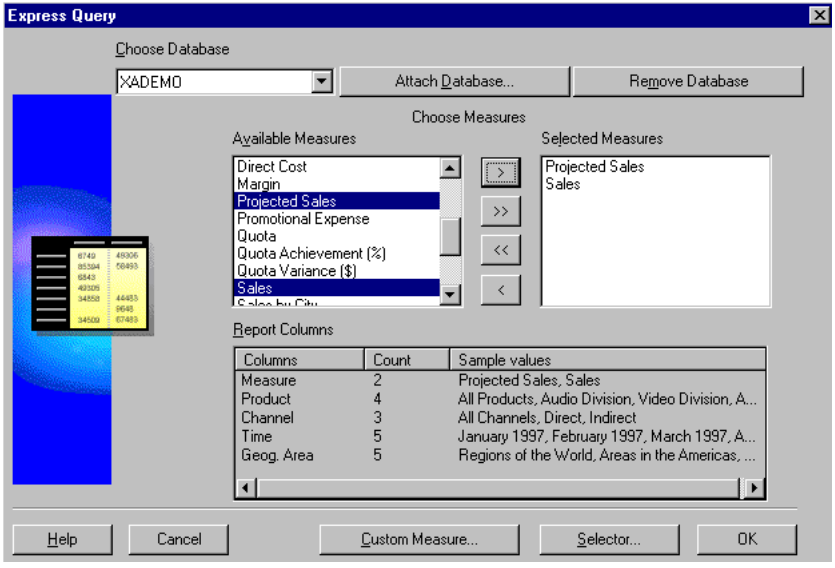
1. Launch Reports Builder (or, if already open, choose **File > New > Report**)
2. In the Welcome or New Report dialog box, select **Use the Report Wizard**, then click **OK**.
3. If the Welcome page displays, click **Next**.
4. On the Report Type page, select **Create Paper Layout only**, then click **Next**.
5. On the Style page, type `Sales Report` as the **Title**, and choose **Matrix with Group** as the report style, then click **Next**.
6. On the Data Source page, choose **Express Server Query**, then click **Next**.
7. On the Data page, click **Query Definition**.

> **Note:** If you have not already connected to an Express Server, then the Connect dialog box appears. Choose the Express Server instance that you want to access. Choose **OK**.

8. In the Express Query dialog box, choose **Attach Database** to choose the path and name of the database that you want to attach to during this session.
9. In the Attach Database dialog box, select the directory with a label such as `/oec632/`. Select `xademo.db`. This is the sample database that is provided with Express Server.
10. Click **Open** to attach the database to the session.
11. In the Express Query dialog box, Ctrl-click to select **Sales** and **Projected Sales** in the **Available Measures** list.

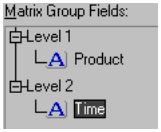
12. Click the right arrow (>) to move **Sales** and **Projected Sales** to the **Selected Measures** list. The Express Query dialog box looks similar to the following:

Figure 43–2 Express Query dialog box



13. Click **OK** to accept the Express query selections. You will return to the dialog box in a later step to refine the dimension values that are associated with the Sales and Projected Sales measures.
14. On the Data page, click **Next**.
15. On the Groups page:
    - click **Product** in the **Available Fields** list and click the right arrow (>) to move this field to the **Matrix Group Fields** list.
    - click **Level1**, then click **Time** and click the right arrow (>) so that the **Matrix Group Fields** list appears as follows:

Figure 43–3 Matrix Group Fields



- click **Next**.

16. On the Rows page, click **Geog_Area** in the **Available Fields** list and click the right arrow (>) to move this field to the **Matrix Row Fields** list, then click **Next**.
17. On the Columns page, click **Channel** in the **Available Fields** list and click the right arrow (>) to move this field to the **Matrix Column Fields** list, then click **Next**.
18. On the Cell page, click **Projected_Sales** in the **Available Fields** list and click the right arrow (>) to move this field to the **Matrix Cell Fields** list.
19. Repeat this step for **Sales**, then click **Next**.
20. On the Totals page, click **Next**.

    You will add summary totals in a later step.
21. On the Labels page, change the labels and field widths as follows, then click **Next**:

Table 43–3 Field Description of Labels page

Fields	Labels	Width
Sales	`Actual Sales`	`7`
Projected_Sales	`Projected Sales`	`7`
Geog_Area	`Region`	`10`
Product	`Product:`	`10`
Time	`Time:`	`10`
Channel	`Channel`	`7`

You change the width of labels at this point, because in a later step you will add a new layout column. This will cause columns to wrap to the next page at their current default width.

22. On the Template page, select **Predefined template** and click **Gray**, then click **Finish** to display your report output in the Paper Design view. It should look something like this:

Figure 43–4 Paper Design view for the Express report

Sales Report

Product: All Products
Time: January 1997

Region	Projected Sales		Projected Sales	
Regions of the World	9262779	8769594	4068757.5	4068757.5
Areas in the Americas	2768417.25	2620113.5	1206539.625	1206539.625
Australia	782333.875	740387	340566.09375	340566.09375
Europe	3982743.75	3773573	1779844	1779844
Asia	1729284.5	1635521.25	741807.6875	741807.6875

23. Choose **File > Save As**. Save the report in the directory of your choice, and name the report `xprs_910.rdf`.

Note: It is good practice when you are designing a report to save it frequently under a different file name. If you generate an error or if you do not like some of the changes that you made, then you easily can go back to the previously saved file and make revisions from that point.

43.3 Refine the Express query

The steps in this section will help you refine the Express query. So far you have developed a useful report that shows the monthly projected and actual sales for each region and channel in a product category. But you are really interested in the yearly projected and actual sales results for each channel and region in a product

division. You can achieve this by restricting the dimension values that you want to view.

> **Note:** For more information, refer to "Selecting data" in the *Reports Builder online help*.

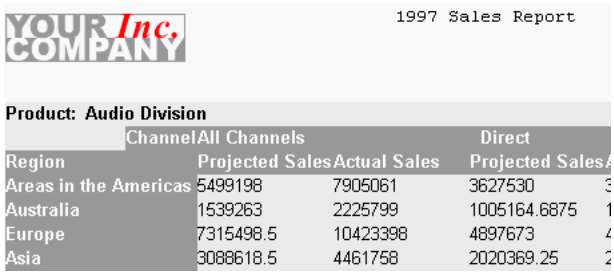
In this exercise, you will specify the following dimension values in the Express Query dialog box:

- projected and actual sales for 1997
- geographic regions, such as Asia and the Americas
- product divisions, such as the Accessory and Audio division

1. In the Paper Layout view, choose **Tools > Report Wizard** to re-enter the Report Wizard.
2. In the Report Wizard, on the **Data** page, click **Query Definition**.
3. In the Edit Query dialog box, choose **Selector**.
4. In the Selector dialog box, choose **Time Period** from the Dimensions option.
5. Click **List** to select the List tool from the toolbar.
6. In the List dialog box, choose **1997** from the **Available Time Periods** list.
7. Click **Select**. Notice that "1997" replaces the previous selections.
8. Click **OK**.
9. In the Selector dialog box, choose **Geographical Area** from the **Dimensions** option.
10. Click the **Level** button to select the Level tool from the toolbar.
11. In the Select by Level dialog box, choose **Continents/Regions** in the **At level(s)** list.
12. Click **OK**.
13. In the Selector dialog box, choose **Product** from the **Dimensions** option.
14. Click the **Level** button.
15. In the Select by Level dialog box, choose **Divisions** in the **At level(s)** list.
16. Click **OK**.

17. In the Selector dialog box, click **OK**.
18. In the Express Query dialog box, click **OK**.
19. In the Report Wizard, on the **Groups** page, click **Time** in the **Matrix Group Fields** list and click the left arrow (**<**) to move this field to the **Available Fields** list. Note that using **Time** as a break group is no longer necessary since the Express query will retrieve only aggregate data for 1997. **Product** should be the only dimension that is listed under **Matrix Group Fields**.
20. On the **Style** page, change the **Title** to `1997 Sales Report`.
21. Click **Finish** to display your report output in the Paper Design view. It should look something like this:

Figure 43–5 Paper Design view for the Express report



22. Save the report as `xprs_920.rdf`.
23. Optionally, you can compare this report with the one that you previously saved as `xprs_910.rdf`.

    Notice the projected and actual sales. In the new report, each cell represents the yearly sales for a region and channel in a product division for 1997, while the previous report displays sales data for a region and channel in a product division for each month.

43.4 Add summary columns and custom measures to the data model

The steps in this section will help you refine the data model to include summary totals for each channel in a product division. Additionally, you are curious about

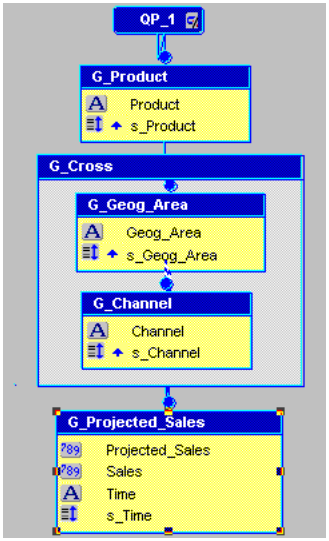
how accurately you predicted the actual sales. You can determine this by creating a *custom measure* that calculates the percent of sales above projected sales.

First, you will create the summary column using the Summary tool in the Data Model view.

Next, you will create the custom measure using the Custom Measure tool in the Express Query dialog box.

Before you begin, examine the data model:

Figure 43–6 Summary column data model



In the Data Model view you may notice additional columns, such as S_GEOG_AREA, or S_CHANNEL. These are *dimension sorting* columns. They are visible only in the data model and are the index used to sort dimensions by logical order, as opposed to alpha-numeric order. If you move a column to a new group, then you must also move the associated sort column into that group as well.

In a later step, you will sort dimension values using the Sort tool in the Edit Query dialog box.

43.4.1 Rename data objects

1. In the Object Navigator, double-click the view icon next to the **Data Model** node for your report to display the Data Model view.
2. In the Data Model view, double-click **QP_1** to display the Property Inspector, and set properties:
   - If you want to modify the Express query, choose the Express Query property under the **Query** node.
   - Under **General Information**, set the Name property to QP_SALES.
3. Repeat the above step to change the name of the **G_PROJECTED SALES** group to G_SALES_DATA.
4. Save the report as `xprs_931.rdf`.

43.4.2 Create summary columns

In this exercise, you will add two summary columns to the G_Cross group. Each summary column will calculate the projected and actual sales totals for each channel (all channels, direct, and indirect) in a product division.

1. In the Data Model view, click the Summary Column tool in the tool palette, then click in the **G_Cross** group to create a summary column.
2. Double-click the new summary column object (**CS_1**) to display the Property Inspector, and set properties:
   - Under **General Information**, set the Name property to CS_PjSalesPerChannel.
   - Under **Column**, set the Product Order property to G_CHANNEL.
   - Under **Summary**, set the Source property to PROJECTED_SALES, and set the Reset At property to G_CHANNEL.
3. Repeat the above steps to create a summary column for actual sales, with the following property settings:
   - Under **General Information**, set the Name property to CS_SalesPerChannel.
   - Under **Column**, set the Product Order property to G_CHANNEL.
   - Under **Summary**, set the Source property to SALES, and set the Reset At property to G_CHANNEL.
4. Save the report as `xprs_932.rdf`.

43.4.3 Create a custom measure

In this exercise, you will create a custom measure that will calculate the percent of actual sales above projected sales for each region and in each product division. To do this, you will use the Custom Measure tool within the Express Query dialog box to build the new measure called Increase.

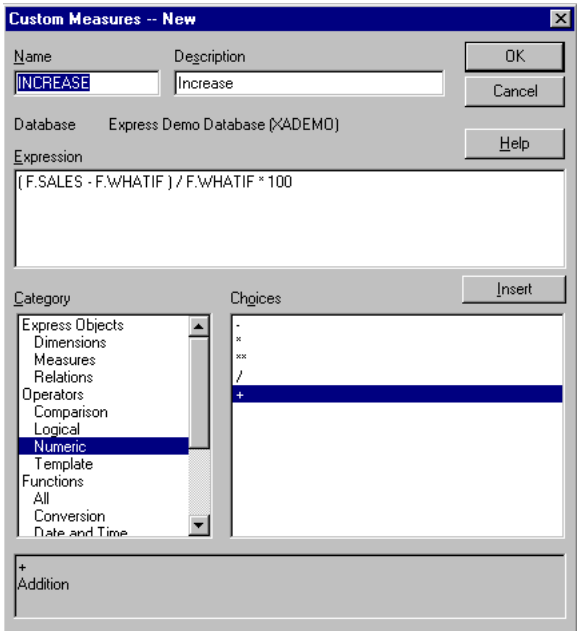
1. In the Data Model view, double-click the **QP_SALES** query object to display the Express Query dialog box.
2. Click **Custom Measure** at the bottom of the Express Query dialog box.
3. Click **New** to display the Custom Measure -- New dialog box.
4. In the **Name** field, type `INCREASE`.
5. In the **Description** field, type `Increase`.
6. In the **Category** list, click **Template** under **Operators**. Notice a list of templates appears under **Choices**.
7. In the **Choices** list, click the left parenthesis and click **Insert**. A left parenthesis appears in the **Expression** field.
8. In the **Category** list, click **Measures** under **Express Objects**.
9. In the **Choices** list, click **F.SALES**, and click **Insert**.
10. Use the following table to build the expression:

Table 43–4 Build Expressions

Category	Sub-category	Choose	or Type:
Operators	Numeric	Minus Sign	-
Express Objects	Measures	F.WHATIF	F.WHATIF
Operators	Template	Right parenthesis	)
Operators	Numeric	Forward slash	/
Express Objects	Measures	F.WHATIF	F.WHATIF
Operators	Numeric	asterisk	*

11. Following the asterisk, type `100` in the **Expression** field.
12. When you are finished, the expression should look similar to the one in the following figure:

Figure 43–7 Custom Measures Expression



13. Click **OK**. Note that "Increase" is listed in the **Custom Measures** text box in the Custom Measures dialog box.

14. Click **Close**.

15. In the Express Query dialog box, scroll through the **Available Measures** list. "Increase" now appears alphabetically. Click **Increase** and click the right arrow (>) to move **Increase** to the **Selected Measures** list, below **Projected Sales** and **Sales**.

16. Click **OK** to return to the Data Model view.

    The group G_SALES_DATA now includes the custom measure that you just created, INCREASE.

17. Click the Run Paper Layout button in the toolbar to view the report in the Paper Design view. Note that neither the summary columns nor the custom measure are available in the report. This occurred because you have not yet added them as fields to the report layout. You will do this in the next few exercises.

18. Save the report as `xprs_933.rdf`.

43.5 Enhance the report layout

The steps in this section show you how to re-arrange the report layout, add the summary and custom measure columns that you created in Section 43.3, "Refine the Express query", and format objects to further enhance the look of the report. You make these changes using the Paper Layout view and the Paper Design view.

43.5.1 Insert summary fields in the report

1. In the Object Navigator, double-click the view icon next to the **Paper Layout** node to display the Paper Layout view.
2. Position the windows to display the Object Navigator and the Paper Layout view side-by-side.
3. In the Object Navigator, fully expand the **Paper Layout** node to view the nested nodes, such as the **M_G_PRODUCT_GRPFR** and **R_G_PRODUCT** nodes.
4. In the Object Navigator, type `M_G_CROSS_GRPFR` in the **Find** field to locate this object. In the Paper Layout view, the master cross-matrix frame is selected.
5. In the Paper Layout view, extend the selected frame down about 1/4 inch.
6. In the Object Navigator, click **F_CHANNEL**.
7. In the Paper Layout view, click the Select Parent Frame button in the toolbar to select the parent frame, **R_G_CHANNEL**.

> **Note:** You may need to resize the Paper Layout window to see the Select Parent Frame button, as it is located on the far right of the toolbar.

8. Extend the frame down about 1/4 inch.
9. Click the Field tool in the tool palette.
10. Click and drag a rectangle in the area directly under the **F_PROJECTED_SALES** field to insert a field object
11. Double-click the new field object to display the Property Inspector, and set properties:

- Under **General Information**, set the Name property to F_PjSalesPerChannel.
- Under **Field**, set the Source property to the CS_PjSalesPerChannel.

12. Change the format of the field as follows:
    - Click the **Fill Color** tool in the tool palette and change fill color to light yellow.
    - Click the **Text Color** tool and change the text color to dark brown.
    - Click the **Line Color** tool and surround the field with dark brown border lines.

    Note: You can turn Snap to Grid on or off as desired to help you arrange objects in the layout. Chose **View > Snap to Grid**. A check mark indicates that the option is on.

13. Repeat the steps above to create another field object, placed directly under F_SALES.
14. Double-click the new field object to display the Property Inspector, and set properties:
    - Under **General Information**, set the Name property to F_SalesPerChannel.
    - Under **Field**, set the Source property to the CS_SalesPerChannel.

    Note: The fill and text colors, as well as the border lines, match the first field that you created, F_PjSalesPerChannel.

15. Click the Text tool in the tool palette, then click and drag a rectangle to fill the space directly under **F_GEO_AREA** to create a new boilerplate text object.
16. In the new boilerplate text object, type `Totals:`.
17. Click the Align Center button in the toolbar and make format changes to match the summary fields that you created above.

18. Click the Paper Design button in the toolbar to view the changes in the Paper Design view.
19. Save the report as `xprs_941.rdf`.

43.5.2 Insert the custom measure field into the report

In this section, you will add a column to display the custom measure that you created in Section 43.4.3, "Create a custom measure" by inserting a field object in the report layout.

To do this, you will add a new column to the layout of the report and insert the field object into the column.

> **Tip:** The new field object also must have the same frequency as F_PROJECTED_SALES and F_SALES. If the field object is not at the same frequency, then the report will fail to run.

1. In the Paper Design view, click the Paper Layout button in the toolbar to display the Paper Layout view. Position the Paper Layout view and Object Navigator side-by-side.
2. In the Object Navigator, Ctrl-click **M_G_PRODUCT_GRPFR** and **R_G_PRODUCT**.

   > **Tip:** M_G_PRODUCT_GRPFR is the underlying master group. It is hidden directly under R_G_PRODUCT. In the Paper Layout view, it may look like only one group is selected when, in fact, both frames are selected.

3. In the Paper Layout view, expand the width of the selected frames to about 4 3/4 inches.

   > **Note:** Click the Flex On button in the toolbar to turn Flex mode on, or click the Flex Off button if you are unable to resize or move an object.

4. In the Object Navigator, click **M_G_CROSS_GRPFR**.

5. In the Paper Layout view, expand the width of the selected frame to about 4 3/4 inches.
6. Click the **F_GEOG_AREA** frame object, then click the Select Parent Frame button in the toolbar to select the parent frame, **R_G_GEOG_AREA**. Expand the width of the selected frame to about 4 3/4 inches.
7. Click **F_CHANNEL** and click the Select Parent Frame button in the toolbar to select the parent frame, **R_G_CHANNEL**.
8. Expand the width of the selected frame to about 4 3/4 inches.
9. Click **F_CHANNEL** again and expand the width of the object to about 4 3/4 inches.
10. Click the Field tool in the tool palette, then click and drag a rectangle to the right of the **F_SALES** object to create a new field object.
11. Double-click the new field object to display the Property Inspector, and set properties:
    - Under **General Information**, set the Name property to F_Increase.
    - Under **Field**, set the Source property to INCREASE.
12. Click the Run Paper Layout button in the toolbar to run the report. You should see an error message that indicates that F_Increase references INCREASE at a frequency below its group. You are unable to run the report.

    To understand why this error occurred, look for **F_INCREASE** in the Object Navigator. It is probably placed at a higher level (and lower frequency) than **R_G_PROJECTED_SALES**. Recall that the column INCREASE calculates the percent of actual sales above projected sales. In order to run this report, F_INCREASE must have the same frequency as F_PROJECTED_SALES and F_SALES to reference the data that it needs to calculate the value.
13. Click **OK** to close the error message.
14. Click the Paper Layout button in the toolbar to display the Paper Layout view.
15. In the Paper Layout view, click the **F_INCREASE** field and delete it.
16. Click **F_SALES**, then click the Select Parent Frame button in the toolbar to select the parent frame, **R_G_PROJECTED_SALES**.
17. Expand the width of the selected frame to about 4 3/4 inches.
18. Repeat steps 10 through 13 to create the field object.

19. With the F_Increase object selected, locate **F_INCREASE** in the Object Navigator to ensure that it has the same frequency as **F_PROJECTED_SALES** and **F_SALES**.
20. Change the format of the **F_INCREASE** field as follows:
    - Click the **Fill Color** tool in the tool palette and change fill color to light yellow.
    - Click the **Text Color** tool and change the text color to dark brown.
    - Click the **Line Color** tool and surround the field with dark brown border lines.
    - Click the Bold button in the toolbar to make the text darker and more noticeable.
21. Click the Text tool in the tool palette, then click and drag a rectangle above **F_INCREASE** to add a new boilerplate text object for the column title.
22. In the new boilerplate text object, type `Increase`.
23. Arrange the text object in the column and change the format to match the field to its left, **Actual Sales**.

> **Note:** You may want to turn off Snap to Grid (**View > Snap to Grid**). in order to extend the text object to cover the entire field. Ensure that the text object is selected when you apply formatting, or it will not take effect.

24. Click the Run Paper Layout button in the toolbar to view your report in the Paper Design view.
25. Save the report as `xprs_942.rdf`.

43.5.3 Sort dimension values

Suppose you want to change the sorting order of the distribution channels in the report. In this exercise, you will change the sorting criteria for the Channel dimension by using the Selector in the Express Query dialog box. Instead of listing the order by the default channel hierarchy (top to bottom), you will display data from the lowest to the highest channel in the hierarchy. Note that the hierarchy is predefined in the database to place "All Channels" first, with "Indirect" placed last.

1. In the Data Model view, double-click the query object, **QP_SALES** to display the Express Query dialog box.
2. In the Express Query dialog box, click **Selector**.
3. In the **Dimensions** list, click **Distribution Channel** and click the **Sort** button.
4. In the Sort Selection dialog box, select the following values:

Table 43–5 Values of Sort Selection dialog box

Criteria	Selection
based on	hierarchy
in order	bottom to top
in hierarchy	Standard

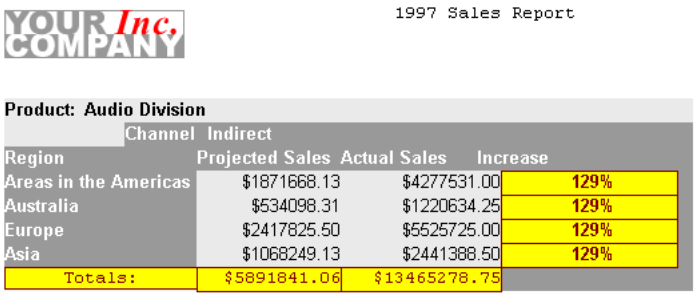
5. Click **OK** in the Sort Selection dialog box.
6. Click **OK** in the Selector dialog box.
7. Click **OK** in the Express Query dialog box.
8. Click the Run Paper Layout button in the toolbar to view the report in the Paper Design view.
9. Save the report as `xprs_943.rdf`.

43.5.4 Make format changes in the Paper Design view

1. In the Paper Design view, Shift-click the columns under **Projected Sales** and **Actual Sales**, and the **Projected Sales total** and the **Sales total** fields.
2. Click the Currency button in the toolbar to change the format mask to currency.
3. Click the Align Right button to right justify the values.
4. Click the Add Decimal Place button twice to insert two decimal places.
5. Under **Increase**, click the column.
6. Click the Percent button to change the format mask to percentage.
7. Click the Align Center button to center the values.

   The report should now look similar to the following:

Figure 43–8 Paper design output



8. Save the report as `xprs_944.rdf`.

43.6 Summary

Congratulations! You have finished the Express sample report. You now know how to:

- use the Report Wizard to define a data model and layout.
- make changes to the Express query by restricting the dimension values.
- use the Data Model view to add summary and custom measures columns to the report.
- use the Paper Layout view to insert fields and re-arrange the layout.
- use the Paper Design view to enhance the look of the report.

For more information on any of the wizards, views, or properties used in this example, refer to the *Reports Builder Online Help*, which you can access in two ways:

- From the Oracle Technology Network (`http://otn.oracle.com/products/reports/`), click **Documentation** and navigate to the *Reports Builder Online Help* for the most recent, hosted online help.
- From Reports Builder, choose **Help > Help Contents**.

Note: You can replace your Reports Builder online help with the most recent update by downloading the latest online help set available on the Oracle Technology Network (`http://otn.oracle.com/docs/products/reports/`). Instructions for replacing your help file are included in the readme.txt in the download file.

A

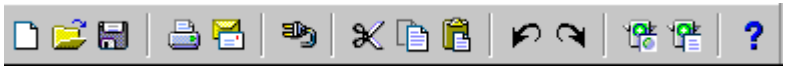
Tool Palette and Toolbar Reference

This appendix provides descriptions of the buttons and tools in the Reports Builder tool palettes and toolbars.

A.1 Main Toolbar

The main toolbar is located at the top of the Reports Builder window, directly beneath the menu bar:

Figure A–1 Main Toolbar



New button. Displays the New Report dialog box.

Open button. Displays the Open dialog box.

Save button. Saves the report. If you haven't saved the report before, the Save As dialog box displays.

Print button. Prints the paper report.

Mail button. Displays the Mail dialog box.

Connect button. Displays the Connect dialog box.

Cut button. Deletes the currently selected item and temporarily places it in the clipboard. Use Paste to paste the selected item.

Copy button. Temporarily places a copy of the selected item in the clipboard. Click the **Paste** button to paste the selected item.

Paste button. Pastes the item in the clipboard in current location of the cursor.

Undo button. Undoes the last action performed.

Redo button. Performs the last action again.

Run Web Layout button. Runs the current report to your Web browser.

Run Paper Layout button. Runs the current report to the Paper Design view in Reports Builder.

Help button. Displays the Reports Builder online help.

A.2 Data Model view tool palette

The Data Model view tool palette is a vertical group of tools located on the left-hand side of the Data Model view.

Select tool. Deselects any selected tool to "turn off" the current tool.

Magnify tool. Zooms in the view on the clicked object. Use **SHIFT + Magnify** to zoom out.

Summary Column tool. Creates a summary column in the query.

Data Link tool. Creates a link between the columns in the queries.

Formula Column tool. Creates a formula column in the query.

Cross Product tool. Creates a matrix (cross-product) group.

Placeholder tool. Creates a placeholder column which you can modify later.

SQL Query tool. Displays the SQL Query Statement dialog box where you can enter a SQL query SELECT statement or use Query Builder to create a query.

Ref Cursor tool. Displays the PL/SQL Editor where you can type a ref cursor query.

XML Query tool. Displays the Define XML Query dialog box, where you can specify the XML data definition and data source.

JDBC Query tool. Displays the JDBC Query dialog box, where you can define the SQL or stored procedure to define the data for the query.

Text Query tool. Displays the Text Query dialog box, where you can specify a text data definition and data source.

Express Server Query tool. Displays the Express Server Query dialog box, where you can specify an Oracle Express data definition and data source.

A.3 Paper Layout view tool palette

The Paper Layout view tool palette is a vertical group of tools located on the left-hand side of the Paper Layout view.

Select tool. Deselects any selected tool to "turn off" the current tool.

Magnify tool. Zooms in the view on the clicked object. Use **SHIFT + Magnify** to zoom out.

Frame Select tool. Selects all objects within the selected frame or repeating frame, depending upon their explicit anchors (first click the tool, then the frame).

Reshape tool. Enables you to reshape the selected boilerplate object.

Text tool. Creates a boilerplate text object.

Rotate tool. Enables you to rotate the direction of the selected boilerplate object.

Line tool. Draws a line boilerplate object.

Rectangle tool. Draws a rectangle boilerplate object.

Arc tool. Draws an arc boilerplate image.

Rounded Rectangle tool. Draws a rounded rectangle boilerplate object.

Polyline tool. Draws an open multilined boilerplate object. Use your mouse to create the multiple lines.

Polygon tool. Draws a multisided boilerplate object. The object must be closed, unlike a polyline object.

Freehand tool. Draws a line where you drag your mouse.

Ellipse tool. Draws an ellipse boilerplate object.

Frame tool. Draws a frame.

Repeating Frame tool. Draws a repeating frame.

Graph tool. Displays the Graph Wizard so that you can to define a graph that will be inserted into your layout.

Field tool. Creates a field object.

Anchor tool. Creates an anchor between two objects in your layout.

File Link tool. Creates a link file object that you can use to link an external file to your report.

Report Block tool. Displays the Report Block wizard so that you can add a new report block to your layout.

Glossary

column

1. A vertical space in a database table that represents a particular domain of data. A column has a column name (e.g., ENAME) and a specific datatype (e.g., CHAR). For example, in a table of employee information, all of the employees' names would constitute one column. A record group column represents a database column.
2. A data model object created automatically for each column expression in a query's SELECT list, or created manually to perform summaries, formulas, or act as a placeholder.
3. The representation of an attribute of an entity.

data model

A relational model that defines what data should be fetched from the data source(s), what values should be computed, and how data should be ordered in a report. Reports Builder objects that define the data model are queries, groups, columns, parameters, and links.

Data Model view

One of the views of the Report Editor that displays a structural representation of the data in a report. The objects do not appear in the report output, but the structure determines the layout style, and the data objects provide the values that appear in the layout objects.

database

1. A set of dictionary tables and user tables that are treated as a unit.

2. (Oracle Express) A single file (possibly accompanied by extension files) that contains objects that organize, store, and manipulate data. In Express, examples of such objects are variables, dimensions, formulas, models, and programs.

data source

A source for data returned by a query, including database objects such as tables, views, synonyms, snapshots, and queries stored as views. OracleAS Reports Services allows you to access any data source.

The new pluggable data source (PDS) architecture replaces Oracle Open Client Adapter (OCA), and the Open Database Connectivity (ODBC) drivers are no longer supported in Oracle Reports 10*g*. However, Java Database Connectivity (JDBC) is one of the pluggable data sources available that can utilize the JDBC-ODBC bridge, allowing access to other data sources.

detail query

When defining a master/detail report, the detail query retrieves all related records for each record retrieved by the master, or parent, query.

dialog box

A partial screen or window that prompts you to enter information necessary to complete an operation.

disabled

An interface element state that means a menu item, button, and so on, cannot be used in the current context (i.e., it does not respond to keyboard or mouse input).

editor

See view.

enabled

An interface element state that means that a menu item, button, and so on, can be used in the current context (that is, it responds to keyboard or cursor/mouse input).

field

1. An interface element in which you enter, edit, or delete data.
2. A layout object that defines how the data for a specific query column appears.

foreign key

A value or column in one table that refers to a primary key in another table.

format mask

A setting that defines the appearance of the value of a field. For example, a format mask is used to specify the display of currency amounts and dates.

format trigger

A PL/SQL function that allows you to dynamically change the formatting attributes of an object.

formula column

A user-created column that gets its data from a PL/SQL function or expression, a SQL statement, or a combination of these.

frame

A layout object used to enclose other layout objects and control the formatting, frequency, and positioning of several objects simultaneously.

group

1. In Reports Builder, a data model object that is created automatically to contain all the columns selected by a query, or created by the user to modify the hierarchy of the data appearing in a report; it is used primarily for creating breaks in a report, as well as for resetting computations.
2. An object that is composed of several other objects.

HTML (Hypertext Markup Language)

Acronym for Hypertext Markup Language. A tag-based ASCII language used to specify the content and links to other documents on Web servers on the Internet. End users with Web browsers view HTML documents and follow links to display other documents.

hyperlink

A reference (link) from some point in one document to (some point in) another document or another place in the same document. A Web browser usually displays a hyperlink in some distinguishing way (in a different color, font or style). When users activate hyperlinks (by clicking on them with a mouse) the browser displays the target of the link.

icon

A graphic representation of a window or tool.

image

A bitmapped object that can be stored and loaded into an application. The client cannot modify an imported image.

intranet

An internal TCP/IP network, access to which is restricted (via a firewall) to individuals inside the company or organization. An intranet provides similar services within an organization to those provided by the Internet, but is not necessarily connected to the Internet. A common example of an intranet is when a company sets up one or more Web servers on an internal network for distribution of information or applications within the company.

Java

A computer language that supports programming for the Internet in the form of platform-independent "applets".

JSP (JavaServer Page)

JavaServer Page (JSP) technology is an extension to the Java Servlet technology from Sun Microsystems that provides a simple programming vehicle for displaying dynamic content on a Web page. JSP is a server-side technology. A JSP is an HTML page with embedded Java source code that is executed in the Web server or application server. The HTML provides the page layout that is returned to the Web browser, and the Java provides the business logic.

layout

See Paper Layout view.

margin

An optional report region that appears at the top and bottom of each logical page in a report section (Header, Main, or Trailer). The margin may include any layout object, but typically contains boilerplate and fields (for page numbers, page totals, grand totals, and current date and time).

object

1. An item that can be placed on the layout. The following are examples of objects: rectangle, line, ellipse, arc, polygon, polyline, rounded rectangle, freehand, chart, text, symbol, and text field.

2. In an Oracle database, an instance of an object type. An object can be a row in an object table, or the portion of a row contained in a column object in a relational table.

Object Navigator

A hierarchical browsing and editing interface that enables you to locate and manipulate application objects quickly and easily. Features include:

- A hierarchy represented by indentation and expandable nodes (top-level nodes show module types, database objects, and built-in packages), enabling tasks such as creating, editing, renaming, and deleting objects.
- A find field and icons, enabling forward and backward searches for any level of node or for an individual item in a node
- Icons in the horizontal toolbar replicating common File menu functions

Oracle Application Server (OracleAS)

A strategic platform for network application deployment. By moving application logic to application servers and deploying network clients, organizations can realize substantial savings through reduced complexity, better manageability, and simplified development and deployment. OracleAS provides the only business-critical platform that offers easy database Web publishing and complete legacy integration while transitioning from traditional client-server to network application architectures.

Oracle Developer Suite

Combines leading Oracle application development and business intelligence tools into a single, integrated product. Built on Internet standards such as Java and XML, the suite provides a complete and highly productive development environment for building applications for Oracle Application Server and the Oracle database.

ORACLE_HOME

An alternate name for the top directory in the Oracle directory hierarchy on some directory-based operating systems. An environment variable that indicates the root directory of Oracle products.

You can refer to the directory specified by *ORACLE_HOME* in syntax:

On UNIX: $ORACLE_HOME

On Windows: %ORACLE_HOME%

OracleAS Portal

An HTML-based development tool for building scalable, secure, extensible HTML applications and Web sites. OracleAS Reports Services uses OracleAS Portal to control end user access to reports published on the Web by storing information about report requests, the secured server, and any OracleAS Reports Services printer used to print report output.

OracleAS Reports Services

See Reports Services.

Paper Design view

One of the views of the Report Editor that displays output for paper reports and allows you to make many commonly required, simple modifications to the layout, such as spacing, formatting fields, color, and editing text, without having to open the Paper Layout view.

Paper Layout view

One of the views of the Report Editor that displays the layout objects in a paper report and allows you to make many modifications to any layout object. All layout objects have properties that you can modify using the Property Inspector. The hierarchy of the layout objects is determined by the Data Model.

Paper Parameter Form view

Displays the layout of the Parameter Form that, at runtime, allows user input of parameter values in the Runtime Parameter Form.

PDF (Portable Document Format)

Acronym for Portable Document Format. A file format (native for Adobe Acrobat) for representing documents in a manner that is independent of the original application software, hardware, and operating system used to create the documents. A PDF file can describe documents containing any combination of text, graphics, and images in a device-independent and resolution independent format.

PL/SQL

Oracle's proprietary extension to the SQL language. Adds procedural and other constructs to SQL that make it suitable for writing applications.

Property Inspector

A window that enables you to view, locate, and set the properties of the currently selected object(s) in the Object Navigator, Report Editor, and Template Editor.

Every Reports Builder object (query, group, frame, parameter, etc.) has associated properties that can be viewed using the Property Inspector. The Property Inspector features:

- expandable and collapsible nodes
- in-place property editing
- search features
- multi-selection
- complex property dialogs
- the ability to invoke multiple instances of the Property Inspector

To get help on any property, click the property in the Property Inspector and press F1.

query

A SQL SELECT statement that specifies the data you wish to retrieve from one or more tables or views of a database.

RDF file

A file that contains a single report definition in binary format. .RDF files are used to both run and edit reports.

record

One row fetched by a SQL SELECT statement.

REP file

A file that contains a single report definition in binary format. .REP files are used solely to run reports; you cannot edit a .REP file.

repeating frame

A layout object used to display rows of data that are fetched for a group.

Report Editor

The Reports Builder window that provides different views to help you handle the data objects and layout objects for Web and paper reports. The views are:

- Data Model view
- Paper Layout view

- Paper Design view
- Paper Parameter Form view
- Web source view

Reports Builder (rwbuilder)

An Oracle Reports executable that starts Reports Builder to enable report developers to create and maintain report definitions.

Reports Runtime (rwrun)

An Oracle Reports executable that runs a report using the OracleAS Reports Services in-process server.

Reports Services

The runtime environment for Reports Developer applications. OracleAS Reports Services executes, distributes, and publishes your reports for enterprise wide reporting. Using OracleAS Reports Services to deploy your reports results in gains of flexibility, time savings, and processing capacity.

Reports Servlet (rwservlet)

An Oracle Reports executable that translates and delivers information between HTTP and the Reports Server, enabling you to run a report dynamically from your Web browser.

row

One set of field values in a table; for example, the fields representing one employee in the example table EMP.

Runtime Parameter Form

A screen or window appearing optionally at runtime in which a user can modify print options and parameters prior to report execution.

schema

A collection of related database objects, usually grouped by database user ID. Schema objects include tables, views, sequences, stored program units, synonyms, indexes, clusters, and database links.

SELECT statement

A SQL statement that specifies which rows and columns to fetch from one or more tables or views.

SQL

A standard interface for storing and retrieving information in a relational database. SQL is an acronym for Structured Query Language.

SQL file

A file that contains a query stored in text (e.g., ASCII or EBCDIC) format.

SQL script

A file containing SQL statements that you can run to perform database administration quickly and easily. Several SQL scripts are shipped with Oracle products.

SQL statement

A SQL instruction to Oracle. A SELECT statement is one type of SQL statement.

style sheet

HTML extensions that provide powerful formatting flexibility in HTML documents. To view an HTML document that takes advantage of style sheets, display it in a browser that supports style sheets.

table

A named collection of related information, stored in a relational database or server, in a two-dimensional grid that is made up of rows and columns.

tabular

A default layout displaying labels at the top of the page and rows of data underneath the labels.

template

A skeleton definition containing common style and standards, and may include graphics. A template provides a standard format to enable quick and easy development of professional standard look-and-feel reports.

Template Editor

A work area in which you can define objects and formatting properties for your templates. It is similar to the Paper Layout view of the Report Editor. You can create, delete, and modify objects (e.g., page numbers, text, and graphics) in the margin area. You cannot create and delete objects in the body area, but you can modify the properties of body objects in the Property Inspector.

tool

An iconic button used to create and manipulate objects in an application.

tool palette

A collection of tools represented by iconic buttons in the user interface that allow a report developer to perform tasks, such as drawing a rectangle in the Paper Layout view or creating a query in the Data Model view.

toolbar

A collection of iconic buttons that perform product commands. Usually aligned horizontally along the top, or vertically down the side of a window.

URL (Uniform Resource Locator)

A compact string representation of the location for a resource that is available through the Internet. It is also the text string format clients use to encode requests to OracleAS.

view

1. In Reports Builder, a work area in which you perform a specific set of tasks, such as defining a report data model, layout, or Parameter Form.
2. A virtual table whose rows do not actually exist in the database, but which is based on a table that is physically stored in the database.

Web browser

A program that end users utilize to read HTML documents and programs stored on a computer (serviced by a Web server).

Web server

A server process (HTTP daemon) running at a Web site which sends out Web pages in response to HTTP requests from remote Web browsers.

Web source view

One of the views of the Report Editor that displays the HTML / JSP source for a report. You can use this view to add dynamic content to a Web page using the Report Block Wizard and the Graph Wizard. Experienced Java developers can edit the Web source directly in this view.

wizard

A step-by-step interface for commonly performed tasks. The wizards in Reports Builder are:

- Report Wizard: guides you through the steps to create a basic paper or Web report. Each page of the wizard asks you for information to help you create your initial report.
- Data Wizard: helps you helps you quickly define or modify a query for a multiquery data models.
- Graph Wizard: Adds variety of charts and graphs, including true 3-dimensional graphs. Implemented in Reports Builder with the Oracle BI graph bean.
- Report Block Wizard: enables you to add data to a static HTML page.

XML

Acronym for Extensible Markup Language. A metalanguage using SGML to define and structure data. Reports Builder supports XML output to enable Web publishing as well as electronic data exchange with third-party applications. You can also use XML to build report definitions that can be merged with other report definitions at runtime or run separately.

Index

A

B

C

D

E

F

G

H

I

J

K

L

M

N

O

P

Q

R

S

T

U

V

W

X