

Converting reports from Business Objects Crystal Reports to Oracle BI Publisher

An Oracle White Paper
February, 2008

Converting reports from Business Objects Crystal Reports to Oracle BI Publisher

Introduction	3
High level comparison of Crystal Reports with BI Publisher	3
Step-by-step conversion of Crystal Reports	4
1. Convert Connection and Data Model	5
1.1: Convert Connection to Data Source.	5
1.2: Convert Crystal SQL to BI Publisher Data Model.....	5
1.3: Convert Parameters	6
1.4: Convert Summary Fields, Running Total Fields, Built-in Functions, Custom Functions and Formula Fields.....	7
2. Convert Layout	9
2.1: Open a Blank RTF.....	9
2.2: Get Sample data	9
2.3: Create layout template.....	9
2.4: Preview and Upload Report.....	10
Crystal report conversion example	10
EmployeeSalaryReport.....	10
1. Analyze Connection and Data Model	12
1.1: Connection to Data Source.....	12
1.2: SQL query	14
1.3: Parameters.....	15
1.4: Summary Fields, Running Total Fields, Built-in Functions, Custom Functions and Formula Fields.....	18
Convert Connection and Data Model	19
1.1: Convert Connection to Data Source.	19
1.2: Convert SQL to Data Model.....	19
1.3: Convert Parameters	21
1.4: Convert Summary Fields, Running Total Fields, Built-in Functions, Custom Functions and Formula Fields.....	24
2. Analyze Report Layout	26
Convert Report Layout	31
2.1: Open a blank RTF.....	31
2.2: Get Sample data	31
2.3: Create Layout Template.....	32
2.4: Preview and Upload Report.....	44
Conclusion.....	45

INTRODUCTION

Oracle Business Intelligence Publisher (BI Publisher), formerly known as Oracle XML Publisher, has been included as part of the Oracle Enterprise Business Suite for over four years, and has successfully evolved to meet a wide range of business document requirements. After its release as an independent product and option to the Oracle Application Server in 2006, many companies have implemented it as a stand-alone system to author, manage and deliver reports and critical business documents. In addition, BI Publisher is included in Oracle Business Intelligence Enterprise Edition, Oracle PeopleSoft Enterprise, and Oracle JD Edwards EnterpriseOne as a way to easily create, generate, and maintain highly formatted documents. Now that the benefits of this product have been proven, there is a lot of interest in how to convert reports from different reporting tools into BI Publisher. This paper provides a step-by-step approach and an example on how to manually convert reports from Crystal Reports® into Oracle BI Publisher reports.

HIGH LEVEL COMPARISON OF CRYSTAL REPORTS WITH BI PUBLISHER

Oracle BI Publisher, with its unique architecture of separating the data model from layout enables multiple layouts per report and reduces the time and cost to create and manage reports. Crystal Reports combines data query and layout definition into a single .RPT file and a new report has to be created every time a requirement for new layout arises even if another report already has the same data.

Oracle BI Publisher leverages familiar desktop applications – Microsoft Office Word, Adobe Acrobat, and Microsoft Office Excel to create layouts, enabling non-technical users to comfortably create layouts for their reports. Crystal Reports has a proprietary developer-oriented tool to create reports.

With the use of XML Localization Interchange File Format (XLIFF), an open standard technology for language translation, Oracle BI Publisher supports up to 185 languages and 244 territories. Users do not need to install Oracle BI Publisher in a specific language to build or view a report in that language. Crystal Reports has limited language

support (11 languages) and users may have to install separate report servers for each language they want to support.

Oracle BI Publisher supports multiple delivery channels – email, ftp, secured ftp, printer, fax, webDAV, AS2, CUPS server. Crystal Reports only supports email, ftp, printer and Inbox delivery channels.

Despite the differences in architecture, design, translation and delivery, Oracle BI Publisher enables you to create documents and reports equivalent to those you can create with Business Objects' Crystal Reports.

STEP-BY-STEP CONVERSION OF CRYSTAL REPORTS

Note:

1. The sample reports used in this white paper were created in Crystal Reports XI version. Even so, these steps are also generally applicable to reports created with other editions and earlier versions of Crystal reporting tools.
2. Pre-requisites for converting reports are BI Publisher Template Builder and BI Publisher Enterprise Server. The Server and Template Builder can be installed on separate systems and downloaded from :

<http://www.oracle.com/technology/software/products/publishing/index.html>

3. Some of the steps assume that you have appropriate access to view Crystal Reports Designer Environment, Administrator level access to BI Publisher Enterprise Server, and Administrator or Developer level access to Template Builder.

We can separate the conversion steps into two phases:

1. Convert Connection and Data Model
 - 1.1. Convert Connection to Data Source
 - 1.2. Convert Crystal SQL to BI Publisher Data Model
 - 1.3. Convert Parameters
 - 1.4. Convert Summary Fields, Running Total Fields, Built-in Functions, Custom Functions and Formula Fields.
2. Convert Layout
 - 2.1. Open a blank RTF
 - 2.2. Get sample data
 - 2.3. Create Layout Template
 - 2.4. Preview and Upload Template

1. Convert Connection and Data Model

Crystal Reports supports various types of data sources. The following steps assume a database as data source.

1.1: Convert Connection to Data Source.

The first step in converting a report is to get the details of the data source for the report. In Crystal Reports Developer®, use the Data Explorer to view the database connection information. Crystal Reports Developer supports several data sources – ODBC, Oracle Server, XML, etc. Once you connect, the Data Explorer displays the database current connection, and on right click displays the properties option. The properties dialog shows the database type (ODBC (RDO) or Oracle Server etc.), Data Source Name or Service, User Id, and other related information. For ODBC (RDO) data source type, you need to look at the Windows ODBC Administrator and locate the DSN name to get the database server information. For Oracle Server data source type, you need to locate the service name in TNSNames.ora file to read the database server information. Crystal also supports JDBC (JNDI) data source type i.e., you can choose either a JDBC or a JNDI connection. The details of this connection are prompted from the Data Explorer or Database Expert.

To create a database connection in BI Publisher, logon and navigate to the Admin tab of Oracle BI Publisher Enterprise. Create a JDBC data source using the database detail that you captured. You can also create a JNDI data source by adding a JNDI service in the Server to connect to the database and then by referring to the JNDI name from BI Publisher JNDI set up.

1.2: Convert Crystal SQL to BI Publisher Data Model

Now that you know the data source, you need to investigate and convert the data logic.

In BI Publisher each report has a Data Model. A Data Model is composed of one or more Data Sets. Data Sets can be one of several types such as a SQL query, a call to a web service, or a reference directly to a file containing XML data. A Data Template is a type of custom Data Set that provides more sophisticated control over report data such as merging data from multiple queries, calling procedural routines, and defining a hierarchy for the resulting XML data.

In Crystal Reports Developer, the data logic definition is found in the Database Expert. Here it shows all the tables, commands or any other data source used in the report. Also the commands can be viewed in the Database Expert. The query which fetches data for the report can be viewed by opening the ‘Show SQL Query’ dialog under the ‘Database’ menu. This generally prompts for entering the parameters and the SQL query includes the parameter value.

In BI Publisher, you can add a new data set under data model and select data source type as SQL Query. Copy the SQL query from Crystal Reports 'Show SQL Query' dialog window and paste it in the text area of BI Publisher data set.

Crystal reports supports database function as a data source, where the data logic is defined in a function and the report simply calls the function. In such a case, you should examine the function to determine what the function does. A Function returning a single value can be handled by a SQL Query Data Set or a Data Template but a complex function is best handled by a Data Template. If the Crystal report uses REF CURSOR to retrieve a recordset, then you will need to use a Data Template. REF CURSOR as a return value of a function is supported by BI Publisher but if the function has a REF CURSOR as OUT or IN OUT parameters then you will need a wrapper function with a return type as REF CURSOR or you can modify the original function to return a REF CURSOR.

If the function contains a DML statement for data insert, update, or delete then you should split the function into two parts; one part to handle the DML statements and the other to handle the select query of the function. Any calculation or any DML statement that needs to be executed before the select query is executed should be done using beforeReport trigger. Similarly, any calculation or DML statement, which needs to be executed after the query is executed and XML data is received on the Server, should be done using an afterReport trigger.

1.3: Convert Parameters

Note: This is an optional step and is required only for reports that have parameters.

To find the parameters in Crystal Report, expand the Parameter Fields in the Field Explorer. There can be several parameters defined in a report, but only those with a check symbol are attached to the report. The details of each parameter can be viewed by right clicking on the parameter and selecting edit option. The Formula Workshop shows the role of these parameters in record selection.

Parameters in BI Publisher are defined in the Report Editor along with the Data Model. These parameters can be passed to the database query or can be passed to the layout for dynamic layout formatting.

Crystal Reports support parameters as input box, List of Values (LOV), range bound parameters, cascading parameters, date parameter using a date picker.

In BI Publisher, input box is defined in the report editor as parameters with parameter type as text. List of Values can be defined in the Report Editor as well; both static and dynamic list can be created. Dynamic LOV is defined by writing an SQL query.

For range bound parameters, you can define two separate parameters; one input box to enter the lower range and the other input box to enter the upper range. In case this range has to be displayed in the report output, these two parameters can be concatenated and displayed on the RTF template. Cascading prompt behavior can be achieved in BI Publisher by defining parameters and associating each of them with a List of Value. Since the List of Value gets data from a query the cascading relationship should be built at the SQL query level where one LOV binds with another LOV. On

the UI of Parameter definition page, there is a check the box for 'Refresh other parameters on change' which enables the cascading behavior. The date parameter with a date picker UI is also supported in BI Publisher.

Finally, to allow these parameters to associate with the report data, you will have to edit the report SQL query which you copied from Crystal. The parameters passed will become host variables in the report SQL query. In case of Data Template, the report SQL query will be treated the same way and additionally you will have to include the parameter names in the parameters section.

1.4: Convert Summary Fields, Running Total Fields, Built-in Functions, Custom Functions and Formula Fields

Summary Fields, Running Total Fields, Built-in Functions, Custom Functions and Formula Fields are very common in Crystal Reports. It is important to understand the role of these features in every report before converting them. Summary Fields and Running Total Fields are purely data calculation fields, but Built-in Functions, Custom Functions and Formula Fields may apply to Report layout formatting as well. Therefore, your first step should be to separate the Built-in functions, Custom functions and Formula Fields into data calculation function and layout formatting function categories. You need to evaluate whether these Formulae or Functions can be handled at the time of Data Extraction, if so they would fall into data calculation function category. If the formula field is a simple calculation then you can allow it to be handled on RTF template as well, but most of the complex formula fields, custom and Built-in functions should be converted into a PL SQL function. The formula fields and Custom or Built-in Functions that contribute to the layout formatting of the report would fall into layout formatting function category and they should be handled on the RTF template.

Summary Fields:

A Summary Field in Crystal Report is used to create summary data within the group footer or header. The summary Field allows you to create different types of calculation – sum, average, count etc. – on any report field. You can check the details of calculation by right clicking on the Summary Field and then selecting Edit Summary.

In BI Publisher, a Summary Field can be handled at the time of data extraction using Data Template, where you can define an aggregate function (SUM, AVG, COUNT, MIN, MAX) at group level in dataStructure section.

Also a Summary Field can be handled in the RTF template in BI Publisher by using the Insert Fields from toolbar or Menu bar of the BI Publisher Template Builder by selecting the calculation function from dropdown list and checking the 'On Grouping' checkbox. For the summary functions that are not available from the drop down list you will have to modify the code beneath the inserted placeholder in the RTF template.

Running Total Fields: The Running Total Fields in Crystal Reports are an advanced version of Summary Fields. It evaluates the running total at different levels – for each record, on change of field, on change of group, or based on a formula. Also it allows the user to reset the running total value at different levels – for each record, on change of field, on change of group, or based on a formula. Running Total Fields can be created on any database column or a formula field.

To handle Running Total Fields in BI Publisher you should use the concept described under title “Page-Level Calculations”, subtitle “Running Totals” in the BI Publisher Users Guide. Use the updateable variable. The Running Total Fields that are reset on change of group can be treated as Summary Fields and converted as described for Summary Fields on the RTF template. If the Running Total Fields is built over a Formula Field, then you can use a PL SQL function to define the Formula field. The PL SQL function, when included in SQL query, will return data as XML element that can be treated like any other XML elements for Running Totals. Another way to handle Formula Fields under Running Total Field is to write the formula using XSL or BI Publisher syntax on the RTF template.

Built-in Functions and Operators: There are several built-in functions and operators provided by Crystal Reports. Some of them already exist as an equivalent function or operator in BI Publisher; however, you will not find an equivalent function for several of them. You will need to identify the complex Built-in functions that are related to data calculation and convert them into PL SQL function. The simple built-in functions can be handled at RTF template using BI Publisher code syntax.

Custom Functions: Custom Functions can be found at three places within Crystal Reports Formula Workshop

- Report Custom Functions
- Repository Custom Functions
- Under Built-in Functions - Additional Functions (Plug-in Functions)

First check if an equivalent function is already available in BI Publisher. For example, Date, Number or Currency formatting etc. is already available in BI Publisher for which you may have written a custom function in Crystal Reports. If the function does not exist in BI Publisher then you should convert those functions that are related to data calculation into a PL SQL function and should be called with in the SQL query statement.

Formula Fields: The formula fields, as mentioned earlier, need to be evaluated and separated out into the data calculation and layout formatting formulae categories. The ones that fall into data calculation formulae category should be included as part of data model in BI Publisher. You can create PL SQL function to handle such formulae and

they will be processed at database level. The PL SQL function can be called from the SQL query and the corresponding calculated data will be part of the XML data.

It is a common practice to use custom Functions or Built-in Functions and Operators in a Formula Field for code re-use. The same code reusability approach can be used while converting them into PL SQL function.

See the user guide of Oracle BI Publisher
10.1.3.3.2 for details on supported features
related to layout design.

2. Convert Layout

2.1: Open a Blank RTF

To start layout design open a blank document in Microsoft Word.

Note: In general starting with blank RTF is easier but in some situations you might want to start with the editable RTF output from Crystal Reports and use it as the initial template for BI Publisher. This technique may be helpful when the editable RTF output file has not lost any formatting feature and the report has complex layout formatting. For dynamic column reports and cross-tab reports blank RTF is the best approach.

2.2: Get Sample data

Log on to the BI Publisher server from the Template Builder Add-in to Word and open the newly created report to load XML data for template design.

Alternatively, you can log in to the BI Publisher server, view the report output data and export the XML data. This XML data can be loaded from the BI Publisher Template Builder.

2.3: Create layout template

At a high level we can categorize a Crystal Report layout into three categories – layout design, calculations and formatting.

2.3.1: Layout Design

Layout Design includes identification of report format – tabular, form or free form, identification of data elements, hierarchy, grouping, sorting, and filtering. These contents acts as the backbone of the report layout design.

To build the data organization in BI Publisher, you can select a wizard in Template Builder. The ‘Table Wizard’ can handle most of the simple data organizations and the Table/Form wizard can handle some complex formats involving nested grouping. You can use the ‘Cross Tab’ wizard to build cross tab reports and ‘chart’ tool to insert various types of charts. The Insert Field dialog allows you to simply insert one field at a time. This can work along with the ‘Repeating Group’ feature in the Insert Menu of toolbar to create complex formats.

2.3.2: Calculations

Calculations refer to all the indirect data values which are obtained from a built-in function, Summary Field, expression, formula fields, custom functions or conditional display logic, Parameters, etc.

In BI Publisher, first check if an equivalent built-in function exists. Next you can check whether the layout formatting is available as Native Microsoft Word or RTF formatting feature. If so, just apply the desired formatting using equivalent Word or RTF formatting feature. If such a function does not exist then they need to be written as code in the Text Form Fields or directly on RTF page using either BI Publisher syntax or XSL syntax.

A sub-template can replace a complex formatting function or a re-usable formatting function. Similar to functions parameters can be passed to these sub-templates and variables can be shared between the main report and the sub-templates. These sub-templates can even replace the sub-reports used for condition based formatting.

2.3.3: Formatting

Formatting refers to the visual display of data elements, table size, table borders, table background, static text, images, background color, font size, font color, font style, alignment, header and footer contents etc. Since BI Publisher can leverage Microsoft Office Word for creating RTF templates, you have the access to all the formatting functionality available in Word.

2.4: Preview and Upload Report

At anytime while creating the RTF layout you can preview what the final report will look like in the desired output format. Navigate to 'Preview Template' under Oracle BI Publisher menu or Toolbar to view the report in PDF, HTML, RTF or Excel formats. Inspect the preview version of the report to check that you have replicated the layout and formatting you want in the report. If not, continue to modify and then preview until the report looks the way you want.

Once the template is complete, navigate to 'Publish Template As' under Oracle BI Publisher menu. Enter a template Name and click OK on the next prompt to complete the upload.

CRYSTAL REPORT CONVERSION EXAMPLE

EmployeeSalaryReport

This is a very simple report covering some of the basic features to outline the conversion steps. At a glance the output shows that the report data has been grouped by Department and then by Manager. Total salary is the sum of annual salary of employees reporting to a Manager and this is displayed as the last row of data under each Manager. The report uses a banded format with alternate rows shaded in grey background.

Department: Production
 Manager: Andrew Hill

EmployeeID	Name	Hire Date	Title	AnnualSalary
8	Ruth Ellerbrock	02/06/1998	Production Technician - WC10	\$32,549.00
10	Barry Johnson	02/07/1998	Production Technician - WC10	\$32,549.00
13	Sidney Higa	03/05/1998	Production Technician - WC10	\$32,549.00
15	Jeffrey Ford	03/23/1998	Production Technician - WC10	\$32,549.00
17	Doris Hartwig	04/11/1998	Production Technician - WC10	\$32,549.00
19	Diane Gimp	04/29/1998	Production Technician - WC10	\$32,549.00
230	Bonnie Kearney	02/02/2000	Production Technician - WC10	\$32,481.75
Total:				\$227,775.75

Manager: Brenda Diaz

EmployeeID	Name	Hire Date	Title	AnnualSalary
31	Alejandro McGuel	01/07/1999	Production Technician - WC40	\$36,300.00
45	Fred Northup	01/13/1999	Production Technician - WC40	\$36,300.00
56	Kevin Liu	01/18/1999	Production Technician - WC40	\$36,300.00
68	Shammi Mohamed	01/25/1999	Production Technician - WC40	\$36,300.00
81	Rajesh Patel	02/01/1999	Production Technician - WC40	\$36,300.00
91	Lorraine Nay	02/05/1999	Production Technician - WC40	\$36,300.00
105	Paula Nariker	02/13/1999	Production Technician - WC40	\$36,300.00
116	Frank Lee	02/18/1999	Production Technician - WC40	\$36,300.00
138	Brian Lloyd	03/02/1999	Production Technician - WC40	\$36,300.00
152	Tawana Nusbaum	03/09/1999	Production Technician - WC40	\$36,300.00
190	Ken Myer	03/28/1999	Production Technician - WC40	\$36,300.00
215	Gabe Mares	04/09/1999	Production Technician - WC40	\$36,300.00
Total:				\$435,600.00

Manager: Cristian Petculescu

EmployeeID	Name	Hire Date	Title	AnnualSalary
224	Betsy Stadick	01/19/2000	Production Technician - WC10	\$32,481.75
234	Kimberly Zimmerman	02/13/2000	Production Technician - WC10	\$32,481.75
245	Patrick Wedge	03/04/2000	Production Technician - WC10	\$32,481.75
252	Danielle Tiedt	03/23/2000	Production Technician - WC10	\$32,481.75
262	Tom Vande Velde	04/10/2000	Production Technician - WC10	\$32,481.75
Total:				\$162,408.75

Manager: Cynthia Randall

EmployeeID	Name	Hire Date	Title	AnnualSalary
33	Jian Shuo Wang	01/08/1999	Production Technician - WC30	\$22,990.00
73	Sandra Reategui Alayo	01/27/1999	Production Technician - WC30	\$22,990.00
110	Jason Watters	02/15/1999	Production Technician - WC30	\$22,990.00
142	Andy Ruth	03/04/1999	Production Technician - WC30	\$22,990.00
180	Rostislav Shabaln	03/23/1999	Production Technician - WC30	\$22,990.00
195	Michael Vandernyde	03/30/1999	Production Technician - WC30	\$22,990.00
Total:				\$137,340.00

Figure 1. Report output in PDF format using export option of Crystal Reports Developer.

Before starting the step-by-step conversion, it is important to understand the report and its design in Crystal Reports. For this you need to view the report in Crystal Reports Developer.


Report Header											
Page Header	 EMPLOYEE SALARY REPORT										
Group Header #1:	Department: Group #1 Name										
Group Header #2: @Manager - A	Manager: Group #2 Name										
Details	<table border="1"> <thead> <tr> <th>EmployeeID</th> <th>Name</th> <th>Hire Date</th> <th>Title</th> <th>AnnualSa</th> </tr> </thead> <tbody> <tr> <td>@EmployeeID</td> <td>@EmpName</td> <td>Hiredate</td> <td>Title</td> <td>@AnnualSa</td> </tr> </tbody> </table>	EmployeeID	Name	Hire Date	Title	AnnualSa	@EmployeeID	@EmpName	Hiredate	Title	@AnnualSa
EmployeeID	Name	Hire Date	Title	AnnualSa							
@EmployeeID	@EmpName	Hiredate	Title	@AnnualSa							
Group Footer #2: @Manager - A	Total: n of @AnnualSa										
Group Footer #1: Command.DepartmentName											
Report Footer											
Page Footer	Page										

Figure 2. EmployeeSalaryReport as seen in Design View of Crystal Reports Developer

The analysis of the report can be categorized into two sections – Report data (i.e. Connection and Data Model) and Report Layout.

1. Analyze Connection and Data Model

1.1: Connection to Data Source

In Crystal Reports Developer, navigate to ‘Log On or Off Server’ under the ‘Database’ menu to open the Data Explorer.

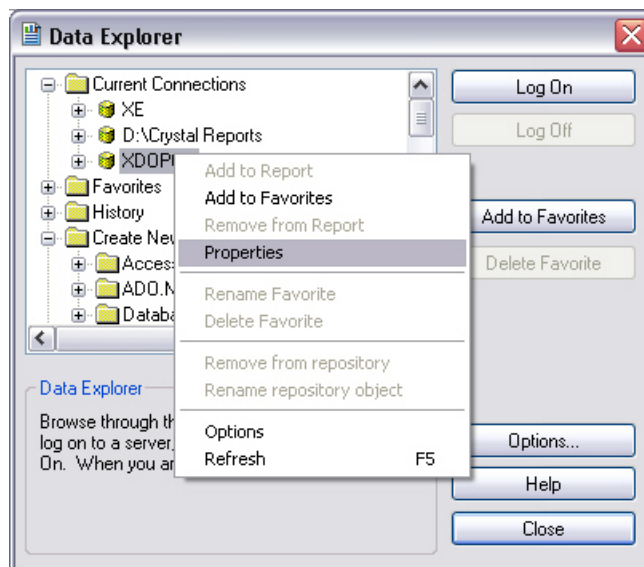


Figure 3. Data Explorer

Right click on the database connection associated with this report and select properties.

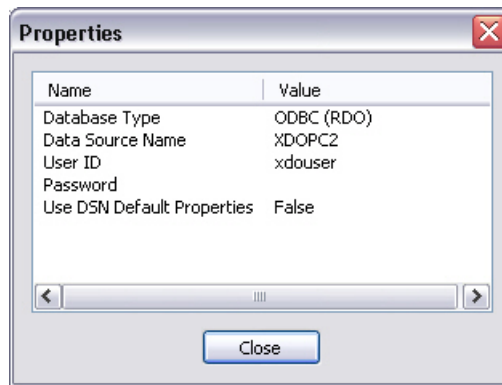


Figure 4. Database connection properties

From the database connection details, you know that this is an ODBC connection with DSN name as XDOPC2. So, next you need to open the ODBC Data Source Administrator and look for this DSN name.

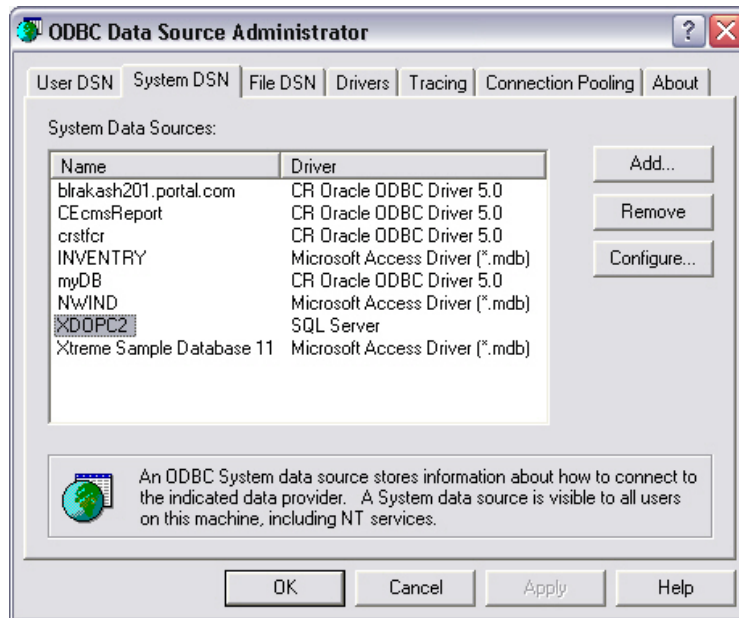


Figure 5. ODBC Data Source Administrator

Click on Configure button to check the details of the ODBC data source.

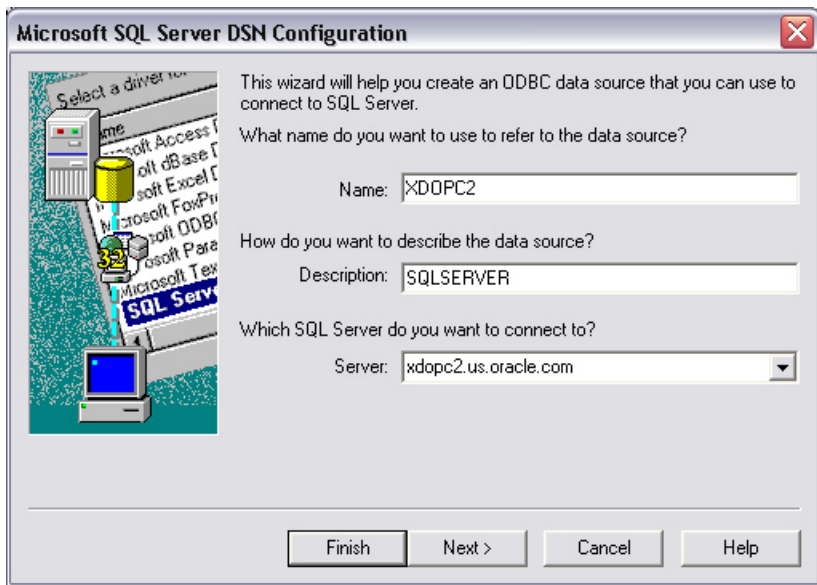


Figure 6. DSN configuration

The database in this example is xdopc2.us.oracle.com which is a Microsoft SQL Server database.

1.2: SQL query

Next, navigate to 'Show SQL Query' to check the query:

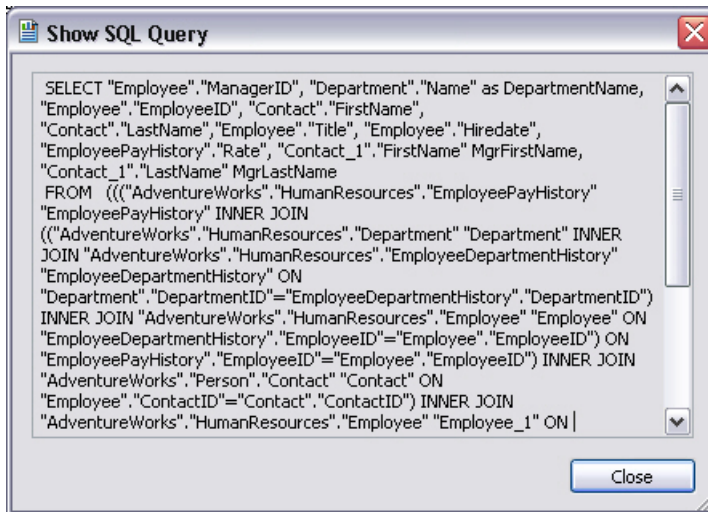


Figure 7. Show SQL query dialog.

The query as copied from Crystal is -

```
SELECT "Employee"."ManagerID", "Department"."Name" as
DepartmentName, "Employee"."EmployeeID",
"Contact"."FirstName",
"Contact"."LastName", "Employee"."Title",
"Employee"."Hiredate", "EmployeePayHistory"."Rate",
"Contact_1"."FirstName" MgrFirstName,
"Contact_1"."LastName" MgrLastName

FROM
(((("AdventureWorks"."HumanResources"."EmployeePayHistory
" "EmployeePayHistory" INNER JOIN
(("AdventureWorks"."HumanResources"."Department"
"Department" INNER JOIN
"AdventureWorks"."HumanResources"."EmployeeDepartmentHis
tory" "EmployeeDepartmentHistory" ON
"Department"."DepartmentID"="EmployeeDepartmentHistory".
"DepartmentID") INNER JOIN
"AdventureWorks"."HumanResources"."Employee" "Employee"
ON
"EmployeeDepartmentHistory"."EmployeeID"="Employee"."Emp
loyeeID") ON
"EmployeePayHistory"."EmployeeID"="Employee"."EmployeeID
") INNER JOIN "AdventureWorks"."Person"."Contact"
"Contact" ON
"Employee"."ContactID"="Contact"."ContactID") INNER JOIN
"AdventureWorks"."HumanResources"."Employee"
"Employee_1" ON
"Employee"."ManagerID"="Employee_1"."EmployeeID") INNER
JOIN "AdventureWorks"."Person"."Contact" "Contact_1" ON
"Employee_1"."ContactID"="Contact_1"."ContactID"

where "EmployeePayHistory"."ModifiedDate" = (select
max("EmployeePayHistory"."ModifiedDate") from
"EmployeePayHistory" where
"EmployeePayHistory"."EmployeeID"="Employee"."EmployeeID
")

ORDER BY "Department"."Name", "Employee"."ManagerID",
"Employee"."EmployeeID"
```

1.3: Parameters

In Crystal Reports Developer, check the Parameters in the Field Explorer. There is one parameter defined for this report - Department

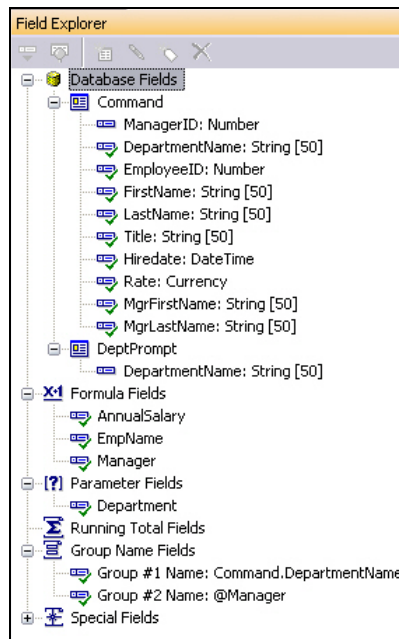


Figure 8. Field Explorer view displaying the Parameter for this report.

Right-click on the parameter and check the details of the parameter.

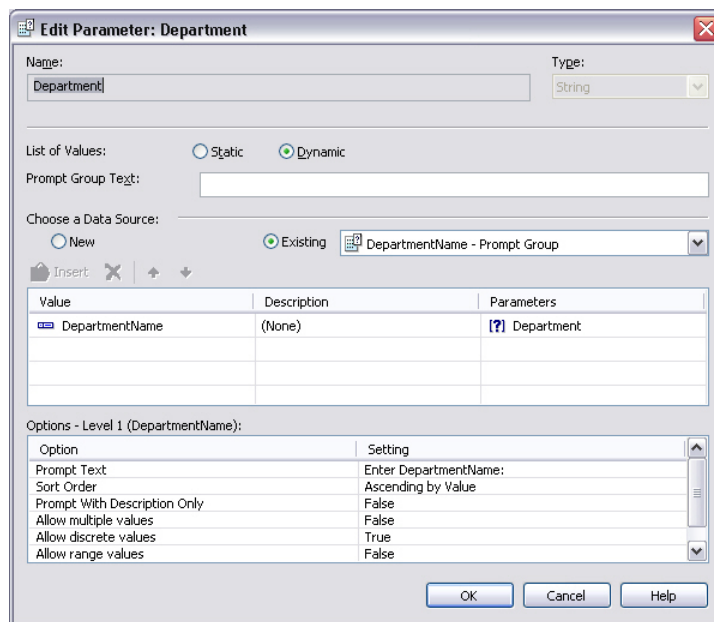
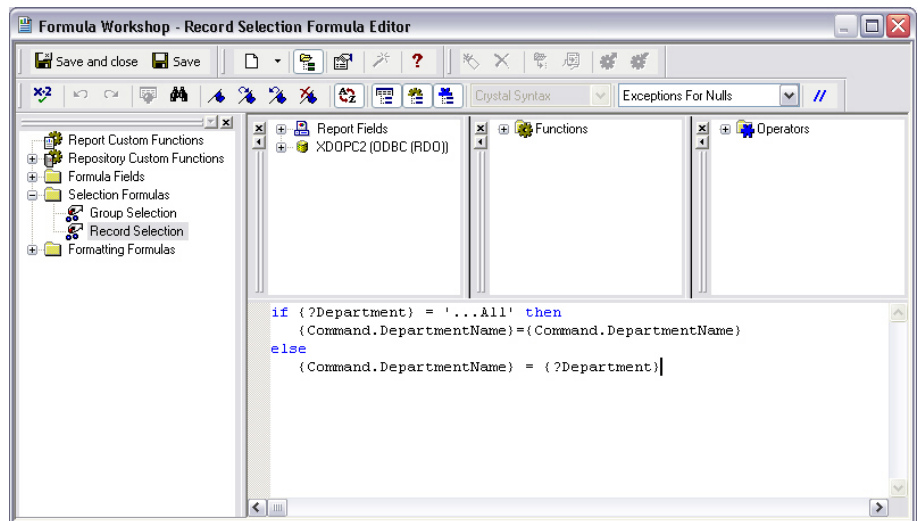


Figure 9. Definition of Department parameter

Next, take a look at the Record Selection Formula Editor to see how the parameter applies to the report. To view this, click on 'Report' Menu and then choose 'Record' under 'Selection Formula'.



The Department parameter checks for value '...All' which will be one of the values to select from a List of Values. In Crystal Reports '...All' can be added using a union in SQL query. This is defined under DeptPrompt as shown below:

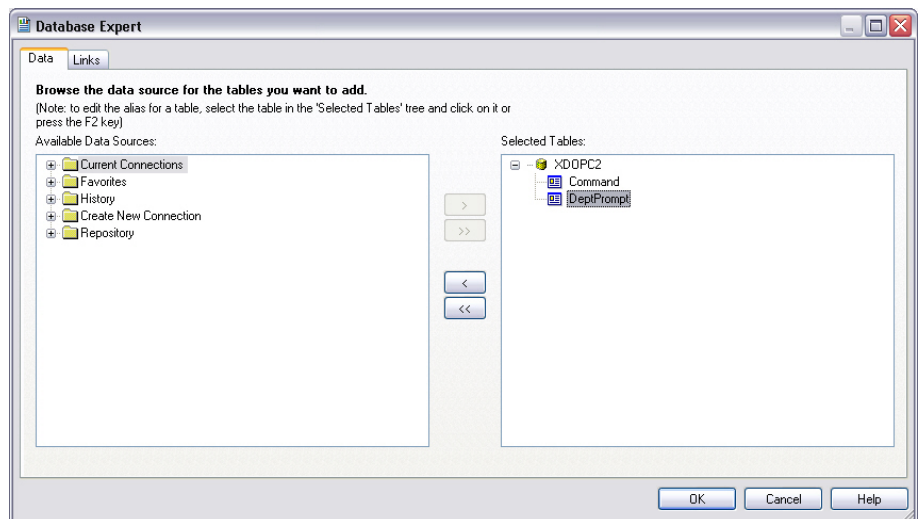


Figure 11. Database Expert showing DeptPrompt

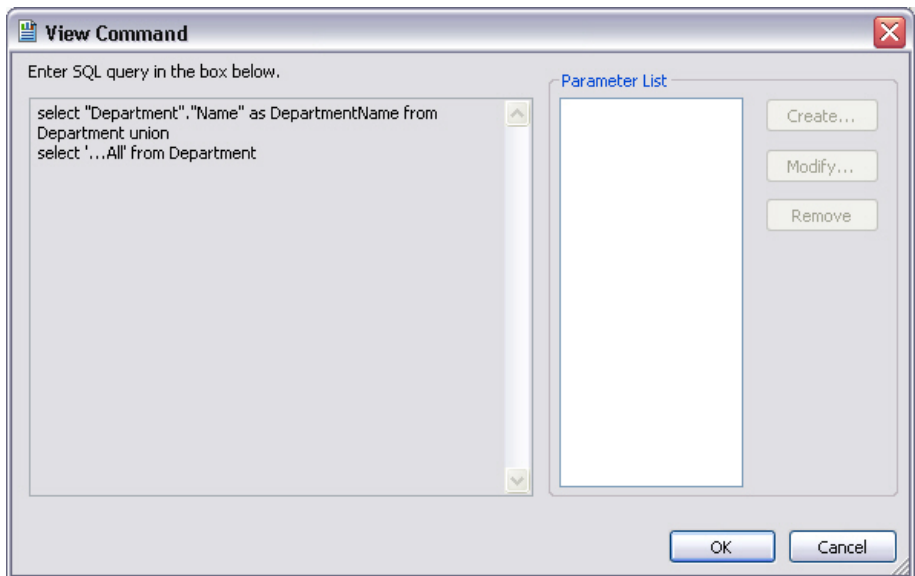


Figure 12. View Command displaying Query to add '"...All"' to the List of Value

1.4: Summary Fields, Running Total Fields, Built-in Functions, Custom Functions and Formula Fields

In Crystal Reports Developer, you can find all the formulae used in a report in the Formula workshop. This sample report has three formula fields (refer to Figure 8) – AnnualSalary, EmpName, and Manager. For demonstration we have used simple Formulae definitions.

EmpName and Manager Formulae are simply concatenation of first name and last name of Employee and Manager respectively. AnnualSalary formula field is a simple calculation of billing rate and employment duration for each employee.

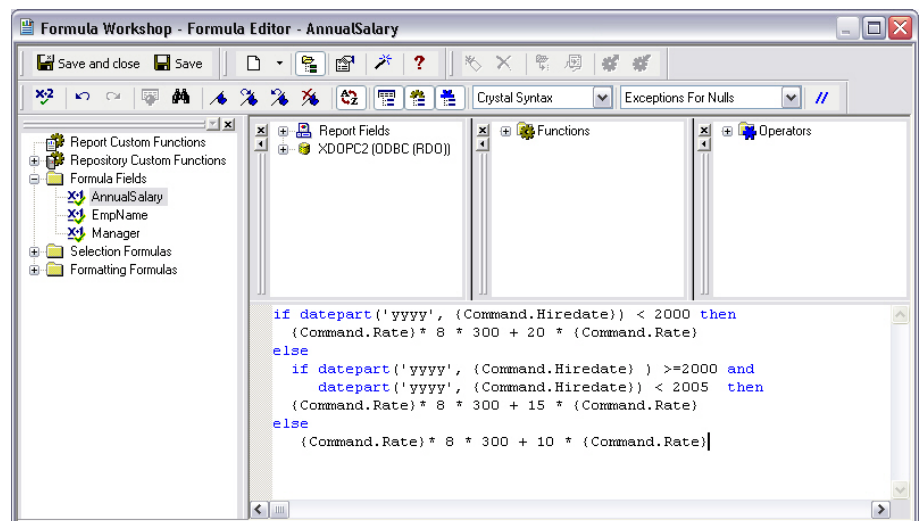


Figure 13. AnnualSalary Formula Field definition

Convert Connection and Data Model

1.1: Convert Connection to Data Source.

In BI Publisher, add a JDBC data source. Under Admin tab, select JDBC data source type and add a new data source.

```
Data Source Name: sqlserver
Connection String:
JDBC:microsoft:sqlserver://xdopc2.us.oracle.com:1433
Username : userid
Password : pwd
Database Driver Class:
com.microsoft.jdbc.sqlserver.SQLServerDriver
```

Note: For BI Publisher to connect to Microsoft SQL Server 2000, the following three jar files are required in the WEB-INF\lib folder:

```
msbase.jar
mssqlserver.jar
msutil.jar
```

To connect to Microsoft SQL Server 2005, `sqljdbc.jar` file is required in the WEB-INF\lib folder and the database driver class is:
`com.microsoft.sqlserver.jdbc.SQLServerDriver`

All the jar files listed above are available with the SQL Server distribution media or can be downloaded from Microsoft.com.

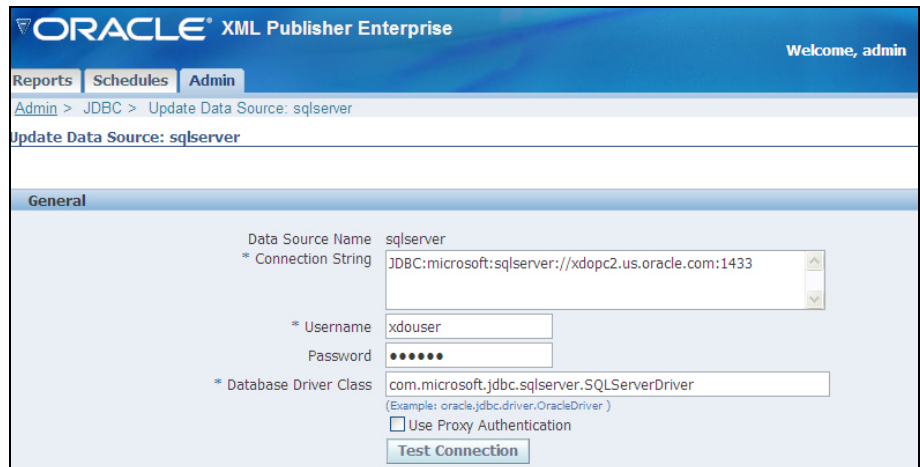


Figure 14. Defining JDBC connection to SQL Server in BI Publisher

1.2: Convert SQL to Data Model

In BI Publisher, create a new report and click on 'Edit' under the report name to view the report editor page. Create a new Data Set under Data Model section with a name,

say 'query'. Select 'SQL Query' as data set type and 'sqlserver' as the data source reference. Copy the SQL from the 'Show SQL Query' dialog of Crystal Reports Developer and paste it into the SQL Query window for the Data Set, and save the report.

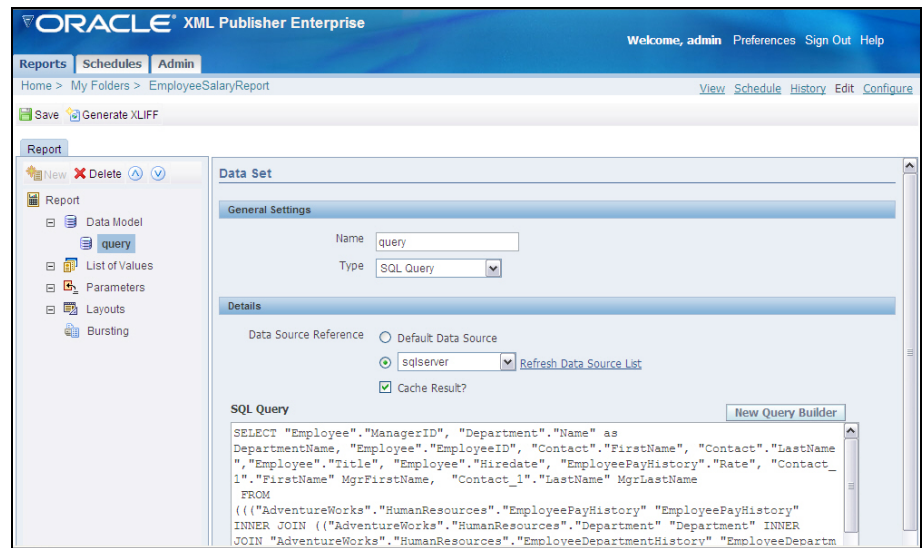


Figure 15. Defining Data Model in BI Publisher

Click on View link on the Report Editor page to verify that the data is generated from the SQL query. At this time, the only output format available to view would be Data.



Figure 16. Verify data in XML format.

1.3: Convert Parameters

First create a List of Values. Select List of Values in the left side Navigation of the Report Editor and click the New button at the top of the tree to create a new List of Values.

Enter a Name for the List of Values (e.g. deptLOV), select the 'sqlserver' data source and enter the SQL Query that returns the values. The query here will be:

```

Select distinct "Department"."Name"
from "Department"
order by "Department"."Name"

```

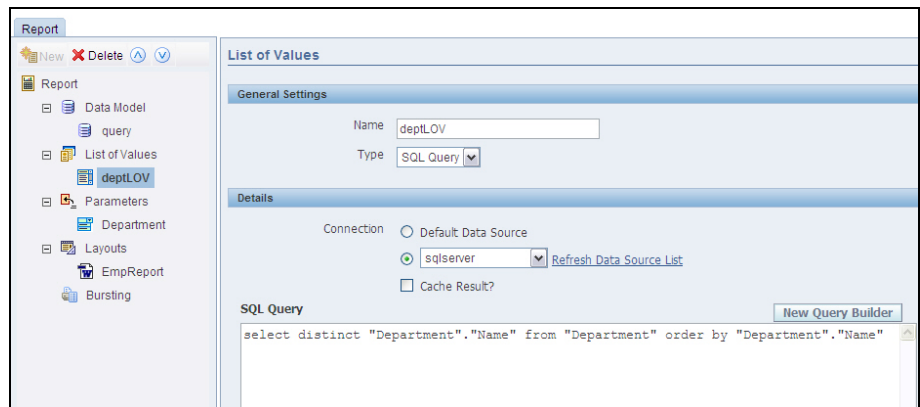


Figure 17. Defining List of Values for Department Name

Save the query on the Report Editor.

Next you can create a Parameter definition with name 'department' and associate it with the List of Value 'deptLOV'.

Select Parameters on the Left Panel of Report Editor and click on 'New' above the report tree to create a Parameter.

In the Parameter definition, enter 'Department' as the Identifier, Data Type of String, Default Value of '*' and select Parameter Type as Menu. This displays the List of Values you just created, i.e. deptLOV. Select deptLOV from the drop down list and check the option 'Can select all' with 'NULL value passed' as the choice. This configures the query to pass NULL value when 'All' is passed as parameter.

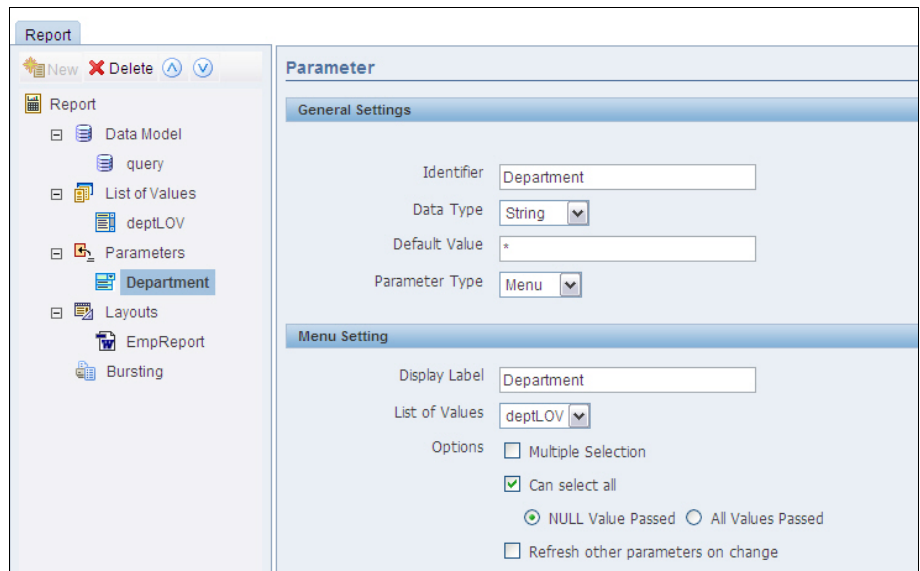


Figure 18. Defining parameter of Department

Click on 'Save' to save this parameter setting .

Next you need to associate the parameter with the SQL query that you defined in the Data Model. In the SQL query you need to retrieve data only for that department which is passed as parameter. The following condition has been added to the Where clause:

and "Department"."Name" = isNull(:department,"Department"."Name")

See the modified SQL below:

```
SELECT "Employee"."ManagerID", "Department"."Name" as
DepartmentName, "Employee"."EmployeeID",
"Contact"."FirstName",
"Contact"."LastName", "Employee"."Title",
"Employee"."Hiredate", "EmployeePayHistory"."Rate",
"Contact_1"."FirstName" MgrFirstName,
"Contact_1"."LastName" MgrLastName
FROM
(((("AdventureWorks"."HumanResources"."EmployeePayHistory"
"EmployeePayHistory" INNER JOIN
(("AdventureWorks"."HumanResources"."Department"
"Department" INNER JOIN
"AdventureWorks"."HumanResources"."EmployeeDepartmentHis
tory" "EmployeeDepartmentHistory" ON
"Department"."DepartmentID"="EmployeeDepartmentHistory".
"DepartmentID") INNER JOIN
"AdventureWorks"."HumanResources"."Employee" "Employee"
ON
"EmployeeDepartmentHistory"."EmployeeID"="Employee"."Emp
loyeeID") ON
"EmployeePayHistory"."EmployeeID"="Employee"."EmployeeID
") INNER JOIN "AdventureWorks"."Person"."Contact"
"Contact" ON
"Employee"."ContactID"="Contact"."ContactID") INNER JOIN
"AdventureWorks"."HumanResources"."Employee"
"Employee_1" ON
"Employee"."ManagerID"="Employee_1"."EmployeeID") INNER
JOIN "AdventureWorks"."Person"."Contact" "Contact_1" ON
"Employee_1"."ContactID"="Contact_1"."ContactID"
where "EmployeePayHistory"."ModifiedDate" = (select
max("EmployeePayHistory"."ModifiedDate") from
"EmployeePayHistory" where
"EmployeePayHistory"."EmployeeID"="Employee"."EmployeeID
")
and "Department"."Name" =
isNull(:department,"Department"."Name")
ORDER BY "Department"."Name", "Employee"."ManagerID",
"Employee"."EmployeeID"
```

IsNull function has been used here to handle 'All', i.e., whenever user Selects 'All' from List of Value, BI Publisher will pass NULL to the query, and in case of null the value is replaced by that of "Department"."Name", thereby removing the condition on Department Name.

Save the query on the report editor page.

To test that the Parameter and List of Values is working correctly, view the data to verify the XML output.

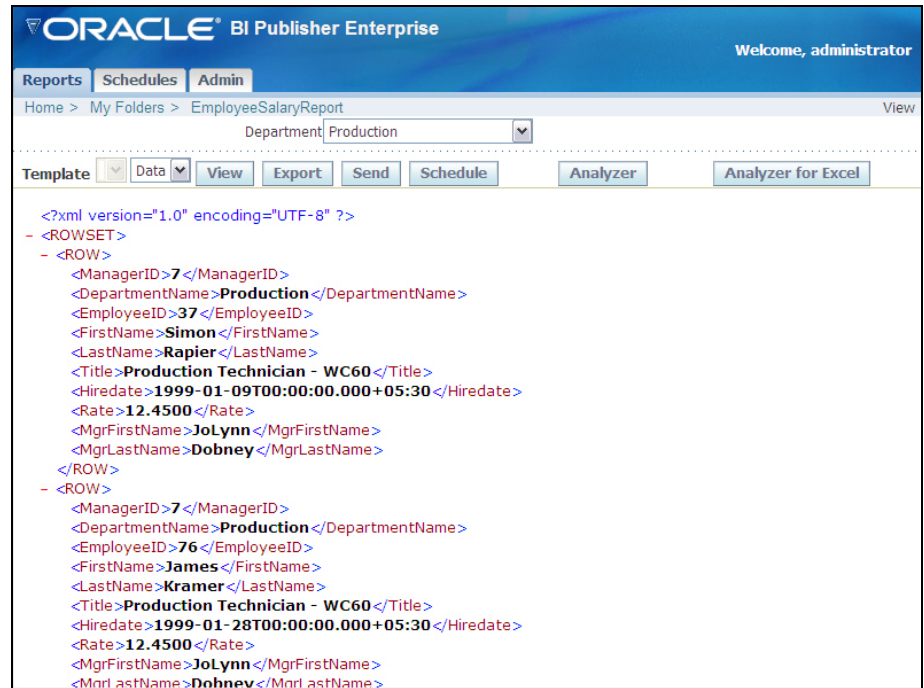


Figure 19. Data using List of Values as Parameter

A dropdown list will appear on the report view screen. On selecting a department from the dropdown list, you will view data for that department.

1.4: Convert Summary Fields, Running Total Fields, Built-in Functions, Custom Functions and Formula Fields

The Formula Field 'AnnualSalary' as defined in Crystal Reports Developer can be converted into a PL SQL function on the SQL server database as shown below.

```

CREATE FUNCTION f_AnnualSalary
( @Val1 float, @Val2 varchar(50) )
RETURNS float
AS
BEGIN
  declare @sal float
  if datepart(yyyy,@Val2) < 2000
    (select @sal= @Val1 * 8 * 300 + @Val1 * 20)
  else
    if datepart(yyyy,@Val2) >= 2000 and
datepart(yyyy,@Val2) < 2005
    (select @sal=@Val1 * 8 * 300 + @Val1 * 15)
  else
    (select @sal=@Val1 * 8 * 300 + @Val1 * 10)

```



```

RETURN (@sal)
END
go

```

Note: The above function definition complies with MS SQL Server syntax. The syntax can be changed accordingly to meet the requirements of other databases.

After compiling the function on the database, it can be called from the SQL query as shown below:

```

SELECT "Employee"."ManagerID", "Department"."Name" as
DepartmentName, "Employee"."EmployeeID",
"Contact"."FirstName" + ' ' + "Contact"."LastName"
EmpName,
"Employee"."Title", "Employee"."Hiredate" as
EmpHireDate, "EmployeePayHistory"."Rate" Rate,
"Contact_1"."FirstName" + ' ' + "Contact_1"."LastName"
Manager,
"AdventureWorks"."dbo".f_annualsalary("EmployeePayHistor
y"."Rate", "Employee"."Hiredate") as AnnualSalary
FROM
((("AdventureWorks"."HumanResources"."EmployeePayHistory
" "EmployeePayHistory" INNER JOIN
(("AdventureWorks"."HumanResources"."Department"
"Department" INNER JOIN
"AdventureWorks"."HumanResources"."EmployeeDepartmentHis
tory" "EmployeeDepartmentHistory" ON
"Department"."DepartmentID"="EmployeeDepartmentHistory".
"DepartmentID") INNER JOIN
"AdventureWorks"."HumanResources"."Employee" "Employee"
ON
"EmployeeDepartmentHistory"."EmployeeID"="Employee"."Emp
loyeeID") ON
"EmployeePayHistory"."EmployeeID"="Employee"."EmployeeID
") INNER JOIN "AdventureWorks"."Person"."Contact"
"Contact" ON
"Employee"."ContactID"="Contact"."ContactID") INNER JOIN
"AdventureWorks"."HumanResources"."Employee"
"Employee_1" ON
"Employee"."ManagerID"="Employee_1"."EmployeeID") INNER
JOIN "AdventureWorks"."Person"."Contact" "Contact_1" ON
"Employee_1"."ContactID"="Contact_1"."ContactID"
where "EmployeePayHistory"."ModifiedDate" = (select
max("EmployeePayHistory"."ModifiedDate") from
"EmployeePayHistory" where
"EmployeePayHistory"."EmployeeID"="Employee"."EmployeeID
") and "Department"."Name" = IsNull(:department,
"Department"."Name")
ORDER BY "Department"."Name", "Employee"."ManagerID",
"Employee"."EmployeeID"

```

The text in bold above is handling the formula functions of Crystal Report.

Now you can see the all the data elements that you need for the report in BI Publisher.

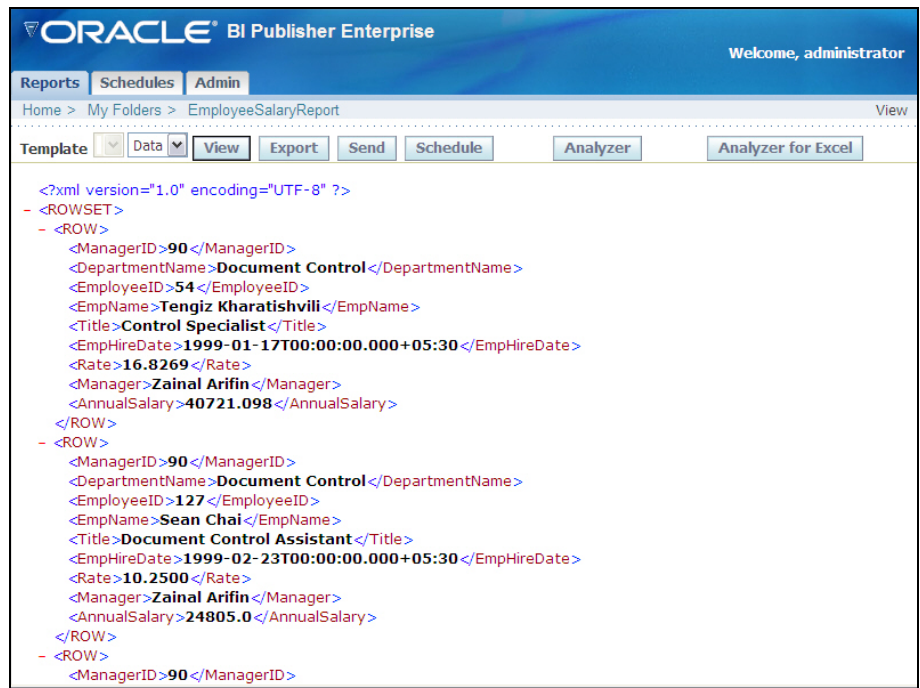


Figure 20. Final data in XML format

2. Analyze Report Layout

2.3.1: Layout Design

The data elements of this sample report are column names of the SQL Query. Then the report output indicates grouping and sorting happening. To check the grouping, open the group expert in Crystal Reports Developer.

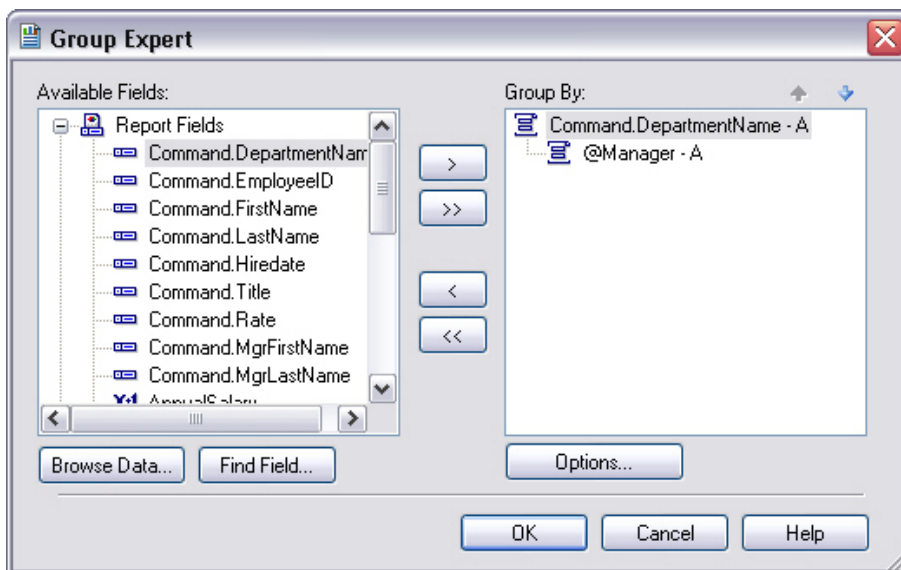


Figure 21. Group Expert

This report has two levels of grouping – By Department Name and then by Manager (which is a Formula Field). Click on ‘Options’ button while a ‘Group By’ item is selected to check the details of the grouping.

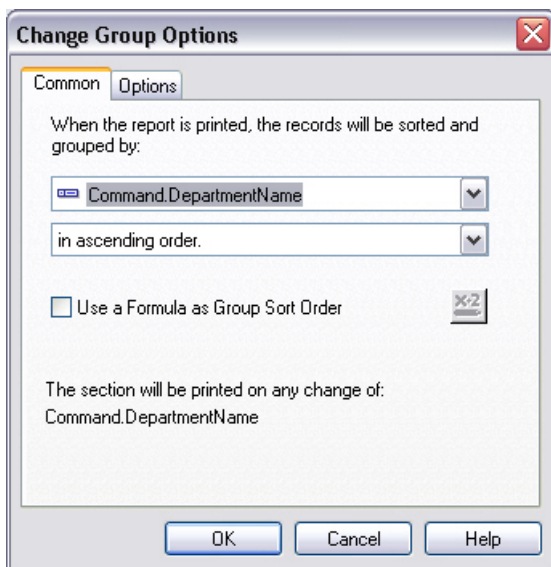


Figure 22. Group Options for grouping by DepartmentName

The DepartmentName will be sorted in ascending order. Select the Options tab to view other details.

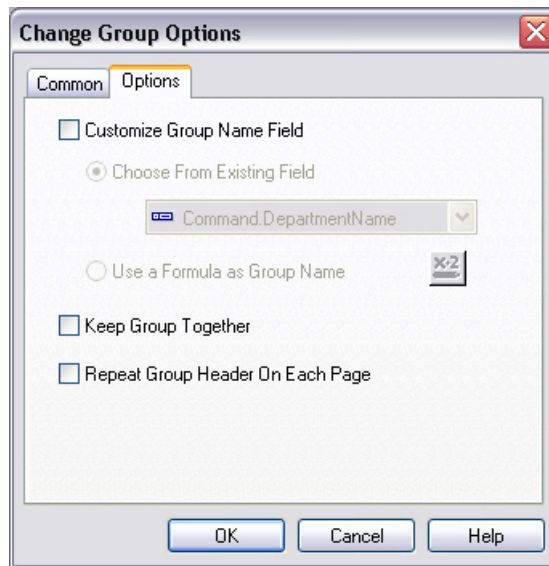


Figure 23. No properties checked under Options Tab.

No option has been selected above. Similarly check the details for the nested Grouping by 'Manager'.

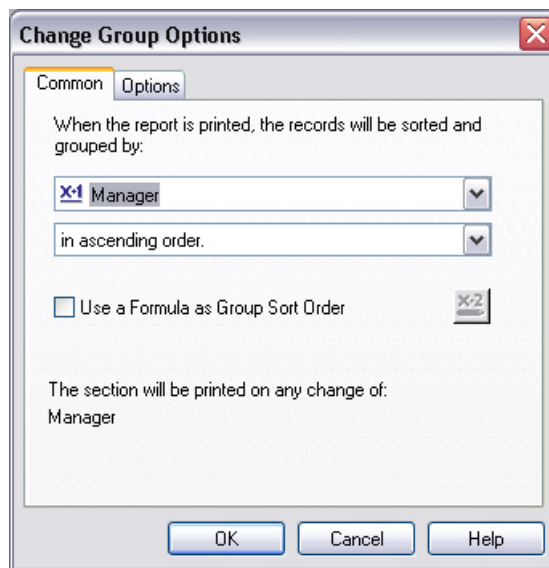


Figure 24. Grouping Options for Grouping by Manager

Here the group data is sorted by 'Manager' in ascending order. Click on Options tab to check the grouping options.

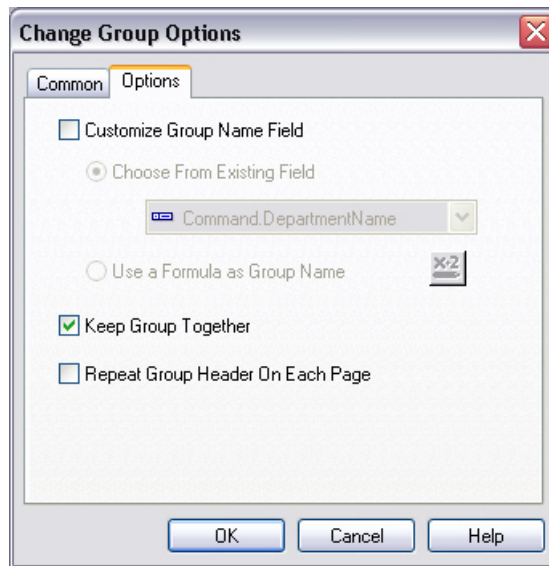


Figure 25. 'Keep Group Together' property checked.

The above configuration 'Keep Group Together' allows the data under a Manager to not break across page.

2.3.2: Calculations

To check the formula used in the report for formatting open the Formula Workshop. All the formulae used in the report layout will appear under section 'Formatting Formulas' in the Formula Workshop.

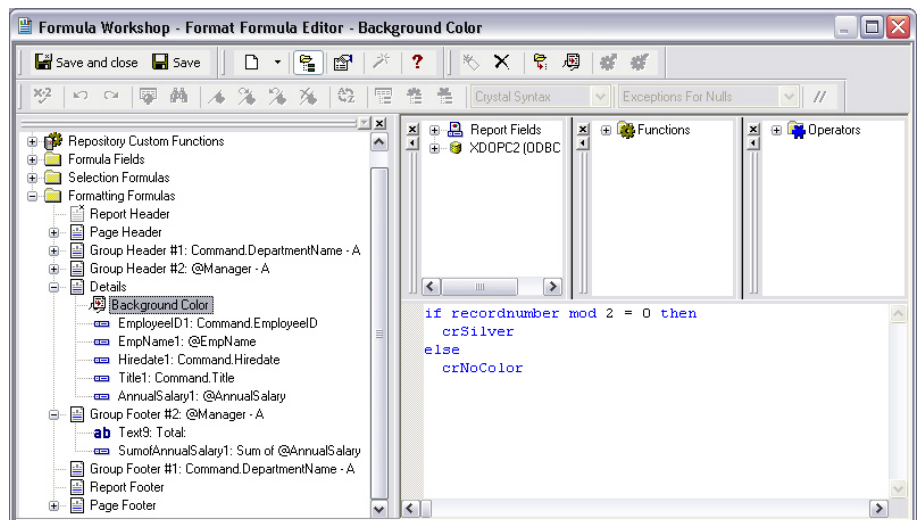


Figure 26. Formula Workshop showing conditional row formatting

The above figure shows the formula used to add background color to the Detail Section. The formula here would simply make the background color silver for alternate rows of data.

Another thing to notice in the Formula Workshop is the Summary Field in the Group Footer #2 with name 'Sum of @AnnualSalary', which is a sum of the formula field – AnnualSalary. To see the details of the summary field point your cursor over the Summary Field on the report design and right click. You will see the option 'Edit Summary' as shown below.

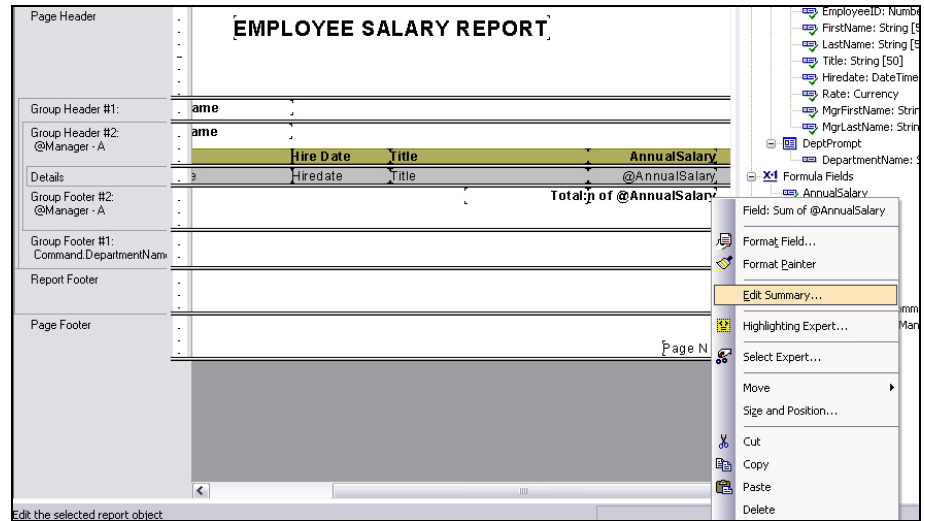


Figure 27. Edit Summary for Summary Field on Crystal Reports Developer

On the Edit Summary Dialogue you can check the field used for summary and the function used. In this report it's a Sum function applied on the formula field 'AnnualSalary'.

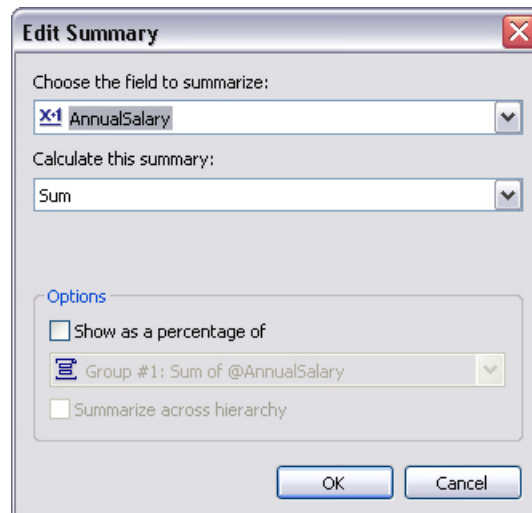


Figure 28. Summary Function used for the Summary Field

2.3: Formatting

Finally, you need to make a note of all the visual formatting applied on the Crystal reports. In this sample report you can see the banded format and the background color. Also the border of table and the column header background color can be noted.

This completes the analysis of this report, so you can now start converting this report into BI Publisher.

Convert Report Layout

2.1: Open a blank RTF

Open a blank RTF file.

2.2: Get Sample data

Log on to the BI Publisher server (e.g. <http://localhost:9704/xmlpserver>) from the Oracle BI Publisher Template Builder menu to load XML data for template design.



Figure 29. Login to Oracle BI Publisher Enterprise Server using Report Server URL

This will open the 'Open Template' dialog where you can see all the reports created under different folders.

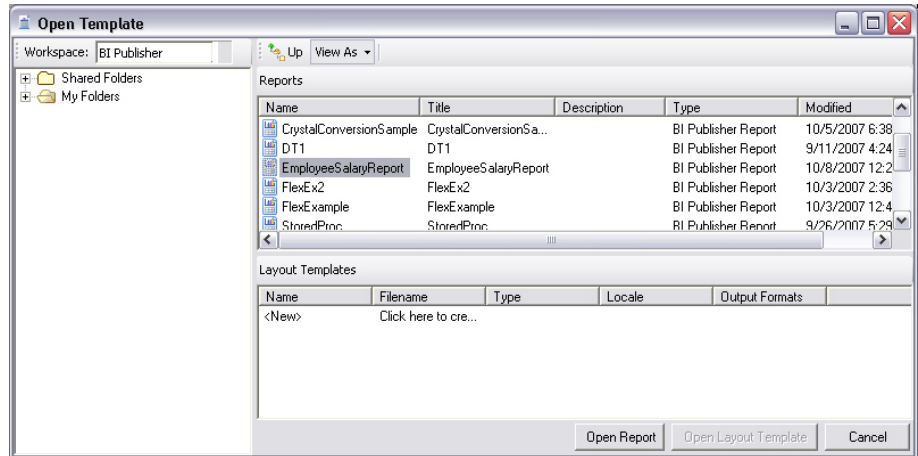


Figure 30. Browse reports in Open Template Dialog

Select the report by navigating to the appropriate folder and click on 'Open Report'. This will load data and the data information into the Template Builder so that you can begin to place data fields, tables and charts onto your layout.

2.3: Create Layout Template

2.3.1: Convert Design Layout

In Microsoft Word document, select 'Table Wizard' from the insert menu of Template Builder toolbar. This will open Table Wizard dialogue as shown below.



Figure 31. Table Wizard – report format selection

For this report you can select Table as the report Format.

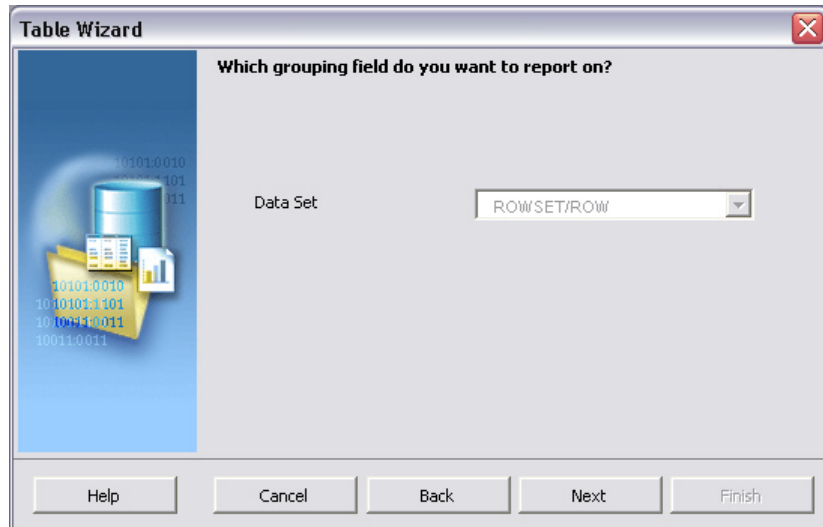


Figure 32. Table Wizard – grouping field selection

This screen allows you to select a Data Set grouping level from a drop down list. Since the data of this report has no hierarchy defined, it selects the default grouping level as ROWSET/ROW. Click on Next button.



Figure 33. Table Wizard – field selection

In this screen you can select the fields that you want to display in the report. Also you can set the order of the fields in top to bottom sequence that will form the order from left to right for these fields when rendered by this wizard. Continue to the next screen.

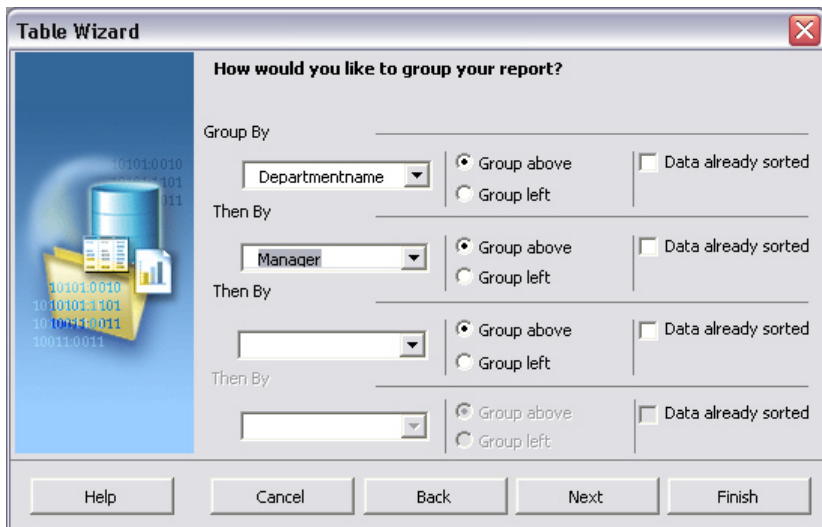


Figure 34. Table Wizard – Grouping

This screen allows setting of Group By fields and their sequence. Here 'Manager' Group By is nested within 'DepartmentName' Group By.

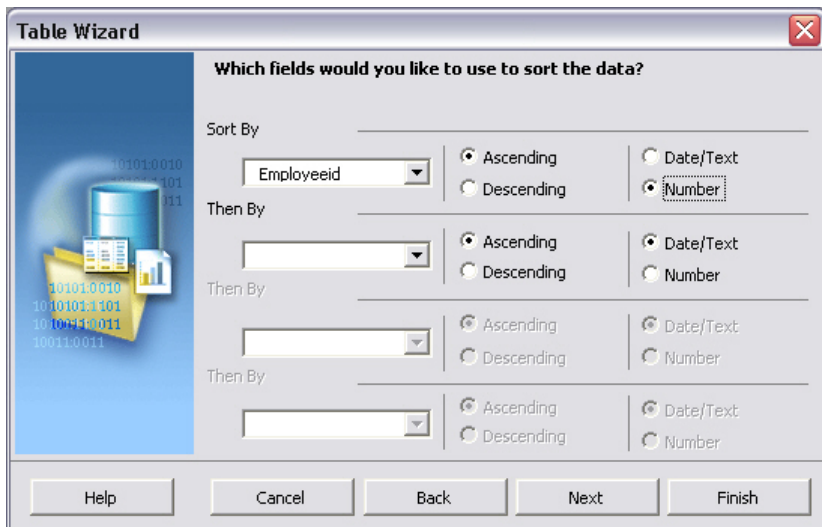


Figure 35. Table Wizard – sorting

Here the sort is for the table data, so select sort by Employeeid.

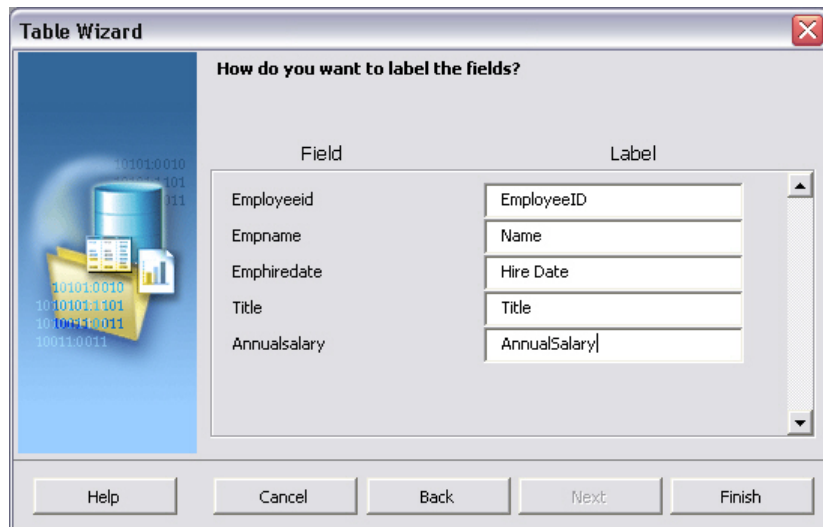


Figure 36. Table Wizard – Edit Label Fields

The Labels default to the field names, so this screen allows you to edit the labels. Now with this screen the table wizard is ready to create the initial template. Click on Finish to get the following template design:

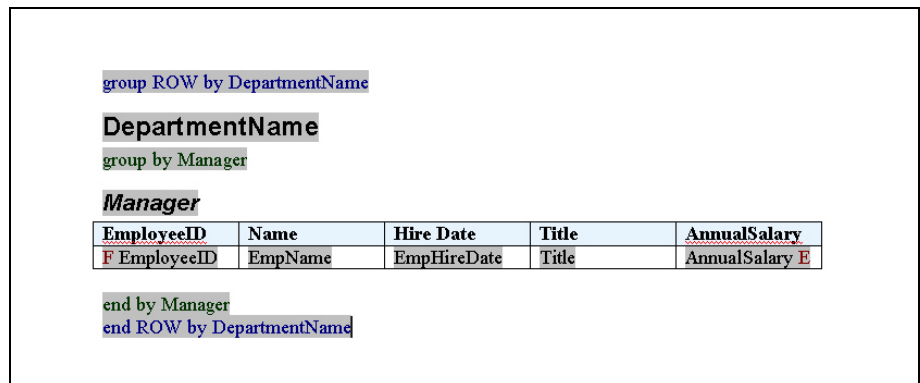


Figure 37. Initial design created by table wizard

Here all the data placeholder can be seen highlighted in grey.

2.3.2: Convert Calculations

This report has two calculations at layout formatting layer – condition for banded formatting and summary field. Also it has date and salary field that need to be formatted in a format.

To add the summary field to the report, insert another row to the table as shown below and use insert Field from the Template Builder toolbar.

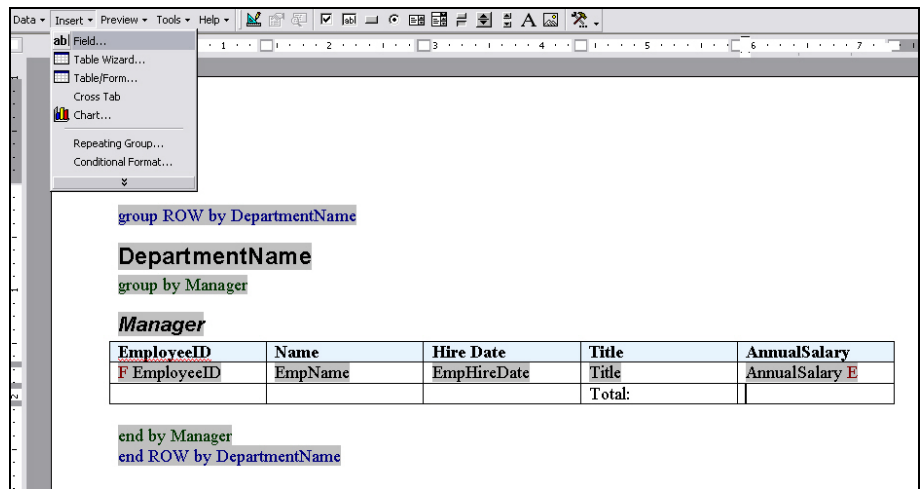


Figure 38. New row for Total Field

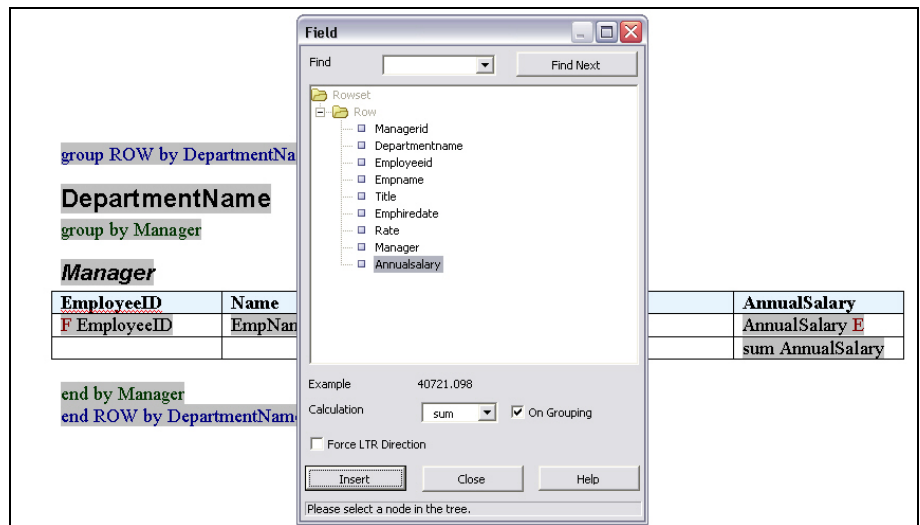


Figure 39. Insert sum of AnnualSalary field

To insert the summary field, select the AnnualSalary field in the dialog, select Calculation function 'sum' from the drop down list and check the 'On Grouping' checkbox. Click on 'Insert' button.

Next you should format the 'AnnualSalary' and 'sum AnnualSalary' fields to match the salary data format of Crystal Reports. Double click on the text form field 'AnnualSalary' to open the 'BI Publisher Properties' dialog. Select formatting type as 'Number' and format as '\$#,###0.00'. You can edit this format manually to include any new format. Add a 'Text to display' label, which is just a label name to this text form field. Follow the same steps for 'sum AnnualSalary' text form field.

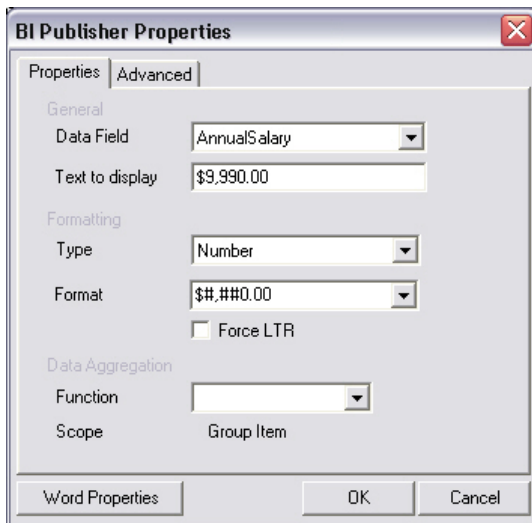


Figure 40. Formatting the AnnualSalary Field

Similarly, 'Hire Date' can be formatted to a date format of 'MM/dd/yyyy'. The format field is editable so you can enter the format manually.

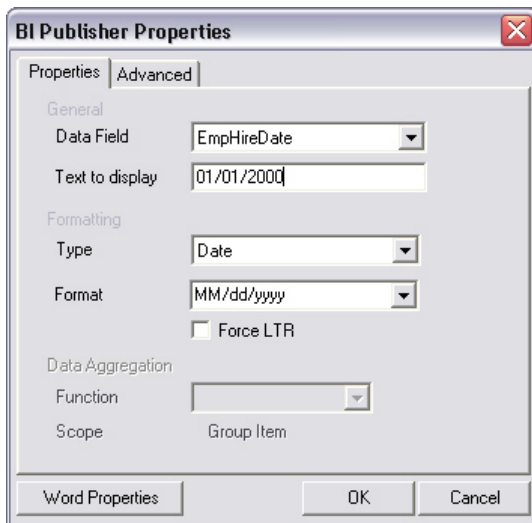


Figure 41. Format Hire Date

group ROW by DepartmentName

DepartmentName

group by Manager

Manager

EmployeeID	Name	Hire Date	Title	AnnualSalary
F EmployeeID	EmpName	01/01/2000	Title	\$9,990.00 E
			Total	\$9,990.00

end by Manager

end ROW by DepartmentName

Figure 42. Intermediate Template format

2.3.3: Convert Formatting

Next you can use the Microsoft Word Office Native feature of a table row to 'allow row to break across pages' to handle the 'Keep Group Together'. Insert a table of single row and single column and right click to edit table properties.

group ROW by DepartmentName

DepartmentName

group by Manager

--	--	--	--	--

Manager

EmployeeID	Name	Hi		AnnualSalary
F EmployeeID	EmpName	01		\$9,990.00 E
				\$9,990.00

end by Manager

end ROW by DepartmentName

Figure 43. Insert a new table for handling keep-together.

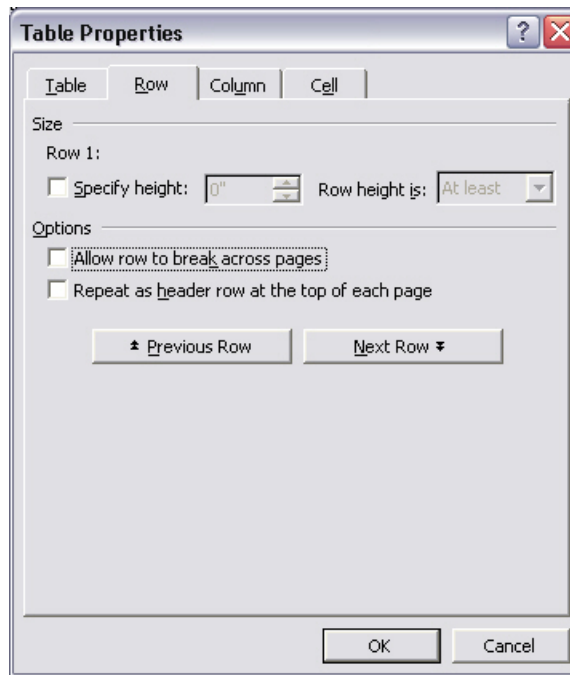


Figure 44. Table Properties to set Options to keep data together

Uncheck the 'Allow row to break across pages' option. This will keep the data in a row together and will not break across pages.

For this report you can include the content within 'group by Manager' inside this table row, thereby making this group data to remain together.

```

group ROW by DepartmentName
DepartmentName
group by Manager


| <b>Manager</b> |         |            |       |              |
|----------------|---------|------------|-------|--------------|
| EmployeeID     | Name    | Hire Date  | Title | AnnualSalary |
| F EmployeeID   | EmpName | 01/01/2000 | Title | \$9,990.00 E |
|                |         |            | Total | \$9,990.00   |


end by Manager
end ROW by DepartmentName

```

Figure 45. Include the employee data and manager inside the table.

Next you adjust the font size, font style, alignment to match the Crystal Report output. One of the steps towards pixel perfect report is to use tables to align the data precisely. So you can add new tables to include Manager and Department labels and data.

```

group ROW by DepartmentName
Department: DepartmentName

group by Manager
Manager: Manager

EmployeeID Name Hire Date Title AnnualSalary
F EmployeeID EmpName 01/01/2000 Title $9,990.00 E
Total: $9,990.00

end by Manager
end ROW by DepartmentName

```

Figure 46. Cell margins removed for Department and Manager Cells and aligning fields.

You can make adjustments in the cell margins to align the Department and Manager labels. Also cell margins can be removed from the outer table to align the employee table with the labels above. The EmployeeID, AnnualSalary labels and data can be aligned to right side.

```

group ROW by DepartmentName
Department: DepartmentName

group by Manager
Manager: Manager

EmployeeID Name Hire Date Title AnnualSalary
F EmployeeID EmpName 01/01/2000 Title $9,990.00 E
Total: $9,990.00

end by Manager
end ROW by DepartmentName

```

Figure 47. Hiding table borders, adding background color to the employee table heading.

Hide the table borders, add background color to the employee table heading. To match the RGB combination of the color you can use any tool that can read RGB values from a document.

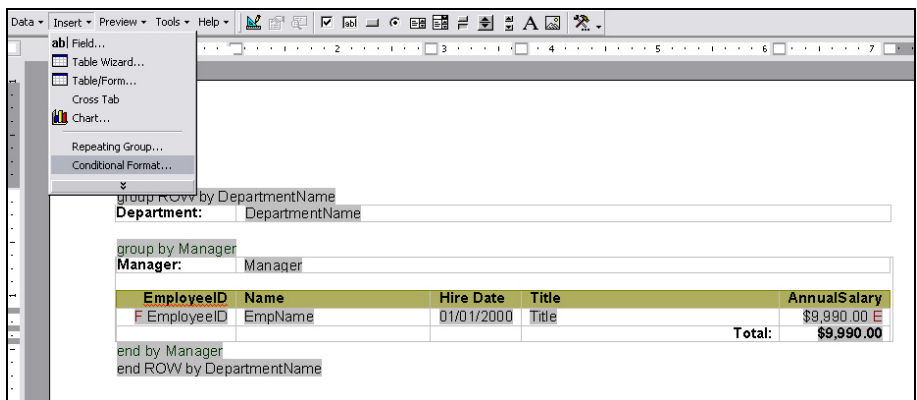


Figure 48. Conditional Formatting in Template Builder

Next, while keeping the cursor in the EmployeeID column after the for-each placeholder, you can select Conditional Formatting from the Insert menu of the Template Builder toolbar. This will prompt a dialog as shown below.

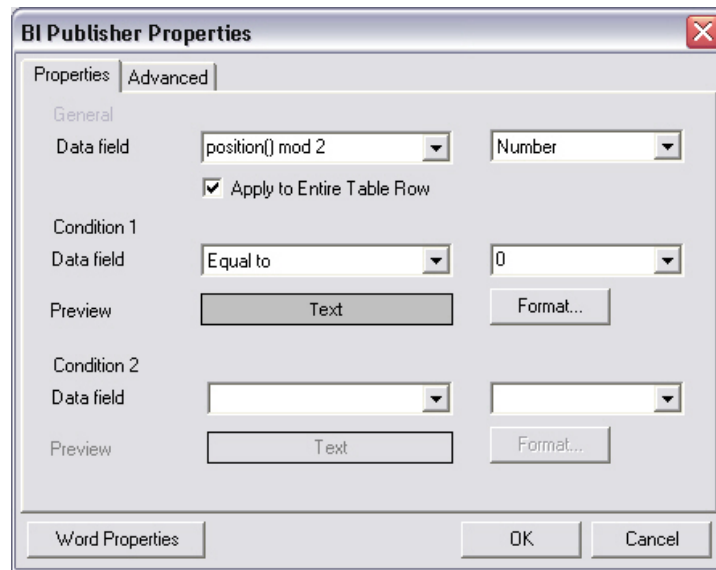


Figure 49. Conditional Formatting for row

The dialog allows you to apply a condition on data elements available in the drop down list. In this case you apply a custom condition that can be entered manually in the Data Field as shown in the figure above.

Be sure to click the checkbox 'Apply to Entire Table Row' as this condition should apply to entire table row. Then you can select the 'Format' Button to determine what format to apply when the condition is met. In this case, background color for the table row should be grey as shown below.

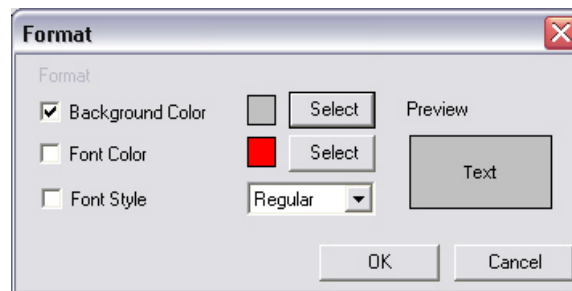


Figure 50. Selecting Background Color

group ROW by DepartmentName				
Department: DepartmentName				
group by Manager				
Manager: Manager				
EmployeeID	Name	Hire Date	Title	AnnualSalary
F C EmployeeID	EmpName	01/01/2000	Title	\$9,990.00 E
				Total: \$9,990.00
end by Manager				
end ROW by DepartmentName				

Figure 51. Template design for report body

With all the changes done so far this is how your report body design would look like.

Next you need to include the header and the footer content. After applying all the formatting to the template, your final template would look like the one below.

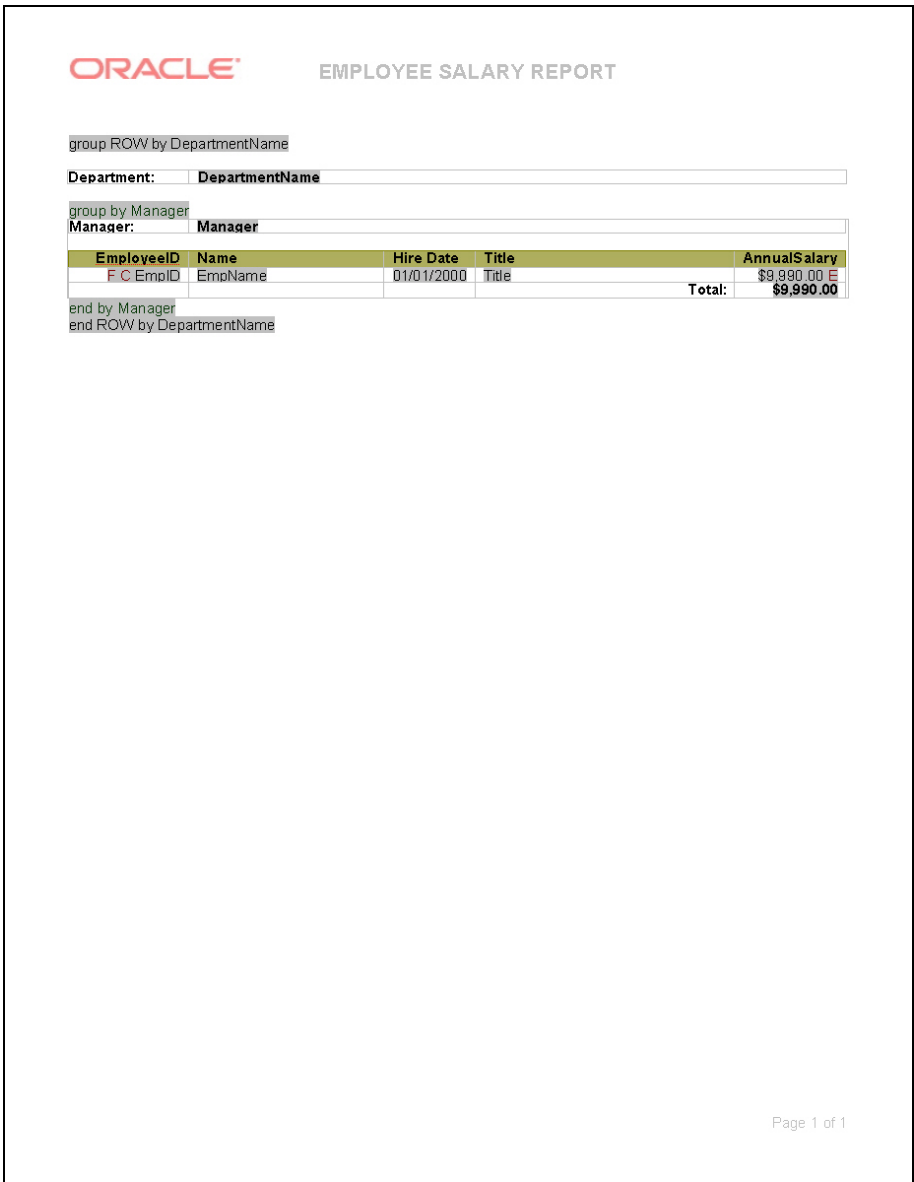


Figure 52. Final template design.

Following is the list of code associated with each Text Form Field placeholders :

Text Form field	Code
group ROW by DepartmentName	<?for-each-group:ROW; ./DepartmentName?> <?sort: ./DepartmentName; 'ascending'; data-type= 'text'?>
DepartmentName	<?DepartmentName?>
group by	<?for-each-group:current-group(); ./Manager?><?sort:current-

Manager	group()/Manager;'ascending';data-type='text'?'>
Manager	<?Manager?>
F	<?for-each:current-group()?><?sort:EmployeeID;'ascending';data-type='number'?'>
C	<?if@row:position() mod 2=0?><?attribute@incontext:background-color;'Silver'?'><?end if?>
EmpID	<?EmployeeID?>
EmpName	<?EmpName?>
01/01/2000	<?HireDate?>
Title	<?Title?>
\$9,990.00	<?AnnualSalary?>
E	<?end for-each?>
\$9,990.00	<?sum(current-group()/AnnualSalary)?>
end by Manager	<?end for-each-group?>
end ROW by DepartmentName	<?end for-each-group?>

2.4: Preview and Upload Report

Now, take a look at a preview of the final report to make sure the layout and formatting is correct. Navigate to 'Preview Template' under the 'Oracle BI Publisher' menu or Toolbar to view the report in PDF, HTML, RTF, Excel or PowerPoint formats. Now select 'Upload Template As' under the 'Oracle BI Publisher' menu to publish this template on the BI Publisher Enterprise Server. Enter a template Name and click OK on the next prompt to complete the upload.

Log on to the BI Publisher Enterprise Server, navigate to the 'EmployeeSalaryReport' under 'My Folders' and you will find that the report is ready to be viewed in PDF, HTML, RTF, Excel and PowerPoint formats.

ORACLE EMPLOYEE SALARY REPORT

Department: Production
 Manager: Andrew Hill

EmployeeID	Name	Hire Date	Title	AnnualSalary
8	Ruth Ellerbrock	02/08/1998	Production Technician - WC10	\$32,549.00
10	Barry Johnson	02/07/1998	Production Technician - WC10	\$32,549.00
13	Sidney Higa	03/05/1998	Production Technician - WC10	\$32,549.00
15	Jeffrey Ford	03/23/1998	Production Technician - WC10	\$32,549.00
17	Doris Hartwig	04/11/1998	Production Technician - WC10	\$32,549.00
19	Diane Glimp	04/28/1998	Production Technician - WC10	\$32,549.00
230	Bonnie Kearney	02/02/2000	Production Technician - WC10	\$32,481.75
Total:				\$227,775.75

Manager: Brenda Diaz

EmployeeID	Name	Hire Date	Title	AnnualSalary
31	Alejandro McGuel	01/07/1999	Production Technician - WC40	\$36,300.00
45	Fred Northup	01/13/1999	Production Technician - WC40	\$36,300.00
56	Kevin Liu	01/18/1999	Production Technician - WC40	\$36,300.00
68	Shammi Mohamed	01/25/1999	Production Technician - WC40	\$36,300.00
81	Rajesh Patel	02/01/1999	Production Technician - WC40	\$36,300.00
91	Lorraine Nay	02/05/1999	Production Technician - WC40	\$36,300.00
105	Paula Nartker	02/13/1999	Production Technician - WC40	\$36,300.00
116	Frank Lee	02/18/1999	Production Technician - WC40	\$36,300.00
138	Brian Lloyd	03/02/1999	Production Technician - WC40	\$36,300.00
152	Tawana Nusbaum	03/09/1999	Production Technician - WC40	\$36,300.00
190	Ken Myer	03/28/1999	Production Technician - WC40	\$36,300.00
215	Gabe Mares	04/09/1999	Production Technician - WC40	\$36,300.00
Total:				\$435,600.00

Manager: Cristian Petculescu

EmployeeID	Name	Hire Date	Title	AnnualSalary
224	Betsy Stadick	01/19/2000	Production Technician - WC10	\$32,481.75
234	Kimberly Zimmerman	02/13/2000	Production Technician - WC10	\$32,481.75
245	Patrick Wedge	03/04/2000	Production Technician - WC10	\$32,481.75
252	Danielle Tiedt	03/23/2000	Production Technician - WC10	\$32,481.75
262	Tom Vande Velde	04/10/2000	Production Technician - WC10	\$32,481.75
Total:				\$162,408.75

Manager: Cynthia Randall

EmployeeID	Name	Hire Date	Title	AnnualSalary
33	Jian Shuo Wang	01/08/1999	Production Technician - WC30	\$22,990.00
73	Sandra Reátegui Alayo	01/27/1999	Production Technician - WC30	\$22,990.00
110	Jason Watters	02/15/1999	Production Technician - WC30	\$22,990.00
142	Andy Ruth	03/04/1999	Production Technician - WC30	\$22,990.00
180	Rostislav Shabalin	03/23/1999	Production Technician - WC30	\$22,990.00
196	Michael Vanderhyde	03/30/1999	Production Technician - WC30	\$22,990.00
Total:				\$137,940.00

Figure 53. Report output from Oracle BI Publisher in PDF format.

The above figure shows the PDF output which matches the Crystal Report PDF output (Figure 1). This completes the conversion of this sample Crystal Report into BI Publisher Report.

CONCLUSION

The Template Builder has lot more features that can be used during design time. Also you can validate the template design for errors at any stage in the Template Builder. The steps to convert will vary from one report to another based on complexity of data extraction and the complexity of formatting involved. However, the step by step approach would apply to most of the reports.



Converting Reports from Business Object Crystal Reports to Oracle Business Intelligence Publisher
February 2008
Author: Pradeep Kumar Sharma
Contributing Author: Mike Donohue

Oracle Corporation
World Headquarters
500 Oracle Parkway
Redwood Shores, CA 94065

U.S.A. in law, including implied warranties and conditions of merchantability or fitness for a particular purpose. We specifically disclaim any liability with respect to this document and no contractual obligations are formed either directly or indirectly by this document. This document may not be reproduced or transmitted in any form or by any means, electronic or mechanical, for any purpose, without our prior written permission. Oracle is a registered trademark of Oracle Corporation and/or its affiliates. Other names may be trademarks of their respective owners.

Worldwide Inquiries:
Phone: +1.650.506.7000
Fax: +1.650.506.7200
oracle.com

Copyright © 2007, Oracle. All rights reserved.
This document is provided for information purposes only and the contents hereof are subject to change without notice.
This document is not warranted to be error-free, nor subject to any other warranties or conditions, whether expressed orally or implied