Ken Hamady's Unofficial Guide to Learning Crystal Reports® 9-14 Part II:

This set of lessons is designed to expand your ability to use Crystal Reports beyond the basics. The lessons are illustrated by examples taken from the sample database that came with earlier versions of Crystal Reports (Xtreme.MDB). If you are using CR 2008 or any later version this MDB is no longer part of the software download. If you don't have the Xtreme.MDB you can download if from the following link in my blog:

https://kenhamady.com/cru/archives/113 (downloading and connecting to the sample data)
https://kenhamady.com/cru/archives/3093 (connecting to the sample data from CR 2020)

The approach I recommend for self study is to read a section until you get to the next exercise number. Then turn to the back and follow the steps for that exercise. Then read the lesson section a second time. Repeat the lesson if the topic is not clear.

Before you decide to send me questions, you should read my FAQ page:

https://kenhamady.com/faq.shtml

Ken Hamady (540) 338-0194
525K East Market St. – PMB 299
Leesburg, VA 20176
www.kenhamady.com
ken@kenhamady.com

General Topics:

Ken Hamady's Unofficial Guide to Learning Crystal Reports® 9-14 Part II:

Specified Order Grouping:

Sometimes you don't want your groups in alphabetic order. For example, if you have groups named "East", "Central" and "West", you probably wouldn't want these to appear alphabetically. The same with groups called "Children", "Teens", "Adults" and "Seniors". You use Specified Order whenever you want to determine the order of the groups. This feature is also available in cross-tabs and in Advanced Charts to allow you to determine the order of the rows, columns, bars and slices.

To Set a Specified Order for report groups:

1) Use the menu options:
   Report ➡ Group Expert ➡ (Select a Group) ➡ Options Button
2) Change from "ascending" setting to "in specified order".
3) Select the group that should appear first from the drop-down list.
   (If your group is not in the list, you can type it in the box and hit enter.)
4) Select your other groups, in the order that they should appear.
5) Click on the "Others" tab to deal with the groups that you don't specify.

Options for Dealing with "Others":

As soon as you start to specify the order, you will see a new tab in the dialogue marked "others". You use this tab to specify how Groups should be treated if they aren't one of your specified groups. You have three options:

1) Discard them. Only the specified groups will be printed in the report.
2) Include them in one final group called "Others", or any other name you decide to use.
3) Leave them in individual groups, in ascending order, after the specified groups.

Specified Headings:

You can also use specified order to combine several groups under one heading. For example you could group states into "MidAtlantic", "New England", etc. Or you could organize people into age ranges with ages 13 to 19 labeled as "Teen", etc. The steps below will show you how to specify headings this way. However, if the group is changed back to ascending, or to another field, these definitions can be easily lost. This is why I recommend using a formula field to collapse groups, instead of specified headings. The formula won't go away when you change your group. A formula can also be copied to another report. You can always use Specified Order to put the formula values in a non-alpha order.

To Define a new Heading on the "Specified Order" tab:

1) Hit "New" and replace "Untitled" with your desired heading.
2) Use the comparisons to define the groups that should appear under that heading.
3) Click OK when all of the comparisons have been entered for that heading.
4) Repeat steps 1 through 3 until you have defined all of your specified headings.

☞ **Specific Order Grouping (See Exercise 1)**

Advanced Formulas:

Uses of the word IN to create a Boolean value:

There are 4 ways to use the word IN within Crystal formulas. In all cases, the result is a Boolean value. Once a valid Boolean expression has been created, it can be used either in the selection formula, or to control an if-then-else statement in a formula field. The first two examples can work with values of any data type, but the last two are specific to just one data type, strings and dates respectively. (Note that you can use the equal sign "=" instead of the word IN for all of these, except for the third one.)

Note: Literal values in Crystal formula fields are case sensitive.

1) To determine if a field value is "Between" or within a range of consecutive values, use IN with the word TO.

```
{Customer.Postal Code} in "190" to "199"
{Orders.Order Amount} in 5 to 75
{Orders.Order Date} in Date(1999,1,1) to Date(1999,1,31)
```

2) To determine if a field value is "One of" a list of non-consecutive values, use IN with a pair of square brackets. Put your list inside the square brackets, separated by commas.

```
{Customer.Region} in ["PA","NJ","DE"]
{Order Details.Quantity} in [12 , 24, 36]
{Orders.Order Date} in [Date(2002,1,1), Date(2002,7,4)]
```

3) To determine if one text string is contained in another, put IN between the strings.

```
"Bike" in {Customer.Customer Name}
```

4) To determine if a date value is in one of Crystal's preprogrammed date ranges, put IN between the date value and the date range function.

```
{Orders.Order Date} in LastFullMonth
```

☞ **Review of Formula Operators (See Exercise 2)**

StartsWith checks to see if one string starts with another and generates a Boolean value.

```
{Customer.Customer Name} StartsWith "Bike"
```

Like checks to see if a text field matches a specified character pattern string. The pattern can contain one of two wildcard characters. A question mark in the pattern (very rarely used) represents any single character in that position. An asterisk in the pattern represents any number of characters in that position. The Like operator returns True when the field matches the pattern and False when it does not. For example:

`{Customer.Phone} Like "(???)338-0194"` matches number in any area code.
`{Customer.Phone} Like "(???)338*"` matches this exchange in any area code.
`{Customer.Phone} Like "*338*"` matches any number with this 3 digit pattern.

Comment mark Operators (//)
Putting a double slash into a line of a formula will cause Crystal to ignore everything to the right of the slashes in that line. This is handy for putting technical notes or other explanations into complex formulas. It can also be used while testing formulas to "shut off" certain sections of a formula without deleting the text. Comments will turn green in the formula editor.

☞ **Using StartsWith, Like and Comment Marks (See Exercise 3)**

Duplicating Field Objects (v11.5 – v14):
I often need to create a series of similar formulas or a series of similar running totals. A new feature introduced in v11.5 is the ability to right-click on a field in the Field Explorer and create a duplicate of that field. This works for Formula Fields, SQL Expressions and Running Totals but not for Parameters. The only difference between the original field and the duplicate is that the name of the duplicate has a "2" added on the end. If I create another duplicate of the original field then the next duplicate will have a "3" on the end, and so on.

Finding where a field is used:
The Field Explorer indicates if a field is in use by putting a check mark next to the name of the field. However this doesn't tell you WHERE the field is being used. It is often necessary to know where a field is being used, especially if the field has to be updated or removed. Crystal v12-v14 allow you to right click on any field in the Field Explorer and list all of the formulas that reference that field. This works for any field in any category of the Field Explorer. And, it will expose all formulas including formula fields, condition formulas and even the selection formulas. Fields used to Sort, Group, Total or just sitting in a section will not be found by this method.

To accomplish something similar in older versions of Crystal you can temporarily change the data type of the field (i.e., change a numeric formula to just "x"). If the changed field is used in any formula, then that formula will most likely generate an error when you preview. This method can be used with Formulas, Parameters, Running Totals and SQL Expressions.

Managing Null Values:

The concept of null values is somewhat confusing since not all databases use null values, and those that do may not use them in the exact same way. In most modern databases, a null value is the value given a field when it is empty or when its contents are deleted.

Formulas that encounter null values:
Formula fields will not process a null value. If a formula tries to use a field with a null value, then the result of the formula will be null for that one record. This is true, even though other fields in the formula may be filled in. For example, you could write a formula called Full Name to combine First, Middle and Last Names. If a record has no middle name entered, the formula will not print anything for that record, even though there is a First Name and Last Name. One null value can make the entire formula print nothing for that record.

The quick solution is to tell Crystal Reports to fill in all of the null values coming from the database. Under **File ➡ Report Options** there is a check mark that causes Crystal to treat all nulls in the report as if they contained their "default" value. A null string field would default to an empty string (""). A null numeric field would default to zero. A null date would default to Date(0, 0, 0). A null logical field, in systems that allow them, would default to False. I don't use this method often because:

1) The select expert can no longer identify null values.
2) Averages will now treat null numbers as 0 and include them in the average.
3) Crystal may sort the new zeros from null numerics before negative numbers.

In version 11 you can do this for individual formulas without affecting the rest of the report. There is a new selector at the top of every formula that allows you to substitute default values for nulls. This can help solve the 3 problems I just mentioned.

Another way to deal with nulls is to use the **IsNull()** function in each affected formula. IsNull(), NextIsNull() and PreviousIsNull() are the only functions that can process null values. You use IsNull() to check and see if a field is null before the formula tries to process it. The IsNull() function returns True when the field is null. Since IsNull() returns a Boolean, you can use it in an **If-Then-Else** statement. For Example:

```
If IsNull( {Product.Size} ) then "UnSized" else {Product.Size}
```

Nulls in the selection formula:
A selection formula may use a field that has null values in it. If so, the records with null values will be skipped. To locate records with a null value you must use the IsNull() function in your selection formula. The formula below would find all records with a null phone number. It is usually easier to create this expression in the selection formula editor. But, once the expression is created you can add other selection criteria using the select expert.

```
IsNull ( {Customer.Phone} )
```

☞ **Managing null values (See Exercise 4)**

InStr:

The **InStr** function calculates the position of one text string within another. You give the function two string arguments, the first being the field that you are examining and the second being the string that you are looking for. If Crystal finds the string within the field, it returns the position number of that string. If Crystal doesn't find the string it returns a zero.

For a practical example, take a 25-character text field that contains both First and Last Names in the format "Last, First". To extract the only the Last Name portion of each record you would need to know the position of the comma between the names. You could use **InStr** to locate the comma in each entry. The formula would look like:

```
InStr( {Customer.FullName}  ,  ","  )
```

If your entry was "Hamady, Ken" the above formula would return a 7, the position of the comma. You would know that the Last Name ends in position 6, just before the comma. To print just the last name from any entry you could use the following formula:

```
{Customer.FullName} [ 1 to InStr( {Customer.FullName}, ",") -1 ]
```

In this example, the portion of the formula that is underlined calculates the ending position of the substring. This value will be recalculated for each entry. On the record "Hamady, Ken" the underlined expression would calculate a 6, so the formula would extract from 1 to 6.

Length:

This function calculates the number of characters stored in a text string. This is useful when determining how certain fields are entered. For example you could determine:

1) If a zip code was entered with 5 characters or 9 characters.
2) If a telephone number was entered with an area code.
3) If a text based date contains a 2 or 4 digit year.

You give the function one argument, a text string or field, and the function returns a number. This number represents the number of characters stored within that field or string. Empty spaces at the end of the field are not counted, however spaces within the text, and spaces in front of the text are included. The formula below checks the length of the phone number to see if there is an area code stored. If so it extracts just the area code. If not it assumes it is in the local area code of "703" and prints that instead.

```
If  Length({Customer.Phone}) > 7
then Left({Customer.Phone},3)
else "703"
```

☞ **Using InStr and Length (See Exercise 5)**

Val:

The **Val** function is used to calculate the numeric equivalent of numeric characters stored as strings. This allows you to sort character values into true numeric order or sum the results. For example, take a system that stores a Product ID as text values such as: 8, 70, and 600. You might expect that these values would sort in that order. However, text strings are sorted from left to right, regardless of their length. Therefore, these values would sort 600 then 70 then 8, based on their first characters. But, you could write a formula using the **Val** function to calculate the numeric equivalent of this field. Then you could sort the records using the formula field, and they would appear in true numeric order.

To use the Val function you put one string value in the parentheses, and Val converts it into a number. If the string is made up of all numeric characters, then Val converts the entire entry to a number. If the string is a mix of numbers and other characters, Val reads the numbers from the left until it hits an alpha character. Val stops at the Alpha character and converts the numbers that it has found to that point. It will ignore spaces and commas, but a period will be interpreted as a decimal point. A dash within the numbers will change the value from positive to negative. For example, if you had an address that was entered as "125-C Beach Street", the formula below would return a negative number, -125.00 :

```
Val ( {Customer.Address} )
```

You can use the **Replace()** function to replace the dashes with "x" before you take the Val() of the string. The formula below would provide a positive number, 125.00:

```
Val( Replace( {Customer.Address} , "-" , "x" ))
```

☞ **Using Val (See Exercise 6)**

ToText:

The **ToText()** function allows you to convert numeric values, date values and time values into strings. ToText is used primarily when you need to concatenate non-string fields onto a string. The function can have up to 5 arguments, and the arguments do different things for each data type. For the full description of the arguments, look up **ToText** in Crystal's help index. Below are the most common 3 arguments used when converting a numeric value to a string value:

ToText (FieldName , Number of Decimals , Thousands Separator String)

On most PC's the default format is 2 decimals and a comma, so:

`ToText( {FieldName} )`	would look like	123,456.00
`ToText( {FieldName} , 0 , "")`	would look like	123456

☞ **Using ToText (See Exercise 7)**

Conditional Formatting:

There are times when you will want a field or section to have a different format based on the data. For instance you may want to print a negative number in red, or you may want debits to be left aligned in a column instead of right aligned. You may want certain fields or sections not to appear at all under specific conditions. This is done with conditional formatting.

Most of the format attributes have a "condition button", which is a square button labeled "X+2". This button opens a formula editor where you can define when and how that attribute will be activated. You will write one of two types of formulas based on whether the attribute properties are determined by a check box or by a pull down list.

Check box attributes:
For attributes controlled by a check box you use a simple Boolean expression. The expression will be either True or False on each record. On records where the expression is True, the attribute will behave as if it were checked. For example, you could put the following formula on the "drop-shadow" attribute of any field:

```
{Orders.Order Amount} > 8000
```

This would cause a drop shadow to appear, on this field, when records are over $8,000.

Pull-down list attributes, and others:
Attributes that aren't set with a check box require an **If-Then-Else** formula instead of a Boolean. Most of these attributes are controlled by a pull-down list of options, (like colors, alignment or border types). You need an **If-Then-Else** statement for these attributes because you have to specify two different formatting options. For example, if you wanted to have negatives in RED you would need to specify two colors: one if the condition is true, and another if the condition is false. The example below would put negative numbers in red, and positive numbers in black:

```
If {Orders.Order Amount} < 0
then  crRed
else  crBlack
```

When an attribute, like colors, has a pull-down list of options, you will find the list of available options at the top of the function list. You can double click a value from this list and Crystal will insert the appropriate value into your formula. In v11-v14 the available choices will be listed as a comment whenever you create a new condition formula.

In some cases, color attributes have a check box that makes the drop down list visible. These attributes still require an If-Then-Else formula to control the attribute even though there is a check mark. You should skip the check mark on these attributes, and go straight to the formula condition. If you use both, the formula would control what happens in preview, while the check mark would control what happens in design mode.

☞ **Conditional Formatting (See Exercise 8)**

Report Alerts:

An alert is a message that notifies you whenever the report has a record or a total that meets specified criteria. Alerts can be used to flag exceptions, errors or targets that have been met. When an alert is triggered you are given the option of opening a separate "drill-down" window which displays the same report, but only includes the records or groups that triggered the alert.

To add an alert use the menu options **Report ➡ Alerts ➡ Create or Modify Alerts.** You click the "New" button. You must give the Alert a name and a message to display. Then you enter a condition. This condition is any Boolean value. The Boolean condition can be based on any field, formula, subtotal or grand total.

The next time you refresh the report, this condition will be compared to every record. If this condition is true for any record or group in the report, the Alert message window will open, showing you which alerts have been triggered along with their messages. There will also be a button in the window which allows you to view the records or grouped that triggered this alert in a separate preview window. If you view the records Crystal will run a version showing what the report would have looked like if the Alert condition had been put into the selection formula. You close this preview window by clicking the red 'x' in the upper right, just above the ruler.

To change the condition or the message of an alert, you should first close any alert windows. Then use the same menu options above to open the Alerts window. Click the Edit button to make changes to any of the settings. Certain types of changes to the conditions (like changing a condition from a detail field to a subtotal) may generate an error, even though the condition is valid. In these cases you must delete the Alert and add a new one.

☞ **Using Alerts (See Exercise 9)**

InRepeatedGroupHeader:

This function is primarily used to print the word "Continued" when a group spans more than one page. There are two steps to making this work.

1) Tell Crystal to repeat each Group header on following pages.

Report ➡ Group Expert ➡ (Select the Group) ➡ Options
Check off the option "Repeat Group Header on each new page"

2) Write a formula like the following and put it in the Group Header:

```
If InRepeatedGroupHeader
then "Continued"
else ""
```

☞ **Using OnFirstRecord and InRepeatedGroupHeader (See Exercise 10)**

Parameter Fields:

Parameter fields are used to prompt the user for input before the report is run. This allows the user to change many features of the report without having to know anything about designing reports. Parameters can be used to:

Specify selection formula values (like selecting a date range or a department).
Select formatting options (like summary vs. detail).
Enter a label or user name (to identifying the person running the report).

To add a parameter field you go into the Field Explorer and locate the category called Parameter Fields. Right-click on this category and pick "NEW". There are only 2 essential steps. You have to give the parameter a name (like Customer), and you have to select the data type. If you don't select a data type Crystal will default to string.

Optionally you can add some instructions for the user. In version 11 Crystal will default to a message like "Enter Customer" with the second word being the name of the parameter. In older versions, you add “prompting text” to provide a message to the user.

Now the parameter is ready to use. A parameter will not prompt the user unless it is being used somewhere on the report. The parameter could be used in any formula in the report, or it could be a title or comment that is simply displayed at the top or bottom of the page. The most common place that people use parameters is in the select expert. This lets the user control the criteria for the report.

Let’s say you created a parameter for the user to enter a Customer ID number. You could go into the Select Expert (or selection formula) and add a rule that says:

{Field.Customer ID} / is equal to / {?Customer ID Prompt}

The select expert will show you the {?Customer ID Prompt} field when you drop down the sample list of customer IDs. It will be at the top of the sample list, along with any other parameters that have the same data type. Once the select expert rule is in place, the report will prompt you for a Customer ID every time the report is run. Then the report will only return records for that customer.

While the select expert is the most common place to use a parameter, it is not the only place. You can use a parameter field in any formula field or formatting condition - any place where you can use a database field or literal value. Or you can simply drag the parameter onto the report. This allows the user to add a label, title or comment to the report layout at runtime.

Adding Default Values in v11-v14:

You are allowed to give your users a list of values to pick from when they fill in a parameter. The list can be static (stored with the report) or a dynamic (drawn from the database at runtime). Static lists can be collected from a field in the database at design time, imported from a text file or entered in manually. A dynamic parameter list is always drawn from the live data at run time. Dynamic parameters can be based directly on fields in one of the report's tables, or it can draw values from a column in a SQL Command. Dynamic parameters can even be "cascading" parameters where the choice made in one parameter determines the values shown in the next parameter.

Static Parameters:

To create a static value list you set the "List of Values" option to "Static". Then go to the middle window (under the word "Value") and click the in box that says "Click here to add Item". You can type as many values as you want into this column. In the second column you can type an associated description for each value in the first column. To delete any row from the list you click in that row and click the black "X".

If you want to add an item from the live data then go up to the box marked "Value Field" and use the drop down to select the appropriate field from the list. The list will include every field available to the report, even those that aren't on the report. Once you select the field, go back down and click the in box that says "Click here to add Item". Now there will be a drop-down selector to the right that will list the values in the database for that field. You can select any value from the list and it will be added. If you want to automatically add ALL of the values for this field you can click on the button marked "Actions" and select "Append All Database Values".

Lists drawn from the database are not always exhaustive, and may not select the specific items you want for your list. One alternative is to import a list of values (or values and descriptions) from a tab delimited file. You create this file with two columns of data and a tab between each pair. The first column is the value and the second (if needed) is the description. Once this file is created you can import it by clicking on the "Actions" button and selecting Import. Once you select the file, those items will be added to the list.

Dynamic Parameters:
A Dynamic Parameter is one that draws its values from a field in the report each time the report is run. To make a parameter Dynamic you set the "List of Values" option to "Dynamic". Then go to the middle window (under the word "Value") and click the in box that says "Click here to add Item". Locate the field you wish to use and select it. The parameter will display up to 1000 values from this field when the report is run. Note that the field chosen does not have to be used by the report. In fact the table doesn't even have linked to the other tables. You could add a table to the report just to make it available for parameter queries. (Note - if you don't link the table to the other tables, don't use any of its fields on the report except for parameters.)

Once the field is selected for the parameters you can click under the word "Description" and select a second field to serve as a description to the first field. For example, when the parameter's "value" field is Customer ID the parameter's "description" field could be Customer Name. The parameter will then show these values to the user as pairs at runtime. The description field doesn't have to come from the same table as the parameter field, but if they are from separate tables these tables should be linked to each other so the descriptions match the values. These tables should be linked to each other but they do NOT have to be linked to the tables actually used by the report. (Note - if you don't link these tables to the other tables, don't use any of their fields on the report, except for parameters.)

There are a few limitations to note for dynamic parameters:

1) You can't specify a default value for users who skip the parameter.
2) The user cannot type in a custom value. They must select from the list.
3) Dynamic parameters will only display 1000 values (unless you modify the registry).

And unless you use a SQL Command to generate your values list:

4) You can't add custom values to the list, like the word "ALL".
5) You can't filter the values selected by the parameter.
6) Values and descriptions cannot include expressions or calculations.

Using a SQL Command to Generate Dynamic Parameters:
A SQL Command is a SQL Query statement that is stored in a report. You can add a Command to a report, and then draw values from it, just like you would a table. If you are handy with SQL you can get around some of the limitations mentioned above. You can:

1) Filter the values by adding a WHERE clause to the command
2) Use a UNION to add custom values to the list, like the word "ALL".
3) You can use expressions to build more sophisticated descriptions.

To add a Command go into the Database Expert and find your list of tables. Just above the tables you will see the "Add Command" option. Highlight this item and click the arrow as if you are going to move it over. Crystal will open a window where you can enter your SQL statement. Once you have entered your SQL, click OK. The Command will appear on the right like any other table. I recommend that you don't link the Command or use fields from it on the report itself. But you can use it for selecting parameter values. If your Command returns 2 or more fields you can use one for the value and another for the description.

Cascading Parameters:
Sometimes it is easier to locate a value in a long list if you can reduce the number of values of one parameter based on a value in another parameter. For instance you might decide to select an employee by first selecting a department and listing the employees in that department. Or, you might select a customer by first selecting a State, then a City, and finally a customer within that City. These are called cascading parameters.

To create a cascading parameter you create a dynamic parameter (Cascading parameters have to be dynamic parameters). But instead of selecting a single field in the middle window, you select several fields, one below the other, in the order you want them to be prompted. In the customer example above I would select the State field on the first row, then the City and the Customer last. If needed, any of these fields can be paired with a field in the second column which will server as the description for the field in the first column.

When the report is run, Crystal queries the database for the selected fields and creates a temporary table in memory. There is one column in the table for every field used. So, for our example Crystal would create a 3 column table. There would be one row in this table per customer and each row would also contain that customer's City and State. Once Crystal builds the temporary table it will ask you to select one of the states. When you select a state it filters the temporary table to the records in that state and shows you a list of cities. When you select a city it filters the records again and shows you the customers from that city. You can then select the desired customer.

The temporary table can be drawn from a single unlinked table, a set of linked tables in the report or even from a command. All of the lists for a cascading parameter must come from the same source and the source must be in the report even if the source is not used by the report for anything other than generating the list of parameters. Once you define a series of fields for a cascading parameter Crystal will save this as a "Prompt Group" and this set can be reused for other parameters in the same report.

☞ **Parameter fields in v11-v14 (See Exercise 11)**

Adding Default Values in v9/v10:

You are allowed to give your users a list of values to pick from when they fill in a parameter. You can enter values manually or you can draw a list of values from a database field. Note that in these versions of Crystal the list you create is static and won't update automatically. If you want the list of values to be read from the database when the report is run you will have to upgrade to v11-v14, create a Visual Basic type program or use a 3rd party viewer that supports dynamic parameters.

To create your sample list you click the "Set Default Values" button in the parameter properties window. You can type a value on the left and click the top arrow to move it to the list on the right. To load a list of sample values from a field in your database, select a table at the top of the window where it says "browse table". Then select a field next to "browse field". You will see a list of sample values appear in the lower left. You can then highlight any individual value and move it to the list by clicking the top arrow. Or, you can select the entire list by clicking the second (double) arrow. The arrows that point to the left let you remove individual values or all values from the sample list.

Adding or Importing Descriptions:

If your field values are codes, you can add a description to be displayed next to each value. This allows the user to see something meaningful, while selecting the appropriate database code. The list can then be shown to the user either sorted by the code or by the description. If you have a long list of descriptions to enter you can create a text file of codes and their descriptions and import them. This file has to be a tab-delimited text file with two values per row. The first column is the default value, and the second column is the description. Once you create the text file , you can use the "import pick list" button to bring your full list back into the parameter.

Response restrictions in all three versions:
You can allow the user to type in values that don't appear in the sample list or you can restrict the user to the values you have provided. You control this in v9/10 using the "allow editing of sample values" check mark in the main properties window of each parameter. In v11-v14 you look in the bottom window for an option named "Allow Custom Values" and change this to True.

You can also warn the user if their response to a parameter is too long or short (character fields), or not within the correct range (numbers and dates). For instance, if you are prompting for state abbreviations, you may want to put in a length limit of 2 characters. This will warn someone if they start to type in the full name of the state. This feature in v9/v10 is at the bottom of the default values window. In v11-v14 there are 2 options in the bottom window called Min Length and Max Length.

Finally, you can also provide an entry pattern, called an "edit mask", for character fields like phone numbers and Social Security numbers. You do this by entering a "formatting string". The characters you use tell Crystal where numbers and letters should go, and what characters should be added automatically. The following string will require the user to enter 10 numeric values, and will format them as a phone number:

(000) 000-0000

Below are the most common format code characters
A – Required Alphanumeric
a – Optional Alphanumeric
0 – Required numeric
9 – Optional digit or space
L – Required Letter
? – Optional Letter
> – convert following characters to uppercase
< – convert following characters to lowercase

The online help has a complete list of the format code characters and what they mean. This feature in v9/v10 is also at the bottom of the default values window. In v11-v14 find the option in the bottom window called Edit Mask.

Note that in v9/v10 the mask is enforced on the user while they enter their value, which prevents them from putting in an incorrect value. For some reason in v11-v14 the mask is not enforced but is simply checked after the user enters his value. They have to reenter it until they match the mask.

☞ **Parameter fields in v9/v10 (See Exercise 12)**

Range and Multiple Parameter Field:

When you add a parameter field, you are allowed to have the user select more than one value in one parameter. You can create a "Range" parameter that asks the user for a low and high value like a date range. Or, you can create a "Multiple Value" parameter that asks the user to enter a list of individual values.

Range Parameters (Between):
Making a range parameter is pretty simple. In v11-v14 you go to the bottom window of options and find the one marked "allow range values. In v9/v10 there is radio button on the left that says "Range Values". After you change it to a range you can still use it in the select expert with an "equal to" comparison, or with an equal sign in any Boolean formula. Crystal will check to see if the value is between or equal to the values entered by the user. In other words, "Between" also includes an exact match of the high and low values entered by the user.

Range parameters add a couple of special challenges. For instance, when you place a range parameter field onto the report, it will display a blank value. To display the values selected by the user you need to write a formula like one of these:

Dates Ranges:

```
"From: " &   Minimum ( {?Date Range} )   &
" To: "    &   Maximum ( {?Date Range} )
```

Numeric Ranges:

```
"From: " & ToText ( Minimum ( {?Number Range}), 0 , "" )  &
" To: "    &  ToText ( Maximum ( {?Number Range}), 0 , "" )
```

String Ranges:

```
"From: " &  Minimum ( {?String Range} )   &
" To: "    &  Maximum ( {?String Range} )
```

Another common problem specific to date ranges is that most databases use DateTime fields even when times are not needed or being used. But, most people don't want to be prompted for times when they enter date ranges. It is possible to use a "Date" parameter to select a date range from a DateTime field if you keep the following two things in mind.

1) Since the data types don't match exactly the select expert will not show a Date parameter in the drop down for a DateTime field. To add the criteria you will have to go into the selection formula and enter something like:

```
{Orders.Order Date} =  {?Date Range}
```

2) If you do this you should always check the SQL statement generated to make sure that you don't lose the last day of the date range. The WHERE clause for the ending date should say less than midnight on the day *after* the end of your range. This is usually done correctly but I have seen some ODBC drivers get this wrong, which means your report will be missing the last day of the selected range.

Multiple Parameters (One of):
If you select "Allow multiple values" the user will be prompted for a list of individual values. They can make the list as long as they wish. The list will be stored as an array, which you can use in the select expert with "equal to", or in any formula with an equal sign. Crystal will check to see if the field value matches any item in the list.

Placing a "Multiple Values" parameter field on the report layout will only display the first value in the list. To display all of the selected values you need to write a formula. If your parameter has multiple string values (the most common) then you can use a formula like this one:

```
Join ( {?YourCharPrompt} ,  ', ' )
```

If your multiple value parameter is not a string, see formula page #7 on my web site for examples.

☞ **Range and Multiple Parameters (for v11-v14 see Ex. 11b, for v9/v10 see Ex. 12b)**

Selecting Records with a Partial Match:
String parameters that are used in the select expert with "Equal To" have to make an exact match. If you use Startswith or Like as a comparison this allows the user to enter only part of the desired value. For instance you could create a parameter for Zip Codes using:

{Table.ZipCode} / Startswith / {?ChosenZip}

Entering 19045 would return only that zip code.
Entering 190 would return all zip codes starting with 190.
Entering 1 would return all Zip Codes starting with 1.

Or if you want to allow the user to enter wildcards into the parameter you could use the "Is Like" comparison. That way the user could enter a value like *Insur* to select all vendors that have the word "Insurance" in their name. When you use "Is Like" the user can simply enter a single asterisk as their parameter entry to mean "ALL". Since the asterisk is a wild card character, any record that has a value will be 'like' an asterisk. The only records this approach won't include will be Null values, which are not 'like' anything.

Selecting "All" records including Null values:
If you want to have an "All" option, you will probably want to use an If-Then expression in your selection formula. This is especially true if your "All" should include null values as well. Taking the Zip Code example above, you could use the following selection formula to allow users to select "All" and get all records, including those with null Zip Codes:

```
( If  {?ChosenZip} = "ALL"  then True
else {ZipCode} Startswith {?ChosenZip} )
```

When the user selects "All", the selection formula is "True" for all records. It has no reason to read the zip code field in the else clause, so it never encounters the null values. It will therefore include all records in the report. If you are using this rule with other rules, you should surround it with a pair of parentheses, as shown. Otherwise the next rule may be evaluated as part of the else condition, rather than as an independent condition.

In some SQL environments the following version does the same thing but process more efficiently because it passes the second line to the SQL statement when it is needed:

```
( not ( IsNull ({ZipCode} ) ) and
      {ZipCode} Startswith {?ChosenZip} )
or "ALL" in {?ChosenZip}
```

☞ **Partial Match Parameters (for v11-v14 see Ex. 11b, for v9/v10 see Ex. 12b)**

Controlling formatting conditions with parameters:
You can also control the format of the report using parameter prompts. For instance, you give the user a choice to print the report in detail or in summary format. To do this you would create a parameter field called "Summary Choice" with static default values of "Summary" and "Detail". You would then format the Details Section and Group Header of the report to suppress on the following condition:

```
{?SummaryFlag} =  "Summary"
```

When the user selects "Summary", the report will suppress any section, field or text that has this condition, creating the summary version of the report that you designed.

Highlighting target values:
You can also have the user select a value to highlight in the report. For instance, you want orders over a certain amount to print with a gray background on the detail band. You would add a numeric parameter field called "Target", and put in a default value of 9999999. You could then go into **Report ➡ Section Expert** and add a condition formula to the background color of the details section, like the formula below. Records with an amount over the user's target will get a background color.

```
If {Orders.OrderAmount} > {?Target}
Then Gray
Else NoColor
```

Controlling the Sort from Parameter Prompt:
You can even give your user control of the sort and group fields from a parameter prompt. For example you could create a text parameter field called "Sort Choice" with three defaults: "Ship Via", "Amount" and "Date". Then you could use this parameter field inside a formula field called "Sort Field". The formula could be something like:

```
If {?Sort Choice} = "Ship Via"
then {Orders.Ship Via}
else  If {?Sort Choice} = "Amount"
      then Totext({Orders.Order Amount},'#######.00')
      else Totext({Orders.Order Date},'yyyy-MM-dd')
```

You could then use this as formula for sorting or grouping.

There is a reason why Amount and Date are inside the Totext() function. In an If-Then-Else formula, the value after the word "then" has to have the same data type as the value after the word "else". When you mix fields of different types, like we have in this example, you must convert all of the values to text. The ToText function converts date and numeric values to their text equivalents. I also added format strings to make sure that the converted values would still sort in the correct order. This is a good example of converting numeric or date fields into text expressions that will still sort appropriately. For more info on ToText formatting, see the Crystal Help index under the word ToText.

☞ Parameter fields (For v11-v14 see Exercise 11c, for v9/v10 see Ex. 12c)

Interactive (Editable) Parameters (v12-v14):
Before version 12, the only way to change the value of a parameter was to refresh the report. This isn't a problem in smaller databases, but can be a real burden in a large database. It is especially frustrating when the parameter is simply a formatting parameter that has nothing at all to do with selecting the records. But starting in version 12 you are allowed to make a parameter "editable" or interactive. Your users can now use the new "Preview Panel" on the left to change the value of a parameter back and forth, without having to refresh the report.

If you edit a parameter in v12-v14 you will see that the first property in the bottom section is called "Show on (Viewer) panel". There are three choices for this property:

Do Not Show – the parameter is not listed on the panel
Read Only – the parameter is listed but can't be changed
Editable – the parameter is listed and its value can be changed.

If you have an editable parameter you can change the value in preview with 5 clicks:

Click on the parameters section of the Preview Panel to list the parameters.
Click on the parameter you want to modify to open it.
Click on the value you want to Modify, then click on it again to change it into a selector.
Select or enter the new value.
Click the check mark in the tool bar of the Preview Panel to apply the value.

Interactive Sort Controls (v12-v14):
You can also allow the user to interactively change the sorting or grouping order of the report by adding a "Sort Control". A Sort Control is simply a text object that has been "bound" or linked to one of the sort or group levels of the report. Once bound a pair of up/down arrows will appear next to the text. When you click the up arrow the sort changes to ascending and the down arrow changes to descending. If you have multiple sort fields and you create controls for two or more of them, then clicking the up arrow not only makes that sort ascending but it also makes it the primary sort – rearranging the fields in the Record Sort Expert. The position change does not happen with groups. Groups can only have their order changed from Ascending to Descending.

There are two ways to add a sort control. If the text object already exists, all you need to do is 'bind' it to the appropriate field. You right-click on the text (like a column heading) and select "Bind Sort Control". Then you select the appropriate group or sort field and click OK.

If the text doesn't exist you can use Insert > Sort Control to open the "bind" window. Then you select the group or sort field and click OK. You then draw a frame on the report for your new text and type the text you want to appear in the control. Click outside the text to create the control.

One note. If you have multiple sort controls Crystal will only remember the last arrow clicked. There is no way to specify a primary and secondary sort using these controls. If the user needs to specify multiple group or sort levels in preview you would need to use the method for sorting based on a parameter mentioned on the previous page, and make the parameters editable as described above.

Subreports:

Crystal allows you to insert one report into another and then run them both together. The inserted report is called the "subreport" and the other report is called the "container". Any report can be used as a subreport or as a container. For example, you could take a report showing this year's salary expenses from payroll, and put it into a container report showing revenue figures from the accounting system. In this way, subreports allow you to combine unrelated data sets into one report.

The subreport and container each have their own set of linked tables, their own selection formula, their own grouping and sorting. Each goes through the database to select the required records. I have seen users add subreports when a cross-tab would work better. If the main report already has the records that you need, it is usually faster to use cross-tabs to summarize these records in a second or third way. Subreports are needed when you need to use records that are not in the main report.

The subreports are generated when the main report begins to layout the actual pages. When the container starts to layout a section that contains a subreport, the container generates that subreport, lays out that section, and then continues with the next section. If you place a subreport in a section that repeats, like a Group Header, the subreport will be repeated each time that section prints. Unlike cross-tabs, placing the subreport in a Group Header or footer does not automatically make the subreport specific to the group. Subreports have to be linked if you want them related to the group. Linked subreports are discussed below.

Subreports are needed when:

A) You want to show data from unrelated tables or databases on one piece of paper.
B) Linking two tables creates a many-to-many relationship.
C) Linking fields are not identical. Formula fields are not available for visual linking.
D) You need to include the same detail records in more than one place on a report.
E) You need an outer join, but also need criteria on the outer table records.

Unlinked Subreports:

Unlinked subreports are used when you want to show data from unrelated tables or databases on one piece of paper. These subreports are usually only shown once, so they are placed in the Report Header or the Report Footer. If you put an unlinked subreport into a repeating section, an unlinked subreport will print the same results each time that section occurs.

Steps to add an unlinked Subreport:

1) From the menu select **Insert ➡ Subreport**.
2) Select one of the following options:
   Select a report to import from your existing .RPT files or
   Create a new subreport 'on the fly' using the subreport expert.
3) Click OK when you have either selected or completed your subreport.
4) Click in the section where the subreport should go (Report Header or Footer).
5) Preview the report, and the subreport will appear with that section.

Linked Subreports:

A subreport may contain a parameter field. If left unlinked, the subreport parameters will appear with the container report parameters. The values you enter are passed to the subreport. Alternately, Crystal allows you to link the subreport parameters to a field in the container. When linked, the subreport parameter will not prompt you. Instead, it will take the current value of the linked field, and use this value to fill in the subreport parameter.

You can link subreport parameter fields to any field in the main report. This includes database fields, formula fields, totals or even parameter fields in the container report. But, the field you are linking to has to have the same data type. For instance, you could link the start date parameter in the subreport to a start date parameter in the container. This way you only have to enter the date once, and it fills in both parameter fields.

Recurring Subreports:

Linked subreports can generate a separate subreport for each group in the container. For instance, if your container has details that are grouped by customer, you might need a subreport at the end of each customer that is specific to that customer. So, you can create a parameter in the subreport for the Customer ID. You then add a rule in the select expert of the subreport that says **Customer ID / is equal to / {?parameter}**. This causes the subreport to only print for one Customer at a time, based on the parameter. You could then place this subreport in the Group Footer of the container and link the subreport parameter to the Customer ID of the container. Each time the container prints a Group Footer, the subreport will request a Customer ID from the container, and uses that Customer ID to fill the subreport parameter. The subreport will only include records that match that Customer ID.

Steps to add a linked Subreport:

Create a subreport that uses a parameter field.
Follow the first 2 steps above for adding an unlinked subreport.
Before you click "OK", select the links tab at the top of the subreport window.
Select the linking field from the container and click "**>**" to move it to the right.
In the lower left select the subreport parameter field to link to.
Click "OK and place the subreport in the correct section of the container.

Letting Crystal Add a Link:

Optionally, if you are creating a simple select parameter, you can let Crystal add a parameter to the subreport for you. When you are in the "Link" window and move over your link field, Crystal will suggest a parameter name using the name of the link field with "?PM-" in front of it. Crystal will add this parameter to the subreport unless you select another parameter.

If you are letting Crystal add the parameter, and you leave the check mark next to *"Select data in subreport..."*, Crystal will also add a new rule in the selection formula of the subreport. This new rule will say that a selected field in the subreport has to be "Equal to" the value of the new parameter field. Crystal will let you pick the subreport field to use, showing you all of the fields with the appropriate data type. Personally, I find it clearer to create my own parameters in the subreport, give them meaningful names and determine how they are used, rather then letting Crystal create them for me.

Editing a Subreport:
Subreports are often imported from existing RPT files, but there is no connection that remains between the subreport and the RPT file. The subreport is a copy of the original RPT settings and these settings are now part of the container's RPT file. Subsequent changes to the source RPT do not affect the subreport (unless you use the "Reimport" option mentioned below).

To edit a subreport click on the subreport's design tab (if visible) or double click on the subreport object in the main report's design screen. This will open the subreport's design environment and almost the entire menu now refers to the subreport. To edit a subreport in preview mode, preview the main report and double click on the subreport in preview. This will open a preview window for the subreport.

Adding/Editing a Subreport Link:
To change the link of a subreport select the subreport object on the design tab and use the menu options **Edit ➡ Subreport Links** to reopen the links window.

On-Demand Subreports:
Subreports can be set to run when needed, or "on demand". The user clicks on a subreport to launch it in a separate preview window, like a drill-down. It will no longer appear with the main report. Like drill-down, you can only print one "on-demand" subreport at a time. To use this feature, select the subreport and use the menu options **Format ➡ Subreport ➡ (Subreport Tab).** Check the option "On-Demand Subreport".

On-Demand subreports have 2 caption formulas strings that you can write. You write them in the formula editor using the "X+2" buttons. Both of these caption formulas can incorporate fields, formulas or totals (if converted to text). The first caption is the "On-Demand" caption which will print as the subreport object's label. So you could click on a label that included the customers name and total to launch that customer's subreport. The second caption will appear on the top tab of each drill-down window. This way, if you drill-down on several subreports, the tabs will be specific to each.

Notes
1) In versions 9 - 11, subreports will have the same page orientation as the container. If the container is in portrait, the subreport must also be in portrait. This can be changed in v12-v14 where you are allowed to have any section change orientation and then revert back once the section is complete. A change in orientation will automatically add a page break since a single page can't print in both portrait and landscape.

2) Subreports that are multiple pages will have the Page Header and Page Footer of the main report. The Page Header and Page Footer of the subreport will be converted into a second Report Header and a second Report Footer. They will only appear once at the beginning and once at the end of the subreport.

☞ **Subreports (See Exercises 13)**

Cross-tabs:

The table below is a simple example of a cross-tab:

	Jones	Smith	Thomas	Willis	Total
DC	2,500	2,400	2,200	1,800	8,900
DE	1,500	1,250	0	2,000	4,750
NJ	4,500	0	3,000	3,500	11,000
NY	5,250	0	4,000	5,000	14,250
PA	4,000	4,200	3,900	3,700	15,800
Total	17,750	7,850	13,100	16,000	54,700

A cross-tab is summary table that you add to a report to recap the data. The most common cross-tab uses two fields that define the rows and columns of a table. The cross-tab automatically adjusts the number of rows and columns in the table based on the number of different values found for each field. It then puts a summary in each cell of the table.

The example above is sales by state and by employee. The cross-tab found 4 different employees and 5 different states in the report's dataset. It then created a row for each state and a column for each employee. Last, it added up all of the sales for each state, subdivided each state by employee, and printed the subtotals in the corresponding cells. If the report was run again, and the new results included one more employee, then another column would be created automatically. Note that v11-v14 puts the totals on top and to the left by default.

Steps to create a cross-tab:

1) Switch to Design mode (this makes it easier to see the sections).
2) Use the menu options **Insert ➡ Cross-tab** to open the Cross-tab Expert.

(Note that in v11-v14 you must place the cross-tab in a report section immediately. Right-click in the upper left corner of the cross-tab to get to the Cross-tab Expert.)

3) Select the row field from the field list and drag it into "Row Fields" box.
4) Select the column field from the field list and drag it into "Column Fields" box.
5) Select the field(s) to subtotal (like sales) and drag it into the "Summarize Fields" box.
6) If needed, use "Change Summary" to change the default summary operation used.
7) Click OK
8) Click in the Report Header, or other appropriate section of the report (see below).

Placement of a Cross-tab:

A cross-tab can be placed in four out of the seven report sections, and will behave differently based on where it is placed. A cross-tab placed in the Report Header or Report Footer will appear only once in the report, and will summarize all of the records in the dataset. A cross-tab placed in a Group Header or Group Footer will appear multiple times, once for every group. Each group will have its own cross-tab that summarizes the records in that group.

Making changes to a Cross-tab:

If you need to make changes to the cross-tab settings, select the cross-tab by clicking in the upper left (empty) corner. Then, use the menu options **Format ➡ Cross-tab Expert** to reopen the Cross-tab Expert.

Cell Width:
You can adjust the width of the cells in design or preview. When you click inside any cell, you will get sizing handles that allow you to change the width of that column. However, changing the width of one column may affect several columns, since interior columns are the same column repeated for every value found.

Specified Order Grouping:
Both the row and column fields have a "Group Options" button in the Cross-tab Expert. These allow you to set the sort order to Ascending, Descending or Specified Order. Specified Order works in cross-tabs just like it does with groups. It allows you to put your groups in non-alphabetical order.

Group Sorting (TopN) within Cross-tabs:
You can set the cross-tab to only show the highest or lowest rows, like the Top 10 rows, by using the Group Sort feature. First click in the upper left corner of the cross-tab, and then use the menu options **Report ➡ Group Sort Expert**. The window you see doesn't say it, but you are now setting the Group Sort properties of the cross-tab, not the report. Unfortunately, you can only use this feature with the rows of the cross-tab, not the columns.

Formulas in Cross-tabs:
You can use most formula fields as row fields, column fields or summarized fields in a cross-tab. Any formula that can be summarized on the main report can also be used in a cross-tab. Formulas that can't be used are those that use summary fields, shared variables, Previous, Next or that start with WhilePrintingRecords.

Manual Cross-tabs:
There are two limitations of cross-tabs that people frequently encounter.
1) There is no way to add a calculated cell to a cross-tab, using other cell values.
2) There is no way to drill-down on a cross-tab row or cell.

These can be done, in a limited way, using a 'manual' cross-tab, which is a summary report constructed to look like a cross-tab. The rows of a manual cross-tab are the Group Footers of the report. The Group Header and details are hidden. Each column is made by writing an If-Then-Else formula and creating a subtotal of that formula. One column's formula might be:

```
If  {Orders.Ship Via} = "FedEx"
then {Orders.Order Amount}  else 0
```

This column will print only the FedEx amounts, so a subtotal of this formula will be a FedEx subtotal. If you hide the details, you will see a column of subtotals that look like a cross-tab column. You have to create a separate formula and corresponding subtotal for each column in your manual cross-tab. The advantage is that you can use these subtotals in formulas on the Group Footer. You can also drill-down on them to the details. The disadvantage is that the number of the columns is now hard-coded. New columns not automatically appear like they do in a true cross-tab. The Formulas page on my web site describes this in more detail.

☞ **Cross-tabs (See Exercise 14)**

Adding Calculated Rows and Columns to Cross-tabs (v12-v14):
Crystal v12-v14 allow you to insert individual rows or columns anywhere in a Cross-tab. These allow you to do calculations based on the other rows or columns in the cross-tab. The calculations can refer to rows or columns that have a specific row or column name (like Sales) or to columns that have a relative position to the calculation (like 2 columns to the left).

Using Specific Named Columns:
Say you have two rows labeled "Gross Sales" and "Cost of Goods Sold". You could insert a new to calculate the difference between those first two rows, and label it "Net Sales". If the calculation specifies rows by name then those rows are used regardless of their actual position from one month to the next.

The first step is to identify the two rows (or columns) to be used in the calculation. Select one of the rows to be first in the calculation. The new calculated row will be placed below whichever row is second in the calculation. Then the steps are:

1) Right-click on the first row you want to use in your calculation
2) Select "Calculated Member" > "Select <field name> as first value".
3) Click OK on the explanation message.
4) Right-click on the second row you want to use in your calculation
5) Select "Calculated Member" and select the correct calculation from the list of options.

Crystal will add the new row and give the row a default label (which we typically change).

Modifying a calculated row.
To change the label for the new row you right-click on the new row label and select "Calculated Member" > "Edit Header Formula". You can enter any literal text in this formula.

To change the position of the row you right-click on the new row label and select "Calculated Member" > "Edit Insertion Formula". Here you use grid functions to define when the row should appear. The following example prints a new row below the row labeled "UPS"

```
GridRowColumnValue("Orders.Ship Via") = "UPS"
```

The calculation starts out as zero. To change the calculation you right-click on the calculated value and select "Calculated Member" > "Edit Calculation Formula".

Using relative positions in calculations:
Say you have monthly columns in your cross-tab. You might want a "variance" column after every month comparing the current month to the prior month. Rather than hard coding specific month values in each calculation, you could refer to each month's position relative to the position of the variance column. So the current month could be the variance column minus 1, while the previous month might be the variance column minus 3 (assuming each month has a variance. To refer to the value of the column to the left you could use a calculation like this:

```
GridValueAt (CurrentRowIndex, CurrentColumnIndex -1, CurrentSummaryIndex )
```

☞ **Cross-tabs (See Exercise 14b)**

Split Sections:

You split a section into subsections when need to format parts of a section differently, or when you need to eliminate overlapping objects. Here are three very common scenarios where subsections are used:

1) Objects that can grow. When one cross-tab is placed above another in the same section, the first may grow over top of the second. The same thing happens if you have subreports or memo fields that "Can Grow". But, you can put each objects or field into separate subsections, and this prevents them from overlapping.

2) Hiding blank lines. You might want to suppress the second address line in an address label, when it is blank. Split sections are often used with the "suppress blank section" property. By giving each line its own separate subsection, you can eliminate space left by empty lines.

3) Choosing between 2 different versions of a section. For example, you could use a parameter to allow the user to choose between 2 different charts, one stored in A and one stored in B. Or your debit records could print on a different detail band than credit records. You could create 2 different versions of the detail band (A and B) and suppress them based on opposite conditions that read the transaction type.

Inserting a subsection:
You split a section from within the Section Expert **(Report ➡ Section Expert)**. Highlight the section you want to split and click "Insert" at the top of the Section Expert window. The highlighted section splits into subsections "A" and "B". Anything contained in the original section will stay within "A". Each time you click "Insert" you create a new subsection, which is placed below the section you have highlighted. If there are already subsections below the highlighted section, these will be moved down and renamed.

Deleting a subsection:
To delete an entire subsection, and all the objects within it, you highlight it in the section expert and select "Delete" at the top. All objects in the section will disappear and the subsections that come after it will be renamed.

Merging two subsections:
To merge two subsections into one you highlight the *upper* section of the two in the section expert and click the merge button. The section just below the highlighted section will be added to the highlighted section. Subsections that come after these will be renamed.

Reorganizing the subsections:
To move a subsection up or down within its section, highlight it and use the arrow buttons. The letter names of each subsection will change to reflect their new position. You can also rearrange the sections in design mode by dragging the gray area along the left up or down to the desired position.

☞ **Splitting Sections (See Exercise 15)**

Charts:

Crystal Reports allows you to add charts (pie, bar, etc.) to your reports. This allows you to visually illustrate your data. In some ways, charts work like cross-tabs. Charts can be placed in the same four sections: Report Header, Report Footer, Group Header, and Group Footer. Charts placed in the Report Header or Footer will print once and will analyze all of the records in the report. A chart placed in a Group Header or Group Footer will print once per group and will reflect all records in that group. Unlike cross-tabs, you can specify the section while in the Chart Expert, as well as by moving the object.

Inserting a Group Chart:
Group Charts are the most common charts. They get their name because they rely on the existing groups and subtotals in the report. Because they use the existing groups, they allow you to drill down from a bar or slice, directly to the data. To add a group chart, use the menu options **Insert ➡ Chart** to open the Chart Expert. (Note that in v11 you must then place the chart in a report section before you can open the Chart Expert. Right-click on the chart to get to the Chart Expert.)

On the "Type" tab select the type and subtype of chart you would like to see.

On the "Data" tab specify the following settings:

1) The left side will default to "Group" if you have any subtotals in your report.

2) In v9/10 you use [Place Chart] to specify the section for the chart. For example, "Once per report" and "Header" would mean the chart goes into the Report Header.

3) In the middle use [On Change Of] to determine which group is used for the slices or bars. Groups will only be listed here if they have subtotals available for charting. If you only have one chart, you won't be able to change the setting.

4) At the bottom use [Show] to specify the subtotal to use for the bars or slices. If there is only one subtotal you won't be able to change the setting.

When you click OK the chart will appear in the section that you specified. Double click on a bar or slice and Crystal will drill-down to that group of details.

Optional Tabs: (if you take out the check mark on the Data tab)

Axis - Used with bar and line charts to set the line spacing and the scale of each axis.
Options - Controls the legend, the chart labels and the size of chart components.
Text - Allows you to enter titles, subtitles, footnotes and format the text of each.

Making changes to a Chart:
You can change the above settings by reopening the Chart Expert. First click in the background of the chart, and then use the menu commands **Format ➡ Chart Expert**. You can also launch the Chart Expert by right clicking in the background of the chart and selecting Chart Expert from the short cut menu.

Advanced Charts:
Advanced charts do not rely on existing groups or subtotals in your reports. Like cross-tabs, they create their own groups and subtotals. The downside is that you lose the ability to drill down to the details of a bar or slice. You start advanced charts the same way, but on the data tab you click "Advanced" instead of "Group". If you don't have any subtotals, this will be your only choice. This changes the data window and allows you to select any report field for both "On Change Of" and "Show Values". By using fields that aren't already group fields and subtotals, you can have the chart analyze the report in a different way.

Advanced charts also allow you to compare multiple grand totals. All Group charts represent a series of subtotals from one column of the report. But there may be times when each slice or bar in the chart should be taken from a different grand total on the Report Footer. For instance, let's say you have 3 columns on the report: quantity ordered, quantity shipped and quantity returned. If you have totals of these three fields, you might want to show them as three bars in a bar chart, or three slices in a pie. This can be done using an advanced chart.

For three grand totals use the following settings on the Data tab:

1. Click "Advanced" on the left.
2. [Place Chart] should be "Once Per Report"
   (in v11, put the chart in the Report Header)
3. [On Change Of] does not get a field, but is set to "for all records"
4. [Show Value] would include all three quantity fields:
   quantity ordered, quantity shipped and quantity returned.

You could even do this for each group in the report, using 3 subtotals. If you were grouped by region, the settings on the Data tab would be:

1. Click "Advanced" on the left.
2. [Place Chart] should be "Once Per Region"
   (in v11, put the chart in the Group Header)
3. [On Change Of] should the field Region
4. [Show Value] would include the three fields:
   quantity ordered, quantity shipped and quantity returned.

More Chart Formatting Options:
For more chart formatting options you, click anywhere on the chart to expose the "Chart" menu column. If you want to format a specific item in the chart, click on that item. Then you use the following selections either listed in the Chart menu or under Chart Options:

General/Chart Options – Adjust most of the general features of the chart
Series – Adjust data labels and trend lines.
Selected Item – Change fonts, colors and borders of a specific item.
Grid - sets the Grid size for Bar Charts or 3D surface charts.
Templates – allow you to save the settings of a chart and reuse them later.

☞ **Adding Charts to a Report (See Exercise 16)**

Keep Together:

When a large object starts at the bottom of a page, it can either split across a page break, or it can leave white space at the bottom of the page and start on the following page. You control this behavior with the "Keep together" property which is available in three different places:

1) **Objects:** Field objects, text objects, charts and subreports all have a "Keep Together" option. You will find this option on the common tab of the objects "Format" window. The option is usually set 'on' by default. If you leave this property checked, object that can't fit will move to the following page, leaving white space on the current page. Taking out the checkmark will allow the object to be split across pages, when needed.

2) **Sections:** If you have a deep section, with room for several lines, you can keep the section intact, or allow it to split across page breaks. The section property is found in the Section Expert under **Report ➡Section Expert.** If a section has been split into subsections, you can set this property for individual subsections or for the entire section. To keep an entire section together, open the section expert, highlight the section heading above the subsections, and check off the "keep together" for this section. Keeping a section together is often confused with the next item, keeping a group together.

3) **Groups:** On occasion, you may want the Group Header, the Group Footer and all the details in between to print together on the same page. This property is not found in the section expert. Use the menu options **Report ➡ Group Expert ➡ Options(button) ➡Options (Tab).** The "Keep Group Together" option is at the bottom of the window. If a group is set to keep together, and it won't fit at the bottom of the page, it will leave white space and start on the next page.

☞ **Using Keep Together (See Exercise 17)**

Underlay:

To underlay means to have one section superimposed on following sections. Users often arrange the objects in each section so that they display side-by-side. This allows you to:

1) Print a list of details beside the Group Header information.
2) Print subtotals beside the bar chart that represents them.
3) Place an image on the report and have it appear under the data as a watermark.

Normally a chart in the Group Header would print above the details. With Underlay, you could have the chart print beside the details. You would put the chart on the left side of the Group Header and put all of the detail fields on the right side of the details band (so they are not directly under the chart). Then use the menu options **Report ➡ Section Expert,** highlight Group Header 1, and check the property that says "Underlay following section". The details will seem to move up, because they now start at the same place as the Group Header, rather than following it. To do a watermark, you put the image in a large Page Header, and then set the Page Header to underlay the other sections.

☞ **Underlay Sections (See Exercise 18)**

Mapping Data:

Crystal has partnered with MapInfo to allow you to illustrate your summary data with maps. Crystal converts geographic values (cities, states, provinces, countries, etc) into map areas and can then shade those areas based on subtotals. Crystal Reports comes with 33 different maps including World Countries, Europe, Asia, Australia, USA, Mexico, Canada, and Japan. Other maps are available from MapInfo.

The key is that your stored geographic values have to match the entries in the MapInfo database. If they don't, the value is considered a mismatch, and has to be manually mapped to an existing entry in the MapInfo tables.

Inserting a Map:
Use the menu options **Insert ➡ Map.** The Map Expert Data tab options are virtually identical to the options on the Data tab in the Chart Expert. Crystal will allow a "Group" Map if it recognizes your group field as geographic. However, I have only been able to use "Advanced" maps.

On the "Type" tab select the type of chart to use. The "ranged" option gives the most meaningful results for simple charts. "Dots" and "Graduated" are handy if you have to use a black and white hard copy for your map. To use the bar or pie chart, you have to have groups and subtotals within each geographic group, since this option is designed to create a separate pie or bar chart for each geographic area.

Once a map has been previewed, you can right-click on it or use the map menu to zoom in, zoom out or pan the map in any direction. You can make changes to the map by selecting it and using the menu options **Format ➡ Map Expert**, or by right-clicking the map to select map expert.

Resolving Mismatches:
The most useful menu option is **Map ➡ Resolve Mismatches.** This menu option lists all of the available maps that you can choose from, and allows you to choose a different map. This is important because the chart expert doesn't ask you to select a map. It must take a guess based on the values it finds.

This menu option will also read through the values of your "on change of" field and compare them to the selected map's database. It will list all of the items that it couldn't identify and you are allowed to match them up to the existing MapInfo values. You can match and unmatch them as many times as necessary.

☞ **Inserting a Map (See Exercise 19)**

Exporting:

Exporting allows you transfer the output of a report to other formats. I put these formats into 3 categories. Document exports maintain the layout of the report. Data exports are used to transfer the dataset to another application. The Definition export shows the internal settings of the report.

Not all export formats support all of report features. The file **cr10_exportlimitations.zip,** available from my LINKS page, lists the features supported in each format. I haven't seen one for v11-v14 yet. You should also download the latest export DLLs before you give up on a feature. These DLLs are updated regularly. And, it is always wise to test an export and see how well the features of the report are exported, before you commit to using a specific format.

Document Exports:
This category includes formats such as PDF, Excel, text, etc. When you export to these formats, Crystal Reports will create a document that is very close to the way the report would print. It will include most headers, footers and text attributes. Most will even include images.

Even if your report will never be printed on paper, Crystal will want to format it for your default printer. This limits exports to the width of your printer. In v12-v14 there is a new page option to "dissociate" or disconnect from the printer and put in your own virtual page size. In older versions you can use one of the free PDF virtual printers, which also allow custom page sizes.

Data Exports:
This category includes formats such as Comma Separated, Character Separated, Tab Delimited and ODBC. When you export to one of these formats you are exporting the raw data, without format attributes. Your export will include one row for every row of your report output. Typically you will only show one section, the Details or the Group Footer, for these exports.

Definitions Exports:
Crystal allows you to export a text file that documents many of the settings in the RPT file. This option is called the Report Definition. Unfortunately, the document that is created is incomplete and the format is not very readable. If you would like to document the settings in your RPT files, there are several third party tools that you can buy. These do a much better job, but they are not free. I have a list of these on the "Links" page of my web site.

How to export your report:
1) Select the menu options **File ➡ Export**
2) Select the format of the export (upper box).
3) Select a destination:
    Disk File - saves the file in the selected format on the disk.
    Application - sends the file to the application for viewing (with an option to save).
    MAPI - prepares the disk file as an attachment to an Email message (if configured).
    Exchange File - sends output file to an Exchange Folder
    Lotus Notes/Domino - send the output file to the Notes database.
4) Answer any option questions that pop up, and then click OK.

☞ **Exporting (See Exercise 20)**

Automatic Running Totals:

Running totals are used to provide:

A) A running balance that changes on every record, like a bank statement.
B) Accurate grand totals in reports that have:
    1. TopN and no "others"
    2. Group selection (selecting groups based on a subtotal)
    3. Duplicate values caused by a one-to-many join.

You add running totals in the Field Explorer. Highlight "Running Totals" and click the button marked "New". The following steps will create a basic running total:

1) Give the running total a meaningful name, or leave the default name.
2) Select the field to summarize by highlighting it in the list and hitting the arrow button.
3) Select the summary operation (like Sum) by using the pull down list.
4) Click OK.
5) Place the new field in a section (like details) depending on how often it should appear.

A basic running total will solve the first 3 of the 4 scenarios mentioned above. If you put this field on the Detail band, it will provide a running balance. If you put this field on the Report Footer, it will provide an accurate total in reports that use TopN or Group Selection.

Using the Evaluate setting:
To eliminate duplicate values you have to have a group field in the report that groups the duplicate values together. If you do, you can use the "Evaluate" setting of the running total to eliminate duplicates. The "Evaluate" setting tells the running total how often to read a value in the column. One of the options is "once per group". For example, linking the customer table to the orders table causes duplicate values for all customer fields. Summing a field from the customer table would sum the duplicates. You could group the duplicates together by adding a group on Customer Name. Then you could "Evaluate" the running total once per Customer group, which would skip the duplicates.

In special situations you can use evaluate to include records that meet a formula condition. Enter the condition as a Boolean expression using the "X+2" button in the "evaluate" section.

Using Reset to Creating Running Subtotals:
If a running total needs to start over at zero periodically, you can use the reset option at the bottom of the running total design window. This setting tells Crystal when the reset should occur. By default the reset is set to "Never", which means your running total is a grand total. You can reset when a group changes, or when a field changes. Either of these would create a subtotal.

In very rare situations you may need to define a reset based on a logical condition. Enter the condition as a Boolean expression using the "X+2" button in the "evaluate" section.

☞ **Adding an Automatic Running Total (See Exercise 21)**

Variables, an Overview:

A variable is a place that you create in Crystal's memory to store a value for later use. By storing a value in a variable, you can:

1) Carry the value from an earlier part of the report to a later part.
2) Pass the value from a subreport back to the main report (or vice versa).
3) Create totals of totals (or create totals of formulas that use totals).

Imagine a report that lists stock trades, and the price of each trade, over the course of a day. If you wanted to compare the latest price to the opening price (taken from the first record) you would have to carry the opening price forward to the latest record.

When you use the field "price" in a formula, it refers to the current record's price. There is no way to specify that it should be using the price of the first record. But, you can store the first record's price in a variable. Then you can use the variable in another formula and it will still hold the price from the first record. This allows you to calculate the net change over the course of the day.

You create variables within your formulas. So, you could write a formula called "Assign" and in it you could create a variable called "Open". Then you would have that variable store the trade price from the first record of the report. A variable holds its value until you assign it another value, so "Open" won't change unless you assign it another value.

You can use a variable name in a formula just like you use a field name. A numeric variable, like "Open", can be used in a formula just like any numeric field. When Crystal calculates the formula it will use the current value of the variable in the calculation. So, if you use "Open" in a formula, anywhere in the report, it will represent the value taken from that first record of the report.

You can periodically assign a different value to that variable. One formula can update a variable several different times in the same report. This happens when a formula is placed in a section that repeats, like the Group Header. Each time that section repeats, the formula gives the variable a new value, taken from the current record. In some reports, several different formulas work together to update the value of a variable at different times in the report.

Take our stock example. If our report included several stocks, and was grouped by the ticker symbol, you would want a new "Open" price at the beginning of each group. You could place the formula "Assign" in the Group Header and "Open" would be updated at the beginning of each group, taking its value from the first record in the group. The variable would hold the value until the beginning of the next group when it would be assigned a new value.

Variables, Getting them to work:

There are five things to keep in mind when using a variable in a formula:

1) Start the formula with the WhilePrintingRecords or WhileReadingRecords:
Crystal goes through the report's data twice. The first pass is as it reads the records sent by the database. The second pass is as it prepares the page layouts for the printer. In between the passes the records are grouped, sorted and totaled. In Crystal terminology these passes are called **WhileReadingRecords** and **WhilePrintingRecords**. Crystal can tell when to calculate a regular formula. Most formulas are calculated on the first pass. But formulas that use a subtotal, a grand total, the Next() function or the Previous() function are automatically held for the second pass. This is because the totals need to be created, and the records need to be sorted before these formulas can be calculated.

However, when you write formulas with variables Crystal may calculate a formula before you intend it to be calculated. So, you need to specify the evaluation time at the beginning of the formula. Formulas that will be used for sorting, grouping, selecting or totals need to be calculated **WhileReadingRecords.** Otherwise the formula will not be ready in time. But, if the formula will use a total or a subtotal, then you need to use **WhilePrintingRecords.** Otherwise these totals won't be ready for your formula to use. I have found that 90% of formulas that use variables need WhilePrintingRecords.

For a more detailed explanation of this topic go into the Crystal Help Contents and find the section titled "Report Processing Model". There is a helpful diagram named the "Multi-pass reporting flow chart" at the end of this section that you might find helpful.

2) Create the variable using a Declaration Statement:
A variable must be "declared" before it can be used in a formula field. This is done using a "declaration statement" made up of two or three words, such as:

```
Shared NumberVar Total
```

The first word is the scope of the variable. This is optional and is described on the next page.

The second word is required and must be one of the seven "Var" words. The most common are NumberVar, DateVar and StringVar. The word you choose tells Crystal the data type of the values you plan to store (number, date, text, etc.). Once a variable has been declared, it can only store values that match its data type. All of the "Var" words are listed at the bottom of the "operators" list (blue gears) in the formula editor.

The third word is the name of the variable, which can be almost any word you choose. It must start with an alpha character, but it can include numbers and the underscore. It can't include spaces or symbols. It also can't be the same as any Crystal function, like "Sum". Each variable used in a formula gets its own declaration. If the variable is used again in another formula field, it must be declared at the beginning of that formula as well.

Scope of a Variable:
You have the option of giving a variable a "scope" when you declare it, which determines how it is shared with other formulas. You do this by adding one of the following three words to the beginning of the declaration statement (i.e. before the word NumberVar). Global is used by default if no other scope is specified.

> **Global** – This is the most common. All of the formulas in a report can share and update a global variable. But each formula must still declare the variable before using it. The variable is not shared with subreport formulas. So, if the same variable name is used in a subreport formula, the subreport variable is treated as separate value. Since Global is the default scope, you don't see it specified very often.
>
> **Shared** – All of the formulas in a container report and its subreports can share and update this variable. Each formula must declare the variable as shared before using it.
>
> **Local** – This is fairly rare. This variable value is not shared with any other formula, and does not even carry to the next record. This is used primarily in complex formulas to carry a calculated value to a later section of the same formula.

3) Separate phrases with a semi-colon:
Formulas that contain variables always have more than one phrase. The Evaluation Time and the Declaration are usually the first two phrases of a formula. You will usually have a third phrase that assigns a value to the variable. You mark the end of each phrase with a semi-colon. You may put a long phrase on more than one line. For instance, **If-Then-Else** statements, even if they are typed on several lines, are considered one phrase. So, the first two lines of most variable formulas would look something like this:

```
WhilePrintingRecords;
NumberVar Total;
```

4) Store a value using Assignment Operator:
Once you declare a variable at the beginning of a formula, you can store a value into the variable by using the "assignment operator". This operator is a colon and equal sign with no space between them. The variable name goes on the left of the assignment operator and the value to store in the variable goes on the right, as in the following 2 examples:

```
Total := {Orders.Order Amount} ;

Invoice := if {Orders.Order Amount} < 50
             then {Orders.Order Amount}
             else {Orders.Order Amount} * .90 ;
```

The value to store can be a field, a literal value, another variable; or a formula expression that combines these. You might even see the variable from the left side used within the formula on the right side of the assignment operator. The calculation would use the current value of the variable, and the result of the calculation would become the new value of the variable. This technique is described in more detail in the section titled "Running Totals - the Old Way".

5) Determine how often the variable is assigned a value.
If you write a formula that assigns a value to a variable, and then don't put that formula on the report layout, the formula will not "fire", meaning it will not update the variable. To get a formula to "fire" you have to place it on the report, in one of the 7 sections. The section you choose is important, because it determines how often the formula will fire. Each time the section containing the formula is processed, the formula will fire, and the variable will be assigned a new value. This happens even if the section containing the formula is hidden or suppressed.

So, if you want your variable to be assigned a new value on every record of the report, you put the formula that does the assignment into the detail section. If you want the variable to be assigned at the end of each group, you put the formula in the Group Footer.

Displaying a Variable without incrementing it:

Occasionally, you will want a formula that simply displays the value of the variable without changing it. This requires a separate formula. If you tried to use a copy of the formula that does the assignment, you will cause it to fire twice. The variable would be assigned values by both copies of the formula. By using a different formula you can display the current value of the variable without repeating the assignment. This "Display Only" version of the formula has to declare the variable, but it does not include an assignment line. The formula below is an example. It will display the current value of the variable without firing:

```
WhilePrintingRecords;
NumberVar Total
```

You could place this formula field anywhere in the report, and it would display the variable's currently stored value each time it appeared.

☞ **Variable Basics (See Exercise 22)**

Running Totals – the Old Way:

One of the most common places that you will find variables used, is in running total formulas. In very old versions of Crystal, this was the only way to create a running total. Now, most running totals can be created using the Automatic Running Total feature described earlier. However, there are still some situations that require the older technique:

1) Accumulating values coming from subreports.
2) Accumulating values from a formula that uses a total.

In both cases you can't use a regular total, and you can't use an Automatic running total. But, you can do these with running total formulas, using variables. For instance, take a formula that applies a discount to each order based on the customer's total sales (a subtotal). Because the discount formula has a subtotal in the calculation, you won't be able to total that formula the normal way, with a summary field. However, you can write a formula using a variable that will accumulate the values to provide a total.

The Running Grand Total:

The formula for a basic running total looks like this:

```
WhilePrintingRecords;
NumberVar  Runtotal;
Runtotal:= Runtotal + {@discount}
```

This formula has three phrases and the first 2 should be familiar. The first tells the formula to wait until the "second pass" to execute, because the discount formula has to wait for the subtotals to be calculated. The second line declares a variable and names it Runtotal.

The third phrase is the key, because it looks like the variable is assigned to itself. What this means is that the formula will take the current value of Runtotal, add the current record's discount, and store the result as the new value for Runtotal. If you place this field on the detail band of the report, it will fire on every record, and each time it will add one more discount to the variable. By the end of the report it will have accumulated all of the Discounts.

If you didn't want to see the accumulation as it occurred, you could suppress this formula and use a display formula in the Report Footer to see the end result.

The Running Subtotal:

To make a running grand total into a running subtotal, you have to add another formula to do a reset. This reset will assign a value of zero to the variable, at the beginning of each group. The formula would look like:

```
WhilePrintingRecords;
NumberVar  Runtotal;
Runtotal:= 0
```

You place this formula in the Group Header so that it fires once per group, setting the variable to zero before the group starts. After the Group Header, the running total will start from zero and accumulate through the group. At the end of each group the variable value will contain the subtotal of that group. You can add a third formula to display the variable on the Group Footer. This is just before it is reset by the next Group Header. This technique is know in Tek-Tips as the "3-formula technique", because it uses 3 formulas to manage the variable,

Accumulating values once per group:

The Automatic Running Totals we discussed earlier have an "Evaluate" setting which tells the total how often to accumulate a value from the column. This allows you to eliminate duplicates by setting the Evaluate to "Once per Group".

You can accomplish the same thing with a running total formula, by moving the accumulate formula to a Group Header or Group Footer. The running total formula would be the same:

```
WhilePrintingRecords;
NumberVar  Runtotal;
Runtotal:= Runtotal + {@discount}
```

But, by placing the formula in the Group Header or Group Footer (instead of the Details section) the formula will only fire once per group, and therefore will only include one record from each Customer Group in the running total. One advantage to using a Running Total formula is that an Automatic Running Total will always take the first value of each group. You don't have a choice. With a Running Total Formula you can choose either the first or last record of the group. You do this by putting the accumulation formula in either the Group Header or Group Footer.

☞ **Running Totals with Variables (See Exercise 23)**

Using Local Variables:

A local variable is not shared with any other formula, and does not carry a value to other records. They used primarily to simplify a formula that has to use a complex expression more than once. The expression can be calculated once, and the result stored in a variable. You can then use the variable name later in the formula, instead of repeating the expression.

I often use local variables when I publish formulas on my website. Since users usually have to substitute their field names into the formulas, I often start a formula by assigning the input fields to variables. I then use the variables in the rest of the formula, instead of the fields, themselves. This way, the user will only have to replace the field names once, in the first lines of my formula. This is safer than having them replace every field name throughout a complex formula. Here is an example:

Finding the number of working days between two dates:
Customers often measure productivity by determining how many days it took to complete a task. They often try to eliminate days where no work is done, such as weekends and holidays. Below is a formula that calculates the number of working days between two dates. The first two local variables (Start and End) allow the user to substitutes their fields into the formula.

The other three variables do independent calculations, and are assembled at the end for the final calculation. Notice that the formula updates the Holiday variable (Hol) several times. Each line checks a different holiday, as specified by the user. These are subtracted from the workdays at the end. Instructions for implementing this formula are on my website.

```
WhilePrintingRecords;
      Local DateVar Start := {table.Start};//Insert your Start Date field
      Local DateVar End := {table.End};  // Insert your End Date field
      Local NumberVar Weeks;
      Local NumberVar Days;
      Local NumberVar Hol;

     Weeks:= (Truncate (End - DayOfWeek(End) + 1 -
                  (Start - DayOfWeek(Start) + 1)) /7 ) * 5;
            //count 5 days for each Calendar "Rows" involved:

     Days := DayOfWeek(End) - DayOfWeek(Start) + 1
            //Adjust for starting the first and last weeks in mid week.
           + (if DayOfWeek(Start) = 1 then -1 else 0)
            //adjust for starting on Sunday:
           + (if DayOfWeek(End) = 7 then -1 else 0);
            //adjust for ending on Saturday:

     //Adjust for midweek Holidays between the start and end dates:
     if Date(2002,09,02) in start to end then Hol:= Hol+1;
     if Date(2002,11,28) in start to end then Hol:= Hol+1;
     if Date(2002,12,25) in start to end then Hol:= Hol+1;

     //Assemble the adjusted workdays
     Weeks + Days - Hol
```

How Crystal Reports works with SQL

All databases use their own format for storing and retrieving data. In the not so distant past, transferring data from one database to another required a driver that was specific to both database formats. Whenever the rules for one format changed, all of the drivers using that format had to be updated to work with the new rules. Crystal still comes with direct drivers for MS Access, DBase, FoxPro, Btreive and Paradox.

The problem became more manageable when SQL (Structured Query Language) was developed. Any database that could "talk" SQL, could get data from any other program that could also "talk" SQL. By using SQL instead of direct drivers, the databases no longer needed to maintain drivers for every other database. They would only have to keep one driver up to date; the one that translated their requests into SQL. It didn't take long for databases and other programs to incorporate SQL capabilities into their database programs.

Crystal can also send SQL requests, which is why Crystal can request data from any SQL compatible database. To see what the SQL language looks like, open a report that was created using the SQL/ODBC option and use the menu options **Database ➡ Show SQL Query**. What you see is called the SQL (sometimes pronounced "Sequel") statement. This is the request that is sent to the database.

The basic SQL Statement has up to five sections:

The "Select" clause, listing the fields that will be retrieved from the data tables.
The "From" clause, listing the tables that will be retrieved, and how they are related.
The "Where" clause lists criteria for record selection, and (sometimes) how tables are related.
The "Order By" clause lists the fields used for sorting the records that are returned.
The "Group By" clause organizes the records into "chunks" based on a field value.

You will always see the first 2 statements (Select and From). The other statements appear when they are needed. The SQL statement window is only for viewing or copying the statement. To make changes to this statement you must use the "command" feature of Crystal Reports.

ODBC allows Database Drivers to Swap SQL with each other:
SQL Statements are generated and interpreted by the database drivers that come with each database, but the statements still have to get from one program to another. To make it easy for applications to swap SQL, Microsoft helped develop the ODBC (Open DataBase Connectivity) protocol. This provides a "switchboard" for getting each SQL request or response to the right place. The heart of this is a program called the ODBC Administrator. The ODBC Administrator keeps a list of all of the SQL databases and programs installed on your PC, and where they store their driver files. Whenever a new SQL compatible program is installed, it "registers" its drivers with the ODBC Administrator. If it doesn't, the Administrator won't be able to understand requests intended for that database. You will find the ODBC Administrator in the Control Panel. It usually called either "Datasources (ODBC)" or "32 Bit ODBC", and it is sometimes found in a folder called "Administrative Tools".

Creating SQL Expressions:

SQL Expressions are just like Crystal formulas, except that they use SQL syntax and SQL functions instead of Crystal formula syntax and functions. SQL Expressions are calculated by the database, not by Crystal Reports. In some situations they can speed up your reports by allowing the database to do more of the work.

The speed of most reports depends primarily on how long it takes Crystal to get the needed records from the database. Crystal starts the process by converting the record selection formula into a SQL statement and sending it to the database. The database uses the SQL statement to select the requested records and then sends them back to Crystal. Crystal then generates the report. But, sometimes part of the selection formula can't be translated into SQL. This means that the database will send back more records than Crystal wants, and Crystal will have to eliminate these records. The time spent to process these extra records (both by the database and by Crystal) can add significantly to the processing time of the report.

There are several ways to make things more efficient. One option is to try to change the selection formula so that all of it can be interpreted into SQL. This is often a hit or miss proposition because each database is different.

A second option is to see if any of your selection formula can be written as a SQL expression. Adding a SQL expression to the report is just like adding a formula field, except that you are limited to the functions that are supported by your ODBC driver. Some formulas can be replaced with a SQL expression that does the exact same thing as the formula. The advantage to using the SQL expression is that it will convert directly into the SQL statement and therefore be processed by the database.

To add a SQL expression you use the Field Explorer and highlight "SQL Expression Fields". When you click "New" you will be asked to give the expression a name. When you click "OK" you will see a formula editor window that is similar to the one used to write formula fields. You will notice, however, that there are fewer functions and operators to choose from. The listed functions and operators are those supported by your ODBC driver. A report that uses a different ODBC driver may provide a different list of operators and functions.

Once a SQL expression is added to a report, you can use it like any formula field, even using it within other formulas. You can't, however, use the name of one SQL expression within another SQL expression. You would have to repeat the first expression within the second.

Using "Select Distinct Records" to eliminate duplicates (only in SQL reports):
If you want Crystal to change the word "SELECT" to "SELECT DISTINCT" in the SQL Statement, you can activate this by using **Database ➡ Select Distinct Records**. This will eliminate a duplicate record from your report, but only if all fields selected for this report are the same as another record. If two records are similar, but use one field that is different, they will not be considered duplicates. Both will show up on the report.

☞ **Using SQL Expressions (See Exercise 24)**

Creating SQL Commands:

Crystal Reports now allows you to write a SQL statement, written against one of your DSNs, and use it as a data source. The statement is called a 'command' and can be stored either in a report, or in the repository. Storing the command in the repository would make it available to all reports that have access to the repository.

To add a command you must be able to write (or at least copy) a SQL statement. It helps to have a valid and tested SQL statement handy (sitting in notepad) when learning this subject. One easy way to get a valid SQL statement is to open an existing Crystal Report and use the SQL statement generated by the report.

To create a report using a command, open the data explorer as if you were going to use a table from a data source. Instead of opening the list of tables to select a table, double click the option just above tables, the one marked "Add a Command". This will open a window where you can type or paste your SQL statement. Click OK at the bottom of this window and the results of your SQL statement will be your source 'table' for the report. When you look at the fields available for this report, the list will be the list of fields selected by the SQL statement.

You can now create any report you like using these fields, as long as you keep the following constraints in mind. You are limited to the records in the database that meet the WHERE clause in the command. You can reduce the number of records by using the Select Expert or the Selection Formula, but these additional criteria will be performed locally, rather than at the server. You also lose the ability to add SQL expression fields. Any SQL expressions would have to be added into the SELECT clause of the command. In other words, nothing you do in the report will affect the SQL statement. To change the SQL you must edit the Command.

Making changes to the command:
If you want to modify the command you need to open the Database Expert. You can right click on the command and select 'Edit Command' to open the command window. When you make changes and click "OK" your modified command will immediately be sent to the database. There is no prompt to ask you if you want to refresh.

Storing the command in the repository:
(**Note** – the repository is only available if you are using v9 or using Crystal Enterprise.)
While you are in "Edit" mode you will see a check mark at the bottom of the window. If you check this box before you click OK, Crystal will add a copy of this SQL command to the repository so that it can be used to create other reports. A window will open which will allow you to give this command a name and make any final changes before it is saved.

The location box at the bottom allows you to place this command into a specific folder. You can use the existing Command" folder, another existing folder within this folder, or you can create a new folder. To select an existing folder you highlight that folder before saving the command. To create a new folder, you right-click on the folder marked "commands" and select the option 'new folder'. You click OK when you are ready to save the command into the repository.

Using parameters in commands.
You can also embed parameter prompts directly into a SQL Command. This is one of the reasons why these commands can perform much better than the SQL generated automatically by Crystal. You can add these prompts when the command is created or later if you edit the command. On the right hand side of the command window you will see the parameters list and buttons labeled, "Create", Modify and Remove. Note that these cannot be multiple value parameters (except in v12-v14) or range parameters. You can use 2 separate parameters to create a range option, if needed.

If you click the Create button Crystal will ask you for four properties for this parameter. These properties are virtually the same as the properties in report parameters. Once you have saved a new parameter you place your cursor in the command and double click the parameter. It will appear wherever your cursor was sitting in your command. For some reason, a string parameter must be placed in single quotes for it to work correctly. These quotes are placed around the brackets of the parameter. This allows you to embed wildcard characters such as '%' into the command, after the parameter.

The command parameters will automatically appear in the report as report parameters. If you want, you can change the name of the report parameter, add a list of default values or place the parameter on the report. However, if you change the command (or the command parameter) in any way, changes made to the report parameter will disappear. The report parameter will be renamed back to the name used in the command. If you had used the renamed parameter in a formula, then the formula will generate an error. Therefore, it is best to rename the command parameter and then to make the corresponding changes to the command. If you remove a parameter from a command, that parameter will be deleted from the list of command parameters, and will also be deleted from the list of report parameters.

Below are syntax examples for using various command parameters in MS Access, but these will vary based on your DBMS.

```
`Customer`.`Customer Name` > 'C'
`Customer`.`Customer Name` > '{?string}'

`Orders`.`Order Date` >= #2000-02-25 00:00:01#
`Orders`.`Order Date` >= {?date}

`Orders`.`Order ID` < 3000
`Orders`.`Order ID` < {?number}
```

☞ **Creating SQL Commands (See Exercise 25)**

Saving a Report with Data:

Saving Data with the Report (File ➡ Report Options):
You can use this technique if your user has Crystal Reports, but doesn't have access to the data used to create your report. You can also use this technique when you want save a version of the same report showing the data "as of" a certain time. When you click the menu options **File ➡ Report Options** you will see an option called "Save Data with Report" in the upper left. It may or may not have a check mark next to it. If there is a check mark next to this option, and you save the report, Crystal will also save the records currently in memory. These will be stored into the RPT. If there is no checkmark, and you save the report, Crystal saves only the settings in the RPT.

You can see the number of records that Crystal will save at the bottom of the preview screen, but not all fields are saved. Fields are only saved if they are used by the report. If you want to know exactly which fields are being used in a report, open the field explorer and expand the database fields of any table**.** The fields in use will have a small green check mark next to their names.

When you open a file that was saved with its data, it goes directly to preview mode, allowing you to view or print the report. It won't try to connect to the database unless you try to refresh the report. If you send this RPT file to a user who has Crystal, they can open it and view the data as it was when you last saved the report. They can even change the layout, formulas, sorting and selection criteria – without having the database. They can make any change, as long as the change does not require the report to refresh the data. If the change requires additional records or additional fields, then it will try to refresh the data and it will need the database connection. If the user doesn't have a connection, this refresh will cause an error message and will cause you to lose your saved data.

This feature of Crystal allows developers to create reports for customers via Email, without ever being on-site. I have many reporting customers that I have never met. As long as the customer can create a simple RPT file with the necessary fields, I can do the rest in my office. If I need different records, or an additional field, I have to send it back to be refreshed.

☞ **Saving a report with data (See Exercise 26)**

Printing Reports Consistently in Different PC Environments:

Reports designed on one PC may look different if run on a PC with a different printer or printer driver. Changing printers may cause text-based objects to truncate or wrap in unexpected ways. This is because each Crystal report calculates object placement based on the printer driver that was used to design it.

Crystal Decisions recommends that you use the following techniques to make your reports less sensitive to printer driver changes:

1. Make text objects slightly wider than they need to be on the development PC.
2. Set "free form placement" to "on" in each section prior to distribution.
3. Remove guidelines from the report prior to distribution.

These changes make the report as generic as possible. It also helps to have the latest drivers for your printers.

Using Printer Names:
When you save a report, Crystal remembers the *name* of the printer that you used. When you run the report on another printer, Crystal will look for that printer on the target machine. It doesn't look for the printer by its driver or model number, but by the name given to that printer in Windows (Like "Accounting LaserJet"). If Crystal doesn't find the exact printer name on the target PC, it will use that PC's default printer. It will bypass other available printers on that PC, even if they are the same model as the original printer.

If you must point some reports to a specific printer, and it won't be the users' default printer, you can create a duplicate printer selection with a specific name to use on each PC. For instance, you could create a printer called "Report Printer" on both the development and target PCs. You can then create reports using the development PC "Report Printer". When distributed, the report will look for a printer named "Report Printer" on the target PC'. The report will use "Report Printer", even though it may point to a different model printer. You can also change the default settings for "Report Printer" without changing the settings of their true default printer.

For more detailed information on how Crystal works with printer drivers, you should visit the technical support area of the Business Objects website. There is a file that you can download called:

SCRprinterdependency.pdf

I have a link to this file on the Links page of my website.

Technical Support and other resources:

Crystal Reports technical support: http://www.sdn.sap.com/irj/boc/support/
Tek-Tips technical forum: http://www.tek-tips.com/
My web site: http://www.kenhamady.com/

Custom Functions:

The Crystal Reports formula editor comes with dozens of standard function, like Val () and Length(). Crystal also allows you to write your own custom functions, to store specific business logic or commonly used calculation patterns. In prior versions of Crystal, custom functions had to be written in a programming language and then compiled as a DLL. This is still an option. However, you can now create custom functions by writing a Crystal formula. These functions can then be stored so that they can be used by different formulas, or even different reports.

Adding a function:
You add new custom functions in the Formula Workshop. First you highlight the Report Custom Functions category. Then you click the "New" button on the toolbar and give the function a name. At this point you can either use the Editor or the Extractor to create your function. With the Editor you write a function formula to create the function. The "Extractor" will convert an existing formula field into a function formula for you.

Using the Editor:
When you use the function editor to write the function you are writing a formula with variables. Below is an example of a function written in the Function Editor. The example function will take any date value and calculate the first day of that date's month. Optionally you could have it return a different day of the month by giving it another number argument. If you gave it the number 10, it would calculate the 10th of the month. It uses the number 1 as a default.

```
Function ( DateTimeVar DateIn, optional NumberVar NumberIn := 1 )
DateIn - Day( DateIn ) + NumberIn
```

Rules for writing a custom function:
The first line always starts with the word "Function" followed by a pair of parentheses.
You declare each user input variable with 2 words that go inside the parentheses.
If there is more than one user input, each declaration is separated by commas.
Every user input needs a "Var" word (NumberVar, DateVar, etc.) to specify the data type.
After the "Var" word you type the name of the variable, which you can choose.
If an input is optional, you put the word "optional" in front of the "Var" word.
If optional, you must also assign a default value to the variable, using the assignment operator.

Once the variables are declared in the first line, you write an expression that tells the function how to use the input variables to perform your calculation. This expression can use any of Crystal's other functions or operators. Once the expression is saved it is added to the "Report Custom Functions" list and is available for use in any formula in that report. Below we will talk about adding to the repository so that it can be used in other reports.

Using the Extractor:
If you have an existing formula that already does your function calculation, Crystal can turn that formula into a custom function by using the Extractor. The Extractor copies the formula into a new function and automatically replaces each field in the formula with a variable name. If the field is used several times, the same variable name will be used each time. The Extractor then writes a declaration that includes all of the variables. It leaves the rest of the expression in place, including operators, literal values, and most standard Crystal functions.

You choose the Extractor just after you enter the name of your new function. Crystal will then open the Extract window. The first step is to select the source formula field from the list on the left. The top box will show the name you entered, while the second box will show the data type for the function's output, based on the source formula. The summary box is for an optional description. The "Arguments" box will list the fields that the Extractor will convert into variables. For each field, it lists the data type of the field, and a default name for the variable (which you can change). There is even room for a description for each argument.

If you enter a checkmark in the lower left corner, Crystal will replace the source formula with an equivalent formula that uses the new function. The advantage of this is that if you update the business logic in the function, the source formula will also be updated at the same time.

Overview of the Repository:
(**Note** – the repository is only available if you are using v9 or using Crystal Enterprise.)
A custom function starts out being available to all formulas in the current report. But, one of the reasons for having a custom function is to have it in multiple reports and then be able to update the function in all reports at once. Storing functions in the repository makes them available to all other users who have access to your repository database. When you update a function within the repository your change will affect all of the reports that read this function from the repository.

Storing and using Functions with the Repository:
It is a simple process to move a function from the "Report Custom Functions" list to the repository. Simply drag the function from the "Report Custom Functions" list to the repository area and it is added to the repository. Once a function is in the repository, that function can be added to any report that has access to the repository database. But, if the report is moved to a PC that doesn't share the repository database the function will not be recognized and will cause the formula to error. Alternately, the function can be disconnected from the repository, but then it will no longer respond to changes in the repository.

Making Changes to a Function in the Repository:
If you need to change a repository function, the first step is to disconnect the function from the repository. This is done by adding the function to the report so that it appears in the "Report Custom Functions" list. You can then right-click on the function to disconnect it. It is now available for changes. Once the function is changed, it can be saved and then re-added to the repository. If added with the same name, it will replace the original function.

☞ **Custom Functions and the Repository (See Exercise 27)**

Reusing Objects with the Repository:

(**Note** – the repository is only available if you are using v9 or using Crystal Enterprise.)
The repository is a database of reusable objects that can be shared across users and reports. There are 4 categories of objects that can be stored in the repository. Custom Functions, Commands, Text Objects and Images. To store a Text Object or an Image in the Repository you simply drag the object from the report design area into the Repository Explorer, and drop it into the appropriate folder.

Once one of these objects is placed into the repository, it is available to any report, as long as the report has access to the same repository database. The object will be refreshed from the repository each time that report is opened. If the object in the repository has changed, then the change will take place the next time the report is opened. The database is usually stored in:
C:\Program Files\Common Files\Crystal Decisions\2.0\bin\Repository_en.mdb

When you use an object from the repository, the object stays 'connected' to the original object in the repository. These objects cannot be modified as long as they are connected to the repository. To modify an object, it must be "disconnected" from the repository. This allows you to change the object within an individual report, but this object will no longer be updated automatically by changes in the repository. The object becomes specific to the report. So, for instance, a repository custom function that is disconnected will now be listed as a report custom function, rather than as a repository custom function.

Modifying Repository Objects:
Crystal doesn't allow the objects to be modified while they are in the repository. If you need to change an object that is in the repository, you must use the object, disconnect it from the repository, modify the object, and then add it back in again after it has been modified. The way that you "use" an object will vary depending on the type of object:

1) Text objects and images must be physically placed onto the report.
2) Functions must be added to the report's custom functions list.
3) Commands must be used in the Database Expert window.

Once the object is in use it can be disconnected, modified, and added back to the repository. If the modified object is added back with the same name, it will replace the original object.

Deleting Repository Objects:
To delete an object from the repository, simply right-click on the object and select "delete". This is usually done in the Repository Explorer window, except for Custom Functions. These do not appear in the Repository Explorer. Custom Functions are only listed in the formula workshop. When you open the Formula Workshop you will see a list of Repository Functions. You can right click on any function, and select "delete".

Deleting Repository Folders:
If you have folders within your repository categories, you can only delete the folder when it is empty. Then you can right click on any folder, and select "delete".

☞ **Using the Repository (See Exercise 28)**

Expert Class Exercises:

Creating the Master Report for use in later lessons.

Create a new report using the report expert.
Use the ODBC Database – "Xtreme Sample Database xx".
Select the two tables Customer and Orders.
The links using Customer ID are correct.
Select the fields Customer ID, Customer Name, Region.
Open the Orders table and select Order ID, Order Amount, Order Date.
Group by Customer Name and move to Summaries.
Remove the summary for Customer ID.
Highlight Order ID on the right and change the summary from "Sum" to "Count".
Click "Next" three times to get to Record Selection.
Move Order ID over to the right.
Change "is any value" to be "is between"
Enter the numbers "2300" and "2900" in the boxes.
Click the "Finish" button to preview the report.
Save under the name Master.rpt and close this report.

Exercise 1 - Specified Order Grouping

Open the Master.rpt and then save it under the name Specified.rpt.
Using Specified.rpt, use the menu options **Report ➡ Group Expert**.
Highlight the group "Customer Name" and click the "Options" button.
Change "ascending" to "specified order"
Click on the Specified Order Tab.
Use the drop-Down under "Named Group" and select the following names:
Fulcrum Cycles, Extreme Cycling, Changing Gears.
Click on the others tab (on the right) and set to "leave in their own group"
Click "OK" twice, and note that these groups are first, followed by the rest.

For Custom Headings:
Change the group field to be Region, and click on the "Specified Order" Tab.
Delete the Customer name entries on this tab.
Now click the button marked "New" and change "Untitled" to "Eastern Region".
Change the Rule to Region / is one of / MA, NJ, PA and say "OK".
Click New, change "Untitled" to "Central", select Region / one of / IA, ID, IL
Click New, change "Untitled" to "West", select Region / is one of / BC, CA, OR.
Switch to the others tab and check "Discard all others".
Preview the report and note that only three groups appear.

Exercise 2 - Review of IN Operator

Open the Master.rpt and then save it under the name IN.rpt.
Delete the Customer ID column and heading

Insert a formula field called Flag that says:

```
if "Bike" in {Customer.Customer Name}
then {Orders.Order Amount}
else  0
```

Place this field on the detail band.
Insert a subtotal and Grand total of this field, then Preview.
Note that only certain groups have Amounts.

Change the first line of this formula to say:

```
If {Customer.Postal Code} in "10000" to "29999"
```

Notice that the second group and others have Amounts to be totaled.
Change the first line of this formula to say:

```
If  {Customer.Region} in ["CA","BC","AZ","OR"]
```

You will see Amounts on pages 2 and 3.
Change the first line of this formula to say:

```
if {Orders.Order Date} in YearToDate
```

Use **Report ➡ Set Print Date** to change today's date to be 7/1/2002.
Preview and note the Amounts on records considered YTD.

Exercise 3 - Using StartsWith, Like and Comment Marks

Start a new report using just the Product Table.
Select the fields Product Name, Size and Price, and click Finish.
Insert a formula field called "Group".

```
if {Product.Product Name} StartsWith "Xtreme"
then "Xtreme Product" else
if {Product.Product Name} Like "*Sadd*" then "Saddle" else
if {Product.Product Name} Like "*Helm*" then "Helmet" else
"Other"
```

Add this to the detail band and preview.
Note that Xtreme Saddles and Helmets are considered Xtreme Products.
Edit the formula and put 2 slashes in front of the first and second lines.
(These lines should turn green)
Save and close the formula editor.
Note that the Xtreme Products now fall into the other categories.
Save this report under the name Patterns.

Exercise 4 – Managing null values

Keep working with the report, Patterns
Note that some products don't have sizes.
Write the following formula called Sized:

```
if {Product.Size} = ""
then "Not Sized"
else "Sized"
```

Note that products with no size value do not print UnSized.
Go to **File ➡ Report Options** and check "Convert Null to Default".
Note that the formula now prints a value.
Go to **File ➡ Report Options** and UNcheck "Convert Null to Default".
Note that the formula now prints blanks again.

Change the formula to be:

```
if Isnull ({Product.Size} )
then "Not Sized"
else "Sized"
```

Note that the formula now prints a value on all records.

Go to the select expert and select where:
Size / is less than / 0

Note that no records are found.
Now change the select expert to say where:
Size / is equal to / (leave third box blank)

Note that no records are found.

Use **Report ➡ Edit Selection Formula ➡Record** to enter the following formula:

```
IsNull( {Product.Size} )
```

Make sure that you have both parentheses and curly brackets in the formula.
Now you will see just the records that have no size.

Exercise 5 - Using InStr, and Length

Start a new report using just the Customer Table.
Add the fields Customer Name, Address 1 and click Finish.
Now, we want to sort the records by street name.

In the Select Expert, add the rule: {Customer.Country} / is equal to / USA
Note that the street name is always after the number and an empty space.

Create a formula field called "Space" that is the following:

```
InStr ({Customer.Address1}, " ")
```

Place on the detail band to see the position of the space in each record.
Create another formula field called "Length" that is the following:

```
Length({Customer.Address1})
```

Place on the detail band to see the length of each entry.

Now create a third formula field called “StreetName”: z

```
{Customer.Address1} [ {@space} + 1 to  {@Length} ]
```

This should give you the Street Name by itself.
You can now sort on this new field, and then delete all formulas from layout.

Exercise 6 - Using Val

Create a new report using the Customer Table
Select the fields Customer Name, Address 2, and Address 1
Set the select expert to:
Address 2 / starts with / Sui
Preview and delete Address 1

Sort on Address2 and note that they are not in numerical order.
Add a formula called “Suite Number” that is:

```
Val ( {Customer.Address2} [ 7 to 12 ] )
```

Place this on the detail band.
Change the sort to use Suite Number instead of Address2
Note that it sorts correctly.
Note that Suite B is at the top, since the letter B has a numeric value of 0.
Delete the formula from the details band and note that this does not affect the sort.

Exercise 7 – Using ToText

Create a report using only the Products table in the Report Expert.
Use the fields Product Name, Product ID and click Finish.
Add a formula called “New Name” to concatenate them and try the following:

```
{Product.Product Name} + " - " + {Product.Product ID}
```

The above will generate an error because you can’t concatenate a number.
Change the formula to say:

```
{Product.Product Name} + " - " + ToText ( {Product.Product ID} )
```

Place the field on the detail band and preview.
The formula works, but gives you an ugly number format. Change the formula to:

```
{Product.Product Name}+" - "+ ToText( {Product.Product ID},0, "")
```

This last option will format the number as a string, rather than a numeric.

Exercise 8 - Conditional Formatting

Open up the report called “Master”, and save it as “Conditional Formatting”.
In design mode, click on the field “Customer Name” on the details band.
Select **Format ➡ Borders and Colors** from the menu.
Click the condition button next to the attribute “Drop Shadow”.

In the formula editor type the following Boolean Formula:

```
{Orders.Order Amount} > 8000
```

Save the formula and preview the report.
You will notice that the records over $8,000 have a drop shadow.

Again, click on the field Customer Name on the details band.
Select **Format ➡ Borders and Colors** from the menu.
Click the condition button next to the attribute "Background".
In the formula editor type the following Formula:

```
if {Orders.Order Amount} > 8000
then Aqua
else NoColor
```

Save the formula and preview the report.
Note that records over $8,000 also have a background.

Exercise 9 - Using Alerts

Reopen the Master and save it under the name Alerts
Use the menu options **Report ➡ Alerts ➡ Create or Modify Alerts.**
Click the NEW button
Enter the name "Amount" and the message "Amount over 10,000"
Enter the condition formula:

```
{Orders.Order Amount} > 10000
```

Save, Close and click OK.
Click NEW to add a Second alert
Enter the name "Subtotal" and the message "Subtotal over 35,000"
Enter the condition formula:

```
Sum ({Orders.Order Amount}, {Customer.Customer Name}) > 35000
```

Save, Close and click OK.
Refresh the report and both Alerts should appear.
Highlight the first one and click "View Records"
Note that the window shows only records over 10,000
Switch back to Preview, highlight the second alert and click "View Records"
Note that only the largest 3 groups are displayed.

Close both Alert windows and switch back to preview
Close the Alert window
Use the menu options **Report ➡ Alerts ➡ Create or Modify Alerts.**
Highlight the first Alert and change the condition to be 11,000
Close and refresh
Note that you are down to 4 records.

Exercise 10 - Using InRepeatedGroupHeader

Re-Open your Master Report and save it as "Continued".
Use **Report ➡ Group Expert** to change the group to be by Country.
Use **Report ➡Select Expert** to change the condition to: Country < > "USA".
Use **Report ➡ Section Expert** and add a "New Page After" each country.

In preview, click on "Canada" in the group tree to go to that group.
Go forward one page and note that there is no indication of the country.
Go to **Report ➡ Group Expert ➡ Options (Button) ➡ Options (Tab)**.
Add a checkmark to the property "Repeat group header on each page".
Last, add a formula field called "Continued" and place it in the group header:

```
If InRepeatedGroupHeader
then "Continued"
else ""
```

Preview the report and notice the word "Continued" prints on some pages.
On the first page of each group it doesn't print, but on other pages it does.

Exercise 11a - Parameter fields in v11-v14

Open the Master Report, and save it as Parameters.
From design mode:

Add a State parameter for selecting records:

1) Open the Field Explorer
2) Right-click on "Parameter Fields" and select "New".
3) Enter the name "State"
4) Select "String" for the data type.
5) Click OK.
6) Open the Selection Expert and delete any rules that exist.
7) Click "New" and add a rule that says:
   Region / is equal to / {?State}

({?State} will appear automatically in the drop-down list, above the other Regions)

Click OK and then click the refresh icon.
Enter "CA" for California and click OK.
You should see only California records.

Adding Static Default Values:

Switch to design mode and find the State parameter in the field explorer.
Right-click the State Parameter and select "Edit".
Make sure the "List of Values" option is set to "Static"
In the drop-down marked "value field" select the "Region" field.
The description field can be left blank.

Under the word "Value" click the in box that says "Click here to add Item".
Type the word ALL in capital letters and hit "Enter" to move to the second line.

Go to the bottom table marked “Options” and scroll to the bottom of the list.
Locate the Min length and Max Length properties.
Under the “Setting” column click next to Min Length and type “2”.
Then next to Max Length type “3”.
Go back to the middle of the window and click on the word “Actions”.
Select the first action labeled “Append all Database Values”.
Note that all regions that are 2 characters are appended to your list.
The only 3 character item will be the first item, “ALL”.

Click OK and then click the refresh icon.
Select “Prompt for new parameter values” and click OK.
Notice there is now a drop-down list of state abbreviations to pick from.
Drop down the list and select FL for Florida.
Click OK and note that only Florida records appear on the report.

Exercise 11b – Range and Multiple Parameters in v11-v14

Multiple Parameter:

Switch back to design mode.
Right-click the State Parameter in the Field Explorer and select Edit.
At the bottom locate the option “Allow Multiple Values” and change the setting to “True”
Click OK and then click the refresh icon.
Select “Prompt for new parameter values” and click OK.
In the “Available Values” list select CA and click the first arrow to move it to the right.
Also select FL and move it to the right.
Enter PA in the box marked “Enter a value”.
Click the third arrow to move it to the right.
Click OK and note that there are 3 states on the report.

Range Parameter:

Switch to design mode, and right-click the parameters category in the field Explorer.
Select “New” , name the parameter Date Range and set the Type to Date.
Locate the option at the bottom marked “Allow Range Values” and change it to True.
Click OK
Use the menu options Report – Selection Formulas – Record.
Add the following below the existing formula:

```
and {Orders.Order Date} = {?Date Range}
```

Save and close the selection formula and then click the refresh icon.
Select “Prompt for new parameter values” and click OK.
Leave the 3 states from above in the State parameter’s selected values list.
Select any date range in 2004
(Enter the dates in 2004-12-31 format)
Click OK to run the report and note that you get these 3 regions within this date range.

Switch back to design mode
Use the menu options Report – Selection Formulas – Record.
Enter the following selection formula:

```
      ( if {?State} = "ALL"  then TRUE else {Customer.Region} =
{?State})
      and {Orders.Order Date} = {?Date Range}
```

Save and close the selection formula.
Click the lightning bolt to refresh and select "Prompt for new…"
In the State parameter, use the "Remove All" button to clear the list of states.
Then select the "ALL" option in the list of states and move it to the right.
Select any date range in 2004.
Click OK to run the report and note that you get all regions for that date range.
Save this report as Parameters

Making a Dynamic Parameter:
Open Parameters and save it as Dynamic Parameters.
Switch to Design mode.
Go into the Select Expert and remove the rule for Order Date.
Open the Field Explorer
Right-click on the State parameter and select Edit.
Change from Static to Dynamic.
Leave the Prompting Group Text blank
Leave the Data Source selection as "New".
Click where it says "Click here to add Item" and select the field Region.
At the bottom find the option for "Allow Multiple Values" and select True.
Click OK and refresh the report, select "Prompt for new parameter values".

The list you see is now dynamically generated each time the report is run.
Note that this eliminates the possibility of selecting or entering "ALL" as an option.
Select a few Regions and click OK.
The selected Regions appear on the report.

Making a Cascading Parameter:
Switch back to design mode and open the Field Explorer
Right-click on the State parameter and select Edit.
Leave the list set to Dynamic.
For "Prompting Group Text" enter "Select a Country and then a Region".
Change the Data Source from "Existing" to "New".
Click where it says "Click here to add Item" and select the field Country.
At the bottom find the option for "Allow Multiple Values" and select True.
Click just below Country where it now says "Click here to add Item" and select Region.
At the bottom find the option for "Allow Multiple Values" and select True.
Click OK and refresh the report, select "Prompt for new parameter values".
Move Australia over into the Country box and note that Australian regions appear below.
Move Canada over into the Country box and note that Canadian regions appear below.
Select ON from the Canadian entries and New South Wales from the Australian entries.
Click OK and note that only these regions appear.

Save the report as Dynamic Parameters.

Using a Command to generate a Dynamic Parameter list:

Open Cascading Parameters and save Command Parameters.
Refresh the report using the current parameters.
Open the Database Expert and expand the current connection.
Highlight the "Add Command" option and move it to the right.
Type the following SQL Statement into the command:

```
Select Country, Region
From Customer
where
country = 'Canada' or
country = 'USA' or
country = 'Australia' or
country = 'England'
```

Click OK 3-4 times, including warning messages, until you get to the Refresh window.
Select "Use current parameters" and Click OK.
Switch to design mode.
Find the State parameter in the field explorer.
Right-Click on the State Parameter and click Edit.
Change the Data Source from "Existing" to "New".
Click where it says "Click here to add Item"
Select the field Country, but this time from the Command at the bottom.
Find the option for "Allow Multiple Values" and select True.
Click just below Country where it now says "Click here to add Item"
Select the field Region but again from the Command at the bottom.
Find the option for "Allow Multiple Values" and select True.
Click OK and refresh the report, select "Prompt for new parameter values".

Note that at first only the countries listed in the command are available.
Select the countries of Canada and USA.
Note that only regions from those countries are in the list below.
From the Regions list select BC and CA.
Those regions should appear on the report.

Exercise 11c - parameter that controls formatting in v11-v14

1) Open the Master and save it as Formatting Parameters
2) Add a new parameter called "Summary Choice".
3) Select "String" for the data type.
4) Click where it says: "Click here to add Item" in the middle window
5) Type the word "Detail" on the first line.
6) Click on the second line and type the word "Summary"
7) Locate the option called "Prompt Text" in the bottom window.
8) Click next to it in the second column under the word "Setting".
9) Type instructions for the user: "Select Detail or Summary".
10) Locate the option called "Allow custom values" in the same window.
11) Change this to "False"

12) Click OK to save the parameter.

Use the menu options **Report ➡ Section Expert**.
Highlight the Details section on the left
Click the condition button next to the Suppress property.
Enter the formula below:

```
{?Summary Choice} = "Summary"
```

Make sure that you spell and capitalize "Summary" just as you did in the prompt's default.
Save and close the formula to get back to the section expert.

Highlight the Group Header section and click the condition button next to Suppress.
Enter the same formula as above and save/close.
Click OK to close the Section Expert.
Refresh and prompt for new.
You will have a new prompt that asks if you want the report in detail or summary.
Run the report in Summary, and not that it prints one line per group.
You can use this same formula to suppress the headings that are not needed in a summary.

Switch to design and delete the Customer ID column
Add the field ShipVia to the detail band.
To control the sort with a parameter field:

1) Add a new parameter called "Sort Choice".
2) Select "String" for the data type.
3) Click where it says: "Click here to add Item" in the middle window.
4) Type the word "Date" on the first line.
5) Click on the second line and type the word "Amount"
6) Click on the third line and type the word "Ship Method"
7) Locate the option called "Prompt Text" in the bottom window.
8) Click next to it in the second column under the word "Setting".
9) Type instructions for the user: "Choose to sort by Date, Amount or Ship Method".
10) Locate the option called "Allow custom values" in the same window.
11) Change this to "False"
12) Click OK to save the parameter.

Add a formula field called "Sort Field" that says the following:

```
If {?Sort Choice} = "Ship Method"
then {Orders.Ship Via}
else
If {?Sort Choice} = "Amount"
then Totext({Orders.Order Amount},'#######.00')
else Totext({Orders.Order Date},'yyyy-MM-dd')
```

Note that the format string on the end of the last line is case sensitive.

Use the menu options **Report ➡ Record Sort Expert**
Add a sort using the new formula field, Sort Field.
Click the lightning bolt to refresh and "Prompt for new…"
Select Details and Sort on Amount.
Note that the report sorts by Amount.

Click the lightning bolt to refresh and "Prompt for new…"
Select Details again but this time Sort on Ship Via.
Note that the report sorts by Ship Via.

Exercise 12a - Parameter fields in versions 9/10

Open the Master Report, and save it as Parameters.

Enter a State parameter

1) Open the Field Explorer
2) Highlight the heading "Parameter Fields".
3) Click the button for "New".
4) Enter the name "State"
5) For prompting text enter "Please select a State, or All"
6) Select "String" for the data type.
7) Click OK.

Open the Selection Expert and delete the existing rules.
Click "New" to add a rule that says:
Region / is equal to / {?State}

{?State} should appear in the list of states in the select expert.

Click OK and then click the lightning bolt to refresh
Select, "prompt for new parameter values".
Enter the State "CA".
You should see only California records.
Click the Refresh Button and select "Prompt for new parameters"
Enter the state FL and click OK
You should see only Florida records.

Adding Default Values:

Switch to design mode and find the State parameter in the field explorer.
Right-click the State Parameter and select "Edit".
Click the button that says "Set Default Values"
At the bottom put a check next to the property called "length limit"
Enter 2 for the Minimum length and 3 for the Maximum length.

In the very top box of the window (Browse Table) select Customer.
In the second box (Browse Field) select Region.
Note the list of States and Regions that appears in the window.
Type the word ALL in the Default values box, just above the list of States.
Click the first arrow to move it to the list on the right.
Then click the double arrow pointing to the right to add all regions to the default list.
(Note that only regions that are 2 or 3 characters move over)
Click the "OK" twice to exit the parameter.

Click the refresh button and select “Prompt for new parameter values”.
Notice there is now a drop-down list of state abbreviations to pick from.
Drop down the list and select PA for Pennsylvania.
Click OK and note that only Pennsylvania records appear on the report.

Exercise 12b - Range and a Multiple Parameters in versions 9/10

Switch to design mode, and highlight the State parameter in the field Explorer.
Right-click on it and select Edit.
In the lower left add a check mark for “Allow Multiple Values”.
Click OK.
Click the lightning bolt to refresh and select “Prompt for new parameters”
Use the drop-down list to select CA.
Click the “ADD” button to add this value into the lower window
Also add CO and FL.
Click OK to run the report and note that you get these 3 regions.

Adding a Date Range:

1) Switch to Design Mode
2) Open the Field Explorer
3) Right-Click on the heading “Parameter Fields” and select “New”
4) Enter the name “Date Range”
5) For prompting text enter “Please select your date range”
6) Select “Date” for the data type.
7) In the lower left select “Range Values”
8) Click OK.

Use the menu options **Report ➡Selection Formulas ➡Record**
Add the following line below the existing selection formula:

```
and {Orders.Order Date} = {?Date Range}
```

Save and close the selection formula.
Click the lightning bolt to refresh and select “Prompt for new parameters”
Highlight the State parameter and use the drop-down list to select CA.
Click the “ADD” button to add this value into the lower window
Also add CO and FL.
Do NOT click OK yet.
Highlight the Date parameter and select a date range in 2001
Click OK to run the report and note that you get these 3 regions within this date range.

Switch back to Design mode.
Use the menu options **Report ➡Selection Formulas ➡Record**
Edit the Record Selection formula to be:

```
{Orders.Order Date} = {?Date Range} and
    ( if {?State} = "All" then True
```

```
else {Customer.Region} = {?State} )
```

Save and exit the selection formula
Click the lightning bolt to refresh and select "Prompt for new…"
Click "Add" once to add the ALL value for the State parameter.
Select any date range in 2001.
Click OK to run the report and note that you get all regions.

Entering Descriptions:

Switch to design mode and highlight the State parameter in the field explorer.
Click the Edit button to open the properties of this parameter.
Click the "Set Default Values" button.
Highlight any State abbreviation in the list and click "Define Description".
Enter the full State Name and click OK.
Repeat this for 2 or 3 states.
Refresh the report
Note that the parameter for State now shows the names of a few states.

Exercise 12c - Parameters that control formatting in versions 9/10

Open the master and save it as Formatting Parameters
Add a new parameter called "Summary Choice".
For prompting text enter "Select Detail or Summary".
Select "String" for the data type.
Click the Default Values button (v9/v10)
Enter 2 default values: "Detail" and "Summary"
Set the property "Allow Custom Values" to false (v11-v14)
Click OK to return to the main parameter window (v9/v10).
Uncheck the setting "Allow editing of default values" (v9/v10).
Click OK to close this parameter.

Use the menu options **Report ➡ Section Expert**.
Highlight the details section on the left and click the condition button next to Suppress.
Enter the formula below:

```
{?Summary Choice} = "Summary"
```

Make sure that you spell and capitalize "Summary" just as you did in the prompt's default.
Save the formula.

Highlight the Group Header section and click the condition button next to Suppress.
Enter the same formula and save it.
Click OK to close the Section Expert.

Refresh and "Prompt for new…"
You will have a new prompt that asks if you want the report in detail or summary.
Run the report in Summary, and note that it prints one line per group.

The same condition can suppress the column headings that are not needed in a summary.

Switch to design and delete the Customer ID and Region columns
Add the field ShipVia to the empty spot on the detail band
Add a new parameter called "Sort Choice".
For prompting text enter "Sort by Date, Amount or Ship Via".
Select "String" for the data type.
Click the Default Values button (v9/v10)
Enter 3 default values: "Date" , "Amount" and "Ship Via"
Set the property "Allow Custom Values" to false (v11-v14)
Click OK to return to the main parameter window (v9/v10).
Uncheck the setting "Allow editing of default values" (v9/v10).
Click OK to close this parameter.

Add a new formula field called "Sort Field" that says the following:

```
If {?Sort Choice} = "Ship Via"
then {Orders.Ship Via}
else
If {?Sort Choice} = "Amount"
then Totext({Orders.Order Amount},'#######.00')
else Totext({Orders.Order Date},'yyyy-MM-dd')
```

The date format string should be exactly as it is printed here, to work.
Also make sure you spell and capitalize the three words to match the parameter.

Use the menu options **Report ➡ Record Sort Expert**
Add a sort using the new formula field, Sort Field.

Click the lightning bolt to refresh and "Prompt for new…"
Select "Details" for the Summary Choice prompt and "Date" for the Sort Choice
Note that the report sorts by Date.
Click the lightning bolt to refresh and "Prompt for new…"
Select Details again but this time Sort on Amount or Ship Via.
Note that the report sorts by that field.

Exercise 12c – Editable Parameters and Interactive Sorting (v12-v14)

Right click on the Summary Choice parameter and select Edit.
Look at the first property "Show on (Viewer) panel" to confim that it is "Editable".
Click OK to exit the parameter and repeat this step for the Sort Choice parameter.
Use the View menu to open the "Preview Panel" (if it isn't already open on the left).
Click on the "Parameters" button at the bottom of the panel to show the parameters.
Each parameter should be showing its current selected value.
Click on the Prompting Text of a parameter to see it open and then close with each click.
Open the Summary Choice parameter and click on the selected value once, then twice.
It should convert to a drop down selector showing both choices.
Change the selected value and then click the checkmark in the preview panel tool bar.
The report will use the new selected value and go from detail to summary or vice versa.

Change the value back again by repeating the process.

Change the value again (if needed) so that the report is in detail mode.
Open the Sort Choice parameter and click on the selected value once, then twice.
It should, again, convert to a drop down selector showing the three choices.
Change the selected value and then click the checkmark in the preview panel tool bar.
The report will use the new selected value and resort the records.
Change the value again by repeating the process.

Use the menu options Report > Record Sort Expert .
Remove the sort on "Sort Field".
Add a primary sort by Order Amount and a secondary sort by Order Date.
Close the Record Sort Expert.
Right click on the column heading for Order Amount and select "Bind Sort Control".
Highlight the field Order Amount and click OK.
You should see up and down arrows next to the heading.
Move the column heading for Order Date slightly to the right to eliminate the overlap.
Right click on the column heading for Order Date and select "Bind Sort Control".
Highlight the field Order Date and click OK.
Again you should see arrows next to the heading (if not make the object narrower).
Click the "up" arrow next to Order Date and notice that the details are ascending by date.
Open the Record Sort Expert and notice that Date is now the primary sort.

Use the menu options Report > Group Expert (NOT the Group Sort Expert)
Add a second group using the field Ship Via (which isn't on the report yet).
Close the Group Expert.
Use the menu option Insert > Sort Control to open the list of group/sort fields.
Highlight Group # 2: (Ship Via) and click OK.
Drag the cross-hairs (plus sign) to draw a small text frame.
When you release, type the words "Ship Via" in the box and click outside the box.
Click the down arrow to put Group 2 in descending order
Click the up arrow to put it back in ascending order

Exercise 13 - SubReports

Creating a report of the Top 5 selling product:

Create a new report from 3 tables, Orders, Orders Detail and Product.
Link the tables and select the fields Product Name and Quantity.
Group on Product Name.
Create one summary that Sums the Quantity for each Product.
On the "Group Sorting" select Top 5 and then click Finish.
Switch to Design mode and suppress all sections except for the group footer.
You should see just 5 lines, with 5 products and their subtotals.
Save the report as Top 5 and close it.

To add an Unlinked subreport of the top 5 Products:

Reopen the Master Report and save it as container.
UnSuppress the Report Header and make it 1/2 inch deep
Use the menu options **Insert ➡ Subreport**
Select the top option "Choose a report" and click Browse.
Locate Top5.rpt and click Open.
Click OK, then drag the subreport object into the report header and click there.

Click on the Preview tab to see the Subreport at the top of the main report.
Click on the new tab at the top of the window called "Top5".
This is the design tab of the subreport.
Open the Group Expert and notice the group is by Product Name.
Click on the original Design tab
Open the Group Expert and notice the group is by Customer Name.
Save the Container Report.

To make this a Linked subreport showing each customers top 5 products:

Make the group Header 1/2 inch deeper than it is.
Move the subreport to the new space in the Group Header.
Preview and note that currently the subreport simply repeats the same numbers.
Switch back to Design mode and right-click on the subreport
Select the option that says "Change Subreport Links".
Select Customer ID from the "Available Fields" list on the left.
Click the arrow to move it to the right.
Note the subreport parameter selection window in the lower left.
It will have a default parameter called "?PM-Customer ID"
The "Select Data" window on the lower right should say "Customer ID".
Click OK and preview.
Note that each subreport has different products and different numbers.
Save the Container report.

Save the report again under the name "On Demand"
Switch to Design mode and click on the subreport in the group footer
Use the menu options **Format ➡ Subreport ➡ (Subreport Tab).**
Check the option "On Demand Subreport".
Click OK and preview the report.
Note that the subreports are just hyperlinks.
Click any hyperlink to open that one subreport as a drill-down window.
Use the red 'x' to close the drill-down window.

Switch back to Design mode.
Use the menu options **Format ➡ Subreport ➡ (Subreport Tab).**
Click the X+2 button next to the On-Demand Subreport Caption
Enter the following formula:

"Click here to see the top 5 products for " + {Customer.Customer Name}

Save and close
Click the X+2 button next to the Subreport Preview tab Caption
Enter the following formula:

{Customer.Customer Name} [1 to 10]

Preview and note that you see your captions for each subreport.
Click one of the captions to launch that one subreport
The preview tab no longer is the name of the subreport, but has your 2nd caption.

Exercise 14 - Cross-Tabs

Reopen Master.rpt and save it as CrossTab.
Unsuppress the report header and make it an inch deep.
Use the menu options **Insert ➡ Cross-Tab.**
Drag the Order ID field into the Summarized Fields box.
Use "Change Summary" to change this from Sum to Count.
Drag the Country field into the Row box from the Customer table.
Drag the Ship Via field into the Column box from the Orders table.
Click OK, and place the cross-tab in the Report Header.
Note that it prints once, and includes all records for the report.

Click in the empty upper left corner of the cross-tab.
Use the menu options **Report ➡ Group Sort Expert.**
Set the report to be the Top 5 Countries, and include others.
Click OK and preview. Note that only 5 Countries and others are listed.

Click in the empty upper left corner of the cross-tab.
Use the menu options **Format ➡ Cross-Tab Expert..**
Drag the Order Date field into the Row box, making it the second field.
Highlight Order Date and click Group options.
Change the middle setting to be "for each year".
Highlight Ship Via in the Columns box and click Group options.
Change from Ascending to Specified Order
Select the values FedEx, UPS
On the "Others" tab select, "put all the others…"
Click OK and preview the report.
Note that each Country is broken down by Year.
Note that only 3 shipping category columns appear.

Click in the empty upper left corner of the cross-tab.
Use the menu options **Format ➡ Cross-Tab Expert.**
Highlight Order Date in the row box and click the arrow to remove it.
Drag the field Order Amount into the Summarized Field box.
Note that it creates a Sum by default, because it is a numeric field.
Use the Change Summary to change this to Average.
Click OK and preview the report.
Note that there are now two summaries for each row of the Cross-Tab.

Click in the empty upper left corner of the cross-tab.
Use the menu options **Format ➡ Cross-Tab Expert.**
Highlight Country in the row box and remove it.
Click OK and preview the report.
Note that the report only includes the Total Row.

Switch to design mode.
Widen the Group footer by about one half inch.
Click in the empty upper left corner of the cross-tab.
Use the menu options **Edit ➡ Copy.**
Use the menu options **Edit ➡ Paste**
You will see a copy of the Cross-tab on your cursor.
Click in the Group Footer, under the group name, to paste the copy.
Preview and note that the original cross-tab still includes all records.
The copy prints after each customer, and only includes that customer's data.
Save this report under the name CrossTab.

Open the Master and save it under the name Manual Crosstab.
Delete the fields Customer ID, Region and Order ID (including totals).
Use the menu options **Report ➡ Group Expert** to change the group to Country.
Hide the Details and the Group Header.
Write a new formula called FedEx that says:

```
If  {Orders.Ship Via} = "FedEx"
then {Orders.Order Amount}  else 0
```

Place this formula on the Details Band.
Add a subtotal and grand total for the FedEx Formula.
Preview and you will see the first column of a manual cross-tab.

Exercise 14b – Adding Calculated Rows to Cross-tabs (v12-v14)

Open the Master report and save it under Cross-tab Calculations.
Change the Select expert to say:

Order ID / is between / 1800 and 2500
Country / is one of / USA
Canada

Go to design mode and insert a Cross-tab in the report header.
Right click in the upper left corner and put in the following settings:

Row Fields – Country (set to Descending order) and Ship Via
Summarized Field – Ship Via (will count by default)
Column Field – Order Date (set “Group Options” to “for each month”)

Your cross-tab should show two countries with each subdivided into six shipping methods.
We will add a new row that is the sum of FedEx and UPS, and label it "Express Delivery".

Right-click on the label "FedEx" within the USA group of the cross-tab.
Select "Calculated Member" > "Select USA -> FedEx as first value".
Click OK on the message.
Right-click on the label UPS heading within the USA group.
Select "Calculated Member" > "Sum of USA -> FedEx and USA -> UPS
A new row appears labeled "Sum" which is the sum of those 2 values.

Right-click on the word SUM.
Select "Calculated Member" > "Edit Header Formula".
Replace the word "Sum" with "All Express" and then save and close the formula.
Note the row is now labeled "All Express".

Right-click on the words "All Express".
Select "Calculated Member" then "Edit Insertion Formula".
Remove the middle part that says:

```
and GridRowColumnValue("Customer.Country") = "USA"
```

Note that the "All Express" line now appears in both groups.
But, in both cases it is showing the total from the USA group.

Right-Click on the summary value just to the right of the "All Express" label.
Select "Calculated Member" > "Edit Calculation Formula".
Make the following two changes to make the formula relative to the current Country:

replace: `GetRowPathIndexOf("USA", "FedEx")`
with: `GetIndexOf("Orders.Ship Via","FedEx")`

replace: `GetRowPathIndexOf("USA", "UPS")`
with: `GetIndexOf("Orders.Ship Via","UPS")`

The "GetRowPathIndexOf" function specifies both levels of this two-level cross-tab, which returns one specific row no matter where you are. In contrast, the "GetIndexOF" function specifies only the level of the field named. When you leave a level unspecified in a multi-level cross-tab, Crystal will default to the current level. This returns a value relative to the cell doing the calculation. So by specifying "FedEx" in the last example I get the Fedex value within the current country, not the one from the literal country “USA”.

Save and close to see that the calculation is correct for both countries.

Creating a relative recurring column

Right-click on the column heading for the second month (August).
Select "Calculated Member > Insert Column" and click OK on the message.

Right click on the new empty heading cell.
Select "Calculated Member > Edit Header Formula".
Put in the literal value "Variance" and save and close.

Right-click on the cell below the heading cell (first summary).
Select "Calculated Member > Edit Calculation Formula".
Replace this formula with the function: `CurrentRowIndex.`

The column now shows the row number for each row.
Note that the first row is zero (not one).

Right-click on the same cell and select "Calculated Member > Edit Calculation Formula".
Change it the function to: `CurrentColumnIndex.`
The new column is number two, which is the third column when you start with zero (0, 1, 2).

Right-click on the word variance in the top cell and select
"Calculated Member > Edit Insertion Formula.
Change that formula to read:

```
CurrentColumnIndex > 0
```

Save the formula.
The new column will be to the right of every column except column 0 (the first column).

Right-click again on the first summary field below the heading.
Select "Calculated Member > Edit Calculation Formula".
Put in the following formula (Note that lines 1, 3 and 4 are very similar):

```
GridValueAt (CurrentRowIndex, CurrentColumnIndex -1  , CurrentSummaryIndex )  -
if CurrentColumnIndex = 2
then GridValueAt (CurrentRowIndex, CurrentColumnIndex -2 , CurrentSummaryIndex )
else GridValueAt (CurrentRowIndex,CurrentColumnIndex -3 , CurrentSummaryIndex )
```

The first line of the formula specifies the value from the current month.
It uses the current row but uses the column one place to the left (index minus one).
We need an IF-THEN to get the prior month.
If is two columns to the left in August but it is three columns to the left in all of the other months. Note that the CurrentColumnIndex has minus 2 and minus 3 in the then and else lines.
Save and close.

Go back into the last formula and make a copy.

Insert the exact same formula into the "Calculation Formula" for the second summary field. Because it uses the CurrentSummaryIndex it knows to now use the bottom number of each cell.

Exercise 15 - Split Sections

Reopen the Cross-tab report and save it under the name Subsection.
Widen the Report Header by an additional ½ inch.
Insert a 2nd cross-tab below the first.
Use a row field of Ship Via / summarized field Amount (Sum).
Place this in the report header just below the first cross-tab.
Preview and note that the first cross-tab overwrites the second.

Switch back to Design and use the menu options **Report ➡ Section Expert.**
Highlight the Report Header and hit the "Insert" button at the top of the window.
When you see both Report Header A and Report Header B, click OK.
Move the second cross-tab into Report Header B, and reduce the size of A.
Preview and note that they do not overlap.

Switch back to Design and use the menu options **Report ➡ Section Expert.**
Highlight Report Header A and hit the Down arrow at the top of the window.
Click OK, preview, and note that the cross-tabs have reversed position.

Switch back to Design and use the menu options **Report ➡ Section Expert.**
Highlight Report Header A and click on "Delete" at the top of the window.
Click OK, and note that one of the cross-tabs is gone.
Click Undo to bring the section back.

Use the menu options **Report ➡ Section Expert.**
Highlight Report Header A and hit the Merge button at the top of the window.
Click OK, and note that both sections have been combined into one
Click Undo to split them back into separate sections.

Exercise 16 – Adding Charts to a Report

Open the Master and Save it as Charts
Open the Select Expert and add a rule that Country = Canada

Use the menu commands **Insert ➡ Chart** to open the Chart Expert.
(Note - in v11-v14 you must place this chart in the Report Header first, then right-click on the chart to open the chart expert.)

On the "Types" tab select the default bar chart
On the Data tab set the options to the following:
- Put Graph - Once per Report - Header (only in v9/v10)
- On Change of - Customer.Customer Name
- Show - Sum of Orders Amount

Click OK to preview the report (simple bar chart)
Double click on a bar to drill-down to the details.
Close the drill-down window.

Single click on the chart to select it.
Use **Format ➡ Chart Expert ➡ Data tab**
Find the Advanced button on the left side and change to an Advanced Chart:
Leave the "Put Graph" set to Once per Report - Header (only in v9/v10)
Find the field ShipVia in the Orders table at the bottom of the field list
Highlight this field and click the ">" to make this the "On Change of" field.
Highlight Order ID and click the second ">" for the "On Change of" field.
Highlight "Sum or Order ID" and click "Change Summary"
Use the "Set Summary" button to change the operation to "Count".
Click OK to preview the report and note the chart is by Shipping Method.
Note that because this is an Advanced Chart there is no Drill-Down option.

Switch to Design mode
Add a Group Header 1B and make it as deep as the Report Header.
Copy the chart and past the copy into Group Header 1B.
Preview the report, and note a different chart for each customer.

Save the Charts report for the next lesson and close it.

Exercise 17 – Keep Together

Object:
Re-open the container report saved in lesson 13.
Preview and look at the bottom of the first few pages.
Locate a page with a large white space at the bottom.
(If there is none, page forward until you find a larger white space at the bottom.)
This is caused by the subreport's "keep together"
Switch to Design and click on the Subreport in the Group Footer.
Use the menu commands **Format ➡ Format Subreport.**
Select the "Common" tab and remove the check for "Keep Object Together".
Click OK and Preview. Note that some subreports will now split across pages.
Save and close the Container Report.

Section:
Re-Open the Charts report and save it under the name Keep Together.
Go into the Select Expert and remove the Country rule, and refresh the report.
Slide to the bottom of the page and page forward a few pages.
Note that some groups have GH1a separated from GH1b by a page break.
Use the menu options **Report ➡ Section Expert**
Highlight the section Group Header 1 Customer Name (above A and B).
Check the option marked "Keep Together" and Click OK.
Note that the 2 Group Header Subsections now print together on the next page.

Group:

Use the menu options Report ➡ Section Expert
Highlight Group Header 1B and click "Delete" at the top of the window.
Click OK and preview.
Note that several groups now split over the page break.
Use menu Report ➡ Group Expert ➡ Options (Button) ➡ Options (Tab).
Place a check mark into the property "Keep Group Together".
Click OK, and Preview
Note that groups will now leave white space if they can't fit on the page.

Exercise 18 – Underlaying a section

Reopen the Charts report and save it as Underlay.
Delete the following 3 fields from the details band:
Customer name, Customer ID, and Region.
Click on the chart, and drag its right edge to the 4 inch mark on the top ruler
Use the menu options **Report ➡ Section Expert**.
Highlight Group Header 1B and check "Underlay following section".
Note that the chart prints beside the details.

Exercise 19 - Mapping

Reopen the Master Report and save it under the name Maps.rpt.
Change the select expert to say: Country / is one of / USA –Canada.
Change the Group to be by Region instead of by Customer Name.

Use the menu options **Insert ➡ Map.**
On the Data tab set the options to the following:
Click "Advanced" on the left.
Place map – Once Per Report
Geographic Field - Customers.Region
On Change of - Customers.Region
Map Values - Orders.Orders Amount (it will add "Sum of")

On the Type tab, set the options to the following:
Number of Intervals - 5
Distribution Method – Equal Count
Color of Highest Interval - Dark Red
Color of Lowest Interval - White

Preview the map.
Notice the legend on the right, showing the meaning of each color.
The gray areas (Canada and Mexico) are not part of the analysis area.
The states that are light green have no records in the data.

Select the menu options **Map ➡ Zoom In.**
Click on the Map, each Click zooms you in closer.
Select the menu options **Map ➡ Zoom Out.**
Click on the Map, each Click zooms you further out.
Select the menu options **Map ➡ Pan.**
Click and drag the map to the right, your view will shift to the West.
Note the "Map Navigator" window in the lower right corner.
The frame within the window is your current map view.
Drag this frame to one side and you will change the part of the map you can see.

Select the menu options **Map ➡ Resolve Mismatches.**
Click on the Resolve Mismatches tab
Notice that US States are matched, but Canadian provinces are not Mapped.
Map all of them to the State of Washington (WA).
Preview and note the color or WA changes from light to very dark.
Save this report.

Exercise 20 - Exporting

Continue with the Map report from above
Hide the Group Header and Details.
Preview and save the report under the name Export.
Export to Excel format:

File ➡ Export
Select "MS Excel" as the Format setting.
Select "Application" as the Destination setting.
Click OK and remove the check mark for "Export Page Header …"
Click OK and note that the spreadsheet looks like the report.

Export to HTML format:

File ➡ Export
Select "HTML 4.0" as the Format setting.
Select "Application" as the Destination setting.
Click OK and note that it creates both a folder and a file
Click OK and note that the page looks like the report.

Export to a CSV format:

File ➡ Export
Select "CSV" and "Application"
Click OK and keep the defaults.
CSV is sometimes opened in Excel.
Note that the column headings and page number are in each row of the list.

Switch back to Crystal and switch back to design mode.
Suppress all sections expect the group footer.
Export again to CSV and note that only the subtotals are exported.

Exercise 21 – Automatic Running Totals

Re-open the report Master.rpt and save it under the name Running Totals.
Delete the fields Customer ID and Region.
In the Field Explorer, highlight the **Running Total** group.
Right-click and select **New** to create a new Running Total
Name it "Running Count" in the top box.
Highlight the field "Order ID" on the left and click the first arrow button.
Set the operation as "Count" and click OK.
Place the field on the detail band and preview page 1 of the report.
Note that it counts one for each record.
Jump to the last page.
The Running Count on the last detail line matches the grand total Count.

Switch back to Design Mode
In the Field Explorer, highlight the **Running Total** group.
Right-click and select **New** to create a new Running Total
Name it "Running Sum" in the top box.
Highlight the field "Order Amount" on the left and click the first arrow button.
Set the operation as "Sum" and click OK.
Place the field on the detail band and preview page 1 of the report.
Note that it accumulates each Amount.
Jump to the last page.
The Running Sum on the last detail line matches the grand total Sum.

Open the Group Sort Expert in the Report menu.
Set the report to only include the Top 10 Groups, with no Others.
Preview the report and go to the last page.
Note that the report is shorter but the original Grand Totals did not change.
Note that your running totals accurately reflect the shorter report.
Re-open the Group Sort Expert in the Report menu.
Change it back to No Sort.
Preview the report and go to the last page.
Note that all of the totals now agree.

Open the select expert and click "New" to add a new rule.
Locate the subtotal labeled:

Group #1: Customer Name - Sum of Order Amount

Add a rule that says:

Sum of Order Amount / is greater than / 35000

Note that the report gets shorter, but that the original Grand Totals don't change.
Note that your running totals accurately reflect the shorter report.

Right-click on Running Count and select “Edit”.
In the bottom section, set the reset to “On Change of Group”
It should default to “Group #1 Customer.Customer Name.
Click OK and preview the report.
The running total is now a Running Subtotal, starting over with each group.
Do the same thing with the Running Sum to make them both running subtotals.

Delete both running totals from the Details band.
Insert the database field Last Years Sales onto the detail band.
Add a grand total of this field.
Note that it is grossly inflated, because it is counting all of the duplicates.
Right-click in the running totals area of the Field Explorer and select New
Name this one LYS Total.
Move over Last Years Sales as the field to summarize
Leave the operation as Sum.
Set the “Evaluate” setting to be “On Change of group” using Group #1.
Click OK and place this new field on the Details Band.
Note that it only changes on the first record of each group.
Place this on the Report Footer and it will provide an accurate total of the LYS.

Exercise 22 – Variable Basics

Re-open the report Master.rpt and save it under the name Variables.
Eliminate the “Suppress” on the Report Header.
Sort the details based on the Order ID.
In the Select Expert, delete the rule for Order ID and add a rule that says:

Region / is Equal to / PA.

Delete the fields Customer ID and Region.
Insert a formula called Assign with the following formula:

```
WhilePrintingRecords;
CurrencyVar Open;
Open := {Orders.Order Amount}
```

Insert a formula called Display with the following formula:

```
WhilePrintingRecords;
CurrencyVar Open
```

Place Assign on the Report Header and Display on the Details.
Note that the Display is the same on every page.

Move Assign to the Page Header.
Note that the Display values change on each page.
Place Assign on the Group Header.
Note that the Display values change for each group.

Place Assign back on the Report Header
Change the formula of Display to be:

```
WhilePrintingRecords;
CurrencyVar Open;
{Orders.Order Amount} - Open
```

This calculates the difference between the first "price" and the current "price".

Exercise 23 - Running Totals with Variables

Running Grand Total:

Switch to design mode.
Remove the Assign and Display formulas from the layout.
Create a new formula called Rebate:

```
if Sum ({Orders.Order Amount}, {Customer.Customer Name}) > 45000
then {Orders.Order Amount} * .05
else 0
```

In other words, if the customer's total is over 45,000 he gets a 5% rebate.

Place this formula on the Detail band and try to Insert a Grand Total.
To total this formula requires a running total using variables.

Insert a formula called "Accum" with the following formula:

```
WhilePrintingRecords;
CurrencyVar RunSum;
RunSum := RunSum + {@rebate}
```

Place this formula on the Detail band so that it fires on every record.

Edit the Display formula to say:

```
WhilePrintingRecords;
CurrencyVar RunSum
```

Insert Display onto the Report Footer and preview.
You have a grand total of the Rebates.
Delete the Rebate formula from the layout and note the same Display value.
Delete the Accum formula from the layout, and the Display value is 0.
Hit UNDO twice to return Accum and Rebate back to the Detail band.

Running Subtotal:

Insert a formula called "Reset" with the following formula:

```
WhilePrintingRecords;
CurrencyVar RunSum;
RunSum :=0
```

Place Reset formula in the Group Header.
Move the Display formula on the Group Footer.
Preview and note that the display shows a Subtotal of rebate for each customer.
Controlling the frequency of accumulation
Delete the field Reset from the layout.
Place Display back on the Report Footer.
The running total is a grand total of rebates again

But, what if only a customer's first order was eligible for rebate.
Move Accum to the Group Header.
Now it will only pick up rebate from the first record of each customer.

Exercise 24 - Using SQL Expressions

Re-open the master Report and save under the name SQL Expression.
Add a new formula called "Left Initial" using the following formula:

```
Left ( {Customer.Customer Name} , 1 )
```

Add a new formula called "SubString Initial" using the following formula:

```
{Customer.Customer Name} [1]
```

Now use the menu options **Insert ➡ Field Object**.
Highlight the heading "SQL Expressions" and click "New" (sparkle in V8).
Give the expression the name "SQL Initial"
From the function list open the category "String" and double click "Left".
Now double click the field Customer Name and add the number 1 so that it reads:

```
LEFT(`Customer`.`Customer Name`,1 )
```

Note the syntax changes. (If you ever need to type the ` character look under the tilde ~.)
Open the Select Expert and add a new rule:

Left Initial / is equal to / B.

Preview the report and notice that only Customers that start with "B" are included.
Use the Menu Options **Database ➡ Show SQL Query** to open the SQL window.
Note that the "Where" clause has a rule for dates, but not for this formula.

Open the Select Expert and delete the "Left Initial" rule. Add a new rule:

SubString Initial / is equal to / B.

Refresh and notice that, again, only Customers that start with "B" are included.
Use the Menu Options **Database ➡ Show SQL Query** to open the SQL window.
Note that the "Where" clause now includes the formula for Initial.

Open the Select Expert and delete the rule for SubString Initial.
Add a new rule that uses the SQL Expression "SQL Initial":

SQL Initial / is equal to / B.

Note that SQL expressions are listed with **%** in front of the name instead of the **@.**
Refresh and notice that, again, only Customers that start with "B" are included.
Use the Menu Options **Database ➡ Show SQL Query** to open the SQL window.
Note that both the "Select" and the "Where" clauses include the SQL Expression.

Exercise 25 – Creating and Using a Command

Reopen the Master Report.
Change the select expert to say:

Customer.Region = CA and
Orders.Order ID > 1000

Use **Database ➡ Show SQL Statement**
Copy the Statement and paste it into Notepad.
Close the report and start a new report.
At the "DATA" step, open the current connection to Xtreme.
Double click the option marked "Add a Command".
Paste the SQL statement into the box on the left and click OK.
When you click "Next" you will see the 6 fields from this SQL statement.
Move them all left to right.
Group by Customer Name.
On the Total step set up a Count of Order ID and a Sum of Order Amount.
Click Finish to see the report.
Note the select expert is empty, but you see records that meet the WHERE clause.

Open the database expert using **Database ➡ Database Expert.**
Right-Click on the command in the right window and select "Edit Command"
Click the Create button on the right to create a command parameter.
Fill in the 4 boxes as follows:

Order ID
Select a Minimum Order ID
Number
1000

Click OK
Highlight the number 1000 inside the SQL Query window.
Double click the Order ID parameter on the right to insert it into the SQL.

Click the Create again to create another command parameter.
Fill in the 4 boxes as follows:

State
Type a State
String
CA

Click OK
Highlight the letters CA in the SQL Query window (don't highlight the quotes).
Double click the State parameter on the right to insert it into the SQL.

Click OK and the prompt will appear.
Click OK again to return to the database expert.
Click OK again to return to the report.
Click the refresh button and select prompt for new parameters.
Enter the number 1500 and the State of PA
Notice that the report only includes those records.
Save this report under the name "Command".

Storing the Command in the repository
Open the database expert using **Database ➡ Database Expert.**
Right-Click on the command in the right window and select "Edit Command"
Place a check mark in the lower left corner next to "Add to Repository".
Click OK and give the command the name "Master".
In the Location box below, open the repository folder.
Right click on the "Command" folder and select "New Folder".
Enter the name "TEST" for the new folder.
Highlight that folder and click OK to put the Master command into this folder.

Exercise 26 – Saving a Report with Data

Open the master and save it under the name "Saved Data".
Notice that it opens on the design screen.
Preview the report by hitting the lightning bolt.
Click on the menu options **File ➡ Report Options.**
Put a checkmark on the property called "Save Data with Report".
Save the report, close it and reopen it.
You will notice that it immediately goes to preview.
Click on the menu options **File ➡ Report Options.**
Take out the checkmark on the property called "Save Data with Report".
Save the report, close it and reopen it.
You will notice that it opens in design mode.

Exercise 27 – Custom Functions

Open the Master Report and save it under the name Functions.
Change the Select Expert to say Customer / is equal to / Alley Cat Cycles.
Sort the report by Order Date, in Ascending Order.

Open the formula workshop.
Highlight the category for Custom Functions at the top and click "New".
Type in the Name FederalYear (no space in name) and click "Use Editor".
Enter the following:

```
Function  (dateTimeVar TransDate)
ToText ( Year (TransDate + 92) ,"0000")
```

Save the function.

Create a new formula called Federal Year.
Go to the bottom of the Functions list to find the Custom Functions.
Notice that the FederalYear() function is listed there.
Double click the function and then double click Order Date.
Save the Formula and place it on the report.
Note that it calculates the Federal Fiscal year, as a string, for each record.

Go back into the formula workshop and open the report Custom Functions.
Right-Click on the FederalYear function and select "Add to Repository".
Select the "Crystal Repository" option and click OK.
Note that it is now listed in the repository functions list.

Using the repository function in another report:

Reopen the Variables Report from the prior lesson.
Open the formula workshop and open the repository custom function list.
Right click on the FederalYear function and select "Add to Report"
Note how the function is listed in the Report Custom Functions.
Now create a new formula called Federal Year.
Double click the function and then double click Order Date.
Save the Formula and place it on the report, in place of Order ID.
Note that it calculates the Federal Fiscal year for each record.

Changing the Custom Function:

Go back into the formula workshop and open the Report Custom Functions list.
Right click on the function and select "Disconnect from Repository".
Then change the number 92 in the function formula to be 184 and save the function.
Right click on the function again and select "Add to Repository"
Click OK on the Crystal Repository and Click "Yes" to add the new version.
Highlight the Repository Function called FederalYear and note it now uses 184.

Exercise 28 – Reusing Objects with the Repository:

Create a new report using the Standard Expert.
At the data step, select the repository and use the Master Command.
At the fields step move all fields over and click finish.
You should see 148 records.
Switch to design and open the repository explorer (**View➡Repository Explorer**).
Open the images section and drag the Xtreme logo onto the page header.
Open the text section and drag the copyright text onto the page footer.
Note that you can't change either of these objects.
Right click on each and select "Disconnect from Repository"
Now the image can be resized and the text can be edited.

Right click on the resized logo and select "Add to Repository"
Give it the name Xtreme Logo, and highlight the images folder.
Crystal will warn you that you are going to update the existing image.
Click Yes

Right click on the edited text and select "Add to Repository"
Give it the name Copyright, and highlight the Text Objects folder.
Crystal will warn you that you are going to update the existing Text Object.
Click Yes

Open the database expert using **Database ➡ Database Expert.**
Right-Click on the Master command (on the right) and select "Disconnect".
Right-Click again on the Master command and select "Edit Command".
Change the > to be >=
Place a check mark in the lower left corner next to "Add to Repository".
Click OK, type in the name Master and select the folder called "Test".
Click "Yes" to update the existing command.