Crystal Reports XI
Designer 3 Workshop

Vision Harvest, Inc.

1-888-236-8087
sales@visionharvest.com

Revised: April 1, 2006
Part # VISI100062S

This course covers Crystal Reports® XI

This course and all materials supplied to the student are designed to familiarize the student with the operation of the software programs. We urge each student to review the manuals provided by the software publisher regarding specific questions as to the operation of the programs. THERE ARE NO WARRANTIES, EXPRESSED OR IMPLIED, INCLUDING WARRANTIES OF MERCHANTABILITY OR FITNESS FOR A PARTICULAR PURPOSE, MADE WITH RESPECT TO THESE MATERIALS OR ANY OTHER INFORMATION PROVIDED TO THE STUDENT. ANY SIMILARITIES BETWEEN FICTITIOUS COMPANIES, THEIR DOMAIN NAMES, OR PERSONS WITH REAL COMPANIES OR PERSONS IS PURELY COINCIDENTAL AND IS NOT INTENDED TO PROMOTE, ENDORSE OR REFER TO SUCH EXISTING COMPANIES OR PERSONS.

VISION HARVEST, INC. NOR ITS RESPECTIVE PRODUCTS OR SERVICES ARE AFFILIATED WITH, OR ENDORSED, LICENSED, OR SPONSORED BY BUSINESS OBJECTS, THE OWNER OF CRYSTAL REPORTS.

This courseware has been developed by the professional team at Vision Harvest, Inc., www.visionharvest.com. Questions, comments or concerns about courseware content should be sent to sales@visionharvest.com or 888.236.8087.

Table of Contents

Introduction

Introduction Objectives

This manual is written to give you a step-by-step guide for your classroom training and a handy reference for your daily work. In this Introduction, you will learn how to use this training guide effectively. This section covers the following topics:

- An introduction to the Crystal Reports application
- Pre-Qualification Exercise
- Class objectives
- Help with using this training guide
- Information on how to start the program

About Crystal Reports

In today's information intensive environment, every business has a database of some sort. After all, business today is all about information and databases give you a handle on the massive amounts of information you must deal with. Therefore, your business has a database and from that database, you need reports. The problem is, most reporting capabilities that come with database programs are limited. They only report on data from that program. Many users need to report on data from multiple sources, even databases that are in SQL (Structured Query Language), such as Oracle, Microsoft SQL Server, Informix or Sybase.

Crystal Reports is one of the most powerful reporting programs available with the ability to pull data from all types of data sources. You can use Crystal Reports to generate reports from any of the standard PC database programs, Access, Paradox, or FoxPro, as well as from a mainframe or server database. Crystal also has a powerful web-reporting server that allows you to distribute your reports over the web.

Crystal Reports is bundled with more than 160 other programs including Visual Basic, some medical applications, many accounting packages and several ERP solutions. It makes report generation easy without requiring you to be a programmer or a database expert. If you know how to work in a Windows environment and are familiar with the data you want to use, you can create a Crystal Report that looks professional and makes sense.

Training Philosophy

Studies show that people retain 10% of information they see, 20% of information they hear, 50% of what they see and hear, and 80% of what they see, hear and do. In line with this, this class utilizes a hands-on method of training. You will see the effects of new procedures on the screen, hear the instructor explain how and why to use features, and perform the actions yourself as you learn.

In addition, this class focuses on your ability to perform tasks using the most productive techniques. The manual may contain several methods of accomplishing a certain task. However, class time does not allow for practice of all methods for each task. Your instructor will guide you in the most effective method of performing a task, but inform you of other methods that are available.

Questions are encouraged. While we give our best effort to explain new concepts in understandable terms, you may need to hear the concept again or have it explained more thoroughly. Please let the instructor know when you need more information!

Class Objectives

This class is a performance based instructional system. It is geared to provide you with the tools you need to build and distribute reports the quickest, most efficient way. After completing this course, you will be able to perform the following tasks:

- Refresher Exercise
- Use multiple sections in reports for advanced formatting techniques
- Underlay graphics and charts to print beside data
- Calculate running totals in reports
- Develop parameter fields to prompt users for information
- Limit user entry into parameter fields
- Solve table link problems with advanced linking techniques
- Understand Crystal Reports data handling and Evaluation Time functions
- Comfortably make use of variables in formulas
- Work with arrays and control structures in formulas for more flexible decision making
- Use additional reports within a main report as subreports
- Create linked subreports to tie data from another report to the current report
- Format subreports to display on demand only
- Customize groups in the report with formulas and custom names and create hierarchical groups
- Perform group selection based on summary field values
- Create and format Cross-Tab reports and create charts from Cross-Tabs
- Working with Report Alerts
- Creating and using Report Templates

About This Manual

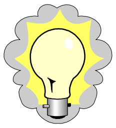
Each section of this manual contains objectives to provide you with the overall goals for the lesson. Lessons have descriptions of features and concepts followed by systematic directions for completing a specific task. Each section ends with a challenge exercise to help you practice the skills you learned in the lesson. Challenge exercises provide you with tasks to accomplish. Try to complete these exercises on your own.

As you work in this Training Guide, certain conventions are used to identify specific procedures. Use the following table as a guide:

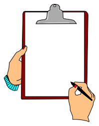
Training Guide Conventions	
Item	Illustrated As
Menu Commands	Underlined letters for accessing menu commands are shown: Example: File/Open
Command Buttons	Commands in dialog boxes are shown as buttons: Example: ENTER
Categories, Radio Buttons, Text Boxes, Check Boxes	All options within dialog boxes are listed in italicized text: Example: the *Keep Group Together* check box the *Other* radio button
Keystrokes	Keyboard keys are indicated by uppercase text: Example: press ENTER Keyboard combinations are shown in uppercase text with a plus sign (+) between the keys that need to be pressed simultaneously. Example: press CTRL + S to save
Toolbar Buttons	Toolbar buttons are indicated by the button name and a graphic image of the button: Example: click the Print 🖨 button
Typing or File Selections	Text to be typed or file names to be selected are printed in bold letters: Example: type Henry select grouping.rpt
Exercises	Step-by-Step exercises in the text are indicated by bold text and the ❖ symbol. For example: ❖ Exercise - Format Objects

Tips, Notes, and Warnings

Tips, notes and warnings display with the following icons. Text for these additional comments display in bold and italics.



This icon indicates a TIP or shortcut.



This icon points out a NOTE of additional information.

This icon calls attention to a WARNING or very important note.

Notes

Notes

Lesson 1
Refresher Exercise

Lesson Objectives

After completing this lesson you will be able to:

- **Plan and create a report from a set of business requirements**
  Review the techniques needed to plan and create a report

- **Add multiple groups to reports**
  Create groups within groups to subdivide your report

- **Conditionally format your report**
  Make your report more informative and attractive with formatting

- **Parameterize your report**
  Use parameter fields to provide flexibility to the user as to the data they are viewing

- **Add multiple charts to a report**
  Use more then one chart and conditional formatting to graphically display data for different groups

Review of Planning a Report

To make the report creation process efficient, you must plan the report. A little work up front before you ever open Crystal Reports can save you a lot of time putting the report together. It can also save problems caused by adding more tables to the report that can change the number of records displayed.

You must look at the business requirements for the report and decide what fields you need and the tables in which they are located. It is also helpful to loom at things such as groupings, whether you need Cross Tabs and what fields they contain, fields needed for formulas and selection criteria.

Suppose someone asks you to create a report based on the Xtreme Sample Database 11. Below are the business requirements for the report:

- The first page of the report needs to show the top N sales people by product sales – N needs to be a user defined variable

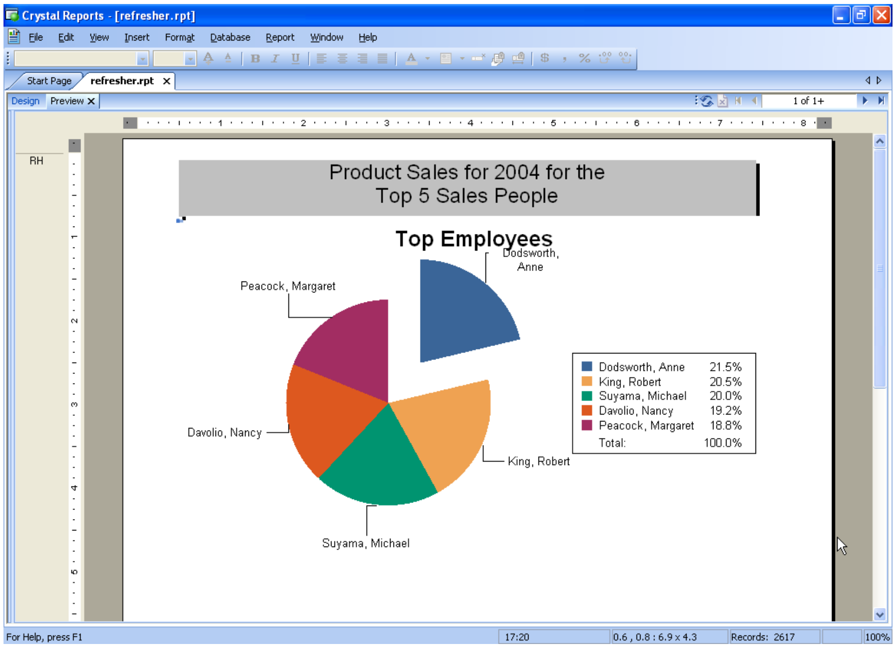
- The body of the report shows the sales detail for each sales person showing number of orders and amount by product type. Beside each sales person should be a bar chart showing order amount by product type for that sales person

- The user needs to be able to choose a year for the report with a default value of 2004

- Need to be able to see quickly any sales person with over a $1,000,000 of sales

- The person requesting the report has supplied you with a print out of what the report should look like – both the first page and detailed pages

The screen shot below is a view of the first page of the report:

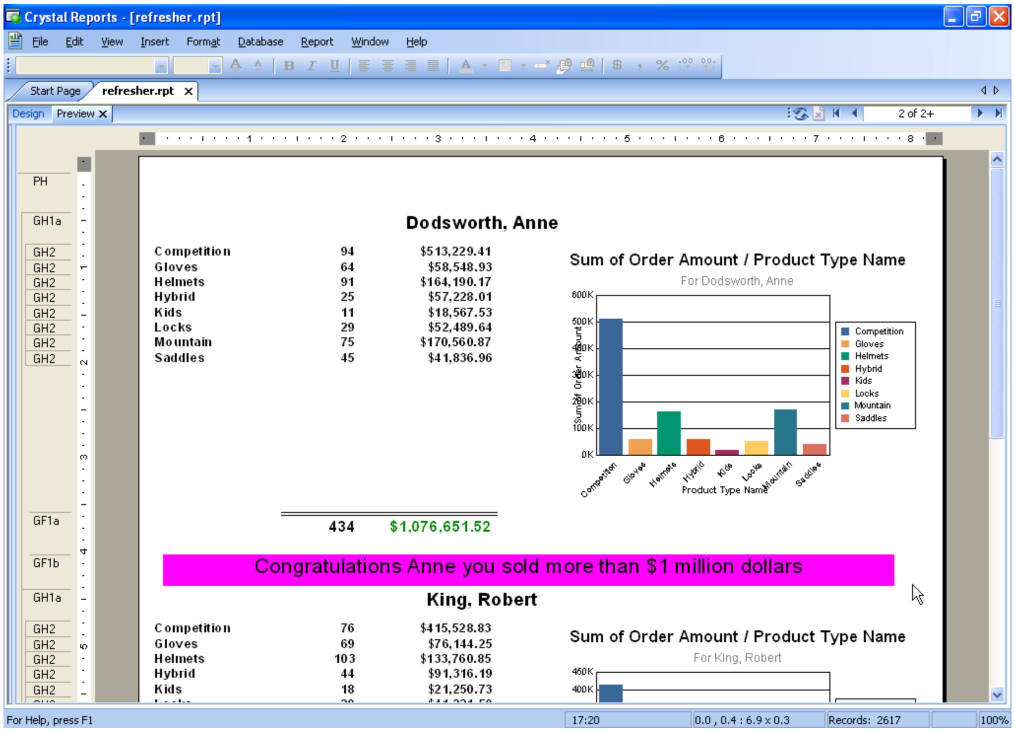


Product Sales for 2004 for the
Top 5 Sales People

Top Employees

🟦 Dodsworth, Anne	21.5%
🟧 King, Robert	20.5%
🟩 Suyama, Michael	20.0%
🟧 Davolio, Nancy	19.2%
🟪 Peacock, Margaret	18.8%
Total:	100.0%

Page 2 of the report should look as follows:



Dodsworth, Anne

Competition	94	$513,229.41
Gloves	64	$58,548.93
Helmets	91	$164,190.17
Hybrid	25	$57,228.01
Kids	11	$18,567.53
Locks	29	$52,489.64
Mountain	75	$170,560.87
Saddles	45	$41,836.96
	434	$1,076,651.52

Sum of Order Amount / Product Type Name
For Dodsworth, Anne

Congratulations Anne you sold more than $1 million dollars

King, Robert

Competition	76	$415,528.83
Gloves	69	$76,144.25
Helmets	103	$133,760.85
Hybrid	44	$91,316.19
Kids	18	$21,250.73

Sum of Order Amount / Product Type Name
For King, Robert

To create this report, first determine which fields you need.

- For the detail section you need order id and order amount
- For grouping you need Employee Id and Product Type Name

- Finally you need to select orders based on the Order Date

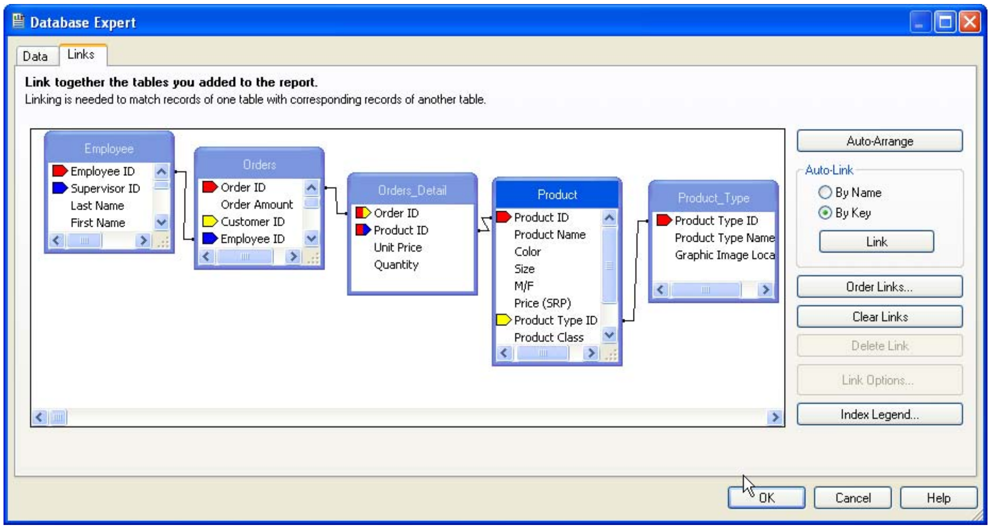
Next decide which tables you need to add to get the required fields.

- Order Id, Order Amount and Order Date are in Orders
- Employee Id and Name are in Employee
- Product Type Name field is in the Product Type table. To get to this table you need Order Details and Product tables

Since the Employee Id is the main grouping, the Employee table should be the primary table. From there you want to link Employee to Orders with the Employee ID; link Orders to Order Detail with the Order ID; link Order Details to Product with Product ID; link Product to Product Type with Product Type ID.

Creating the Report

Since you have given this some thought and figured out the tables and the links, adding the tables in the correct order and creating the links is easy. The links should look like the following illustration.



Link together the tables you added to the report.
Linking is needed to match records of one table with corresponding records of another table.

Remember that you can add multiple tables to a report from the Database Expert. You work with links using the Links tab on the Database Expert. The Database Expert is accessed in the Database menu or by selecting the Database Expert

*The order in which the tables are displayed in the **Link tab** is partly determined by the order they are displayed in the **Selected Tables** section of the **Data tab** in the **Database Expert**. Order and links of the tables can be manually changed once they are displayed in the **Link tab**.*

Placing Fields on the Report

Remember that you have five ways of placing fields on a report. Select the field in the Field Explorer, then:

- Click the Insert button, then click in the report
- Double click the field, then click in the report
- Press ENTER, then click in the report
- Drag the field to the report
- Right mouse click, choose Insert to Report from the short cut menu, then click in the report

Remember, when you place a field in the Details section, Crystal Reports adds a detail field title in the Page Header section and aligns the field and field title with a guideline. You can drag the guideline marker in the ruler to move the field and its title together.

Now that you have planned the report, you are ready to use Crystal Reports to create the report.

❖ Exercise 1.0 - Begin the Refresher Exercise Report and Link the Tables Needed

1. Start a new report *as a blank report*. Open the **Create a New Connection** folder then the **ODBC (RDO)** data sources and the **Xtreme Sample Database 11** data source. No password is needed to access this database. Add the tables indicated by the business requirements. Right click on the data source and select **Add to Favorites** as we shall be using this database throughout the class.

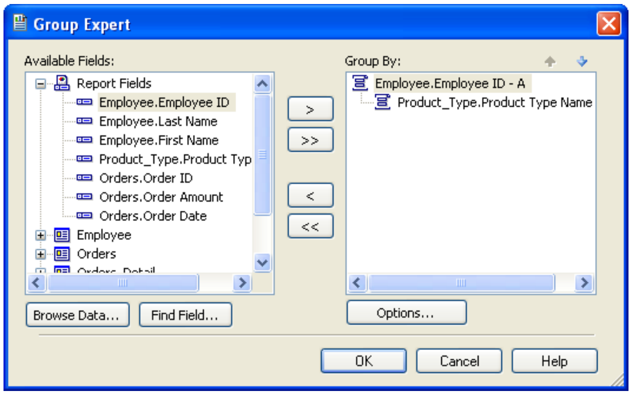
2. The **Auto-Link** feature should link the tables automatically. Clear the links and do them yourself.
   When you are finished the links should look like the illustration on the previous page.

3. Add the appropriate fields to the **Details** section.

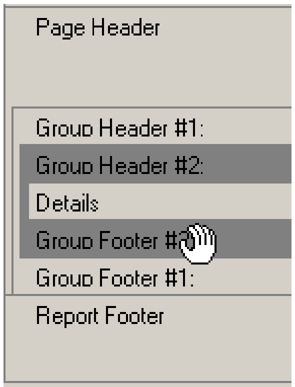
4. Change the **Printer Setup** to a Portrait page layout, if necessary. Change the margins to be .5” on all four sides.

Advanced Grouping

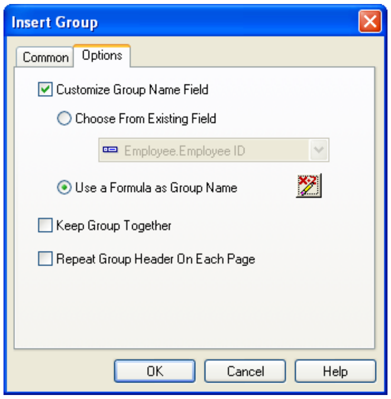
You can group data by using the Insert/Group… command or by clicking the Insert Group button on the Insert Tools toolbar or the Group Expert button on the Expert Tools toolbar. This report requires an Product Type Name group nested inside an Employee Name group, as shown in the Group Expert illustration below.



As you create groups, Crystal Reports nests each group inside the previous one you created. However, you can reorder groups in any order you want. To reorder groups, click the Group Header section name at the left of the Design screen, for the group you want to move. The group header and footer sections are highlighted. Drag the group to the new location. The mouse pointer changes to a grabbing hand as you drag. In the Group Expert shown above you simply move the group headers using the arrows in the top right of the window.



Remember that you can group on a field but display a formula or another field name as the group name field. When you group on Employee Id using the option shown below (Using a Formula as Group Name) you can display the employee name.



☑ Customize Group Name Field

○ Choose From Existing Field

Employee.Employee ID

● Use a Formula as Group Name

□ Keep Group Together

□ Repeat Group Header On Each Page

Remember that you can summarize any field by RIGHT clicking it, then choosing Insert Summary… You must specify the type of calculation you want Crystal Reports to perform and the group level at which you want the summary to appear.

❖ Exercise 1.1 – Create groups and summarize Order Id and Order Amount field.

1. Open the **Group Expert** dialog using the **Group Expert** button. Create a group on the **Product Type Name** field.

2. Create a second group on the **Employee Id** field. Choose to use a formula as a group name displaying the Employee Name as Last Name, First Name.

3. The report requirements call for the report to display information by Employee Id and then Product Type Name. You need to change the order of the groups. Select the **Product Type Name** field in the **Group By** window of the **Group Expert**. Select the Down Arrow to reorder the groups.
   The Employee Id should be the first group in the Group Expert with the Product Type name nested underneath.

4. Create a summary for the Order Id and Order Amount column. You will need to do this three times, once for each group.
   All the summary fields should automatically line up on the guideline to which the field is attached.

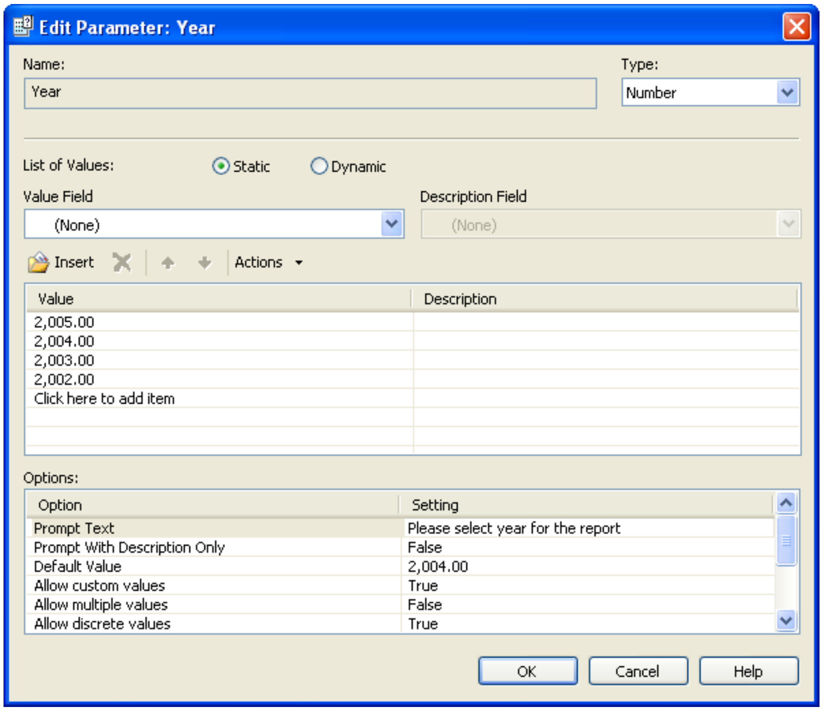
   NOTE: You can cut and paste the summary fields from one group to another.

Selecting Certain Records

Remember that Crystal Reports pulls *all* records from the database unless you filter the records based on some criteria. The business requirements for this report requests only records from a user selected year. First you need to create a parameter field and then use it in the record selection formula.

❖ Exercise 1.2 – Select records for a specific year

1. Create a parameter field called year with a numeric data type and a default value of 2004 as shown in the dialog below.



Edit Parameter: Year

Name: Year

Type: Number

List of Values: ● Static ○ Dynamic

Value Field: (None)

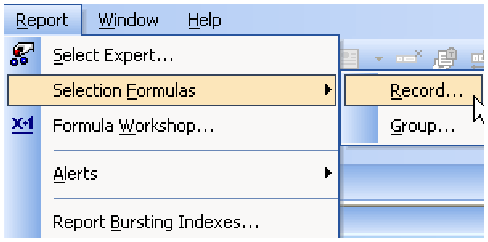
Description Field: (None)

Value	Description
2,005.00	
2,004.00	
2,003.00	
2,002.00	
Click here to add item	

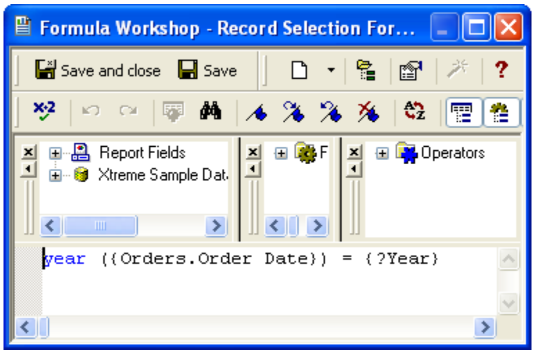
Options:

Option	Setting
Prompt Text	Please select year for the report
Prompt With Description Only	False
Default Value	2,004.00
Allow custom values	True
Allow multiple values	False
Allow discrete values	True

2. Under the Report Menu choose to edit the record selection formula as shown in the dialog below:



3. In the formula editor you need to provide the following formula:



```
year ({Orders.Order Date}) = {?Year}
```

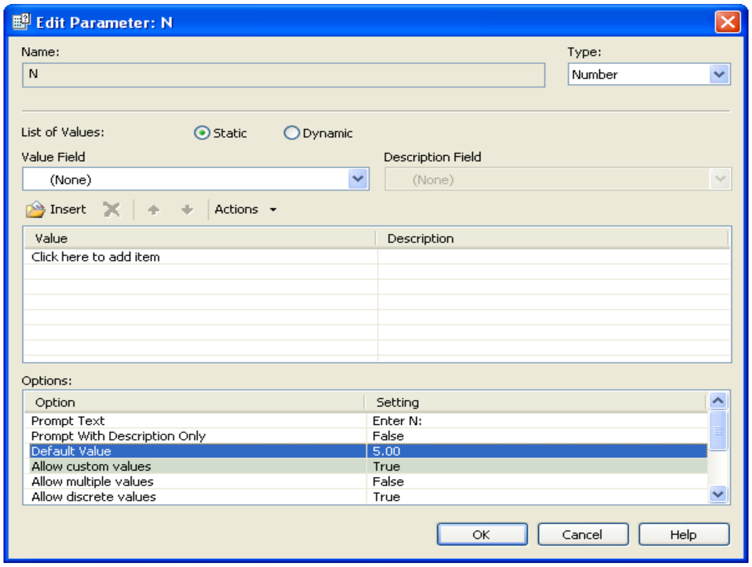
4. At this point save and preview the report. Name the report **Refresher.rpt**

Group Sorting

An additional requirement of this report is group sorting. Now you have the group summary fields you can choose how to sort the groups. One of the requirements outlined is the ability to look at the Top N sales people and top is measured by order amount. It is also a requirement to be able to select the value of N at run time.

❖ Exercise 1.3 – Sort Groups

1. Before you start sorting your groups you need to create a parameter field for N as the N value needs to be user defined. Create a simple parameter field with a default value of 5 as shown below.



Edit Parameter: N

Name: N

Type: Number

List of Values: ● Static ○ Dynamic

Value Field: (None)

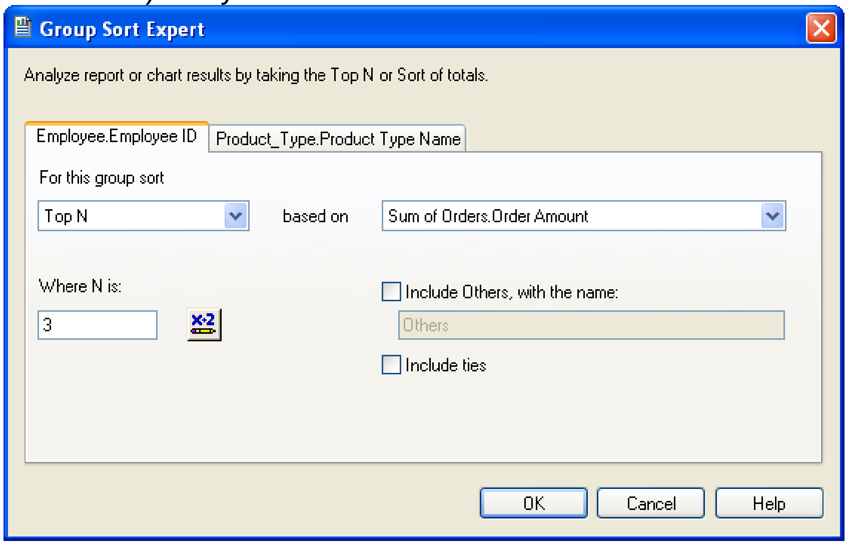
Description Field: (None)

Value	Description
Click here to add item	

Options:

Option	Setting
Prompt Text	Enter N:
Prompt With Description Only	False
Default Value	5.00
Allow custom values	True
Allow multiple values	False
Allow discrete values	True

2. Click on the Group Sort Expert and you will be able to select the type of sort for your group as shown below. The group sort expert has a tab for each group on the report and within that tab you can choose the summary field on which you want to base your sort. You then decide which type of sort (Top N, Bottom N, All Sort etc) that you would like to do.



Group Sort Expert

Analyze report or chart results by taking the Top N or Sort of totals.

For this group sort: Top N based on Sum of Orders.Order Amount

Where N is: 3

□ Include Others, with the name: Others

□ Include ties

3. In this case you want to sort on Employee ID; do a Top N sort and exclude Others.

4. In order for N to be user defined you need to click on the conditional box (next to Where N is). Simply add the parameter field ?N to the formula box.

5. Preview your report and select values for both parameter fields. Save your report.

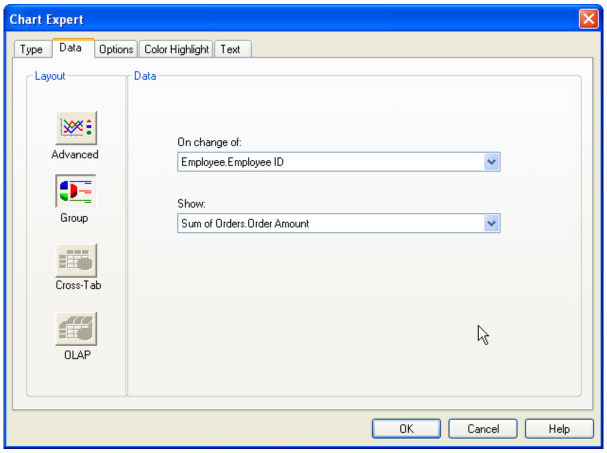
The Report Header

Now the report is providing the correct information before you begin formatting you should think about the business requirements for the report header. The report needs a pie chart on the front page with a text title that contains the parameter values for the report.

❖ Exercise 1.4 – Report Header

1. Insert a text object into the report header with an appropriate title for the report. Make sure you insert both parameter fields so their values at runtime are displayed. Format the title to stand out.

2. Insert a pie chart by clicking on the Chart Expert ; Choose a pie chart and then make sure the correct data is selected as shown below.



On change of: Employee.Employee ID

Show: Sum of Orders.Order Amount

3. Under Options you choose to show a label and also detach the largest piece of the pie as shown below.

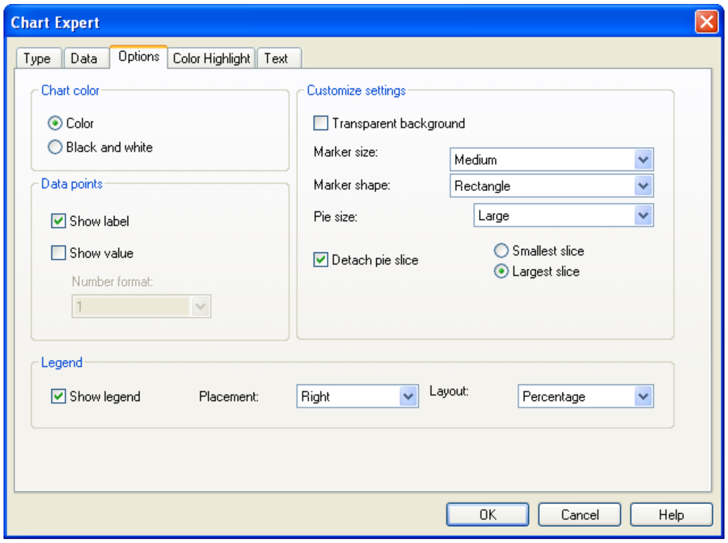


Chart color
● Color
○ Black and white

Data points
☑ Show label
□ Show value
Number format: 1

Customize settings
□ Transparent background
Marker size: Medium
Marker shape: Rectangle
Pie size: Large
☑ Detach pie slice
○ Smallest slice
● Largest slice

Legend
☑ Show legend
Placement: Right
Layout: Percentage

4. Finally in the section expert choose to format the report header with a new page after to make sure the report header is the front page of the report.

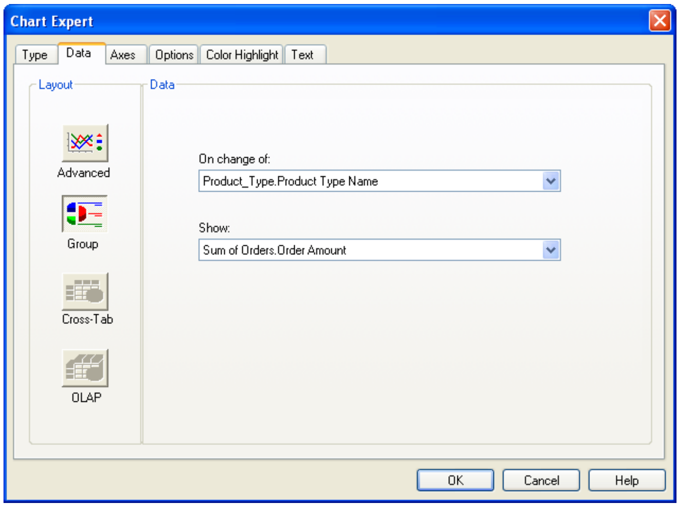
5. Save the report.

The Group Chart

As well as the chart in the report header there is also a chart in the main report for each employee. This chart is showing the product types sold for an employee.

❖ Exercise 1.5 – Group Chart

1. Click on the Chart Expert button . Choose a bar chart and then click on the data tab to choose Order Amount by Product Type Name as shown below.



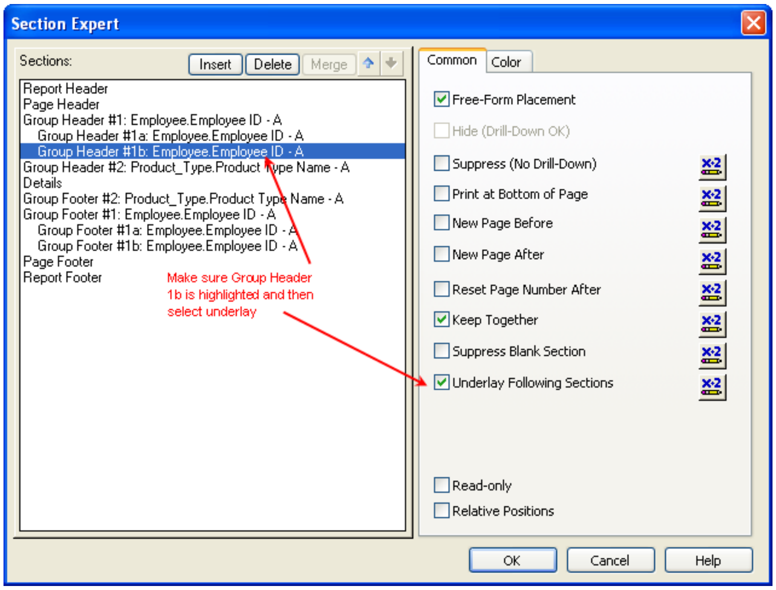
On change of: Product_Type.Product Type Name

Show: Sum of Orders.Order Amount

2. Insert the chart into Group Header 1. An additional requirement is to have the chart beside the detailed data for the sales person. To do this, use Underlay in the section expert. However you need to put the chart in its own section to prevent the group title from being underlayed as well.

3. Insert a new group header section for group 1. Move the chart to group header 1b. Reformat the group section 1a to remove the excess white space.
Hint use Fit Section.

4. In the section expert select Group Header 1 b and choose to underlay the following section.



Sections:

- Report Header
- Page Header
- Group Header #1: Employee.Employee ID - A
  - Group Header #1a: Employee.Employee ID - A
  - Group Header #1b: Employee.Employee ID - A
- Group Header #2: Product_Type.Product Type Name - A
- Details
- Group Footer #2: Product_Type.Product Type Name - A
- Group Footer #1: Employee.Employee ID - A
  - Group Footer #1a: Employee.Employee ID - A
  - Group Footer #1b: Employee.Employee ID - A
- Page Footer
- Report Footer

☑ Free-Form Placement
□ Hide (Drill-Down OK)
□ Suppress (No Drill-Down)
□ Print at Bottom of Page
□ New Page Before
□ New Page After
□ Reset Page Number After
☑ Keep Together
□ Suppress Blank Section
☑ Underlay Following Sections
□ Read-only
□ Relative Positions

Make sure Group Header 1b is highlighted and then select underlay

5. Save the report.

Your report is almost complete but needs further formatting to ensure any sales person who sold over $1,000,000 in a year needs to be recognized. You can achieve this with conditional formatting on the employee group footer adding a message when the sales are over $1,000,000.

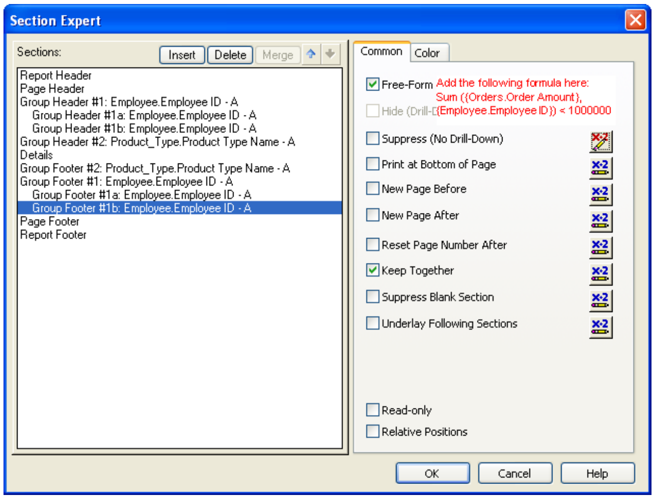
❖ Exercise 1.6- Conditionally Format the Report

1. Return to the **Design** view. It is easier to add and format objects from **Design** view.

2. Insert an additional group footer section for group footer 1 and add a text object to it with the sales person's first name embedded The text object would look similar to the following.

   Congratulations {First Name} you sold more than $1 million dollars

3. If you preview the report now you see that this text object displays for every sales person regardless of what they have sold. You now need to conditionally suppress the section. You need to section to suppress if sum of sales by employee id is less than $1,000,000. In the section expert add a formula to the conditional box next to the suppress option as shown below:



Sections:

Report Header
Page Header
Group Header #1: Employee.Employee ID - A
Group Header #1a: Employee.Employee ID - A
Group Header #1b: Employee.Employee ID - A
Group Header #2: Product_Type.Product Type Name - A
Details
Group Footer #2: Product_Type.Product Type Name - A
Group Footer #1: Employee.Employee ID - A
Group Footer #1a: Employee.Employee ID - A
Group Footer #1b: Employee.Employee ID - A
Page Footer
Report Footer

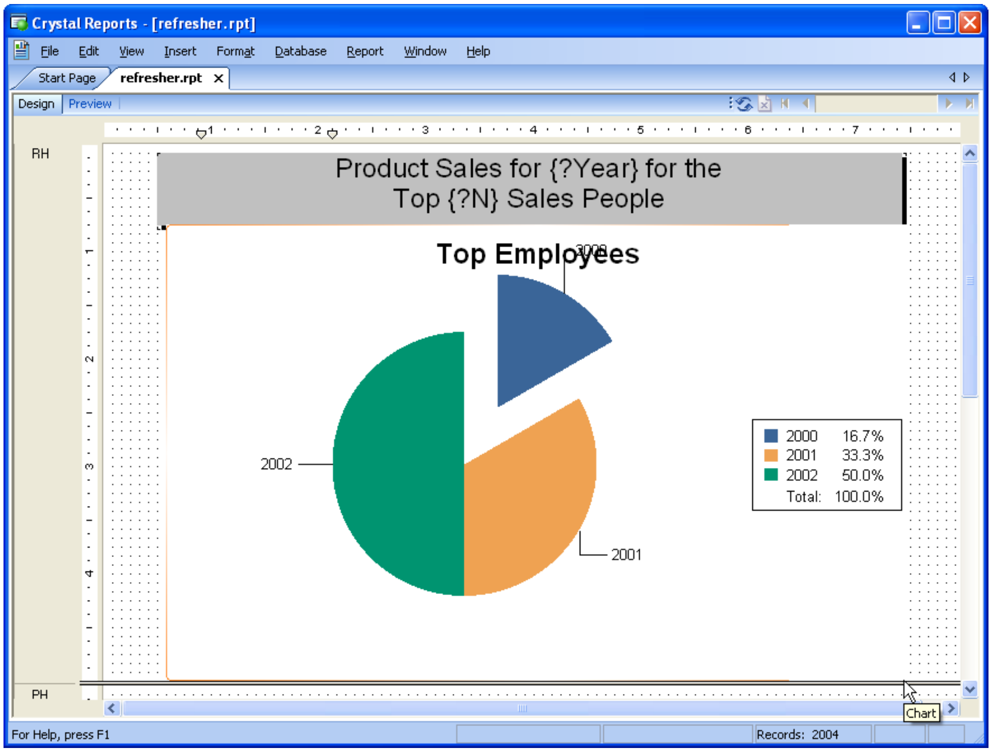
☑ Free-Form
□ Hide (Drill-D
□ Suppress (No Drill-Down)
□ Print at Bottom of Page
□ New Page Before
□ New Page After
□ Reset Page Number After
☑ Keep Together
□ Suppress Blank Section
□ Underlay Following Sections
□ Read-only
□ Relative Positions

Add the following formula here:
Sum ({Orders.Order Amount},
{Employee.Employee ID}) < 1000000

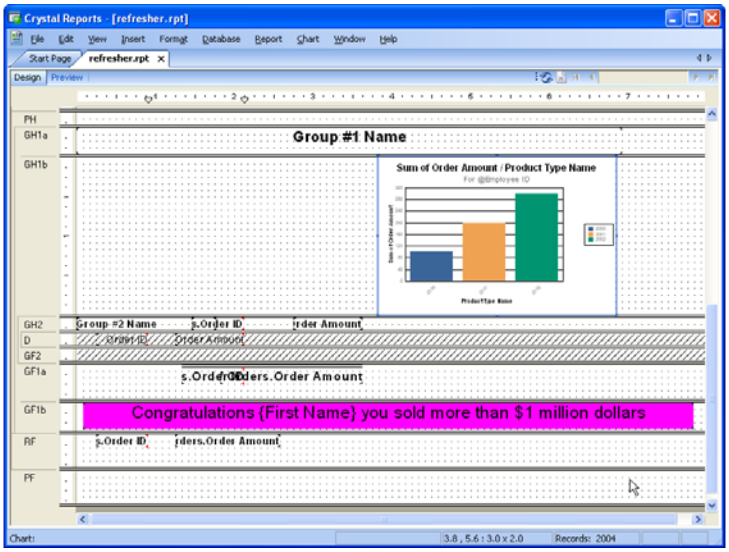
4. Save your report and run it with different values for N and the year. Make sure it meets the business specifications.

The Design view of your report should look as follows:

This is the Design View of the Report Header – note the chart is a place holder and does not show the data.



The Design view for the rest of the report should look as follows:



Group #1 Name

Group #2 Name   s.Order ID   rder Amount

Order ID   Order Amount

s.Orders.Orders.Order Amount

Congratulations {First Name} you sold more than $1 million dollars

s.Order ID   rders.Order Amount

NOTES

Lesson 2

Custom Functions

Lesson Objectives

After completing this lesson, you will be able to:

- **Describe a Custom Function**
  Understand what a custom function is and when to use them

- **Creating Custom Functions**
  Learn the 2 ways to create a custom function by either extracting from a formula or creating from scratch

- **Using the Formula Expert and Custom Functions**
  Learn about the formula expert and how it relates to custom functions

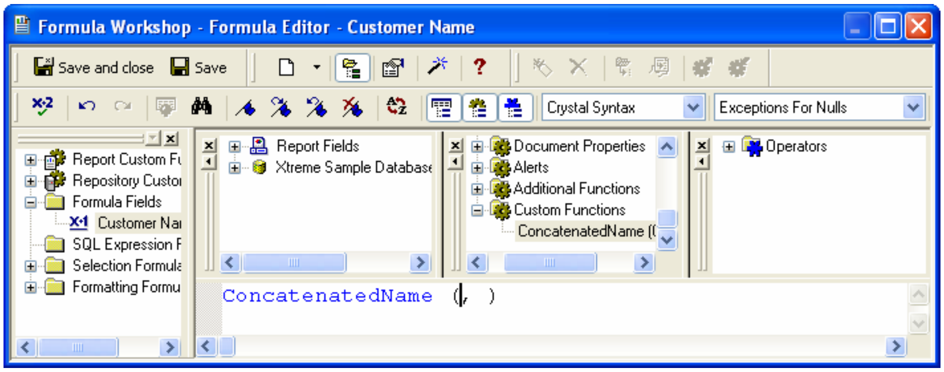
What is a Custom Function

The Formula Workshop of Crystal Reports is made up of many functions, such as the ToText function that converts a number value to a text object, or the CurrentDate function that returns the current system date to the report. These functions have been created by Business Objects and included as common functions within the application. However, you may have requirements for other functions not included in the Formula Workshop. When this happens, you are presented with the dilemma of creating these functions yourself. Any function you create is considered to be a custom function within the Crystal Reports application. Custom functions are different from formulas. Functions allow you to specify the procedure by which calculations, evaluations or comparisons are done. The function does not contain field objects, but rather custom functions contain arguments that you, as the report developer, can specify when you use it in a formula. Custom functions can involve basic or advanced logic, depending on the need. The key benefit of creating custom functions is the ability to utilize the functions' logic from report to report.

For example, suppose you have a database with First Name and Last Name fields for Employees, Customers, Vendors, Investors, etc. Now consider the number of reports used in your company that may contain these names. Suppose the First Name field may or may not be populated based on the table you are looking at, i.e., the Employee table has all fields with data, but the Customer table requires only the Last Name field to have data. You could write different formulas to take all of the possible options into consideration. However, you may or may not remember the logic every time you need to create a new name formula. Or you could create the logic within a custom function and then simply specify the arguments when you need them in a report. Using custom functions also helps users who are unfamiliar with advanced formula creation, to easily create formulas by specifying the fields to use in the formula.

NOTE: Custom Functions are designed to be shared amongst reports and report developers. Custom functions can be tied to the Crystal Repository. You are only able to access the Repository if you have Crystal Enterprise established within your organization. This lesson focuses on how to set up custom functions within a single report. If you are using Crystal Reports Professional or Developers additions, a limited version of Business Objects Enterprise is located on a separate installation CD. By installing and configuring this component, you will be able to access the Repository and store custom functions you create to share with others.

Custom Functions that have been created in a report appear at the bottom of the functions list in the Custom Function folder as shown below:



```
ConcatenatedName (, )
```

Custom Function Arguments

Custom Functions like Crystal Reports defined functions have arguments. An argument is the value that is passed to the function in your formula. For example, if you use the ToText function in its simplest form it has one argument that you supply such as a database field that you would like converted to text. ToText can also have 2 arguments where you supply, both a database field and number of decimal places. In fact, ToText can have up to 5 arguments.

When you establish custom functions you will also define arguments that can be populated in the Formula Workshop. The name of the arguments can be as simple as v1, v2 or more defined as FirstName, LastName. The same rules for naming variables apply to the names of the arguments.

This is an example of function logic that contains 2 arguments:

//This function can be used to concatenate string fields such
//as FirstName, and LastName fields in a data table.

Function (stringvar v1, stringvar v3)

v1 + " " + v3

When to use a Custom Function

It may sometimes be hard to know when to use a custom function and when to use a formula. Here are some advantages of Custom Functions that may help in that decision process:

- **Repetitive code**: Custom functions can replace sections of formula logic common to many different formulas that, in turn, may be used in many different reports
- **Sharing:** Custom functions can be shared between reports by storing them to the Crystal Repository and adding them to reports as needed
- **Simplicity**: Custom function can reduce the need for your users to write formulas themselves; you can provide them with the formula logic they need in a custom function
- **Flexibility:** Custom functions can have arguments, which make them flexible
- **Ease of Use**: You can define help text and default argument values for custom functions making it easier for your users to use them

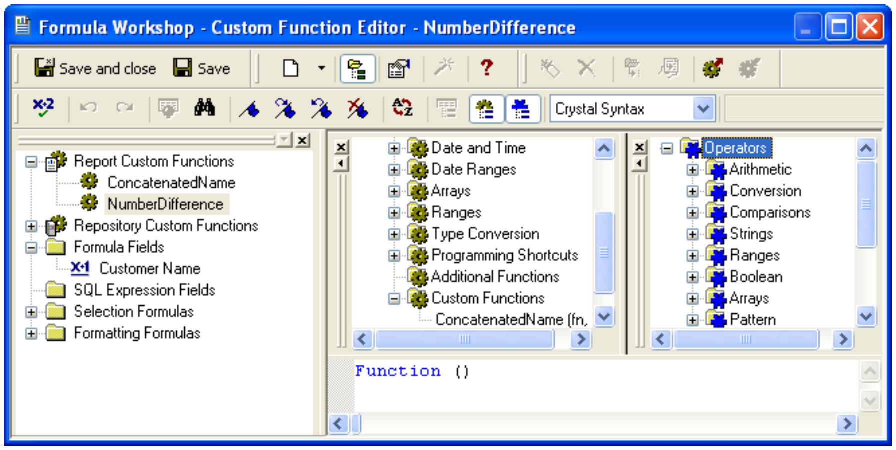
NOTE: Remember, to gain the full benefits of using custom functions it is best to have Business Objects Enterprise configured so these objects can be shared via the Business Objects Enterprise Repository.

Creating Custom Functions

There are two ways to create a custom function. One way is to extract the function from a formula. The other is to use the Custom Function Expert. If you do extract it from a formula and want to make changes to it, you will have to use the Custom Function Editor.

Creating a Custom Function using the Custom Function Editor

A Custom Function like a formula can be created using the Custom Function Editor, which is very similar to the Formula Editor without the database field tree. Instead of choosing Use Extractor when creating a new Custom Function, choose Use Editor and you will be presented with the following dialog:



```
Function ()
```

Notice this dialog is very similar to the Formula Editor except it does not have the field tree which shows report and database fields. Database fields cannot be included in a custom function since functions are designed to be independent of a report. If a custom function includes database fields and is added to the repository there is no guarantee that the next report to use the custom function will have access to that database field.

Before we start creating our custom function we need to decide which syntax to use - Crystal or Basic. This really depends on which syntax is more familiar to you. To create your custom function, simply begin typing or selecting functions and operators from their respective trees. Let's create a simple custom function which adds two numbers together. We need to declare two number variables, num1 and num2 and then type the logic to add them together.

Depending on the syntax you selected, the structure of the function will look like one of the following examples:

Using Crystal Syntax

```
//Function name Add2NumbersBasic
Function(numbervar num1, numbervar num2)
Num1 + num2
```

Using Basic Syntax

```
Function Add2NumbersBasic(num1 as Number, num2 as number)
as Number
    Add2NumbersBasic = num1 + num2
End Function
```

❖ Exercise 2.0 – Creating Custom Functions for Report Usage

The goal is to create Customer Information report listed by Employee. You will create one function that can be utilized to join two objects together. This function will be used in several formulas.

1. Create a new report *As A Blank Report*. Add the **Customer, Orders and Employee** tables. Verify your links and click **OK**.

2. Click the **X+1** button on the **Expert Tool** bar to enter the **Formula Workshop**.

3. Select the **New button** and create a **Custom Function**.

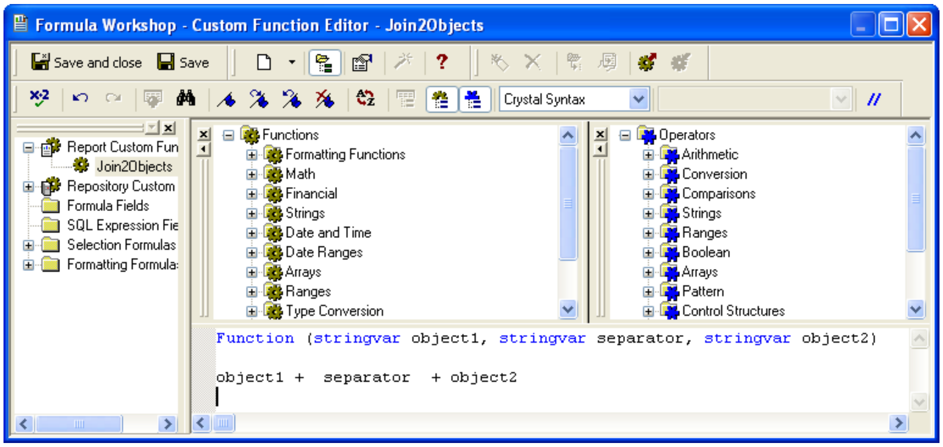
4. Name the function **Join2Objects**. Select the **Use Editor** button.

5. This function will allow the user to concatenate two objects together with a separator of the users choosing. The function structure should look like the output below:

   Function (stringvar object1, stringvar separator, stringvar object2)

   object1 + separator + object2

6. Check the function to make sure you have no errors and select the **Save** button.

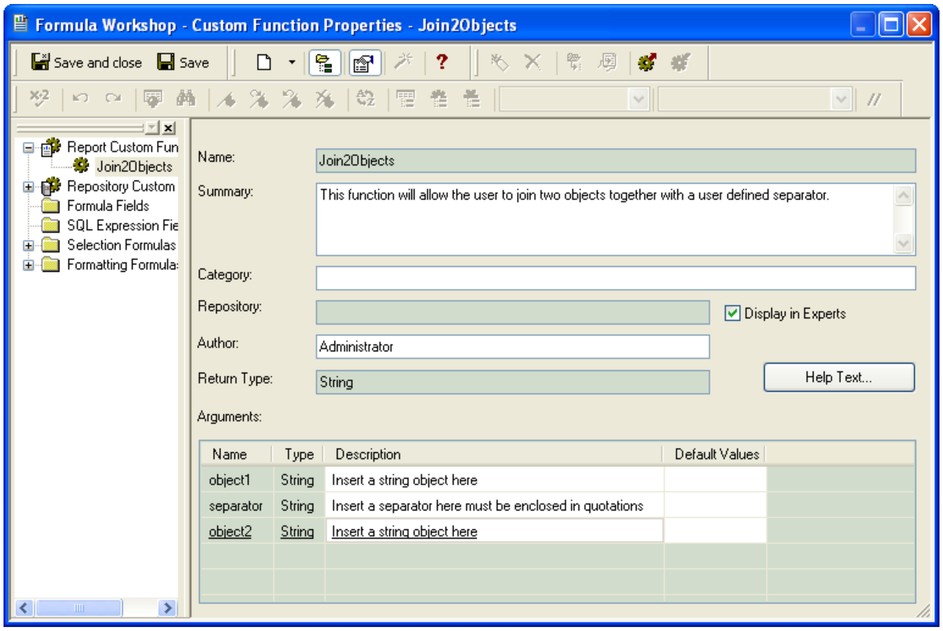


```
Function (stringvar object1, stringvar separator, stringvar object2)

object1 +  separator  + object2
```

Custom Functions Properties

When a custom function is created, you have the ability to provide more information about the custom function through the button. By providing more information, report developers who may utilize the function in the future will have a more thorough understanding about the logic of the function.



Name: Join2Objects

Summary: This function will allow the user to join two objects together with a user defined separator.

Category:

Repository:

☑ Display in Experts

Author: Administrator

Return Type: String

Help Text...

Arguments:

Name	Type	Description	Default Values
object1	String	Insert a string object here	
separator	String	Insert a separator here must be enclosed in quotations	
object2	String	Insert a string object here	

- **Summary**: Provides a detailed description of how the custom function works

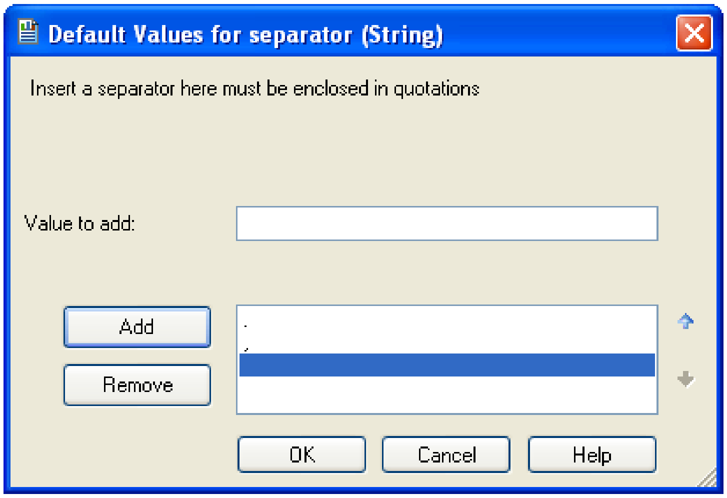
- **Category**: This allows you to set up hierarchies for your custom functions similar to a folder structure. The categories appear as subfolders in the Report Custom Functions folder of the Formula Workshop tree view and also in the Functions window of the Editor dialog box. To create further levels use a forward slash in the category definition

- **Repository**: This would show the repository name if you had added the custom function to the repository. Unless you have a valid Repository setup via Business Objects Enterprise, do not populate this text window

- **Display in Experts**: By checking this box you will see the custom function in the Formula Expert otherwise you will only see it in the Formula Editor

- **Author:** Generally the name of the person who created the custom function

- **Default Values:** You can supply a value or a list of values that a user can choose from when using the Custom Function in the Formula Expert. When you click on the box you will see the following dialog:



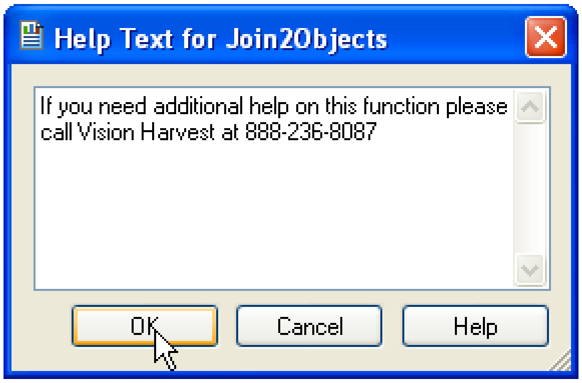
Insert a separator here must be enclosed in quotations

Value to add:

.
,

NOTE: Click Add to add values to the list. Use the arrows on the right to prioritize the values.

- **Help Text:** Here you can type in text that may be viewed in the Formula Expert for the user using the function



If you need additional help on this function please call Vision Harvest at 888-236-8087

❖ Exercise 2.1 – Modify the Custom Function Properties

1. With the **Formula Workshop** open, select the Properties button on the toolbar.

2. Specify the following text in the summary text box:

This function will allow the user to join two string objects together with a user-defined separator.

3. Create a Category **Strings/**.

4. Provide basic descriptions for each string argument.

5. Set *default values* for the separator to a dash (**-**) and a comma (**,**).

6. Add additional help text by selecting the **Help Text** button. When finished click **OK**.

7. Click the **Save** button to close the function setup, but remain in the **Formula Workshop.**

❖ Exercise 2.2 – Use a Report Custom Function in a Formula

You are now ready to use the function in a formula. The first formula you need to create will concatenate the Customer ID field to the Customer Name field.

1. With the **Formula Workshop** open, select the **New** button and create a new formula.

2. Name it **Customer** and click **OK**.

3. Locate the **Join2Objects** function in the **Custom Functions** list. Notice that the arguments you specified in the functions design are displayed.

4. Join the **Customer ID** and the **Customer Name** fields together with a **dash (-)** as the separator. You need to convert the **Customer ID** field from a Number to a String. Specify 0 decimal places and no thousand separator.

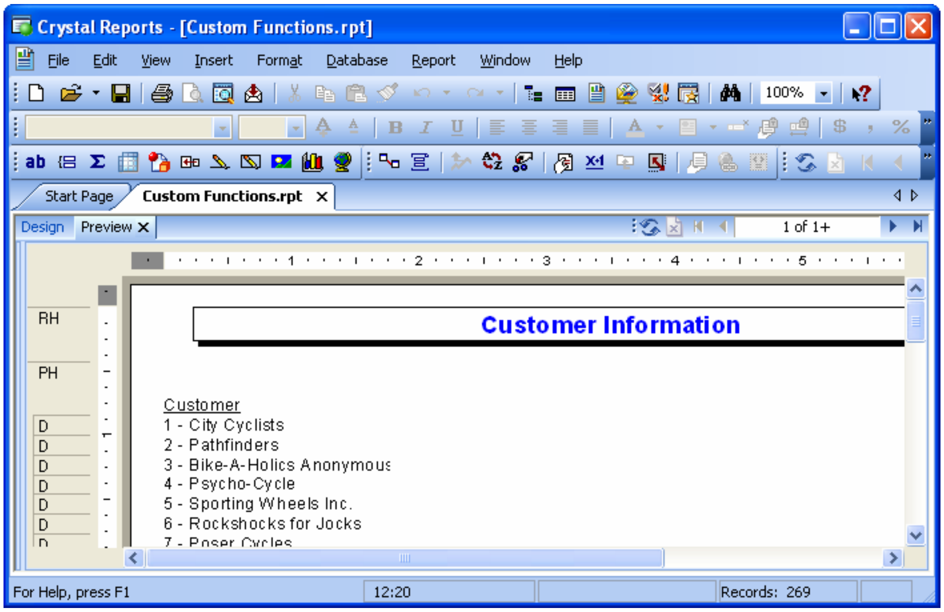
The final formula will look like the following:

**Join2Objects (ToText({Customer.Customer ID},0,""), " - " ,
{Customer.Customer Name})**

5. Save the formula and place it into the **Details** section of the report.

6. Add a text object to the Report Header with the text **Customer Information**. Format the text object so it looks good to you.

7. Save the report as **Custom Functions.rpt** and **Preview** the results.

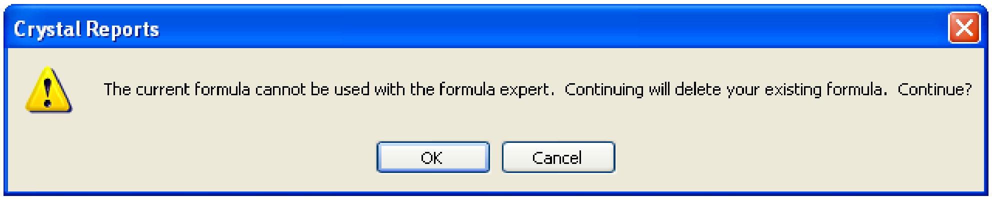


Customer Information

Customer
1 - City Cyclists
2 - Pathfinders
3 - Bike-A-Holics Anonymous
4 - Psycho-Cycle
5 - Sporting Wheels Inc.
6 - Rockshocks for Jocks
7 - Poser Cycles

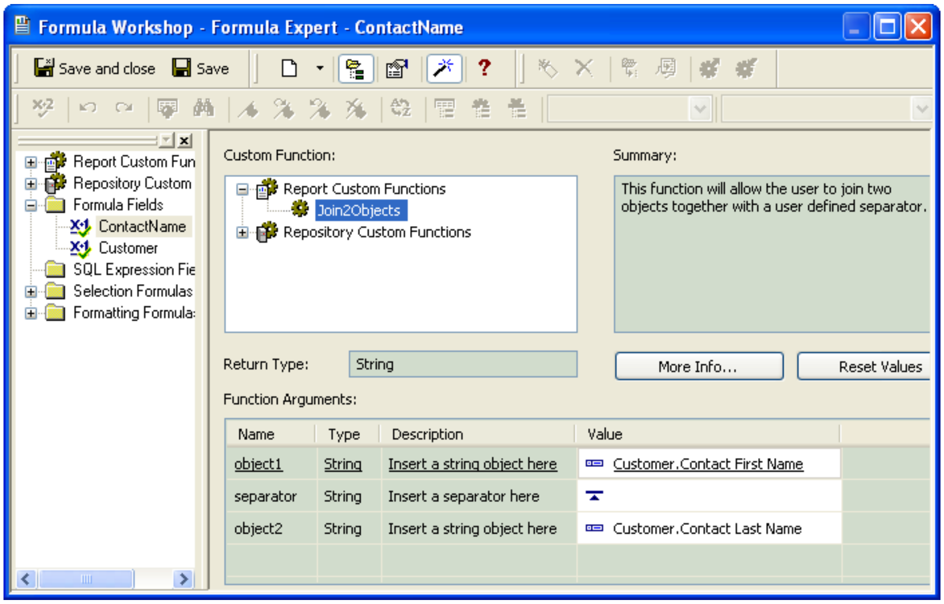
Using the Formula Expert and Custom Functions

The Formula Expert allows a user who is unfamiliar with the formula language to create formulas. There is one major caveat though and that is the formulas created using the Expert can only use custom functions. To use the expert click on the toggle Expert/Editor button while editing your formula. If the formula you are editing contains any database fields then you will not be able to use the expert and will get the following error message if you click the expert button.



The current formula cannot be used with the formula expert. Continuing will delete your existing formula. Continue?

Suppose you want to create a formula to join the Contact First Name and the Contact Last Name together in one formula. Although there are several ways to accomplish this, you are going to use the **Join2Objects** custom function created in the previous lesson.



Custom Function:

- Report Custom Functions
  - Join2Objects
- Repository Custom Functions

Summary:

This function will allow the user to join two objects together with a user defined separator.

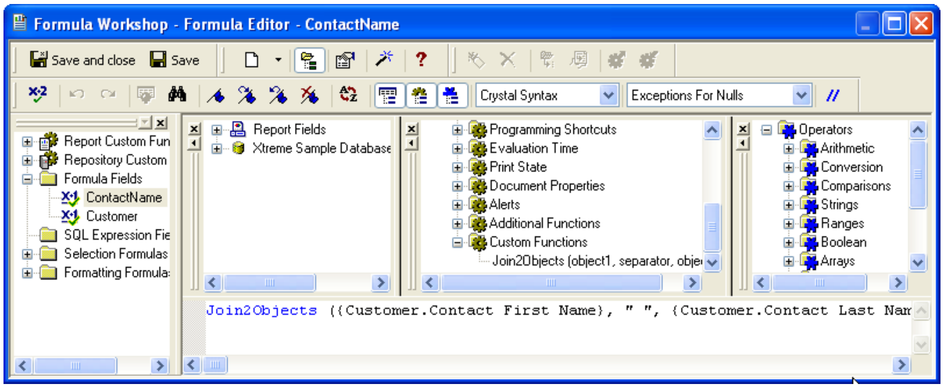
Return Type: String

Function Arguments:

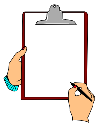
Name	Type	Description	Value
object1	String	Insert a string object here	Customer.Contact First Name
separator	String	Insert a separator here	
object2	String	Insert a string object here	Customer.Contact Last Name

Notice that the Value column prompts you to Select Field or Enter Value. This column is where you define the value for the argument of the function. To select a value, simply click the drop down list and select the Contact First Name field as object1 and Contact Last Name as object2 and choose a default value or enter a value for the separator.

To view the function in the editor, select the button. You can see by the illustration that the function along with its argument has been defined.



```
Join2Objects ({Customer.Contact First Name}, " ", {Customer.Contact Last Nam
```



NOTE: If this function is stored in the Repository and not currently being used in the report, you will be prompted to add it.

❖ Exercise 2.3 - Using the Formula Expert

The goal of this exercise is concatenate the Contact First Name and Contact Last Name together with a dash (-) as well as the Employee First Name and Last Name. You will group on the Employee formula to organize the data by employee.

1. Using the **Custome Functions.rpt** report, create a **New Formula** titled **Contact Name**. Click on the toggle button to use the expert.

2. To locate the function, expand the *Report Custom Functions* heading and select the **Join2Objects** function.

3. Locate **Object1** and click the **value** column. In the drop down list select the *Choose Other Field* option. This will display the **Choose Field** window. Select the **Contact First Name** field from the list and click **OK**.

4. Specify a **space** for the separator by simply adding it to the value column.

5. Set **Object2** equal to the **Contact Last Name** field.

6. When finished, save and close the **Formula Workshop** and place the Contact Name formula into the **Details** section of the report and **Preview** the results.

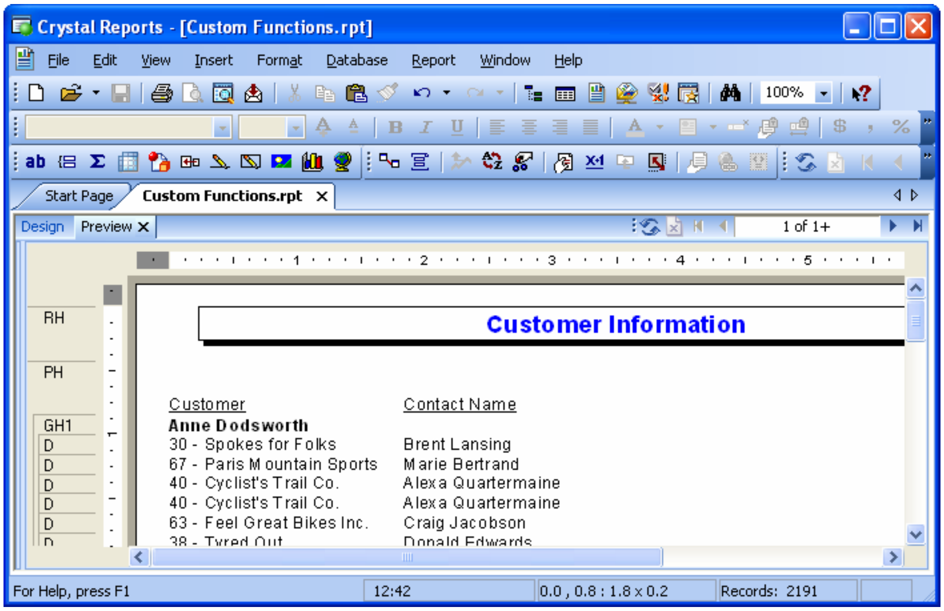
7. Create a new formula named **Employee** and use the toggle button to use the Expert.

8. Use the **Join2Objects** function to join the **Employee First Name** field as **Object1** and the **Employee Last Name** field as **Object2** with a **space** separator.

9. Save and close the **Formula Workshop**.

10. Select the **Group Expert** button and create a new group based on the **Employee** formula.

11. Save the report as (**Custom Functions**) and Preview the results.



Customer Information

Customer	Contact Name
Anne Dodsworth	
30 - Spokes for Folks	Brent Lansing
67 - Paris M ountain Sports	M arie Bertrand
40 - Cyclist's Trail Co.	Alexa Quartermaine
40 - Cyclist's Trail Co.	Alexa Quartermaine
63 - Feel Great Bikes Inc.	Craig Jacobson
38 - Tyred Out	Donald Edwards

For Help, press F1 | 12:42 | 0.0 , 0.8 : 1.8 x 0.2 | Records: 2191

Using the Formula Extractor

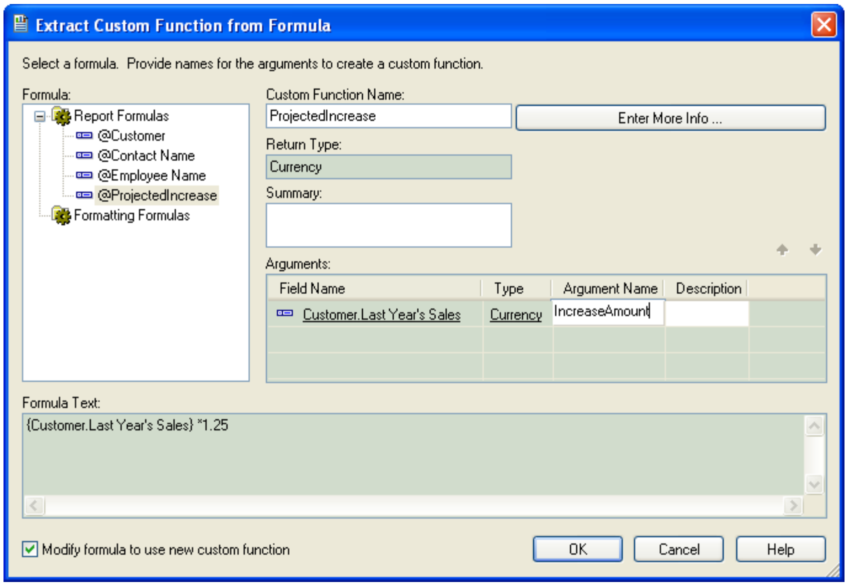
When you extract a custom function from an existing formula, Crystal Reports copies the formula and replaces its fields with arguments. The number of arguments depends on the number of unique database fields in the formula.

NOTE: A custom function created using the Extractor can only return simple types, like strings and integers. It cannot return arrays.

In order to extract a custom function from an existing formula, you must use the Formula Workshop. When a new function is created, a function name must be provided and the Use Extractor option selected. Crystal will display the Extract Custom Function from Formula dialog box. This dialog box displays a list of all of the formulas contained in the report. As you select the individual formulas in the *Formula* list, the properties, arguments and formula text change.

There are several key elements within the extractor that cannot be modified. The Return Type, Field Name and Type are fixed. They cannot be edited. You can, however, provide more description argument names. The default name Crystal provides is v1, v2, v3, etc. These are generic names and can be difficult to keep up with and possibly confusing if there are many arguments in the function. Changing the name to a more descriptive argument may help eliminate some of the potential confusion.

When you work with the extractor to create custom functions, Crystal also provides you with the ability to update the content of the original formula. By placing a check mark in the *Modify formula to use new custom function* option, the formula will be updated once the custom function has been completed.


Extract Custom Function from Formula

Select a formula. Provide names for the arguments to create a custom function.

Formula:
- Report Formulas
  - @Customer
  - @Contact Name
  - @Employee Name
  - @ProjectedIncrease
- Formatting Formulas

Custom Function Name: ProjectedIncrease

Return Type: Currency

Summary:

Arguments:

Field Name	Type	Argument Name	Description
Customer.Last Year's Sales	Currency	IncreaseAmount	

Formula Text:

```
{Customer.Last Year's Sales} *1.25
```

☑ Modify formula to use new custom function

❖ Exercise 2.4 - Extract a Custom Function from a Formula

The goal of this exercise is to create a function showing this year's projected sales amount which is estimated to be a 25% increase over last year's sales.

1. Open the **Custom Functions.rpt** report.

2. Create a formula called **This Years Sales** which increases the **Last Year's Sales** field by 25%. Your formula syntax should be as follows:

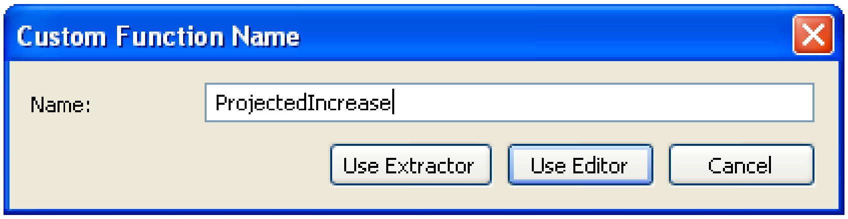
   {Customer.Last Year's Sales} * 1.25

3. Save and close the new formula.
   Hint: Pressing Ctrl + S will save and close the formula at the same time.

4. Open the **Formula Workshop** by clicking the X+1 button.

5. Select **Report Custom Functions**, RIGHT-click and select **New...**

6. Enter the name **ProjectedIncrease** and click to **Use Extractor** as shown below.



Name: ProjectedIncrease

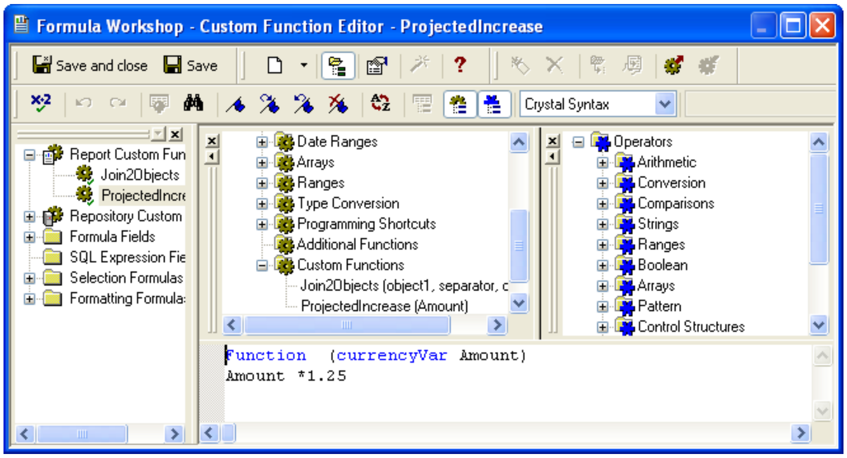
7. Locate the **This Year's Sales** formula.

8. Change the variable name to **Amount** instead of v1.

9. Click the **Enter More Info…** button and add a description to the *Summary:* section for this variable. Click **OK** when done.

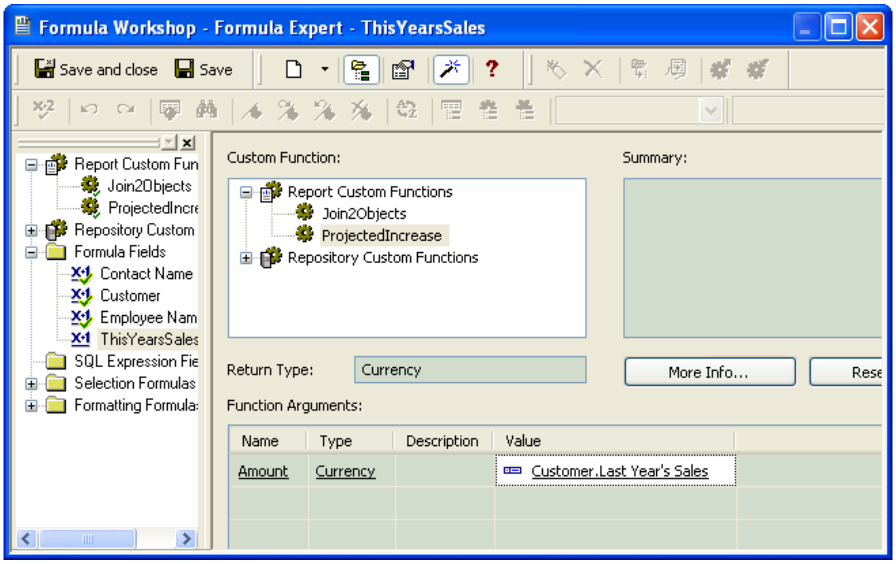
10. Before closing the **Extract Custom Function** from **Formula** dialog box, check the *Modify formula to use new custom function* box so the original formula uses this function.

11. Click **OK** and you should see the function in the formula workshop as shown below:



```
Function  (currencyVar Amount)
Amount *1.25
```

12. If you select the original formula and choose to use the editor you will see the formula syntax using the custom function as shown below:



Return Type: Currency

Function Arguments:

Name	Type	Description	Value
Amount	Currency		Customer.Last Year's Sales

13. Return to the **Design** view and add the **Last Year's Sales** field and the **This Year's Sales** formula to the **Details** section of the report. Save the report (**Custom Functions**) and Preview the report.

Custom Function Limitations

As discussed above one of the key features of custom functions is they are able to be shared among reports via the repository. This independence poses some limitations as follows:

- Custom functions cannot contain database field items or items derived from database fields such as summaries. Functions are independent of data sources containing only the logic to be used in a formula

- There are some naming limitations to custom functions

    - Custom Functions must begin with a letter and you can only include letters, numbers and _ in the name. No other special characters are allowed
    - Custom Function names CANNOT contain spaces
    - The Custom Function name has to be unique i.e. it cannot have the same name as a Crystal Reports function like CurrentDate

- Shared and global variables are not allowed in custom functions. You can, however, pass these variables through arguments of a custom function when it is utilized in a formula. Local and argument variables are allowed

- Custom functions cannot use direct or indirect recursion

- UFL’s cannot exist directly in a custom function

- Evaluation times, document properties, or print states cannot be used by custom functions. Functions pertaining to Cross-Tab dimensions are not available for use. These types of functions will not display in the Custom Functions Editor when a function is being edited

Challenge Exercise – Creating and Using Custom Functions

Let's say the criteria for getting a bonus is whether your salary is over a certain amount. This information is used in many reports so we want to create a custom function in order to make the formula reusable in several reports.

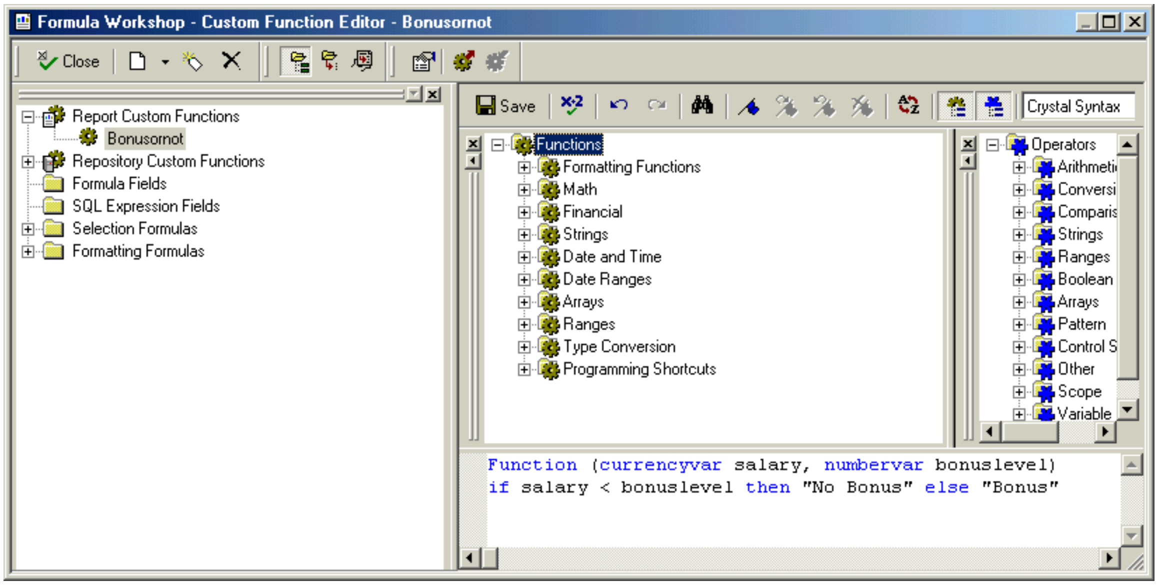
1. Create a blank report using the **Employee** table from the **Xtreme Sample Database 11**.

2. Insert the **Employee Last Name** , **Hire Date** and **Salary** fields into the **Details** section.

3. Add a text object to the Report Header section with the text "**Employee Bonus Listing**".

4. Open the **Formula Workshop** and select **Report Custom Functions**, RIGHT-click and select **New…**

5. Create a custom function called **Bonusornot**. Our custom function will have two variables, a currency variable called **salary** and a number variable called **bonuslevel**. The logic is: if the salary < bonuslevel then no bonus else bonus. Your function should look as follows:



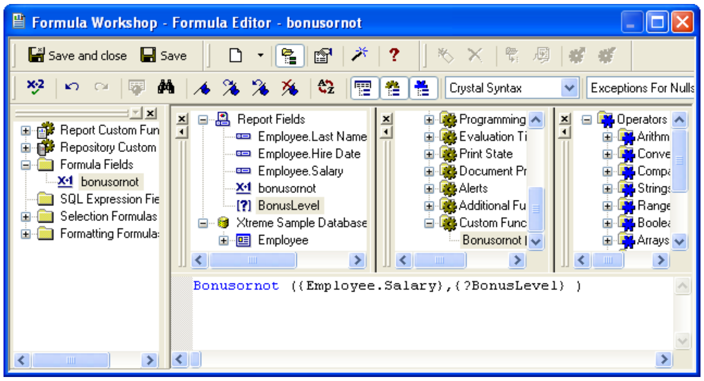
```
Function (currencyvar salary, numbervar bonuslevel)
if salary < bonuslevel then "No Bonus" else "Bonus"
```

6. Save the custom function and close the **Formula Workshop**.

7. Now create a **Number Parameter** field called **Bonuslevel** with a default value of **50000**.

8. Create a formula field called **bonusornot**.

9. Scroll down the custom functions at the bottom of the **Functions** tree and select the **Bonusornot** custom function. Our two parameters will be the **Employee.Salary** field and the **Bonuslevel** parameter field we just created. Your formula should look as follows:

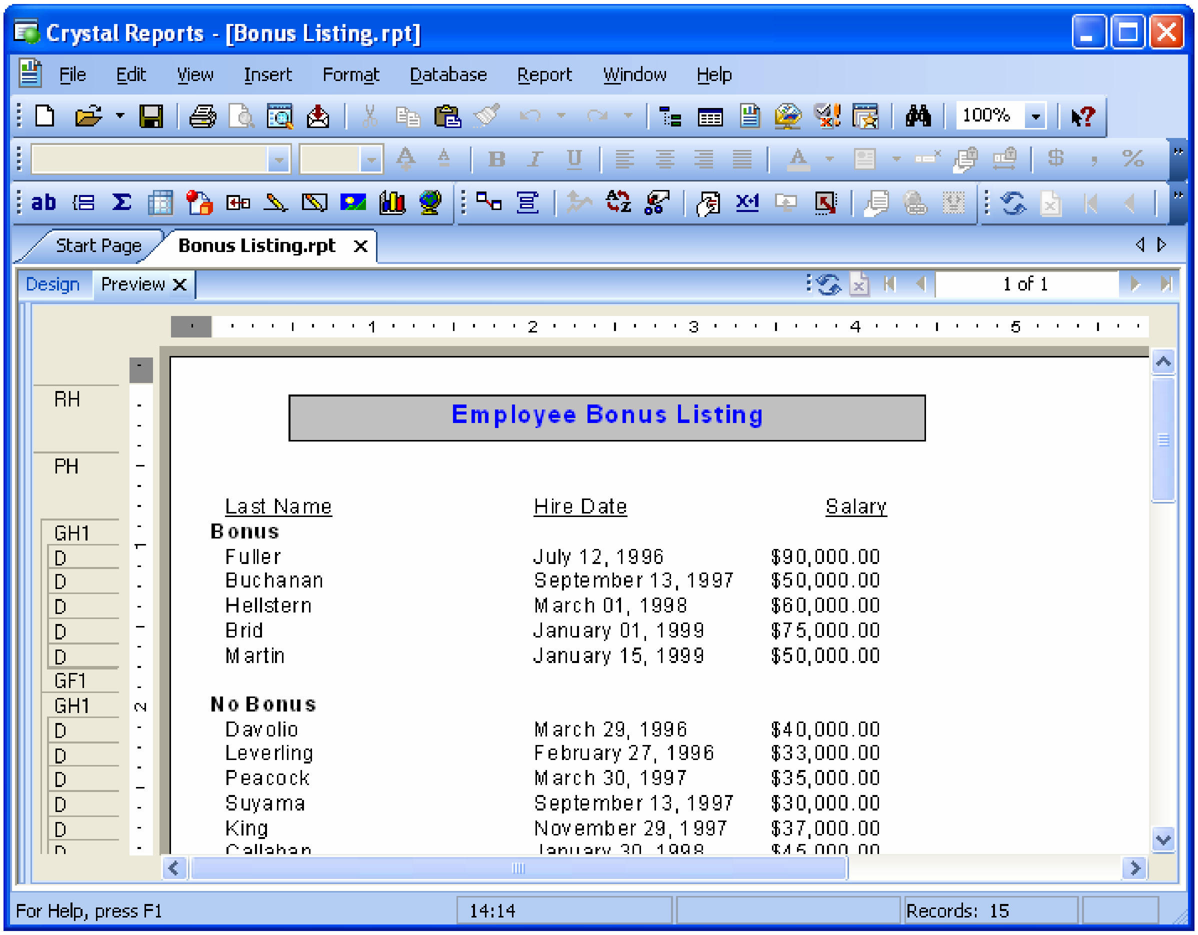


```
Bonusornot ({Employee.Salary},{?BonusLevel} )
```

10. Insert a group on the **bonusornot** formula field and run the report with the default bonus value.

11. Save the report and preview the results. Your report should look similar to the sample on the next page.

12. Save the report as **Bonus Listing.rpt**.



Employee Bonus Listing

Last Name	Hire Date	Salary
Bonus		
Fuller	July 12, 1996	$90,000.00
Buchanan	September 13, 1997	$50,000.00
Hellstern	March 01, 1998	$60,000.00
Brid	January 01, 1999	$75,000.00
Martin	January 15, 1999	$50,000.00
No Bonus		
Davolio	March 29, 1996	$40,000.00
Leverling	February 27, 1996	$33,000.00
Peacock	March 30, 1997	$35,000.00
Suyama	September 13, 1997	$30,000.00
King	November 29, 1997	$37,000.00
Callahan	January 30, 1998	$45,000.00

NOTES