Reporting with Visual Studio and Crystal Reports

Create a reporting application from scratch using Visual Studio and Crystal Reports

Mahmoud Elkoush
Reporting with Visual Studio and Crystal Reports

Create a reporting application from scratch using Visual Studio and Crystal Reports

Mahmoud Elkoush

PACKT enterprise
BIRMINGHAM - MUMBAI
Credits

**Author**
Mahmoud Elkoush

**Reviewers**
Mohammed Imran Alam
Adeel Nasir

**Acquisition Editor**
Joanne Fitzpatrick

**Lead Technical Editor**
Govindan. K

**Technical Editors**
Pooja Arondekar
Anita Nayak

**Copy Editors**
Roshni Banerjee
Alfida Paiva
Adithi Shetty
Laxmi Subramanian

**Project Coordinator**
Amigya Khurana

**Proofreader**
Jenny Blake

**Indexer**
Tejal Soni

**Production Coordinator**
Nitesh Thakur

**Cover Work**
Nitesh Thakur

www.allitebooks.com
About the Author

Mahmoud Elkoush has over six years experience in software development. He has used many technologies throughout his career such as Microsoft Visual Studio, Microsoft SQL Server, Oracle database, Crystal Reports, and so on. He has worked as a Project Manager in the Kingdom of Saudi Arabia for nearly two years, and currently works as the Chief Information Officer of Geomeric.

This is the first book that Mahmoud has written, and he hopes it will be the first of many.

______________________________________

I want to thank my wife because she convinced me to write this book and supported me a lot to produce it.

______________________________________
Mohammed Imran Alam can be described as a simple, hardworking, technology-enthusiast, and a fun-, family-, and God-loving person. He comes from a small village in India known as Adilabad and loves to mention his village whenever he gets a chance because he would like people to know about it. He had his elementary education in one of the best schools, St Joseph's Convent School, Adilabad, and he owes a lot to this school for making him who he is today, of course after his parents' upbringing. Other hobbies include playing and watching cricket, and he gives it his 100 percent when he is at ground playing this great sport. There are still a few teams in the US who think that he has still got potential to be a match winner, and constantly want him in their team. He hopes they don't regret their decision to include him in playing eleventh.

With regards to his career, he always thinks that it was his destiny to be landed in the field of Business Intelligence. After completing his Bachelors in ECE (Electronics and Telecommunications) from Osmania University, he landed an offer to work in Saudi Arabia to work as a Crystal Reports developer based on his credentials as an Oracle programmer. This was the first time he was introduced to Crystal Reports (Version 7 at that time), and it was like "love at first sight", with reporting and BI field of course. Since then, 14 long years have passed with him working on all kinds of products that are related to Crystal Reports. He has seen this beautiful product grow with with the introduction of Enterprise Software as a distribution tool for reports in different formats (which is now SAP Business Objects Enterprise 4.0) and with the implementation of other awesome products such as Explorer/Dashboard Design etc etera, all of which complement the core reporting.

He considers himself to be blessed with gifts from God, a beautiful wife, and three awesome kids who keep him going. He is also surrounded by very good friends that he made starting from his kindergarten days to his Engineering days. Twitter (@ReportExpert is his handle) and other social media sites make it easier to be in touch with them and he likes to keep in touch with everyone. Along with this great bunch, he is able to do some philanthropic work and help poor and needy people back home.
He has this personal message: "This is all I had to mention about me, thanks for reading it and I hope you will love this book and gain the knowledge you were looking for."

He writes some topics that are personal/technical on his blog www.imranalam.com. You are most welcome to read it and make comments. He likes to read all his comments especially if they are critical about a topic or article that he has posted, as that helps him understand their perspective and allows him to correct it.

I would like to thank my mom, wife, and kids, who have always sacrificed the quality time I could have spent with them.

---

**Adeel Nasir** is an IT professional with a total of nine and a half years experience in the IT industry on software/web analysis, engineering, development, and deployment. He currently works in a software company in Dubai as a Systems Analyst.

He is experienced in ASP.NET(C#), web services, and Microsoft SQL Server. He has also worked on Ultimus BPM suite (workflow automation tool) for three years. He also worked for one year in configuration/administration, development, and management of MOSS 2007.

He is a Microsoft Certified Technology Specialist for TS: Microsoft .NET Framework - Application Development Foundation (70-536), Web-based Client Development (70-528), and Programming in HTML 5 with JavaScript and CSS3 (70-480).

He has been Microsoft's Guest Speaker at different universities and has written many articles on IT in different local magazines. He was one of the trainers of ASP.NET Workshop at a well-known university in June 2008 as well as managing training on ASP.NET 2.0 at the same time. He also judged the software competition called Bits Bytes and Colors 2007 at a well-known university.
He is also managing a blog (http://fuzon.biz) to help the IT community with the solutions of problems that they face. Actually, this blog is a mixture (fusion) of content on different categories, for example, technology, travel, news, fun, and much more.

He is also working with a bunch of IT professionals on a freelance basis to provide technology solutions. Details can be found at (http://ibs.fuzon.biz). You may contact him via e-mail at adeel.nasir@gmail.com or adeel@fuzon.biz.

I would like to thank my family and friends for their continuous support in this regard.
Table of Contents

Preface 1

Chapter 1: Setting Up the Requirements 5

Installing Microsoft Visual Studio 2012 5
  Getting the source 5
  Installation 6

Installing Crystal Reports for Visual Studio 2012 8
  Getting the source 8
  Installation 11

Installing Microsoft SQL Server 2008 13
  Getting the source 13
  Installation 15

Installing Northwind database in Microsoft SQL Server 2008 22
  Getting the source 22
  Installation 24

Summary 28

Chapter 2: Designing a Sample Application 29

Creating a new application 29

Adding controls to the form 33
  Employees 33
    Beginning your design process 34
  Products and orders 35
    Beginning your design process 35
  Employees and orders 36
    Beginning your design process 36

Validating form data 37

Sketching your reports on paper 38

Summary 43
Chapter 3: Selecting the Data 45
SQL queries in Microsoft Visual Studio 45
  Connecting to our database 45
Add database diagram 47
  Test queries 49
Populate the ComboBoxes with data 50
  Add DataSet to the application 50
  Populate the cbEmpTitle ComboBox 50
  Populate the cbEmpCity ComboBox 55
  Populate the cbEmpCountry ComboBox 56
  Populate the cbProductsCategory ComboBox 58
  Populate cbOrdersCountry ComboBox 60
Summary 61

Chapter 4: Creating First Report 63
Preparing a report data 64
  Selecting the report data 64
  Creating a TableAdapter 67
Adding a report to the application 67
Working with lines and boxes 73
Using special fields 74
Adding the report to a form 74
Summary 76

Chapter 5: Enhancing Reports 77
Using parameter fields 78
Working with formula fields 87
Working with subreport 88
Grouping and sorting data 90
Summary 94

Chapter 6: Working with Cross-Tab Reports 95
Creating our report 96
Customizing the Cross-Tab appearance 102
Performing page setup 108
Summary 109

Chapter 7: Working with Charts 111
Creating the reports data source 113
Creating the first report – bar chart 114
  Removing the legend 118
  Removing the chart's title 118
  Changing the chart's appearance 119
<table>
<thead>
<tr>
<th>Section</th>
<th>Page</th>
</tr>
</thead>
<tbody>
<tr>
<td>Creating the second report – pie chart</td>
<td>120</td>
</tr>
<tr>
<td>Removing the legend</td>
<td>122</td>
</tr>
<tr>
<td>Removing the chart's title</td>
<td>122</td>
</tr>
<tr>
<td>Changing the chart's appearance</td>
<td>122</td>
</tr>
<tr>
<td>Creating the third report – gauge chart</td>
<td>123</td>
</tr>
<tr>
<td>Removing the legend</td>
<td>125</td>
</tr>
<tr>
<td>Removing the chart's title</td>
<td>125</td>
</tr>
<tr>
<td>Changing the chart's appearance</td>
<td>125</td>
</tr>
<tr>
<td>Creating the fourth report – all in one</td>
<td>126</td>
</tr>
<tr>
<td>Summary</td>
<td>127</td>
</tr>
<tr>
<td>Index</td>
<td>129</td>
</tr>
</tbody>
</table>
Preface

Reports are of paramount importance for a developer who wants to create successful software. Crystal Reports is an amazing reporting tool because it is compatible with Visual Studio and other developing tools. Crystal Reports is compatible with a lot of databases and is easy to use. It is a powerful reporting tool and can be used to create any report with the least amount of effort. This book is different from other technical books because it not only describes the tools and features in Crystal Reports and Visual Studio, but also contains a sample application with eight reports. Every report has a new idea and teaches the reader a new tool. The reader not only learns to create reports but also learns how to display these reports in Microsoft Visual Studio applications.

What this book covers

Chapter 1, Setting Up the Requirements, will show how to install the tools that we will use.

Chapter 2, Designing a Sample Application, contains a description of the sample application that we will create.

Chapter 3, Selecting the Correct Data, will show how to connect to a database and run SQL queries using Microsoft Visual Studio and also talk about dataset and TableAdapter.

Chapter 4, Creating Your First Report, will show how to create a simple report and display this report in our application.

Chapter 5, Enhancing the Report, will show how to enhance the report using grouping, filters, and subreports.
Chapter 6, Working With Cross-Tab Reports, covers what cross-tab is and describes how to use it.

Chapter 7, Working with Charts, covers the different types of charts in Crystal Reports and describes how to use them.

What you need for this book
You will require the following softwares to be installed on your system to run the examples in this book:

- Microsoft Visual Studio 2012
- Crystal Reports for Visual Studio 2012
- Microsoft SQL Server 2008
- Northwind Database

Who this book is for
This book is for developers who are new to Crystal Reports. It will also prove useful to intermediate users who wish to explore some new techniques in Crystal Reports using Microsoft Visual Studio. Readers are expected to have basic knowledge of C#, Microsoft Visual Studio, and Structured Query Language (SQL).

Conventions
In this book, you will find a number of styles of text that distinguish between different kinds of information. Here are some examples of these styles, and an explanation of their meaning.

Code words in text are shown as follows: "After downloading the source, extract it, and double-click on the vs_premium.exe file to begin the installation."

New terms and important words are shown in bold. Words that you see on the screen, in menus or dialog boxes for example, appear in the text like this: "Click on the OK button to save the changes and run the application to test the new appearance."
Warnings or important notes appear in a box like this.

Tips and tricks appear like this.

Reader feedback
Feedback from our readers is always welcome. Let us know what you think about this book—what you liked or may have disliked. Reader feedback is important for us to develop titles that you really get the most out of.

To send us general feedback, simply send an e-mail to feedback@packtpub.com, and mention the book title via the subject of your message.

If there is a topic that you have expertise in and you are interested in either writing or contributing to a book, see our author guide on www.packtpub.com/authors.

Customer support
Now that you are the proud owner of a Packt book, we have a number of things to help you to get the most from your purchase.

Errata
Although we have taken every care to ensure the accuracy of our content, mistakes do happen. If you find a mistake in one of our books—maybe a mistake in the text or the code—we would be grateful if you would report this to us. By doing so, you can save other readers from frustration and help us improve subsequent versions of this book. If you find any errata, please report them by visiting http://www.packtpub.com/submit-errata, selecting your book, clicking on the errata submission form link, and entering the details of your errata. Once your errata are verified, your submission will be accepted and the errata will be uploaded on our website, or added to any list of existing errata, under the Errata section of that title. Any existing errata can be viewed by selecting your title from http://www.packtpub.com/support.
Piracy

Piracy of copyright material on the Internet is an ongoing problem across all media. At Packt, we take the protection of our copyright and licenses very seriously. If you come across any illegal copies of our works, in any form, on the Internet, please provide us with the location address or website name immediately so that we can pursue a remedy.

Please contact us at copyright@packtpub.com with a link to the suspected pirated material.

We appreciate your help in protecting our authors, and our ability to bring you valuable content.

Questions

You can contact us at questions@packtpub.com if you are having a problem with any aspect of the book, and we will do our best to address it.
Setting Up the Requirements

In this book we will create a sample reporting application and we need to install some tools to be ready to start. We will use Microsoft Visual Studio 2012 to design our sample reporting application, Crystal Reports to design the reports, Microsoft SQL Server 2008 as the database engine, and Northwind as a sample database. We will have to install all these applications before we get started.

In this chapter, we will cover the following topics:

- Installing Microsoft Visual Studio 2012
- Installing Crystal Reports
- Installing Microsoft SQL Server 2008
- Installing Northwind database

Installing Microsoft Visual Studio 2012

Since we are going to use Microsoft Visual Studio to create our sample reporting application, let's begin by installing it on our machine.

Getting the source

Microsoft Visual Studio 2012 source can be downloaded from the Microsoft website by going to the following URL:

http://www.microsoft.com/visualstudio/eng/downloads#d-express-windows-desktop

We will use the premium edition in our sample. After downloading the source, extract it, and double-click on the vs_premium.exe file to begin the installation.
Setting Up the Requirements

Installation
Now that we have downloaded and extracted the source file, let's install it on our machine.

1. **License agreement**: As we can see in the following screenshot, we can change the installation directory by clicking on the **Browse** button and selecting the directory in which we wish to install the files. Check the **I agree to the License terms and conditions** checkbox to complete the installation and click on the **Next** button.
2. **Optional features**: As we can see in the following screenshot, we can customize the installation features in Visual Studio. In our reporting sample application, we need **Microsoft SQL Server Data Tools** and, therefore, we can deselect all other features. But it is better to install all the features on our machine. After we select the features, click on the **INSTALL** button.
3. **Installation process**: As we can see in the following screenshot, the dialog box simply displays the status of the installation process. This may take several minutes.

![Installation Status](image)

4. **Setup successful**: After a few minutes, a dialog box will pop up telling us that we have successfully installed Visual Studio on our machine.

### Installing Crystal Reports for Visual Studio 2012

Now that we have finished installing Visual Studio, let us install Crystal Report.

### Getting the source

Crystal Reports source can be downloaded from the SAP website at the following URL: [http://www.sap.com/index.epx](http://www.sap.com/index.epx).

Following are the steps to download Crystal Reports:

1. Click on the **Free Trials** link as we see in the following screenshot:
2. Click on **SAP Crystal Reports, developer version for Microsoft Visual Studio** as we see in the following screenshot:

![Analytics Table](image)

3. Click on **Download SAP Crystal Reports, developer version for Microsoft Visual Studio now**, as shown in the following screenshot:

![Download Link](image)
4. As we see in the following screenshot, if you have an account, click on **log in now** else register and click on **Download Now** to start the download.

This version is compatible with Microsoft Visual Studio 2012. After downloading the source, double-click on the `CRforVS_13_0_5.exe` file to begin installation.
Installation

1. **Installation language**: Select English language and click on the OK button as we see in the following screenshot:

![Installation Language Selection](image1.png)

2. **Welcome**: Read notes and warnings in this dialog box and click on the Next> button as shown in the following screenshot:

![Welcome Screen](image2.png)
3. **License Agreement**: Read the License Agreement carefully. Then select the *I accept the License Agreement* checkbox and click on the **Next** button as we see in the following screenshot:

![License Agreement Screenshot]

4. **Confirmation**: As shown in the following screenshot, click on the **Next** button to begin the installation. It will take a few minutes to finish installing the software on our machine.

![Confirmation Screenshot]
5. **Successfully installed**: This step tells us that Crystal Reports is installed successfully and asks us if we wish to install 64-bit runtime. Select the checkbox and click on the **Finish** button as we see in the following screenshot:

![SAP Crystal Reports, version for Microsoft Visual Studio Setup](image)

**Installing Microsoft SQL Server 2008**

Now let's install SQL Server 2008.

**Getting the source**

Microsoft SQL Server 2008 source can be downloaded from the Microsoft website at:

Following are the steps to download Microsoft SQL Server 2008:

1. Search for SQL Server 2008 r2 service pack 2 by typing in `download sql server 2008 r2 service pack 2` as we see in the following screenshot:

2. Select Downloads and click on the first result:

3. Click on Download:
4. Choose a compatible version of SQL Server 2008 with your windows. I have Windows 7, 64-bit so I select the `SQLServer200BR2SP2-KB2630458x64-ENU.exe` version of SQL Server 2008 as we see in the following screenshot. Click on the Next button to begin the download.

![Choose the download you want](image)

5. After we finish downloading, we can find a file named `SQLServer2008R2SP2-KB2630458-x64-ENU.exe`; double-click on the downloaded file to begin the installation.

**Installation**

Following are the steps to install Microsoft SQL Server 2008:

1. **SQL Server Installation Center**: As we see in the following screenshot, select Installation, and click on the New SQL Server stand-alone installation or add features to an existing installation link.

![SQL Server Installation Center](image)
2. **Setup Support Rules**: As shown in the following screenshot, Microsoft checks for problems that occur during the installation in order to fix them before the installation process starts. If any of the checks fail, click on **Show details >>** to find out what went wrong. Correct it, and then click on **Re-run**. If the test is completed successfully, click on **OK**.

![Setup Support Rules](image1.png)

3. **Product Key**: As shown in the following screenshot, we can enter the product key by selecting the **Enter the product key**: radio button and filling in the product key. To proceed without entering a product key, select **Specify a free edition**: for evaluating SQL server 2008 and click on **Next**.

![Product Key](image2.png)
4. **License terms**: After reading the terms and conditions, select the **I accept the license terms** checkbox and click on **Next**.

5. **Setup Support Files**: Simply click on **Next** to complete installation.

6. **Setup Support Rules**: As shown in the following screenshot, Microsoft checks for problems that might occur while installing SQL Server Setup Support files. If there are any problems, we need to fix it before we finish installation. If there are no issues, we can complete the installation and click on **Next**.

![Setup Support Rules](image-url)
7. **Feature Selection**: As shown in the following screenshot, we will choose what feature we will install on our machine. For basic installation, we need to install only **Database Engine Services** and **Management Tools – Basic**. But it is always better to install all the features. After choosing what features you want on your machine, click on **Next**.
8. **Instance Configuration**: As shown in the following screenshot, we can choose to keep Default instance name MSSQLSERVER, or change it to any other name. It is recommended that you keep the default name though. On the other hand, we can change the installation directory to any other directory that has the required space. Click on Next.

![Instance Configuration Screenshot](image)

9. **Disk Space Requirement**: Simply click on Next to complete the installation.
10. **Server Configuration**: As shown in the following screenshot, we will set up **Service Accounts** that will be used to run SQL Server. We will use the built-in network service account for all services (this account does not require a password) because we are installing SQL Server on our machine. In the **Account Name** column, select `NT AUTHORITY\NETWORK SERVICE` for all the services and click on **Next**.

11. **Database Engine Configuration**: As shown in the following screenshot, we will choose the authentication security mode. There are two types of authentication:

   - **Windows Authentication**: When a user connects through a Microsoft Windows user account, SQL Server validates the account name and password using information in the Windows operating system. This is the default authentication mode, and is much more secure than Mixed Mode.
° **Mixed Mode**: This authentication mode allows users to connect by using Windows Authentication and SQL Server Authentication. Users who connect through a Windows user account can use trusted connections that are validated by Windows. If you choose Mixed Mode Authentication, you must set strong passwords for all SQL Server logins.

12. Choose **Mixed Mode (SQL Server authentication and Windows authentication)** and enter the password (let’s say we use asd123 as SQL Server password). Retype the password to confirm it and click on **Add Current User**. When this is done, click on **Next**.

13. **Analysis Services Configuration**: If we don't choose analysis service in the feature selection dialog, we will skip this step. Click on **Add Current User** and then click on **Next**.
14. **Reporting Services Configuration**: This step, like the previous one, will be skipped if we don't choose **Reporting Services Configuration** in the feature selection dialog. Keep the default options and click on **Next**.

15. **Error and Usage Reporting**: This step asks us if we wish to send the error information to Microsoft. Keep the option selected and click on the **Next** button.

16. **Installation Rules**: This dialog checks if there are any processes or other installations running, which will interrupt the installation of SQL Server 2008. If there are any issues, we have to fix it before installation. If there are no issues, click on **Next**.

17. **Ready to Install**: This step displays all the features that will be installed on our machine. Click on **Install** to complete installation.

18. **Installation Progress**: SQL server will now be installed on our machine. After the installation process is done, we will see a table of all the features that have been installed on our machine.

19. **Complete**: We will see a small dialog box telling us that Microsoft SQL server installation has been completed successfully. Click on **Close** to close this dialog box. Then we will see the main installation center. We can close it. Now you will have to restart the machine.

### Installing Northwind database in Microsoft SQL Server 2008

Now that we have finished installing Microsoft SQL Server, let us install the Northwind database.

### Getting the source

Northwind database source can be downloaded from the Microsoft web site at:

1. Search for the Northwind database download by typing in download Northwind database as shown in the following screenshot:

2. Click on the first result as we see in the following screenshot:

3. Click on Download as shown in the following screenshot to start the download:

4. After we finish downloading, we will get a file named SQL2000SampleDb.msi. Double-click on this file to begin the installation.
Installation
Following are the steps to install the Northwind database:

1. **Welcome Dialog**: As shown in the following screenshot, it is simply a welcome dialog. Click on the **Next** button.

![Welcome Dialog Screenshot]

2. **License Agreement**: Select the **I Agree** radio button and click on **Next** as shown in the following screenshot:

![License Agreement Screenshot]
3. **Choose Installation Options**: We will leave the default settings as we see in the following screenshot and click on **Next**.

![Choose Installation Options](image1)

4. **Confirm Installation**: We can now install the Northwind database on our machine and click on **Next**.

![Confirm Installation](image2)
5. **Installing the Database**: As shown in the following screenshot, the database scripts will be installed on our local drive.

![Database Installation Screenshot]

6. **Installation Complete**: After the scripts have been installed, we will find a folder named *SQL Server 2000 Sample Databases* created in the `C:\`. We will use one of these files to install the Northwind database into SQL Server 2008.

7. **Open SQL Server Management Studio**: From the Start menu select All Programs, then select Microsoft SQL Servers 2008, and click on SQL Server Management Studio.

8. **Connect to SQL Server 2008**: As shown in the following screenshot, select Database Engine from the Server type: menu, Server name: is localhost (or "."), which represents our machine. The Authentication type is SQL Server Authentication, the same option that we selected during installation. The Login name is `sa`, which represents the default user of SQL Server, and the password is `as123` or the password you had entered while installing the SQL Server. After filling in all the fields, click on Connect.

![SQL Server Management Studio Screenshot]
9. **Create Northwind database in SQL Server 2008**: Open C:\SQL Server 2000 Sample Databases. This is where the database scripts are installed. Drag the file named instnwnd.sql file, which contains all SQL scripts required to create the Northwind database on our server, and drop it in SQL Server Management Studio. As we can see in the following screenshot, click on the **Execute** button to run the scripts.

![SQL Server Management Studio](image1.png)

10. After all scripts are executed correctly, we will see Northwind in our databases, as can be seen in the following screenshot:

![SQL Server Management Studio](image2.png)
Setting Up the Requirements

Summary
In this chapter, we learned how to set up all the tools we will need to complete our sample application. In the next chapter, we will start with the design of our application and we will see that this application contains most types of Crystal Reports.
Designing a Sample Application

In this chapter, we will learn how to use Microsoft Visual Studio 2012. We will create our application, and by studying the working of this application, we will learn the features of Visual Studio. We will also design our interface and validate our data. By the end of this chapter, we will have the final design of our application.

In this chapter, we will cover the following topics:

- Creating a new application
- Adding controls to the form
- Validating form data
- Sketching our reports on paper

Creating a new application

We will now start using Microsoft Visual Studio, which we installed in the previous chapter:

1. From the Start menu, select All Programs, then select the Microsoft Visual Studio 2012 folder, and then select Visual Studio 2012.
2. Since we are running Visual Studio for the first time, we will see the following screenshot. We can select the default environment setting. Also, we can change this setting in our application. We will use C# as the programming language, so we will select **Visual C# Development Settings** and click on the **Start Visual Studio** button.

3. Now we will see the **Start Page** of Microsoft Visual Studio 2012. Select **New Project** or open it by navigating to **File | New | Project**. This is shown in the following screenshot:
4. As shown in the following screenshot, we will select **Windows** as the application template. We will then select **Windows Forms Application** as the application type. Let us name our application **Monitor**. Choose where you want to save the application and click on **OK**.
Now we will see the following screenshot. Let's start to design our application interface.

5. Start by right-clicking on the form and navigating to Properties, as shown in the following screenshot:
6. The Properties explorer will open as shown in the following screenshot. Change Text property from Form1 to monitor, change the Name property to MainForm, and then change Size to 322, 563 because we will add many controls to our form. After I finished the design process, I found that this was the suitable form size; you can change form size and all controls sizes as you wish in order to better suit your application interface.

![Properties Explorer Screenshot]

Adding controls to the form

In this step, we will start designing the interface of our application by adding controls to the Main Form.

We will divide the Main Form into three main parts as follows:

- Employees
- Products and orders
- Employees and products

Employees

In this section, we will see the different ways in which we can search for our employees and display the result in a report.
Beginning your design process

Let's begin to design the Employees section of our application using the following steps:

1. From the Toolbox explorer, drag-and-drop GroupBox into the Main Form. Change the GroupBox Size property to 286, 181, the Location property to 12, 12, and Text property to Employees.

2. Add Label to the Main Form. Change the Text property to ID and the Location property to 12, 25.

3. Add TextBox to the Main Form and change the Name property to txtEmpId. Change the Location property to 70, 20.

4. Add GroupBox to the Main Form. Change the Groupbox Size property to 250, 103, the Location property to 6, 40, and the Text property to filters.

5. Add Label to the Main Form. Change the Text property to Title and the Location property to 6, 21.

6. Add ComboBox to Main Form. Change the Name property to cbEmpTitle and the Location property to 56, 17.

7. Add Label to Main Form. Change the Text property to City and the Location property to 6, 50.

8. Add ComboBox to Main Form. Change the Name property to cbEmpCity and the Location property to 56, 46.

9. Add Label to Main Form. Change the Text property to Country and the Location property to 6, 76.

10. Add ComboBox to Main Form. Change the Name property to cbEmpCountry and the Location property to 56, 72.

11. Add Button to Main Form. Change the Name property to btnEmpAll, the Location property to 6, 152, and the Text property to All.

12. Add Button to Main Form. Change the Name property to btnEmpById, the Location property to 99, 152, and the Text property to By Id.

13. Add Button to Main Form. Change the Name property to btnEmpByFilters, the Location property to 196, 152 and the Text property to By filters.
Chapter 2

The final design will look like the following screenshot:

Products and orders
In this part, we will see the relationship between products and orders in order to demonstrate, which products have higher orders. Because we have a lot of product categories and orders from different countries, we filter our report by products category and countries.

Beginning your design process
After we finished the design of employees section, let's start designing the products and orders section.

1. From Toolbox explorer, drag-and-drop GroupBox to the Main Form. Change the GroupBox Size property to 284, 136, the Location property to 12, 204 and the Text property to Products – Orders.
2. Add GroupBox to the Main Form. Change the GroupBox Size property to 264, 77, the Location property to 6, 20, and the Text property to filters.
3. Add Label to the Main Form, change the Text property to Category, and the Location property to 6, 25.
4. Add ComboBox to the Main Form. Change the Name Property to cbProductsCategory, and the Location property to 61, 20.
5. Add Label to the Main Form. Change the Text property to Country and the Location property to 6, 51.
6. Add ComboBox to the Main Form. Change the Name property to cbOrdersCountry and the Location property to 61, 47.
7. Add Button to the Main Form. Change the Name property to btnProductsOrdersByFilters, the Location property to 108, 103, and the
Designing a Sample Application

**Text property to By filters.**

The final design looks like the following screenshot:

![Screenshot of the design](image)

**Employees and orders**

In this part we will see the relationship between employees and orders in order to evaluate the performance of each employee. In this section we will filter data by a period of time.

**Beginning your design process**

In this section we will start to design the last section in our application employees and orders.

1. From **Toolbox** explorer, drag-and-drop **GroupBox** to the Main Form. Change the **Size** property of GroupBox to 284,175, the **Location** property to 14,346, and the **Text** property to Employees - Orders.

2. Add **GroupBox** to the Main Form. Change the GroupBox **Size** property to 264,77, the **Location** property to 6,25, and the **Text** property to filters.

3. Add **Label** to the Main Form. Change the **Text** property to From and the **Location** property to 7,29.

4. Add **DateTimePicker** to the Main Form and change the **Name** property to dtpFrom. Change the **Location** property to 41,25, the **Format** property to Short, and the **Value** property to 1/1/1996.

5. Add **Label** to the Main Form. Change the **Text** property to To and the **Location** property to 7,55.

6. Add **DateTimePicker** to the Main Form. Change the **Name** property to dtpTo, the **Location** property to 41,51, the **Format** property to Short, and the **Value** property to 12/30/1996.

7. Add **Button** to the Main Form. Change the **Name** property to btnBarChart, the **Location** property to 15,117, and the **Text** property to Bar chart.
8. Add Button to the Main Form and change the Name property to btnPieChart. Change the Location property to 99, 117 and the Text property to Pie chart.

9. Add Button to the Main Form. Change the Name Property to btnGaugeChart, the Location property to 186, 117, and the Text property to Gauge chart.

10. Add Button to the Main Form. Change the Name property to btnAllInOne, the Location property to 63, 144, and the Text property to All In One.

The final design looks like the following screenshot:

![Screen Shot](https://example.com/screenshot.png)

You can change Location, Size, and Text properties as you wish; but don’t change the Name property because we will use it in code in the next chapters.

**Validating form data**

When you start designing your application, you need to think of some facts, such as performance and graphical user interface (GUI). One of the most important rules you need to take into account to build a good application is the ease of use. You need to make sure that the user is comfortable with your application. Reduce the amount of data that the user has to enter into your application. In our application therefore, you will see that we use GroupBox more than Textbox because selecting data is easy and safe when we use GroupBox. We will see how to fill all GroupBox controls with the corresponding data in the next chapter.
Sketching your reports on paper

Our application will be kept simple and complete and will be designed as a single form with common reporting types. All reports will be displayed in another form containing a Crystal Report Viewer as we will see in the next chapters. Every button in the Main Form is designed to display a report.

- The btnEmpAll button will display a report containing all employees' data and it will look like the following screenshot:

```
<table>
<thead>
<tr>
<th>ID</th>
<th>Full Name</th>
<th>Title</th>
<th>Hire Date</th>
<th>Country</th>
<th>City</th>
<th>reports To</th>
</tr>
</thead>
<tbody>
<tr>
<td>1</td>
<td>Ms. Nancy Davids</td>
<td>Sales Representative</td>
<td>06/01/1992</td>
<td>USA</td>
<td>Seattle</td>
<td>Dr. Andrew Fuller</td>
</tr>
<tr>
<td>2</td>
<td>Dr. Andrew Fuller</td>
<td>Vice President, Sales</td>
<td>08/14/1992</td>
<td>USA</td>
<td>Tacoma</td>
<td></td>
</tr>
<tr>
<td>3</td>
<td>Mr. John Smith</td>
<td>Sales Representative</td>
<td>04/01/1993</td>
<td>USA</td>
<td>Kirkland</td>
<td>Dr. Andrew Fuller</td>
</tr>
<tr>
<td>4</td>
<td>Ms. Margaret Peacock</td>
<td>Sales Representative</td>
<td>05/03/1993</td>
<td>USA</td>
<td>Redmond</td>
<td>Dr. Andrew Fuller</td>
</tr>
<tr>
<td>5</td>
<td>Mr. Steven Buchanan</td>
<td>Sales Manager</td>
<td>10/17/1990</td>
<td>UK</td>
<td>London</td>
<td>Dr. Andrew Fuller</td>
</tr>
<tr>
<td>6</td>
<td>Mr. Michael Suyama</td>
<td>Sales Representative</td>
<td>10/17/1990</td>
<td>UK</td>
<td>London</td>
<td>Mr. Steven Buchanan</td>
</tr>
<tr>
<td>7</td>
<td>Mr. Robert King</td>
<td>Sales Representative</td>
<td>01/02/1994</td>
<td>UK</td>
<td>London</td>
<td>Mr. Stewart Buchanan</td>
</tr>
<tr>
<td>8</td>
<td>Ms. Laura Callahan</td>
<td>Inside Sales Coordinator</td>
<td>03/06/1994</td>
<td>USA</td>
<td>Scotts</td>
<td>Dr. Andrew Fuller</td>
</tr>
<tr>
<td>9</td>
<td>Ms. Anne Batwolth</td>
<td>Sales Representative</td>
<td>11/10/1994</td>
<td>UK</td>
<td>London</td>
<td>Mr. Steven Buchanan</td>
</tr>
</tbody>
</table>
```

- The btnEmpById button will display a report containing the data for a specific employee and it will look like the following screenshot:

```
<table>
<thead>
<tr>
<th>ID</th>
<th>Full Name</th>
<th>Title</th>
<th>Hire Date</th>
<th>Country</th>
<th>City</th>
<th>Home phone</th>
<th>Notes</th>
</tr>
</thead>
<tbody>
<tr>
<td>4</td>
<td>Mrs. Margaret Peacock</td>
<td>Sales Representative</td>
<td>05/03/1993</td>
<td>USA</td>
<td>Redmond</td>
<td>205-555-8123</td>
<td>Margaret holds a BA in English literature from Concordia College (1989) and an MA from the American Institute of Culinary Arts (1998). She was assigned to the London office temporarily from July through November 1992.</td>
</tr>
</tbody>
</table>

Employee orders:
```
<table>
<thead>
<tr>
<th>Order ID</th>
<th>Customer ID</th>
<th>Order Date</th>
<th>Shipped Date</th>
<th>Ship Country</th>
<th>Ship City</th>
</tr>
</thead>
<tbody>
<tr>
<td>10,352</td>
<td>HNAR</td>
<td>7/6/95</td>
<td>7/23/95</td>
<td>Brazil</td>
<td>Rio de Janeiro</td>
</tr>
<tr>
<td>10,352</td>
<td>SUIRO</td>
<td>7/6/95</td>
<td>7/14/95</td>
<td>Belgium</td>
<td>Charleroi</td>
</tr>
<tr>
<td>10,357</td>
<td>HLAAA</td>
<td>7/10/96</td>
<td>7/22/95</td>
<td>Venezuela</td>
<td>San Cristobal</td>
</tr>
</tbody>
</table>
```
• The btnEmpByFilters button will display a report containing employee data and filter these employees using data in cbEmpTitle, cbEmpCity, and the cbEmpCountry GroupBoxes as we see in the following screenshot:

<table>
<thead>
<tr>
<th>ID</th>
<th>Full Name</th>
<th>Title</th>
<th>Hire Date</th>
<th>Country</th>
<th>City</th>
<th>reports To</th>
</tr>
</thead>
<tbody>
<tr>
<td>1</td>
<td>Mt. Nancy Davolio</td>
<td>Sales Representative</td>
<td>05/01/1992</td>
<td>USA</td>
<td>Seattle</td>
<td>Dr. Andrew Fuller</td>
</tr>
<tr>
<td>3</td>
<td>Mr. Janet Levering</td>
<td>Sales Representative</td>
<td>04/01/1992</td>
<td>USA</td>
<td>Kirkland</td>
<td>Dr. Andrew Fuller</td>
</tr>
<tr>
<td>4</td>
<td>Mr. Margaret Peacock</td>
<td>Sales Representative</td>
<td>05/03/1993</td>
<td>USA</td>
<td>Redmond</td>
<td>Dr. Andrew Fuller</td>
</tr>
</tbody>
</table>

• The btnProductsOrdersByFilters button will display a cross-tab report to see the relationship between products and orders. We will filter the data using data in cbProductsCategory and the cbOrdersCountry GroupBoxes, as shown in the following screenshot:

<table>
<thead>
<tr>
<th></th>
<th>UK</th>
</tr>
</thead>
<tbody>
<tr>
<td><strong>Beverages</strong></td>
<td></td>
</tr>
<tr>
<td>Total</td>
<td>502</td>
</tr>
<tr>
<td>Colchester</td>
<td>112</td>
</tr>
<tr>
<td>Cowes</td>
<td>100</td>
</tr>
<tr>
<td>London</td>
<td>290</td>
</tr>
<tr>
<td><strong>Outback Lager</strong></td>
<td></td>
</tr>
<tr>
<td>Total</td>
<td>119</td>
</tr>
<tr>
<td>Colchester</td>
<td>53</td>
</tr>
<tr>
<td>Cowes</td>
<td>50</td>
</tr>
<tr>
<td>London</td>
<td>16</td>
</tr>
<tr>
<td><strong>Guarani Fantástica</strong></td>
<td></td>
</tr>
<tr>
<td>Total</td>
<td>105</td>
</tr>
<tr>
<td>Colchester</td>
<td>25</td>
</tr>
<tr>
<td>Cowes</td>
<td>0</td>
</tr>
<tr>
<td>London</td>
<td>80</td>
</tr>
</tbody>
</table>
• The `btnBarChart` button will display a bar chart report to see relations between employees and orders. We will filter data using the data in `dtpFrom` and the `dtpTo` DateTimePicker, as shown in the following screenshot:

![Bar Chart Screenshot](image)

• The `btnPieChart` button will display a pie chart report to display the relationship between employees and orders. We will filter data using data in `dtpFrom` and the `dtpTo` DateTimePicker as shown in the following screenshot.

![Pie Chart Screenshot](image)
• The btnGaugeChart button will display a gauge chart report showing the relationship between employees and orders. We will filter data using data in the dtpFrom and the dtpTo DateTimePicker as shown in the following screenshot:
• The btnAllInOne button will display a report similar to the following screenshot:
Summary
In this chapter we discussed the design of our application. The application contains most Crystal Reports types. In the next chapter, we will start the development of our application. We will also learn how to select the data we need to display before creating our report.
Selecting the Data

Reports are formatted presentation of data. Reports are used to present the information stored in database(s). Hence the first step is to connect to a database to read data from it. In this chapter, we will see how we can connect to the database and run SQL queries using Microsoft Visual Studio. We will also talk about DataSet and TableAdapter and see how we can use them to fill the ComboBoxes in our application.

In this chapter, we will cover:

- SQL query in Visual Studio
- Populate the ComboBoxes with data

SQL queries in Microsoft Visual Studio

Generally SQL queries are categorized into data definition language and data manipulation language.

There are many types of SQL queries such as Select, Update, Delete, and Insert query. In this book we will talk only about the Select query because reports use the Select query. We will use Northwind database in our example queries and we will use Microsoft Visual Studio to run all SQL queries. So first, we need to connect to our database.

Connecting to our database

1. Open our application, Monitor, in Microsoft Visual Studio.
2. Click on View and click on Server Explorer.
3. In Server Explorer, right-click on Data Connections and select Add Connection.... A dialog box similar to the one shown in the next screenshot will appear:

![Add Connection dialog box]

4. Make sure that Data source: is Microsoft SQL Server(SqlClient).
5. Because we have installed Microsoft SQL Server on our machine, we can type in localhost as the server name or write the machine name.
6. Select Use SQL Server Authentication and enter sa as the User name: and enter your password that you used while installing Microsoft SQL Server.
7. Enter Northwind in the database field and click on the Test Connection button.
8. If a message box pops up saying that the test succeeded, it means we can go ahead. If not, reenter the data or troubleshoot if there is any connection problem with SQL Server.

9. Click on the **OK** button to connect to the database, and we will see our database in the **Server Explorer**, as shown in to the following screenshot:

![Server Explorer screenshot]

**Add database diagram**

To understand the Northwind database and relation between its tables we will create a new database diagram:

1. Right-click on **Database Diagrams** and choose **Add New Diagram**. Click on **Yes** in the confirmation message box

2. Select all tables in the **Add Tables** dialog and click on the **Add** button. We can see all the tables and relations between these tables in the next diagram.
3. Save the diagram with the default name Diagram1.
Test queries

We need a tool to run queries and test results. Microsoft Visual Studio has a good IDE to run SQL queries. We can see this IDE by right-clicking on any table name in our database in Server Explorer and selecting New Query. Then we will choose the tables that we will use in our query from the Add Table dialog box and click on the OK button. Refer to the following screenshot:

As we can see in the previous screenshot, there are four sections in our Query IDE. Section one displays all tables we used in our query, section two displays simple query builder, section three displays SQL query code, and section four displays the query result. In our discussion we will talk about sections three and four.
Populate the ComboBoxes with data

In our application, we have many ComboBoxes. How do we populate them with data? In this section we will populate the ComboBoxes with data and talk about the basics of select SQL query. We have four main steps to populate any ComboBox in our application: SQL query, TableAdapter, DataSource, and Load Event. Before we start filling the first ComboBox, we need to add DataSet to our application to save the retrieved data by TableAdapter, in to it.

Add DataSet to the application

1. Right-click on the Monitor application in the Solution Explorer and navigate to Add | New Item.
2. From the dialog box select DataSet and we will keep it with the default name DataSet1.

Populate the cbEmpTitle ComboBox

1. **Select data**: We need to populate this ComboBox with all employees' titles without any duplication.
   
   As we can see in the following screenshot, we can use a Select query to select Title from Employees table and the DISTINCT keyword to avoid duplication.

```
SELECT DISTINCT Title
FROM Employees
```

<table>
<thead>
<tr>
<th>Title</th>
</tr>
</thead>
<tbody>
<tr>
<td>Inside Sales Coordinator</td>
</tr>
<tr>
<td>Sales Manager</td>
</tr>
<tr>
<td>Sales Representative</td>
</tr>
<tr>
<td>Vice President, Sales</td>
</tr>
</tbody>
</table>

2. **Create TableAdapter**: Open `DataSet1`, right-click on it and navigate to **Add** | **TableAdapter**, as shown in the following screenshot:

![Choose Your Data Connection](image)

In this step, we can create a new connection string to our database or use the connection string that we created to connect to our database in **Server Explorer**. Here we will use the one we created before and then click on the **Next** button; as we will see the following screenshot:

![Choose a Command Type](image)

In this step, we can create a new connection string to our database or use the connection string that we created to connect to our database in **Server Explorer**. Here we will use the one we created before and then click on the **Next** button; as we will see the following screenshot:
Selecting the Data

In our application, we will use the SQL statement to access the database, so select **Use SQL Statements** and click on the **Next** button to see the following screenshot:

In this dialog we will write the SQL query that we prepared in the previous step and click on the **Next** button to see the following screenshot:
In this dialog we will change the method names to FillTitle and GetTitle then click on the Next button to see the following screenshot:

![Wizard Results]

We need to see this dialog to be sure that all the processes are configured successfully, then click on the Finish button to see the TableAdapter shown in the following screenshot:

![TableAdapter Screenshot]

In the previous screenshot, we changed the DataTable name to EmpTitle and TableAdapter name to EmpTitleTableAdapter.
Selecting the Data

3. **Set comboboxDataSource**: Open the Main Form; right-click on it and select **View Code**. We will create a new method to publish our ComboBox data. Refer to the following screenshot:

```csharp
private void PublishEmpTitle()
{
    DataSet ds = new DataSet1();
    EmpTitleTableAdapter Adapter = new EmpTitleTableAdapter();
    Adapter.FillTitle(ds.EmpTitle);
    cbEmpTitle.DataSource = ds.Tables["EmpTitle"]; 
    cbEmpTitle.DisplayMember = "Title";
}
```

In the `PublishEmpTitle` method, we create the object from `DataSet1` named `ds` and create another object from `EmpTitleTableAdapter` named `Adapter`. Then we fill our dataset `ds` by `tableadapterAdapter` using the `FillTitle` method. In the end, we assign the `cbEmpTitleDataSource` to `ds` and `DisplayMember` to `Title`.

4. **Update Load event**: In the Main Form Load event `MainForm_Load`, we will call our method `PublishEmpTitle` as we can see in the following screenshot:

```csharp
private void MainForm_Load(object sender, EventArgs e)
{
    PublishEmpTitle();
}
```

5. **Test the result**: Run our application to test the result. We can see that our ComboBox title is now filled with data; as seen in the following screenshot:
Populate the cbEmpCity ComboBox

1. **Select data**: We need to fill this ComboBox with all cities without any duplication. Refer to the following screenshot:

![Select DISTINCT City](image)

2. **Create TableAdapter**: Please follow the same steps as in the last section to create the TableAdapter and name it as EmpCity. The TableAdapter will look like the following screenshot:

![EmpCity TableAdapter](image)

3. **Set comboboxDataSource**: Refer to the code in the following screenshot:

```csharp
private void PublishCities()
{
    DataSet ds = new DataSet();
    EmpCityTableAdapter Adapter = new EmpCityTableAdapter();
    Adapter.FillCity(ds.EmpCity);
    cbEmpCity.DataSource = ds.Tables["EmpCity"];        cbEmpCity.DisplayMember = "City";
}
```
Selecting the Data

4. **Update Load event**: Refer to the code in the following screenshot:

```csharp
private void MainForm_Load(object sender, EventArgs e)
{
    PublishEmpTitle();
    PublishCities();
}
```

5. **Test the result**: The result will look like the following screenshot:

![Screenshot](image)

Populate the cbEmpCountry ComboBox

1. **Select data**: We need to fill this ComboBox with all employee countries without any duplication, as shown in the following screenshot:

![Screenshot](image)
2. **Create TableAdapter**: Please follow the same steps as before to create the TableAdapter. The TableAdapter will look like the following screenshot:

![TableAdapter Screenshot](image)

3. **Set comboboxDataSource**: Refer to the code in the following screenshot:

```csharp
private void PublishEmpCountries()
{
    DataSet1 ds = new DataSet1();
    EmpCountryTableAdapter Adapter = new EmpCountryTableAdapter();
    Adapter.FillCountry(ds.EmpCountry);  
    cbEmpCountry.DataSource = ds.Tables["EmpCountry"];
    cbEmpCountry.DisplayMember = "Country";
}
```

4. **Update Load event**: Refer to the code in the following screenshot:

```csharp
private void MainForm_Load(object sender, EventArgs e)
{
    PublishEmpTitle();
    PublishCities();
    PublishEmpCountries();
}
```
5. **Test the result:** The result will look like the following screenshot:

![Employee details screenshot]

**Populate the cbProductsCategory ComboBox**

1. **Select data:** We need to fill this ComboBox with all product categories without any duplication. Refer to the following screenshot:

![Product categories screenshot]

---

[58]
2. **Create TableAdapter**: Again, please follow the same steps as before to create the TableAdapter. The TableAdapter will look like the following screenshot:

![TableAdapter Screenshot](image)

3. **Set comboBoxDataSource**: Refer to the code in the following screenshot:

```csharp
private void PublishCategories()
{
    DataSet1 ds = new DataSet1();
    CategoriesTableAdapter Adapter = new CategoriesTableAdapter();
    Adapter.FillCategories(ds.Categories);
    cbProductsCategory.DataSource = ds.Tables["Categories"];  
    cbProductsCategory.DisplayMember = "CategoryName";
}
```

4. **Update Load event**: Refer to the code in the following screenshot:

```csharp
private void MainForm_Load(object sender, EventArgs e)
{
    PublishEmpTitle();
    PublishCities();
    PublishEmpCountries();
    PublishCategories();
}
```

5. **Test the result**: The result will look like the following screenshot:

![Test Result Screenshot](image)
Selecting the Data

Populate cbOrdersCountry ComboBox

1. **Select data**: We need to fill this ComboBox with all the countries in the Customers table without any duplication. Refer to the following screenshot:

2. **Create TableAdapter**: Please follow the same steps as before to create the TableAdapter. The TableAdapter will look like the following screenshot:

3. **Set comboboxDataSource**: Refer to the code in the following screenshot:

```csharp
private void PublishCustomerCountries()
{
    DataSet ds = new DataSet();
    CustomerCountryTableAdapter adapter = new CustomerCountryTableAdapter();
    adapter.FillCountry(ds.CustomerCountry);
    cbOrdersCountry.DataSource = ds.Tables["CustomerCountry"];  
    cbOrdersCountry.DisplayMember = "Country";
}
```
4. **Update Load event**: To do this, please refer to the code in the following screenshot:

```csharp
private void MainForm_Load(object sender, EventArgs e)
{
    PublishEmpTitle();
    PublishCities();
    PublishEmpCountries();
    PublishCategories();
    PublishCustomerCountries();
}
```

5. **Test the result**: The result will look like the following screenshot:

![Screenshot of test result]

**Summary**

Valid data is more critical than the report design, so we need to select the right data before displaying this data in the report. Before we start any reporting application, we need to understand the database/data very well to ensure that our report has the correct data. In this chapter, we learned how to write our Select query in Visual Studio and how to run this query. We learned how to fill the ComboBox using the DataAdapter. In the next chapter, we begin to use Crystal Reports and we will see how to display this report in our application.
In this chapter, we will see how to create a simple report and display this report in our application. We will begin by preparing the report data and connecting that data to our report. Then we will formulate our report data and display the resulting report in our application.

We will cover the following topics:

• Preparing a report data
• Adding a report to the application
• Using a database expert
• Choosing a report template
• Creating report sections
• Working with lines and boxes
• Using special fields
• Adding the report to a form
Preventing a report data

Before we begin to add Crystal Reports to our application, we need to understand the data that we will be displaying and know what data will be displayed in this report. Look at the design of the report in the following screenshot:

<table>
<thead>
<tr>
<th>ID</th>
<th>Full Name</th>
<th>Title</th>
<th>Hire Date</th>
<th>Country</th>
<th>City</th>
<th>reports To</th>
</tr>
</thead>
<tbody>
<tr>
<td>1</td>
<td>Ms. Nancy Davolio</td>
<td>Sales Representative</td>
<td>08/01/1992</td>
<td>USA</td>
<td>Seattle</td>
<td>Dr. Andrew Fuller</td>
</tr>
<tr>
<td>2</td>
<td>Dr. Andrew Fuller</td>
<td>Vice President, Sales</td>
<td>06/14/1992</td>
<td>USA</td>
<td>Tacoma</td>
<td>Dr. Andrew Fuller</td>
</tr>
<tr>
<td>3</td>
<td>Ms. Janet Levering</td>
<td>Sales Representative</td>
<td>04/01/1993</td>
<td>USA</td>
<td>Kirkland</td>
<td>Dr. Andrew Fuller</td>
</tr>
<tr>
<td>4</td>
<td>Mrs. Margaret Peacock</td>
<td>Sales Representative</td>
<td>05/03/1993</td>
<td>USA</td>
<td>Redmond</td>
<td>Dr. Andrew Fuller</td>
</tr>
<tr>
<td>5</td>
<td>Mr. Steven Buchanan</td>
<td>Sales Manager</td>
<td>10/17/1993</td>
<td>UK</td>
<td>London</td>
<td>Dr. Andrew Fuller</td>
</tr>
<tr>
<td>6</td>
<td>Mr. Michael Suyama</td>
<td>Sales Representative</td>
<td>10/17/1993</td>
<td>UK</td>
<td>London</td>
<td>Mr. Steven Buchanan</td>
</tr>
<tr>
<td>7</td>
<td>Mr. Robert King</td>
<td>Sales Representative</td>
<td>01/02/1994</td>
<td>UK</td>
<td>London</td>
<td>Mr. Steven Buchanan</td>
</tr>
<tr>
<td>8</td>
<td>Ms. Laura Callahan</td>
<td>Inside Sales Coordinator</td>
<td>03/05/1994</td>
<td>USA</td>
<td>Seattle</td>
<td>Dr. Andrew Fuller</td>
</tr>
<tr>
<td>9</td>
<td>Ms. Anne Dodsworth</td>
<td>Sales Representative</td>
<td>11/15/1994</td>
<td>UK</td>
<td>London</td>
<td>Mr. Steven Buchanan</td>
</tr>
</tbody>
</table>

Selecting the report data

As we can see in the preceding screenshot, all report data comes from the Employees table. The SELECT query will appear as shown in the following screenshot:

```
SELECT EmployeeID, TitleOfCourtesy, FirstName, LastName, Title, HireDate, Country, City, ReportsTo
FROM Employees
```

When we compare the report data with the data retrieved through the query, we will find two differences changes:
• **The Full Name column**: The Full Name column is displayed in the report as one part but in the database it consists of three parts (TitleOfCourtesy, FirstName, and LastName). We can address this problem here in the `SELECT` query or in the report itself. We can concatenate the three fields into one field using the SQL query by the plus (+) operator, as shown in the following screenshot:

```sql
SELECT EmployeeID, TitleOfCourtesy + FirstName + LastName AS FullName,
       Title, HireDate, Country, City, ReportsTo
FROM Employees;
```

<table>
<thead>
<tr>
<th>EmployeeID</th>
<th>FullName</th>
<th>Title</th>
<th>HireDate</th>
<th>Country</th>
<th>City</th>
<th>ReportsTo</th>
</tr>
</thead>
<tbody>
<tr>
<td>1</td>
<td>Ms. Nancy Davolio</td>
<td>Sales Rep.</td>
<td>5/1/1992</td>
<td>USA</td>
<td>Seattle</td>
<td>2</td>
</tr>
<tr>
<td>2</td>
<td>Dr. Andrew Fuller</td>
<td>Vice Presid.</td>
<td>8/14/1992</td>
<td>USA</td>
<td>Tacoma</td>
<td>NULL</td>
</tr>
<tr>
<td>3</td>
<td>Ms. Janet Levering</td>
<td>Sales Rep.</td>
<td>4/1/1992</td>
<td>USA</td>
<td>Kirkland</td>
<td>2</td>
</tr>
<tr>
<td>4</td>
<td>Mrs. Margaret Peacock</td>
<td>Sales Rep.</td>
<td>5/3/1993</td>
<td>USA</td>
<td>Redmond</td>
<td>2</td>
</tr>
<tr>
<td>5</td>
<td>Mr. Steven Buchanan</td>
<td>Sales Mgr.</td>
<td>10/17/1993</td>
<td>UK</td>
<td>London</td>
<td>2</td>
</tr>
</tbody>
</table>

• **The Reports To column**: The Reports To column is displayed in the report as a name, but as we can see in the preceding screenshot it is displayed as number. If we look at the `Employee` table we will see that this table has a self-join relationship, see the following screenshot:
We can modify our query to display names rather than numbers in two ways:

- **Using subquery:** We will update the query to look like the following screenshot:

```
SELECT EmployeeID,
      TitleOfCourtesy + ' ' + FirstName + ' ' + LastName AS fullname,
      Title,
      HireDate,
      City,
      Country,
      ReportsTo AS reports_to_fullname
FROM Employees AS emp
WHERE (emp.EmployeeID = Employees.ReportsTo) AS reports_to_fullname
```

- **Using self join:** We will update the query to look like the following screenshot:

```
SELECT Emp_1.EmployeeID,
      Emp_1.TitleOfCourtesy + ' ' + Emp_1.FirstName + ' ' + Emp_1.LastName AS fullname,
      Emp_1.Title,
      Emp_1.HireDate,
      Emp_1.City,
      Emp_1.Country,
      Emp_2.EmployeeID
FROM Employees AS Emp_1 INNER JOIN
     Employees AS Emp_2 ON Emp_1.ReportsTo = Emp_2.EmployeeID
WHERE (Emp_1.EmployeeID = Emp_2.EmployeeID)
```

```
Creating a TableAdapter
Here are the steps for creating a TableAdapter:

1. Open **DataSet1** from **Solution Explorer**.
2. Right-click on it and navigate to **Add | TableAdapter**.
3. Select your connection settings and click on the **Next** button.
4. Select the **Use SQL Statements** option and click on the **Next** button.
5. Write our selection query as we see in the preceding screenshot and click on the **Next** button.
6. Click on the **Next** button on the subsequent screens and click on the **Finish** button for the last screen.
7. Change your adapter name as shown in the following screenshot:

![Employees](https://via.placeholder.com/150)

Adding a report to the application
In this section, we will see how to add Crystal Reports to our application to display all **Employee** data from the database.
Creating First Report

1. Right-click on our application (Monitor) in Solution Explorer and navigate to Add | New Item. We will get a result as shown in the following screenshot:

![Add New Item - Monitor screenshot]

2. From the menu on the left-hand side, select Reporting and select Crystal Reports from the options on the right-hand side. Change the name to crAllEmployees.rpt and click on the Add button. We will get a result as shown in the following screenshot:

![Crystal Reports Gallery screenshot]
3. We have three options to create our report. From the preceding screenshot, we select **Standard** and use the report wizard to create the report. We will be using the other methods later and they will be explained in the following chapters. Click on the **OK** button to see the following screenshot:

![Standard Report Creation Wizard](image)

4. In this step, we will choose our TableAdapter that we created before to be used in this report. We will expand **Project Data** | **ADO.NET DataSets** | **Monitor.DataSet1** and choose the **Employees** TableAdapter and move it to the **Selected Tables** list. Then click on the **Next** button as shown in the following screenshot:

![Standard Report Creation Wizard](image)
5. In this step, we will choose the fields that we need to display in our report, we can add/remove these fields any time we need to after the report creation. Select all the required fields, move them to **Fields To Display** menu, and click on the **Next** button to see the following screenshot:

![Fields To Display](image1.png)

6. In this step we can add a group to our report, but we don't need any grouping in this report. We will use grouping in other reports and we will see that we can add any group after report creation. Click on the **Next** button to see the following screenshot:

![Grouping](image2.png)
7. In this step we can add a filter to our report, but we don’t need any filter in this report. We will use filters in other reports and we can add any filter after the report is created. Click on the Next button to see the following screenshot:

8. In this step we will choose the report style. You can browse the styles available and choose the one you like. We will choose the Standard style and click on the Finish button to see the following screenshot:
9. This report has five main sections:
   - **Section 1 (Report Header):** Any data (text, image, and so on) inserted in this section will be displayed in the report for one time in the top of the first paper only
   - **Section 2 (Page Header):** Any data (text, image, and so on) inserted in this section will be displayed in the header of each page in the report
   - **Section 3 (Details):** This section displays the data from the data source containing multiple rows of returned results like Microsoft SQL Server in our sample
   - **Section 4 (Report Footer):** Any data (text, image, and so on) inserted in this section will be displayed for one time in the end of the last page in the report only
   - **Section 5 (Page Footer):** Any data (text, image, and so on) inserted in this section will be displayed in the footer of each page in the report

10. We need to add *All Employees* text as a header to our report so we will add it in the **Report Header** section:
   1. Right-click on **Section2 (Report Header)** and navigate to **Insert | Text Object**.
   2. Double-click on **Text Object** and enter *All Employees*.
   3. Right-click on **Text Object** and choose **Format Object**, navigate to the **Font** tab, and change **Style to Bold** and **Size to 12**. Navigate to the **Paragraph** tab, change **Alignment to Centered** and click on the **OK** button.
   4. Move the **Text Object** to the horizontal middle of the section.

11. We will format all report objects and fields as we did with *All Employees Text Object* so the report will look like the following screenshot:
Working with lines and boxes
In this section we will see how to use boxes and lines to enhance our report design.

1. Right-click on Section 2 (Page Header) and navigate to Insert | Box and draw a box around our fields as shown in the following screenshot:

2. Right-click on Section 2 (Page Header) and navigate to Insert | Line and draw a line as shown in the following screenshot:

3. Add some lines between fields, as shown in the following screenshot:
Creating First Report

Using special fields
In Crystal Reports, there are a lot of special fields that we can use in our reports; in this section we will see how to use these special fields.

We need to add the page number and total page number in the page footer to display the page location in the total page count. The steps for this purpose are as follows:

1. Right-click in our report and navigate to Field Explorer.
2. From Field Explorer, open the Special Fields node and drag Page N of M to Section5 (Page Footer). The Page N of M special field will replace N with the page number and M with the total pages count when we run the report. See the following screenshot:

![Field Explorer Screenshot]

Adding the report to a form
In the preceding steps, we have completed the report design; the next step is displaying this report when the user clicks on the btnEmpAll button.

1. First, we will add a form that we will use to display all reports; we will name this form ReportForm.
3. Double-click on btnEmpAll to navigate to the earlier code. We will see the click event of the btnEmpAll button.
4. In the `click` event of the button `btnEmpAll`, enter the code as shown in the following screenshot:

```csharp
private void btnEmpAll_Click(object sender, EventArgs e)
{
    DataSet ds = new DataSet();
    EmployeesTableAdapter adapter = new EmployeesTableAdapter();
    adapter.Fill(ds.Employees);

    crAllEmployees report = new crAllEmployees();
    report.SetDataSource(ds);

    ReportForm form = new ReportForm();
    ((CrystalReportViewer)form.Controls["crystalReportViewer1"]).ReportSource = report;
    form.ShowDialog();
}
```

5. This code has three parts:
   - **Prepare report data**: In the first three lines of code, we fill the dataset by the TableAdapter (`EmployeesTableAdapter`).
   - **Pass data to our report**: In the next two lines of code, we will pass the dataset to our report after we fill it with the data.
   - **Display report in form**: In the last three lines of code we pass our report to `crystalReportViewer1` in `ReportForm` and open this form to display our report.

6. Run our application (Press F5) and click on the `btnEmpAll` button (All) to see our report, as shown in the following screenshot:

```
<table>
<thead>
<tr>
<th>ID</th>
<th>full name</th>
<th>Title</th>
<th>Hire Date</th>
<th>Country</th>
<th>City</th>
<th>Reports to</th>
</tr>
</thead>
<tbody>
<tr>
<td>1</td>
<td>Ms. Nancy Davolio</td>
<td>Sales Representative</td>
<td>05/01/1992</td>
<td>USA</td>
<td>Seattle</td>
<td>Dr. Andrew Fuller</td>
</tr>
<tr>
<td>2</td>
<td>Dr. Andrew Fuller</td>
<td>Vice President, Sales</td>
<td>08/14/1992</td>
<td>USA</td>
<td>Tacoma</td>
<td></td>
</tr>
<tr>
<td>3</td>
<td>Ms. Janet Leverling</td>
<td>Sales Representative</td>
<td>04/01/1992</td>
<td>USA</td>
<td>Kirkland</td>
<td>Dr. Andrew Fuller</td>
</tr>
<tr>
<td>4</td>
<td>Mrs. Margaret Peacock</td>
<td>Sales Representative</td>
<td>05/03/1993</td>
<td>USA</td>
<td>Redmond</td>
<td>Dr. Andrew Fuller</td>
</tr>
<tr>
<td>5</td>
<td>Mr. Steven Buchanan</td>
<td>Sales Manager</td>
<td>10/17/1993</td>
<td>UK</td>
<td>London</td>
<td>Dr. Andrew Fuller</td>
</tr>
<tr>
<td>6</td>
<td>Mr. Michael Suyama</td>
<td>Sales Representative</td>
<td>10/17/1993</td>
<td>UK</td>
<td>London</td>
<td>Mr. Steven Buchanan</td>
</tr>
<tr>
<td>7</td>
<td>Mr. Robert King</td>
<td>Sales Representative</td>
<td>01/02/1994</td>
<td>UK</td>
<td>London</td>
<td>Mr. Steven Buchanan</td>
</tr>
<tr>
<td>8</td>
<td>Ms. Laura Callahan</td>
<td>Inside Sales Coordinator</td>
<td>03/05/1994</td>
<td>USA</td>
<td>Seattle</td>
<td>Dr. Andrew Fuller</td>
</tr>
<tr>
<td>9</td>
<td>Ms. Anne Dodsworth</td>
<td>Sales Representative</td>
<td>11/15/1994</td>
<td>UK</td>
<td>London</td>
<td>Mr. Steven Buchanan</td>
</tr>
</tbody>
</table>
```
Summary
In this chapter, we learned how to create a simple report and display this report in our application. We have seen how to connect the report to the database and how to display and format the data in the report. In the next chapter, we will create new reports to learn grouping and sorting data. We will also learn how to use parameters to filter the table data.
Enhancing Reports

In the previous chapter we saw how to create a simple report and display this report in our application. In this chapter, we will create a new report but this report will be more complex than the previous one. We will see how to group data using one or more fields, how to filter data by parameters, and how to add a subreport.

We will cover the following topics:

- Using parameter fields
- Working with formula fields
- Working with subreport
- Grouping and sorting data
- Adding the report to a form

In this chapter, we will have two sections. In the first section we will create a new report to display employees, like the one we had created in the previous chapter, but we will add some filters to this new report. This will allow the user to select those employees who have the same data. We see in the following screenshot that all employees in the report have the same data (Country, City, and Title). This can be achieved by using parameter fields. In the first section we will see how to use these.

<table>
<thead>
<tr>
<th>ID</th>
<th>Full Name</th>
<th>Title</th>
<th>Hire Date</th>
<th>Country</th>
<th>City</th>
<th>Reports To</th>
</tr>
</thead>
<tbody>
<tr>
<td>6</td>
<td>Mr. Michael Suyama</td>
<td>Sales Representative</td>
<td>10/17/1993</td>
<td>UK</td>
<td>London</td>
<td>Mr. Steven Buchanan</td>
</tr>
<tr>
<td>7</td>
<td>Mr. Robert King</td>
<td>Sales Representative</td>
<td>01/02/1994</td>
<td>UK</td>
<td>London</td>
<td>Mr. Steven Buchanan</td>
</tr>
<tr>
<td>9</td>
<td>Ms. Anne Dodsworth</td>
<td>Sales Representative</td>
<td>11/15/1994</td>
<td>UK</td>
<td>London</td>
<td>Mr. Steven Buchanan</td>
</tr>
</tbody>
</table>
Enhancing Reports

In the second section, we will create a new report to display all the Employee Details and their Employee orders as shown in the following screenshot. We will also see how to use subreports, grouping, sorting, and formula fields.

Using parameter fields

We will create a new report the same way we created one in the previous chapter, using the following steps:

1. We will use the same TableAdapter that we created in the previous chapter (EmployeesTableAdapter) as shown in the following screenshot:
2. Add a new Crystal Reports to our application and rename it to crFilteredEmployees.

3. As shown in the following screenshot, we will select the As a Blank Report radio button to learn how to connect to database with your report after the report creation. Click on the OK button.
4. From **Field Explorer**, right-click on **Database Fields** and choose **Database Expert**. Select the **Employees** table as shown in the following screenshot. Click on the **OK** button.

![Database Expert screenshot](image)

5. Now you will see the table **Fields** when you navigate to **Field Explorer** | **Database Fields** | **Employees** as shown in the following screenshot:

![Field Explorer screenshot](image)
6. Drag Employees fields to the Details section in our report and format the report, as shown in the following screenshot:

![Screenshot of report with Employees fields in Details section]

Until now our report would have displayed all employees, so now we need to update this report to filter employees by their data (Country, City, and Title). In the next steps we will see how to add filters to our report.

7. In Field Explorer, right-click on Parameter Fields and select New.

As we see in the following screenshot, we will name this parameter pmCountry and select its Type as String. We will keep the default values for other fields in the form and click on the OK button:

![Screenshot of Create New Parameter dialog box]
8. Create two more parameters with the type **String** and name them `pmCity` and `pmTitle`. Now we have three parameters, as shown in the following screenshot:

![Field Explorer](image1)

Until now we hadn't used these parameters and they don't have any effect in our report.

9. Right-click on the report and navigate to **Report | Selection Formula | Record.** We will see this in the following screenshot:

![Formula Workshop](image2)
10. We will write some code to apply parameters to our reports, as shown in the following screenshot:

- In the first line of code: `{Employees.Title}={?pmTitle}`, we filtered Employees Title by pmTitle parameter.
- In the second line of code `{Employees.Country}={?pmCountry}`, we filtered Employees Country by pmCountry parameter.
- In the third line of code `{Employees.City}={?pmCity}`, we filtered Employees City by pmCity parameter.
- At the end we tied the three lines with the and keyword to ensure that the three filters must be applied together; if one filter failed the others will fail.

You can drag-and-drop Table and Parameter fields to the code box from the report fields in the upper box (field tree).
11. Save and close the form from the top.

Until now we finished our report design, now we need to display our report in the application; when the user clicks `btnEmpByFilters` button (By Filters), the values of Title, City, and Country ComboBoxes are dynamically passed to our report parameters `pmTitle`, `pmCity`, and `pmCountry`.

12. Open MainForm from Solution Explorer. Double-click on the `btnEmpByFilters` button to move to the click event code and write the code that we see in the following screenshot:

```csharp
private void btnEmpByFilter_Click(object sender, EventArgs e)
{
    ReportForm form = new ReportForm();
    crfFilteredEmployees report = new crfFilteredEmployees();
    DataSet ds = new DataSet();
    EmployeesTableAdapter adapter = new EmployeesTableAdapter();
    adapter.Fill(ds.Employees);
    report.SetDataSource(ds);
    report.SetParameterValue("pmTitle", cbEmpTitle.Text);
    report.SetParameterValue("pmCity", cbEmpCity.Text);
    report.SetParameterValue("pmCountry", cbEmpCountry.Text);
    ((CrystalReportViewer)form.Controls["crystalReportViewer1"]).ReportSource = report;
    form.ShowDialog();
}
```

We have seen most of the previously mentioned lines of code in our previous chapter, but here we added only three extra lines of codes to pass the three parameters to our report dynamically. The three lines of code are as follows:

```csharp
report.SetParameterValue("pmTitle", cbEmpTitle.Text);
report.SetParameterValue("pmCountry", cbEmpCountry.Text);
report.SetParameterValue("pmCity", cbEmpCity.Text);
```

13. When we run the application and test the result, we will see the report results, depending on what data was provided in the filters.

We need to improve our report so the user can use one filter or two rather than three filters, or he can stop all filters. We will allow the user to stop any filter by writing the "ALL" string in the combo boxes used to filter the report data. We only need to change the parameters code in our report to meet this improvement.

14. Right-click on any Parameter Fields in Field Explorer and select Find in Formula.
15. Update the code we wrote before to the new one as in the following screenshot then save and close.

\[
\begin{align*}
&\text{if}([?pmTitle]="ALL") \text{ then } (\text{Employees.Title}=[?pmTitle] \text{ else true}) \text{ and} \\
&\text{if}([?pmCountry]="ALL") \text{ then } (\text{Employees.Country}=[?pmCountry] \text{ else true}) \text{ and} \\
&\text{if}([?pmCity]="ALL") \text{ then } (\text{Employees.City}=[?pmCity] \text{ else true})
\end{align*}
\]

16. Run the application to test the new change. Type "ALL" in the three combo boxes used to filter the report data, and click on the By filters button. You will now see the data of all the employees in the report. You can change one or two filters and keep the others to test the report result.

In the next section, we will create a report to display the data of all the employees with the employee orders. This report contains a main report that consists of the employee data and another subreport that contains the employee orders. We will see how to use subreport by filtering data in the main report fields.

1. First we will create a TableAdapter that contains all the data from Employees table, the select query will look like the following screenshot:

```
SELECT emp1.*, emp2.TitleOfCourtesy AS ReportsTo_TitleOfCourtesy, emp2.FirstName AS ReportsTo_FirstName, emp2.LastName AS ReportsTo_LastName
FROM Employees AS emp1 LEFT OUTER JOIN Employees AS emp2 ON emp1.ReportsTo = emp2.EmployeeID
```

<table>
<thead>
<tr>
<th>EmployeeID</th>
<th>LastName</th>
<th>FirstName</th>
<th>Title</th>
</tr>
</thead>
<tbody>
<tr>
<td>1</td>
<td>Devolio</td>
<td>Nancy</td>
<td>Sales Rep</td>
</tr>
<tr>
<td>2</td>
<td>Fuller</td>
<td>Andrew</td>
<td>Vice Pres</td>
</tr>
<tr>
<td>3</td>
<td>Leverling</td>
<td>Janet</td>
<td>Sales Rep</td>
</tr>
<tr>
<td>4</td>
<td>Peacock</td>
<td>Margaret</td>
<td>Sales Rep</td>
</tr>
<tr>
<td>5</td>
<td>Buchanan</td>
<td>Steven</td>
<td>Sales Man</td>
</tr>
<tr>
<td>6</td>
<td>Suyama</td>
<td>Michael</td>
<td>Sales Rep</td>
</tr>
<tr>
<td>7</td>
<td>King</td>
<td>Robert</td>
<td>Sales Rep</td>
</tr>
<tr>
<td>8</td>
<td>Callahan</td>
<td>Laura</td>
<td>Inside Sales</td>
</tr>
<tr>
<td>9</td>
<td>Dodsworth</td>
<td>Anne</td>
<td>Sales Rep</td>
</tr>
</tbody>
</table>
Enhancing Reports

In the preceding query, we used **LEFT OUTER JOIN** to select all data from the left table that contains all employees' data and selected the supervisor (ReportsTo) data from the second table if it exists. We didn't use **INNER JOIN** because the employee with ID 2 (**Fuller**) does not have a supervisor (ReportsTo). We are also using **SELF JOIN** with the same table, which we learned in the previous chapters. The TableAdapter will look like the following screenshot:

2. We will create a new crystal report with the name **crEmployeeDetails** and pass **Employee1** as the report data source. Drag-and-drop data fields to the **Employee Details** section in our report and format the fields to look like the following screenshot:

Please note that in the previous reports we used SQL queries to display the employees' and supervisors' full names in our report. In this section we will use a formula field to display their full name. As we saw in the previous screenshot, **Full Name** and **Reports To** are formula fields (@fmu_FullName, @fmu_ReportTo). We will see how to use formula fields now.
Working with formula fields
In this section we will see how to use formula fields and how they are useful and powerful for improving our reports.

1. From Field Explorer, right-click on Formula Fields and select New. Type in the formula name fmu_FullName and click on the OK button. We will see this in the following screenshot:

   ![Formula Field Screenshot]

   This formula field will display the employee's full name in the same format that we created before using SQL query. Here we use the concatenate operator (+) to concatenate the title with the first and last name of the employee.

2. Create another formula field and name it fmu_ReportsTo. You will see the formula code in the following screenshot:

   ![Formula Field Screenshot]

3. Drag-and-drop the two formula fields to the report as we did previously.
4. Add a parameter field to the report and name it EmpId. We will use this parameter to filter the employees by ID as we see in the following screenshot:

   ![Parameter Field Screenshot]
5. Open **MainForm** from **Solution Explorer**. Double-click on the **btnEmpById** button to move to the click event code and write the code that we see in the following screenshot:

```csharp
private void btnEmpById_Click(object sender, EventArgs e)
{
    if (string.IsNullOrEmpty(txtEmpId.Text))
    {
        MessageBox.Show("Please insert employee id first");
    }
    else
    {
        ReportForm form = new ReportForm();
        crEmployeesDetails MainReport = new crEmployeesDetails();

        DataSet ds = new DataSet();
        EmployeesTableAdapter MainAdapter = new EmployeesTableAdapter();
        MainAdapter.Fill(ds, "Employees");
        MainReport.SetDataSource(ds);

        MainReport.SetParameterValue("EmpId", txtEmpId.Text);
        ((CrystalReportViewer)form.Controls["crystalReportViewer1"]).ReportSource = MainReport;
        form.ShowDialog();
    }
}
```

6. Run the application to test our report, write an ID number in the **txtEmpId** textbox and click on the **By id** button to test the result.

With this step we finished the main report that contains the employee data; in the next several steps we will add a subreport to the main report, which will contain the employee orders.

**Working with subreport**

Now we will see another powerful tool in Crystal Reports; that is subreport. Subreport is used to create a report inside another report and link the two reports with a relation or a filter.
1. Create TableAdapter, which contains the employee orders and all the data in the table Orders. We will rename the adapter to OrdersByEmployeeIdTableAdapter as we see in the following screenshot:

![OrdersByEmployeeIdTableAdapter](image1)

2. In the main report, in the Details section, right-click and navigate to Insert | Subreport. Drag-and-drop this subreport at the end of the main report (after the Notes field in the Employee Details section). Then we will see the following screenshot:

![Insert Subreport](image2)
As we see, we can choose the subreport from the application reports, export it from hard disk, or create a new report using Report Wizard. We will choose the last option, so click on the Report Wizard... button and complete the wizard as we have learned before. Make sure that you use the OrdersByEmployeeId table as a data source for the subreport. We will name the subreport crOrdersByEmployeeId and click on the OK button.

3. Right-click on the subreport and select Edit Subreport. In subreport we will group the Employee orders by the country so in the next few steps we will see how to use grouping in Crystal Reports.

**Grouping and sorting data**

In this section we will see how to group and sort report data with one or more fields.

1. In Field Explorer, right-click on Group Name Fields and select Insert Group. We will see this in the following screenshot:

![Insert Group dialog box]

Choose the group field (ShipCountry) and the sort type (in ascending order). We can change the field title from the Options tab. Click on the OK button.
2. Format the subreport as we see in the following screenshot. You can see the two new sections that were added to the report after we created the group.

![Subreport Screenshot]

Now we need to filter data in the subreports using the parameter field in the main report.

3. Go back to the main report, right-click on the subreport, and select the Change subreport Links.... We will see the following screenshot:

![Subreport Links Screenshot]
Enhancing Reports

In this dialog, we will map the EmpId parameter field from the main report to EmployeeID in the subreport. From the Available Fields list, we will select the EmpId parameter and move it to the Field(s) to link to: list. In the last part in the dialog we will choose EmployeeID from the right drop-down list to link it with the EmpId parameter in the left drop-down list. Click on the OK button.

Now the two reports are connected together with one parameter: EmpId. The last step is to fill the subreport with data before displaying your report to the user.

4. Navigate back to the button click event to update your code as shown in the following screenshot:

```csharp
private void btnEmpId_Click(object sender, EventArgs e)
{
    if (string.IsNullOrEmpty(txtEmpId.Text))
    {
        MessageBox.Show("Please insert employee id first");
    }
    else
    {
        ReportForm form = new ReportForm();
        crEmployeeDetailsMainreport = new crEmployeeDetails();

        DataSet ds = new DataSet();
        EmployeeIDTableAdapter MainAdapter = new EmployeeIDTableAdapter();
        MainAdapter.Fill(ds.EmployeeID);
        Mainreport.SetDataSource(ds);

        OrdersByEmployeeIDTableAdapter SubAdapter = new OrdersByEmployeeIDTableAdapter();
        SubAdapter.Fill(ds.OrdersByEmployeeID);
        Mainreport.Subreports[0].SetDataSource(ds);

        Mainreport.SetParameterValue("EmpId", txtEmpId.Text);
        form.ShowDialog();
    }
}
```

We only added three lines of code to the old code, the three lines to fill the subreport with data using the OrdersByEmployeeIDTableAdapter TableAdapter.
5. Run our application to test the last changes. The final report will look like the following screenshot:

6. As we see in the previous screenshot, the date format contains the time; we want to change this format to display the date only. Navigate back to our report, right-click on any date field in the report, select **Format Object**, and select the date format as shown in the following screenshot:
After changing the date format for all date fields (BirthDate, HireDate, OrderDate, and ShippedDate), the report will look like the following screenshot:

![Employee Details](image)

**Summary**

In this chapter, we created two reports and we learned from the first one how to use parameter fields and how to use these fields to filter the report data. In the second report we learned how to use subreports and how to use grouping and sorting. In the next chapter, we will create special type of reports and we will see how to create a crosstab report.
In this chapter we will see one of the most important type of Crystal Reports: the Cross-Tab report. It is one of the most powerful tools in Crystal Reports. A Cross-Tab report is used to summarize data efficiently as we will see in this chapter.

We will cover the following topics:

- Creating a Cross-Tab report
- Customizing the Cross-Tab appearance using the Cross-Tab Expert
- Doing page setup

The Cross-Tab report is very useful for the user because it summarizes a lot of information in a simple form to give the user the information he needs. We can see the final report as shown in the following screenshot:
As we see in the preceding screenshot, the report displays the sum of the product orders for each city. In our application, the user will select the product category and the country. The report will display the result shown in the previous screenshot.

Creating our report

So let's start creating our report:

1. The SQL query that we will use in our report is shown in the following screenshot:

2. We will create a new TableAdapter using this SQL query and name it ProductsOrdersTableAdapter, as seen in the following screenshot:
3. Add a new crystal report to our application and name it crProductsOrders. As we see in the following screenshot, we can create our report by selecting the Using the Report Wizard option and then choosing Cross-Tab as an Expert, or we can select As a Blank Report and we will add the Cross-Tab from the report itself. We will take the second option to see how we can add or modify our Cross-Tab at any time after creating the report, so we will choose As a Blank Report as shown in the following screenshot:
4. Right-click on **Database Fields** in **Field Explorer** and select **Database Expert**, and then a window appears, as shown in the following screenshot. As shown below, we will choose our TableAdapter **ProductsOrders** as a data source:
5. Right-click on the report and navigate to **Insert | Cross-Tab** and drag it to **Section1 (Report Header)**. As we see in the following screenshot, we will add **ShipCountry** and **ShipCity** to the **Columns** list, **CategoryName** and **ProductName** to the **Rows** list, and **Quantity** to **Summarized Fields** list. From the **Style** tab, we can choose any style, however need but the original style is preferred. We will see how to use the **Customize Style** tab in the next few steps. Click on the **OK** button.
6. Until now our report displays all the products and countries, so we will add two parameter fields to filter the report data by product categories and shipping countries. The first parameter field is `pmCountry` and the second one is `pmCategory`. See the following screenshot:

7. Add a text object to the report as the report header and enter Best-selling products for the countries in it.

8. Go to the main form in our application, double-click on the `btnProductsOrdersByFilters (By filter)` button to navigate to the code behind, and write the code to display the report in our application, as shown in the following screenshot:

```csharp
private void btnProductsOrdersByFilters_Click(object sender, EventArgs e)
{
    ReportForm form = new ReportForm();
    crProductsOrders report = new crProductsOrders();
    DataSet1 ds = new DataSet1();
    ProductsOrdersTableAdapter adapter = new ProductsOrdersTableAdapter();
    adapter.Fill(ds.ProductsOrders);
    report.SetDataSource(ds);
    report.SetParameterValue("pmCountry", cbOrdersCountry.Text);
    report.SetParameterValue("pmCategory", cbProductsCategory.Text);
    ((CrystalReportViewer)form.Controls["crystalReportViewer1"]).ReportSource = report;
    form.ShowDialog();
}
```
9. Run the application to test the result. Choose **Dairy Products** from the **Category** list and **UK** from the **Country** list as we see in the following screenshot. Click on the **By filters** button to display the report.

![Filters Screenshot]

10. The report will look like the following screenshot:

![Report Screenshot]
Customizing the Cross-Tab appearance

In the next few steps, we will see how to update the report to resemble the design at the beginning of this chapter.

As we saw in the preceding screenshot, the two last columns and the last two rows have the same date, so we will hide the last column and the last row in our report.

1. Right-click in the top-left of the Cross-Tab and select Cross-Tab Expert, and navigate to the Customize Style tab. We will select the Suppress Row Grand Totals and Suppress Column Grand Totals checkboxes as we see in the following screenshot. Click on the OK button.
2. Run the application to test the result. It will look like the following screenshot:

![Best-selling products for countries](image)

3. We can see in this report that the number of products displayed is more than the products that we have seen in the designed report in the beginning of this chapter. So, we need to sort the products according to the number of orders.

4. In the designed report at the beginning of this chapter, we displayed the two products that have the top rate of orders only, so the report header is **Best-selling products for countries** and not all the products. In the next step, we will see how to control the number of products displayed in our report.
5. Right-click on the top left of the Cross-Tab, select **Group Sort Expert**, and navigate to the **ProductsOrders.ProductName** tab as shown in the following screenshot:

![Group Sort Expert](image)

6. We will select the **Top N** value from the **For this group sort** list and enter 2 in the **where N is** textbox to display only two products in the Cross-Tab. Click on the **OK** button and run the application to see the result as shown in the following screenshot:

![Best-selling products for countries](image)
7. If we compare this report with the one designed in the beginning of this chapter, we will find no difference between them except the formatting of the text and the numbers. You can format it as per your need.

8. In the next step, we will see how we can change the format of the grid lines and the background colors of the cells.

9. Right-click on the top left of the Cross-Tab and select Cross-Tab Expert and navigate to the Customize Style tab. As we see in the following screenshot, we will select the row or column we want to change the background color of and select the color we need to apply to it.
10. Click on the **Format Grid Lines** button to format the Cross-Tab lines as we can see in the following screenshot. Select the line. The line color, style, or width can be altered as desired.

![Format Grid Lines](image1.png)

11. Click on the **OK** button to save the changes and run the application to test the new appearance. This is shown in the following screenshot:

![Best-selling products for countries](image2.png)
12. What if the user needs to display products of all categories to a specific country with the ability to filter data by the products category?

13. We will make a small change to apply this request: we will change the parameter condition in our report as shown in the following screenshot:

14. When the user writes string `ALL` in the category list, the report will display all categories. Run the application and test the change, it will look like the following screenshot:
Performing page setup

When we work with Cross-Tab we need to be careful with the page size because the width of the report grows dynamically. If we try to test the Cross-Tab by inputting the value USA, we will see that the report is divided into two parts as shown in the following screenshot:

This problem occurs because the value USA has many corresponding values for cities in our database and our report is designed to display and print in a portrait letter form. We can change the basic page settings by right-clicking on the report and navigating to Design | Page Setup. This is shown in the following screenshot. The page size has been changed to A3 and the Orientation to Landscape.
If we test by inputting the value USA again, the result will look like the following screenshot:

<table>
<thead>
<tr>
<th></th>
<th>Alaska</th>
<th>Anchorage</th>
<th>Boise</th>
<th>Butte</th>
<th>Elgin</th>
<th>Eugene</th>
<th>Portland</th>
<th>Portland</th>
<th>Seattle</th>
<th>Wichita</th>
<th>Total</th>
</tr>
</thead>
<tbody>
<tr>
<td>Dairy Products</td>
<td>Buttermilk</td>
<td>120</td>
<td>6</td>
<td>141</td>
<td>9</td>
<td>50</td>
<td>0</td>
<td>0</td>
<td>6</td>
<td>1</td>
<td>54</td>
</tr>
<tr>
<td>Yogurt</td>
<td>French Vanilla</td>
<td>31</td>
<td>34</td>
<td>125</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
</tr>
<tr>
<td>Total</td>
<td>251</td>
<td>42</td>
<td>285</td>
<td>23</td>
<td>98</td>
<td>0</td>
<td>94</td>
<td>23</td>
<td>178</td>
<td>10</td>
<td>1000</td>
</tr>
</tbody>
</table>

**Summary**

In this chapter we have seen how useful Cross-Tab is. We learned to create Cross-Tab report and use the Cross-Tab Expert to control every part of the Cross-Tab. In the next chapter, we will see another fantastic tool in Crystal Reports; we will see how we can create reports containing charts.
In this chapter we will see how to use charts in Crystal Reports. Crystal Reports supports many types of charts but we will create four reports that contain different samples of charts.

We will cover:

- Types of charts
- Creating charts
- Using the Chart Expert

We will use the same data table in our reports and see how to display this data in different charts. The first report will display data in a bar chart as we can see in the following screenshot:
The second report will display data in a pie chart as we can see in the following screenshot:

The third report will display data in a gauge chart as we can see in the following screenshot:
In the final report, we will see how to display different charts in one report. Refer to the following screenshot:

Creating the reports data source

In this section we will create the data source that we will use in all the reports in this chapter. All the reports in this chapter display the relation between employees and orders. These reports display the number of orders for each employee.
Working with Charts

1. The SQL query that we will use in our report looks like the following screenshot:

2. We will create a new TableAdapter using the previous SQL query and we will name it **EmployeeOrdersTableAdapter**, as shown the following screenshot:

---

Creating the first report – bar chart

Let’s start to create the first chart report. We will start with a bar chart.

1. Add a new Crystal Reports to our application and name it **crEmpPerBar**. We will choose the As a Blank Report option to see how we can add or modify our chart at any time after the report creation.
2. Right-click on Database Fields in Field Explorer and select the Database Expert option. As shown in the following screenshot, we will select our TableAdapter EmployeeOrders as a data source:

3. Right-click on the report and navigate to Insert | Chart and drag it in Section1 (Report Header). As we can see in the following screenshot, in the Type tab we will select Bar as a chart type and Side by side bar chart as the type of bar chart, from the submenu that comes up in the right window:
4. Navigate to the **Data** tab as we can see in the following screenshot:

![Chart Expert window](image)

5. In this step we will see how to display data in a bar chart. We need to display the name of the employee on the X-axis and the numbers of orders on the Y-axis. We will add the `EmployeeOrders.fullName` field to the upper-right list to display it on the X-axis and add the `EmployeeOrders.Quantity` field to the lower list to display it on the Y-axis. We can display only a specific number of employees using the TopN button as we have seen before in **Cross-Tab**. We will keep the other options in the other tabs and click on the **OK** button.

6. We want to filter the orders by a period of date so we will add two filter parameters to our report with type **Date**. We will name the first parameter `pmFrom` and the second one `pmTo`. Have a look at the following screenshot to see the filter code:

![Filter parameters](image)
7. Go to the main form and double-click on the `btnBarChart` button (Bar chart) to navigate to the code window behind it. Write the code as shown in the following screenshot for the `click` event:

```csharp
private void btnBarChart_Click(object sender, EventArgs e)
{
    ReportForm frm = new ReportForm();
    ctegoryBar report = new ctegoryBar();
    DataSet ds = new DataSet();
    EmployeeOrdersTableAdaptor adaptor = new EmployeeOrdersTableAdaptor();
    adaptor.Fill(ds, EmployeeOrders);
    report.SetDataSource(ds);
    report.SetParameterValue("mFrom", ds.Tables[0].Rows[0]["From Date"]);
    report.SetParameterValue("mTo", ds.Tables[0].Rows[0]["To Date"]);
    CrystalReportViewer1 Controls["CrystalReportViewer1"].ReportSource = report;
    form.ShowDialog();
}
```

8. Run the application to test the result. Enter the from and to dates as shown in the following screenshot:

9. Click on the Bar chart button to see the report result as shown in the following screenshot:

Now we will see how to update our report to get the required report, which we designed at the beginning of this chapter.
Removing the legend
The legend is the box that appears on the right side as shown in the previous screenshot; by default it's displayed beside the chart. In the following steps we will see how to remove it:

1. Right-click on the chart and navigate to Chart Options | General. Navigate to the Legend tab as shown in the following screenshot:

   ![Legend Settings](image)

   2. Uncheck the Show Legend checkbox. The dialog has many options that you can use. Try to change these options and test the result. Click on the OK button.

Removing the chart's title
In the next few steps we will see how to remove or change the default chart title:

1. Right-click on our chart and navigate to Chart Options | Titles, as shown in the following screenshot:
2. Clear the Title textbox and the Group Title textbox. Click on the OK button.

Changing the chart's appearance

The Crystal Reports allows us to modify the appearance of our charts to meet the user requirements. In the following steps we will see how to achieve this:

1. Right-click on our chart and navigate to Chart Options | General.
2. Navigate to the Appearance tab; update the appearance properties as shown in the following screenshot:
3. Select the **Use Depth** checkbox. Change the **Depth** value as shown in the preceding screenshot.
4. Resize the chart size in the report to fit the report width.
5. Run the application to test the result; we will see that the report is the same as the one we had designed before.

---

**Creating the second report – pie chart**

The second chart report that we will create is a pie chart. Let's start the work:

1. Add a new Crystal Reports to our application and name it `crEmpPerPie`. We will choose the **As a Blank Report** option.
2. Right-click on **Database Fields** in **Field Explorer** and select **Database Expert**. We will choose our TableAdapter `EmployeeOrders` as a data source.
3. Right-click on the report and navigate to **Insert | Chart** and drag it in **Section1 (Report Header)**. As shown in the following screenshot, in the **Type** tab we will select **Pie** as a chart type:
4. Move to the **Data** tab and select the data fields as we did in the previous report. Click on the **OK** button.

5. Add two **date** filters as we did in the previous report.

6. Go to the main form and double-click on the **btnPieChart** button (**Pie chart**) to navigate to the code window behind it. Write the code as shown in the following screenshot for the **click** event:

   ```csharp
   private void btnPieChart_click(object sender, EventArgs e)
   {
   ReportForm form = new ReportForm();
   crmEmpPie report = new crmEmpPie();
   DataSet ds = new DataSet();
   EmployeeOrdersTableAdapter adapter = new EmployeeOrdersTableAdapter();
   adapter.Fill(ds.EmployeeOrders);
   report.SetDataSource(ds);
   report.SetParameterValue("pmFrom", dtpFrom.Value.ToShortDateString());
   report.SetParameterValue("pmTo", dtpo.Value.ToShortDateString());
   ((CrystalReportViewer)form.Controls["CrystalReportViewer1"]).ReportSource = report;
   form.ShowDialog();
   }
   ```

7. Run the application to test the result. We will see the report result as shown in the following screenshot:

Now we will see how to update our report to meet the required report that we designed at the beginning of this chapter.
Removing the legend
In the next few steps we will remove the legend as we did before with the bar chart:

1. Right-click on our chart and navigate to Chart Options | General.
2. Navigate to the Legend tab.
3. Unselect the Show Legend checkbox.
4. Click on the OK button.

Removing the chart's title
In the next few steps we will remove the chart's title again, as we did before with the bar chart.

1. Right-click on our chart and navigate to Chart Options | Titles.
2. Clear the Title textbox.
3. Click on the OK button.

Changing the chart's appearance
We will change the chart's appearance as we did before with the bar chart by performing the following steps:

1. Right-click on our chart and navigate to Chart Options | General.
2. Navigate to the Appearance tab; update the appearance properties as shown in the following screenshot:

![Chart Options Screenshot]

3. Resize the chart size in the report to fit the report width.
Creating the third report – gauge chart

The next chart report that we will create is a gauge chart report. Let's start:

1. Add a new Crystal Reports to our application and name it `crEmpPerGauge`. We will choose the As a Blank Report option.

2. Right-click on Database Fields in Field Explorer and navigate to Database Expert. We will choose our TableAdapter `EmployeeOrders` as a data source.

3. Right-click on the report and navigate to Insert | Chart and drag it in Section1 (Report Header). As shown in the following screenshot, in the Type tab we will select Gauge as a chart type:

![Chart Expert](image)

4. Navigate to the Data tab and choose the data fields as we did in the first report. Click on the OK button.

5. Add the two date filters as we did in the first report.
6. Go to main form and double-click on **btnGaugeChart** button (Gauge chart) to navigate to the code window behind it. Write the code as shown in the following screenshot for the **click** event:

```csharp
private void btnGauge_Click(object sender, EventArgs e)
{
    ReportForm form = new ReportForm();
    crEmpPerGauge report = new crEmpPerGauge();
    DataSet ds = new DataSet();
    EmployeeOrdersTableAdapter adapter = new EmployeeOrdersTableAdapter();
    adapter.Fill(ds, EmployeeOrders);
    report.SetDataSource(ds);
    report.SetParameterValue("prmFrom", dtpFrom.Value.ToShortDateString());
    report.SetParameterValue("prmTo", dtpo.Value.ToShortDateString());
    ((CrystalReportViewer)form.Controls["crystalReportViewer1"]).ReportSource = report;
    form.ShowDialog();
}
```

7. Run the application to test the result. We will see the report result as shown in the following screenshot:

![Sum of Quantity / fullName chart](image)

Now we will see how to update our report to meet the report that we had designed at the beginning of this chapter.
Removing the legend
In the next few steps we will remove the legend as we did before with the bar chart:

1. Right-click on our chart and navigate to Chart Options | General.
2. Navigate to the Legend tab.
3. Deselect the Show Legend checkbox.
4. Click on the OK button.

Removing the chart's title
In the next few steps we will remove the chart's title as we did before with the bar chart:

1. Right-click on our chart and navigate to Chart Options | Titles.
2. Clear the Title textbox.
3. Click on the OK button.

Changing the chart's appearance
We will change the chart's appearance as we did before with the bar chart by performing the following steps:

1. Right-click on our chart and navigate to Chart Options | General.
2. Navigate to the Appearance tab; update the appearance properties as shown in the following screenshot:

![Chart Options Appearance](image)

3. Resize the chart size in the report to fit the report width.
Creating the fourth report – all in one

In this report we use different charts in one report and display the data with the charts.

1. Add new Crystal Reports to our application and name it crEmpAllInOne. We will choose the As a Blank Report option.
2. Right-click on Database Fields in Field Explorer and choose Database Expert. We will choose our TableAdapter EmployeeOrders as a data source.
3. Right-click on the report and navigate to Insert | Chart and drag it in Section1 (Report Header). In the Type tab we will select Bar and resize the chart to fit the width of the report.
4. Right-click on the report and navigate to Insert | Chart and drag it in Section1 (Report Header). In Type tab we will select Pie as a chart type and locate it under the Bar chart and resize the chart to fit the width of the report.
5. Right-click on the report and navigate to Insert | Chart and drag it in Section1 (Report Header). In Type tab we will select Gauge as a chart type. Locate it under the Pie chart and resize the chart to fit the width of the report.
6. Update the appearance of all the three charts as we did before.
7. Add two date filters as we did in the previous report.
8. Drag-and-drop the report fields to the Details section and format them as shown in the following screenshot.

---

```plaintext
<table>
<thead>
<tr>
<th>ID</th>
<th>fullName</th>
<th>Order ID</th>
<th>Quantity</th>
<th>Order Date</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
</tbody>
</table>
```
9. Go to main form and double-click on the **btnAllInOne** button (**All In One**) to move to the code window behind it. Write the code as shown in the following screenshot for the `click` event:

```csharp
class ReportForm
{
    private void btnAllInOne_Click(object sender, EventArgs e)
    {
        ReportForm form = new ReportForm();
        crEmpAllInOne report = new crEmpAllInOne();
        DataSet ds = new DataSet();
        EmployeeOrdersTableAdapter adapter = new EmployeeOrdersTableAdapter();
        adapter.Fill(ds.EmployeeOrders);
        report.SetDataSource(ds);
        report.SetParameterValue("mFrom", dtpFrom.Value.ToShortDateString());
        report.SetParameterValue("mTo", dtpTo.Value.ToShortDateString());
        ((CrystalReportViewer)form.Controls["CrystalReportViewer1"]).ReportSource = report;
        form.ShowDialog();
    }
}
```

10. Run the application to test the result. It will be the same as the report we designed earlier.

## Summary

In this chapter we learned how to use charts to build fantastic and powerful reports. We have seen that Crystal Reports support many types of charts and how easy it is to use these charts.

This chapter was the last one in our book. We learned how to create a reporting application from beginning to the end. I tried to give knowledge to the reader not only about tools such as Crystal Reports but also tried to teach them the skills and techniques that may help them to use these tools quickly and easily. We have seen how to design the application, and what tools we can use to create our application. We learned how to install our tools, how to use each tool, and how these tools communicate with each other. We learned Crystal Reports tools such as database expert, parameters, grouping, sorting, lines, and boxes. We learned to create different types of reports such as tabular, cross-tab and charts, and how to mix these types in one report. We learned how to display our reports in Visual Studio and passing parameter values dynamically using C# code.
Index

A
application
creating 29-33

B
bar chart report
about 111
creating 114-117
bar chart report updation
appearance, modifying of charts 119, 120
default chart title, removing 118
legend, removing 118
boxes
used, for enhancing report design 73
btnAllInOne button 42
btnBarChart button 40
btnEmpAll button 38
btnEmpByFilters button 39
btnEmpById button 38
btnGaugeChart button 41
btnPieChart button 40
btnProductsOrdersByFilters button 39

c
cbEmpCity ComboBox
populating 55, 56
cbEmpCountry ComboBox
populating 56-58
cbEmpTitle ComboBox
populating 50-54
cbOrdersCountry ComboBox
populating 60, 61
cbProductsCategory ComboBox
populating 58, 59

c
charts
using, in single report 126
ComboBoxes
populating, with data 50
controls
adding, to form 33-37
Cross-Tab report
about 95
appearance, customizing 102-107
creating 96-101
page setup, performing 108, 109
Crystal Reports
about 111
adding, to application 67-72
installing, for Visual Studio 2012 8

D
data
ComboBoxes, populating with 50
displaying, with charts 126
database diagram
adding 47
queries, running 49
dataSet
adding, to application 50

F
form
controls, adding to 33-37
report, adding to 74, 75
form data
validating 37
formula fields
used, for improving reports 87, 88
gauge chart report
about 112
creating 123, 124
gauge chart report updation
appearance, modifying of charts 125
default chart title, removing 125
legend, removing 125

installation, Crystal Reports for Visual Studio 2012
installation, confirming 12
installation language, selecting 11
license agreement, reading 12
source, obtaining 8-10
successful installation 13
welcome screen 11
installation, Microsoft SQL Server 2008
Analysis Services Configuration 21
Database Engine Configuration 20
disk space requirements 19
Error and Usage Reporting 22
Installation Complete 22
Installation Progress 22
Installation Rules 22
Instance Configuration 19
license terms 17
product key 16
Ready to Install 22
Reporting Services Configuration 22
Server Configuration 20
setup support rules 16
source, obtaining 13, 15
SQL Server Installation Center 15
support files, setting up 17
support rules, setting up 17, 18
installation, Microsoft Visual Studio 2012
installation process 8
license agreement, reading 6
optional features 7
source, obtaining 5
successful setup 8
installation, Northwind database
connecting, to SQL server 2008 26
database, installing 26
installation, confirming 25
installation, completing 26
installation options, selecting 25
license agreement, reading 24
source, obtaining 22, 23
SQL Server Management Studio,
opening 26
Welcome dialog 24

lines
used, for enhancing report design 73

Microsoft SQL Server 2008
installing 13
Microsoft Visual Studio 2012
installing 5

Northwind database
creating, in SQL Server 2008 27

page setup
performing, for Cross-Tab report 108, 109

parameter fields
using 78-86

pie chart report
about 112
creating 120, 121
pie chart report updation
appearance, modifying of charts 122
default chart title, removing 122
legend, removing 122

queries
running 49
R

report data
  grouping 90-94
  preparing 64
  selecting 64, 65
  sorting 90-94

reports
  about 45
  adding, to form 74, 75
  sketching, on paper 38-42

reports data source
  creating 113, 114

S

self join
  using 66

special fields
  using 74

SQL queries, in Microsoft Visual Studio
  about 45
  connecting, to database 45, 46

subquery
  using 66

subreport
  working with 88, 90

T

TableAdapter
  creating, steps 67
Thank you for buying
Reporting with Visual Studio and Crystal Reports

About Packt Publishing
Packt, pronounced 'packed', published its first book "Mastering phpMyAdmin for Effective MySQL Management" in April 2004 and subsequently continued to specialize in publishing highly focused books on specific technologies and solutions.

Our books and publications share the experiences of your fellow IT professionals in adapting and customizing today's systems, applications, and frameworks. Our solution based books give you the knowledge and power to customize the software and technologies you're using to get the job done. Packt books are more specific and less general than the IT books you have seen in the past. Our unique business model allows us to bring you more focused information, giving you more of what you need to know, and less of what you don't.

Packt is a modern, yet unique publishing company, which focuses on producing quality, cutting-edge books for communities of developers, administrators, and newbies alike. For more information, please visit our website: www.packtpub.com.

About Packt Enterprise
In 2010, Packt launched two new brands, Packt Enterprise and Packt Open Source, in order to continue its focus on specialization. This book is part of the Packt Enterprise brand, home to books published on enterprise software – software created by major vendors, including (but not limited to) IBM, Microsoft and Oracle, often for use in other corporations. Its titles will offer information relevant to a range of users of this software, including administrators, developers, architects, and end users.

Writing for Packt
We welcome all inquiries from people who are interested in authoring. Book proposals should be sent to author@packtpub.com. If your book idea is still at an early stage and you would like to discuss it first before writing a formal book proposal, contact us; one of our commissioning editors will get in touch with you.

We're not just looking for published authors; if you have strong technical skills but no writing experience, our experienced editors can help you develop a writing career, or simply get some additional reward for your expertise.
Learning SQL Server 2008 Reporting Services

A step-by-step guide to getting the most of Microsoft SQL Server Reporting Services 2008

1. Everything you need to create and deliver data-rich reports with SQL Server 2008 Reporting Services as quickly as possible
2. Packed with hands-on-examples to learn and improve your skills
3. Connect and report from databases, spreadsheets, XML Data, and more
4. No experience of SQL Server Reporting Services required

Mastering SQL Queries for SAP Business One

Utilize the power of SQL queries to bring Business Intelligence to your small to medium-sized business

1. Practical SAP query examples from an SAP Business One expert
2. Detailed steps to create and troubleshoot SQL queries for Alerts, Approvals, Formatted Searches, and Crystal Reports
3. Understand the importance and benefit of keeping SQL queries simple and easy to understand

Please check www.PacktPub.com for information on our titles
Visual Studio 2012 Cookbook

50 simple but incredibly effective recipes to immediately get you working with the exciting features of Visual Studio 2012

1. Take advantage of all of the new features of Visual Studio 2012, no matter what your programming language specialty is!

2. Get to grips with Windows 8 Store App development, .NET 4.5, asynchronous coding and new team development changes in this book and e-book

3. A concise and practical First Look Cookbook to immediately get you coding with Visual Studio 2012

Learning SQL Server Reporting Services 2012

Get the most out of SQL Server Reporting Services 2012, both Native and SharePoint Integrated modes

1. Build applications using the latest Microsoft technologies: SSIS 2012, SSDT, WPF, and SharePoint 2010

2. Reach out to the cloud and master Windows Azure Reporting Services

3. Learn the ins and outs of SQL Server Reporting Services 2012 for Native and SharePoint Integrated modes