



Quick answers to common problems

# Pentaho Business Analytics Cookbook

Over 100 recipes to get you fully acquainted with the key features of Pentaho BA 5 and increase your productivity

**Sergio Ramazzina**

**[PACKT]** open source\*  
PUBLISHING community experience distilled

[www.allitebooks.com](http://www.allitebooks.com)

# Pentaho Business Analytics Cookbook

Over 100 recipes to get you fully acquainted with the key features of Pentaho BA 5 and increase your productivity

**Sergio Ramazzina**

**[PACKT]** open source   
PUBLISHING community experience distilled

BIRMINGHAM - MUMBAI

# **Pentaho Business Analytics Cookbook**

Copyright © 2014 Packt Publishing

All rights reserved. No part of this book may be reproduced, stored in a retrieval system, or transmitted in any form or by any means, without the prior written permission of the publisher, except in the case of brief quotations embedded in critical articles or reviews.

Every effort has been made in the preparation of this book to ensure the accuracy of the information presented. However, the information contained in this book is sold without warranty, either express or implied. Neither the author, nor Packt Publishing, and its dealers and distributors will be held liable for any damages caused or alleged to be caused directly or indirectly by this book.

Packt Publishing has endeavored to provide trademark information about all of the companies and products mentioned in this book by the appropriate use of capitals. However, Packt Publishing cannot guarantee the accuracy of this information.

First published: June 2014

Production reference: 1190614

Published by Packt Publishing Ltd.  
Livery Place  
35 Livery Street  
Birmingham B3 2PB, UK.

ISBN 978-1-78328-935-6

[www.packtpub.com](http://www.packtpub.com)

Cover image by Gagandeep Sharma ([er.gagansharma@gmail.com](mailto:er.gagansharma@gmail.com))

# Credits

**Author**

Sergio Ramazzina

**Project Coordinator**

Neha Thakur

**Reviewers**

David Fombella Pombal

Ankur Gupta

Alex Meadows

**Proofreaders**

Maria Gould

Ameesha Green

Paul Hindle

**Commissioning Editor**

Kartikey Pandey

**Indexers**

Mehreen Deshmukh

Tejal Soni

**Acquisition Editors**

Anthony Albuquerque

Rebecca Youé

**Graphics**

Ronak Dhruv

Valentina Dsilva

**Content Development Editor**

Priya Singh

**Production Coordinators**

Manu Joseph

Saiprasad Kadam

**Technical Editors**

Shubhangi H. Dhamgaye

Rosmy George

Manal Pednekar

Anand Singh

**Cover Work**

Manu Joseph

Saiprasad Kadam

**Copy Editors**

Roshni Banerjee

Dipti Kapadia

Insiya Morbiwala

Aditya Nair

Stuti Srivastava

# About the Author

**Sergio Ramazzina** is an experienced software architect/trainer with more than 25 years of experience in the IT field. He has worked on a broad number of projects for banks and major Italian companies and has designed complex enterprise solutions in Java, JavaEE, and Ruby. He started using Pentaho products from the very beginning in late 2003. He gained thorough experience by deploying Pentaho as an open source BI solution, standalone or deeply integrated in other applications as the analytical engine of choice.

In 2009, due to his experience in the Java/JavaEE world and appreciation for the open source world and its main ideas, he began participating actively as a contributor to some of the Pentaho projects such as JPivot, Saiku, CDF, and CDA and rose to the Pentaho Active Contributor level. At that time, he started participating as a BI architect and Pentaho expert on a wide number of projects where open source BI and Pentaho were the main players. In late 2010, he founded Serasoft, a young Italian consulting firm that specializes in delivering high value open source Business Intelligence solutions. With the team in Serasoft, he shared his passion and experience in designing and delivering highly innovative enterprise solutions to help users make their work more effective. In July 2013, he published his first book, *Instant Pentaho Data Integration Kitchen*, Packt Publishing. He is also passionate about skiing, tennis, and photography, and he loves his young daughter, Camilla, very much.

You can follow him on Twitter at @sramazzina. You can also look at his profile on LinkedIn at <http://it.linkedin.com/in/sramazzina/>.

# Acknowledgments

Firstly, I want to thank my little daughter, Camilla, because every time she would see her dad working on the book, either late in the evening or early in the morning, she would always plant a wonderful kiss and say some words of encouragement.

Thanks to my colleagues and friends who continuously encouraged me in writing this book.

Thanks to my wonderful friends from the Pentaho Community who, during these years, helped me understand the inner workings of the product. A big, special thanks to Pedro Alves, Pentaho SVP for Community and Webdetails founder, for the help he gave me in getting access to a Pentaho EE license for the time required to write this book and for quickly answering me, on a Sunday afternoon, about an unexpected issue I had on CDE while writing its recipe. Thanks a lot, Pedro!

Another big thanks to all my technical reviewers for the wonderful work they did in reviewing my book and in suggesting areas to be improved. Thanks to Packt Publishing for trusting me with this new adventure and thanks to all of the staff at Packt Publishing that supported and assisted me during the writing of the book.

# About the Reviewers

**David Fombella Pombal** was born in Oviedo, Spain, in 1987. He earned his Bachelor's degree in Computer Science from the University of Oviedo, developing a business analytics asset management system as the final project using Pentaho and open source database tools.

He has dedicated more than five years to developing Business Intelligence solutions. At the start of his career, he developed analytical solutions using SAP Business Objects software in the banking industry. Over the last four years, he has been dedicated to developing business analytics solutions full time using Pentaho Suite across all industries (banking and finance, insurance, aerospace, and so on). He is an open source enthusiast and maintains a cool blog with quick tips about Pentaho tricks. He also trains clients on the use of Pentaho Suite.

You can follow him on Twitter at @pentaho\_fan. He has technically proofread and reviewed *Mondrian in Action*, *Manning Publications* and *Pentaho Reporting Video Course*, Packt Publishing.

**Ankur Gupta** was born and raised in Lucknow, India. He belongs to a family of four: he, his mother, his father, and his younger brother. His father is an engineer, his mother is a lecturer, and his brother is pursuing MBBS. He is full of life and his hobbies are cyber gaming, coding, and exploring new technologies and tools. He always brings fun to whatever he does and is kind and always helps his colleagues and friends.

He completed his schooling from Lucknow and, from a very young age, developed a keen interest in computers and decided to pursue his career in it. Since childhood, he was acquainted with computers, video games, outdoor camping as well as adventurous trips that included rafting, deep sea diving, cliff jumping, and many more. He has traveled almost all parts of India and is looking forward to going abroad.

Ankur currently works as a software developer at Mindfire Solutions, India. He received a B.Tech in Computer Science in the year 2012 from ICFAI University, Dehradun. Apart from being a software developer, he is a writing enthusiast and is a regular contributor to the Jaspersoft community. He leads an individual blogging site as well, in which he writes blogs both on technical and general topics. He also runs a YouTube video blog that contains a series of tutorial videos on JasperReports. He believes in the *knowledge grows when shared* ideology. So, in turn, he evangelizes his knowledge about various technologies through his blogs and videos. In the near future, Ankur is looking forward to pursuing his Master's after gaining strong experience in his field.

He has a vision to establish a non-profit school and an advanced computer education center in his native city, Lucknow. The center and school would be places that impart free primary education as well as advanced computer education to lesser privileged kids. His base line is that computer education would be very essential in near future, and for unprivileged kids, it is of paramount importance that they receive computer education to be in the league with all other kids so that they can lead a prosperous life.

---

I pay my sincere respect and thanks to God for all my achievements so far and in upcoming future. I would like to thank my parents for giving me support and a wonderful life. I also thank my friends and colleagues for always supporting me.

---

**Alex Meadows** has worked with open source Business Intelligence solutions for nearly 10 years and has worked in various industries, such as plastics manufacturing, social and e-mail marketing, and most recently, with software at Red Hat, Inc. He has been very active in Pentaho and other open source communities and is keen to learn, share, and help newcomers with the best practices in BI, analytics, and data management. He received his Bachelor's degree in Business Administration from Chowan University, Murfreesboro, North Carolina, and his Master's degree in Business Intelligence from St. Joseph's University, Philadelphia, Pennsylvania. He recently updated *Pentaho Data Integration Cookbook, Packt Publishing*, to its second edition.



# www.PacktPub.com

## Support files, eBooks, discount offers, and more

You might want to visit [www.PacktPub.com](http://www.PacktPub.com) for support files and downloads related to your book.

Did you know that Packt offers eBook versions of every book published, with PDF and ePub files available? You can upgrade to the eBook version at [www.PacktPub.com](http://www.PacktPub.com) and as a print book customer, you are entitled to a discount on the eBook copy. Get in touch with us at [service@packtpub.com](mailto:service@packtpub.com) for more details.

At [www.PacktPub.com](http://www.PacktPub.com), you can also read a collection of free technical articles, sign up for a range of free newsletters and receive exclusive discounts and offers on Packt books and eBooks.



<http://PacktLib.PacktPub.com>

Do you need instant solutions to your IT questions? PacktLib is Packt's online digital book library. Here, you can access, read and search across Packt's entire library of books.

## Why Subscribe?

- ▶ Fully searchable across every book published by Packt
- ▶ Copy and paste, print and bookmark content
- ▶ On demand and accessible via web browser

## Free Access for Packt account holders

If you have an account with Packt at [www.PacktPub.com](http://www.PacktPub.com), you can use this to access PacktLib today and view nine entirely free books. Simply use your login credentials for immediate access.

# Table of Contents

<b>Preface</b>	<b>1</b>
<b>Chapter 1: Getting Familiar with Pentaho User Console</b>	<b>7</b>
Introduction	8
Adding content to favorites	8
Accessing a solution's folders and files	11
Creating a new solution folder	13
Renaming a solution's folder	15
Moving a folder to Trash	17
Showing basic folder properties	18
Changing a folder's permissions	20
Renaming a file in a folder	23
Showing a basic file's properties	24
Changing a file's permissions	27
Moving a file to Trash	28
Moving a file to a different folder	29
Restoring content items from Trash	30
Permanently deleting content from Trash	31
Uploading content to a solution folder	33
<b>Chapter 2: Configuring Your BA Server Instance</b>	<b>39</b>
Introduction	40
Accessing the Administration perspective	40
Creating a new user	42
Deleting an existing user	44
Editing an existing user	45
Creating a new role	47
Deleting an existing role	52
Editing an existing role	53
Managing system roles	55

<b>Configuring authentication through the LDAP server (EE version)</b>	<b>57</b>
<b>Configuring authentication through the LDAP server (CE version)</b>	<b>64</b>
<b>Managing the mail server configuration</b>	<b>69</b>
<b>Cleaning up aged generated files immediately</b>	<b>72</b>
<b>Scheduling the cleanup of aged generated files</b>	<b>73</b>
<b>Chapter 3: Defining BA Server Data Sources</b>	<b>77</b>
<b>Introduction</b>	<b>77</b>
<b>Creating a new native JDBC data source</b>	<b>78</b>
<b>Defining a JNDI connection in the BA Server</b>	<b>83</b>
<b>Creating a new JNDI JDBC data source</b>	<b>87</b>
<b>Updating an existing JDBC data source</b>	<b>91</b>
<b>Creating a new analysis data source</b>	<b>92</b>
<b>Updating an existing analysis data source</b>	<b>96</b>
<b>Creating a new metadata data source</b>	<b>97</b>
<b>Exporting an existing data source</b>	<b>100</b>
<b>Creating a new data source from a CSV file using the wizard</b>	<b>101</b>
<b>Deleting an existing data source</b>	<b>104</b>
<b>Chapter 4: Defining Business Models with the Pentaho Metadata Editor</b>	<b>107</b>
<b>Introduction</b>	<b>108</b>
<b>Using a JNDI connection for development</b>	<b>110</b>
<b>Managing JDBC database connections</b>	<b>113</b>
<b>Defining the physical layer</b>	<b>118</b>
<b>Defining concepts</b>	<b>121</b>
<b>Reviewing physical layer tables' columns</b>	<b>126</b>
<b>Deriving business models from the physical layer</b>	<b>129</b>
<b>Reviewing business tables' column properties</b>	<b>133</b>
<b>Applying formatting properties to business tables' fields</b>	<b>138</b>
<b>Adding new calculated columns to model entities</b>	<b>139</b>
<b>Defining joins between business tables' entities</b>	<b>142</b>
<b>Creating business view categories</b>	<b>146</b>
<b>Testing metadata layer results</b>	<b>149</b>
<b>Applying security to the domain model elements</b>	<b>153</b>
<b>Publishing metadata definitions to BA Server</b>	<b>157</b>
<b>Chapter 5: Creating Reports Using Pentaho Interactive Reporting</b>	<b>161</b>
<b>Introduction</b>	<b>161</b>
<b>Creating a simple interactive report</b>	<b>162</b>
<b>Editing an existing report</b>	<b>169</b>
<b>Adding groups and totals to reports</b>	<b>171</b>
<b>Changing the labels in an interactive report</b>	<b>175</b>

---

Reorganizing table columns	176
Adding filters to limit a report's output	178
Adding prompts to get user input	181
Exporting reports in the Excel or PDF format	184
<b>Chapter 6: Creating Analysis Reports</b>	<b>185</b>
Introduction	185
Creating and publishing a Mondrian schema	187
Creating a new analysis report from scratch	195
Adding subtotals to rows' categories	199
Adding graphical indicators to table's cells	202
Changing the columns' sort order	204
Adding a simple calculated measure	207
Creating visualizations with Pentaho Analyzer	211
Exporting reports in the Excel or PDF format	215
Creating an analysis report using Saiku	216
<b>Chapter 7: Creating Reports Using Pentaho Report Designer</b>	<b>223</b>
Introduction	223
Configuring JNDI connections for development	225
Managing JDBC connections	227
Managing ETL connections	234
Using layouts to simplify report development	237
Using style sheets to consistently manage fields' formats	245
Changing field properties at runtime with formulas	250
Using input parameters	254
Using groups to define report aggregations	258
Using functions to add calculated fields	263
Using subreports to embed content	267
Embedding microcharts in reports with sparklines	270
Adding charts to our report	274
<b>Chapter 8: Creating Dashboards</b>	<b>279</b>
Introduction	279
Creating a simple dashboard from scratch	281
Adding prompts to get user input	287
Creating a multiple-content dashboard	291
Linking different content and enabling interaction	293
Creating dashboards using CDE	296
<b>Chapter 9: Scheduling Content</b>	<b>305</b>
Introduction	305
Scheduling task execution	306
Updating schedule properties	311

<b>Deleting schedules</b>	<b>313</b>
<b>Executing a scheduled task ahead of time</b>	<b>314</b>
<b>Filtering schedules in the schedule entries list</b>	<b>316</b>
<b>Running a user's tasks in the background</b>	<b>318</b>
<b>Checking schedule execution</b>	<b>320</b>
<b>Stopping running executions</b>	<b>321</b>
<b>Preventing the creation of schedules by content</b>	<b>323</b>
<b>Preventing the creation of schedules by setting blockout time intervals</b>	<b>325</b>
<b>Chapter 10: Working with Pentaho Mobile BI</b>	<b>329</b>
<b>Introduction</b>	<b>330</b>
<b>Accessing BA server from a mobile device</b>	<b>331</b>
<b>Accessing folders and files</b>	<b>335</b>
<b>Adding files to favorites</b>	<b>337</b>
<b>Changing the default startup screen</b>	<b>339</b>
<b>Chapter 11: Customizing Pentaho BA to Meet Your Business Needs</b>	<b>343</b>
<b>Introduction</b>	<b>343</b>
<b>Adding a company's logo to the Pentaho User Console login page</b>	<b>344</b>
<b>Using themes to customize Pentaho User Console</b>	<b>347</b>
<b>Adding new languages to Pentaho User Console</b>	<b>357</b>
<b>Checking Pentaho BA Server logs from inside of Pentaho User Console</b>	<b>360</b>
<b>Easily managing content in Pentaho Solution</b>	<b>362</b>
<b>Index</b>	<b>369</b>

# Preface

Pentaho has become one of the most commonly used open source BI platforms during the last few years. In recent times, a lot of different books came out that covered topics on the Pentaho platform, but none of these gave a complete overview of the platform and its tools in one place to help users become productive quickly.

*Pentaho Business Analytics Cookbook* is the first book to go deep in Pentaho with a reasonable level of detail of all the pieces of the platform in one single book. By going through all of the book's recipes, you will gain the necessary knowledge to be productive in a very short period of time in order to start using this platform effectively. The book also covers topics such as Pentaho Metadata Editor and Pentaho Mobile, making this book the first comprehensive source of information on Pentaho.

## What this book covers

*Chapter 1, Getting Familiar with Pentaho User Console*, introduces the reader to the new Pentaho User Console. Because of the new look and feel of the Pentaho User Console, this chapter is a good overview for novice users and also for navigated users who would like to quickly familiarize with the new user interface.

*Chapter 2, Configuring Your BA Server Instance*, goes into the details of the new Administration perspective, the place where administrative tasks should be carried out.

*Chapter 3, Defining BA Server Data Sources*, explains the new way to define data sources in Pentaho BA 5. We will go deep in the definition all of the major data source types by giving a complete and detailed description.

*Chapter 4, Defining Business Models with the Pentaho Metadata Editor*, describes the advantages of implementing a metadata domain model and explains, at a good level of detail, how to define a Pentaho metadata domain model from the ground up as well as shows how to deploy it.

*Chapter 5, Creating Reports Using Pentaho Interactive Reporting*, shows how to use a metadata domain model easily and how we can define a tabular report using the Pentaho User Console.

*Chapter 6, Creating Analysis Reports*, starts by giving a brief recap about what a Mondrian schema is and how to define it using Pentaho Schema Workbench. Then, it goes through the details of defining an analysis report by using Pentaho Analyzer and Saiku.

*Chapter 7, Creating Reports Using Pentaho Report Designer*, shows how to use the powerful Pentaho Report Designer to create wonderful reports easily and without difficulty.

*Chapter 8, Creating Dashboards*, goes through the design of interactive dashboards by using Pentaho Dashboard Designer and CDE.

*Chapter 9, Scheduling Content*, takes us through the details of the new scheduler interface and shows how to easily schedule our recurring jobs.

*Chapter 10, Working with Pentaho Mobile BI*, is the first instance where we have an overview of the Pentaho Mobile application, showing how our content can be easily accessed from a mobile device.

*Chapter 11, Customizing Pentaho BA to Meet Your Business Needs*, gives a brief introduction about how to customize the Pentaho User Console by defining new themes and presents some interesting plugins taken from Pentaho Marketplace that can help us in our everyday tasks.

## What you need for this book

Pentaho BA Server comes in two different versions, Community Edition and Enterprise Edition. The vast majority of the samples provided with the book can run on Pentaho BA Server Community Edition. *Chapters 5, Creating Reports Using Pentaho Interactive Reporting; Chapter 6, Creating Analysis Reports* (with the exception of the last recipe); *Chapter 8, Creating Dashboards* (with the exception of the last two recipes); and *Chapter 10, Working with Pentaho Mobile BI*, mandatorily require Pentaho Enterprise Edition to run the samples provided in the book.

The remaining software needed to run the samples are provided for free and are specified as follows:

- ▶ Pentaho Metadata Editor: This is downloadable from <http://community.pentaho.com/>
- ▶ Pentaho Schema Workbench: This is downloadable from <http://community.pentaho.com/>
- ▶ Pentaho Report Designer: This is downloadable from <http://community.pentaho.com/>
- ▶ MySQL Rel. 5.x: This is downloadable from <http://dev.mysql.com>


As soon as MySQL is installed and fully ready, it is time to create the database and the sample user and to restore the dump provided to run the samples. To do this, perform the following steps:

- ▶ Go to the <chapters\_samples>/db directory and unzip the foodmart\_mondrian.zip file
- ▶ Open a command-line window and go to the <chapters\_samples>/db directory where we previously unpacked the db archive
- ▶ Connect to MySQL by using the command-line MySQL client by typing the following command:  

```
mysql -uroot -p<root_password>
```
- ▶ As soon as you are connected to the command-line client, type the following commands:  

```
mysql>create database foodmart_mondrian default character set utf8;  
  
mysql>grant all on foodmart_mondrian.* to 'cookbook_usr'@'localhost' identified by 'password';  
  
mysql>grant all on foodmart_mondrian.* to 'cookbook_usr'@'%' identified by 'password';  
  
mysql>flush privileges;  
  
mysql>exit;
```
- ▶ As soon as the MySQL command-line client closes, restore the dump by typing the following command:  

```
mysqldump -uroot -p<root_password> foodmart_mondrian < ./foodmart_mondrian.sql
```

 Remember to substitute <root\_password> with our root user's password.

## Who this book is for

This book is the first comprehensive, single source of information about all the main Pentaho platform's components and gives you the opportunity to gain knowledge about the various aspects of the platform that contribute in building a complete BI solution. All the chapters give a clear understanding about each topic involved in implementing a BI solution so that by going through all of them, we are able to learn how to implement a complete BI solution from the beginning.



The book is an invaluable source of information for novice users who are coming to Pentaho for the first time and want to be productive as easily and as quickly as possible, and it is also a good source of information for advanced users who want to become productive with the new version as easily as possible.

## 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, database table names, folder names, filenames, file extensions, pathnames, dummy URLs, user input are shown as follows: "In the `<baserver_home>/pentaho-solution/system/common-ui/resources/themes/cookbook` directory, look for the `globalCookbook.css` file."

A block of code is set as follows:

```
<crystal>
  <file>mantleCrystal.css</file>
</crystal>
```


When we wish to draw your attention to a particular part of a code block, the relevant lines or items are set in bold:


```
<crystal>
  <file>mantleCrystal.css</file>
</crystal>
```

Any command-line input or output is written as follows:

```
mysql>create database foodmart_mondrian default character set utf8;
```

**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: "Select the target folder by clicking on it, and then from the **Folder Actions** menu, select **Upload**."

 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](mailto: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](http://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.

## Downloading the example code

You can download the example code files for all Packt books you have purchased from your account at <http://www.packtpub.com>. If you purchased this book elsewhere, you can visit <http://www.packtpub.com/support> and register to have the files e-mailed directly to you.

## 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](mailto: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](mailto:questions@packtpub.com) if you are having a problem with any aspect of the book, and we will do our best to address it.

# 1

## Getting Familiar with Pentaho User Console

In this chapter, we will cover the following topics:

- ▶ Adding content to favorites
- ▶ Accessing a solution's folders and files
- ▶ Creating a new solution folder
- ▶ Renaming a solution folder
- ▶ Moving a folder to Trash
- ▶ Showing basic folder properties
- ▶ Changing a folder's permissions
- ▶ Renaming a file in a folder
- ▶ Showing a basic file's properties
- ▶ Changing a file's permissions
- ▶ Moving a file to Trash
- ▶ Moving a file to a different folder
- ▶ Restoring content items from Trash
- ▶ Permanently deleting content from Trash
- ▶ Uploading content to a solution's folder

## Introduction

Let's start our tour around the new Pentaho Business Analytic Server by taking a deep dive into Pentaho User Console. The new BA Server version features a brand new design for Pentaho User Console that is far more clean and intuitive. Moreover, a lot of new features were introduced in this release, for example, copying, pasting, and moving content between folders, uploading and downloading content, renaming folders, hiding content, and moving content to `Trash`.

This chapter takes us through the user interface by explaining all its secrets to get a wonderful and intuitive experience in our everyday activities. All the recipes in this chapter apply to both the Community Edition and to the Enterprise Edition of Pentaho BA Server.

The recipes in this chapter are explained assuming that we are able to successfully log in to Pentaho User Console. To do this, we are free to use any user, not necessarily a demo user. However, to access and experience the complete set of functionalities offered by the user interface in a better manner, we use a user who is part of the administrator role. The administrator role is the role associated with all the Pentaho super users (we will cover these aspects in *Chapter 2, Configuring Your BA Server Instance*) and by that reason, it has full access to any functionality.

In case we want to use demo users, remember that we can use the following logins to access our system:

- ▶ `admin/password`: This is the new Pentaho's demo administrator after the famous user `joe` (the Pentaho-recognized administrator until Pentaho 4.8) has been dismissed in this new version.
- ▶ `suzy/password`: This is another simple user we can use to access the system. Because `suzy` is not a member of the administrator role, it is useful to see what changes here in case a user who is not an administrator tries to use the system.

## Adding content to favorites

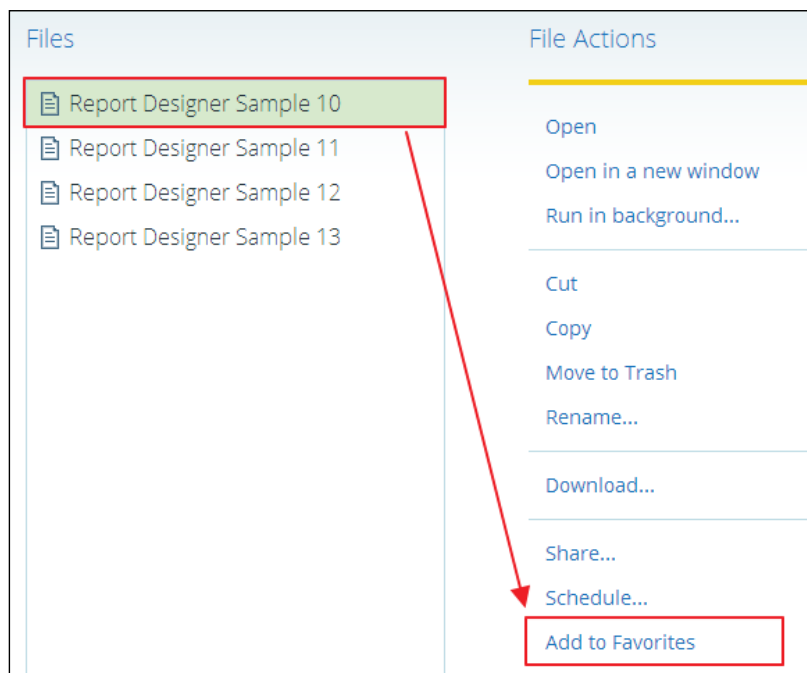
As soon as a user logs in to Pentaho User Console, he lands in the **Home** perspective. The concept of perspective in Pentaho is like a sort of view that logically organizes a set of common UI items that represent a particular context. The **Home** perspective is a sort of home page; here users can quickly access a set of favorite content items. By content item or simply content, we imply things such as reports, dashboards, **Online Analytical Processing (OLAP)** views, or any other kind of data representation in a specific content.

Favorites are a great way to group frequently used content items and make them immediately accessible in the **Home** perspective as soon as the user enters in Pentaho User Console. It is possible to mark any kind of content accessible in the BA server as favorite. This recipe will show you how you can quickly mark a particular content item as a favorite and have it easily accessible in the **Home** perspective.

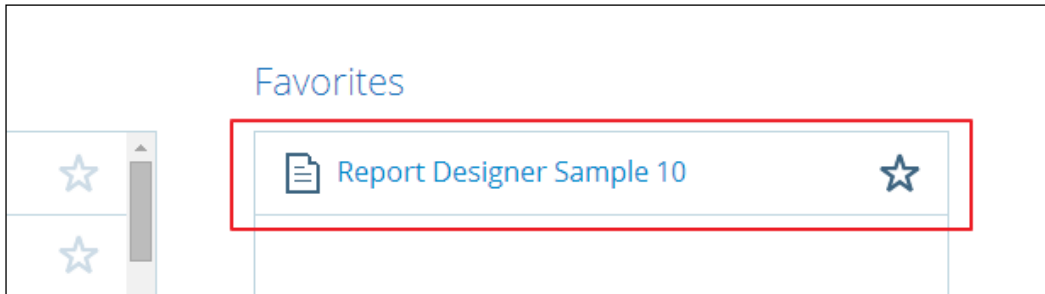
## How to do it...

The following steps describe how we can easily define a new favorite item and have it added in the favorites section in the **Home** perspective:

1. Click on the drop-down list located immediately below the upper-left corner of Pentaho User Console; the actual selected item is the name of the perspective we are currently in.
2. Select the **Browse Files** item entry.
3. From the tree in the solution explorer, select the folder that contains the content we are going to make a favorite.
4. To the right of the solution explorer in the file view, click the file from the **Files** list.
5. Select the **Add to Favorites** entry from the **File Action** menu to the right. The following screenshot will show you the flow of these actions in Pentaho User Console:



6. As we can see in the following screenshot, the content we selected is marked as a favorite and appears in the related list in the **Home** perspective.



### How it works...

It's a good thing to favorite items as it speeds up the access to frequently used content items. We can favorite almost everything accessible from Pentaho User Console and make it all accessible from the **Home** perspective just after the user logs in to Pentaho User Console. So, it is a great new feature.

To bookmark any content item, navigate to the solution's folders looking for the content item to mark it as a favorite and then choose **Add to Favorites** from the **File Actions** menu.

### There's more...

We saw how easily we can favorite something, but now let's take a look at how easily we can remove a content item from the **Favorites** list.

### Removing a content item from the favorites list

To remove a favorite content item from **Favorites** list, we can use three different ways:

- ▶ **Starting from the Home perspective:** Look at the **Favorites** list and click on the star icon located to the immediate right of the item we want to remove from the **Favorites** list.  
The item will immediately be removed from the **Favorites** list.
- ▶ **Starting the Browse Files perspective:** Select the folder from the tree in the solution explorer that contains the content to be removed from the **Favorites** list. Then look at the **Files** list to the right of the solution explorer.

Select the file that you want to mark as "unfavorite" from the **Files** list, then from the **File Action** menu to the right, click on **Remove from Favorites**. The item will immediately be removed from the **Favorites** list.

- ▶ **Clearing the Favorites list:** The third option gives us the possibility to completely clear the **Favorites** list from any item contained herein. To do this, navigate to **File | Favorites | Clear Favorites List** from Pentaho User Console menu, and we will get an empty list.

## Accessing a solution's folders and files

Content in Pentaho BA Server is contained in a repository called the Pentaho solution or generally, *the solution*. In the solution, content is organized through a set of folders that the user is free to organize by themselves based on their business needs. This recipe will show us how easy it is to access folders and files in the solution.

### How to do it...

The following steps describe how easy it is to access folders and files in the Pentaho solution:

1. Click on the drop-down list located immediately below the top-left corner whose actual selected item is the name of the perspective we are currently in.
2. Select the **Browse Files** item entry.
3. The **Browse Files** perspective is displayed by showing us folders (to the left) and files contained in every single directory (to the right).

### How it works...

Content and files in the Pentaho solution are accessible in Pentaho User Console through the **Browse File** perspective. To go to the **Browse Files** perspective, we can follow two paths depending on where we are.

To go to the **Browse Files** perspective from any perspective, perform the following steps:

1. Click on the drop-down list located immediately below the top-left corner whose actual selected item is the name of the perspective we are currently in.
2. Select the **Browse Files** item entry.
3. Pentaho User Console will immediately take us to the **Browse Files** perspective.

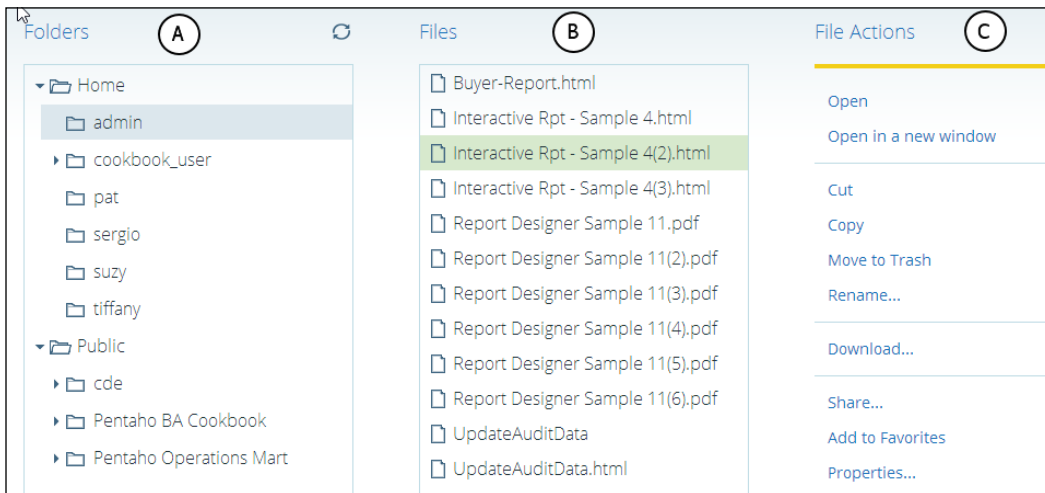


By going to the **Browse Files** perspective from the **Home** perspective, we can also choose to click on the big blue **Browse Files** button located on the left side of the screen.

As soon as we are in the **Browse Files** perspective, we can see the following things:

- ▶ On our left (marked as A in following screenshot), we have the tree that contains the list of the folders in the solution (**Folders**)
- ▶ By selecting a folder (marked as B in following screenshot), to the right, we have the list of files contained in selected folder

As we can see in the following screenshot, on the very right of the **Browse Files** perspective screen, we have the **Files Actions** menu (marked as C in following screenshot). This is a contextual menu and depending on the selection (either a file or a folder), some items in the menu can either display or not display different dialogs.



## There's more...

We just said the new BA server completely changed the look and feel of Pentaho User Console by organizing user interaction through perspectives. Let me give you a brief overview of the existing perspectives and their intended use.

## Introducing Pentaho User Console perspectives

In the new Pentaho User Console, we have five different perspectives by default:

- ▶ **Home:** This is the place where the user lands after a successful login to the system. It contains brief links to user documentation and samples, a view of recently used content items, and a list of favorites for quick access to frequently used content items.
- ▶ **Browse Files:** This gives the user access to folders and files in the Pentaho solution.
- ▶ **Opened:** This is the perspective that will contain any actively displayed (open) content items. As soon as we start a report, a dashboard, an analysis view, or anything else, it will always be displayed here.
- ▶ **Scheduled:** This perspective replaces the old **My Workplace** view we had in the previous releases of Pentaho BA Server. It contains a view of schedules made by the user, giving us the opportunity to completely manage schedules and access content produced once any schedule is terminated successfully.
- ▶ **Administration:** Previous releases of Pentaho BA Server were delivered with a separate console for Pentaho server administration tasks (the enterprise console for the EE version and the administration console for the CE version). Starting from this release of Pentaho BA Server, this console is no longer available and all the administrative tasks are available through this new perspective.

## Creating a new solution folder

As we saw in previous recipes, the user can organize content in the Pentaho solution according to his or her business needs. Let us see how this works.

### Getting ready

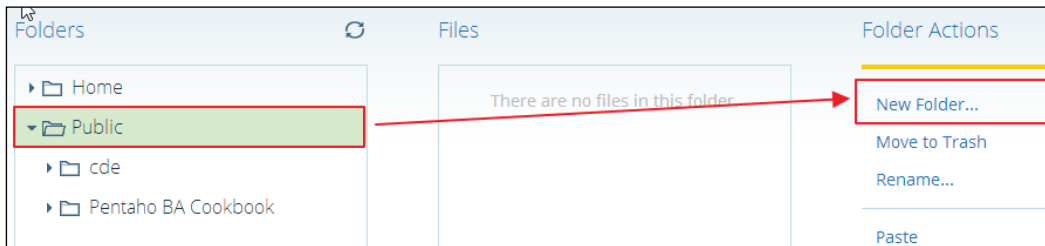
For this recipe, it is important that we use an administrator role's user to be freely able to create a folder in the public folders. If the user we are using to access Pentaho with is a normal user, they can create new folders only in their home directory folders, or they must be the owner of the parent folder under which they want to create new folders.

### How to do it...

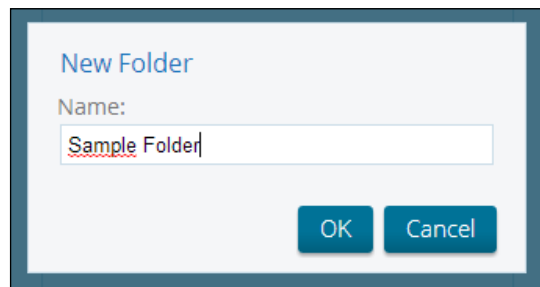
The following steps will describe how easy it is to create a new folder in the Pentaho solution:

1. Go to the solution explorer and browse the solution's folder structure for the folder under which we are going to create the new folder.

2. As we can see in the following screenshot, select the parent folder by clicking on it. Then, from the **Folder Actions** menu, select **New Folder...**



3. The **New Folder** dialog box appears. As shown in the following screenshot, type the name of the folder we are going to create. Finally, click on **OK** to confirm.



4. The new folder we just created is displayed in the solution explorer.

### How it works...

After we plan the structure of our Pentaho solution, we need to define it in Pentaho. To do this, we must create a correct folder structure to fill with content items and apply correct security rules.

Creating a new folder in the Pentaho solution is a fairly easy task. Just navigate the solution looking for the parent directory; once it is located, select the parent directory and choose **New Folder** from the **Folder Actions** menu. The **New Folder** dialog box opens. Fill the **Name** field with the name of the directory we are going to create, click on the **OK** button, and everything is done.

### There's more...

Designing a good repository structure is one of the more important things to think about in order to manage it the right way in the production and with minimal effort. To obtain this result, we must carefully think about its structure during the planning phase.

## Think about a good repository design

Designing a good structure for our solution repository is always a good rule of thumb to let the user easily navigate the repository and quickly find out what they are looking for. Another reason to go for a reasoned design of the repository structure is to easily assign security constraints to repository content items.

There could be various ways through which we can give a proper folder structure to the content repository; an example could be to start by separating the public root folder into a set of child folders, one for each department. Then, divide any department folder by the business topic and then by content. In any case, apart from the organization strategy we choose, there are two main points we need to take care of:

- ▶ The first thing to do to find out a good repository structure is to have a meeting with the system's key users and stakeholders and try to find out together which strategy would be the best one to design a reasonable repository structure. This would be a good opportunity to also discuss the security rules related to each folder in the repository.
- ▶ Think about a structure that can enable the repository growth following the company growth. Companies grow dynamically over time by enabling new business units or by changing existing ones. Build a repository so that it can follow the company growth with a minimal organizational impact on how that maps on Pentaho repository.

## Renaming a solution's folder

There are times when we would like to change the name of an existing folder to a new one.

### Getting ready

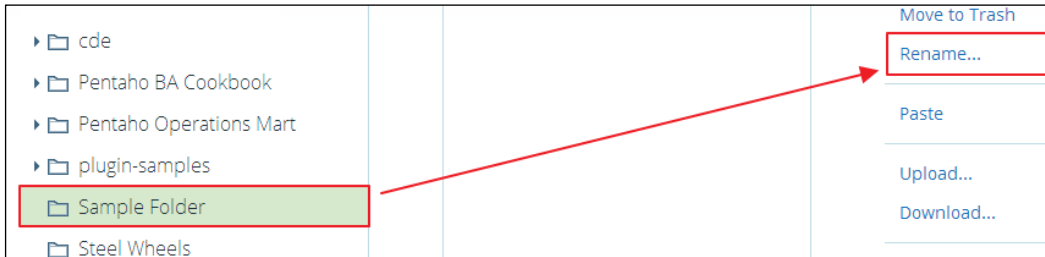
For this recipe, it is important that we use an administrator role's user to be able to freely rename a folder in the public folders. If the user we are using to access Pentaho is a normal user, they can rename folders only in their home directory folders, or they must be owner of the parent folder under which they want to rename the existing folders.

### How to do it...

The following steps will describe how easy it is to rename existing folders in the Pentaho solution:

1. Go to the solution explorer and browse the solution's folder structure for the folder that we want to rename.

2. As shown in the following screenshot, select the folder by clicking on it, then from the **Folder Actions** menu, select **Rename Folder....**



3. A warning dialog box appears by warning us that after renaming the dashboard, schedule, or favorites that refer to files contained in this folder, they may no longer work.
4. Clicking on the **Yes, Rename** button displays the **Rename** dialog box. If we click on **No**, the rename operation terminates and we will be taken back to the **Browse Files** perspective.
5. Type in the new name of the folder and click on **OK** to confirm.
6. The folder we just renamed is immediately displayed with its new name in the solution explorer.

### How it works...

The rename folder function is needed every time we need to change the name of a folder in the Pentaho solution. Renaming a folder in the solution is a process similar to the one we saw in the previous recipe. Navigate the solution and look for the directory we want to rename; once it is located, select **Rename Folder** from the **Folder Actions** menu. The **Rename** dialog box opens. Fill the **Name** field with the name of the directory we are going to create and click on the **OK** button. This time, a warning dialog box appears, saying that the rename operation could break some content that can make use of the item we are going to rename. To continue, click on the **Yes, Rename** button and the folder will be renamed.

### There's more...

Starting from this new release, the repository also provides a concept of the user's Home directory folder, which is useful to store the user's private content.

## A brief look into a user's home directory

Any time a user generates content of any type, there is always a need to make some of this content private. Generally speaking, content could be private either to a group of users or to the user who is working on it. In case the content is private to a single user, this could be either because that generated content is not fully ready to be shared or because the user's work is not related to anything that gives value to anything or anyone in the company.

Looking at the previous releases, this requirement was solved by playing with roles and assigning specific grants to the targeted content or to the folders that contain it. Starting from this release, BA Server 5.0, there is a concept of private user's folders or home directories. The home directories have the following properties:

- ▶ They are user specific and are visible only to the user who owns them
- ▶ They are created automatically by Pentaho any time an administrator creates a new user in the system
- ▶ Only users logged in with an administrator role can see all users' home directories and can act on any other user's home directory

So now, if we have a user's private content, why not start using this useful feature without wasting any time? Have fun with it!

## Moving a folder to Trash

There could be a need to remove some folders from our content repository, either because we are restructuring the repository or because we don't need them anymore. With Pentaho BA 5.0, deleted items are moved to a temporary area called `Trash`, and it will wait for us to decide what to do (as with the Recycle Bin in Windows or any other operating system). Let's see how we can move items that aren't needed to `Trash` with this recipe.

### Getting ready

For this recipe, it is important that we use an administrator role's user to be able move to trash folders and files that are contained in public folders. If the user we are using to access Pentaho is a normal user, they can move only folders and files in their home directory folders to trash.

### How to do it...

The following steps describe how we can easily move content items and folders to trash.

1. Go to the solution explorer and browse the solution's folder structure, looking for the folder that we want to move to trash.

2. Select the folder by clicking on it, then from the **Folder Actions** menu, select the **Move to Trash** menu entry.
3. The **Move to Trash** warning dialog box appears asking if we are completely sure that we want to delete the folder that we selected.
4. If we click on the **Yes, Move to Trash** button, the system deletes the selected folder and takes us to the **Browse Files** perspective. If we click on **No**, the **Move to Trash** operation gets terminated and we will be taken back to the **Browse Files** perspective without deleting any folder.
5. The solution is immediately updated and the deleted folder is no longer present in the list of folders.

### How it works...

The `Trash` folder is a special area of content repository (also known as the Pentaho solution) where folders and files that are not needed anymore are moved. The items are kept in that temporary area until we decide what to do. What we can do is either permanently purge the items or restore them. We are used to this concept because it is normal in modern operating systems (Windows, Linux, and so on) when deleting files and folders. These are not immediately removed because they are moved to `Trash` (or to Recycle Bin). The `Trash` folder is a new feature for Pentaho BA Server 5.0; in previous releases, we only had the opportunity to permanently delete any unwanted content. The idea that sits behind `Trash` is that in case we wrongly delete an item, we can easily get it back to the original location by undeleting it.

To move items to `Trash`, we just need to select them and then go to either the **Folders Action** or **Files Actions** menu, depending on item we are deleting, and then click on **Move to Trash** to delete the selected item. Isn't it that easy? Cool!

## Showing basic folder properties

Any object in our solution has basic properties we can be interested in looking at. This recipe starts by looking at how we can display a basic folder's properties by using the **Properties** dialog box.

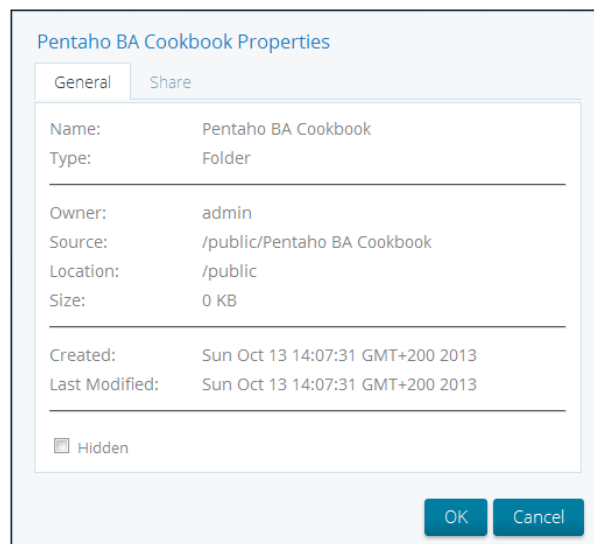
### Getting ready

For this recipe, it is important that we use an administrator role's user to be able to access the basic folder's properties for folders contained in public folders. If the user we are using to access Pentaho with is a normal user, he or she can access basic folders' properties only for folders in his or her home directory folders.

## How to do it...

The following steps describe how we can access a basic folder's properties.

1. Go to the solution explorer and browse the solution's folder structure looking for folder that we want to move to `Trash`.
2. Select the folder by clicking on it, then from the **Folder Actions** menu, select the **Properties...** menu entry.
3. As shown in the following screenshot, the **Display Properties** dialog box opens up with the **General** tab selected by default:



4. We can either click on the **OK** or **Cancel** button to close it.

## How it works...

The **Display Properties** dialog box is the dialog box that shows us all the basic attributes related to an object in the Pentaho solution. Let's take a brief look at these attributes:

- ▶ **Name:** This is the name of the object whose properties we're going to display. In our case, this is a folder and it is named Pentaho BA Cookbook.
- ▶ **Type:** This is the type of the object; the folder in our case.
- ▶ **Owner:** This is the name of the user who owns this object.
- ▶ **Source:** This gives us the complete path to the folder in the Pentaho solution where this specific object is located. In our case, because it is a folder in public folders, we have `/public/Pentaho BA Cookbook`.



- ▶ **Location:** This is the path to the object in the repository. It can start with either `public` or `home` depending on the base location. It is `home` if the object is stored in the home folders. It is `public` if the object is stored in the public folders.
- ▶ **Size:** This is the size of the object. In this case, because it is a folder, the size displayed is 0 bytes.
- ▶ **Created:** This shows us the date and time related to the creation of this object in the repository.
- ▶ **Last Modified:** This shows us the date and time related to the last update for this object.
- ▶ **Hidden:** This is a new feature of Pentaho BA Server 5.0. This flag is displayed only to users who are members of the administrator role and is left unchecked if this content must be made visible to any user.



Remember that whenever we talk about the path, and public, and home folders, we are always talking about the path related to the solution repository, unless specified otherwise.

## See also

We might find it useful to continue our exploration of the folder's metadata by looking at the *Changing a folder's permissions* recipe and looking at how to deal with managing permissions for folders in the Pentaho solution.

## Changing a folder's permissions

An important aspect of the enterprise systems is setting appropriate security permissions on any content item in our Pentaho solution so that we can properly make a content item public or private to a user or a set of users. The home directories, on their own, automatically define the content stored in it as private to the owner of a specific home directory. This recipe gives us an idea about how we can manage permissions for content items stored in the public section of the repository about how to manage the security of folders.

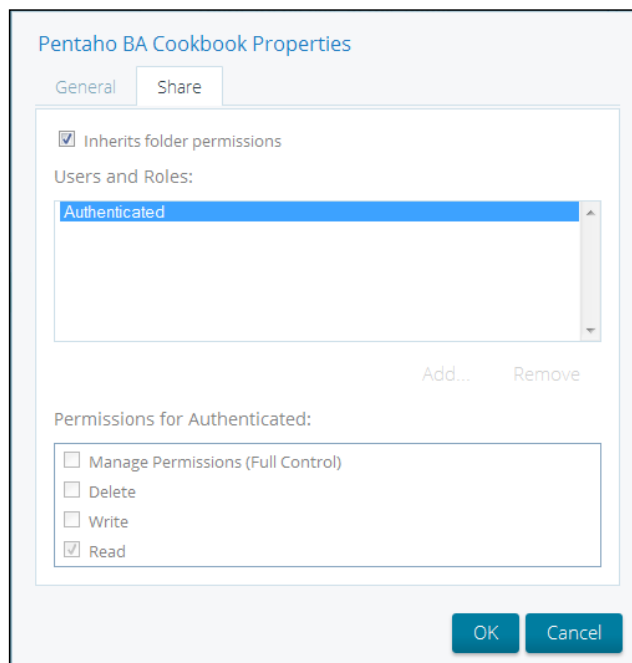
## Getting ready

For this recipe, it is important that we use an administrator role's user to be able to change permissions for folders contained in public folders. If the user we are using to access Pentaho is a normal user, they can only change permissions for folders in their home directory folders.

## How to do it...

The following steps describe how we can easily assign or change permissions for folders in the Pentaho solution:

1. Go to the solution explorer and browse the solution's folder structure for the folder in which we want to display the properties.
2. Select the folder by clicking on it.
3. From the **Folder Actions** menu, select the **Properties...** entry.
4. As shown in the following screenshot, the **Display Properties** dialog box opens with the **General** tab selected by default. Select the **Share** tab as shown in the following screenshot:



5. Change the permissions according to our needs or assign new permissions to a new set of users and/or roles.
6. Click on the **OK** button to confirm changes or click on **Cancel** to close the dialog box without modifying the existing permissions.

## How it works...

Pentaho Security is based on users and roles defined through the **Administration** perspective (we will talk about the **Administration** perspective and what we can do to administer Pentaho in the next chapter). Once we find the folder whose permissions we want to change, select it and click on **Properties** from the **Folders Actions** menu.

To set permissions, click on the **Share** tab in the **Properties** dialog box. Here, the new version of Pentaho BA Server has a new flag called **Inherits folder permissions**. As soon as we select the **Share** tab for the first time, that flag is checked. This means that the folder inherits the permissions from the parent folder. This would be fine for the vast majority of our applications but if we need to override it, we must uncheck this flag. As soon as the flag is unchecked, we can manually change either the set of users and roles that share this object or the permissions for a specific user or the role of that specific folder.

As soon as the **Inherits folder permissions** flag is unchecked, the first thing we can do is manually change the set of users and/or roles whose specific folder is shared. To add either a new user or a new role, perform the following steps:

1. With the **Share** tab of the folder properties dialog box selected and the **Inherits folder permissions** unchecked, click on the **Add...** button.
2. The **Select Users or Roles** dialog box pops up.
3. Click on the user or role we want to add to the **Users and Roles** list in the **Share** tab. We can't select multiple users and roles at once.
4. Click on the **OK** button to close the **Select Users or Roles** dialog box and add the selected user or role to the **Users and Roles** list.
5. Click on **Cancel** if we want to close the **Select Users or Roles** dialog box without updating the current **Users and Roles** list.

Another thing we can do after **Inherits folder permissions** is uncheck and manually change authorizations related to a specific user or role who shares the specific folder. We can associate four possible permission levels to a specific object, summarized as follows:

- ▶ **Manage Permissions (Full Control):** This gives us the ability to widely manage the folder in terms of elementary operations (read, write, and delete).
- ▶ **Write:** The user/role can write into this specific folder (that is, publish specific content).
- ▶ **Deleted:** The user/role can delete (**Move to Trash**) the specified folder.
- ▶ **Read:** Selected by default, this means that the user/role can access the content in the folder.

## See also

A good introduction to the folder's metadata information is given in the *Showing basic folder properties* recipe. It could be our natural starting point to get into the complete overview of the folder properties.

## Renaming a file in a folder

We just saw how we can rename a folder in the solution; let me show you how we can rename a file in a solution's folder.

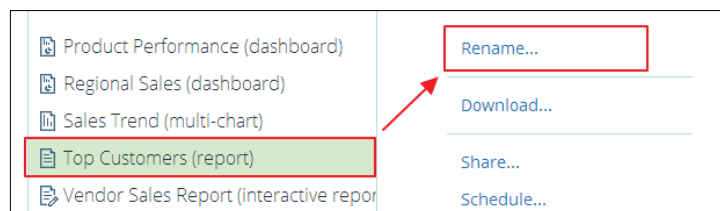
## Getting ready

For this recipe, it is important that we use an administrator role's user to be able to freely rename a file in a public folder's folder. If the user we are using to access Pentaho is a normal user, they can only rename files in their home directory folders, or they must be owner of the parent folder under which they want to rename the existing files.

## How to do it...

The following steps describe how we can easily assign or change permissions to folders in the Pentaho solution:

1. Go to the solution explorer and browse the solution's folder structure for the folder that contains the file we want to rename.
2. Select the folder by clicking on it.
3. As shown in the following screenshot that shows us the flow of actions, select the file to be renamed, and then from the **File Actions** menu, select **Rename File....**
4. A warning dialog box appears warning us that after renaming the content item, any other content that refers to the file that we are renaming may no longer work.



5. By clicking on the **Yes, Rename** button, the system displays the **Rename** dialog box. By clicking on **No**, the rename operation gets terminated and we will be taken back to the **Browse Files** perspective.
6. Type in the new file's name and click on **OK** to confirm.
7. The file we just renamed is immediately displayed with its new name in solution explorer.

### How it works...

Renaming a file in the Pentaho solution is a very easy thing. Just go through the solution's folders by looking for the content item file we want to rename. Select the content item from the **Files Action** menu, click on **Rename**, and set the new name for our content item file. Remember that renaming it can break the functionality of other content item files if the files link this file during their execution.

## Showing a basic file's properties

In this recipe, let's see how we can display the basic properties of the files contained in a solution's folder.

### Getting ready

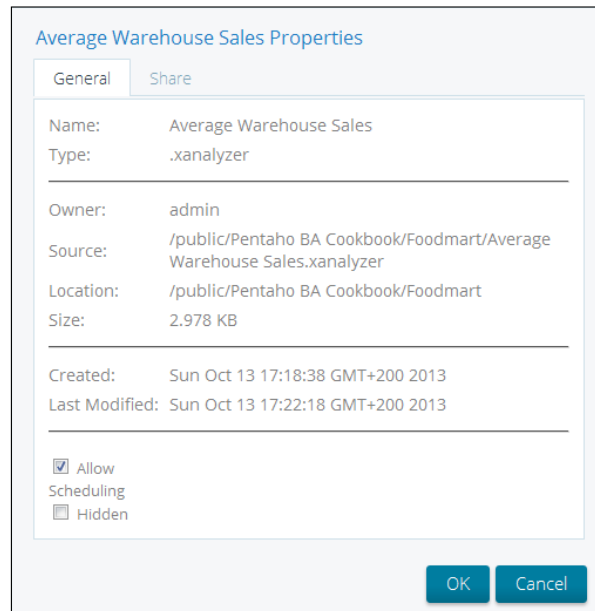
For this recipe, it is important that we use an administrator role's user to be able to access a basic file's properties for files contained in public folders. If the user we are using to access Pentaho is a normal user, they can access the basic files' properties only for files in their home directory folders.

### How to do it...

The following steps describe how we can easily show basic file properties in the Pentaho solution:

1. Go to the solution explorer and browse the solution's folder structure for the folder that contains the file in which we want to display the properties.
2. Select the folder by clicking on it and then on the file whose properties we want to display.
3. From the **File Actions** menu, select the **Properties...** entry.

- The **Display Properties** dialog box opens with the **General** tab selected by default, as shown in the following screenshot:



- We can either click on the **OK** button to confirm the changes (eventually) or click on the **Cancel** button to close the dialog box.

## How it works...

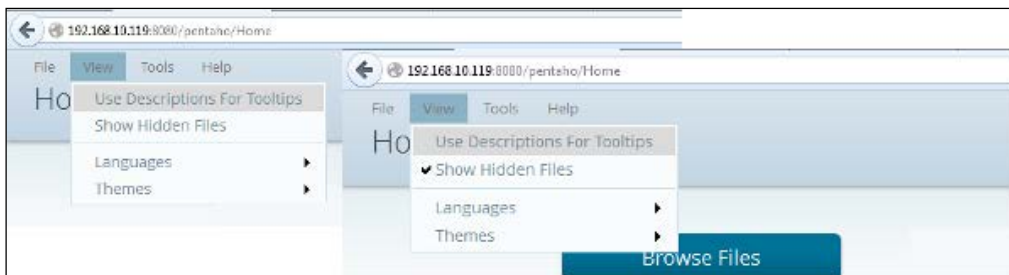
The dialog box to display the file's properties is almost the same as the dialog box to display the folder's properties, with minor differences. This dialog box displays a set of attributes related to content item files. Let's take a brief look at them:

- ▶ **Name:** This is the name of the object whose properties we are going to display. In our case, it is a folder and it is named `Average Warehouse Sales`.
- ▶ **Type:** In case of a file, the dialog displays the extension of the file. The case I showed in the previous screenshot is for an analyzer report, so the `.xanalyzer` extension is shown.
- ▶ **Owner:** This is the name of the user who owns this object.

- ▶ **Source:** This is the complete path to the file in the solution where this specific object is located. In our case, because it is a file in the public folders, we have `/public/Pentaho BA Cookbook/Foodmart/Average Warehouse Sales.xanalyzer`
- ▶ **Location:** This is the path to the object in the repository. It can start with either `public` or `home` depending on the base location. It is `home` if the object is stored in the home folders. It is `public` if the object is stored in the public folders.
- ▶ **Size:** This is the size of the object.
- ▶ **Created:** This is the date and time related to the creation of this object in the repository
- ▶ **Last Modified:** This is the date and time related to the last update of this object.
- ▶ **Allow Scheduling:** This is a new feature of Pentaho BA Server 5.0. This flag is displayed only to users who are members of the admin role and is checked if this content can be scheduled. If the user is a member of the administrator role and we don't want to let other users schedule this item, set this flag to unchecked.
- ▶ **Hidden:** This is a new feature of Pentaho BA Server 5.0. This flag is displayed only to users who are members of the admin role and is set to unchecked if this content must be made visible to any user.

## There's more...

As we showed you in one of the previous recipes, a user member of the administrator role can hide a content item by setting a flag as checked in the content item properties dialog. Now, if we are logged in with a user member of the administrator role, we have an entry in our **View** menu, called **Show Hidden Files**. By clicking on this menu entry, a tick will be displayed on the left-hand side of the menu entry to indicate that the feature is enabled. From that moment, all the hidden content items in the Pentaho solution will be visible and we can freely access them. However, remember that this function is only available to the administrator role users.



---

## Changing a file's permissions

Let's see how easy it is to change permissions for content items contained in a folder's repository according to our business needs.

### Getting ready

For this recipe, it is important that we use an administrator role's user to be able to change the permissions of the files in the folders contained in the public folders. If the user we are using to access Pentaho is a normal user, he or she can only change permissions to files in their home directory folders.

### How to do it...

The following steps describe how we can easily assign or change the permission to files in the Pentaho solution's folders:

1. Go to the solution explorer and browse the solution's folder structure for folder that contains the files whose permissions we want to change.
2. Select the folder by clicking on it, and then select the file whose permissions we want to change.
3. From the **File Actions** menu, select the **Properties...** entry.
4. The **Display Properties** dialog box opens with the **General** tab selected by default.
5. Select the **Share** tab.
6. Change the permissions according to our needs or assign new permissions to a new set of users and/or roles.
7. Click on the **OK** button to confirm the changes or on **Cancel** to close the dialog box without modifying the existing permissions.

### How it works...

Basically, assigning or changing permissions to files is similar to what we already saw for folders in the *Changing a folder's permissions* recipe. Remember to look for the file we want to assign or change the permissions for, and then once they're found, click on the **Properties...** entry from the **File Actions** menu.

The dialog box is the same as the dialog box we saw to set permissions for the folders. If we select the **Inherit folder permissions** option, we can set the permissions for our file and then click on **OK** to confirm the new permissions for the selected users or roles.



## See also

We can find it useful to continue our exploration of the file's metadata by looking at the *Changing a folder's permissions* recipe and looking at how to deal with managing permissions for files in the solution.

## Moving a file to Trash

In another recipe, we showed you how we can move the existing folders to trash. This recipe shows you how to do the same with files contained in the Pentaho solution's folders.

## Getting ready

For this recipe, it is important that we use an administrator role's user to be able to move files contained in folders and files contained in public folders to trash. If the user we are using to access Pentaho is a normal user, he or she can move only files contained in his home directory folders to trash.

## How to do it...

The following steps describe how we can easily move the content items files to trash:

1. Go to the solution explorer and browse the solution's folder structure for the folder that contains the file that we want to move to `Trash`.
2. Select the folder by clicking on it, and then select the file to move to `Trash`.
3. From the **File Actions** menu, select **Move to Trash**.
4. The **Move to Trash** warning dialog box appears by asking us if we are completely sure that we want to delete the file we selected.
5. If we press the **Yes, Move to Trash** button, the system deletes the selected file and takes us to the **Browse Files** perspective. If we press **No**, the **Move to Trash** operation gets terminated and we will be taken back to the **Browse Files** perspective without deleting any file.
6. The solution is immediately updated and the deleted file is no more present in the selected folder's file list.

## How it works...

The steps to move a file to `Trash` are more or less the same as what we saw when we talked about moving a folder to `Trash`.

Navigate the solution through folders until we locate the folder that contains the files we are going to move to `Trash`. Select the content item file, and then click on **Move to Trash** from the **Files Actions** menu. A confirmation dialog box asks us to confirm the operation. By clicking on the **OK** button, the dialog box is closed and the content item is successfully moved to `Trash`.

## See also

The *Moving a folder to Trash* recipe gives us an idea about how to do the same thing with folders. Moreover, take a look at the *Restoring content items from Trash* and *Permanently deleting content from Trash* recipes to see either how to restore content that is inadvertently deleted or how can we permanently delete space in the `Trash` folder and make our system administrator happy!

## Moving a file to a different folder

This is an operation we need to do sometimes to change the location of some files because of an error or because we are going to restructure the repository. There is no cut functionality to support us in this operation. So, to do this, we need to concatenate more basic operations.

## Getting ready

Apart from what we specified as general advice to be followed in this chapter's recipes, in the introduction of this chapter, the following advice must be followed to apply concepts in the current recipe:

- ▶ If we want the user to move files located in public folders to targets located in public folders, they must either be a member of the administrator role or must be the owner of both the source folder and the file and target folder.
- ▶ If we want the user to move files located in public folders to a target located in the `Home` folder, they must either be a member of the administrator role or must be the owner of both the source folder and file.
- ▶ If we want the user to move files located in the home folders to a target located in the public folders, they must either be member of the administrator role or the owner of the target folder.

## How to do it...

The following steps describe how we can move files between different folders:

1. Go to the solution explorer and browse the solution's folder structure for a source folder that contains the file we want to move to a new location.

2. Select the source folder by clicking on it and then select the file we want to move to new location.
3. From the **File Actions** menu, select **Cut**.
4. From the solution explorer, browse the solution's folder structure for a target folder location for the current file.
5. Select the target folder by clicking on it.
6. From the **Folder Actions** menu, select **Paste**.
7. The source file is successfully moved to the new location.

### How it works...

The move action we just showed you is only for files. At the time of writing this book, there was no corresponding action to move folders; we would like to have it that way. To move a complete folder to a new location, we basically need to copy and paste the folder to the new location, which is the same thing we do with single files. Then, after the copy has been created successfully, we need to go back to the original folder location and delete the original folder reference. A little bit more effort for the same result!

## Restoring content items from Trash

Until now, we just talked about how to move specific Pentaho content to `Trash`. We also said that `Trash` is a temporary area where files are stored waiting to be permanently deleted. This recipe will show you how we can restore a content item to its original location at the time of the deletion.

### Getting ready

Apart from what we specified as general advice to be followed in this chapter's recipes in the introduction of this chapter, the following advice must be followed to apply concepts in the current recipe. If the user's target folder where we're going to restore the content item is located in public folders, the user must be a member of the administrator role or the owner of both the source folder and files (in `Trash`, actually) and the target folder.

### How to do it...

The following steps describe how we can restore content items from `Trash` to their original location:

1. Go to the solution explorer and browse the solution's folder structure for the `Trash` folder. Instead of the normal folder icon, the `Trash` folder is indicated with a trash icon.

2. Select the `Trash` folder by clicking on it.
3. Select either the file or the directory we want to restore from the **Trash Content** list.
4. From the **Actions for Trash** menu, select **Restore**.
5. The item we selected is restored to its previous location.

### How it works...

Even if the files and folders are in the `Trash` folder, they are always related to their sharing permissions. This means that even if content items are in the `Trash Content` folder, a user can access only files and folders in the `Trash` folder shared with that specific user or owned by that specific user. As usual, only users who are members of the administrator role can access any kind of files and folders located in the `Trash` folder.

To access the `Trash` folder, click on the folder with the trash icon symbol. Once we enter the `Trash` folder, we can see content items and folders moved to `Trash`. To restore one of them to their original location, select it and choose **Restore** from the the **File Actions** menu.

### See also

The *Moving a folder to Trash* and *Moving a file to Trash* recipes give us an idea about how to delete files and folders. Moreover, take a look at the *Permanently deleting content from Trash* recipe to see either how can we permanently delete space in the `Trash` folder and make our system administrator happy!

## Permanently deleting content from Trash

Suppose we are sure that the content in the `Trash` folder is no longer needed. Let me show you how we can permanently delete this unwanted content from the `Trash` folder, freeing up some space in the solution.

### Getting ready

Apart from what we specified as general advice to be followed in these chapter's recipes in the introduction of this chapter, the following advice must be followed to apply concepts in the current recipe. If the user's target folder or content items we're going to permanently delete were located in public folders, the user must be a member of the administrator role or the owner of the content items and target folders we are deleting.

## How to do it...

To permanently delete content from the `Trash` folder one item at a time, we must perform the following steps:

1. Go to the solution explorer and browse the solution's folder structure for the `Trash` folder. Instead of the normal folder icon, the trash folder is indicated with a trash icon.
2. Select the trash folder by clicking on it.
3. Select either the file or the folder we want to delete permanently from the **Trash Content** list.
4. From the **Actions for Trash** menu, select **Permanently Delete**.
5. A warning dialog box appears warning us that we are going to permanently delete the selected file or folder (depending on what we are going to delete) and that this action cannot be undone.
6. If we press the **Yes, Permanently Delete** button, the selected content item is immediately deleted from the `Trash` folder, and it will never be accessible again.
7. If we press the **No** button, we will be taken back to the `Trash` folder and our content will not be deleted.

To permanently delete all the content from the `Trash` folder in an easy way all at once, we must do the following:

1. Go to the solution explorer and browse the solution's folder structure for the `Trash` folder. Instead of the normal folder icon, the `Trash` folder is indicated with a trash icon.
2. From the **Actions for Trash** menu, select **Empty the Trash**.
3. A warning dialog box appears warning us that we are going to permanently delete all the items from the `Trash` and that this action cannot be undone.
4. If we click on the **Yes, Empty Trash** button, all the content items are immediately deleted from the `Trash` folder and it will never be accessible again.
5. If we click on the **No** button, we will be taken back to the `Trash` folder and our content will not be deleted.

## See also

The *Moving a folder to Trash* and *Moving a file to Trash* recipes give us an idea about how to delete files and folders. Moreover, take a look at the *Restoring content items from Trash* recipe to see how we can restore content that we mistakenly deleted to its original location.

## Uploading content to a solution folder

Starting from this release, Pentaho BA Server 5.0, the solution repository supports the Java JSR 170 specification. Specifically, BA Server implements an Apache Jackrabbit-compliant repository. That said, the first important thing that must be taken into consideration is that the content stored in it is no longer directly accessible from the outside but must always be uploaded to the repository using the Pentaho development tools' publishing feature. Another possibility is to use the upload feature directly from Pentaho User Console.

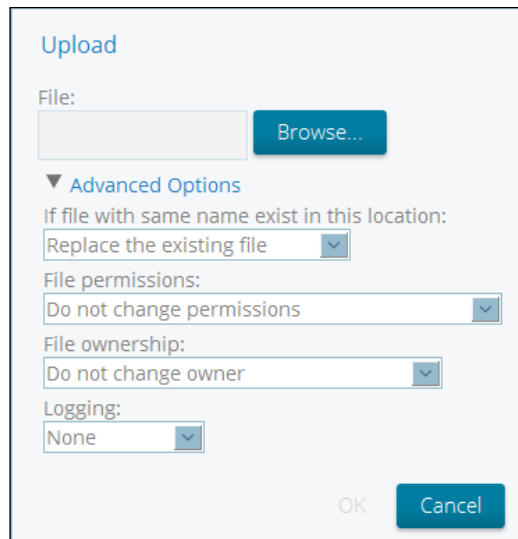
### Getting ready

For this recipe, it is important that we use an administrator role's user to be able to upload content in public folders. If the user we are using to access Pentaho with is a normal user, he or she can only upload content items in his home directory folders.

### How to do it...

The following steps describe how we can upload content to a solution's folder:

1. Go to the solution explorer and browse the solution's folder structure for the folder where we want to upload the new content.
2. Select the target folder by clicking on it and then from the **Folder Actions** menu, select **Upload**.
3. The upload dialog box opens as shown in the following screenshot:



The screenshot shows the 'Upload' dialog box. It has a title bar 'Upload' and a 'File:' label above an empty text input field. To the right of the input field is a 'Browse...' button. Below the input field is a section titled 'Advanced Options' with a downward arrow. Under 'Advanced Options', there are four labels with corresponding dropdown menus: 'If file with same name exist in this location:' with 'Replace the existing file' selected; 'File permissions:' with 'Do not change permissions' selected; 'File ownership:' with 'Do not change owner' selected; and 'Logging:' with 'None' selected. At the bottom right of the dialog are 'OK' and 'Cancel' buttons.

4. Click on the **Browse** button. The **Select file** dialog box opens up.
5. Browse local filesystem directories looking for content to upload.
6. Select content to upload by double-clicking on it. The complete path to the content is displayed in **File** field.
7. If required, open the **Advanced Options** accordion by clicking on the triangular icon to the left of the control label.
8. If required, change some default options according to our needs.
9. Click on the **OK** button to confirm the upload.
10. Click on the **Cancel** button to close the dialog box without uploading any content.
11. If the **OK** button is clicked on and the upload starts, the system displays an error dialog box with an error message that contains details about the condition that is making things go wrong.

### How it works...

Uploading content to the repository is a useful functionality that puts new content items into our Pentaho solution. As soon as we are going to upload content to a solution's folder, we can access an **Advanced Options** menu that gives us the opportunity to change some default flags by modifying the way the upload procedure behaves. Let's examine the various options shown in the preceding screenshot, in the following paragraphs:

- ▶ **If a file with same name exists in this location:** This option gives us the possibility of deciding what we should do if the content already exists in the specified location. The possible choices are:
  - **Replace the existing file:** If the file already exists, overwrite it.
  - **Do not upload:** Stop the upload and exit immediately.
- ▶ **File permissions:** This options has a connection with the securities assigned to the file. As soon as we download content from the repository, the downloaded content is packed into a zipped archive that contains a manifest file with all the metadata information that is related to that file.

```

<?xml version="1.0" encoding="UTF-8" standalone="true"?>
- <ns2:ExportManifest xmlns:ns2="http://www.pentaho.com/schema/">
  <ExportManifestInformation rootFolder="/public/Pentaho BA Cookbook/Foodmart/" exportBy="admin"
    exportDate="15-10-2013 04:14:47 CEST"/>
  - <ExportManifestEntity path="Sales by Period.xanalyzer">
    - <ExportManifestProperty>
      <EntityMetaData title="Sales by Period" path="Sales by Period.xanalyzer" locale="en_US"
        isHidden="false" isFolder="false" createDate="2013-10-14T13:12:50.211+02:00" name="Sales
        by Period.xanalyzer"/>
    </ExportManifestProperty>
    - <ExportManifestProperty>
      - <EntityAcl>
        - <aces>
          <permissions>READ</permissions>
          <recipient>Authenticated</recipient>
          <recipientType>ROLE</recipientType>
        </aces>
        - <aces>
          <permissions>READ</permissions>
          <permissions>WRITE</permissions>
          <permissions>DELETE</permissions>
          <recipient>sergio</recipient>
          <recipientType>USER</recipientType>
        </aces>
        <entriesInheriting>>false</entriesInheriting>
        <owner>admin</owner>
        <ownerType>USER</ownerType>
      </EntityAcl>
    </ExportManifestProperty>
  </ExportManifestEntity>
</ns2:ExportManifest>

```

As we can see, the file contains information about **Access Control List (ACL)** assigned to either users or roles and a reference to the user who is the owner of the file. That said, let's talk about the possible values for the **Advanced Options** menu choices:

- ▶ **Do not change permissions** (default): This assigns permissions inherited from the parent folder.
  - **Retain permissions on the uploaded file:** This maintains permissions from the uploaded file's metadata information
  - **Remove all permissions:** This removes any permissions assigned to the file and lets the user assign a new set
- ▶ **File ownership:** This lets us change the file ownership as follows:
  - **Do not change owner** (default): This assigns the owner inherited by the parent folder
  - **Set owner based on the uploaded file:** This maintains the owner set in the uploaded file's metadata information
- ▶ **Logging:** This changes the logging level during the upload process according to three levels: **None** (default), **Short**, or **Verbose**.



## There's more...

In this section, we just want to use a few words to give you an idea about the kind of content you can upload to the Pentaho solution and how you can upload content to the Pentaho solution using the command-line tools.

### What kind of content can I upload to my BA Server?

There is a lot of content around that could be interesting while implementing a **Business Intelligence** solution, but not all content types are managed by our Pentaho Server. Usually we can upload an entire folder or a bunch of files to the Pentaho solution repository. The following file types are supported and can be uploaded to the solution:

- ▶ **Reporting** (.prpt, .prpti, .xml): These files are produced with Pentaho Report Designer. *Chapter 7, Creating Reports Using Pentaho Report Designer*, will talk about how we can design reports using such a tool. Pentaho BA Server also starts the BIRT report and JasperReport report by using a set of appropriate plugins.
- ▶ **Analyzer** (.xanalyzer): These are produced by using Pentaho Analyzer, a tool we will cover in *Chapter 6, Creating Analysis Reports*.
- ▶ **Dashboards and CTools (CDA, CDF, CDE) related files**: These are produced by using Pentaho Dashboard Designer and CTools. We will cover these things in *Chapter 8, Creating Dashboards*.
- ▶ Other Pentaho solution files (.xaction, .locale)

Another set of files is supported to be uploaded, but they remain hidden once they are uploaded successfully:

- ▶ Web (.html, .htm)
- ▶ Reporting (.xml)
- ▶ Properties files (.properties)
- ▶ Graphics (.png, .jpg, .gif, .svg)

### Uploading content from the command line

Apart from the ability to upload files from Pentaho User Console, we can also upload files by using a command-line utility. Depending on the command-line arguments, this utility lets us import or export content to or from the Pentaho BA Server solution.

The utility is located in the <biserver-home> directory. As soon as we go to the <biserver-home> folder, we can start the utility by calling the following script on Linux/Mac operating systems:

```
./import-export.sh
```

On Windows platforms, we can use the following script:

```
import-export.bat
```

As an example, assume that we want to import a file called `SalesbyPeriod.xanalyzer.zip` in a folder called `Test` under the public folders. If the file to be imported is located in the `/home/cookbook_samples` directory, the command to start our import script and import that file in Pentaho under a folder called `Test` located in `/public` on a Linux/Mac environment is as follows:

```
./import-export.sh --import --url=http://localhost:8080/pentaho
--username=admin --password=password --source=file-system --type=files
--charset=UTF-8 --path=/public/Test --file-path=/home/cookbook_
samples/SalesbyPeriod.xanalyzer.zip --overwrite=true --permission=true
--retainOwnership=true
```

On a Windows environment, it is as follows:

```
import-export.bat --import --url=http://localhost:8080/pentaho
--username=admin --password=password --source=file-system --type=files
--charset=UTF-8 --path=/public/Test --file-path=c:/cookbook_samples/
SalesbyPeriod.xanalyzer.zip --overwrite=true --permission=true
--retainOwnership=true
```

While importing files to Pentaho BA Server from the command line, remember to always check access permissions at the filesystem level for files that we are going to import. Also check the execution permission for the script we are going to execute to import the file to. This aspect is relevant on any system.



# 2

## Configuring Your BA Server Instance

In this chapter, we will cover the following topics:

- ▶ Accessing the administration perspective
- ▶ Creating a new user
- ▶ Deleting an existing user
- ▶ Editing an existing user
- ▶ Creating a new role
- ▶ Deleting an existing role
- ▶ Editing an existing role
- ▶ Managing system roles
- ▶ Configuring authentication using the LDAP Server (EE version)
- ▶ Configuring authentication using the LDAP Server (CE version)
- ▶ Managing the mail server configuration
- ▶ Cleaning up aged generated files immediately
- ▶ Scheduling the cleanup of aged generated files

## Introduction

In this chapter, we continue where *Chapter 1, Getting Familiar with Pentaho User Console*, left off by providing recipes related to the BA Server administration and configuration. For anyone used to administrative tasks with the previous Pentaho releases, this is a part of the new BA Server where we can see a major change with previous versions of Pentaho BI Server. Any time we were involved in administrative tasks with older Pentaho releases, we needed to start a different, separate web application:

- ▶ In case we were dealing with BI Server EE (Enterprise Edition), we needed to start Enterprise Console
- ▶ In case we were dealing with BI Server CE (Community Edition), we needed to start Administration Console

Starting from Version 5, everything changed. All administrative tasks and operations have been organized in a new perspective called the **Administration** perspective. This perspective is accessible from the Pentaho User Console to every member of the administrator role.

Recipes in this chapter are given assuming that we are able to successfully log in to Pentaho User Console as a user who is part of the administrator role. The administrator role is the role associated with all the Pentaho super users, and for that reason, it has full access to any functionality that is related to the administration of Pentaho BA Server.

In case we want to use demo users, remember that we can use the administrator username and password for the demo users' username and password. This user is the new Pentaho's demo administrator, after the famous user Joe (the Pentaho-recognized administrator until Pentaho 4.8) has been dismissed starting from Version 5.

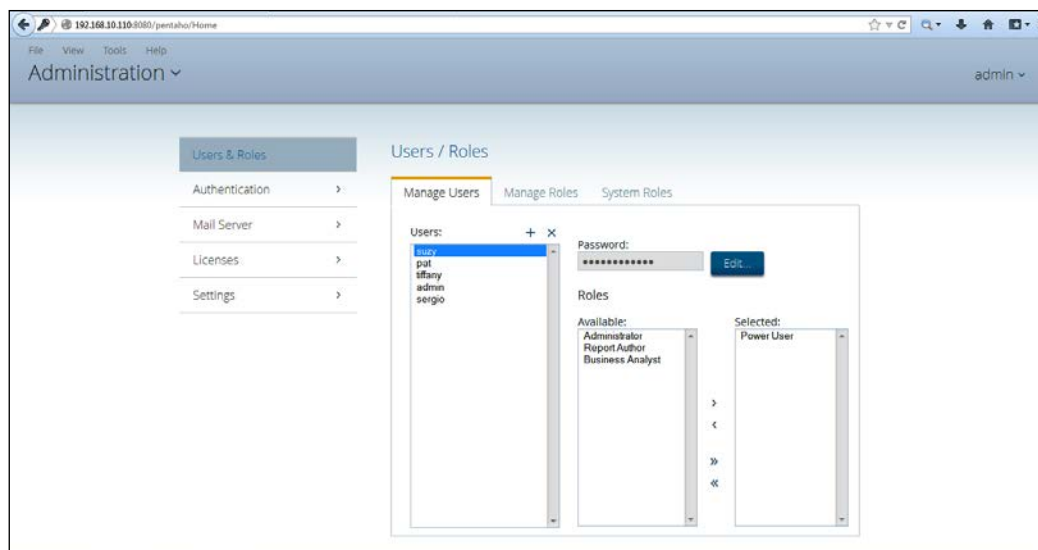
## Accessing the Administration perspective

To have everything at our fingertips, we are going to introduce you to how to get into the **Administration** perspective.

### How to do it...

To access the **Administration** perspective, you must perform the following steps:

1. Pull down the perspective combo box in the upper-left area of Pentaho User Console.
2. Select the entry named **Administration**.
3. You will immediately be taken to the **Administration** perspective.



## How it works...

The **Administration** perspective is the new place where all the administrative tasks take place. This perspective, accessible from the upper-left combo in Pentaho User Console, is organized through a left-side menu that contains the main administrative areas:

- ▶ For Pentaho BA Server 5 CE, we have functions to manage users and roles, a place to configure settings for the mail server (until Version 4.8, this was created through external configuration files), and general settings such as scheduling the deletion of unneeded content files.
- ▶ For Pentaho BA Server 5 EE, we have added menus to configure the authentication through an LDAP sever and another menu to configure the server components' licensing.

We will work through any of these functionalities in the following recipes of this chapter.

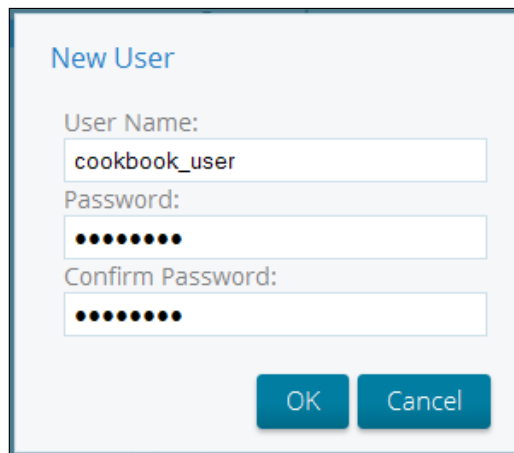
## Creating a new user

Now that we understand how to get into the new **Administration** perspective, we are going to see how we can easily define a new user in Pentaho BA Server. The user will be saved to the internal Pentaho user database.

### How to do it...

The following steps will show us how we can define a new user in Pentaho BA Server:

1. The **Users / Roles** entry of the **Administration** perspective's left menu is automatically selected by default.
2. On the right part of the screen, the **Users / Roles** tab is selected by default.
3. Click on the plus (+) icon in the upper-right corner of the **Users** list. The **New User** dialog box opens, as shown in the following screenshot:



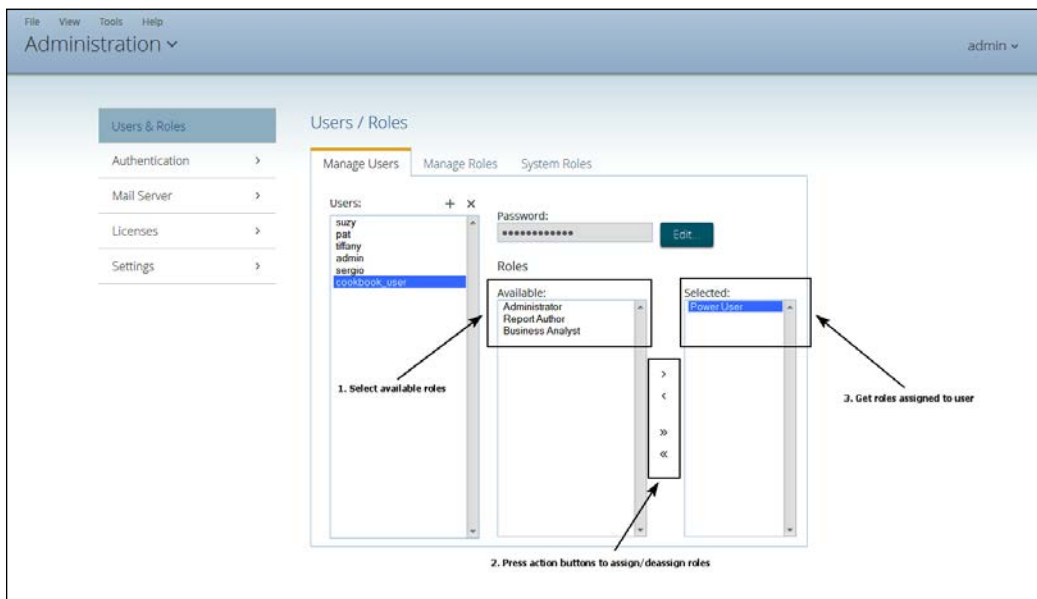
The screenshot shows a 'New User' dialog box with the following fields and values:

- User Name:** cookbook\_user
- Password:** [masked with 8 dots]
- Confirm Password:** [masked with 8 dots]

Buttons: OK, Cancel

4. In the **User Name** field, type the name of the user we are going to add. This field is mandatory.
5. In the **Password** field, type the password of the user we are going to add. This field is mandatory.
6. In the **Confirm Password** field, retype the password of the user we are going to add as a confirmation. This field is mandatory.
7. As soon as the **Confirm Password** field is filled and the content inserted herein is equal to the content in the **Password** field, the **OK** button appears.
8. If we click on the **Cancel** button, the dialog box closes and the new user data is not saved.

9. If we click on the **OK** button, the **New User** dialog box closes, the new user is saved, and it appears in the **Users** list.
10. Now that we have our new user, we must assign that user to one or more roles. Select the user inserted so far from the **Users** list.
11. Look at the **Roles** lists shown in the following screenshot. Select the role we want to assign to this user from the **Available** list and click on the **>** button. If we want, we can assign more than one role at a time by pressing *Ctrl* + right-clicking on any role you are going to assign to the user.
12. The role we just selected moves to right **Roles** list.
13. In case of an error, we can remove the role from right **Roles** list by selecting one or more wrong roles and clicking on the **<** button.
14. If we want to assign all the roles available to this user in one shot, we can click on the **>>** button without selecting an individual role from the **Available** list. The roles set will move from the **Available** list to the **Selected** list.
15. If we want to remove all roles from the current user in one shot, click on the **<<** button. The roles set will move from the **Selected** list back to the **Available** list, as shown in the following screenshot:





## How it works...

Defining a new user is almost the same procedure as the one used in the older version. After we access the **Administration** perspective, the **Users / Roles** menu entry is already selected and we are ready to start inserting a new user. By clicking on the plus (+) icon in the upper-right corner of the **Users** list, the **New User** dialog box opens and we are ready to fill the required fields. After any field is properly filled, click on the **OK** button and close the dialog box. The new user is immediately visible in the **Users** list. Now we must select the user and assign them a set of roles. To do this, select the roles we want to assign to our new user by clicking on them in the **Roles** list box and then clicking on the > button to move them to the right-hand side list box. We can select more than one role at a time by pressing *Ctrl* + right-clicking on any of the selected roles.

## See also

Following this recipe, the user will be created in the Pentaho user database. We might be interested in knowing how to delete a user in the *Deleting an existing user* section or how to update a user in the *Editing an existing user* section. If we are interested in how to create a new role for this user and how to associate it with a set of users, look at the *Creating a new role* recipe later in this chapter.

## Deleting an existing user

After seeing how to create a new user in Pentaho BA Server, this recipe is going to show us how we can delete an existing user.

## How to do it...

The following steps will show us how we can delete a user from the set of available users in Pentaho BA Server:

1. The **Users / Roles** entry of the left menu is automatically selected by default.
2. On the right part of the screen, the **Users / Roles** tab is selected by default.
3. Select the user we want to delete from the **Users** list and click on the **x** icon in the upper-right corner of the **Users** list.
4. A warning dialog box pops up asking us whether we are really sure about deleting the selected user.
5. If we click on the **Yes** button, the user will be deleted and we will be immediately taken back to the **Users** list. The deleted user will never appear anymore in the list of users.
6. If we click on the **No** button, the selected user will not be removed from the **Users** list, and we will immediately be taken back to the **Users** list.

## How it works...

Deleting an existing user is easy as as creating a new one. After we access the administration perspective, the **Users / Roles** menu entry is already selected and we are ready to delete the user. To do this, select it from the **Users** list, and then delete it by clicking on the **x** icon in the upper-right corner of the **Users** list. The selected user is immediately deleted and is no longer available in the system.

## See also

Following this recipe, our user will be deleted from the Pentaho user database. You might be interested in learning how to either create a new user in the *Creating a new user* recipe or about how to update a user in the *Editing an existing user* recipe.

## Editing an existing user

Finally, here are some last words on how to update an existing user. Updating a user in Pentaho means updating the user's password or adding or removing roles from or to the user's roles set.

## How to do it...

Let's start with changing a user's password. To do this, you must perform the following steps:

1. The **Users / Roles** entry of the **Administration** perspective's left menu is automatically selected by default.
2. On the right part of the screen, the **Users / Roles** tab is selected by default.
3. Select the user whose password we want to change from the **User's** list.
4. Click on the **Edit** button located on the left-hand side of the **Password** field. The **Change Password** dialog box opens.
5. In the **New Password** field, type a new password for this user. This field is mandatory.
6. In the **Confirm Password** field, retype the password for this user as a confirmation. This field is mandatory.
7. As soon as the **Confirm Password** field is filled and the content inserted herein is equal to the content in the **Password** field, the **OK** button appears.
8. If we click on the **Cancel** button, the dialog box closes and the updated password is not saved. The user continues to use their old password.
9. If we click on the **OK** button, the **Change Password** dialog box closes and the new password for this user is saved.

To add a new role to a user by updating the list of existing roles, you must perform the following steps:

1. The **Users / Roles** entry of the left menu is automatically selected by default.
2. On the right part of the screen, the **Users / Roles** tab is selected by default.
3. Select the user we want to change by updating the set of roles; the user is the member from the **Role** list.
4. From the **Available** roles list, select the roles we want to assign to the user and click on the **>** button. If we want, we can assign more than one role at a time by pressing *Ctrl* + right-clicking on any role we are going to assign to this user.
5. The users we just selected move from the **Available** to the **Selected** users list.
6. In case of an error, we can remove the role from the **Selected** list and reassign it to the **Available** list by selecting the wrong role and clicking on the **<** button. If we want, we can remove more than one role at a time by pressing *Ctrl* + right-clicking on any of the roles we want to remove.
7. If we want to assign all available roles to this role in one shot, we can click on the **>>** button without selecting any individual role from the **Available** list. The roles set will move from the **Available** list to the **Selected** list.
8. In case of an error, if you were to remove all users from the current role in one shot, press the **<<** button. The users set will move from the **Selected** list back to the **Available** list.

To remove an existing role from a user's role set, we must perform the following steps:

1. The **Users / Roles** entry of the left menu is automatically selected by default.
2. On the right part of the screen, the **Users / Roles** tab is selected by default.
3. Select the user we want to change to update the roles from the **User's** list.
4. From the **Selected** roles list, select the roles we want to remove from the user and press the **>** button. If we want, we can select more than one role at a time by pressing *Ctrl* + right-clicking on any role we are going to remove from this user.
5. The roles we just selected move from the **Selected** to the **Available** users list.
6. In case of an erroneous selection, we can remove the role from the **Available** list and reassign it to the **Selected** list by selecting the wrong role and pressing the **<** button. If we want, we can remove more than one role at a time by pressing *Ctrl* + right-clicking on any of the roles we want to remove.
7. If we want to remove all the roles from the current user in one shot, press the **<<** button. The roles set will move from the **Selected** list back to the **Available** list.

## How it works...

Managing existing users is a matter of changing the user's password and/or changing the set of roles assigned to a user.

To change the user's password from the **Users** list, we select the user whose password we want to change, and then we click on the **Edit** button located to the left of the **Password** field. The **Change Password** dialog box opens and gives us the ability to insert the new password by filling the **New Password** and **Change Password** fields. After we have filled the two fields, we can click on the **OK** button to confirm the new password we just set.

By updating the list of **Roles** a user is a member of is very simple; procedure is almost similar in cases of adding and removing user's membership from a role. After selecting the user we want to act upon, the list to the left contains the set of available roles, and the list to the right contains the set of roles assigned to this user. If we want to add a user's membership for a role, select the role we want to make the user a member of, and press the **>** button. If we want to remove a user's membership from a role, select the role we want to remove the user membership from, and click on the **<** button.

## See also

We might be interested in knowing how to create a new user in the *Creating a new user* recipe or how to delete an existing user in the *Deleting an existing user* recipe. If we are interested in how to create a new role for this user and how to associate that role to a set of users, look at the *Creating a new role* recipe later in this chapter.

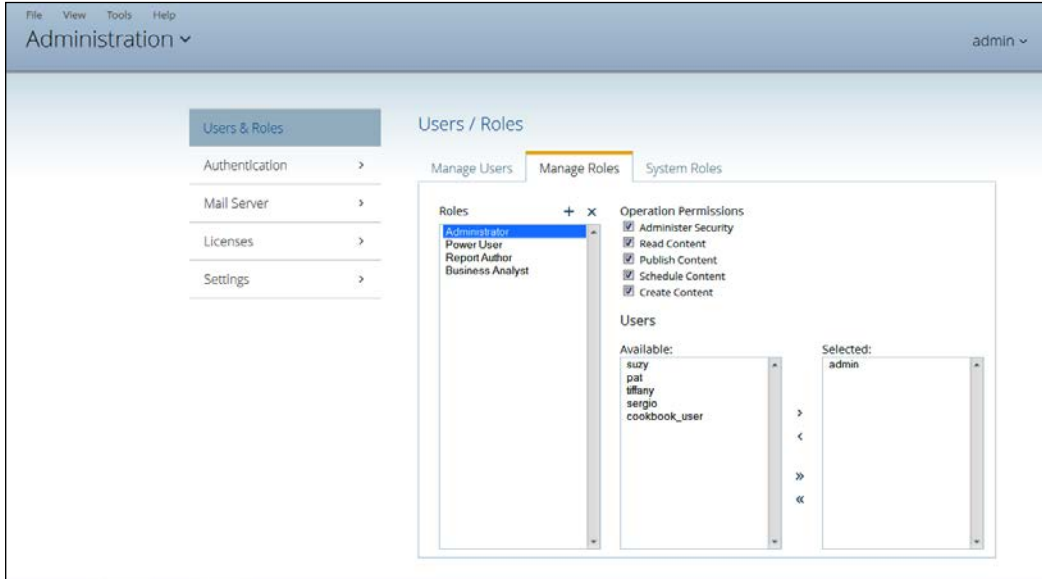
## Creating a new role

We opened this chapter with a set of recipes on how to manage users in Pentaho BA Server. Now, in this recipe, we are going to see how to create a new role and how to associate it with a predefined set of operation permissions. Operation permissions are the set of things a user can or cannot do if they would be a member of this role. That said, the role is a way to associate a set of users with the same collection of operation permissions.

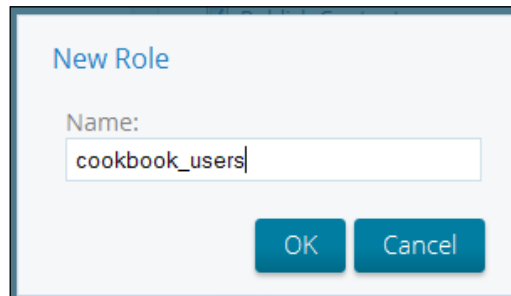
## How to do it...

The following steps will show us how to define a new role and associate it with a set of **Operation Permissions**:

1. From the **Users / Roles** entry of the left menu, select the **Manage Roles** tab.

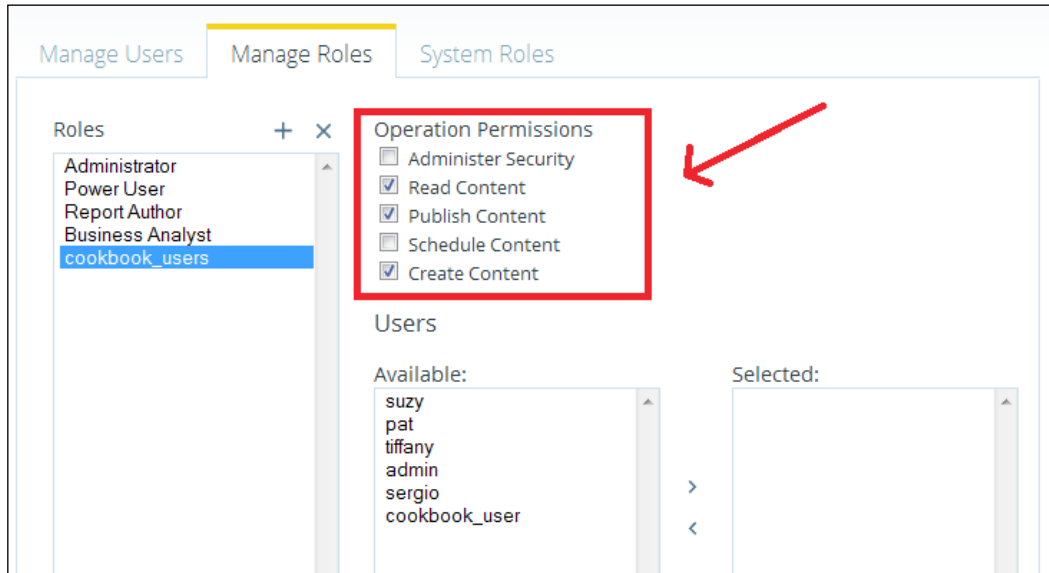


2. Click on the plus (+) icon in the upper-right corner of the **Roles** list and the **New Role** dialog box will open, as shown in the following screenshot:



3. In the **Name** field, type the name of role we are going to create. This field is mandatory. As soon as we start filling the field with some characters, the **OK** button appears.
4. If we click on the **Cancel** button, the dialog box closes and a new role is not created.

5. If we click on the **OK** button, the **Change Password** dialog box closes and a new role is created.
6. The **Roles** list is updated with the new role we just created in a selected state.
7. The new role does not have any permission assigned by default. As shown in the following screenshot, select the permissions we want to assign to a new role by clicking on items in the **Operation Permissions** checkboxes list:



8. Now that we have created our new role, we can make some users members of this new role. Select the role inserted so far from the **Roles** list.
9. Look at the **Users** lists. Select the users we want to assign to the role and press the **>** button. If we want, we can assign more than one user at a time by pressing **Ctrl** + right-clicking on any user we are going to assign to this role.
10. The roles we just selected move from the **Available** to the **Selected** roles list.
11. In case of an error, we can remove the user from the **Selected** list by selecting the wrong users and pressing the **<** button. If we want, we can remove more than one user at a time by pressing **Ctrl** + right-clicking on any of the roles we want to remove.
12. If we want to assign all the users available to this role in one shot, we can press the **>>** button without selecting any individual users from the **Available** list. The users set will move from the **Available** list to the **Selected** list.
13. If we want to remove all users from the current role in one shot, press the **<<** button. The users set will move from the **Selected** list back to the **Available** list.

## How it works...

**Operation Permissions** relates to giving a class of users (all the users who are members of a specific role) the ability to manage certain operations when authenticated to the Pentaho platform. As we will see in the *Managing system roles* recipe, we can also give **Operation Permissions** to unauthenticated users by associating permissions to **Anonymous System Role**.

**Operation Permissions** are basic permission items that we can assign to a specific role. The resulting users' permissions depend on a combination of the **Operation Permissions** items we have assigned to a particular role that we are going to define. We have several permission items and they are summarized as follows:

- ▶ **Read Content:** This is the basic permission item level we can assign to a specific role. By giving this permission, a user can only browse files in the solution, access the content they are is enabled to access, and start the execution of a report, dashboard, or other executable items. As an example, when we try to create a new folder, the system gives us a warning message that we do not have any rights to do this. Lastly, the user can't do the following things:
  - ❑ Copy, paste, and rename anything both in the **Home** and **Public** section of the solution
  - ❑ Move to trash any content item both in the **Home** and **Public** section of the solution
  - ❑ Assign or manage any privilege on the solution's content items
  - ❑ Upload or download any item from the solution
  - ❑ Create any interactive report, dashboard, analyzer report, and data source in the Pentaho User Console

The **Administration** perspective is not accessible. The **Schedule** perspective gives you only the ability to view your schedules. In the **Home** perspective, the toolbar displays only the **Browse Files** and the **Documentation** buttons to the user.

- ▶ **Create Content:** This item grants the permissions to create new content of any type. By adding this permission content, you are enabled to do the following:
  - ❑ Copy, paste, and rename anything in the **Home** section of the solution
  - ❑ Create new folders in the solution
  - ❑ Create new data sources
  - ❑ Create any interactive report, dashboard, analyzer report, and data source in Pentaho User Console

The user can't do the following things:

- ❑ Copy, paste, and rename anything in the **Public** section of the solution
- ❑ Move to trash any content item both in the **Home** and the **Public** section of the solution
- ❑ Assign or manage any privilege in the solution's content items
- ❑ Upload or download any content item

Other things that are valid when this operational permission is assigned are as follows:

- ❑ The **File Actions** menu displays a **Schedule** menu item, but as soon as we try to access it, we get a message that warns us that we don't have the right to do that kind of action.
  - ❑ The **Administration** perspective is not accessible. The **Schedule** perspective gives us only the ability to view our schedules. In the **Home** perspective, the toolbar displays all of the possible buttons to the user.
- ▶ **Schedule Content:** This item only grants the permissions to schedule content by enabling all of the related features in the **Schedule** perspective. Permissions to access the content items are not implicitly assigned by granting this permission item because they depend on the combination of the **Read** and **Create** operational permissions that we're going to define.
  - ▶ **Publish Content:** This item only grants the permissions to publish any content to the solution. Permissions to the content items are not implicitly assigned by granting this permission item because they depend on the combination of the **Read** and **Create** operational permissions we're going to define.
  - ▶ **Administer Security:** This item only grants permissions to assign privileges to the content items by enabling the **Share** tab in the content item's **Properties** dialog box. Permissions to access the content items are not implicitly assigned by granting this permission item because they depend on the combination of the **Read** and **Create** operational permissions we're going to define.

## See also

As soon as we install your brand new instance of Pentaho BA Server and we get to the **Administration** perspective to manage users and roles, we notice that few default roles that have already been defined exist. For details about these out-of-the-box roles, take a look at the article *Define Security for the BA Server in Pentaho Infocenter* at the following URL: [http://infocenter.pentaho.com/help/index.jsp?topic=%2Fconfig\\_ba\\_server%2Ftask\\_managing\\_users\\_and\\_roles.html](http://infocenter.pentaho.com/help/index.jsp?topic=%2Fconfig_ba_server%2Ftask_managing_users_and_roles.html).



## Deleting an existing role

This recipe will show us how we can easily delete a role from the existing set of roles.

### How to do it...

The following steps will give us an idea about how we can easily delete a role from an existing set of roles:

1. The **Users / Roles** entry of the left menu is automatically selected by default.
2. Select the **Manage Roles** tab and then select the role we want to delete from the **Roles** list.
3. Click on the **x** icon in the upper-right corner of the **Roles** list.
4. A warning dialog box opens, asking if we are really sure that we want to delete the selected role.
5. If we click on the **Yes** button, we will immediately be taken back to the **Roles** list, the selected role will be deleted, and it won't appear anymore in the **Roles** list.
6. If we click on the **No** button, we will immediately be taken back to the **Roles** list and the selected role will not be removed from the **Roles** list.

### How it works...

Deleting an existing role follows more or less the same procedure as the one we saw while deleting an existing user. After we access the **Administration** perspective, we see that the **Users / Roles** menu entry is already selected by default. Choose the **Manage Roles** tab, and then select the role we want to delete from the **Roles** list. Then, after the role has been selected, delete it by clicking on the **x** icon in the upper-right corner of the **Roles** list. The selected role is immediately deleted, and it is no longer available in the system. Any reference to users that are members of this role are automatically deleted.

### See also

Maybe we are interested in creating or updating an existing role. See the *Creating a new role* or *Editing an existing role* recipes to get details on this. In the *There's more...* section of *Creating a new role* recipe, we can find a detailed illustration of **Operation Permissions**, which is a good topic to get comfortable with in order to properly configure Pentaho roles.

## Editing an existing role

Finally, here are some words on how to update an existing role. Updating a role in Pentaho means either updating the set of users assigned to this role or changing the operation permissions for this role, or both. This recipe shows us how we can do this.

### How to do it...

To update a role by adding a new user or a new set of users to this role, we must perform the following steps:

1. Select the **Manage Roles** tab from the **Users / Roles** menu entry.
2. From the **Role** list, select the role we want to change by updating the set of users who are members of that role.
3. From the **Available** users list, select the users we want to assign to the role and click on the **>** button. If we want, we can assign more than one user at a time by pressing *Ctrl* + right-clicking on any user we are going to assign to this role.
4. The roles we just selected move from the **Available** to **Selected** roles list.
5. In case of an error, we can remove the user from the **Selected** list and reassign it to the **Selected** list by selecting the wrong user and pressing the **<** button. If we want, we can remove more than one user at a time by pressing *Ctrl* + right-clicking on any of the users we want to remove.
6. If we want to assign all the users available to this role in one shot, we can click on the **>>** button without selecting any individual users from the **Available** list. The user set will move from the **Available** list to the **Selected** list.
7. In case of an error, if we want to remove all the users from their current role in one shot, click on the **<<** button. The user set will move from the **Selected** list back to the **Available** list.

To update this role by removing an existing user or an existing set of users from this role, we must perform the following steps:

1. Select the **Manage Roles** tab from the **Users / Roles** menu entry.
2. From the **Roles** list, select the role we want to change by removing a user or a set of users who are members of that role.
3. From the **Selected** users' list, select the users we want to remove from the role and click on the **>** button. If we want, we can assign more than one user at a time by pressing *Ctrl* + right-clicking on any user we are going to remove from this role.
4. The roles we just selected move from the **Selected** to the **Available** roles list.

5. In case of an error, we can remove the user from the **Available** list and reassign it to the **Selected** list by selecting the wrong user and clicking on the **<** button. If we want, we can remove more than one user at a time by pressing *Ctrl* + right-clicking on any of the roles we want to remove.
6. If we want to remove all the users from the current role in one shot, click on the **<<** button. The users set will move from the **Selected** list back to the **Available** list.
7. In case of an error, if we want to reassign all the users available to this role in one shot, we can click on the **>>** button without selecting any individual user from the **Available** list. The users set will move from the **Available** list to the **Selected** list.

To update this role by changing **Operation Permissions**, we must perform the following steps:

1. Select the **Manage Roles** tab from the **Users / Roles** menu entry in the **Administration** perspective.
2. From the **Roles** list, select the role we want to change by updating **Operation permissions**.
3. Change the permissions currently assigned to the role by checking or unchecking the items in the **Operation Permissions** checkboxes list.
4. The new set of **Operation Permissions** for the selected role are immediately saved and applied.

### How it works...

Editing an existing role means changing the set of **Operation Permissions** assigned to a role or updating the set of users who are members of that role.

To change **Operation Permissions** for an existing role, just select the role and either check or uncheck the **Operation Permissions** checkbox until we reach the desired configuration for the selected role.

Updating the list of **Users** members of a selected role is a very simple procedure. After selecting the role we want to act upon, we see that the list to the left of **Role** contains the set of available **Users**, and the list to the right contains the set of **Users** assigned to this user. If we want to add the user membership for a role, we select the user we want to make a member of selected **Role** and click on the **>** button. If we want to remove the **Users** membership from selected **Role**, we select the user we want to remove from the **Users** membership and click on the **<** button.

### There's more...

Unfortunately, a rather useful thing that is missing from the list is the ability to rename a role whenever required. If we need to rename a role, the only thing we can do is delete the wrong role and recreate it. This is not so straightforward, but it is the only way to do this.

## See also

If we are interested in creating a new role or going through the details of the **Operation Permissions** settings, take a look at the *Creating a new role* recipe.

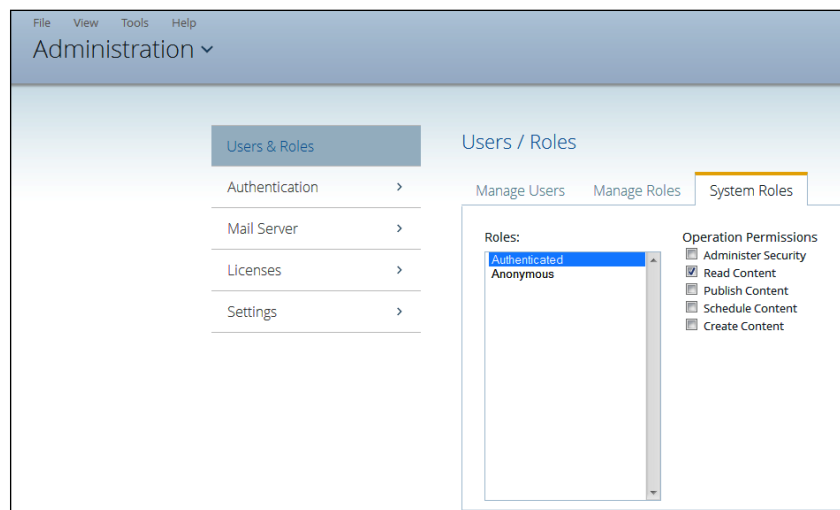
## Managing system roles

As soon as any user logs into Pentaho, the system grants a set of predefined roles called the **System** roles. Starting from Pentaho BA Server 5.0, the management of permissions for these roles is in the **Administration** perspective. This recipe shows us how we can manage the permissions for these two roles.

## How to do it...

The following steps will detail how we can change **Operation Permissions** to predefined **System** roles:

1. Select the **System Roles** tab from the left menu entry, **Users / Roles**, in the **Administration** perspective. The following screenshot shows us the set of **System** roles that are predefined in the system and their assigned operational permissions:



2. Select the role we want to change by updating **Operation permissions** from the **Roles** list.
3. Change the permissions currently assigned to the selected role by checking or unchecking items in the **Operation Permissions** checkboxes list.
4. The new set of permissions are immediately saved and applied to the current role.

## How it works...

System roles are particular roles that are predefined in the system and we can only manage the assignment of **Operation Permissions**. To change **Operation Permissions** assigned to a specific system role, select the role we want to make the changes to and check or uncheck the **Operation Permissions** flags until we get the desired configuration.

## There's more...

Let's go deep into these two roles and see which role they play in the Pentaho authentication process.

### Things to notice about the authenticated role

The authenticated role is the system role that is injected into the set of user roles during the authentication process. It is extremely important to know that it exists because it adds a set of default **Operation Permissions** for a user who is going to log in to Pentaho User Console. Therefore, any time we are considering the permissions associated to a specific user, don't forget this hidden contribution to the final result!

### How to bypass the Pentaho BA Server security

In Pentaho BA Server, it is impossible to remove completely the security mechanisms of the Pentaho platform. For this reason, the only thing we can do is bypass the user authentication mechanisms by introducing an **Anonymous** role. **Anonymous** is **System Roles** that is automatically assigned to any unauthenticated user that is going to log in to the BA server platform.

To bypass the BA server security and allow unauthenticated users to connect to Pentaho, we first need to stop the BA server and apply a set of configuration changes to wire Pentaho Spring Security beans in a different way. By wiring Spring beans correctly, we can assign users an **Anonymous** role with the correct permissions for our particular use case by choosing from the set of available **Operation Permissions**.

For a detailed description about how to remove the security and allow anonymous access, follow the guidelines detailed in the article *Remove Security by Allowing Anonymous Access* in *Pentaho Infocenter* at the following URL: [http://infocenter.pentaho.com/help/index.jsp?topic=%2Fsecurity\\_guide%2Ftask\\_removing\\_security.html](http://infocenter.pentaho.com/help/index.jsp?topic=%2Fsecurity_guide%2Ftask_removing_security.html).

## See also

It could be a good idea for you to refresh your knowledge about the **Operation Permissions** settings by going through the details in the *Create a new role* recipe.

## Configuring authentication through the LDAP server (EE version)

The *Configuring authentication through the LDAP server (EE version)* recipe covers one of the hottest topics when configuring Pentaho. Usually, as soon as we introduce a new application to our network, that application must be integrated enterprise-wide with other systems. The first thing we care about is the security and because of this, the first thing we are required to take great care of is the integration with our customer or our own company's enterprise authentication system. This, for the vast majority of cases, is an LDAP server or an MS Active Directory.

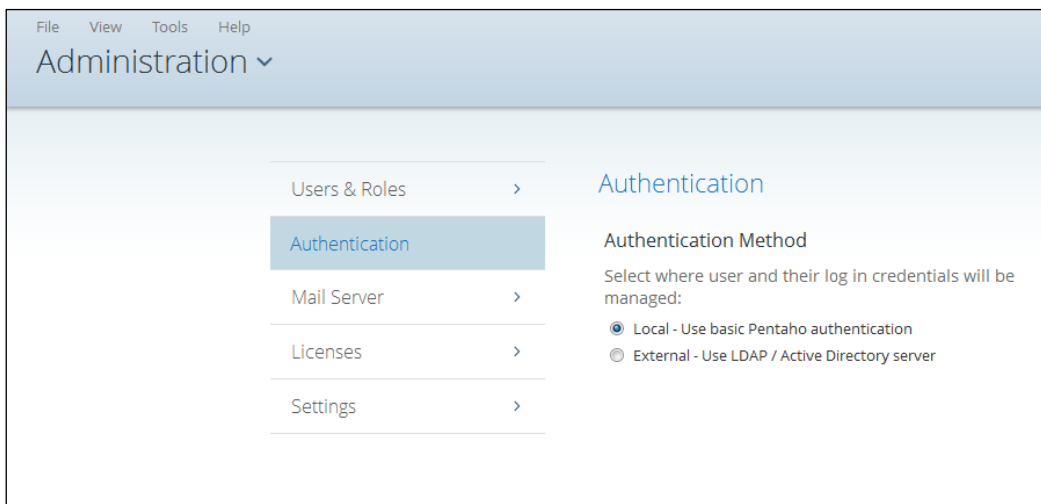
This recipe will show us how to configure the Pentaho BA server EE to authenticate users to an LDAP server (OpenLDAP for the current example) without pain. It is a critical configuration procedure, partially made through a wizard implemented in the **Administration** perspective by partially editing some files by hand.

### How to do it...

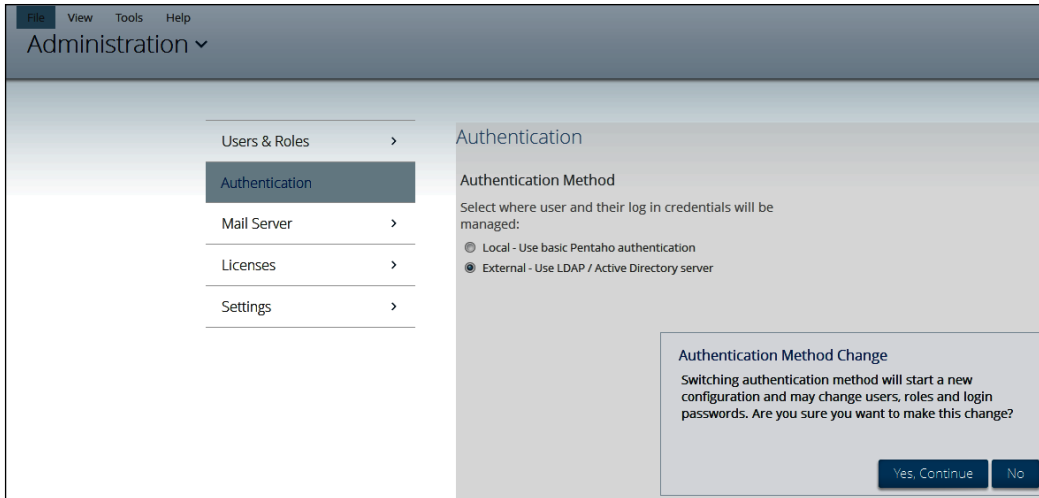
The following steps detail how we can easily configure the Pentaho BA server EE (Enterprise Edition) to connect to an LDAP server:

1. Select the **Authentication** entry from the **Administration** perspective's left-side menu.

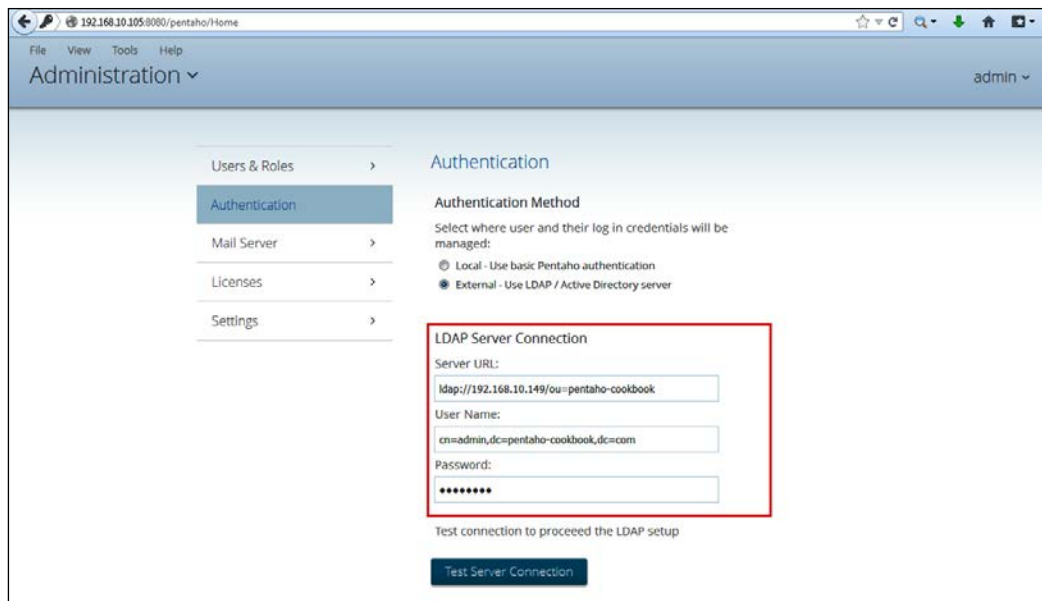
As we can see in the following screenshot, the **Authentication** configuration form opens by showing our actual authentication mechanism. If this is a brand new installation and we didn't change anything, the default authentication method's radio button is selected as **Local – use basic Pentaho authentication**.



2. Select the **External – Use LDAP / Active Directory server** authentication method.
3. As shown in the following screenshot, a message box warns us that by switching the authentication method, we are going to start a new configuration and may change users, roles, and the login password:



4. If we click on **No**, we are taken back to the main **Authentication** page without starting the switching process. Our Pentaho BA server authentication's source remains unchanged.
5. If we click on **Yes, Continue**, we are going to start the reconfiguration of the authentication system to use **External – Use LDAP / Active Directory server**.
6. The server tries to connect to the LDAP server that was defined at that time.
7. If this is our first time configuring the connection to the LDAP server, Pentaho will try to connect to a default LDAP location and, most probably, it will fail. If so, a message box will warn us about that. Do not worry about this and click on the **Close** button to continue.
8. As shown in the following screenshot, after we click on the **Close** button, the **LDAP Server Connection** form appears:



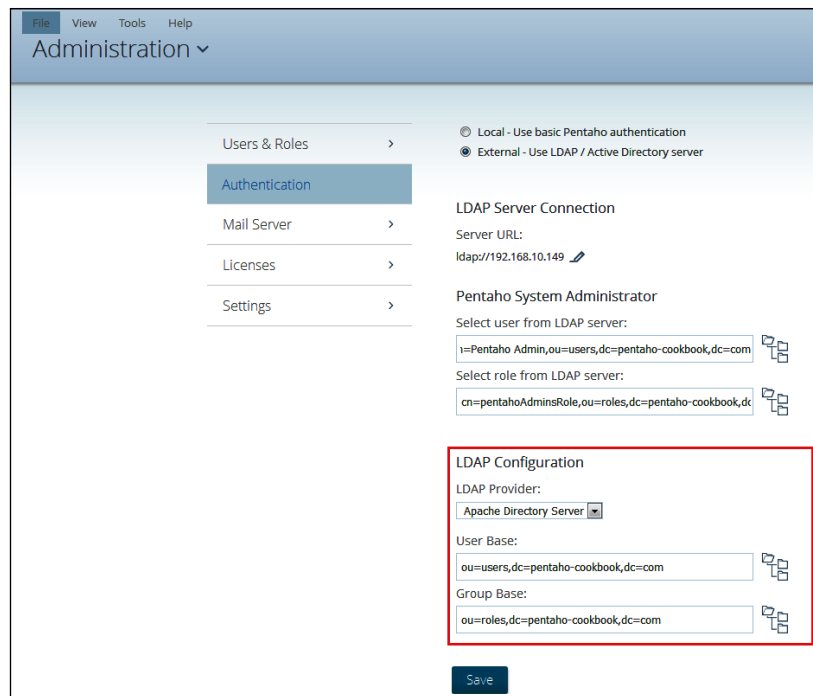
9. Type LDAP Server URL in the **Server URL** field. This field is mandatory.
10. Type the user's distinguished name in the **User Name** field. This field is mandatory.
11. Type the user's password in the **Password** field. This field is mandatory.
12. Click on the **Test Server Connection** button to verify that Pentaho connects successfully with your LDAP Server.
13. If the connection test does not work properly, a message box warns us about this and we will be taken back to LDAP server configuration parameters to apply the needed corrections.
14. If the LDAP server connection works properly as configured, the LDAP configuration form expands vertically by adding two new sections: **Pentaho System Administrator** and **LDAP Configuration**.



15. Take a look at the **Pentaho System Administrator** section as shown in the following screenshot:

The screenshot shows the Pentaho System Administrator configuration interface. The navigation menu on the left includes 'Users & Roles', 'Authentication' (selected), 'Mail Server', 'Licenses', and 'Settings'. The main content area is divided into sections: 'Local - Use basic Pentaho authentication' and 'External - Use LDAP / Active Directory server' (selected). Below this is the 'LDAP Server Connection' section with a 'Server URL' field containing 'ldap://192.168.10.149'. The 'Pentaho System Administrator' section is highlighted with a red box and contains two fields: 'Select user from LDAP server:' with the value 'cn=Pentaho Admin,ou=users,dc=pentaho-cookbook,dc=com' and 'Select role from LDAP server:' with the value 'cn=pentahoAdminsRole,ou=roles,dc=pentaho-cookbook,dc=com'. Below this is the 'LDAP Configuration' section with fields for 'LDAP Provider:' (Apache Directory Server), 'User Base:' (ou=users,dc=pentaho-cookbook,dc=com), and 'Group Base:' (ou=roles,dc=pentaho-cookbook,dc=com). A 'Save' button is at the bottom.

16. In the **Select user from LDAP server** field, type the complete name of the LDAP user, which we will consider as the Pentaho server administrator. In our case, the distinguished name is `cn=Pentaho Admin,ou=users,dc=pentaho-cookbook,dc=com`. This field is mandatory.
17. In the **Select roles from LDAP server** field type, complete the name for the LDAP group that we will consider as a group that contains users who we will consider as the Pentaho server administrator. This is a group in LDAP Server that will be mapped as the **Administrator** role. In our case, the complete group name is `cn=pentahoAdmin Roles,ou=users,dc=pentaho-cookbook,dc=com`. This field is mandatory.
18. Look at the **LDAP Configuration** section of the form, as shown in following the screenshot:



19. Select **LDAP Provider** from the related drop-down list. The default value, **Apache Directory Server**, is good enough in any standard case.
20. In the **User Base** field, type the complete path to consider as the base path for all of the user searches. In our case, the base path for users is `ou=users,dc=pentaho-cookbook,dc=com`. This field is mandatory.
21. In the **Group Base** field, type the complete path to consider as the base path for all of the group searches. In our case, the base path for groups is `ou=roles,dc=pentaho-cookbook,dc=com`. This field is mandatory.
22. Click on the **Save** button to confirm the settings.
23. A message box informs you that the changes have been saved successfully. Remember that new changes will not take effect until you restart your Pentaho server.  
Now that we have completed the LDAP configuration, we need to refine some things before having everything fully working.
24. Go to the `<biserver_home>/pentaho-solution/system` directory.
25. Edit the `Context-security-ldap.properties` file application. This property file is generated when we click on the **Save** button at the end of the **LDAP Server Connection** configuration.
26. The file is divided into five sections, as specified in the *There's more...* section of this recipe.

27. Go to the **Populator** section identified by keys with the `populator` prefix. Verify that the `populator.groupSearchFilter` key is defined as `populator.groupSearchFilter=(amp(objectClass=posixGroup)(memberUid={1}))`. If it is not, apply changes according to the sample provided.
28. Go to the **All Authorities Search** section identified by keys with the `allAuthoritiesSearch` prefix. Verify that the `allAuthoritiesSearch.searchFilter` key is defined as `allAuthoritiesSearch.searchFilter=(objectClass=posixGroup)`. If it is not, apply the changes according to the sample that is provided.
29. Go to the **All Username Search** section identified by keys with the `allUsernamesSearch` prefix. Verify that the `allUsernamesSearch.searchFilter` key is defined as `allUsernamesSearch.searchFilter=(objectClass=posixAccount)`. If it is not, apply the changes accordingly to the sample that is provided.
30. Save the file and close the editor.



The configuration steps listed here are valid considering the connection of the Pentaho BA server to an OpenLDAP server. Assume a similar configuration in case we want to connect to an MS Active Directory server with the only exception of names of MS Active Directory server attributes used.

The LDAP configuration is finished. Remember that we have to restart our BA server before the new configuration changes take effect.

### How it works...

The following is a definition that we can find on Wikipedia:

*"LDAP stands for Lightweight Directory Access Protocol and is an application protocol for accessing and maintaining distributed directory information services over an Internet Protocol (IP) network. Directory services may provide any organized set of records, often with a hierarchical structure, such as a corporate e-mail directory."*



See the Wikipedia link at <http://en.wikipedia.org/wiki/LDAP>.

In large organizations, an LDAP server is typically a good way to map all of the company's organizational structure in terms of domains (they can differentiate various branches or internal organization or other), roles, and users who relate to those domains. Because of this, it is a good rule of thumb to use this approach to have all of this information and the management of this information in one place in a centralized repository. To agree to this enterprise schema, any application must map its authentication and authorization mechanisms on such an enterprise security architecture to be fully compliant with the company policies.

To access a particular object in the LDAP tree, a client has to submit a so-called LDAP query to search for a specific object or a specific set of objects located in the tree. The query is composed of the following three main elements at least:

- ▶ A base object that sets the position in the tree relative to which we are going to perform the query
- ▶ A scope that sets the level of depth the query will act upon (`singleLevel` or `wholeSubtree`)
- ▶ A filter that helps us point rightly to the element or set of elements that we are looking for

After we complete the LDAP configuration wizard as explained in the recipe body, the wizard process generates a set of LDAP property configuration files located in the system configuration directory. They collect all the possible configuration properties for the entire LDAP configuration in one place.

The file we are referring to is named `applicationContext-security-ldap.properties` and is located in the `<biserver-home>/pentaho-solutions/system` directory.

This file is divided into five sections as follows:

- ▶ **Connection information:** This is identified by all keys with the `contextSource` prefix. It defines the connection to the LDAP server and the information about the user that can perform the searches over the LDAP tree (typically an LDAP user, not necessarily an administrator).
- ▶ **Users:** This is identified by all the keys with the `userSearch` prefix. It defines how the LDAP tree is searched for by users who are going to log in to Pentaho to authenticate them.
- ▶ **Populator:** This is identified by all the keys with the `populator` prefix. It defines how to match fully distinguished user names coming from `userSearch` to distinguished role names for the roles those users belong to.
- ▶ **All Authorities Search:** This is identified by all the keys with the `allAuthoritiesSearch` prefix. The entries that we get from this search populate the BA server **Access Control List (ACL)** roles.
- ▶ **All Username Search:** This is identified by all the keys with the `allUsernamesSearch` prefix. The entries that we get from this search populate the BA server **Access Control List (ACL)** users.

While going to configure this file, we will notice that in search filters, we will often have the `{0}` or `{1}` tokens. Basically, they let us parametrically build a dynamic filter based on the value of the login information typed by the user, specifically the following:

- ▶ `{0}` will be replaced by the user DN (Domain Name) found during a given search
- ▶ `{1}` will be replaced by the login username typed by the user

## See also

If we are using Pentaho BA server CE, we may be interested in understanding how to configure the LDAP connection manually by looking at the *Configuring authentication through the LDAP server (CE version)* recipe. To get all of the required information about the LDAP properties and their meanings, look at Pentaho Infocenter at the following URL [http://infocenter.pentaho.com/help/index.jsp?topic=%2Fsecurity\\_guide%2Freference\\_ldap\\_properties.html](http://infocenter.pentaho.com/help/index.jsp?topic=%2Fsecurity_guide%2Freference_ldap_properties.html).

## Configuring authentication through the LDAP server (CE version)

In case we are going to use Pentaho BA server CE, things are almost the same as for the EE version. This recipe will show us how to configure Pentaho BA server CE to authenticate it using an LDAP server (OpenLDAP for the current example) without pain. In this case, the procedure is completely manual but everything will become simple by following the recipes correctly.

## How to do it...

1. Go to the `<biserver_home>/pentaho-solution/system` directory.
2. Open the `applicationContext-security-ldap.properties` file with your favorite editor. If this is the first time we are playing with the LDAP authentication, this property file is filled with its default values.
3. Look at the following excerpt; this is a complete configuration to connect and make searches over a typical LDAP server. We can find it in the resource files distributed with this book; it is a fully valid example that we can use to configure our configuration:

```
contextSource.providerUrl=ldap://ldap.acme.com
contextSource.userDn=cn=admin,dc=pentaho-cookbook,dc=com
contextSource.password=password

userSearch.searchBase=ou=users,dc=pentaho-cookbook,dc=com
userSearch.searchFilter=(uid={0})

providerType=ldapApacheConfiguration

populator.rolePrefix=
populator.groupSearchBase=ou=roles,dc=pentaho-cookbook,dc=com
populator.convertToUpperCase=false
populator.searchSubtree=false
populator.groupRoleAttribute=cn
populator.groupSearchFilter= (&(objectClass=posixGroup)
(memberUid={1}))
```

```

adminUser=uid\=pentaho_admin,ou\=users,dc\=pentaho-
cookbook,dc\=com
adminRole=cn\=pentahoAdminsRole,ou\=roles,dc\=pentaho-
cookbook,dc\=com

allAuthoritiesSearch.roleAttribute=cn

allAuthoritiesSearch.searchBase=ou\=roles,dc\=pentaho-
cookbook,dc\=com
allAuthoritiesSearch.searchFilter=(objectClass\=posixGroup)

allUsernamesSearch.searchFilter=objectClass\=posixAccount
allUsernamesSearch.searchBase=ou\=users,dc\=pentaho-
cookbook,dc\=com

allUsernamesSearch.usernameAttribute=cn

```

4. As a general case, in case we are going to use this file as a starting point to configure a Pentaho BA server connection to an LDAP server, remember to change the fully qualified domain name (in our case, `dc=pentaho-cookbook,dc=com`) provided in the sample with the fully qualified domain name for our LDAP server.
5. Look at the `contextSource.providerUrl` key. Change the value according to the configuration specified for our LDAP server.
6. Look for `contextSource.userDn`. Change the content with the distinguished name of a valid LDAP user for our LDAP server. Typically, this could be an LDAP user who has the right to make queries all over the user's tree, not necessarily an LDAP administrator.
7. Look for `contextSource.password`. This is the password for the user mentioned at the previous step. Change it accordingly.
8. Do not forget to make any other appropriate changes either in search expressions or in configuration flags or both; they are needed in case we use different attributes to describe the same things.
9. Save the file and close the editor.



The configuration in the preceding excerpt is valid considering the connection of Pentaho BA Server to an OpenLDAP Server. Assume a similar configuration in case you want to connect to an MS Active Directory Server with the only exception of names the attributes used.

Now that we have completed the manual configuration of the LDAP properties file, we need to activate the newly configured LDAP provider. Pentaho is a Spring application with a lot of configuration files so it is very easy to do this. Of course, we need to know where we can make the required changes.

10. Go to the `<biserver_home>/pentaho-solution/system` directory.
11. Open the `security.properties` file with your favorite editor.
12. Change the value for the key provider from Jackrabbit (the default) to LDAP. This enables the LDAP provider to start working by connecting your BA server to your LDAP server.
13. Save the file and close the editor.

The LDAP configuration is finished. Remember that we have to restart our BA server before the new configuration changes take effect.

### How it works...

If someone ever tried to configure LDAP connections to the old BI server CE, it will be very easy to understand that something changed in this new release. Anything about Pentaho's LDAP configuration files was already said in the *Configuring authentication through the LDAP server (EE version)* recipe, so now is a good time to analyze which are the main points where we can expect some differences.

As soon as anyone used to manually configuring LDAP connections in older releases starts to manually configure an LDAP connection in this new release, it will be clear how this procedure is different and easier in the new release.

Let's go over the different sections of the LDAP configuration file. In old releases, we enabled the usage of a new authentication provider by properly wiring up an appropriate set of spring beans by reconfiguring the `pentaho-spring-beans.xml` file. Now it's really easier; all the required beans are already wired properly and, by configuration, we can select the authentication provider to be used. To select the right authentication provider, we just need to change the `security.properties` configuration file located in the `<biserver-home>/pentaho-solutions/system` directory by specifying the name of the provider to be enabled (LDAP, in our case). This is a very easy and clean approach with respect to the past where we had to go through the reworking of a spring file to wire up different authenticator beans. This approach was hard because it required knowledge of how to properly manage spring files and the names of the files to wire up.

Then, the major change happened. As soon as we changed the authentication provider by mapping the security to the LDAP provider, we set the following two things:

- ▶ We mapped an existing enterprise LDAP group to the role of the Pentaho administrator
- ▶ We also mapped a set of default ACLs to specific Enterprise LDAP groups that will be recognized as Pentaho roles

---

Before Pentaho BA 5.0, these things were created by changing some lines in the `pentaho.xml` file and specifically changing the following two things:

- ▶ Change the value of the `pentaho-system/acl-voter/admin-role` element to the LDAP group name you want to consider as the Pentaho administrators role
- ▶ Change values of the `pentaho-system/acl-publisher` children elements (`default-acls` or `overrides`) to assign default or overrides to certain LDAP groups

Now, things radically changed by prioritizing the ease of configuration.

- ▶ There is no need to explicitly map the name of the LDAP group that will contain the Pentaho administrators because it is implicitly mapped to the default Pentaho's role administrator. While we configure the `applicationContext-security-ldap.properties` file located in `<biserver-home>/pentaho-solutions/system`, we automatically set the fully qualified name of the LDAP group that will be considered as the Pentaho administrators' role. We do this implicitly by setting the value of the `adminRole` key.
- ▶ The default ACLs are implicitly set by setting the appropriate set of **Operations Permissions** from the **Manage Roles** tab in the **Users / Roles** wizard of the **Administration** perspective.

So what else can we say except that the exact same thing seems easier to do?

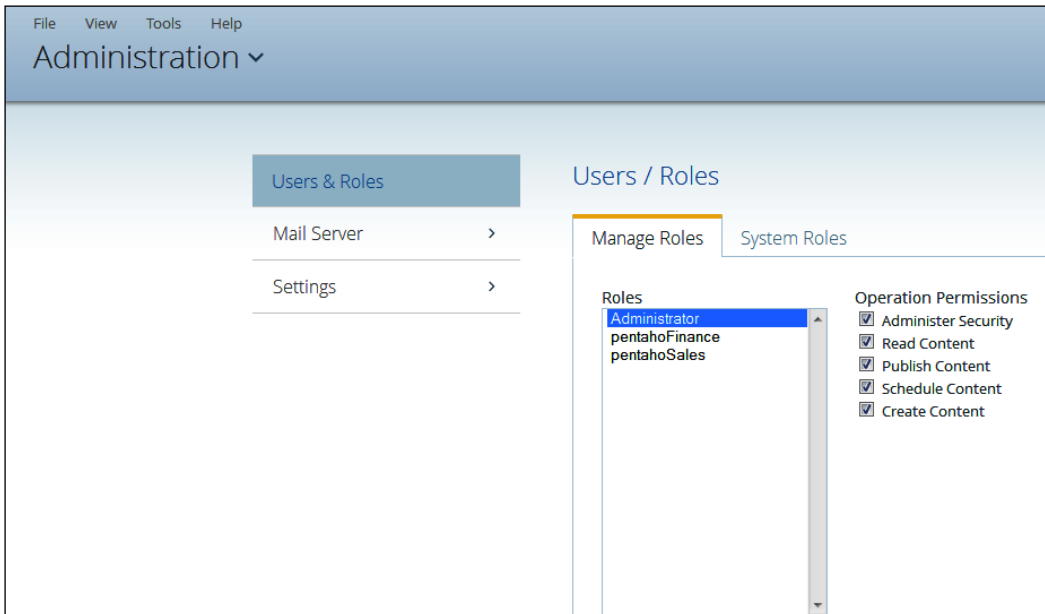
### There's more...

As soon as we connect our security to an LDAP server, the management of our users and roles becomes completely different.

The users' creation and maintenance is completely delegated to the LDAP server as a central unified place in our network that will accomplish this task. Usually, it is the responsibility of our system administrator to manage the task of creating and maintaining users. They know better than us how to manage these kind of things (and we will be very happy to delegate this task). Apart from this, the management of Pentaho roles takes a bit more of our time because even if we associate users to Pentaho roles in the LDAP Server by using LDAP groups, we need to assign to any inherited LDAP group (that will act as a Pentaho role) the related Pentaho **Operation Permissions** as in any other case.



To do this, we always follow the *Editing an existing role* recipe, look at paragraph where it talks about setting **Operation Permissions**, and do what is described to assign **Operation Permissions** to the roles we get from LDAP server. The interesting thing to notice in this case is that (as shown in the following screenshot) as soon as we go in the **Users / Roles** section of the **Administration** perspective, Pentaho will show us a different **Users / Roles** wizard.



As we can see, the following points are clear:

- ▶ The **Manage Users** tab is not present
- ▶ The **Manage Roles** tab dialog box contains only the set of roles we get from LDAP with the option to manage **Operation Permissions**

A lot of this is because the real user management is delegated to our LDAP Server. The only thing we can do in Pentaho is manage Pentaho security.

## See also

Look at the *Configuring authentication through the LDAP server (EE version)* recipe for a detailed description about sections of the `applicationContext-security-ldap.properties` file and its intended use.

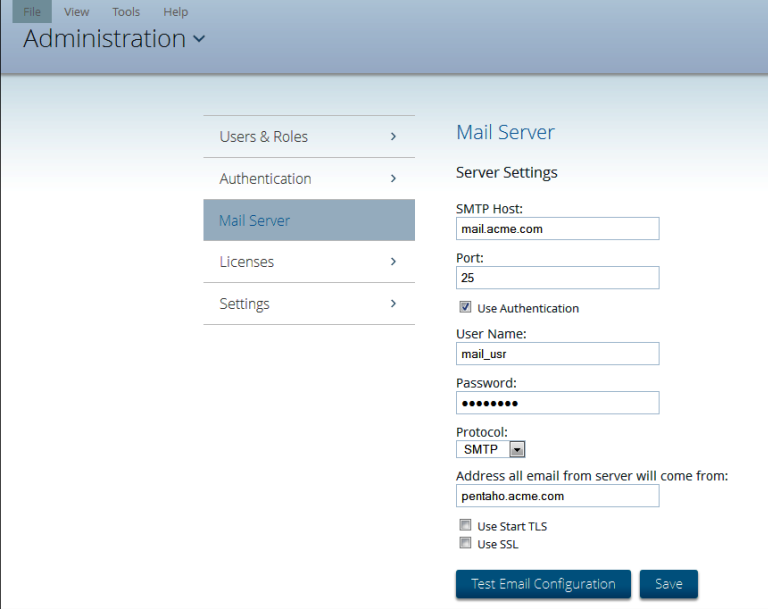
## Managing the mail server configuration

Pentaho, as a Business Analytics platform, gives us all the tools needed to analyze and get insights from our data. A useful functionality is using the platform to monitor our KPIs and send a set of alerts or notifications as soon as these KPIs are going or have already gone out of acceptable threshold values. To do this, we can use e-mail and send out messages to warn users about these unacceptable conditions. Until Version 4.8, e-mail configuration for Pentaho was a cumbersome task because in the CE version, for example, we had to work directly with configuration files. Starting from this new version, the e-mail configuration wizard has been unified both for the CE and EE version in the **Administration** perspective.

### How to do it...

These steps will show us how we can easily configure the e-mail subsystem in Pentaho BA Server:

1. The **Users / Roles** entry of the left menu is automatically selected by default.
2. Select the **Mail Server** entry from the left menu and the **Mail Server** configuration form opens in the **Administration** perspective.
3. If it is the first time we are configuring a connection to a **Mail Server**, the configuration form will be empty. If this is not first time we are going to configure the **Mail Server** connection, the form will display the current **Mail Server** configuration as shown in the following screenshot:

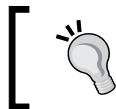


The screenshot shows the Pentaho Administration interface. The top navigation bar includes 'File', 'View', 'Tools', and 'Help'. Below it, the 'Administration' menu is open, showing options like 'Users & Roles', 'Authentication', 'Mail Server' (which is selected), 'Licenses', and 'Settings'. The main content area is titled 'Mail Server' and contains a 'Server Settings' section. The settings are as follows:

- SMTP Host: mail.acme.com
- Port: 25
- Use Authentication
- User Name: mail\_usr
- Password: [masked]
- Protocol: SMTP
- Address all email from server will come from: pentaho.acme.com
- Use Start TLS
- Use SSL

At the bottom of the form, there are two buttons: 'Test Email Configuration' and 'Save'.

4. In the **SMTP Host** field, type the hostname or the IP address of the **Mail Server** that we are going to connect to. This field is mandatory.
5. In the **Port** field, type the hostname or the IP address of the **Mail Server** that we are going to connect to. The field is already filled with a value, **25**, (the default port number for an SMTP server) but we are free to change it according to our needs. This field is mandatory.
6. If our **Mail Server** is an authenticated mail server, please check the **Use Authentication** checkbox. Uncheck it in case our server does not require any authentication. The default value for the checkbox is checked. This field is mandatory.
7. In case we checked the **Use Authentication** checkbox, in the **Username** field, type the name of the user to be used to get authenticated to the **Mail Server** to which we are going to connect. This field is mandatory.
8. In case we checked the **Use Authentication** checkbox, in the **Password** field, type the password of the user to use to be authenticated to the **Mail Server** to which we are going to connect. This field is mandatory.
9. From the **Protocol** drop-down list, select the type of protocol we want to use to communicate with the **Mail Server**. You can choose either **SMTP** or **SMTPS**.
10. In the **Address all emails from server will come from the** field, type the e-mail address of all of the incoming e-mails received by our users with whom we want to be associated. This is the e-mail address that will be displayed by our e-mail client as the **From** e-mail address. This field is mandatory.
11. Check **Use Start TLS** if our server requires **Start TLS connection**.
12. Check **Use SSL** if our server requires **SSL connection**.
13. Press the **Test SSL** connection to verify that everything is correct in the configuration that we have defined so far.
14. If something in the configuration is wrong, the server returns with a warning that suggests us to review our configuration, correct any errors, and try to test it again.
15. If everything is correct, the server returns with a success message and informs you to check your inbox for the account specified in the **Address** field all the e-mail from the server will come from for a confirmation message.
16. Press the **Close** button to close the information dialog box.
17. If the configuration is correct, click on the **Save** button to confirm our configuration.



Unfortunately, in this case, the server does not inform us about the success of the save operation; it only gives us an exception in case the operation fails.



## How it works...

Starting from this release, e-mail configuration has become a bit friendlier because Pentaho developed a configuration form wizard to assist us with this in any version (either CE or EE). Until the release of Version 4.8, to configure the e-mail in the CE version, a user had to edit an XML configuration file named `email-config.xml`, which is located in `<biserver_home>/pentaho-solution/system/smtp-email`. Even if the file was well commented, it was always a little bit trickier for a normal user, and it required quite a bit of technical knowledge to locate the file on Pentaho BA Server directories and also to know the meaning of the various properties and their values. Only the EE version had the wizard to help the user configure the **Mail Server** connection. Now, the wizard is present in the CE version too and requests for the same information that we found in the file we mentioned so far but in a friendlier way. Lastly, when the user clicks on the **Save** button, the server updates that same file.

## There's more...

Many users use Google's Gmail as the e-mail provider of choice so it is a good idea to spend a few words talking about this.

### Configuring e-mail to connect to Gmail

The SMTP e-mail configuration has a few things to take into consideration as detailed in the following steps:

1. Remember that the name of Google's **SMTP Host** is `smtp.gmail.com`. Type this name in the **SMTP Host** field.
2. Gmail requires an authenticated connection so be sure that the **Use Authentication** checkbox is checked.
3. Insert a valid Gmail username and password. Remember to type the username followed by its domain (for example, `sample.username@domainname.com`) in order for it to be correctly understood by Google.
4. The port to connect to Google's SMTP server is 587. Type this value in the **Port** field.
5. From the **Protocol** drop-down list, select **SMTPS**.
6. Set the value for the **Use Start TLS** checkbox to the checked value.
7. Set the value for the **Use SSL** checkbox to the checked value.

## See also

If you are a newbie looking for details about the SMTP protocol, look at the article at <http://en.wikipedia.org/wiki/SMTP>. For details on how to configure a Gmail account with POP3 or IMAP, see the article at <https://support.google.com/mail/troubleshooter/1668960?rd=1#ts=1665119,1665162>.

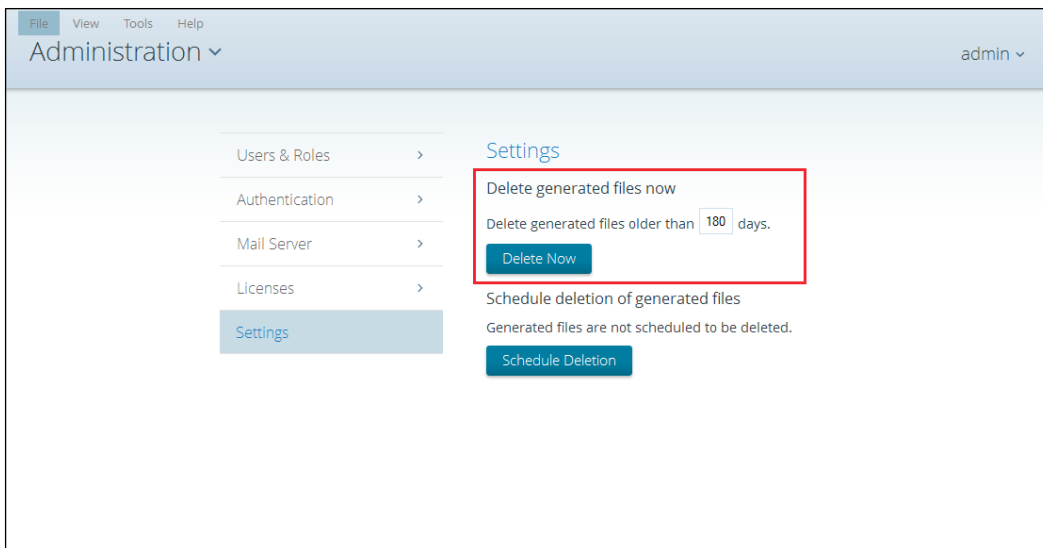
## Cleaning up aged generated files immediately

In any case where we run the execution of a content item, we have the production of temporary files (either iPDF, Excel, images, or any other thing). This content is produced in a specified location and must be regularly cleaned up so it doesn't fill the disk at a level that could be dangerous for the system. This recipe shows us how we can run a task from the **Administration** perspective to immediately clean up that kind of content.

### How to do it...

The following steps detail how we can run a task from the **Administration** perspective to immediately clean up aged generated files:

1. Select the **Settings** entry from the left menu in the **Administration** perspective.
2. The **Settings** configuration form is opened, as shown in the following screenshot:



3. Look at the form's **Delete generated files now** section as shown in the previous screenshot.

4. Set the maximum age of the files in the **Delete generated files older than** field.
5. Click on the **Delete Now** button.
6. The task of deletion starts immediately and the temporary content is immediately deleted.

### How it works...

Cleaning up aged generated files is an important operation to keep the usage of our disk under control. By going in the **Administration** perspective and choosing the **Settings** entry, we can set an aging value for our files by filling the **Delete generated files older than** field. By clicking on the **Delete Now** button, a background task will immediately delete the older content files.

### See also

We can take a look at the *Scheduling cleanup of aged generated files* recipe if we are interested in scheduling the maintenance and deletion of aged files.

## Scheduling the cleanup of aged generated files

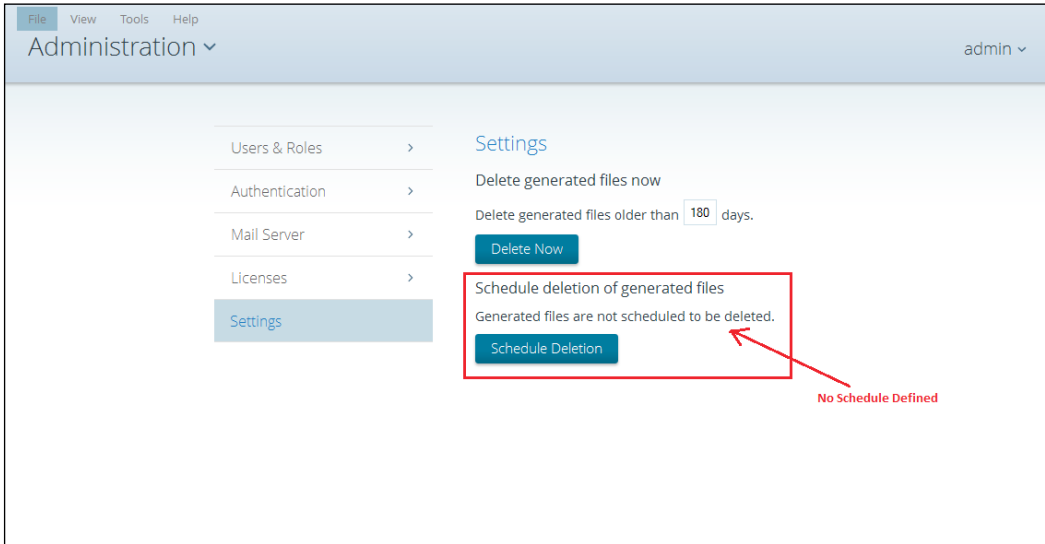
The previous recipe showed us how we can delete temporary files immediately. The current recipe shows us how we can do the same thing but by scheduling the task that deletes the temporary files so that it can automatically be executed periodically and maintain the temporary directory cleanup without any effort.

### How to do it...

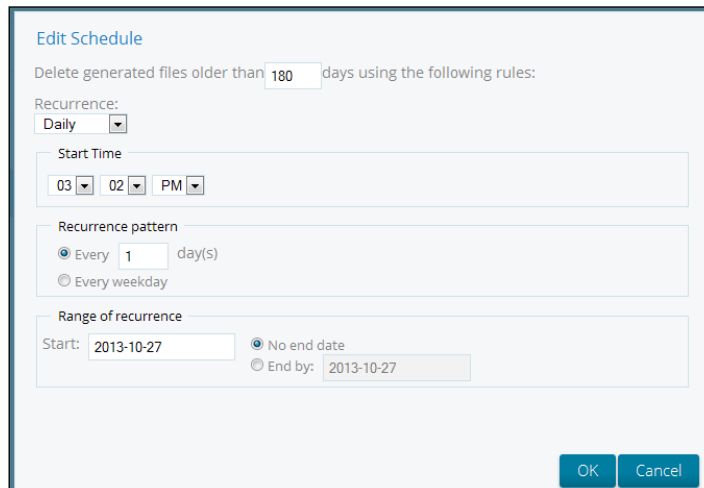
The following steps detail how we can schedule the cleanup of aged generated files:

1. Select the **Settings** entry from the left menu from the **Administration** perspective.

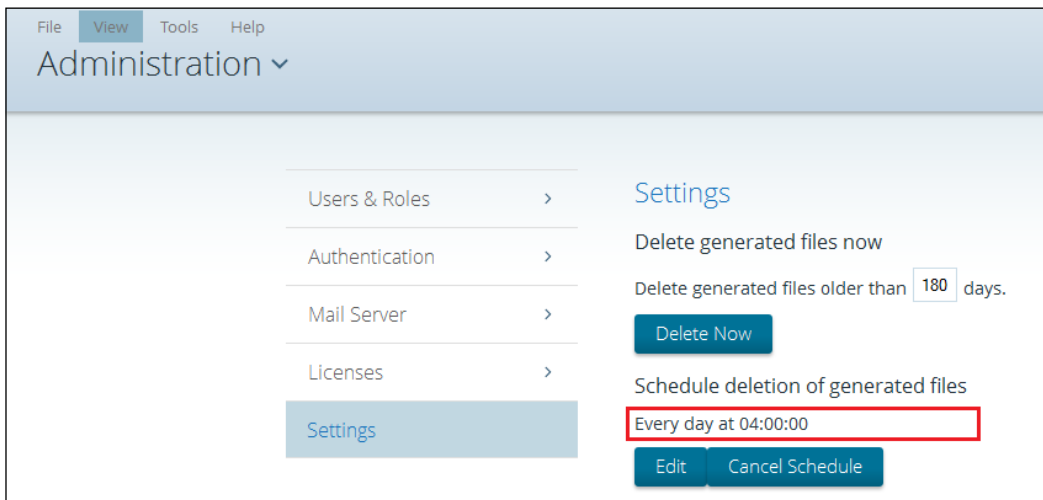
- The **Settings** configuration form opens as shown in the following screenshot:



- Look at the **Schedule deletion of generated files** section of the form as shown in the previous screenshot.
- Initially, if we never set a schedule, a label will inform us that no schedules are defined to delete temporary files.
- Click on the **Schedule Deletion** button.
- The **Edit Schedule** dialog box opens to let us set the properties and launch the deletion task according to our needs.



7. Set the maximum age of the files to delete by typing a value in the **Delete generated files other than** field. The default value is **180** days.
8. Fill in all the parameters needed to rightly schedule the files' deletion.
9. Click on the **Cancel** button to close the dialog box without confirming the new schedule. In this case, no schedules are created at all.



10. Click on the **OK** button to confirm the schedule and close the dialog box. A detail of the schedule we just set is shown in the **Settings** screen as shown in the previous screenshot.
11. If we want to reset the schedule that we just created, we click on the **Cancel Schedule** button shown in the previous screenshot.

### How it works...

Managing the deletion of aged files using the schedule is very simple. It creates a new schedule behind the scenes to automatically start the execution of the deletion task. The task started is the same task that we start if we want to immediately delete the temporary files, and you can start it by pressing the **Delete Now** button (see the previous recipe for details about this). Remember that this schedule will not be visible to users because it is considered a system schedule.

### See also

To see how we can delete aged temporary files immediately, see the *Cleaning up aged generated files immediately* recipe.





# 3

## Defining BA Server Data Sources

In this chapter, we will cover the following topics:

- ▶ Creating a new native JDBC data source
- ▶ Defining a JNDI connection in the BA Server
- ▶ Creating a new JNDI JDBC data source
- ▶ Updating an existing JDBC data source
- ▶ Creating a new analysis data source
- ▶ Updating an existing analysis data source
- ▶ Creating a new metadata data source
- ▶ Exporting an existing data source
- ▶ Creating a new data source from a CSV file using the wizard
- ▶ Deleting an existing data source

### Introduction

The previous two chapters took us through the Pentaho User Console's user interface. We covered the common use cases of user's activities in depth, and we showed new ways to accomplish administrative tasks through the new **Administration** perspective.

This chapter introduces data sources and how to define them in a new Pentaho BA Server user interface. Data sources are the natural entry points to each and every possible source of data that can fill our BI objects (reports, analysis views, dashboards, or others).

Pentaho BA Server is a J2EE web application. In any Java application, an application's connection to a database is made through a JDBC connection. Basically, a JDBC connection is a way for a Java application to communicate with an RDBMS database by means of a driver (the JDBC driver) and a set of connection information, such as server URL, username, and password. In our Pentaho Business Analytics Server, any components such as reports, OLAP cubes, and dashboards are filled with data, so they require an active connection to our data.

To make things easier and clearer for users, Pentaho uses the concept of **Data Source** to indicate an active connection to a source of data. Any **Data Source** is identified through a type and a name. The type identifies the type of data we are accessing (RDBMS, OLAP, metadata, and so on) at an abstract level. The name is referenced in our Pentaho applications and is used to get a reference to an active connection back to the referenced source of data. As we can see, in Pentaho, the concept of **Data Source** is completely generic and covers different possible types of sources.

This **Data Source** concept was common in all the older releases of Pentaho BA Server. What has changed in its last release is a new configuration dialog box that is available in Pentaho User Console. The next set of recipes will show the data sources' configuration to help us in our everyday **Data Source** configuration tasks.

The recipes in this chapter are based on the assumption that we are able to successfully log in to Pentaho User Console as a user that is a part of the **Administrator** role. Only the members of this role can access the **Data Source** configuration wizard in our Pentaho BA Server.

In case we want to use demo users, remember that we can use the administrator's username and password for the demo users' username and password. This user is the new Pentaho demo administrator after the famous user **Joe** (the Pentaho-recognized administrator until Pentaho 4.8) was dismissed starting from Version 5.

## Creating a new native JDBC data source

When we talk about a JDBC **Data Source** in Pentaho BA Server, we mean a Pentaho **Data Source** that connects to an RDBMS database.

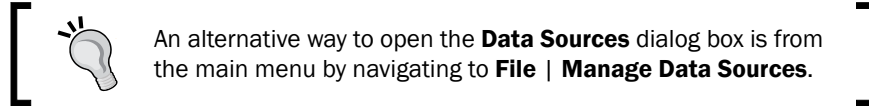
A JDBC **Data Source** is used by all our reports, OLAP cubes, dashboards, and so on, to get data from a relational database. A Native JDBC Data Source is the easiest type of JDBC data source that we can configure in Pentaho. In this specific type of Pentaho **Data Source**, all of the databases' connection properties are specified directly in the **Data Source** configuration dialog box. This recipe will show you how to define a new **Native JDBC Data Source**.

### How to do it...

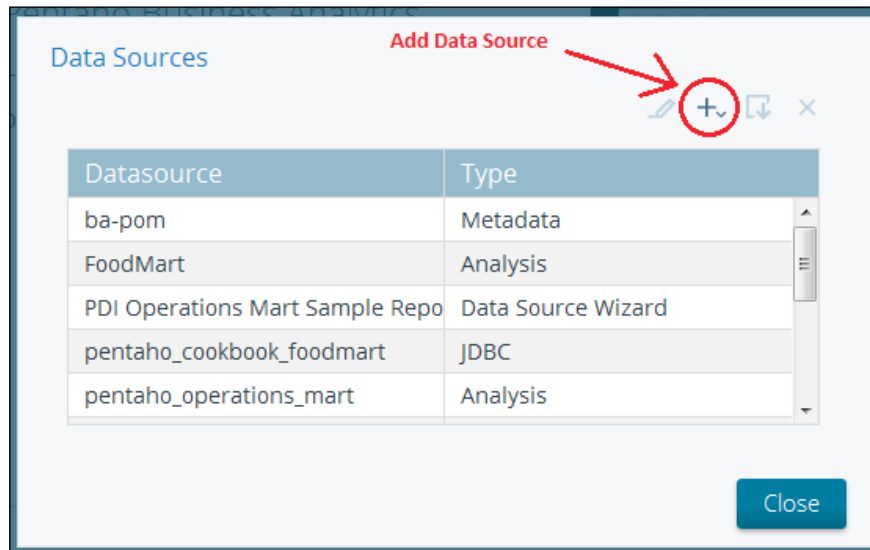
The following steps detail how we can define a new **Native JDBC Data Source**.

1. From the **Home** perspective, click on the **Manage Data Source** button.

- The **Data Sources** dialog box appears.



- Click on the **Add Data Source** icon as shown in the following screenshot:



- The **Datasource** and **Type** combobox drops down and shows the various types of data sources that we can define. From the list, choose **JDBC**.
- JDBC's **Database Connection** dialog box opens.
- Configure the **JDBC Native Datasource** connection by filling in all of the required connection information. Click on the **Test** button to verify whether everything works. A message box will inform us whether the connection succeeded or failed.
- If the connection test was successful, click on the **OK** button to confirm, and close the JDBC's **Database Connection** dialog box. If the connection test was unsuccessful, check the errors and click on the **Test** button again.
- As soon as we click on the **OK** button and the JDBC's **Database Connection** dialog box closes, the **Data Sources** dialog box will show again. Now, the data sources list is updated with the data source just added.
- Click on the **Close** button to close the **Data Sources** dialog box. The new data source is immediately visible to Pentaho BA Server without the need to restart the application.

## How it works...

A JDBC data source is one of the many possible types of data sources that we can define in the system. This lets us connect to our RDBMS in order to get data from it. As soon as the JDBC's **Database Connection** dialog box opens, we have plenty of configuration items that need to be filled, which are divided into a set of categories on the left-hand side. The **General** item entry is selected by default from the menu on the left. Under this category, we configure the very basics of a database connection as detailed in the following points:

- ▶ In the **Connection name** field, type the name of the connection we are going to create. This field is mandatory, and it is very important because it will be the name that you use in any report or dashboard in order to get data from the data source. We will talk about how this works from a technical standpoint in the next recipe.
- ▶ From the **Database Type** list box, select the database type we are going to connect to. There are a lot of database types available. This list is automatically populated by Pentaho as soon as a new JDBC driver is put into the right place under the Pentaho server. We will talk about this in a minute in the next *There's more...* section.
- ▶ From the **Access** method list box, select the **Native (JDBC)** access method. This access method type requires that we specify all the RDBMS connection information (username, password, and server URL) in the **Data Source** configuration. In the *Creating a new JNDI JDBC data source* recipe, we will see how we can externalize the RDBMS connection information from the **Data Source** configuration for more flexibility. The following screenshot shows the set of fields required to properly configure the **Data Source**.

- ▶ On the right-hand side, we find the set of fields to configure the required RDMBS information. This set of fields can be different depending on the **Database Type** we selected.
- ▶ As soon as all the information is filled in, we can test the connection by clicking on the **Test** button. We immediately get a response about whether the connection to the database succeeded or failed.

### There's more...

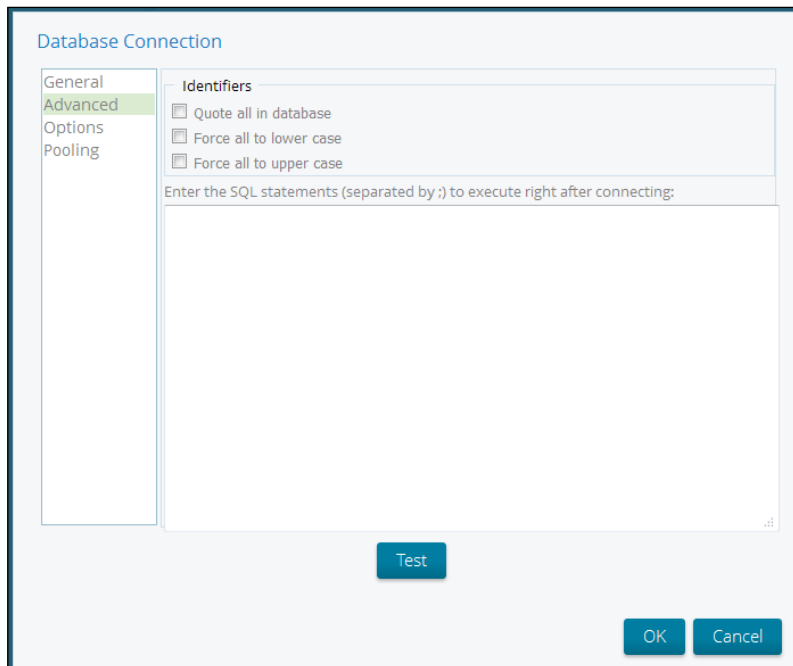
As we saw, the **Database Type** list is already filled with a set of the default database types. In any case, we can always easily add other database drivers to the Pentaho BA Server by enriching the available database types. Moreover, the left-hand side of the **Database Connection** dialog box contains a set of menu item entries either to specify further properties to the connection configuration or to enable advanced features such as pooling. The next two paragraphs will be about how to add new JDBC drivers and about the specification of a set of advanced configuration properties.

## Adding another database type to the list

Adding a database JDBC driver that is missing from the list is very easy. First, we add the JDBC driver's jar file in the <biserver\_home>/tomcat/webapps/pentaho/WEB-INF/lib directory under our BA Server directory structure. As soon as the server loads all the libraries from the classpath, the added jar file will be immediately recognized as a JDBC driver, and the name of the newly added database will appear in the database names list.

## Using advanced configuration parameters

We can access some advanced JDBC configuration properties by selecting the **Advanced** menu entry from the left-hand side menu of the **Database Connection** dialog box. The following screenshot shows the **Advanced** section of the **Database Connection** dialog box:



By selecting the **Advanced** menu entry, we have a set of available fields as detailed in the following points:

- ▶ **Quote all in database:** This will force all the fields and table names, in the query that is sent to the database, to be enclosed with the " character
- ▶ **Force all to lower case:** This will force all the fields and table names, in the query that is sent to the database, to be written using lowercase character strings
- ▶ **Force all to upper case:** This will force all the fields and table names, in the query that is sent to the database, to be written using uppercase character strings

- ▶ **Enter the SQL statements (separated by ;) to execute right after connecting:** This large textbox is available to add a script that can be executed immediately after the connection with the RDBMS has been established in order to set connection variables, execute auditing queries, or add other commands that must be executed to initialize the connection

Remember that these definitions will be confirmed as soon as we click on the **OK** button in the JDBC **Database Connection** dialog box.

### See also

If you are interested in understanding the details of the **Options** and **Pooling** configuration menu items in JDBC's **Database Connection** dialog box, take a look at the *Creating a new JNDI JDBC data source* recipe.

## Defining a JNDI connection in the BA Server

**JNDI (Java Native Directory Interface)** is a Java API that lets a client application look up data and objects published under a name in a directory service. This API is used by a J2EE server to publish various kinds of resources initialized by the application server and made available to various clients. Because Pentaho BA Server runs on a Java server, we can define a set of JNDI database connections in the Java server and have them available as a Pentaho Data Source (we will see this in the next recipe). The ability to define a JNDI connection in the Pentaho server is a pre-requisite for our next recipe.

### Getting ready

To get ready for this recipe, remember to check whether our Pentaho BA Server instance is switched off. If not, please switch it off before you continue with this recipe. It would also be considered a good thing to have a little background knowledge about how to configure a Tomcat server instance.



## How to do it...

Usually, JNDI connections are configured directly in the Java server configuration. This is because it is Java server's responsibility to open the pool of connections during system startup and make these connections available to all the requesting clients using a connection name. Pentaho BA Server is distributed out of the box in a bundle with Tomcat, but because it is a standard Java web application, it can be deployed to any other Java application server too. In any case, as Tomcat is the default engine for the Pentaho server, we are going to show you how to define a set of JNDI connections in Tomcat. The same configuration for other server types must be searched in the related server's documentation toolset. Let's configure the set of JNDI connections as detailed in the following steps.

To define a new JNDI database connection, we need to update a Tomcat configuration file. Depending on the visibility of the connection we are going to define, the same configuration file is located in two different places, as we will see later. Remember that we can define a public JNDI database connection (visible to all of the web applications deployed under this Tomcat instance) or a private JNDI database connection (visible only to Pentaho BA Server).

1. To define a public JNDI database connection, go to the `<biserver_home>/tomcat/conf` directory and open the `context.xml` file.
2. To define a private JNDI database connection, go to the `<biserver_home>/tomcat/webapps/pentaho/META-INF` directory and open the `context.xml` file.
3. As we can see in the first two points, the two files are the same; the different places where we locate them relate to the different scopes of the database connection.
4. Let's define a new MySQL connection named `theCookbookConnection` to a database named `myCookbookSamples` on `cookbookServer`. The connection uses a user, `cookbook_user`, with the password as `password`. To do this, add the following XML fragment to the Tomcat configuration file we opened in any place we prefer inside the `<Context>` element:

```
<Resource name="jdbc/theCookbookConnection"
  auth="Container" type="javax.sql.DataSource"
  factory="org.apache.commons.dbcp.BasicDataSourceFactory"
  maxActive="20"
  maxIdle="5"
  maxWait="10000"
  username="cookbook_user"
  password="password"
  driverClassName="com.mysql.jdbc.Driver"
  url="jdbc:mysql://cookbookServer:3306/myCookbookSamples"
/>
```

5. This is a standard Tomcat resource factory configuration definition. We can find further details about the attributes used to configure the connection by looking at the Tomcat documentation.

6. We need to add an XML fragment similar to our sample for any new JNDI JDBC connection we want to add to the Pentaho BA Server Tomcat server.
7. Now that we have defined the connection in Tomcat, we need to inform the Pentaho application about the new connection.
8. Go to the `<biserver_home>/tomcat/webapps/pentaho/WEB-INF` directory.
9. Open the `web.xml` file. This file is the web application's deployment descriptor for the Pentaho web application. This file contains all of the information used by the Java servlet engine to configure the Pentaho User Console web application and is usually created by the Pentaho development team.
10. Look for the `<!-- insert additional resource-refs -->` tag at the end of the page.
11. Immediately after the tag, add the following XML fragment:

```
<resource-ref>
  <description>The Just Added Connection Name</description>
  <res-ref-name>jdbc/theCookbookConnection </res-ref-name>
  <res-type>javax.sql.DataSource</res-type>
  <res-auth>Container</res-auth>
</resource-ref>
```

12. This fragment is used to inform our web application that a database connection named `jdbc/theCookbookConnection` is available in the Java server JNDI registry. We need to add a code fragment similar to this for any connection we defined in the Tomcat server.

## How it works...

In J2EE architectures, JNDI solves a very common problem: having a centralized place where we can publish and share application resources. As you know, an instance of any J2EE application server can manage a wide set of web applications. Imagine a scenario where any of these applications could individually manage their own database connections. Furthermore, suppose that more than one application must use the same connection. This practically means the following:

- ▶ There are different places where the connection information is specified
- ▶ We have different and separate connection attempts

The first problem in a typical J2EE architecture is solved by using a JNDI registry under which we have a set of connections available for our applications. Any specific connection to any database is identified by a name, as a usual way to access any JNDI resource. Therefore, a JNDI connection is a connection that is preventively opened by a J2EE application server (or by a J2EE servlet engine provided with the necessary support to do this) and is published with a name under a standard JNDI registry. All of the applications deployed in the context of the J2EE application server can access that specific connection through its name.

The second problem is solved through the usage of a connection pool. Let our server open a set of connections to the specific database, and suppose that any time the application requires a connection, the server doesn't create it but just takes this connection from a set of available connections. As soon as it finishes using the connection, the connection is not really closed but is returned to the pools and flagged as available. This is the idea that stays behind the concept of a connection pool.

We will see that Pentaho, while defining a data source, gives you the ability to associate that data source with a connection pool in order to optimize the database access. To effectively tune our pool, we have a set of parameters that let us specify its initial size, maximum size, and growth percentage.

### There's more...

Connecting to our data warehouse through **Data Sources** has some plus points, explained as follows:

- ▶ Getting data from a data source identifies one central location to maintain the connection information outside of our objects. Why is this so important? Suppose our BA Server installation is deployed with a great number of reports, dashboards, or OLAP cubes and that something changes in our database connection parameters. If any object has a referenced set of connection information, we need to spend a certain amount of time in doing the repetitive job of modifying all the connection information defined in the BA objects we deployed. By using a data source, this will not happen because we just take care of this information in one place.
- ▶ Getting data from a data source enables the usage of connection pooling capabilities by enabling better performance to get data from our sources. Connection pooling is about maintaining a pool of opened connections to our RDBMS source and by reusing this pool of connections by removing any latency due to the time required to open a new connection to the database. This is another good opportunity to favor the use of a data source.

### See also

For further details about how to configure a JDBC connection in the Tomcat resource factory, look at the Tomcat documentation at [http://tomcat.apache.org/tomcat-6.0-doc/jndi-resources-howto.html#JDBC\\_Data\\_Sources](http://tomcat.apache.org/tomcat-6.0-doc/jndi-resources-howto.html#JDBC_Data_Sources). Other information about how to manually configure a JNDI connection in order to use it in the Pentaho platform can be found in the Pentaho infocenter at [http://infocenter.pentaho.com/help/index.jsp?topic=%2Fconfig\\_ba\\_server%2Ftask\\_adding\\_jndi\\_data\\_source.html](http://infocenter.pentaho.com/help/index.jsp?topic=%2Fconfig_ba_server%2Ftask_adding_jndi_data_source.html).

---

## Creating a new JNDI JDBC data source

As soon as we define a new JNDI connection in the Java server, we can use it under the Pentaho server by creating a new **JNDI JDBC Data Source**. **JNDI JDBC Data Source** is the second type of **JDBC** data source we can configure in Pentaho BA Server. We can configure it by defining the name of the JNDI connection we configured in the previous recipe in the configuration dialog box. Therefore, we are talking about another kind of JDBC data source here, but all of the databases' connection information is specified outside of the data source definition.

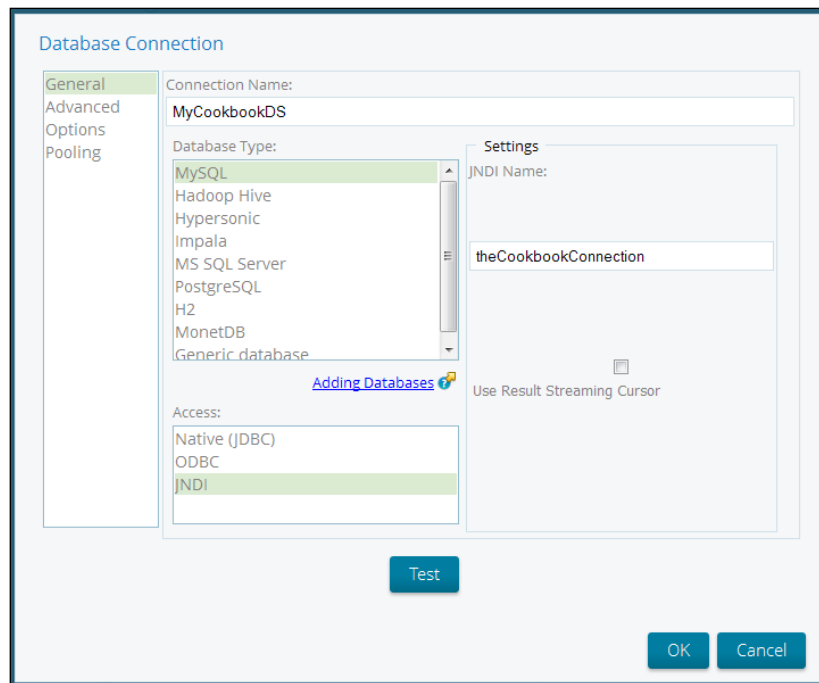
### How to do it...

To define a new JNDI data source, we need to perform the following steps:

1. From the **Home** perspective, click on the **Manage Data Source** button.
2. Open JDBC's **Database Connection** dialog box as specified in steps 2 to 6 of the *Creating a new native JDBC data source* recipe.
3. Configure the **JNDI JDBC Data Source** connection by selecting **Database Type**. Then, select **JNDI** from the **Access** list box.
4. Type the name of the JNDI connection we want to use in the **JNDI Name** text field.
5. Click on the **Test** button to verify whether everything works. A message box will inform us whether the connection succeeded or failed.
6. If the connection test was successful, click on the **OK** button to confirm and close JDBC's **Database Connection** dialog box. If the connection test was unsuccessful, check the errors and click on the **Test** button again.
7. As soon as we click on the **OK** button and JDBC's **Database Connection** dialog box closes, the **Data Sources** dialog box will show again. Now, the data sources list is updated with the data source we just added.
8. Click on the **Close** button in the **Data Sources** dialog box to close it and terminate the **Data Source** definition process.
9. The new data source is immediately visible to Pentaho BA Server without the need to restart the application.

## How it works...

When defining a **JNDI JDBC Data Source**, as soon as JDBC's **Database Connection** dialog box opens, we have a lower number of configuration items that need to be filled. This is because all of the database configuration parameters were already filled in the body of the JNDI connection definition that we are going to refer. Even in this case, the **General** item entry is selected by default from the menu on the left. However, as soon as the **JNDI JDBC** entry is selected from the **Access** list, all of the database configuration item fields are hidden and the only visible input field is **JNDI Name**, as shown in the following screenshot:



The **JNDI Name** field must be filled by typing the name of the JNDI database connection defined in the Java server configuration (in our case, `theCookbookConnection`). We showed you how to define a JNDI database connection in the previous recipe.



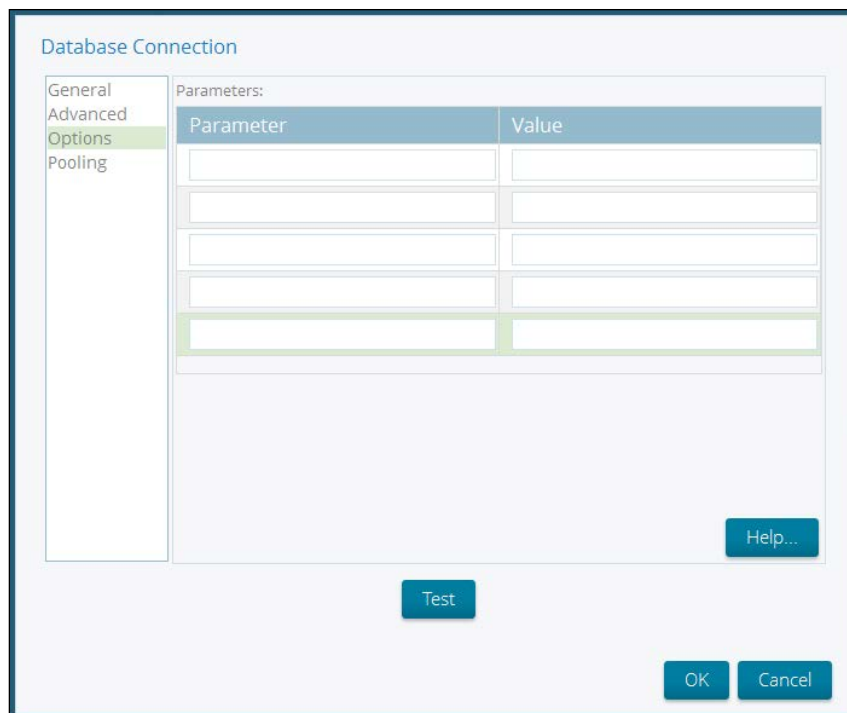
The **Connection Name** box of the Pentaho data source should be identical to the JNDI name of the connection configured in the Pentaho application server; otherwise, our connection will not work. So, remember to double check it before proceeding.

## There's more...

Let's continue analyzing the last JDBC **Database Connection** dialog box configuration feature by looking at the last two item entries of JDBC's **Database Connection** dialog box menu.

### Specifying JDBC connection properties manually

If we are not satisfied with the base connection configuration, we can specify other custom and database-specific JDBC **Database Connection** properties by accessing the **Options** item entry from the left-hand side menu of JDBC's **Database Connection** dialog box. The following screenshot shows the **Options** in the **Database Connection** dialog box:

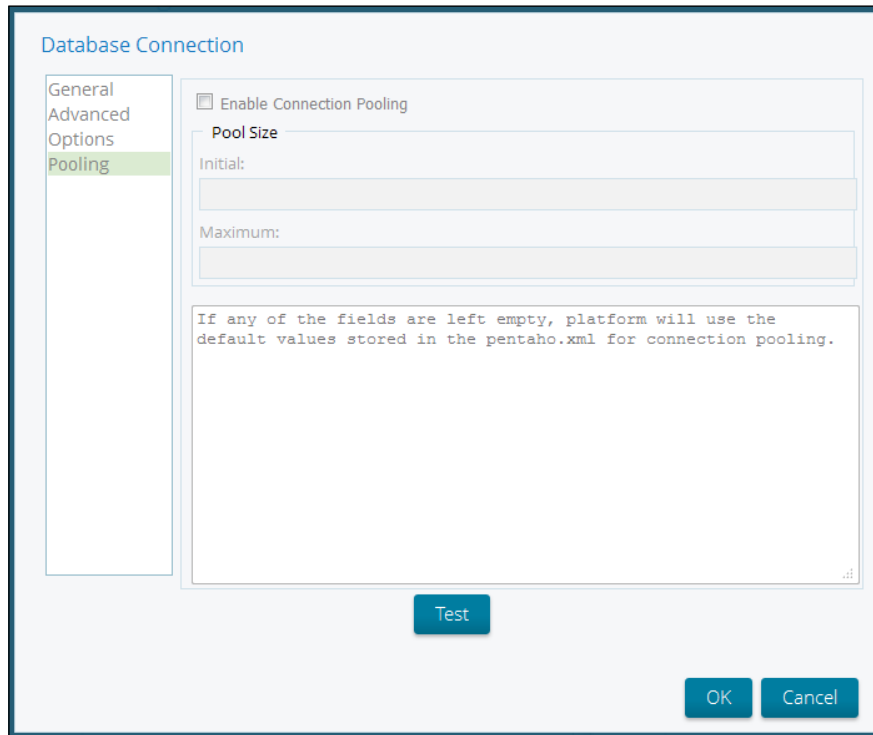


As soon as we select the **Options** menu entry, the dialog box displays a set of **Parameter** and **Value** pairs organized through a set of five rows (this could be the limit, but we cannot add any further pairs to our custom configuration). As soon as we finish specifying the needed pairs of JDBC connection parameters, it is very important to remember that they will be confirmed as soon as we click on the **OK** button in JDBC's **Database Connection** dialog box.

Remember that custom connection parameters are database-specific, so refer to the documentation of our selected database for further details.

## Enabling database connection pooling

Database connection pooling is an important feature to optimize our JDBC connection resources usage. To enable connection pooling for the selected **JDBC Data Source**, we need to select the **Pooling** item entry from the left-hand side menu of JDBC's **Database Connection** dialog box, as shown in the following screenshot:



After we select the **Pooling** menu entry, we can enable the **Pooling** functionalities by checking the **Enable Connection Pooling** checkbox. The next step is to define the initial and maximum pool size by filling the **Initial** and **Maximum** fields respectively. Remember that these definitions will be confirmed as soon as we click on the **OK** button in JDBC's **Database Connection** dialog box.

### See also

If you want to review how to define a JNDI database connection, if you are interested in understanding the details of the Options and Pooling configuration menu items in JDBC's Database Connection dialog box, take a look at the *Creating a new JNDI JDBC data source* recipe. Moreover, we suggest that you review the *Creating a new native JDBC data source* recipe for details about the **Advanced** configuration menu items in JDBC's **Database Connection** dialog box or about how to add a new JDBC driver.

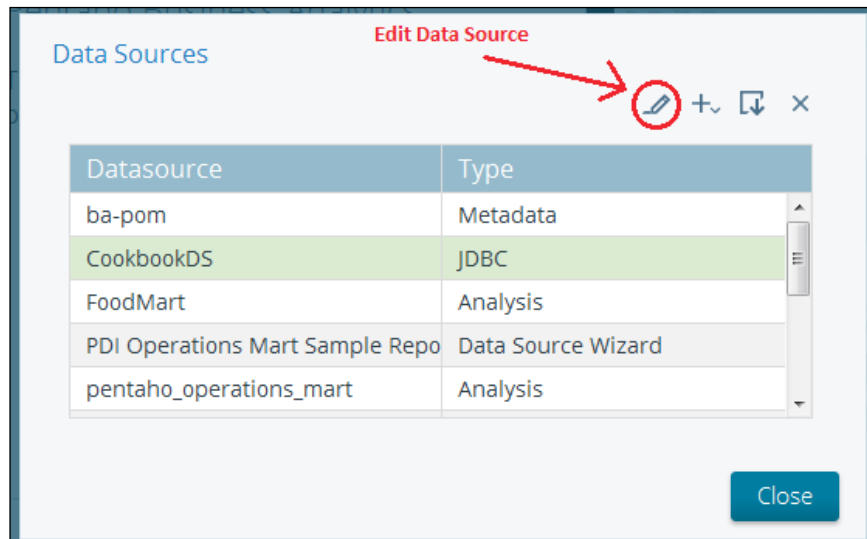
## Updating an existing JDBC data source

Let's see how we can update an existing JDBC data source.

### How to do it...

To update an existing JDBC data source, perform the following steps:

1. From the **Home** perspective, click on the **Manage Data Source** button.
2. The **Data Sources** dialog box appears.
3. An alternative way to open the **Data Sources** dialog box is by navigating to **File | Manage Data Sources** in the main menu.
4. From the data sources list, select the data source you want to update by clicking on the data source item.
5. Click on the **Edit Data Source** icon as shown in the following screenshot:



6. JDBC's **Database Connection** dialog box opens with the configuration fields filled by the data source configuration values.
7. Change the configuration according to your needs, and click on the **Test** button to verify the data source connection after the change.
8. If the connection test was successful, click on the **OK** button to confirm the new data source definition and close JDBC's **Database Connection** dialog box. If the connection test was unsuccessful, check the errors and click on the **Test** button again.



9. As soon as we click on the **OK** button and JDBC's **Database Connection** dialog box closes, the **Data Sources** dialog box will show again.
10. Click on the **Close** button in the **Data Sources** dialog box to close it and terminate the **Data Source** definition process.

### How it works...

Editing an existing JDBC data source is a very easy task. JDBC's **Database Connection** dialog box is always the same, but depending on the access type of our data source, we can have a different set of input fields. We already went through these details when we were discussing about how to create the two different types of JDBC data sources (either **JDBC Native** or **JDBC JNDI**). When editing our data source, we can also change between the two access types without any issues.



If we change anything in the definition of our data source, these changes will not be effective until we restart the Pentaho BA Server instance.

### See also

If you are interested in reviewing the details about the definition of a JDBC data source, take look at the *Creating a new native JDBC data source* or *Creating a new JNDI JDBC data source* recipes.

## Creating a new analysis data source

A second kind of data source is the OLAP or **Analysis** data source. The **Analysis** data source lets our reports and dashboards connect with an OLAP cube and obtain data from it. This recipe will show you how to create an **Analysis** data source.

### How to do it...

To define a new **Analysis** data source, perform the following steps:

1. From the **Home** perspective, click on the **Manage Data Source** button.
2. The **Data Sources** dialog box appears.

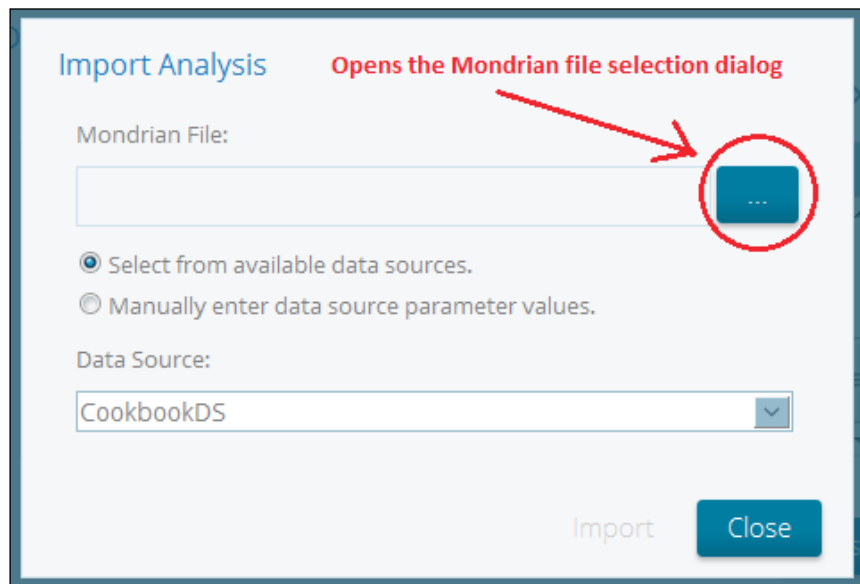


An alternative way to open the **Data Sources** dialog box is by navigating to **File | Manage Data Sources** in the main menu.

3. The **Data Source Types** combobox drops down and shows us the various types of data sources we can define. From the list, select **Analysis**.
4. The **Import Analysis** dialog box opens. Proceed with the definition of the analysis data source by filling all the required fields and loading the schema definition for the OLAP cube we are going to map under this data source. If we want to confirm the new data source configuration, click on the **Save** button. In case we want to close the dialog box and lose any configuration item defined, click on the **Close** button.
5. The **Import Analysis** dialog box closes, and the **Data Sources** dialog box will show again. The data sources list is updated with the data source we just added.
6. Click on the **Close** button in the **Data Sources** dialog box to close it and terminate the data source definition process.
7. The new data source is immediately visible to Pentaho BA Server without the need to restart the application.

### How it works...

An analysis data source is the way a report or a dashboard accesses the data contained in an OLAP cube. Therefore, when defining this type of data source, the first thing we need do is to select the Mondrian schema file that contains the cubes we want to publish as a multidimensional data source. As we will see in a later recipe, a schema can contain more than one cube. An **Analysis** data source lets us access all the cubes defined in the schema published under that data source.



To upload the schema file, click on the button identified by three dots on the right-hand side of the **Mondrian File** input textbox. The previous screenshot shows the location of this button in the **Import Analysis** dialog box. The **File Upload** dialog box opens by giving us the ability to look, on our local filesystem, for the schema file we want to publish on our BA Server. As soon as we find it, select it and click on the **Open** button. When the **Import** button becomes visible, we have set the Mondrian schema file in **File Upload**.

As we know, Mondrian is a **ROLAP (Relational Online Analytical Process)** engine, which means it is an OLAP engine whose data resides on a relational database. To map the relational schema to a multidimensional structure, a metadata file called a Mondrian schema file is needed. This file describes, for example, the relationship between our star schema tables and dimensions, hierarchies and levels in a multidimensional structure.

Now that we know what a ROLAP engine is, we understand that to properly feed our Mondrian cubes, we need to connect Mondrian through an appropriate JDBC data source to an RDBMS source to properly feed our cubes. To do this, we have two options as detailed in the following points:

- ▶ Leave the **Select from available data sources** radio button selected (it is by default) in case you want to select the data source from the list of available data sources.
- ▶ Check the **Manually enter data source parameter values** radio button in case you want to manually specify the connection parameter. This way, we have the highest degree of control over our connection to get the most from it with better performance. One of the necessary connection parameters is the name of the data source to use in order to fill the cubes. See how to add data source parameters in the *There's more...* section for details about data source parameters.

As soon as we finished defining all of the required configuration items for the current analysis data source, click on the **Import** button, upload the schema file to the BA Server, and confirm the data source definition.

### There's more...

There could be some situations where we need to manually define the set of parameters that configure the analysis data source. These parameters are the reference to the JDBC data source, but there could be many other depending on the situation. Let's see how to do this.

### Defining data source parameters manually

As we saw, if we checked the **Manually enter data source parameter values** radio button, we can manually define the parameters that will configure the JDBC connection used to fill the OLAP cubes. In the **Import Analysis** dialog box, we have a **Parameters** table that shows the parameter keys and values we already specified for this analysis data source.

**Import Analysis**

Mondrian File:  
 Delete

Select from available data sources. Add Parameter  
 Manually enter data source parameter values.

Parameters Edit Parameter

Name	Value
DataSource	SampleData
EnableXmla	false

To manage the parameters in this table, we must use the toolbar on the left-hand side of the **Parameters** table header as shown in the previous screenshot. Let's describe how we can add a new parameter that represents a new property for the connection, which we are going to configure:

1. Click on the **Add** parameter icon.
2. An empty **Parameters** dialog box opens up.
3. Fill the **Name** and **Value** fields with the name and the value of the parameter, respectively.
4. Each of the fields are mandatory. As soon as we fill each field, the **OK** button will become enabled.
5. At any moment, we can leave the dialog box without confirming the new parameter we are going to insert by clicking on the **Close** button.
6. Click on the **OK** button to close the dialog box and confirm the parameter value we just inserted.

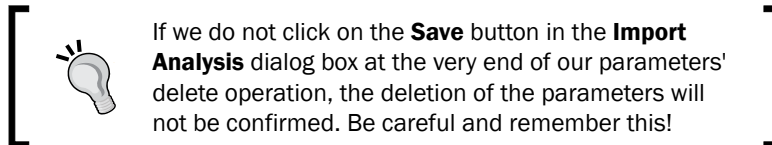
Similarly, we can update an existing parameter by performing the following instructions:

1. Select the parameter to be updated from the **Parameters** table, and click on the **Edit** parameter icon.
2. The **Parameters** dialog box opens up with the **Name** and **Value** fields filled with their respective actual values. Update the fields according to our needs.

3. At any moment, we can leave the dialog box without confirming the new parameter we are going to insert by clicking on the **Close** button.
4. Click on the **OK** button to close the dialog box and confirm the parameter value inserted.

Lastly, the deletion of a parameter that is not needed can be done using the following points:

1. Select the parameter we are going to delete from the **Parameters** list and click on the **Delete** icon.
2. The parameter is immediately removed from the **Parameters** list. Remember that the deletion will be confirmed only if we click on the **Save** button in the **Import Analysis** dialog box at the very end of the analysis data source configuration session.
3. Repeat this operation for any other parameter we want to delete.
4. Click on the **Save** button in the **Import Analysis** dialog box to confirm the delete operation.



### See also

If we want to update an existing analysis data source, take a look at the *Updating an existing analysis data source* recipe. If we want to delete it, take a look at the *Deleting an existing data source* recipe.

## Updating an existing analysis data source

We have seen how easy it is to define a new analysis data source, but let's briefly talk about how to update it. This is the topic of this recipe.

### How to do it...

To update an existing analysis data source, perform the following steps:

1. From the **Home** perspective, click on the **Manage Data Source** button.
2. The **Data Sources** dialog box appears.
3. An alternative way to open the **Data Sources** dialog box is from the main menu by navigating to **File | Manage Data Sources**.
4. From the data sources list, select the data source we want to update by clicking on the data source item.

5. Click on the **Edit Data Source** icon.
6. The **Import Analysis** dialog box opens and displays the properties of the data source we are going to update. Take the time to make the necessary changes to update the data source. Click on the **Save** button to confirm the updated data source configuration.
7. The **Import Analysis** dialog box closes, and the **Data Sources** dialog box will show again. Now, the data sources list is updated with the data source we just added.
8. Click on the **Close** button in the **Data Sources** dialog box to close it and terminate the data source definition process.

### How it works...

The process to update an existing analysis data source is the same as the process to create a new one. After selecting the data source from the data sources list, we can edit it by clicking on the **Edit data source** button located in the toolbar on the right-hand side corner of the data sources table list. We can change any configuration item data and update anything we need for the selected data source.



If we want to change the Mondrian schema that refers to an analysis data source, we cannot just update the schema file that is related to the data source. We need to delete the data source reference with the wrong schema file and create a new one.

### See also

If we are interested in creating an existing analysis data source, take a look at the *Creating a new analysis data source* recipe. If we want to delete an existing analysis data source, take a look at the *Deleting an existing data source* recipe.

## Creating a new metadata data source

The last interesting data source type is the metadata data source. Metadata is a valuable tool in our analytic solution. It is a way to abstract the final user from the physical structure of our data model while maintaining a direct and controlled access to the underlying data model. A metadata data model is implemented through an XML metadata file that describes the model, its properties, and the business rules that govern the model. This file is the resource required to define that kind of data source. This recipe will show you how to use a metadata model XML file to define a data source, called a metadata data source, that lets our reports and dashboards get data from a metadata model.

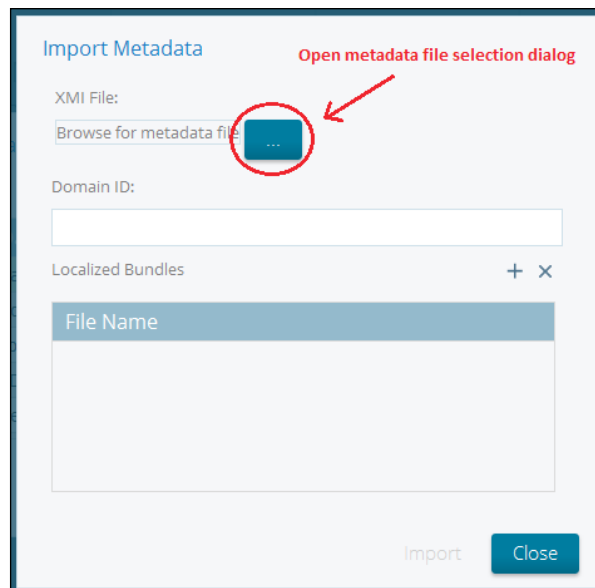
## How to do it...

To define a metadata data source, perform the following steps:

1. From the **Home** perspective, click on the **Manage Data Sources** button.
2. The **Data Sources** dialog box appears.
3. An alternative way to open the **Data Sources** dialog box is from the main menu by navigating to **File | Manage Data Sources**.
4. The **Data Source Types** combobox drops down and shows the various types of data sources we can define. From the list, select **Metadata**.
5. The **Import Metadata** dialog box opens. Fill all of the required information and at the very end, confirm the new data source configuration. Click on the **Save** button. To close the dialog box and lose any configuration item we defined, click on the **Close** button.
6. The **Data Sources** dialog box will show again, and the data sources list is updated with the data source we just added.
7. Click on the **Close** button in the **Data Sources** dialog box to close it and terminate the Data Source definition process.

## How it works...

The definition of this kind of a data source is strictly related to the availability of a metadata domain's file definition.



---

That said, as a first thing in our metadata data source definition, we need to select the XML file that contains the metadata model we want to publish as a metadata data source. By clicking on the button that can be identified by three dots on the right-hand side of the **XMI File** input textbox, we can open the **File Upload** dialog box. We can look in the filesystem directories for the file containing the metadata model we want to import.

As soon as we find the metadata file to import, we start the import operation published on our BA Server by clicking on the **Open** button. The file will be immediately uploaded to the BA Server.

Type the **Domain ID** of the model in the related input field; remember to use the domain name we defined while developing the metadata model. If necessary, add any localization bundle to the **Localized Bundles** table. The new data source is immediately visible to Pentaho BA Server without the need to restart the application and will be called using the schema name we imported into Pentaho BA Server.

### There's more...

Strangely, at the time of writing this book, there is no possibility to edit any defined **Metadata Data Source**. We do not know if this is a bug in the platform or if it is made this way, but at the time of writing the book, the only chance we have to update an existing metadata data source is by deleting the data source and creating a brand new one with the same name. Now, let's show you how to manage localized bundles in the metadata data source configuration.

### Managing localized bundles

One of the interesting things regarding metadata is that any label shown to the user can be internationalized in the final user's language. This is defined by referencing, for example, on any metadata attribute or on any business view name a set of message keys. Therefore, by having all of this work, whenever we are going to define a new Metadata data source, we also need to think about the definition of the required bundles to have anything work properly for all of the languages that are identified as a possible target.

### See also

If we want to delete an existing metadata data source, take a look at the *Deleting an existing data source* recipe.



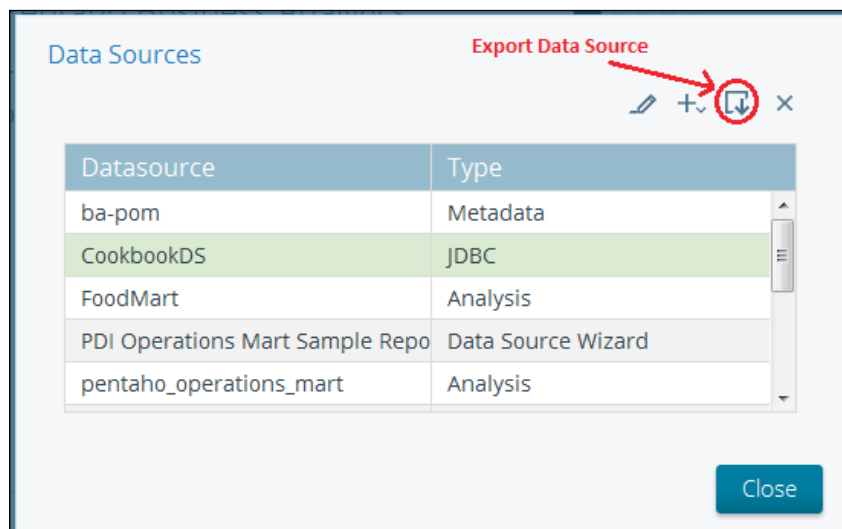
## Exporting an existing data source

The **Data Sources** dialog box also gives you the possibility to export the model definition data sources such as analysis and metadata. This recipe shows you how to do that.

### How to do it...

The following steps detail how to easily export an existing **Data Source**.

1. From the **Home** perspective, click on the **Manage Data Source** button; the **Data Sources** dialog box appears.
2. An alternative way to open the **Data Sources** dialog box is from the main menu by navigating to **File | Manage Data Sources**.
3. Click on the **Export Data Source** button, as shown in the following screenshot:



4. The **Save** file dialog box appears and asks you to locate a place on the disk where you want to save the data source descriptor. Click on the **Save** button to save the file and close the **Save as** dialog box.
5. Click on the **Close** button to close the **Data Sources** dialog box.

### How it works...

The export functionality works only for metadata and analysis data source types. By exporting the data source from Pentaho, we can export the schema file:

- ▶ In case we export a **Metadata** data source, Pentaho returns the .xml file used to define the data source.
- ▶ In case we export an **Analysis** data source, Pentaho returns the .xml Mondrian file that is the schema descriptor for Mondrian. We will briefly introduce what we intend to do with the Mondrian XML schema files and how to configure a Mondrian schema file later in *Chapter 6, Creating Analysis Reports*.

## Creating a new data source from a CSV file using the wizard

A useful feature we can get through in the Pentaho BA Platform is the ability to define a data source as the output of a query. This can be achieved as a result of the CSV load operation or, at the very end, as the result of accessing a plain table. Let's show you how to do this by using a CSV file as the input for our virtual data mart, accessed through this interesting kind of new data source.

### How to do it...

The following steps detail how we can use the wizard to define a new **Data Source**, starting from a CSV file:

1. From the **Home** perspective, click on the **Manage Data Source** button; the **Data Sources** dialog box appears.
2. An alternative way to open the **Data Sources** dialog box is from the main menu by navigating to **File | Manage Data Sources**.
3. The **Data Source Types** combobox drops down and shows the various types of data sources we can define. From the list, select **Data Source Wizard**.
4. The **Data Source Wizard** dialog box opens. Follow all of the steps of the wizard to get through all the phases of the definition of the new data source based on the CSV file.
5. At the end of the wizard, click on the **Close** button and close the **Data Sources** dialog box.
6. The new data source we just created will be immediately visible.

### How it works...

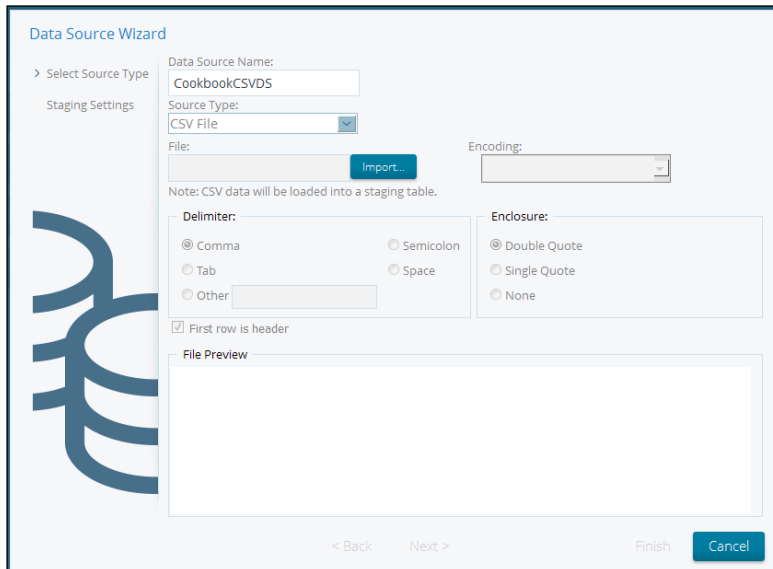
**Data Source Wizard** is a very quick and easy way to define a data source on the fly, starting from **CSV files**, **SQL Queries**, and existing **Database Tables**. We decided to illustrate this functionality by using the CSV file type because we consider this option to be more interesting for the common user. This way, the user has the ability to upload a file and analyze its data by leveraging the power of the analytical tools of Pentaho. We can easily access the data source and use it in a dashboard in an Interactive report or in an Analyzer report.

As soon as the wizard starts, it is mandatory to fill in the **Data Source Name** field. Then, as shown in the following screenshot, from the **Source Type** drop-down list, select the source type for the data we are going to use in order to feed the data source we are creating. The possible values are **CSV File**, **SQL Query**, and **Database Table(s)**. Select **CSV File**.



The screenshot shows the 'Data Source Wizard' window at the 'Select Source Type' step. The 'Data Source Name' field contains 'CookbookCSVDS'. The 'Source Type' dropdown menu is open, showing options: 'Select Source Type', 'CSV File', 'SQL Query', and 'Database Table(s)'. The 'CSV File' option is highlighted. At the bottom, there are '< Back', 'Next >', 'Finish', and 'Cancel' buttons.

Click on the **Next** button to get to the next step in the wizard. At any moment, we can click on the **Cancel** button to exit from **Data Source Wizard** and leave the things unchanged.



The screenshot shows the 'Data Source Wizard' window at the 'Staging Settings' step. The 'Data Source Name' field contains 'CookbookCSVDS'. The 'Source Type' dropdown menu is set to 'CSV File'. The 'File' field is empty with an 'Import...' button. The 'Encoding' dropdown menu is open. Below, there are radio buttons for 'Delimiter' (Comma, Tab, Other) and 'Enclosure' (Double Quote, Single Quote, None). The 'First row is header' checkbox is checked. At the bottom, there are '< Back', 'Next >', 'Finish', and 'Cancel' buttons.

As shown in the preceding screenshot, the new wizard's dialog box lets us import the CSV file containing the data feed for the data source, set the **Delimiter** option, and set the **Enclosure** and **Encoding** options of the file. To load the data source feed, click on the **Import...** button and select the CSV file we want to analyze. Then, set the parameters as defined in the following points:

1. Select the file encoding by selecting an item from the **Encoding** drop-down list.
2. Define the field delimiter character by selecting a radio button in the **Delimiter** field group. We can choose between **Comma**, **Semicolon**, **Tab Space**, or **Other**. In case we select **Other**, we can manually specify the right separator by typing it in the field to the right.
3. Check the value of the **First row is header** flag based on the file content (whether it has a header or not).
4. If required, we specify the enclosure of the strings by selecting one of the radio button options: **Double Quote**, **Single Quote**, or **None**.

We can always see a preview of the file in the **File Preview** text area; this is useful to check whether we are correctly declaring the characteristics of the feed's file. If the preview is fine, click on the **Next** button.

The new wizard's dialog box gives you the ability to verify and change some structural characteristics of the temporary table that the system is going to define inside Pentaho to contain the data feed.

**Data Source Wizard**

Select Source Type

> Staging Settings

Select the columns you want to stage on the Pentaho BA Server and configure their attributes. Use Source Format to specify how numeric or date fields are formatted in the source CSV file.

	Name	Type	Source Format	Length	Precision
<input checked="" type="checkbox"/>	CUSTOMERNUM	NUMERIC	#	0	0
<input checked="" type="checkbox"/>	CUSTOMERNAME	STRING		42	0
<input checked="" type="checkbox"/>	CONTACTLASTNAME	STRING		22	0
<input checked="" type="checkbox"/>	CONTACTFIRSTNAME	STRING		13	0
<input checked="" type="checkbox"/>	PHONE	STRING		25	0
<input checked="" type="checkbox"/>	ADDRESSLINE1	STRING		60	0
<input checked="" type="checkbox"/>	ADDRESSLINE2	STRING		13	0
<input checked="" type="checkbox"/>	CITY	STRING		19	0
<input checked="" type="checkbox"/>	STATE	STRING		12	0
<input checked="" type="checkbox"/>	POSTALCODE	STRING		12	0
<input checked="" type="checkbox"/>	COUNTRY	STRING		13	0
<input checked="" type="checkbox"/>	SALESREPEMPLID	NUMERIC	#	0	0
<input checked="" type="checkbox"/>	CREDITLIMIT	NUMERIC	#	0	0

Select All   Deselect All   Show File Contents

< Back   Next >   Finish   Cancel

After we have verified and fixed anything (if needed), click on the **Finish** button. Pentaho creates the table inside its own database and starts to load the feed data in it. A success message informs us about the success of the data source definition.

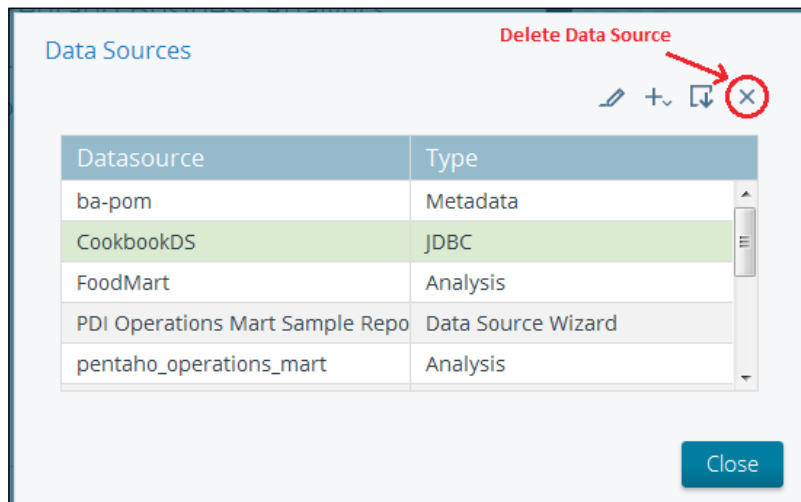
## Deleting an existing data source

Let's see how we can delete an existing data source from the list of available data sources.

### How to do it...

To delete an existing data source, perform the following steps:

1. From the **Home** perspective, click on the **Manage Data Source** button; the **Data Sources** dialog box appears.
2. An alternative way to open the **Data Sources** dialog box is from the main menu by navigating to **File | Manage Data Sources**.
3. Select the **Data Source** we want to delete and click on the **Delete Data Source** icon, as shown in the following screenshot:



4. The **Remove Data Source** warning dialog box appears and asks whether you are sure that you want to delete the selected data source. Click on the **Remove** button to remove the data source and close the **Remove Data Source** dialog box. The data source is removed from the list of available data sources.

5. Click on the **Close** button to close the **Remove Data Source** dialog box without deleting the selected data source. The data source is removed from the list of available data sources.
6. Click on the **Close** button in the **Data Sources** dialog box to close the **Data Sources** dialog box and return to the Pentaho User Console.

### How it works...

When we click on the **Remove** icon, the data source is immediately removed by the system. This is a critical operation. Remember to check, before proceeding with the deletion, that the data source is not used by any of the BI objects (reports, dashboards, OLAP views, or others) published under the Pentaho solution.



If you are not sure whether the **Data Source** we are going to remove is unused, please do not hesitate to contact the key users and check. If we delete a data source that is in use, we break the functionality of all of the BI objects that are fed by this data source. So, be careful!



# 4

## **Defining Business Models with the Pentaho Metadata Editor**

In this chapter, we will cover the following topics:

- ▶ Using a JNDI connection for development
- ▶ Managing JDBC database connections
- ▶ Defining the physical layer
- ▶ Defining concepts
- ▶ Reviewing physical layer tables' columns
- ▶ Deriving business models from the physical layer
- ▶ Reviewing business tables' column properties
- ▶ Applying formatting properties to business tables' fields
- ▶ Adding new calculated columns to model entities
- ▶ Defining joins between business tables' entities
- ▶ Creating business view categories
- ▶ Testing metadata layer results
- ▶ Applying security to domain model elements
- ▶ Publishing metadata definitions to the BA server



## Introduction

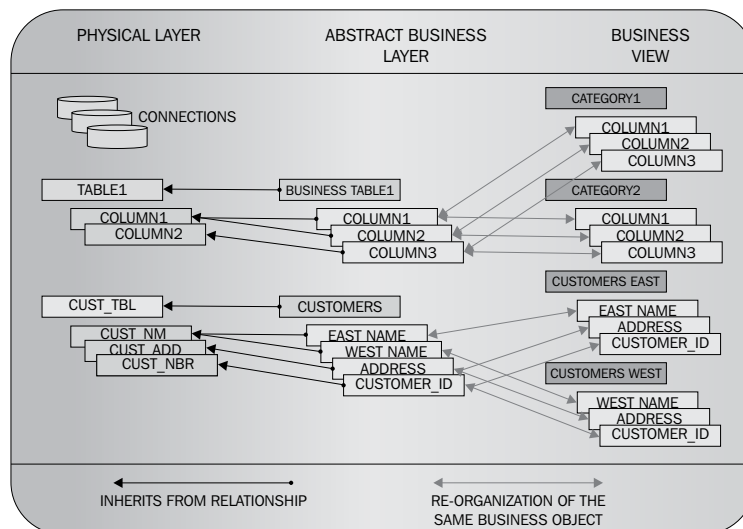
This chapter looks at the Pentaho Metadata Editor, a developer tool, so we can build a set of metadata model domain objects that could be useful for our reporting needs.

The metadata model domain is made up of a set of business domains and models that abstract users from the physical aspects of the data warehouse model. The model is saved on a server in a centralized location for better accessibility.

The following points illustrate the main benefit gained by implementing a metadata model domain:

- ▶ It gives the user a better understanding of the business model by explaining concepts in a language that is more related to the business, so it is more understandable.
- ▶ Because a metadata model is abstracted from a physical database model, it lowers the impact of any changes in the physical database structure.
- ▶ The metadata model domain gives you the ability to define security information through **access control lists (ACLs)** to its objects and attributes by granting finer access control and security at any level in the model.
- ▶ The metadata model adds formatting options to the values extracted from the business model entities. We can unify the management of formatting options for strings, dates, numbers, and other data types in a fully consistent way and in a centralized place, without the need to take care of these aspects at the report's design time.
- ▶ This model adds localization to the information displayed based on the user's regional settings.

The following figure shows the business objects and their relationships in the metadata model domain:



From the preceding figure, we can see that the metadata domain is made up of three layers: the physical layer (the database underneath), the abstract business layer, and the business view layer.

The abstract business layer abstracts the physical layer through a set of objects that are inherited from the physical database tables. Users, by enriching the existing attributes or by adding more attributes (we will see how we can do this in the recipes that follow), can enhance the business model objects in the abstract business layer. The business model objects are then exposed to the user for querying through an organized set of categories in the business view layer. The good thing about this is that security rules can be applied at any of the three levels in the metadata model domain by reaching an unrivaled level of access control over the information exposed through the model.

The Pentaho Metadata Editor is a client development tool. There are two versions of the Pentaho Metadata Editor:

- ▶ In the Pentaho EE version, the tool is already present in the Pentaho distribution, and we can find it under the `<pentaho_ee_home>/design-tools/metadata editor` directory
- ▶ In the Pentaho CE version, it must be downloaded separately from <http://community.pentaho.com/> and installed on our system

Once the Pentaho Metadata Editor is installed, we need to check whether our Java environment is configured properly. The recipes in this chapter are based on the assumption that we have checked whether the `JAVA_HOME` environment variable is set appropriately. Even if the Metadata Editor tries to guess the value of the `JAVA_HOME` environment variable from the system while starting, it is always a good rule of thumb to set the `JAVA_HOME` environment variable in order to be sure that everything works as required.



Setting the `JAVA_HOME` variable properly and referring all of the Java paths to this environment variable is a good way to keep your system clean and gives you the ability to manage more than one version of the JDK at the same time on the same system.

## Using a JNDI connection for development

Let's show you how to configure a development JNDI data source on your development client. We will use this definition later on in the *Managing JDBC database connections* recipe to configure a JDBC connection inside the Pentaho Metadata Editor.

### Getting ready

For this recipe, check the following conditions:

1. Go to the Pentaho Metadata Editor directory and once there, go to the `simple-jndi` directory.
2. Be sure the Pentaho Metadata Editor application is closed. If not, please close it before you start working on this recipe.

### How to do it...

The following recipe shows you how to configure a JNDI connection to be used for development with the Pentaho Metadata Editor:

1. Open the `jdbc.properties` file with your favorite editor. The `jdbc.properties` file contains a set of definitions for any of the development connections defined locally on our Pentaho Metadata Editor instance. By default, the file is filled with a set of demo connections that map to the standard Pentaho demo database.
2. Take one of the sample definitions, such as the following excerpt:

```
SampleData/type=javax.sql.DataSource
SampleData/driver=org.hsqldb.jdbcDriver
SampleData/url=jdbc:hsqldb:file:sampladata/
sampladata;ifexists=true
SampleData/user=pentaho_userSampleData/password=password
```

Copy and paste it at the very end of the file. We assume that you are already familiar with the details about what a property file is and how a property file is defined. The key part of a property line definition is made up of two parts separated by the `/` character. The part on the left represents the name of the connection we are going to create. The part on the right side of the key identifies an attribute in the data source definition configuration for the same connection. In our case, we cloned the definition of a connection named `SampleData`.

3. Modify the name of the connection in the copied excerpt of the properties file from `SampleData` to `foodmart_mondrian`. Our properties set now looks like the following code:

```
foodmart_mondrian/type=javax.sql.DataSource
```

```

foodmart_mondrian/driver=org.hsqldb.jdbcDriver
foodmart_mondrian /url=jdbc:hsqldb:file:sampledatab/
sampledata;ifexists=true
foodmart_mondrian/user=pentaho_user
foodmart_mondrian/password=password

```

- Let's go through various connection's attributes one at a time. Remember, by connection attribute, we mean the name on the right side of the / character in the key name. The `type` attribute is a default, and its value must never be changed for any reason.
- To configure a connection named `foodmart_mondrian` that connects to the `Foodmart` database on MySQL using the user, `foodmart_user`, with a password as `password`, the previous set of sample property values must be changed in the following way:

```

foodmart_mondrian/type=javax.sql.DataSource
foodmart_mondrian /driver=com.mysql.jdbc.Driver
foodmart_mondrian /url=jdbc:mysql://<hostname>:<port>/foodmart_
mondrian
foodmart_mondrian /user=foodmart_user
foodmart_mondrian /password=password

```

Here, `<hostname>` and `<port>` are our database's hostname and port.

- Save the file and close the editor.

## How it works...


We already explained the idea of using data sources in Pentaho objects to access the data in our BI system database. It is a safer way to abstract all of our BI objects from database connection details and mitigate the risk of a change in the databases' configuration in our BI objects.




If we configure a Pentaho data source and use this data source name in all of our Pentaho BA objects, any change in any of the database connection parameters will remain contained in the data source configuration. When we change the Pentaho data source configuration, the change will immediately affect all our BI objects.

However, there could be a problem during the development phase using this approach. A Pentaho data source is a J2EE server-side concept and resides on the server. This means that the development tool cannot directly access the required data source because it is only shared with any application deployed to the Pentaho BA Server instance where this particular data source was created.


Therefore, for an easier development approach, we can use a development JNDI connection to emulate the Pentaho data source that we will find on the Pentaho BA server once the metadata object is deployed.

 Development JNDI connection is a term that we use to identify the specific execution method applied to all of the Pentaho development tools.

While declaring our new JNDI connection, we use the definitions contained in a property file in the `simple-jndi` directory located under the Pentaho Metadata Editor directory. This file contains all of the connection attributes we must configure to connect to our database. To configure a new connection, copy and paste one of the connection definitions that are already in the file and change it according to your needs.

 The development JNDI connection is created by the Pentaho Metadata Editor using `simple-jndi`, a library to create a simple JNDI objects' registry to be used by our application. It is an easy way to enable JNDI registry features in any non-J2EE server applications. This library can be used by any Pentaho development tool.

- ▶ As soon as we need a reference to a JNDI connection in our metadata domain model, we type the name of that connection in the JDBC JNDI data access connection
- ▶ When we test the metadata domain model, the JNDI name we set in the JDBC JNDI data access connection configuration is available through `simple-jndi`, and our object works better by referencing the JNDI name as it will be at deployment time

 Keep in mind that to have everything port smoothly to our server, the JNDI connection name we set in the JDBC JNDI data access connection must be the same as the name of the JDBC JNDI data source we defined on the server. As we explained in *Chapter 3, Defining BA Server Data Sources*, a JDBC JNDI data source, created from the **Manage Data Source** wizard in Pentaho User Console, is a real JNDI connection to a relational database.

## See also

- ▶ <http://code.google.com/p/osjava/wiki/SimpleJNDI> for more in-depth details about the `simple-jndi` library and how it works

## Managing JDBC database connections

To create a metadata model, we start from the data warehouse physical model. For this reason, the Pentaho Metadata Editor first needs an available JDBC connection to look into the data warehouse and extract its physical model structures. This recipe shows you how to define a JDBC connection in order to connect to your database.

### Getting ready

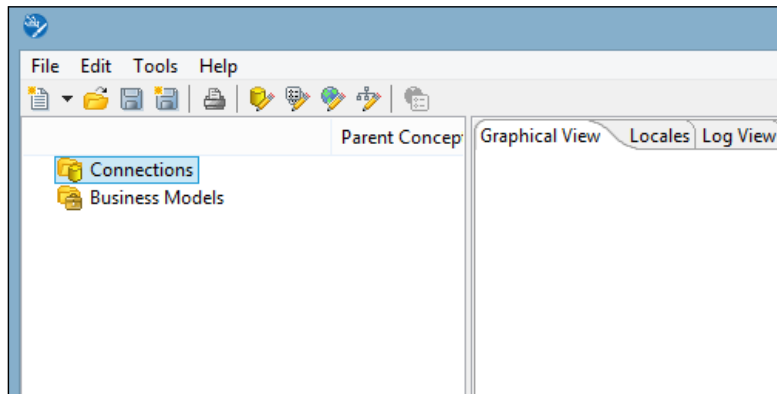
For this recipe, check the following conditions:

- ▶ The Pentaho Metadata Editor has been started
- ▶ A metadata domain model is created and available

### How to do it...

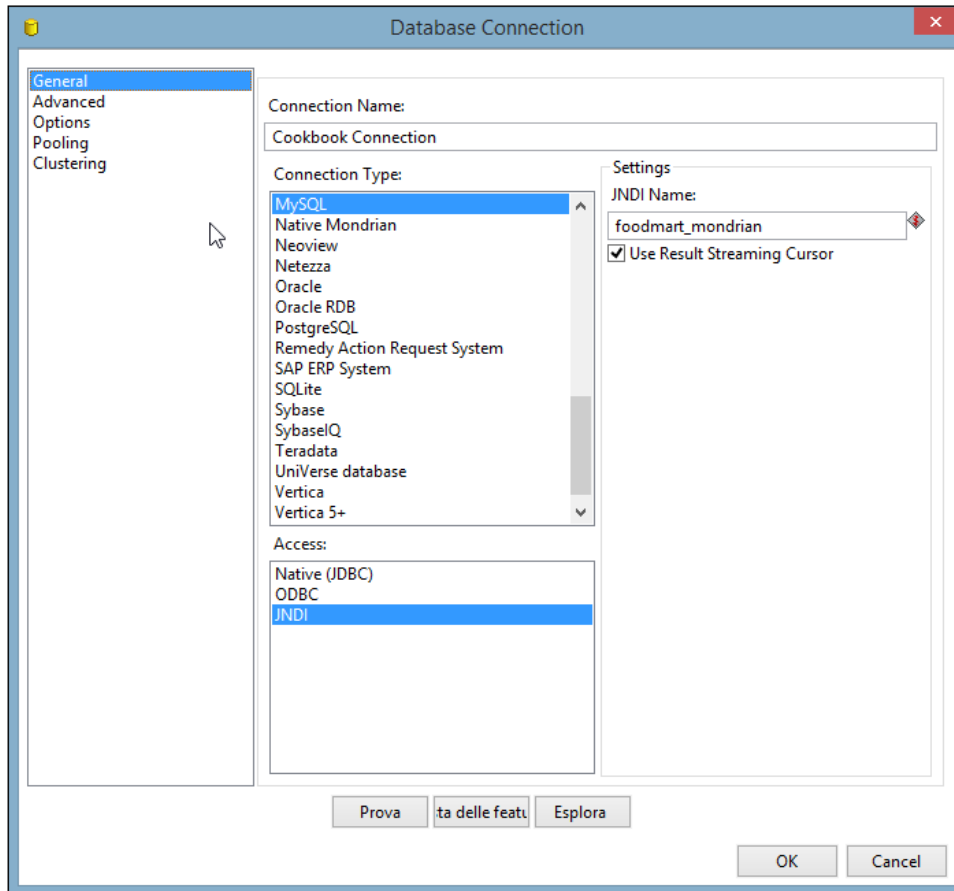
To define a JDBC connection to be used in our metadata model, we need to perform the following steps. We are going to define a JDBC connection that makes use of the JNDI connection we defined in the preceding recipe:

1. Click on the **Connections** folder from the view on the left-hand side of the working area, as shown in the following screenshot:



2. Right-click on the **Connections** folder. The context menu will appear, then select the **New Connection** item entry.
3. The **Database Connection** dialog box will open.
4. Type the name of the connection we are going to create in the **Connection Name** field. We will call this connection `Cookbook Connection`.

5. Select **MySQL** from the **Connection Type** field, **JNDI** as the access type from the **Access** field, and then set `foodmart_mondrian` as **JNDI Name** (the name of the JNDI connection we configured in the last recipe, *Using a JNDI connection for development*) as shown in the following screenshot:



Remember to check whether the appropriate JDBC driver is present in the JDBC driver's folder before running the Metadata Editor the first time you connect to a particular database. To see how to do this, look at the *Adding a JDBC driver* section under the *There's more* section at the end of this recipe.

6. Click on the **Test** button to verify that the connection is working properly.

7. If everything works properly, click on the **OK** button to confirm the configuration information and close the **Database Connection** dialog box.
8. Now, on expanding the **Connections** folder, the new connection is immediately visible as a child of this folder.

### How it works...

In this recipe, we illustrated how to create a JDBC JNDI database connection. This is the best way to define a database connection to lower the impact of changes in the database configuration parameters.

After we open the **Database Connection** dialog box, from the **Connection Type** list, select the database type we are going to connect to. In our case, we select **MySQL**. Under **Settings**, on the right-hand side of the dialog box in the **JNDI Name** field, type the name of the JNDI connection you want to use for this Pentaho Metadata Editor JDBC connection. In our case, type `foodmart_mondrian`, which is the connection name we configured in the previous recipe, *Using a JNDI connection for development*.

Click on the **Test** button to verify whether the connection is configured properly. If the connection test was successful, click on the **OK** button to confirm the new data source definition and close the **Database Connection** dialog. If the connection test was not successful, check the errors and click on the **Test** button again.

As soon as we close the **Database Connection** dialog box, the new connection is immediately visible as an immediate child of the **Connections** folder.

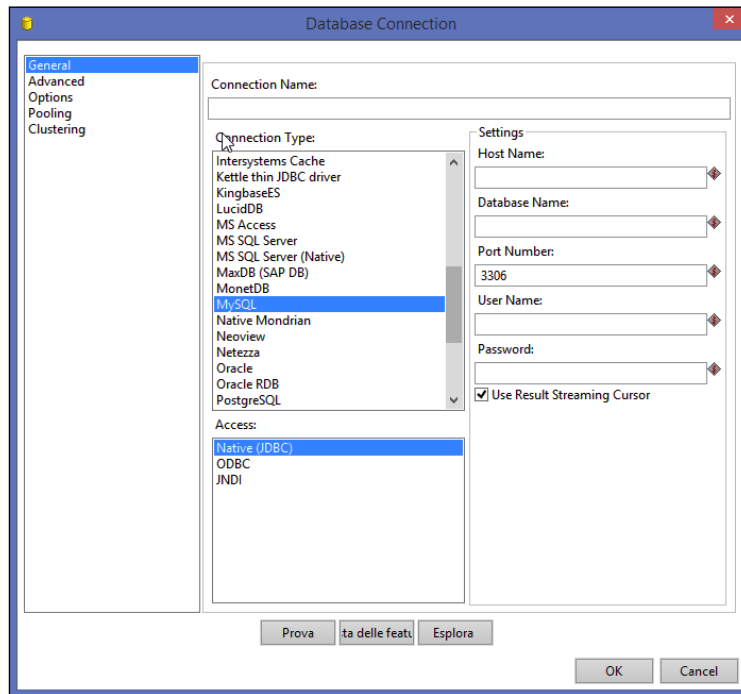


As soon as you complete a set of configurations on your metadata domain file, always remember to save the domain file in order to make all the changes persistent; otherwise, you may lose something.

Another way to define a database connection is when all of the database configuration parameters are specified directly in the connection definition. This is the easiest way to define a JDBC connection, but it is not the best way. In fact, if one or more of the database connection's configuration changes, we must update our connection's configuration in the metadata domain file too. This type of database connection is defined as a JDBC native.



Open the **Database Connection** dialog, name the connection we are creating, and select **Native** in the **Access** listbox, as shown in the following screenshot:



Under the **Settings** field's group of the **Database Connection** dialog, we have the following database configuration properties' fields:

- ▶ In the **Host Name** field, type the IP address or DNS name of the host where the database resides. This field is mandatory.
- ▶ In the **Database Name** field, type the name of the database we are going to connect to. In our case, the name of the database is `foodmart_mondrian`. This field is mandatory.
- ▶ In the **Port** field, type the port number at which the database listens for connections from clients. This field is mandatory.
- ▶ In the **User Name** field, type the database username to be used during the connection. This field is mandatory.
- ▶ In the **Password** field, type the username's password to be used during the connection to the database. This field is mandatory.

Click on the **Test** button to verify if the connection is configured properly. If the connection test was successful, click on the **OK** button to confirm the new data source definition and close the **Database Connection** dialog. If the connection test was not successful, check the errors and click on the **Test** button again.

## There's more...

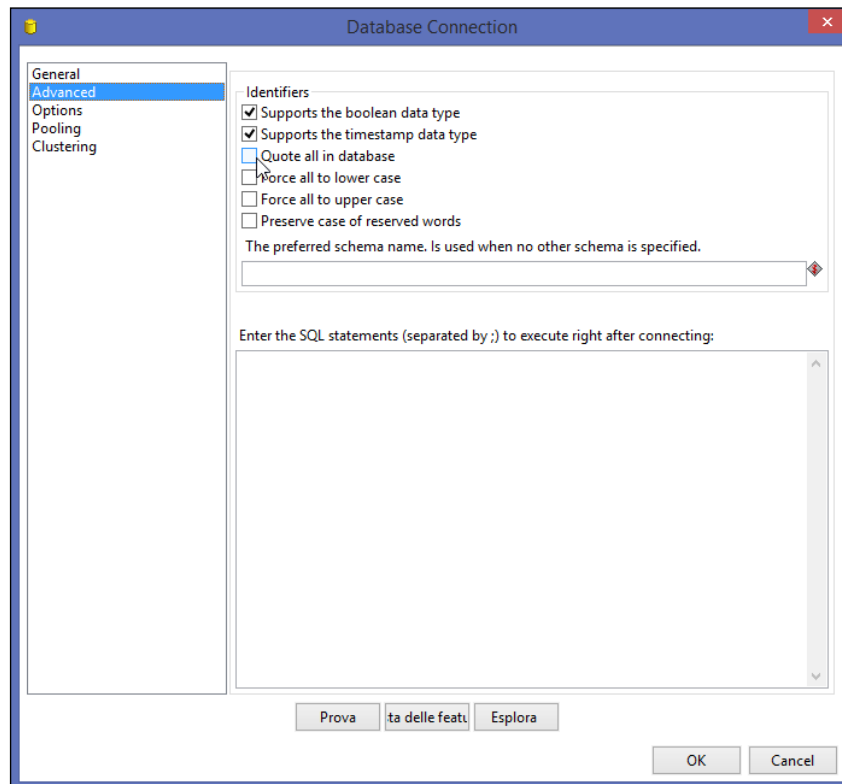
It is interesting to know where we can put JDBC drivers for use with the Pentaho Metadata Editor. Then, it is interesting to look at how we can further specify properties for the connection configuration. The next two paragraphs will give a brief illustration about all these things.

### Adding a JDBC Driver

Even if the list of available databases is long, not all the databases contained in that list have a related driver delivered in the bundle with the tool and ready to be used. So, we need to manually deploy it in the right place. To deploy a JDBC driver for use with the Pentaho Metadata Editor, we must copy the JAR file to the `<pentaho_metadata_editor_home>/libext/JDBC` directory. After we have copied the driver, remember to restart the Pentaho Metadata Editor application.

### Using advanced configuration parameters

We can access some advanced JDBC configuration properties by selecting the **Advanced** menu entry from the left-hand side menu of the **Database Connection** dialog, as shown in the following screenshot:



When we select the **Advanced** menu entry, we have a set of fields that need to be configured as follows:

- ▶ **Supports the Boolean data type:** This must be set to `true` at any time we want values such as `Y` or `N` to be treated as Booleans. This field is checked by default.
- ▶ **Supports the timestamp data type:** This adds support to the timestamp data type. This field is checked by default.
- ▶ **Quote all in the database:** This forces all field and table names in the query that are sent to the database to be enclosed with the `"` character.
- ▶ **Force all to lowercase:** This forces all fields and table names in the query that are sent to the database to be written using lowercase character strings.
- ▶ **Force all to uppercase:** This forces all fields and table names in the query that are sent to the database to be written using uppercase characters strings.



Remember that these definitions will be confirmed as soon as we click on the **OK** button in the **Database Connection** dialog.

## Defining the physical layer

As soon as we have created our JDBC connection, we can define the physical layer for the Pentaho metadata model we are developing. The physical layer is the lowest layer of our metadata domain. It maps all of the physical database structure, and it is the starting point to develop our metadata business objects. In this recipe, we will learn how to automatically define the physical layer by starting from a JDBC connection.

### Getting ready

For this recipe, check the following conditions:

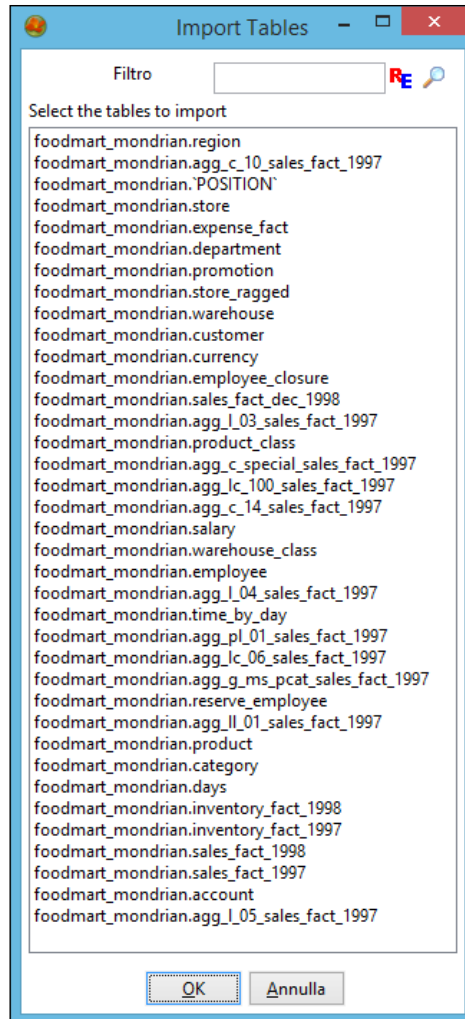
- ▶ The Pentaho Metadata Editor has been started
- ▶ A metadata domain model must be created and available
- ▶ A JDBC connection that we can use must be created

### How to do it...

To use the Import Tables feature from the database connection contextual menu, perform the following steps:

1. Expand the **Connections** folder and right-click on the configured connection's folder. Then, select the **Import Tables** menu entry.

2. The **Import Tables** dialog box appears as shown in following screenshot:

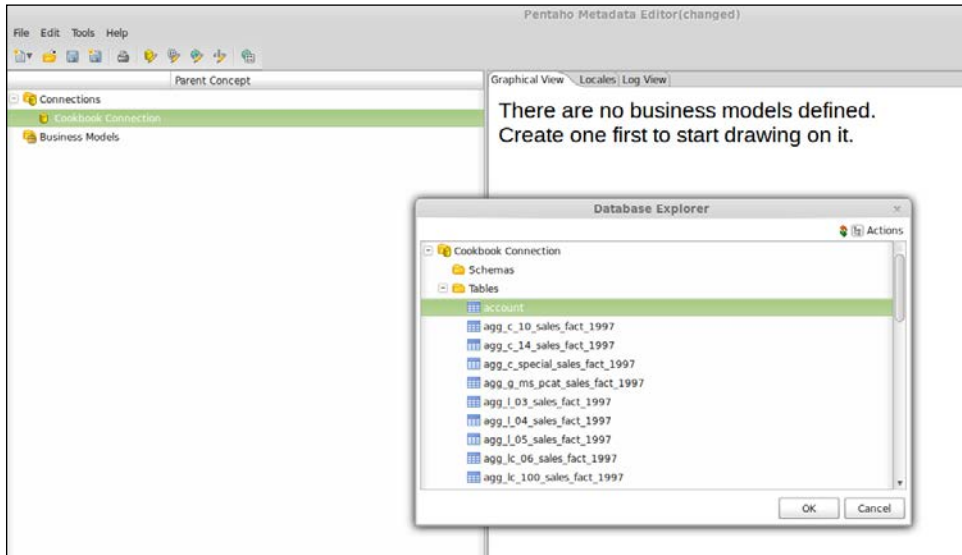


3. Select the tables we want to import and click on the **OK** button to close the **Import Tables** dialog and confirm the selection.
4. The tables that have been imported will appear as immediate children of the database connection we just configured (identified by its name).

To import the database tables from **Database Explorer**, perform the following steps:

1. Expand the **Connections** folder, right-click on the configured connection's folder, and select the **Import from Explorer** menu entry.

2. The **Database Explorer** dialog box appears, showing all the tables contained in the database, as shown in the following screenshot:



3. Expand the **Table** node. Its immediate children will be the database's tables.
4. Double-click on the table you want to import. The **Database Explorer** dialog closes and the tables will be displayed as the immediate children of the database connection we just configured (identified by its name).
5. To add other tables, repeat steps 1 to 4.

## How it works...

There are two ways to define the tables' structure of the physical layer.

The first, quicker way uses the Import Tables feature from the **Database Connection** contextual menu. From the view on the left-hand side of the working area, expand the **Connections** folder and right-click on the configured connection's folder. Select the **Import Tables** menu entry. The **Import Table** dialog box appears, showing us all the tables contained in the database.

We have the following two ways to add tables to the model:

- ▶ **Select all of the tables at once:** To do this, select the first table in the list; then go to the end of the list and by pressing the *Shift* key and the left mouse button, select the last one and all of the tables in the list will be selected.
- ▶ **Select the tables one by one:** Press the *Ctrl* key and by keeping this key pressed, one by one, select the tables we want to import by pressing the left mouse button. The selected tables will be highlighted in the list of tables.

Click on the **OK** button to close the **Import Tables** dialog box and confirm the selection. By clicking on the **Cancel** button, we can close the **Import Tables** dialog at any time without doing anything. As soon as we have clicked on the **OK** button, the tables selected will appear as immediate children of the database connection we just configured (identified by their name).

The second way to import database tables in the metadata model is by using the **Database Explorer** dialog box. The **Database Explorer** dialog box is a little tool we can find in all of the Pentaho development tools (whenever required) to explore the structure of the database we connected to. It allows us to see the set of tables contained in the database, get a sample of the records in the tables, get DDL information, truncate the tables, and other interesting features.

To open the **Database Explorer** dialog box at any time, expand the **Connections** folder and right-click on the configured connection folder. Select the **Import from Explorer** menu entry. The **Database Explorer** dialog box appears, showing all the tables contained in the database as a collapsed tree with the root named as the name of the database connection we just started in order to open the dialog.

Expand the root; as immediate children we have a set of folders representing the database's main structures: **Schemas**, **Tables**, **Views**, and **Synonyms**. By expanding the **Table** node, all the database's tables will be displayed. We can import one or more tables by double-clicking on them. Unfortunately, we have to do this one table at a time; also, when we double-click on a table to select it, the **Database Explorer** dialog closes. Therefore, to import more than one table, we have to reopen it and repeat everything we described here from the very beginning.



If we have to import a very large set of tables, we suggest that you follow the first way; using the **Import Tables** functionality gives you the possibility to select a set of tables so you can process all of them at once.

## See also

- ▶ If you are interested in understanding how to define a JDBC connection, a prerequisite to connect to your database, look at the *Managing JDBC database connections* recipe

## Defining concepts

The goal of a metadata model is to give the user an enriched view of the underlying data model. A feature in the Pentaho Metadata Editor is a reusable elementary expressive element identified by a set of properties with keys and values. It can define a set of formatting options, model descriptors, security constraints, calculations, or others to express basic types that can better specify your model tables or tables' attributes. Basic concepts can be chained in hierarchies to build sophisticated expressive structures.

As a general rule of thumb, we can consider the following approach when dealing with metadata object properties:

- ▶ Trying to unify the way concepts are expressed to our users is a very important goal. We can do this by defining appropriate metadata concepts with a targeted set of properties, particularly suited to specific concepts' categories. For example, we defined the concept of a number to be applied in any case that deals with numbers. This way, numbers are always represented in the same way all over the reports or other tools that make use of metadata. Any number is always aligned to the right and has a specific formatting mask that manages the appearance of the number and the number of decimals.
- ▶ After we apply metadata-wide concepts to any specific case, where required, we take into consideration specific cases by going to override by table's field, case by case, the settings inherited by the applied metadata-wide concept.

It is not that difficult to set up the metadata domain model in this way. It is really straightforward from the point of view of our users because we are going to construct a vocabulary of concepts that we can reuse in our model.

This recipe is all about creating concepts to use in the metadata domain model.

## Getting ready

For this recipe, check the following conditions:

1. The Pentaho Metadata Editor must be started.
2. A metadata domain model must be created and available.

## How to do it...

To define a new concept, perform the following steps:

1. Open the **Concept Editor** window by either accessing the **Concept Editor...** menu under **Tools** or clicking on the **Concept Editor** button from the Pentaho Metadata Editor toolbar.
2. Select the **Base** concept from the tree view and click on the add new concept button.
3. The **New Concept** dialog appears. Type the name of the new concept. Suppose we are going to express a number type as a concept, then type `Number` in the **Concept Name** field of the **New Concept** dialog.
4. Change the alignment options. In the **Settings** field, go to the **Text Alignment** section and change the **Alignment Type** field to **Right**.

5. If we want a number displayed in a specific format in our report, we must define a new format mask. Click on the add property button located in the right corner toolbar of the **Available** list.
6. The **Add New Property** dialog opens. From the list, add the **Mask for Number and Date** property and click on the **OK** button.
7. The **Add New Property** dialog closes, and the new property is added in the list of available properties. In the **Value** field, type the new formatting mask, # , ### . ##.
8. Press the **OK** button to close the **New Concept** dialog and confirm the changes. The new concept will be immediately visible in the **Concept Editor** hierarchy as a child of the **Base** concept.

### How it works...

To create a concept, we need to use the **Concept Editor** window. We can either start the **Concept Editor** window by going to **Tools | Concept Editor...** or by clicking on the **Concept Editor** button from the Pentaho Metadata Editor toolbar.

The **Concept Editor** dialog shows all of the concepts defined as a tree view. This is because we can define hierarchies of concepts to express, in the easiest way, what we are going to model in our metadata. As an example, we can define the concept of **Number** and then we can say that we have a concept of **Amount** that is a specification of the concept **Number** because it is a **Number** with a different formatting mask. Therefore, we can say that **Amount** inherits from **Number** and only overrides the formatting mask. We will look at how to do this in our example in this recipe.

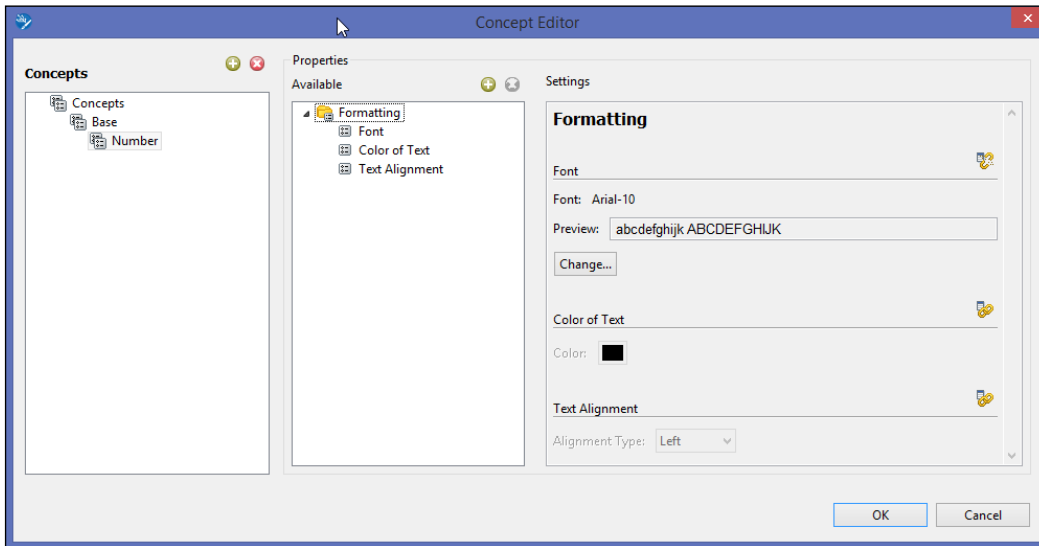
As soon as we start **Concept Editor**, there already exists, at a minimum, a concept called **Base**. The **Base** concept is the default concept and contains only formatting properties. We can use it as the base concept for all of our concepts.

Let's go through the creation of the **Number** concept. Select **Base** as the parent concept. We always need to specify a parent concept to define a new one because of the inheritance between different concepts. Now, from the toolbar located in the top-right corner of the **Concepts** tree view, click on the add new concept button (the icon with the + sign on it).

The **New Concept** dialog appears. Type `Number` in the **Concept Name** field of the **New Concept** dialog. This field is mandatory. Next, click on the **OK** button to confirm; the **New Concept** dialog closes and the concept we just defined appears as an immediate child of its parent.

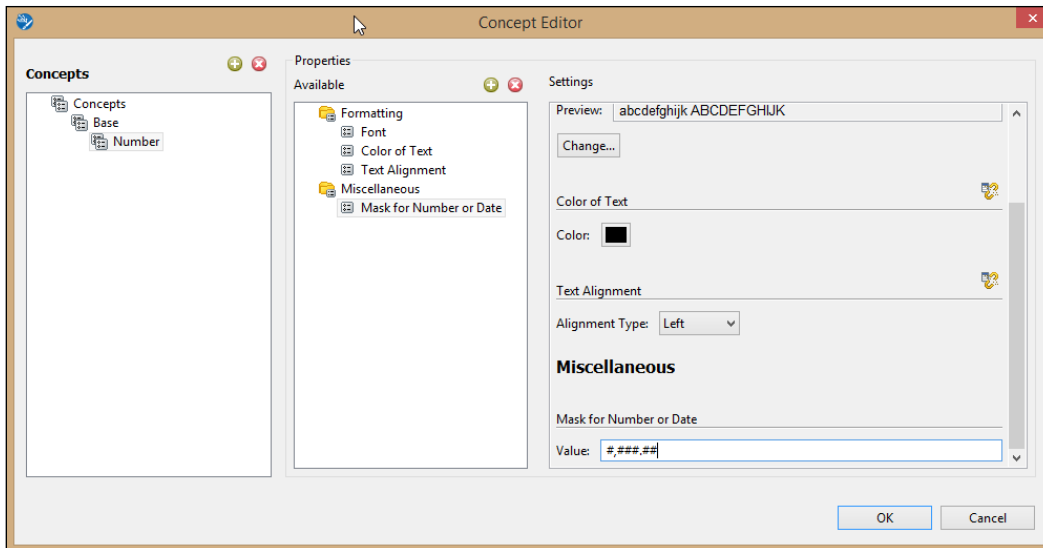


Now, we do need to specialize our concept to represent a number. In particular, we need to change the **Text Alignment** value from **Left** to **Right** and specify a formatting mask because of the differences in how, generally speaking, a number is represented in a report. The alignment options can be changed easily. In the **Settings** field, go to the **Text Alignment** section and change the **Alignment Type** value to **Right**.



To define the new formatting, click on the add property button located in the right-corner toolbar of the **Available** list. The **Add New Property** dialog opens. We need to add a new property called **Mask for Number and Date**; we can easily find it under the **Miscellaneous** category. Select the **Mask for Number and Date** property and click on the **OK** button.

As soon as the **Add New Property** dialog closes, the new property is added to the list of available properties for **Number**. Select the newly added property from the **Available** tree view list, and type the mask pattern in the **Value** field for the **Mask for Number and Date** property. As a number format pattern, we can set #, ###.##, but we can choose to set the pattern we need depending on our needs.



Click on the **OK** button to confirm the changes and close the **New Concept** dialog box.

Now, let's make a further refinement by defining an **Amount** concept that inherits from **Number**. To do this, from the **Concepts** tree view, select the **Number** concept as the parent concept and click on the add new concept button.

The **New Concept** dialog appears. Type `Amount` as the value of the **Concept Name** field in the **New Concept** dialog. Immediately after we click on the **OK** button, the **New Concept** dialog closes and the concept we just defined appears as an immediate child of the **Number** concept.

Select the **Mask for Number and Date** property from the **Available** tree view list. Because the value for the property was already set in the parent concept, the value for this property is defined as read-only. To set the new mask, we basically need to override this value by clicking on the **Override** button located to the right of the field value (the button with the chain icon on it). Now the field value turns to writeable; type the new mask pattern as `#,###.##`.

Click on the **OK** button to confirm the changes and close the **New Concept** dialog box. We can see the **Amount** concept as a child of **Number**, which is a child of **Base**. Finally, click on **OK** to close the **Concept Editor** dialog and return to the Pentaho Metadata Editor.

## See also

- ▶ If you are interested in moving forward and want to see how to apply concepts to better define your model, see the *Reviewing physical level tables' columns* recipe.

## Reviewing physical layer tables' columns

As soon as we get the physical model's structure defined, we can review some of the table properties. This recipe gives you an idea about how to edit the physical tables' properties.

### Getting ready

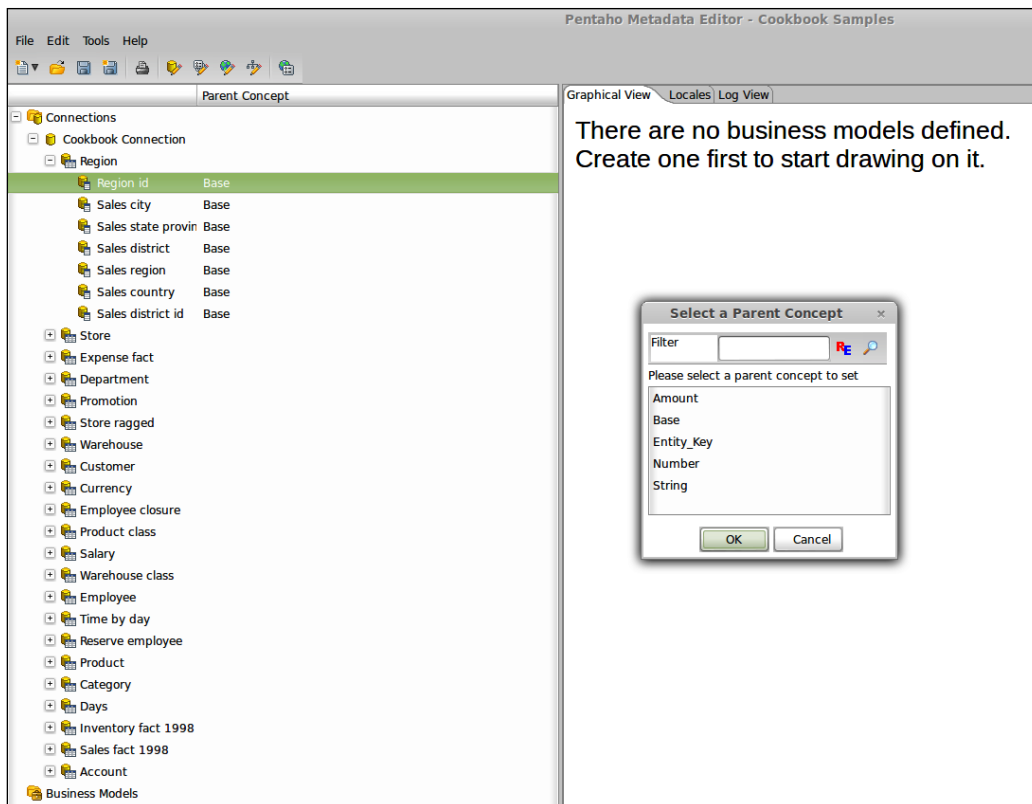
For this recipe, check the following conditions:

1. The Pentaho Metadata Editor must be started.
2. A database physical layer must already be imported and available.
3. Import a metadata domain model to start our recipe. To import a model from the Pentaho Metadata Editor menu, go to **File | Import from XMI file** and then go to the `<cookbook_samples>/ch4/resources` and load the file `cookbook-samples-model-start.xmi` directory. This file contains a predefined model with a physical layer already imported and a set of concepts already created and ready for use. As soon as the system asks for a domain name, type `Cookbook Samples` and click on **Yes** when asked if you want to overwrite the existing domain.

### How to do it...

First, we are going to show you how to assign concepts to columns in a model's entities. To do this, perform the following steps:

1. Expand the connection folder identified by the connection's name to access the list of tables for this connection.
2. Let's suppose that you want to edit the `Region` table in order to assign a concept to the `Region id` column. Right-click on the `Region id` column of the `Region` table and select the **Set Parent Concept...** menu entry.



3. Double-click on the **Entity\_Key** concept and assign it to the **Region id** attribute. This assignment clearly identifies that the **Region id** attribute has the concept of primary key. This concept has already been created in the sample model file we imported and is available for use.
4. As soon as we double-click on the concept, the **Set Parent Concept...** dialog closes.

As the second example, we want to review a set of columns' properties. To do this, let's perform the following:

1. Double-click on the table we want to check or update the columns' properties. The **Physical Table Properties** dialog will appear.
2. Choose the property we want to update by selecting it from the **Available** tree view. The properties inherited from the field's parent concept must be overridden in the same way as you did for concepts. To do this, click on the override icon and apply the required changes to the value in the field.
3. After we have made changes to the table's field properties in any order, confirm the changes, close the dialog, and then click on the **OK** button.

## How it works...

As soon as our physical model has been imported, we can find the set of the model's database tables imported as immediate children of our JDBC connection. To start showing you how to work on them, we can use two examples.

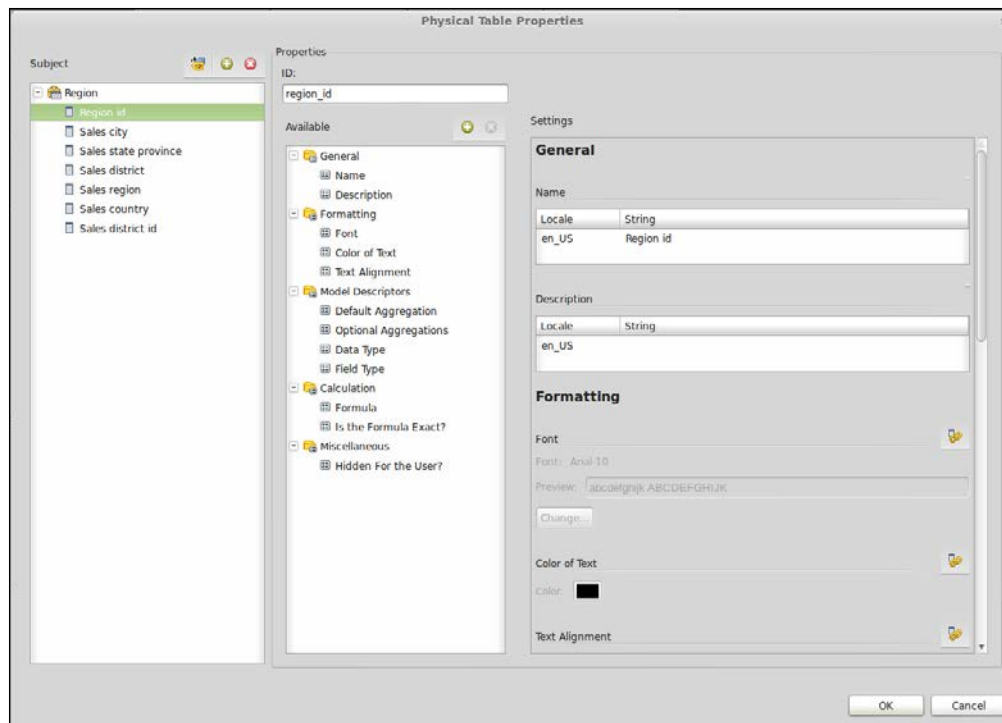
The first example is about assigning concepts to the fields in the model's entities. To do this, we first need to get to the tables' list from the database associated with the database connection we configured and choose the first table we want to work on. Suppose we start working on the `Region` table by assigning a concept to the `Region id` column. The idea is to use concepts to better specify what this field represents in the table; in this case, the field is a key for the `Region` table, so we want to better specify this idea. Right-click on the `Region` table's `Region id` column and select the **Set Parent Concept...** menu entry from the contextual menu. A dialog with the set of concepts just defined appears. Double-click on the **Entity\_Key** concept and assign it to the **Region id** field. This is the concept that we defined in our model to express the idea of "key of a table."

We can assign the same concept to more than one table's field at a time by selecting all of the table's fields at once. As soon as the **Set Parent Concept...** dialog closes, the concept's name is displayed to the right, with respect to the column's name.



A good approach suggests that we better specify the concepts associated with tables' columns by widely assigning a set of well-designed metadata concepts.

The second example shows you how to review and update some table fields' properties. To review a set of table fields' properties, we must first double-click on the table we want to update and let the **Physical Table Properties** dialog appear, showing the fields' properties. This dialog is similar to the **Parent Concept Editor** dialog. It displays the set of table's fields under the left-hand side tree view, named **Subject**. To the right-hand side of the **Subject** tree view, we have the **Available** view, which shows the field's properties organized through a categorized tree view. The property categories are the exact same categories we saw in the **Parent Concept** dialog. By selecting any property from the **Available** tree view, we have the value for the selected property in the **Setting** field set.



After we have made any changes to the table column properties in any order, we can confirm the changes and close the dialog by clicking on the **OK** button. If we want, we can close the dialog any time without confirming the changes by clicking on the **Cancel** button.

## See also

To review all the steps that take us to the definition of the metadata domain model's physical layer, look at the following recipes:

- ▶ *Managing JDBC database connections*
- ▶ *Defining the physical layer*

## Deriving business models from the physical layer

As soon as we have a consistent physical layer defined, we can proceed by taking our metadata model a step further. This recipe defines the metadata business layer. At this level, we can enrich our model with new attributes, multilingual labels and descriptions, new calculated attributes, or other interesting things.

## Getting ready

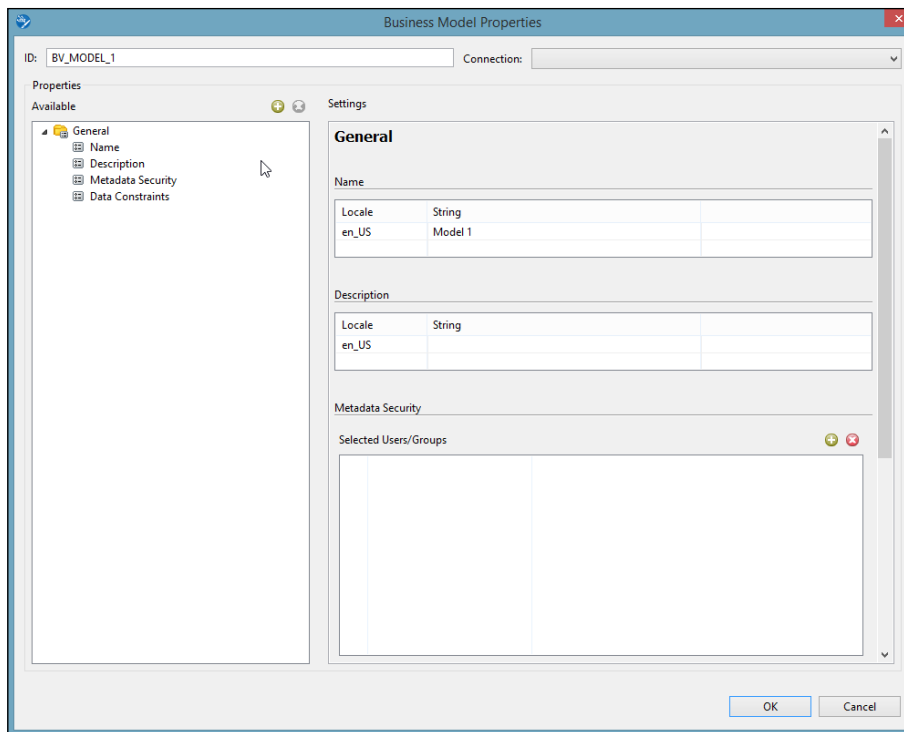
For this recipe, check the following conditions:

1. The Pentaho Metadata Editor must be started.
2. A physical layer of the database must already be imported and available.

## How to do it...

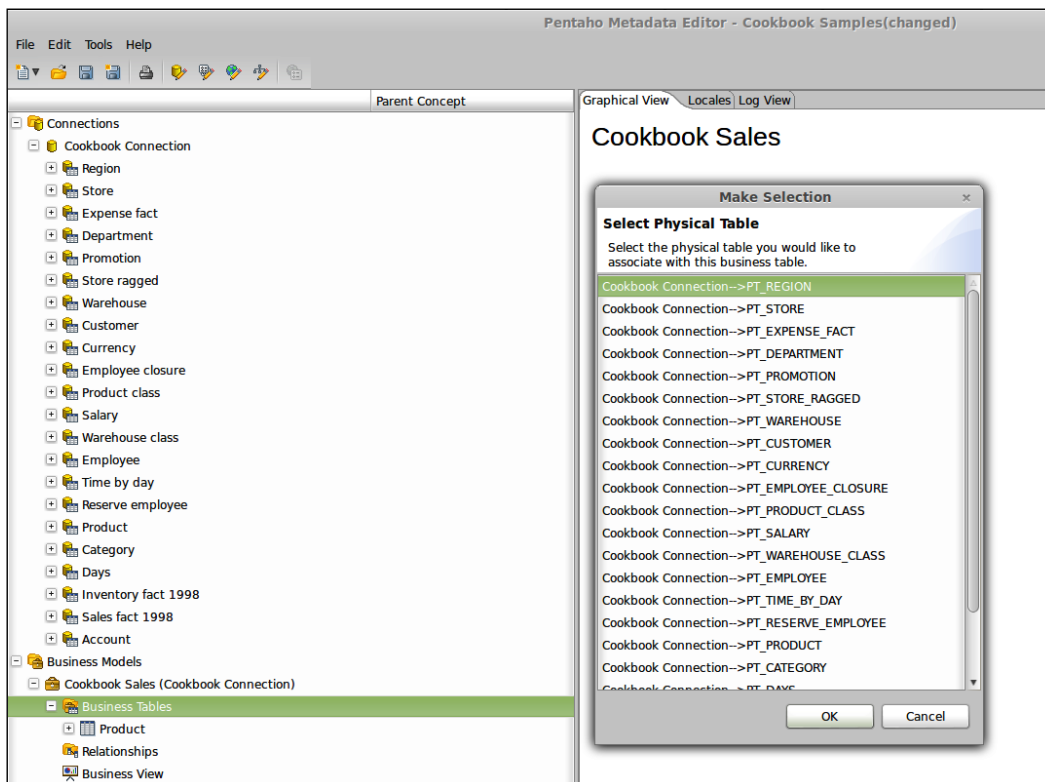
To start building our business model in the Pentaho Metadata Editor, we must perform the following steps:

1. From the left-hand side view in the Pentaho Metadata Editor, right-click on the **Business Models** folder and select **New Business Model...**
2. The **Business Model Properties** dialog box appears.



3. From the **Connection** drop-down list, select **Cookbook Connection**.
4. Under the **General** section of the **Settings** field set, the **Name** field contains the name of the model. The default assigned by the tool in this case is `Model 1`. We can change it to a more meaningful name by typing, for example, `Cookbook Sales`.

5. Click on **OK** to confirm the changes and close the dialog. We can leave the dialog at any time without confirming anything by clicking on the **Cancel** button.
6. If we expand the **Business Models** folder, we have three folders, which represent the object categories that compose a business model. Let's define the business tables for our model.
7. The first way to import tables in our model is by using drag-and-drop. Identify the tables we want to import from the physical level and drag-and-drop them under the **Business Tables** folder.
8. A second way is to use the **Business Tables** contextual menu. Right-click on the **Business Tables** folder and select the **New Business Table** menu item entry.
9. The **Select Physical Table** dialog box opens, as shown in following screenshot. Select the tables we want to import and click on **OK**:



10. To complete the model, we decided to import the following tables: Product, Store, Time by day, Store, Promotion, Sales facts 1998, and Inventory facts 1998.



## How it works...

From the left-hand side view in the Pentaho Metadata Editor, right-click on the **Business Models** folder, select **New Business Model...**, and open the **Business Model Properties** dialog. This dialog contains the properties associated with a specific business model. Any model has an **ID** field with a default value. We can leave the default value for a field as is; there is no need to change it.

To properly configure our new model, first choose the connection that contains the tables we want to map in the business model from the **Connection** drop-down listbox. Then, under the **General** section of the **Settings** field set, look at the **Name** field. This is the name that is assigned to the model, and it is the value displayed to the user in any Pentaho tools. The default is `Model 1`. We can change it to a more meaningful name such as `Cookbook Sales`. Eventually, we can set a description by typing it in the **Description** field and set one or more security constraints by adding granted users/groups in the **Selected Users/Groups** field. Click on **OK** to confirm the changes and close the dialog. At any time, we can leave the dialog without confirming anything by clicking on the **Cancel** button.

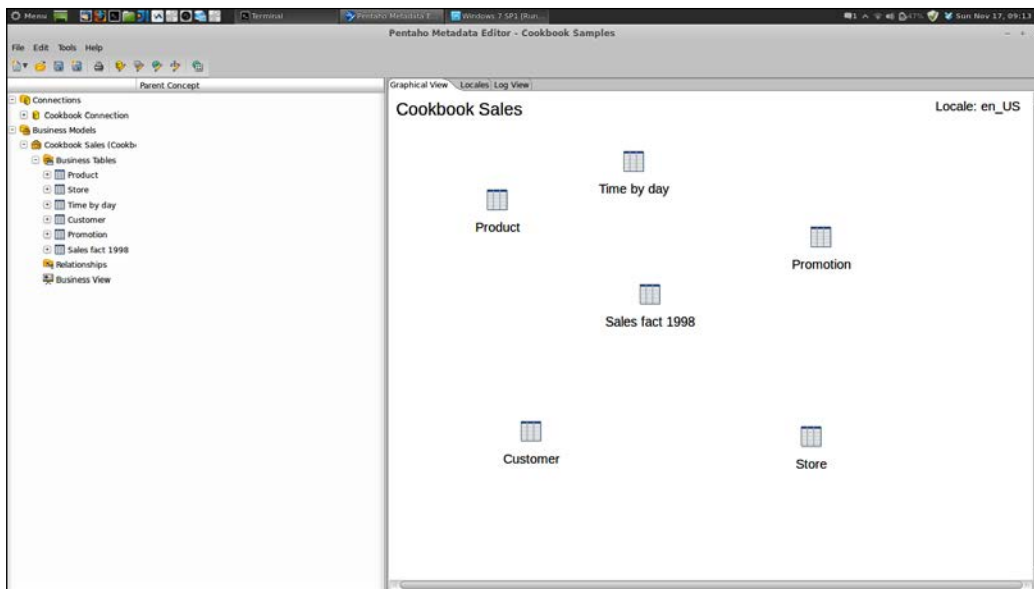
Now that we initialized the business model, under the `Business Models` folder, we have three folders that represent the object categories that compose a business model. The first thing we need to do in our business model definition is to define our logical model in terms of tables and relations. Therefore, we first need to take the physical layer's tables and import them in the business model as a set of business tables so that we can start working on them.

To import tables in the **Business Tables** folder, we have two possible ways. The first is to use drag-and-drop. In this case, to define a physical table as a business table for our model, identify the table we want to import from the physical level and drag-and-drop it under the **Business Tables** folder. The drawback of this approach is that we must process one table at a time; so, in the case of a great number of tables, it is very slow.

The second way is to use the **Business Tables** contextual menu. As soon as we get into the context menu, select the **New Business Table** item entry; the **Select Physical Table** dialog box opens. Select the tables we want to import (one or more in one shot) and click on **OK**.

For both cases, if we either dropped the table in the `Business Tables` folder or selected the table in the **Select Physical Table** dialog box, the **Business Tables Properties** dialog opens once for every table to review its properties. When we finished reviewing the **Business Table Properties** window, click on the **OK** button to close the dialog and confirm the properties for the current business table.

On the right-hand side of your screen, as shown in following screenshot, we see a **Business Tables** diagram that starts to appear. Any time we import a new table from the physical level, we notice that the table is added to the view on the left and appears in the model to the right. We can freely reposition the tables in this area as desired and follow our taste to get a better understanding of the model.



The screenshot shows business tables diagram that is starting to appear in design area

## Reviewing business tables' column properties

It is always a requirement to be able to review the properties associated with a business table, either to correct some errors from the import phase or to improve the actual definition of a business table. This recipe deals with the properties of the business table fields and their values.

### Getting ready

For this recipe, check the following conditions:

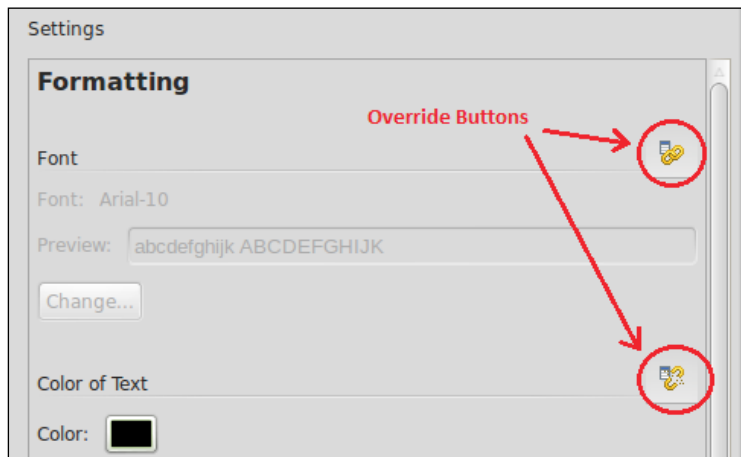
1. The Pentaho Metadata Editor must be started.
2. A set of business tables must be available in our business model to work on.

### How to do it...

Suppose, for example, we want to edit the properties for the **Promotion** business table. To review and update the properties for a business table, we must perform the following steps:

1. Double-click on the **Promotion** table or right-click on the **Promotion** table and select **Edit...** from the contextual menu.

- Both cases open the **Business Tables Properties** dialog.
- Suppose, we want to set the **Promotion Name** field's characters in bold. Select the **Promotion Name** field from the **Subject** tree view.
- Go to the **Available** tree view and select the property named **Font**. Under the **Settings** field set, we are taken directly to the value for the **Font** property. Click on the **Override** icon button at the upper-right corner of the **Font** value field to enable the update for this value (see the following screenshot for details about the button's position):



- Click on the **Change...** button. The **Choose a Font** dialog appears. Select **Bold** from the **Style** listbox and click on **OK**.
- As soon as the dialog closes and we go back to the **Business Tables Properties** dialog, we can immediately see the effect by checking the **Preview** field.
- Click on the **OK** button in the **Business Tables Properties** dialog to close the dialog and confirm the changes.

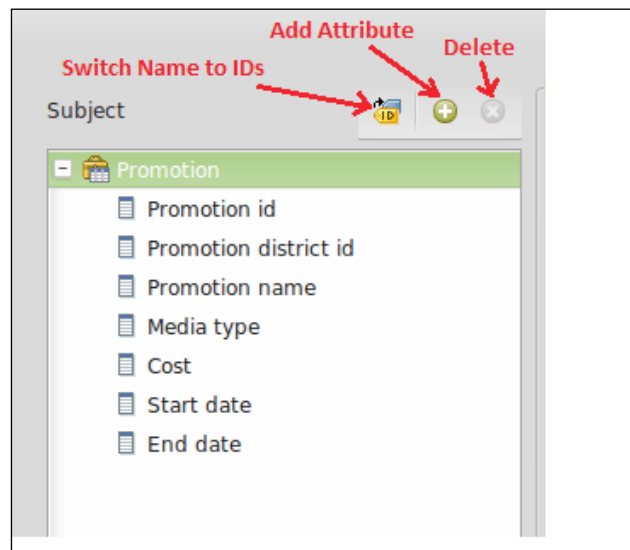
### How it works...

Business tables and business tables' fields have a set of properties associated with them. As we said in the previous recipes, even if we have properties for tables and columns in the physical layer, it is always a good approach to make any change or enhance our model in the business model layer.

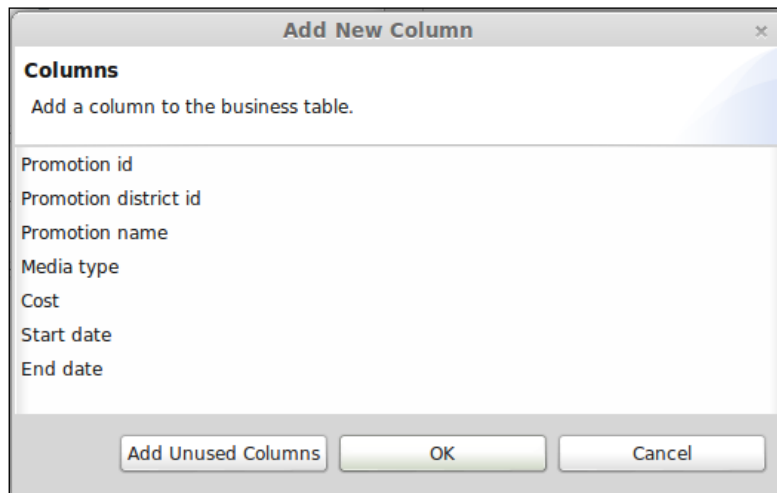
If we want to customize the business table after we have identified it, we can open the **Business Tables Properties** dialog by either right-clicking on the table and choosing **Edit** from the contextual menu or by double-clicking on that table.

As soon as the dialog opens, we have a tree view called **Subject** on the left that contains all of the table's fields. Look at the upper-right corner (see the following screenshot for details about the toolbar buttons' positions); we have a toolbar with three buttons, as follows:

- ▶ The first (leftmost) button switches the tree view items' labels by displaying their object IDs instead of names. We can switch between IDs and labels by clicking on the button.
- ▶ The second button adds a new attribute to the collection of fields for this table.
- ▶ The third will delete a selected field. Before confirming the deletion of the selected item, a confirmation dialog appears, asking if we are sure that we want to continue.



If we decide to add a new field to the business table (maybe a new field that will display a calculated value), the **Add New Column** dialog appears (see the next screenshot). This dialog shows us all the fields available in the associated physical table. This field set is the complete set of fields available in the associated physical table, and it does not consider the fields already present in the business table. The window to add a new column looks as follows:

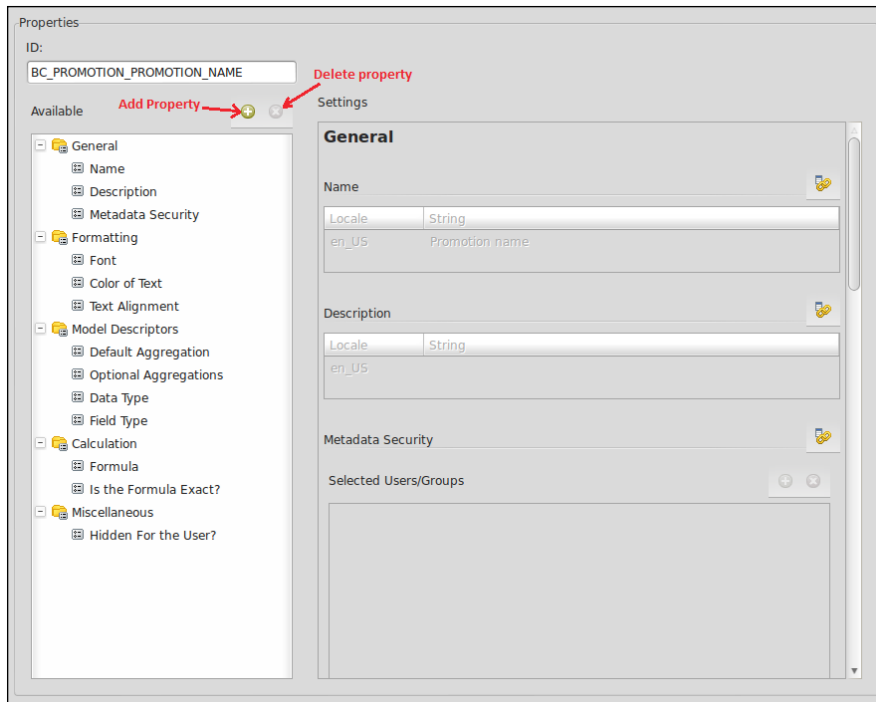


To add a column, select it by clicking on the desired column name and then click on the **OK** button. We can select more than one column at a time and, eventually, add all of the unused columns at once by clicking on the **Add Unused Columns** button.



In the Pentaho Metadata Editor, we cannot really add a brand new field to the table's field collection. Therefore, any time we talk about adding a new field to a business table, we create a new field in the business table by adding a physical table's field (no matter if the field is already a field in our business table), renaming it, and then modifying the associated properties set. This is a very important tweak.

As soon as we select a field from the **Subject** view, the **Properties** field's group displays the properties related to the selected field.



For each of the fields in the table, we can display the following information:

- ▶ An **ID** field in the top-left corner of the **Properties** section displays the internal **ID** of the business table field.
- ▶ A tree view name, **Available**, that contains all of the properties associated with this business table field.
- ▶ By selecting a property, the **Settings** form on the right shows the property name and its/their associated value. We can freely add or remove properties by clicking on the **Add** button located in the toolbar in the top-right corner of the **Available** tree view.

To update a value for a specific property, we first need to select it from the **Available** tree view. By default, any existing default property value will be read-only, and we can override the value for the property by clicking the override icon button.



Remember, as soon as we complete any change to the business table (both add and remove columns and any operation on the business table columns' properties), the changes are confirmed only by clicking on the **OK** button in the **Business Table Properties** dialog box. Finally, don't forget to save the domain model to make all of the changes persistent either by clicking the **Save** button on the Pentaho Metadata Editor toolbar or by navigating to **File | Save** from the menu.

## See also

- ▶ You can review what concepts are and how we can create them by reviewing the *Defining concepts* recipe
- ▶ Move a step further to see how we can change the formatting properties for the displayed values in the *Applying formatting properties to business tables' fields* recipe
- ▶ Define new calculated columns to enrich our model in the *Adding new calculated columns to model entities* recipe

## Applying formatting properties to business tables' fields

This recipe and the next will show some examples about how to update the existing business tables and fields. Let's start by showing you how to change some formatting attributes for a business table's fields.

### Getting ready

For this recipe, check the following conditions:

1. The Pentaho Metadata Editor must be started.
2. A business table must be created and a set of tables must be available.

### How to do it...

To override the text's color for a business table's field, perform the following steps:

1. Double-click on the **Promotion** table or right-click on the **Promotion** table and choose **Edit...** from the contextual menu.
2. Both cases open the **Business Tables Properties** dialog.
3. Suppose we wanted to override the **Promotion Name** field's text color by choosing another color.
4. The value for the **Colour of Text** property is displayed in the **Properties** form fields. Click on the **Override** button to open the **Colour Picker** dialog box.
5. Select the new color for the text. Click on **OK** to close the **Colour Picker** dialog and confirm the selection. The new value for the **Colour of Text** property is immediately visible.
6. Click on **OK** to close the **Business Table Properties** dialog box and confirm the changes.

## How it works...

Even if we define a wide vocabulary of basic concepts, it is sometimes necessary to override some of them because of specific business conditions. For this reason, we have the possibility to override business table fields' properties as required depending on the case. As for concepts, properties are grouped into categories to easily search and find them. This recipe explains a very simple example about how to change the text color for a business table's field. We do not spend much time on this because the process is similar to what we saw to create or update concepts or physical tables' fields.

Double-click on the **Promotion** table or right-click on the **Promotion** table and choose **Edit...** from the contextual menu to open the **Business Table Properties** dialog. As soon as the dialog opens, select the **Promotion Name** field and look for the **Colour of Text** property in the **Available** tree view. Because of the type of the property, a previewer (sort of) shows the actual color of the text. As usual, by default, the value is read-only. To enable the update, click on the override button located at the very right-hand side of the field and identified by the usual icon with a little chain. The property field is writable now. Select the new value for the text color from the **Colour Picker** dialog and click on **OK**. The **Colour Picker** dialog closes and the new value for the property is set and visible in the little previewer.

## See also

To get a complete description about the concepts and property values, see the following recipes:

- ▶ *Defining concepts*
- ▶ *Reviewing business tables' column properties*

## Adding new calculated columns to model entities

One good feature we get from metadata is that we can enrich our model by adding calculated fields not present in the physical model to solve specific business requirements. An example of this could be a calculated field that gets the total amount for an order line by multiplying the number of pieces sold with the unit price of that set of items. This recipe is about how to add new calculated fields to a business table.

## Getting ready

For this recipe, check the following conditions:

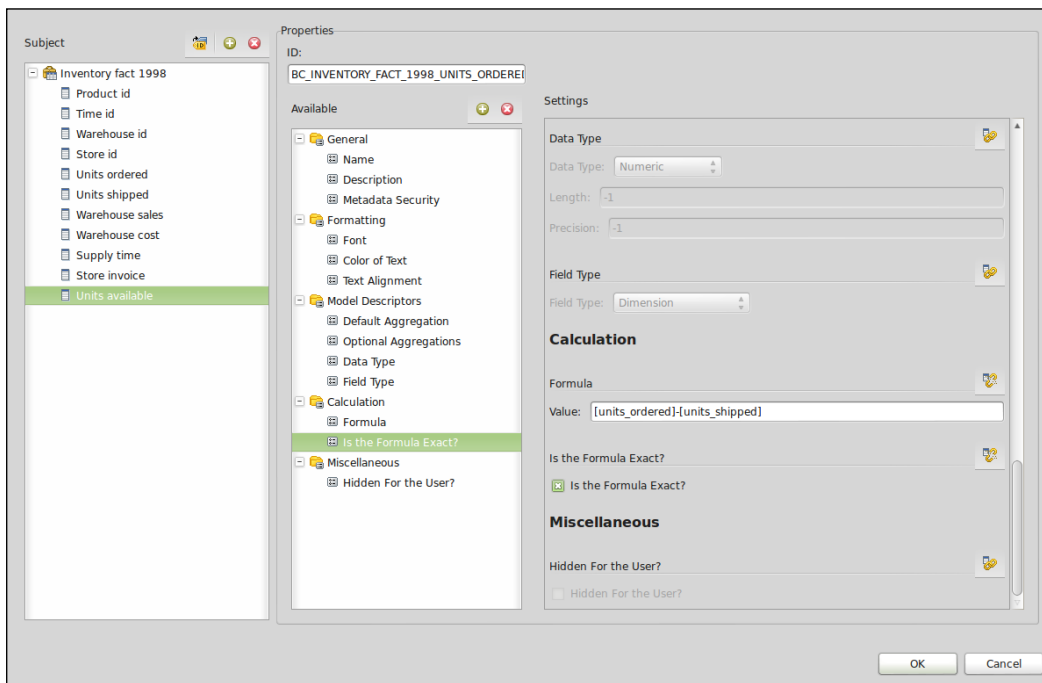
1. The Pentaho Metadata Editor must be started
2. A set of business tables must be available to work on.



## How to do it...

This recipe shows you how to define a field in order to calculate the number of items to be shipped by defining a field to subtract the values of the **Units Ordered** and **Units Shipped** fields. To define these new calculated fields, perform the following steps:

1. Double-click on the **Inventory facts 1998** table or right-click on the **Inventory facts 1998** table and select **Edit...** from the contextual menu.
2. In the toolbar located on the right-hand side of the **Subject** tree view, click on the **Add New Column** toolbar button. Select one of the fields (doesn't matter which field), for example, the **Units Ordered** field, add it another time, and rename it to **Units Available**.
3. Look for the **Is Formula Exact?** checkbox and set it to checked (see the following screenshot for details).
4. Look for the **Formula** field and type `[unit_ordered] - [unit_shipped]`.



5. The new field, **Units Available**, is displayed in the set of fields available for the **Inventory facts 1998** business table.
6. Click on the **OK** button in the **Business Table Properties** dialog box to confirm the changes and close the dialog box.



Any new field added to the table will appear in the last position of the field set for that business table. For the time being, it is not possible to change the position of the business table's fields and relocate them to other more appropriate locations. For example, it would be more clear if we had the capability to relocate the new field **Units Available** immediately after the **Unit Shipped** field.

### How it works...

Defining new calculated fields is a good feature. It lets us add new behavior to our tables. In this recipe, we went through the creation of a new field, **Units Available**, for the `Inventory facts 1998` table to calculate the difference between **Units Ordered** and **Units Shipped**.

Double-click on the **Inventory facts 1998** table or right-click on the **Inventory facts 1998** table and select **Edit...** from the contextual menu to open the **Business Table Properties** dialog. On the toolbar located in the right of the **Subject** tree view, click on the **Add New** column button to display the **Add New Column** dialog box. As we have already said, the addition of a brand new field to a fields' collection is not possible. Therefore, to workaround this, we again add an existing field and rename it accordingly. The Pentaho Metadata Editor does not complain about this. The idea is to add a field that has characteristics similar to the field we would like to have. Therefore, let's add, for example, the **Units Ordered** field a second time and rename it **Units Available**.

Let's type the formula to calculate the field's value. First, look for the **Is Formula Exact?** checkbox and set it to `true` (checked). What does the **Is Formula Exact?** property represent? The following points will provide a brief explanation about the meaning of this property:

- ▶ If the value for the **Is Formula Exact?** property is set to `true` (checked), the fields involved in the formula expression are identified by using the name of the corresponding physical table's field. In this case, the fields in the formula are identified by the physical field's name enclosed in between two square brackets, [`field_name`], where the value for the field name is specified.
- ▶ If the value for the **Is Formula Exact?** property is set to `false` (unchecked), the fields involved in the formula expression are identified by using the name of the corresponding business column's name. In this case, the fields in the formula are identified by the complete business table's field name enclosed in between two square brackets, [`table_name.<field_name>`], where the value for the field name is specified.

Then, we need to write the formula depending on the settings of the **Is Formula Exact?** property. In our case, because the value of the property **Is Formula Exact?** is set to `true`, the calculation formula is expressed as `[unit_ordered] - [unit_shipped]`. Click on the **OK** button in the **Business Table Properties** dialog box to confirm the data and close the dialog box.

## See also

For a complete description about the concepts and property values, see the following recipes:

- ▶ *Defining concepts*
- ▶ *Reviewing business tables' columns properties*

## Defining joins between business tables' entities

To completely describe our business model, we need to correlate our business tables by creating relations between tables using joins. This recipe illustrates how to link business tables together by using tables' relationships.

### Getting ready

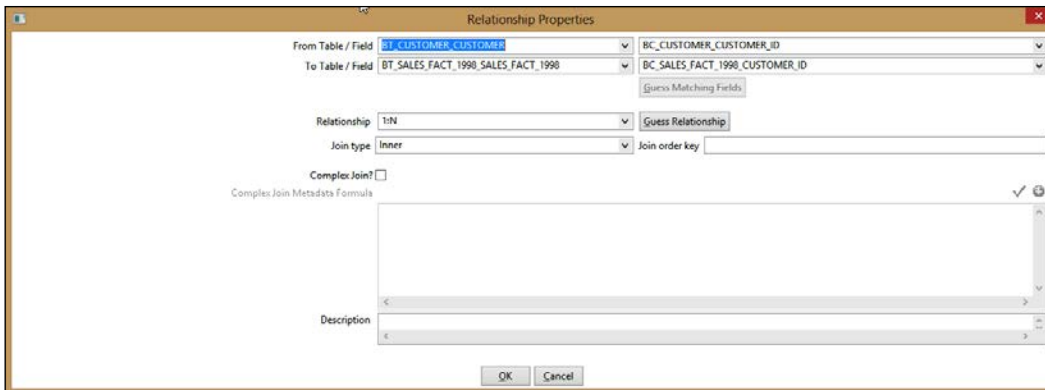
For this recipe, check the following conditions:

1. The Pentaho Metadata Editor must be started.
2. A set of business tables must be available to work on.

### How to do it...

To define a relationship between tables, the following steps must be performed:

1. Imagine, for example, we want to define a relationship between **Sales facts 1998** and **Customer tables**.
2. Look at them in the **Graphical** view. Select both the tables by pressing the *Ctrl* key and then clicking on each of them. Press the right mouse button and select **Add Relationship...** from the contextual menu.
3. The **Relationship Properties** dialog box opens as shown in the following screenshot:

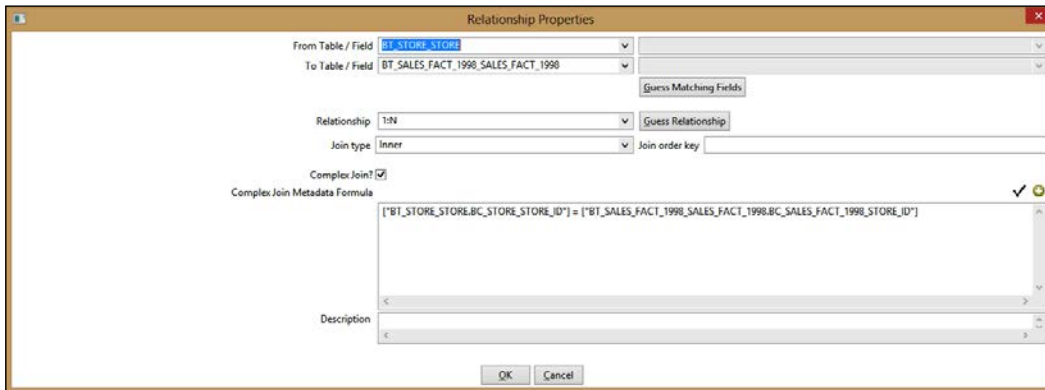


4. From the right drop-down list of the **From Table / Field** field set, select the **BC\_CUSTOMER\_CUSTOMER\_ID** entry.
5. From the right drop-down list of the **To Table / Field** field set, select the **BC\_SALES\_FACT\_1998\_CUSTOMER\_ID** entry.
6. Look at the **Relationship** drop-down list and choose the item that best fits with our model; set the relationship type to **1 : N**.
7. Look for the **Join type** drop-down list. This list contains the available join types. Select the **Inner** join type entry.
8. Click on the **OK** button to confirm the relationship we just created and close the **Relationship Properties** dialog.
9. As soon as we close the **Relationship** dialog in the **Graphical** view workspace, we can graphically see the relationship we just created.

We can also define join conditions by using metadata formulae. Let's rewrite the previous example by expressing the join condition the following way:

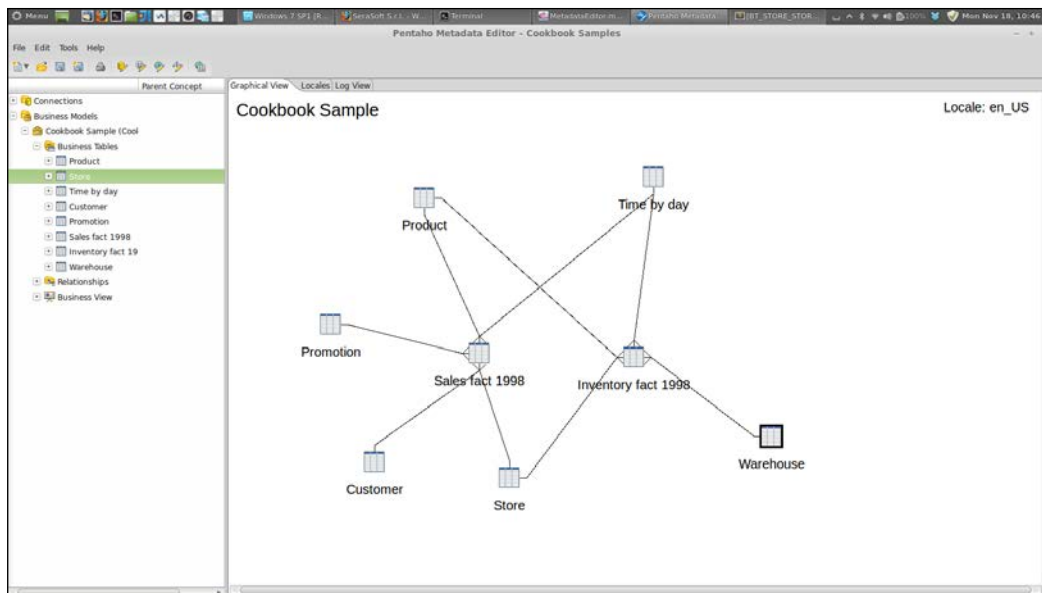
1. Look at the **Graphical** view. Select both the tables by pressing the **Ctrl** key and then clicking on each of them. Press the right mouse button and select **Add Relationship...** from the contextual menu.
2. The **Relationship Properties** dialog box opens. Do not select anything from the **From Table / Field** and **To Table / Field** drop-down lists.
3. From the **Relationship** drop-down list, select **1:N**.
4. From the **Join type** drop-down list, select **Inner**.

5. Set the **Complex Join?** checkbox field to checked as shown in the following screenshot. The **Complex Join?** metadata formula text area will be enabled, so we can enter the join formula. Use the helper dialog box to correctly write the join condition:



6. Look at the toolbar in the upper-right corner of the **Complex Join Metadata** text area field. Click on the **Add Join Condition** toolbar button.
7. The **Add Join Condition** dialog box opens. Select the two fields (basically, the `CUSTOMER_ID` field) that participate in the join condition from the two drop-down lists. Click on the **OK** button to confirm the join field and to close the dialog.
8. As soon as the dialog closes, the text of the join formula will be visible in the text area field. Verify the correctness of the formula by clicking on the **Validate Formula** button. If it's fine, click on **OK** in the **Relationship Properties** dialog box to confirm the join and close the dialog.

As an exercise, try to define all of the remaining relationships in the model. At the very end, we will have all of the relationships defined in the graphical view of the Pentaho Metadata Editor as displayed in the following screenshot:



## How it works...

Any time we have a fact table in our model, we will definitely have dimension tables to link to these facts. In our model, we have two fact tables (*Inventory facts 1998* and *Sales facts 1998*) and a set of dimensional tables. A set of relationships must be defined in between these tables to complete the definition of our model.

Let's start by, for example, defining the relationship between the **Customer** dimension and the *Sales facts 1998* fact table. To do this, look for the two tables in the **Graphical** view of the Pentaho Metadata Editor. Select both the tables by pressing the *Ctrl* key and clicking on each of them. With the two tables selected, select **Add Relationship...** from the contextual menu to open the **Relationship Properties** dialog box. As soon as it opens, two sets of drop-down lists labelled **From Table / Field** and **To Table / Field** let you easily choose the join fields. For any of the two sets, the first drop-down identifies the table involved in the relationships by using its internal ID. The second drop-down list identifies the table's field to be used as the join key for the specific table. Type `BC_CUSTOMER_CUSTOMER_ID` in the **From Table / Field** drop-down and `BC_SALES_FACT_1998_CUSTOMER_ID` in the **To Table / Field** drop-down field.

What we just defined is the relationship between the primary key of the *Customer* dimension table and the same primary key from the *Sales fact 1998* fact table. Now let's define the kind of relationship and the cardinality. The **Relationships** drop-down list contains the possible relationship types. Select `1 : N`. The **Join type** drop-down list contains the possible join types, the same as we have in SQL. From this list, select `Inner`. Click on the **OK** button to confirm the relationship we just created and close the **Relationship Properties** dialog. Now, in the **Graphical** view workspace, we can see the relationship we just created graphically.

Another way to define a join condition is by using a metadata formula. From Pentaho Version 5.0, this is easier because we have a sort of editor that helps us to write the complex join condition. The case for a complex join condition would be when our tables use compound keys instead of surrogate keys because more than one field is involved in the join condition. We do not have such a condition in our model but to show the functionality, we decided to rewrite the condition of our previous example using this way. In this case, instead of selecting the keys from the **From Table / Field** and **To Table / Field** drop-down lists, click on the **Add Join Condition** toolbar button located in the upper-right corner of the **Complex Join Metadata Formula** field to open the **Add Join Condition** dialog box. Here, let's use a little wizard that guides through the creation of the join condition and then writes the correct formula for us:

1. Select the two fields we want to add to the join condition from the two drop-down lists, one for each of the tables participating in the join. In our case, the field is named `CUSTOMER_ID` for both.
2. Click on the **OK** button to confirm the join fields and close the dialog.
3. As soon as the dialog closes, the text of the join formula is visible in the text area field. We can verify the formula's correctness by clicking on the **Validate Formula** button in the toolbar. Type `1 : N` in the **Relationship** drop-down list and `Inner` in the **Join type** list.
4. Click on the **OK** button in the **Relationship Properties** dialog box to confirm the join and close the **Relationship Properties** dialog.



Whenever we consider the use of a complex join, it is not required to use the **Add Join Condition** helper dialog to define the condition. We can always write the complex join expression manually, but this may lead to possible errors. Remember to always validate the typed expression by clicking on the **Validate Expression** toolbar button.

## Creating business view categories

Now that we have completely defined our model, we need to think about how the user will access the model. In the Pentaho Metadata Editor, the user's access to the model is through business view categories. Business view categories are named collections of attributes related to a specific business subject matter. For example, the `Customer` category may expose the attributes strictly correlated to the customer data, such as `customer code`, `category`, `name`, `city`, or other similar attributes. This recipe shows how easily we can define our own categories to let the user access our model.

## Getting ready

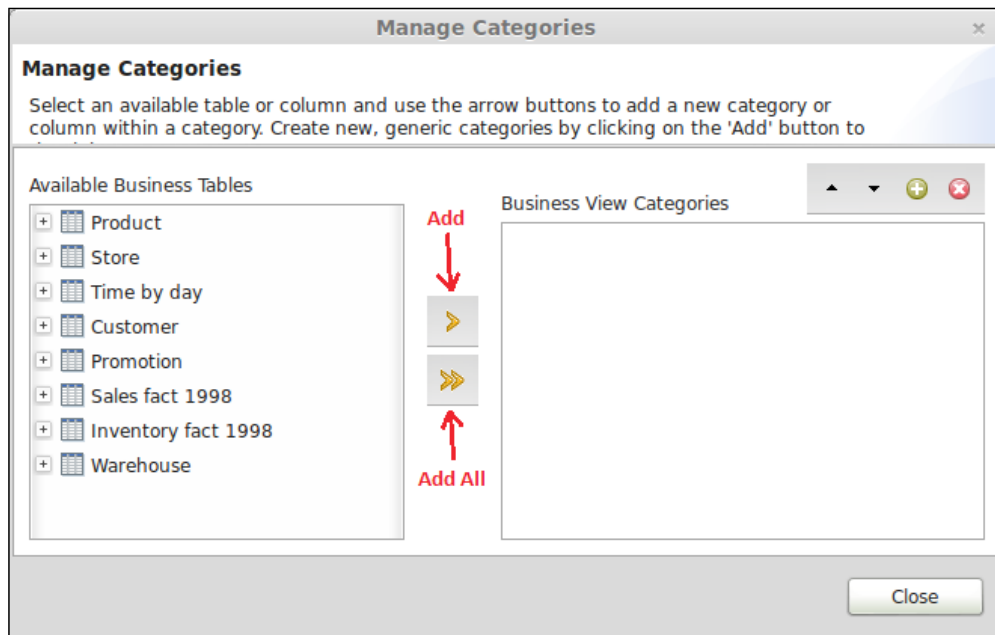
For this recipe, check the following conditions:

1. The Pentaho Metadata Editor must be started.
2. Our business model has been completely modelled, in terms of business tables' definitions and their relationships.

## How to do it...

To define a sample business view category, perform the following steps:

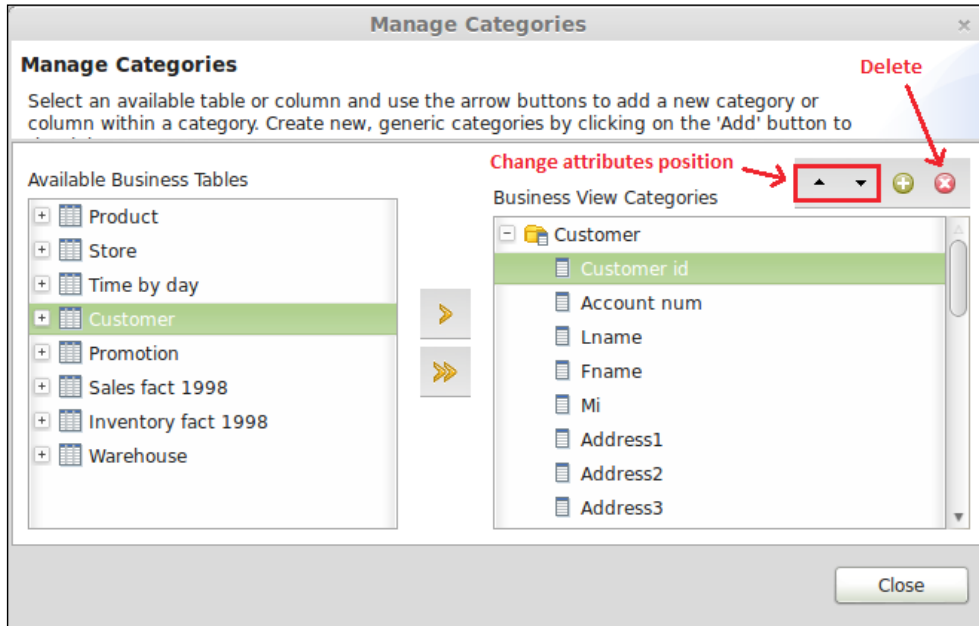
1. In the Pentaho Metadata Editor, right-click on the **Business View** folder and select **Manage Category** from the context menu.
2. The **Manage Category** dialog box opens as shown in the following screenshot:



3. Suppose we want to map a **Customer** category. Select the **Customer business table** from the **Available Business Tables** tree view on the left-hand side and click on the add button to move the business tables to the right.



- Now that we moved all of the table's fields, we need to remove the unneeded fields. Expand the **Customer** business view category in the tree view on the right-hand side, as shown:



- Select the **Customer id** field and click on the delete button (look at the previous screenshot) in the top-right corner toolbar of the **Business View Category** tree view. Repeat with the **Customer region id** field. We can find it in the list below.
- Add all the dimensions' tables as business view categories and remove ID fields from any category of the key attributes by repeating steps 6 to 9.
- As soon as you have finished with the dimension tables, repeat this for the fact tables. Even in this case, remove all of the meaningless key attributes. Again, we can easily do this by iteratively following steps 6 to 9.
- Click on the **OK** button and close the **Manage Categories** dialog box.
- Suppose we now want to rename the **Sales fact 1998** category to **Sales fact**.
- Double-click on the **Sales fact 1998** category. The **Business Category Properties** dialog box opens.
- Select the **Name** property from the **General** category in the **Available** properties tree view and change the value from **Sales facts 1998** to **Sales facts**. Click on **OK** to confirm the change and close the **Business Category properties** dialog.
- Repeat this for the **Inventory fact 1998** table and change the name to **Inventory facts** by repeating steps 14 to 15.

## How it works...

Business view categories represent a sort of view over our business tables. Defining a business view category is possible through the **Manage Category** dialog box. To open it, from the Pentaho Metadata Editor, right-click on the business view folder and select **Manage Category** from the context menu.

The **Manage Category** dialog box is divided in two parts; on the left-hand side, we find a tree view containing the business tables we defined in our model (**Available Business Tables**). As soon as we have at least one business view category, on the right-hand side, we will have a tree view with the business view categories we just defined.

Suppose we want to map a **Customer** category. Select the **Customer** business table from the **Available Business Tables** tree view on the left and click on the add button to move the table to the right. This automatically defines a category with the same name as the business table name and the same set of fields. Now we need to remove all of the unneeded fields (ID fields or others) that are meaningless for the user. Look at the **Customer** business view category in the tree view on the right and select the **Customer id** field. Click on the delete button in the top-right corner toolbar of the **Business View Category** tree view to remove the field from the set of fields exposed by this business view category. Repeat this for **Customer region id**, which we can find below in the list. To fully expose all the models, add all of the dimensions and fact tables as business view categories and remove the key attributes from any category by repeating the steps 6 to 9.

At this point, we have our complete set of business view categories defined. Click on the **OK** button and close the **Manage Categories** dialog box. Now, as immediate children of the **Business View** folder, we can see all our business view categories. To complete our task, we can rename the **Sales fact 1998** category to **Sales fact**. To do this, double-click on the **Sales fact 1998** category and open the **Business Category Properties** dialog.

Select the **Name** property from the **General** category in the **Available** properties tree view and change the value of the name property from `Sales facts 1998` to `Sales facts`. Click on **OK** to confirm the change and close the **Business Category Properties** dialog.

Repeat this for the `Inventory fact 1998` table and change the business view category name to **Inventory facts**.

## Testing metadata layer results

The Pentaho Metadata Editor contains a simple tool called Query Builder that uses your metadata model and queries it for testing purposes. The tool is a sort of wizard that, by browsing business view categories, gives you the ability to specify an output dataset, a query condition, and an ordering scheme for our records (the last two are not mandatory). It is more or less like building a SQL query but in a more abstract way, which is especially useful for nontechnical users.

Before deploying a domain model to our Pentaho BA Server, it is a good rule of thumb to use this tool and test the model to see whether it works. This recipe shows you how we can use Query Builder to query our model and test it.

## Getting ready

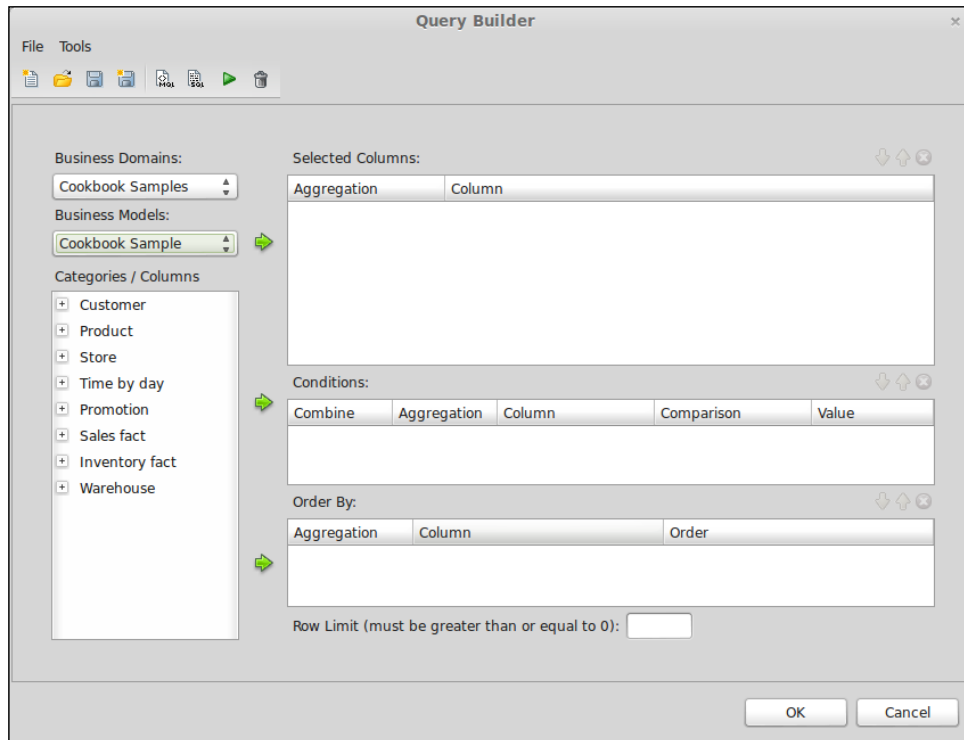
For this recipe, check the following conditions:

1. The Pentaho Metadata Editor must be started.
2. A set of business view categories must be defined to be used.

## How to do it...

To build a query on your brand new metadata model without specifying any filter, perform the following steps:

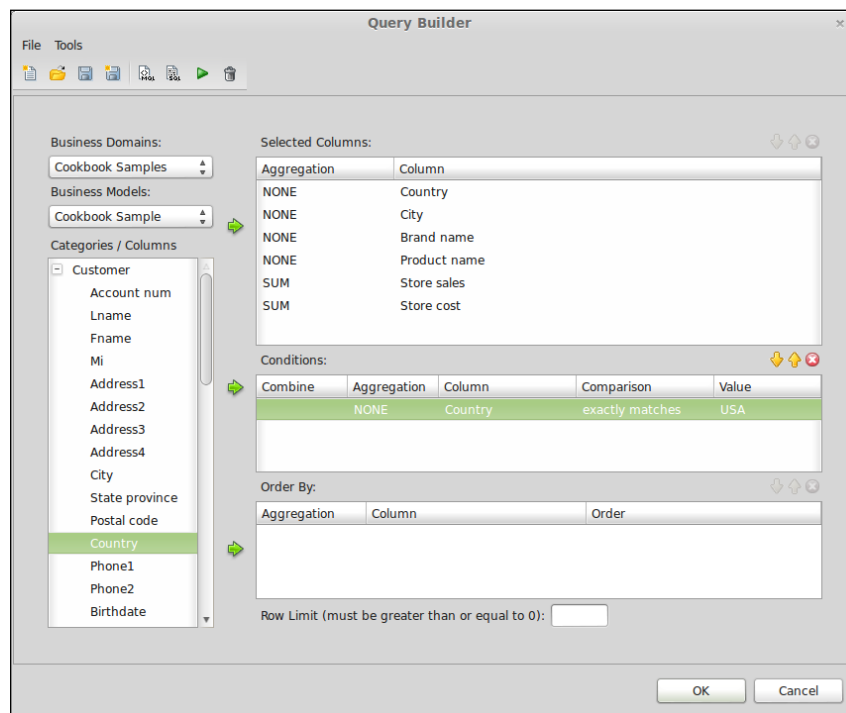
1. To open the **Query Builder** dialog box, either click on the related toolbar button or go to **Tools | Query Editor...** from the Pentaho Metadata menu.
2. The **Query Builder** dialog box opens as shown in the following screenshot:



3. Our business domains and associated business models are already selected by default in their related drop-down lists. They are selected by default because we just have one business domain and one business model.
4. From the **Categories / Columns** tree view, expand the **Product** category; press and hold the *Shift* key down and select the **Brand name** and **Product name** columns. Click on the green arrow to move the selected fields to the **Selected Columns** table.
5. Expand the **Customer** category and repeat step 4 for the **Country** and **City** columns.
6. Expand the **Sales fact** category and repeat step 4 for the **Store sales** measure column.
7. Click on the play button in the **Query Builder** dialog toolbar to execute the query. The **Examine Preview Data** dialog will open, displaying all the results coming from your query. Click on the close button and close the **Examine Preview Data** dialog to return to the **Query Builder** dialog.

Starting from where the previous example ended, let's specify a filter to display only the rows related to `Country = 'USA'`. To do this, perform the following steps:

1. In the **Customer** category, select the **Country** field and click the green arrow in the middle. The **Country** column will appear in the **Conditions** table. Open the drop-down box under the **Comparison** column and select the value, **Exactly matches**. Then, set the value for the **Value** column to **USA**, as shown in following screenshot:



2. Click on the play button in the **Query Builder** dialog toolbar to execute the metadata query. The **Examine Preview Data** dialog will open, displaying all the results coming from our query. Click on the close button and close **Examine Preview Data** to return to the **Query Builder** dialog.

## How it works...

Pentaho Metadata Query Builder is a simplified query builder to test our metadata model by querying it and see if, given a specific metadata query, we can get out a set of expected results. To open the Metadata Query Builder, we can either click on the related toolbar button or go to **Tools | Query Editor...** in the Pentaho Metadata Editor menu. The **Query Builder** dialog box opens.

The **Query Builder** dialog box has a typical "wizard-like" structure, and we will find a similar, if not the same, dialog in any case we are required to query a metadata model in any tool of the Pentaho family. The first thing to do is select the domain model and the business model both from the business domain's and business model's drop-down lists (we just have one in both cases, so they are already selected by default). As soon as this selection is made, the model categories and their columns are loaded in the **Categories / Columns** tree view.

On the right-hand side, we have three main tables that will be populated with items selected from the **Categories / Columns** option. The upper table (**Selected Columns**) contains the set of columns we are going to define as the select columns; they will represent the output dataset's layout. The table in the middle (**Conditions**) represents an area where we can define the where clauses in a simple and effective way. The lower table (**Order by**) is an area where we can build order by clauses.

Let's build a simple query. Imagine we want to find the amount of all the store sales by product and customer. From the **Categories / Columns** tree view, expand the **Product** category, then press and hold the *Shift* key, and select the **Brand name** and **Product name** columns. Click on the green arrow. The two fields will move to the **Selected Columns** table. Repeat this for the **Country** and **City** fields from the **Customer** category and the **Store sales** field from the **Sales fact** category.

Click on the play button in the **Query Builder** dialog toolbar to execute the metadata query. The **Examine Preview Data** dialog will open, displaying all the results that come from our query. Click on the close button and close the **Examine Preview Data** dialog to return to the **Query Builder** dialog.

Suppose that we now want to set a filter to the **Country** field so that we can display only the rows related to `Country = 'USA'`. Expand the **Customer** category and select the **Country** column. Click on the right arrow in the middle. The **Country** column will appear in the **Conditions** table. The **Comparison** column has a drop-down list that contains all of the possible comparison operators available. Select an exact match. Finally, set the **Value** column to the value **USA**. This is equivalent to the right part of the evaluation in the SQL where clause.

Click on the play button in the **Query Builder** dialog toolbar to execute the metadata query. The **Examine Preview Data** dialog will open, displaying all the results that come from our query. Click on the close button and close the **Examine Preview Data** dialog to return to the **Query Builder** dialog.

As soon as we finish testing our metadata domain model, click on the **OK** button and close the **Query Builder** dialog.

## See also

- ▶ Recap how to prepare business view categories by looking at the *Creating business view categories* recipe

## Applying security to the domain model elements

One goal of a well-designed domain model is how deep we can apply a set of constraints to control access to the domain model elements based on users' roles. This recipe will introduce how you can define a set of security constraints on your domain model to implement an effective access control policy.

## Getting ready

For this recipe, check the following conditions:

1. The Pentaho Metadata Editor must be started.
2. A database physical layout must be already imported and available.
3. One or more set of business tables must be available to work on.
4. One or more set of business view categories must be available for use.

## How to do it...

Suppose we want to limit access to the **Inventory facts** business view category only to the members of the `cookbook_users` role:

1. Expand the **Business View** categories folder and double-click on the **Inventory fact** category.
2. The **Business Category Properties** dialog box opens.
3. From the **Available** tree view on the left-hand side of the dialog, select the **Metadata Security** property. Click on the **Add New Users/Roles** button in the toolbar located in the top-right side of the **Selected Users/Group** table.

4. The **Add Permissions** dialog box opens, showing all the users and roles in the **Available Users/Roles** list on the left. Select the `cookbook_users` role and click on the right arrow to move the selected item to the list on the right.
5. Click on the **OK** button to confirm and close the **Add Permissions** dialog.
6. The selected role, `cookbook_users`, now appears in the **Selected Users/Group** table of the **Metadata Security** property.
7. Click on the **OK** button of the **Business Category Properties** dialog to confirm the changes and close the dialog.

## How it works...

After we make sure that the Pentaho Metadata Editor can access the Pentaho BA Server security provider to get users and roles (see the *Linking the Pentaho Metadata Editor security to Pentaho BA server* section in the *There's more...* section), we can proceed by applying the security constraints to our domain model objects.



A preliminary session with our end users is needed to define an access control matrix in order to map our company's users and roles to metadata domain model objects. This session must be a part of the requirement gathering process to be conducted before the system implementation.

Suppose we want to give access to the **Inventory facts** business view category only to a particular member of the `cookbook_users` role. Any object in our domain model has a property named **Metadata Security** that lets us specify the set of users and roles that can access the specific metadata element.

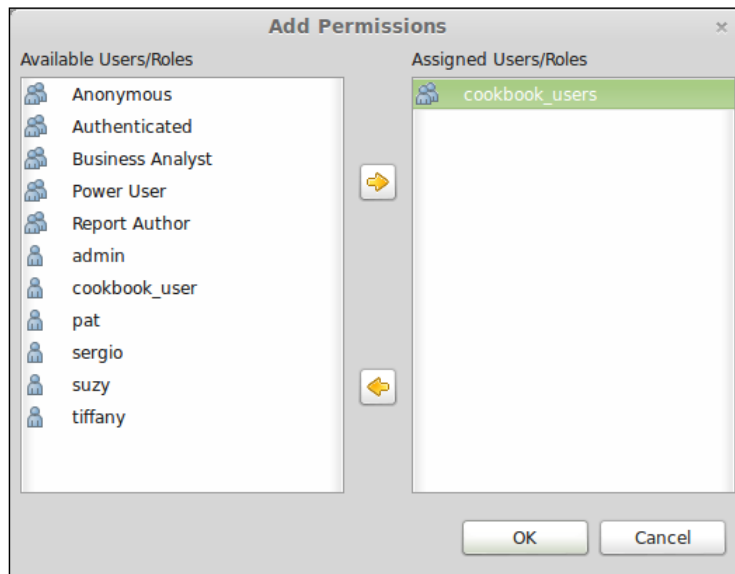
Going back to our example, to apply security constraints to the **Inventory facts** business view category, expand the **Business View categories** folder and double-click on the **Inventory fact** category. As soon as the **Business Category Properties** dialog box opens, from the **Available** tree view on the left-hand side of the dialog, select the **Metadata Security** property.



Any domain model object that has not got the **Metadata Security** property set will always be visible to any users and roles of the Pentaho platform.

Click on the **Add New Users/Roles** button in the toolbar located in the top-right side of the **Selected Users/Group** table. The **Add Permissions** dialog box opens, showing all users and roles in the **Available Users/Roles** list to the left. All these users and roles are taken from Pentaho BA Server, which manages the connection with a chosen enterprise authentication platform (for example, LDAP, MSAD, and so on).

The dialog, shown in following screenshot, gives you the possibility to choose the users and/or roles to apply your constraints to the selected object. As per our example, select the role, `cookbook_users`, and click on the right arrow to move the selected item to the list on the right-hand side named **Assigned Users/Roles**. This list contains the users and/or roles that have already assigned constraints on this object, if there are any.



Click on the **OK** button to confirm the selection and close the **Add Permissions** dialog. The selected role, `cookbook_users`, now appears in the **Selected Users/Group** table of the **Metadata Security** property. Click on the **OK** button of the **Business Category Properties** dialog to confirm the changes and close the dialog.

### There's more...

The first thing we need to do before applying security constraints to users and roles is to set up the connection between the Pentaho Metadata Editor and our BA Server security provider. This is detailed in the following section.

## Linking the Pentaho Metadata Editor security to Pentaho BA Server

To link the Pentaho Metadata Editor to the Pentaho BA Server security provider, perform the following steps:

1. From the Pentaho Metadata Editor menu, go to **Tools | Security...**



- The **Security Service** dialog box opens, as shown in following screenshot. This dialog collects the configuration information used to connect to the security provider of Pentaho BA Server. With this connection in place, we will be able to apply security constraints to metadata objects based on our corporate users and roles.

**Security Service**

Retrieve the security information from the Pentaho server. Choose to set up a security service, or get the values from a local file.

Service Proxy File

Service URL:   
Example: http://localhost:8080/pentaho/ServiceAction

Detail Type: 

- All
- Users
- Roles

Username:

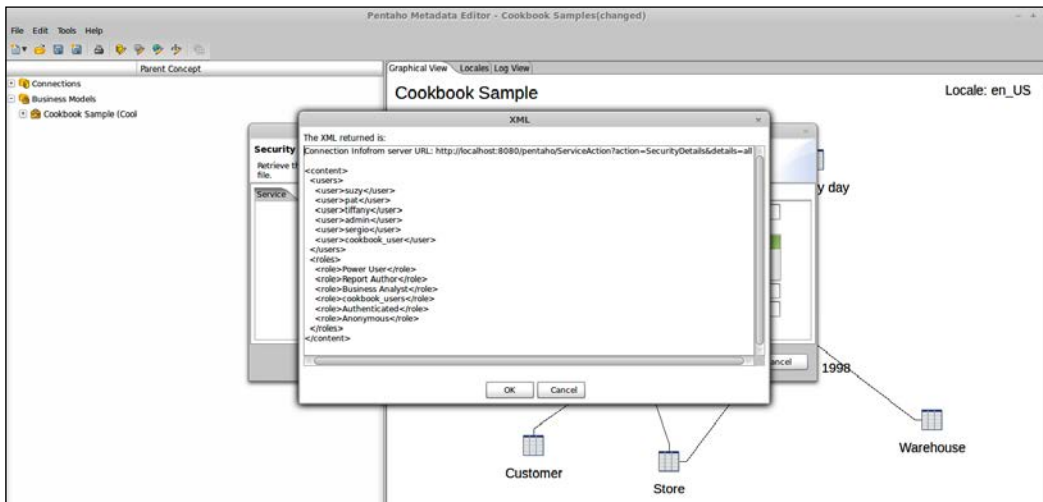
Password:

Test

OK Cancel

- Check whether the **Service URL** field is filled correctly according to our server DNS name (or IP address) and port. If not, apply the changes needed and correct the values.
- In the **Detail Type** listbox, select the level of detail we want to retrieve from the security provider: **Users**, **Roles**, or **All** (default). Leave it as it is for this example.
- In the **Username** field, type the username of the user we are going to use to authenticate the Pentaho Metadata Editor to Pentaho BA Server. This field is mandatory.
- In the **Password** field, type the password for the user we are going to use to authenticate the Pentaho Metadata Editor to Pentaho BA Server. This field is mandatory.

- By clicking on the **Test** button, Metadata Editor tries to authenticate to Pentaho BA Server. If the authentication succeeds, we get an XML file back from the security provider whose content depends on the value chosen in the **Detail Type** listbox. If we choose **All** in the **Detail Type** list, then the XML file will contain a subset of both the users and roles we have in the authentication system. See the following screenshot for details:



- Click on the **OK** button and close the **Security Service** dialog box.

## See also

To recap how to define and manage users and roles in the Pentaho BA platform, go through the recipes in *Chapter 2, Configuring Your BA Server Instance*. There is a set of recipes on how you can manage users and roles in the Pentaho users' database and on how you can connect Pentaho to an external authentication source such as LDAP.

## Publishing metadata definitions to BA Server

Our model is ready and now it is time to publish it to our Pentaho BA Server in order to let the users access it through the reporting tools. Some of the tools that make use of a metadata model are the Interactive Report Designer (for the EE version), WAQR, and Saiku Reporting (the last two can be used with both the versions of Pentaho BA Server). We can also make use of Metadata in Pentaho Report Designer. This recipe shows how you can publish our model successfully to the Pentaho BA Server.

## Getting ready

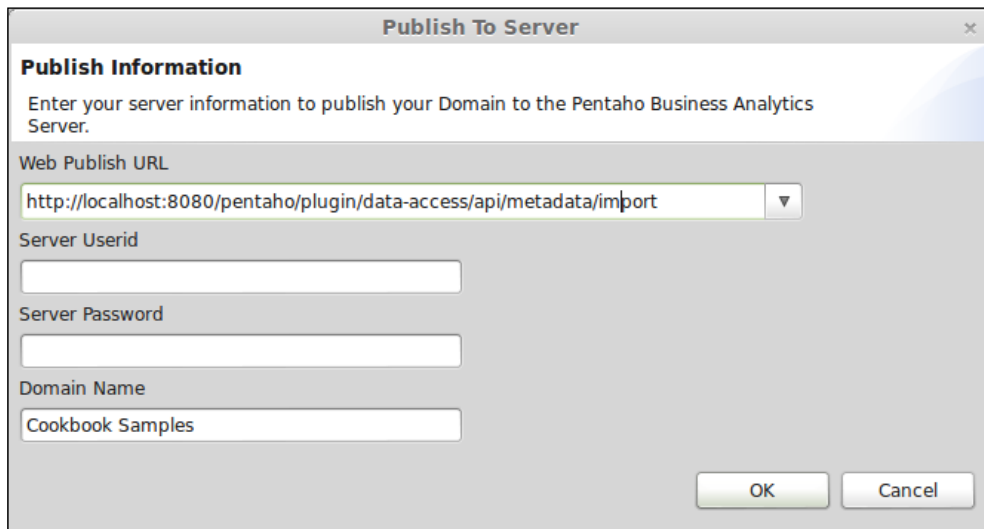
For this recipe, check the following conditions:

1. The Pentaho Metadata Editor must be started.
2. The user we are going to use to publish the Metadata model to Pentaho BA Server must be a member of a role with the Publish Content operational permission.
3. A complete Pentaho metadata model with a set of business view categories must be available.
4. A data source named `foodmart_mondrian` to connect to our SQL samples database must be available on Pentaho BA Server.

## How to do it...

To publish the Pentaho Metadata model to Pentaho BA Server, perform the following steps:

1. Select the **Publish to Server...** entry under **File** from the Pentaho Metadata Editor.
2. The **Publish to Server** dialog box opens as shown in following screenshot:



**Publish To Server**

**Publish Information**

Enter your server information to publish your Domain to the Pentaho Business Analytics Server.

Web Publish URL  
http://localhost:8080/pentaho/plugin/data-access/api/metadata/import

Server Userid

Server Password

Domain Name  
Cookbook Samples

OK Cancel

3. Check that the URL in the **Web Publish URL** field is correct. If not, apply the necessary changes according to the server's DNS name (or IP address) and port.
4. In the **Server Userid** field, type the username of the user we are going to use to authenticate to the Pentaho server. This field is mandatory.

5. In the **Server Password** field, type the password for the user we are going to use to authenticate to the Pentaho server. This field is mandatory.
6. The **Domain Name** field is already set to the right value but, in any case, check it's correct.
7. Click on the **OK** button to publish the metadata domain model to Pentaho Server. A message box informs you about the success of the metadata model publication.

### How it works...

The publication of a metadata domain model to Pentaho BA Server is a necessary step to make the model available to the user.

To open the **Publish to Server** dialog box, go to **File | Publish to Server...** from the Pentaho Metadata Editor. This dialog box lets us configure all of the required parameters to connect to Pentaho BA Server and publish the metadata model.

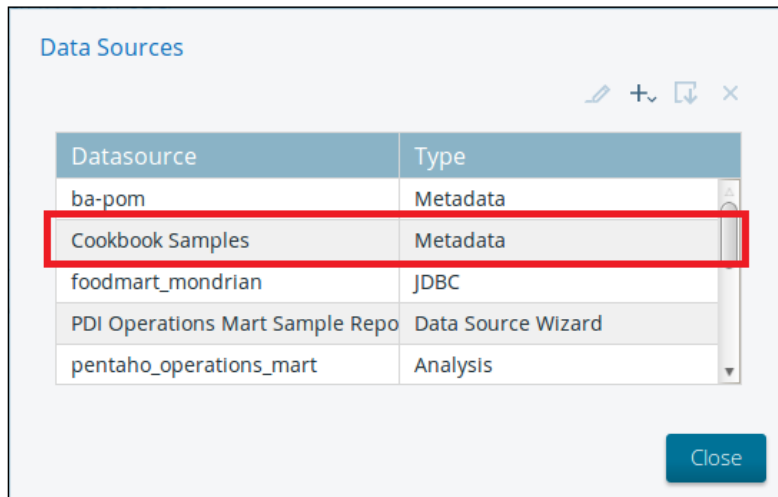


With the new Pentaho BA Server 5.0, the metadata domain publishes the URL changed from the old value `http://<server_name>:<port>/pentaho/RepositoryFilePublisher` to the new value `http://<server_name>:<port>/pentaho/plugin/data-access/api/metadata/import`. Verify whether the actual URL for the **Web Publish URL** field is set to the new value. If it is not, change it according to the sample URL provided in this hint.

Check that the web published URL is correct according to the server's DNS name (or IP address) and port. If not, apply the necessary changes. Then, set the **Server Userid** and the **Server Password** fields to the username and password, respectively, for the user that we are using to authenticate to the Pentaho server. These fields are mandatory. The **Domain Name** field is already set to the right value but, in any case, check it's correct. As soon as every field is filled correctly, click on the **OK** button to publish the metadata domain model to Pentaho BA Server. A message box informs you about the success of the metadata model publication.

The domain model we published just now is available in Pentaho BA Server as a new data source in the data sources list. To verify this, log in to Pentaho BA Server as a user that is a member of the **Administrator** group. From the **Home** perspective, click on the **Manage Data Source** button.

As expected, in the **Data Sources** dialog box, a new data source (of the same name as the domain model we just published) is present in the list of available data sources, as shown in following screenshot:



# 5

## Creating Reports Using Pentaho Interactive Reporting

In this chapter, we will cover the following topics:

- ▶ Creating a simple interactive report
- ▶ Editing an existing report
- ▶ Adding groups and totals to reports
- ▶ Changing the labels in an interactive report
- ▶ Reorganizing table columns
- ▶ Adding filters to limit a report's output
- ▶ Adding prompts to get user input
- ▶ Exporting reports in the Excel or PDF format

### Introduction

After we spent quite a bit of time becoming familiar with the metadata editor, it is time to use the metadata domain model that we defined earlier and build a very simple tabular report. As we will see in this chapter's recipes, the very first and immediate advantage of using a metadata domain model to build a report is the more business-user-oriented approach because no SQL knowledge is needed to build the query and all of the formatting aspects are already covered in the definition of the metadata domain model. Therefore, everything is easier, especially for the average business user who does not have any particular technical background.

Pentaho interactive reporting is a web tool, available only in Pentaho Enterprise Edition (EE), which lets the user build a very simple but effective tabular report directly from inside Pentaho User Console. For anyone using Pentaho BA Server CE, it is possible to design a tabular report interactively directly from the web interface by using the old **WAQR (Web Ad-Hoc Query Reporting)** or the newest Saiku Reporting. Whatever tool we choose, Pentaho metadata domain models and Business View categories are exclusively used to define the report so that the user is not concerned about writing SQL queries to get the data from the database.

Using Pentaho metadata domain models gives an easier way to build reports by using a simple wizard to select and position the fields in the report area. Therefore, it is a high-level and easier approach to designing simple summary reports. Remember that we can only design tabular reports with these kinds of interactive tools, but they are good enough for most daily needs. If we need something more advanced, we need to use Pentaho Report Designer (this tool will be illustrated in *Chapter 7, Creating Reports Using Pentaho Report Designer*, with an entire set of recipes devoted to its usage).

The recipes in this chapter will take us through all of the basics of Pentaho interactive reporting, the tool we can find in the Enterprise Edition version of Pentaho BA Server, by helping us gain a full understanding of anything required to design a tabular report.

The recipes in this chapter assume that we are able to successfully log in to Pentaho User Console as a user who is either a member of a role with create operational permission assigned or is member of the administrator role.

If we want to use demo users and we want to log in using a Pentaho administrator, we can use as username and password for the demo administrator **admin/password**. This user is the new Pentaho demo administrator after the famous user **joe** (the Pentaho recognized administrator until Pentaho 4.8) has been dismissed starting from Version 5.

## Creating a simple interactive report

Let's start with a recipe that shows how to start designing a new report using Pentaho interactive report. In the next examples, we will continuously refine the report so that we can discover all of the main features that make this tool simple to use.

### How to do it...

To design a new report using Pentaho interactive reporting, we must perform the following steps:

1. Start **Pentaho Interactive Report** from the Pentaho User Console.
2. Select the metadata data source we want to use. In the case of our example, we will select the data source we have used so far, **Cookbook Samples**.

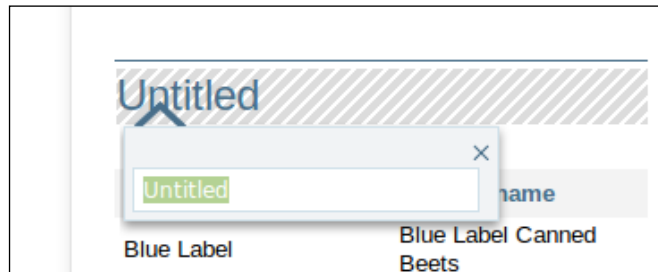
3. From the **General** tab to our left, select the template we want to be applied to the report. Let's imagine we choose the template named **Indented – Jade** (feel free to choose a different one if you want). The layout of the empty working area on the right immediately changes according to the template.
4. In the **Data** tab to the left, we have the set of Business Views for the selected data source and their related attributes. To position the fields on the report canvas, drag the fields we want to the report canvas to our right. While dragging, a vertical blue line indicates the relative position of the field we are going to drop on the table.
5. From **Product Category**, select the **Brand Name** and **Product Name** fields and drag them in the given order to the working area.
6. From **Store category**, select the **City** field. Drag-and-drop it to the right of the **Product Name** field.
7. From the **Sales fact** category, select the **Store sales** and **Unit sales** attributes. Drag-and-drop them respectively to the **City** attribute to the right.
8. As soon as any of the fields are dropped to the working area, the **Auto-Refresh** feature lets the report execute. This will give us a view of how the report will show at this point in development, as shown in following screenshot:

The screenshot shows a software interface for editing an interactive report. On the left, there is a 'Data' tab with a tree view of available fields for 'Cookbook Sample'. The 'Customer' category is expanded, showing fields like Account num, Lname, Fname, Mi, Address1-4, City, State province, Postal code, Country, Phone1-2, Birthdate, Marital status, Yearly income, Gender, Total children, Num children at home, Education, Date acct opened, Member card, Occupation, Houseowner, Num cars owned, and Fullname. The main area displays the report canvas with the title 'Interactive Report Cookbook Sample'. A table is rendered with the following data:

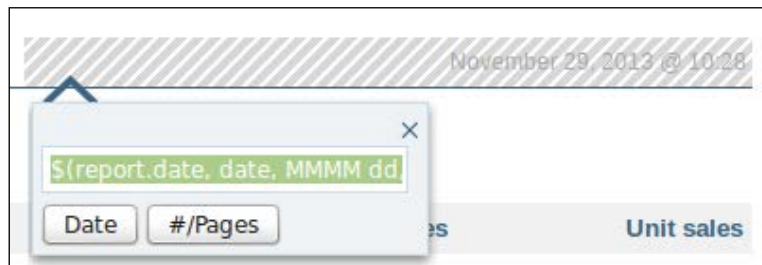
Brand name	Product name	Store city	Store sales
ADJ	ADJ Rosy Sunglasses	Acapulco	60,72 \$
ADJ	ADJ Rosy Sunglasses	Bellingham	5,52 \$
ADJ	ADJ Rosy Sunglasses	Beverly Hills	74,52 \$
ADJ	ADJ Rosy Sunglasses	Bremerton	55,20 \$
ADJ	ADJ Rosy Sunglasses	Camacho	52,44 \$
ADJ	ADJ Rosy Sunglasses	Guadalajara	2,76 \$
ADJ	ADJ Rosy Sunglasses	Hidalgo	80,04 \$
ADJ	ADJ Rosy Sunglasses	Los Angeles	69,00 \$
ADJ	ADJ Rosy Sunglasses	Merida	77,28 \$
ADJ	ADJ Rosy Sunglasses	Mexico City	5,52 \$
ADJ	ADJ Rosy Sunglasses	Orizaba	60,72 \$
ADJ	ADJ Rosy Sunglasses	Portland	82,80 \$
ADJ	ADJ Rosy Sunglasses	Salem	60,72 \$
ADJ	ADJ Rosy Sunglasses	San Andres	91,08 \$



- Now let's give an appropriate name to the report. Double-click on the **Untitled** label in the report header. An inline editor field will appear, as shown in the following screenshot. Type in the new report name (for example, *Interactive Report Cookbook Sample*) and press *Enter*. The inline edit dialog box closes and the new report title is displayed.



- On the upper-right corner of our report, in the **Report Header** section, we can see a field with the present day's date and time. The date and time has a default format, but if we are not satisfied, we can change it. To do this, double-click on the date-time label. The inline editor will appear as shown in the following screenshot:



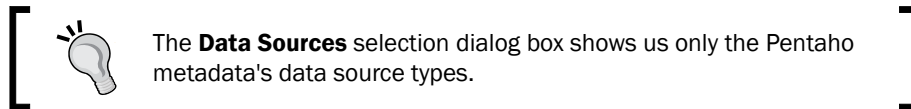
- As we can see, the actual value is displayed by means of the function `$(report.date, date, MMMM dd, yyyy @ hh:mm)`. The date-time formatter is the last parameter on the right, `MMMM dd, yyyy @ hh:mm`.
- Imagine that we want to display the date and time through this format: `29/11/2013 22:00`. Go to the last parameter on the right and change the formula to this new one: `$(report.date, date, dd/MM/yyyy hh:mm)`.
- We are almost there with our first simple report. Size it appropriately by setting a valid page layout (landscape or portrait) and format (A4, letter, or other).
- Click on the **Save As...** button on the **Opened** perspective's toolbar. The **Save As...** dialog box opens. Save the report in our usual Cookbook's exercises directory by setting the target location in Pentaho Solution (for example, `/Public/Pentaho BA Cookbook/Interactive Reports`). Type in the name of the report (`Interactive Rpt Sample 1`) and then click on the **Save** button to confirm.

## How it works...

To start Pentaho interactive reports from the Pentaho User Console, we can follow any one of these two ways:

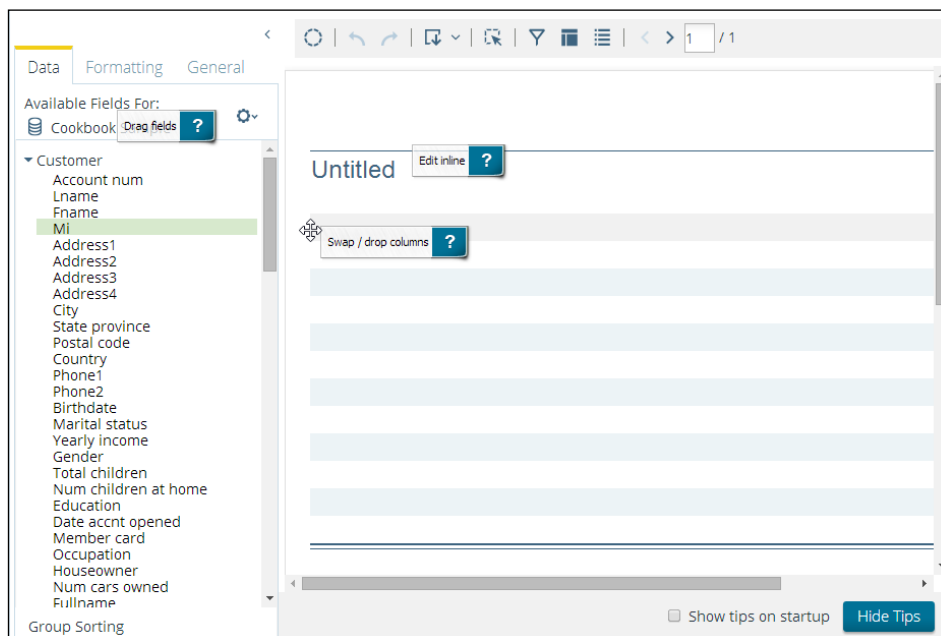
- ▶ If we are in **Home Perspective**, click on the big blue button, **Create New**, and then select **Interactive Reporting**
- ▶ From any perspective, from the **File** menu, navigate to **New | Interactive Report**

Because it is newly opened content, the **Interactive Report** designer opens it in the **Opened** perspective. As soon as we started defining a new interactive report, the first thing that it asked us to do was to select the Pentaho metadata data source that will feed our report. For this reason, as soon as we start a brand new Pentaho interactive report from scratch, the **Data Sources** dialog box opens and asks us to select a valid data source.

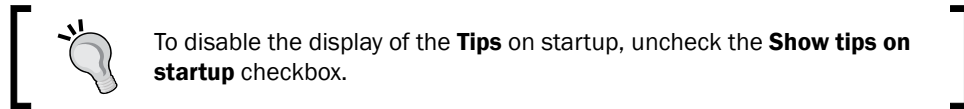


In our case, we can look for the sample metadata data source we created in the last chapter; we called it `Cookbook_samples`. Look for it in the list, select it, and click on the **OK** button.

The **Data Sources** selection dialog box closes and the Pentaho **Interactive Report** designer starts, as shown in following screenshot:



As we can see, the work area is full of tip icons to help us quickly get used to the main features of the tool. We can hide them by clicking on the **Hide Tips** button located at the lower-right corner of the screen.



To our left, we have a set of tabs:

- ▶ The **Data** tab contains the set of available **Business View** categories and their related fields
- ▶ The **Formatting** tab contains all of the available formatting options that let us style the object according to our taste
- ▶ The **General** tab lets us choose the template to apply to the report, gives some options related to the report's printing, and allows us to apply some other generic configuration flags

Let's start designing the report. Firstly, we want to style our report appropriately. To do this, select the **General** tab to the extreme right and select the template to apply to our report. With Pentaho interactive report designer, we can't define our report layout freely; we must work using predefined templates. This approach, which seems a bit limiting, is actually because the aim of the tool is to be easier to use and be very intuitive. If we want to design a more complex report layout, we need to use Pentaho report designer. As we can see in the following screenshot, we have a secret set of different templates available that we can choose from with various formats and colors:

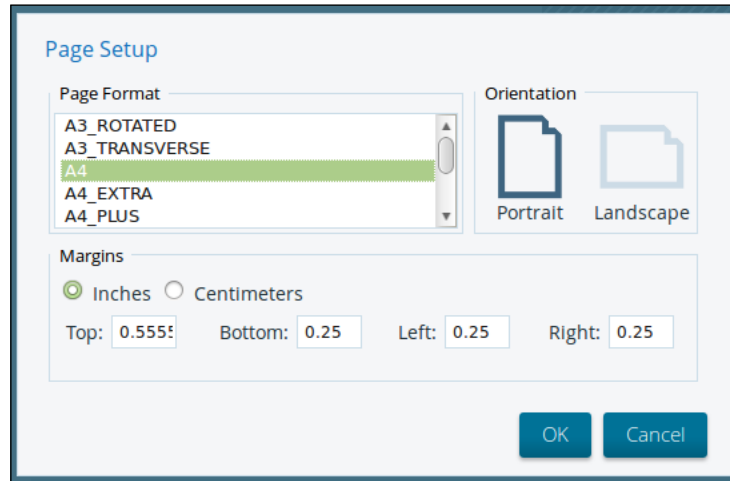


The screenshot shows how to select a different template for the report

We can go through all of the available templates by clicking on the two big arrows located on both sides of the template icon. The template selection is implicitly done as soon as we have the icon for that particular template displayed to us. However, in case you don't like any of the available templates, you can always define a new custom template and use it in Pentaho interactive reporting.

Another way to select a template is to click on the **Select...** button. On doing that, a dialog box is displayed and it shows us the set of available templates, one page at a time. Any one page contains a set of templates that share the same base template style. To apply a selected template, double-click on the template icon.


Something else we may be interested in is checking the page size so that we are able to print it the right way with the right size. To check the page settings, go to the **General** tab and click on the **Page Setup...** button.



The **Page Setup** dialog box opens as shown in the previous screenshot; here, we can check and (eventually) change the **Page Format**, **Orientation**, and **Margins** values. Let's imagine we want to change the page format from **LETTER** (the standard setting) to **A4**. To confirm the change and close the dialog box, click on the **OK** button; otherwise, click on **Cancel** to close the dialog box without saving the changes.

Now it is time to dispose our report columns in the right order by taking them from the set of available Business Views we have under the **Data** tab to our left. Simply drag-and-drop them in the correct order to the working area to our right. Make any changes to the title or to the date format, and then we are there. Remember that because the **Auto-Refresh** feature is enabled, if any change is made in the report layout, Pentaho will show us the resulting report.

Now, if we are satisfied with our work, it is time to save it. Click on the **Save As...** button on the **Opened** perspective's toolbar. The **Save As...** dialog box opens. Remember that if we want to save the report for our exclusive use, we can save it in **Home** subfolders. If we intend to share our report with other users in the team, save it in the **Public** section subfolders.

[  Remember that when you create new content, by default, you as the content's creator are its only owner. Therefore, to share it with our team, we first need to assign the right permissions. ]

To save the report, select the target location in Pentaho solution (for example, `/Public/Pentaho BA Cookbook/Interactive Reports`). Type in the name of the report (`Interactive Rpt Sample 1`) and then click on the **Save** button to confirm.

Our first interactive report has been successfully and easily developed and saved to Pentaho solution.

## See also

If you are unfamiliar with managing files' and folders' share permissions, go to the first chapter and look at the *Creating a simple interactive report* recipe.

## Editing an existing report

We just created our first interactive report. Let's see how to edit that report by changing, for instance, the formatting of some of the values in the report.

## How to do it...

To edit the report that we created, perform the following steps:

1. Go to the `/Public/Pentaho BA Cookbook/Interactive Reports` folder and select `Interactive Rpt Sample 1`, which we have created and saved in the previous recipe.
2. Select **Edit...** from the **File Actions** menu.
3. The **Interactive Report** designer starts and it opens the report in the **Opened** perspective.
4. Let's apply all of the necessary changes to the report, not necessarily only formatting changes. We can also add and remove columns from the report table, modify grouping, or similar things.
5. As soon as we are satisfied with the changes, we can confirm the new layout by clicking on the **Save** button on the **Opened** perspective's toolbar.

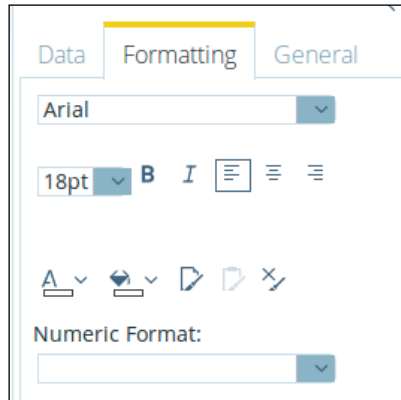


As soon as we click on the **Save** button from the toolbar in Pentaho User Console, the previous version of the object we modified is overwritten. Use the **Save As...** button if you want to save the modified object as a different copy without overwriting the original.

## How it works...

When we are editing our interactive report, we can also update anything on the existing report. Let's suppose we are going to edit the formatting properties to change some properties of the report's text.

To change the text formatting, select the **Formatting** tab from the left-hand-side tab set, as shown in the following screenshot:



That tab collects all of the formatting options we can assign to the text in the report.



Pentaho metadata already adds some styling capabilities to Business Tables fields. By using the options under the **Formatting** tab, we override the settings defined in the metadata domain model.

Select some column fields first and try to change the character type, the size, the weight (normal/bold), and the text alignment. We can also try to change the font's text color and the background's cell color. Because the **Auto-Refresh** feature is enabled, all changes are immediately visible on the report output.

Another thing we can do here is apply different formatting masks to numeric and date fields. To do this, select a numeric field from the report and select the right formatting mask from the combo box labelled **Numeric Format**.

As a last example, let's imagine we want to make the title bolder. First, select the title of the report; due to this action, the items under the **Formatting** tab get enabled. Select the bold character button in the toolbar. The report's title has immediately become bold.

As a sort of homework, we can try to apply other stylistic changes to the current report and see the results.

## See also

Not sure about how to create a report with Pentaho interactive report designer, or not familiar enough with creating a report with this tool? Have look at the *Creating a simple interactive report* recipe.

---

## Adding groups and totals to reports

Very often, it would be useful to group our data by specific keys composed of one or more fields. Then, maybe we would like to add a **Grand Total** line for our grouped measures. This recipe shows you how to define grouping by a particular columns' context and get the grand total related to some of the measure columns with respect to that group.

### How to do it...

The following steps show how easily we can add a group and a total to our report:

1. Go to the `/Public/Pentaho BA Cookbook/Interactive Reports` folder and select `Interactive Rpt Sample 2`, which we created and saved in the previous recipe.
2. Select **Edit...** from the **File Actions** menu.
3. The **Interactive Report** designer starts and opens the selected report in the **Opened** perspective.
4. From the **Store Category Business View**, select the **Store Type** attribute and drag-and-drop it to the working area immediately under the report title, a little bit above the columns header. As soon as we drop it, a new **Store Type** grouping level is created. Very easy!
5. Define the totals for the **Unit sales** and **Store sales** measures; from each column's contextual menu actions, go to **Summary | Sum**. Each time the values in the grouping level changes, we get the totals required.
6. As a last refinement to our report, add the label **TOTAL** at the bottom-left of the displayed group level total.
7. Double-click to the immediate left of the first group total value. The inline editor appears. Type in the string **TOTAL** and press the *Enter* key. The **TOTAL** label will appear immediately to the left of the group level total.

### How it works...

Creating a grouping level in Pentaho interactive report designer is a really easy task. The left-hand side of the screen contains all the Business View categories and their fields. To define a new grouping field, we can drag-and-drop the field we want to group to the working area immediately under the report title, a little bit above the columns header.





As soon as we are positioned by dragging-and-dropping a grouping field into a specific place of the report canvas, a horizontal bar will show us the relative location (relative to the other grouping levels we have already defined) where the field will be finally set. As for the column positioning, this is a nice feature that lets us have an exact idea about where our grouping level will be inserted.

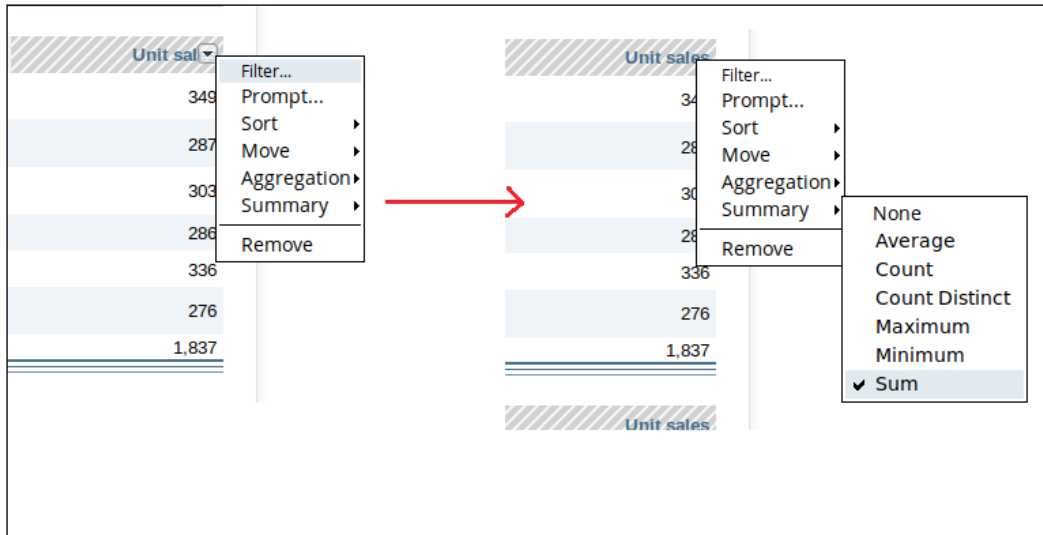
The following screenshot clearly shows us the horizontal line that indicates the relative position of the grouping level:

The screenshot displays the Pentaho Interactive Reporting interface. On the left, a 'Data' pane shows 'Available Fields For: Cookbook Sample'. Under the 'Store' category, 'Store type' is selected. The main report canvas shows a table titled 'Interactive Report Cookbook Sample' with columns: Brand name, Product name, Store city, Store sales, and Unit sales. A horizontal line is visible above the 'Store type' column header, indicating its relative position in the report's hierarchy.

Brand name	Product name	Store city	Store sales	Unit sales
Blue Label	Blue Label Canned Beets	Spokane	1,003.09 \$	283
Blue Label	Blue Label Canned String Beans	Hidalgo	796.29 \$	287
Blue Label	Blue Label Canned Tomatoes	Walla Walla	667.00 \$	300
Blue Label	Blue Label Canned Yams	Victoria	881.26 \$	317
Blue Label	Blue Label Chicken Soup	Vancouver	1,113.31 \$	349
Blue Label	Blue Label Creamed Corn	Tacoma	858.13 \$	287
Blue Label	Blue Label Noodle Soup	Yakima	588.00 \$	336
Blue Label	Blue Label Regular Ramen Soup	San Diego	576.09 \$	333
Blue Label	Blue Label Vegetable Soup	San Andres	706.99 \$	303
Jeffers	Jeffers Corn Puffs	Salem	757.90 \$	286

Defining an operation like the totals, for example, over one or more measure columns in the report, is very simple. Position the cursor to the right of the header field label related to the measure we want to apply the arithmetic operation to. A down arrow will appear, as shown to the left of the following screenshot. Click on the little down arrow; a contextual menu appears and shows us the possible actions we can perform on the field.

As an example, to make a summary total by group, from the contextual action menu, go to **Summary | Sum**. After the operation has been selected, a little checkmark will show that the operation has been applied to the measure, as shown to the right of the following screenshot:



There are a wide set of operations we can apply to numeric measures, for example, **Average**, **Count**, **Count Distinct**, **Maximum**, and **Minimum**. We can apply the same functions either as aggregations in a rolling fashion or as summarization functions. We suggest you practice with some easy exercises to make yourself comfortable with this tool and get a complete picture of all of its capabilities.

As a last refinement to our report, we want to show how to decorate our totals row by adding the label **TOTAL** to the bottom left of the total value displayed at the bottom end of every grouping level items. Go to the row after the last item displayed in the **Store City** column and double-click at the immediate left of the first group total value. The inline empty edit dialog box appears. Type in the `TOTAL` string in it and press the `Enter` key. The **TOTAL** label will appear to the immediate left of the grouping level's total value. Feel free to make it nicer by changing the character's properties according to your taste. In my case, I decided to make the font bolder and to align the **TOTAL** label to the right of its containing cell so that it is more close to the totals, as shown in following screenshot:

BEFORE THE ADDITION OF THE TOTAL LABEL				
Brand name	Product name	Store city	Store sales	Unit sales
Blue Label	Blue Label Chicken Soup	Vancouver	1,113.31 \$	349
Blue Label	Blue Label Creamed Corn	Tacoma	858.13 \$	287
Blue Label	Blue Label Vegetable Soup	San Andres	705.99 \$	303
Jeffers	Jeffers Corn Puffs	Salem	757.90 \$	286
Jeffers	Jeffers Oatmeal	Hidalgo	517.44 \$	336
Washington	Washington Orange Juice	Merida	714.84 \$	276
			4,667.61 \$	1,837

AFTER THE ADDITION OF THE TOTAL LABEL				
Brand name	Product name	Store city	Store sales	Unit sales
Blue Label	Blue Label Chicken Soup	Vancouver	1,113.31 \$	349
Blue Label	Blue Label Creamed Corn	Tacoma	858.13 \$	287
Blue Label	Blue Label Vegetable Soup	San Andres	705.99 \$	303
Jeffers	Jeffers Corn Puffs	Salem	757.90 \$	286
Jeffers	Jeffers Oatmeal	Hidalgo	517.44 \$	336
Washington	Washington Orange Juice	Merida	714.84 \$	276
			<b>TOTAL</b>	4,667.61 \$
				1,837

**See also**

Look at how to change some other interesting aspects of your report either in the *Changing the labels in an interactive report* or *Reorganizing table columns* recipe.

---

## Changing the labels in an interactive report

With these first steps, we obtained our first reports designed using the metadata defined in the previous chapter's recipes. Now we are going to show you how to change some of the basics of our report, for example, the name of the labels in our report.

### How to do it...

Let's imagine we want to change the grouping label from **Store Type** to **Store Category**. To do this, we must perform the following steps:

1. Go to the `/Public/Pentaho BA Cookbook/Interactive Reports` folder and select `Interactive Rpt Sample 3`, which we created and saved in the previous recipe.
2. Select **Edit...** from the **File Actions** menu.
3. The **Interactive Report** designer starts and it opens the report in the **Opened** perspective.
4. Double-click on one of the **Store Type** grouping labels (it does not matter on which of the labels). The inline editor will appear populated with the value of the original grouping label.
5. Type in the new label's value, `Store Category`, and press the *Enter* key. The report will immediately be refreshed and the new value of the grouping label will be displayed immediately.
6. Click on the **Save As...** button on the **Opened** perspective's toolbar. The **Save As...** dialog box opens.
7. Select the `/Public/Pentaho BA Cookbook/Interactive Reports` target folder. Type the name of the report (`Interactive Rpt Sample 4`) and then click on the **Save** button to confirm.
8. The new report is immediately visible in Pentaho solution.

### How it works...

Changing report labels is quite an interesting thing to know about because the interactive report designer considers the default name of the metadata field as the label's value for that field. However, often we are not satisfied with this and want to make them clearer.

Doing this is very easy; just double-click on the label we want to customize. The inline editor will appear and let us type the value of the new label. Press the *Enter* key and that's it—we have changed our label's value to the newer one we just typed.

## Reorganizing table columns

An interesting thing to know is that we can reorganize the columns by changing their relative positions or resizing them. Let's see how we can reorganize a table's columns.

### How to do it...

If we want to change the relative position of columns, we must perform the following steps:

1. Go to the `/Public/Pentaho BA Cookbook/Interactive Reports` folder and select `Interactive Rpt Sample 4`, which we created and saved in the previous recipe.
2. Select **Edit...** from the **File Actions** menu.
3. The **Interactive Report** designer starts and it opens the report in the **Opened** perspective.
4. Switch the position of the **Unit Sales** column with that of **Store Sales** by dragging-and-dropping one to the other.
5. Click on the **Save As...** button on the **Opened** perspective's toolbar. The **Save As...** dialog box opens.
6. Select the `/Public/Pentaho BA Cookbook/Interactive Reports` target folder. Type in the name of the report (`Interactive Rpt Sample 5`) and then click on the **Save** button to confirm.
7. The new report is immediately visible in Pentaho solution.

If we want to remove a column either from a grouping level or from the table's columns, we must perform the following steps:

1. Go to the `/Public/Pentaho BA Cookbook/Interactive Reports` folder and select `Interactive Rpt Sample 5`, which we created and saved in the previous recipe.
2. Select **Edit...** from the **File Actions** menu.
3. The **Interactive Report** designer starts and opens the report in the **Opened** perspective.
4. From the table columns' layout, remove **Unit sales** by dragging-and-dropping it over the Trash icon.
5. Click on the **Save As...** button on the **Opened** perspective's toolbar. The **Save As...** dialog box opens.
6. Select the `/Public/Pentaho BA Cookbook/Interactive Reports` target folder. Type the name of the report (`Interactive Rpt Sample 6`) and then click on the **Save** button to confirm.
7. The new report is immediately visible in Pentaho solution.

If we want to move a table's column to become a grouping level, we must perform the following steps:

1. Go to the /Public/Pentaho BA Cookbook/Interactive Reports folder and select *Interactive Rpt Sample 6*, which we created and saved in the previous recipe.
2. Select **Edit...** from the **File Actions** menu.
3. The **Interactive Report** designer starts and opens the report in the **Opened** perspective.
4. Move the **Brand name** field from the table columns' layout to become a grouping level immediately below the **Store type** grouping level field.



Interactive Report Cookbook Sample	
Store Category: Deluxe Supermarket ^	
Brand name: ADJ ^	
Product name	Store city
ADJ Rosy Sunglasses	Salem

5. Click on the **Save As...** button on the **Opened** perspective's toolbar. The **Save As...** dialog box opens.
6. Select the /Public/Pentaho BA Cookbook/Interactive Reports target folder. Type in the name of the report (*Interactive Rpt Sample 7*) and then click on the **Save** button to confirm.
7. The new report is immediately visible in Pentaho solution.

## How it works...

This recipe goes through some of the main topics regarding the management of report columns.

Let's start by looking at reordering columns; suppose we want to switch the position of the two measures by putting the **Unit sales** column before the **Store sales** column. Move the cursor over the **Store sales** column in any position, not necessarily over the column's header. As soon as we move the cursor over the column, its background assumes a light gray color; this indicates the column we are going to act on. Drag the **Stores Sales** column over to the right of the **Unit sales** column. While we are dragging, a vertical line appears to the right of the **Units sales** column to indicate the new position where we can drop the column. Drop the **Store sales** column in its new position. The report will immediately be refreshed by showing its results with the new column layout.

Another interesting topic is how to remove a column, either from a grouping level or from the table's columns. We must perform the steps outlined here. Let's suppose that we want to remove **Unit sales** from the table columns' layout. Move the cursor over the **Unit sales** column in any position, not necessarily over the column's header. As soon as we move the cursor over the column, its background assumes a light gray color. Drag the column over the Trash icon located at the bottom-right corner of the **Interactive Report** designer tool. As soon as we are over the Trash icon, drop the **Unit sales** column and it will be removed. The report will immediately reload, and show its results with the new column layout. The same actions must be taken to remove a field from a grouping level.

The last topic will focus on converting a table's column field to a grouping level field. Let's suppose we want to remove the **Brand name** field from the table columns' layout and use it as a grouping level after the **Store type** level field. That means we will have the data grouped by **Store type, Brand name**. Move the cursor over the **Brand name** column in any position, not necessarily over the column's header. As soon as we move the cursor over the column, its background assumes a light gray color; this indicates the column we are going to act on. Drag the column above the table's columns header but immediately below the **Store type** grouping level. While we are dragging the column above the table columns headers, a horizontal line appears to indicate the next position where we can release the column. As soon as you have reached the position above the columns header but immediately below the **Store type** grouping level (the horizontal positioning line will help us in finding the right position), drop the **Brand name** column to its new position. The report will immediately reload by showing its results with the new grouping levels and columns layout.

## Adding filters to limit a report's output

Until now, we have seen a set of recipes that will make us confident about creating a simple report using Pentaho interactive report. Now we are going to present two recipes to help us become familiar with different ways of limiting the report output. This recipe shows you how to apply a filter to the report.

### How to do it...

A filter is a sort of predefined condition we put at design time to limit or point our report output to a specific set of records we are looking for. While executing the report, the user is not able to change the filter. It is a sort of WHERE condition for anyone who is familiar with the SQL language.

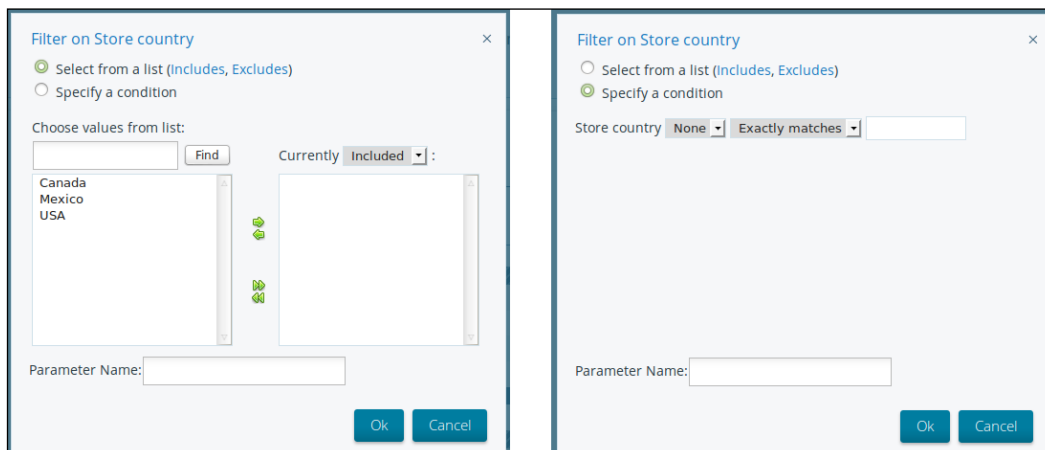
1. Go to the `/Public/Pentaho BA Cookbook/Interactive Reports` folder and select `Interactive Rpt Sample 7`, which we created and saved in the previous recipe.
2. Select **Edit...** from the **File Actions** menu.
3. The **Interactive Report** designer starts and opens the report in the **Opened** perspective.

4. Let's suppose we want to limit our report output to consider only records that are related to the stores located in the **USA**.
5. Click on the **Filter** icon in the **Interactive Report** toolbar.
6. From the **Store Business View** category, select the **Store country** field and drag-and-drop it to the filter area. Then specify the **Store country** values that we want to filter on.
7. Click on **OK** to close the dialog box and apply the filter. Click on **Cancel** to close the dialog box and lose all the work that you have performed so far.
8. Click on the **Save As...** button on the **Opened** perspective's toolbar. The **Save As...** dialog box opens.
9. Select the `/Public/Pentaho BA Cookbook/Interactive Reports` target folder. Type in the name of the report (`Interactive Rpt Sample 8`) and then click on the **Save** button to confirm.
10. The new report is immediately visible in Pentaho solution.

## How it works...

Applying filters on an interactive report is a very easy and straightforward process. Click on the **Filter** icon in the **Interactive Report** toolbar. The filter area rectangle appears immediately below the **Interactive Report** toolbar. Let us suppose we want to define a filter on the **Store country** field.

From the **Store Business View** category, select the **Store country** field and drag-and-drop it to the filter area we have already discussed. As soon as we drag the field over the filter area, the background color for the filter area becomes green. Immediately after we drop the field, the **Filter on** dialog box appears. The **Filter On** dialog box is shown to the left of the following screenshot:





On the upper side of the dialog box, by means of the two radio buttons, we can choose how to specify the filter values. We can get it either from a list (the **Select from a list** option) or by manually specifying a condition (**Specify a condition**, the default).

If we consider selecting the value from a list by choosing the **Select from list** option, we can set a list of values and a condition to consider those values included or excluded.

If we select **Specify a condition**, we can manually set a comparison expression by choosing the comparison operator from the list (see it to the right of the previous screenshot). In our case, we select **Specify a condition** and set **Store Country Exactly Matches** (the default from the list) and then type in **USA**.

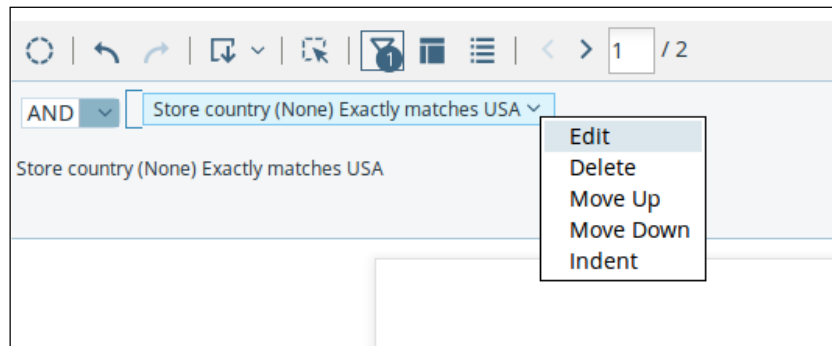
Click on **OK** to close the dialog box and apply the filter. Click on **Cancel** to close the dialog box and lose all the work you have done so far. As the dialog box closes, the filter is made explicitly visible in the filter area above and the report will immediately reload by showing its results according to the newly applied filter.

## There's more...

As soon as we have a filter created, we can further change it by using a contextual menu available in the field. Let's see how to do this.

### Changing or removing the filter

To change or remove a filter, click on the down arrow located to the right of the condition we want to change, as shown in the following screenshot:



That action will activate a contextual menu and give us the following options:

- ▶ **Edit:** This will show us the filter dialog box we saw in the recipe and give us the ability to remove the filter
- ▶ **Delete:** This removes the selected filter condition
- ▶ **Move Up/Down:** This moves the condition up or down in case of multiple WHERE conditions

## See also

If we want to specify the values for our filters at runtime, we need prompts. See how to define prompts in the next recipe, *Adding prompts to get user input*.

## Adding prompts to get user input

Prompts are the second way to limit records that are going out from the report. The idea is the same as that for filters, but they differ because they give the user the ability to dynamically choose the filter values at runtime before running the report.

### How to do it...

To apply a prompt to our report, we must perform the following steps:

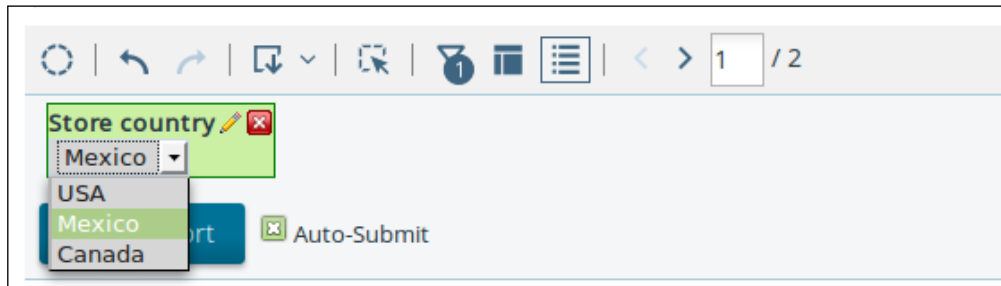
1. Go to the `/Public/Pentaho BA Cookbook/Interactive Reports` folder and select `Interactive Rpt Sample 7`.
2. Select **Edit...** from the **File Actions** menu.
3. The **Interactive Report** designer starts and opens the report in the **Opened** perspective.
4. Let's suppose we want to replicate the same use case as the previous recipe but by using prompts. Click on the prompt toolbar icon. The prompt area appears immediately below the **Interactive Report** toolbar.
5. Select the **Store country** field from the **Store Business View** category and drag it to the prompt area. The prompt on that field is immediately available.
6. Click on the **Save As...** button on the **Opened** perspective's toolbar. The **Save As...** dialog box opens.
7. Select the `/Public/Pentaho BA Cookbook/Interactive Reports` target folder. Type in the name of the report (`Interactive Rpt Sample 8`) and then click on the **Save** button to confirm.
8. The new report is immediately visible in Pentaho solution.

### How it works...

As we said in the introduction of this recipe, prompts are a way for the user to set filter condition values dynamically at runtime, before the report execution is going to be submitted.

Let's suppose we want to set prompts on the report to let the user select the country values to filter on. Click on the prompt toolbar icon. The prompt area rectangle appears immediately below the **Interactive Report** toolbar.

Now, from the **Store Business View** category, select the **Store country** field and drag-and-drop it to the prompt area we have uncovered so far. As soon as we drag the **Store country** field over the prompt area, the prompt area assumes a green background. As soon as we drop the field over the prompt area, the prompt field immediately becomes available as a combo box from which we can choose the values.



Try to change the selected **Store country** value and we will see the records in our report changing.

## There's more...

Let's see how we can remove all of the unneeded prompts very easily from the prompts area. Another interesting trick is how to choose the prompt format and activate different processing logic on the selected prompts.

### Removing unnecessary report prompts

Removing unnecessary prompts from the prompt area is an easy task. It is only a matter of clicking on the little red button with the white **x** on it, located in the upper-right corner of the prompt that is to be removed. Nothing else is required, and we are done!

### Changing the prompt's appearance and behavior

On clicking on the little pencil icon located in the upper-right corner of the prompt, the **Prompt properties** dialog box opens, as shown in the following screenshot. There, we can change some of the prompt properties, for example, the appearance and default values.

We can change the prompt's appearance by choosing from a combo box (the default one) to a set of radio buttons, a set of checkboxes, a list with multiple choices, or other types depending on our needs.

In case we are dealing with a list of values to choose from, another interesting point is that we can set the default value for our prompt. For example, if we set the prompt to appear as a combo box because it is a multivalued list, we can choose which value of the prompt combo-box's list is to be used as the selected value.

We have two different options when setting the list default value, as follows:

- ▶ We can choose any value in that list by selecting the **Use First Value** radio button under **Initially Selected**
- ▶ We can choose to use the first value by selecting the **Specify** radio button under **Initially Selected** and then selecting the value to use as the default value from the **Label** and **Value** drop-down lists

## See also

If we want to specify fixed filter conditions, see how to define static filters in the previous recipe, *Adding filters to limit a report's output*.

## Exporting reports in the Excel or PDF format

After we have finalized the design of our interactive report, let's see how to export the report in one of the export formats available (PDF, Excel, other).

### How to do it...

To export your report to one of the available formats, perform the following steps:

1. Go to the `/Public/Pentaho BA Cookbook/Interactive Reports` folder and select `Interactive Rpt Sample 9`.
2. Double-click on the report.
3. The **Interactive Report** designer starts and opens the report in view mode in the **Opened** perspective.
4. Click on the **Export** icon button in the **Interactive Report** toolbar.
5. A combo-box with the complete set of available export types appears. It is possible to export reports in PDF, Excel, Word, CSV, and other formats.
6. Select the target format, for example, PDF.
7. Pentaho opens the viewer for that format in a new browser window and displays the exported report.

### How it works...

Exporting a report in a portable format is a good way to distribute the report to our users. The task is as easy as drinking coffee. Just click on the **Export** icon button in the interactive report toolbar and select the exporting format from the formats combo-box list. All of the main portable formats are available for our convenience: PDF, Excel, CSV, and other. As soon as we select the exporting format, Pentaho opens the viewer for that format and allows us to see the export results. In case a convenient viewer is not available, the **Save As...** dialog box will pop up, which allows us to directly save the exported report to disk.

# 6

## Creating Analysis Reports

In this chapter, we will cover the following recipes:

- ▶ Creating and publishing a Mondrian schema
- ▶ Creating a new analysis report from scratch
- ▶ Adding subtotals to rows' categories
- ▶ Adding graphical indicators to a table's cells
- ▶ Changing a column's sort order
- ▶ Adding a simple calculated measure
- ▶ Creating visualizations with Pentaho Analyzer
- ▶ Exporting reports in the Excel or PDF format
- ▶ Creating an analysis report using Saiku

### Introduction

Here, we are going to show you everything about one of the most powerful features of our Pentaho BA Server: use of the OLAP's power to slice and dice data. **Online Analytical Processing (OLAP)** is a functionality that gives the user the power to analyze data from multiple perspectives interactively and dynamically because of the ability to combine new analysis context, drill up and down into the data, and continuously create new views over the analyzed data.

Usually, OLAP is implemented through a set of tools that enable this highly interactive level of data analysis. The tools are typically categorized into three different types:

- ▶ **Relational Online Analytical Processing (ROLAP):** These systems typically use a relational database for the storage of their data and generate SQL queries based on users' MDX queries. Usually, the user builds a matrix where he/she defines an analysis context by disposing dimensions and measures over a matrix's rows and columns. Mondrian, the OLAP server bundled in the Pentaho platform, is a typical example of a ROLAP engine.
- ▶ **Multidimensional Analytical Processing (MOLAP):** These systems usually store the data in a proprietary and optimized datastore, where each aggregation is precomputed at the time of loading the data into the datastore. This different kind of approach is the main differentiator between MOLAP and ROLAP systems.
- ▶ **Hybrid Online Analytical Processing (HOLAP):** These systems take a hybrid approach by giving the business analyst the ability to decide which portion of the data has to be put into the ROLAP store and which portion should be put in the MOLAP datastore.

Any OLAP system is built around the concept of a cube or hypercube. A cube is a multidimensional structure composed of numeric facts called measures and a set of perspectives of the analysis called dimensions.

This chapter will explain a set of recipes that help a cube to interact by using Pentaho Analyzer, the OLAP client tool available only in Pentaho EE. As a last recipe, we will make an OLAP analysis by using Saiku, the OLAP client available for both the versions of Pentaho. The interesting thing to note here is that the Mondrian schema is a universal metadata descriptor that works with any of the OLAP client tools we use.

As a starter recipe and for your reference, we will take you through a brief definition of a Mondrian schema just to recap the main concepts regarding the definition of a Mondrian cube. Mondrian is the ROLAP server written by Julian Hyde and distributed on the Pentaho platform.

Pentaho Schema Workbench is a client development tool. In the Pentaho EE version, the tool is already present in the Pentaho distribution in the `<pentaho_ee_home>/design-tools/schema-workbench` directory. In the Pentaho CE version, it has to be downloaded separately from <http://community.pentaho.com> and installed wherever desired on the local filesystem.

Once Pentaho Schema Workbench is installed, we need to check whether our Java environment is configured properly. To do this, check whether the `JAVA_HOME` environment variable is set appropriately. Even if the schema workbench, while starting up, tries to guess the value of the `JAVA_HOME` environment variable from the system, it is always a good rule of thumb to set the `JAVA_HOME` environment variable to be sure that everything works as required.



Having the `JAVA_HOME` variable properly set and referring all of the Java paths to this environment variable is a good way to keep the system clean and to easily manage more than one version of the JDK at the same time on the same system.

The recipes in this chapter (with the exception of the first one, *Creating and publishing a Mondrian schema*) are based on the assumption that we are able to successfully log in to the Pentaho User Console. To do this, we are free to use any user and not necessarily a demo user.

If we want to use demo users, remember that we can use the following logins to access our system:

- ▶ `admin/password`: This is Pentaho's new demo administrator after the famous user, `joe` (the Pentaho-recognized administrator until Pentaho 4.8), was dismissed in this new version.
- ▶ `suzy/password`: This is another simple user that we can use to access the system. Because `suzy` is not a member of the administrator role, it is useful to see what is different in case a user that is not an administrator tries to use the system.

## Creating and publishing a Mondrian schema

As the first step in our walkthrough of Pentaho Analyzer, we start by giving a quick introduction to a Mondrian schema definition. To start using OLAP functionalities in Pentaho, we first need to define a *Mondrian schema*. It is an XML file that contains the definition of all the multidimensional entities, for example, cubes, dimensions, and measures that relate to a specific business area. The definition of a Mondrian schema is a preliminary and mandatory operation to access multidimensional analysis capabilities. This recipe will show how to create a simple Mondrian schema by using the developer tool called Pentaho Schema Workbench, trying to get you into some basic concepts. We do not intend to provide a detailed guide about how to design a Mondrian schema, but just a brief review of the basic steps and concepts needed to design it.

### Getting ready

For this recipe, check the following conditions:

- ▶ Pentaho Schema Workbench must be started
- ▶ To publish the schema, a user role with the **Publish Content** operational permissions granted must be available



## How to do it...

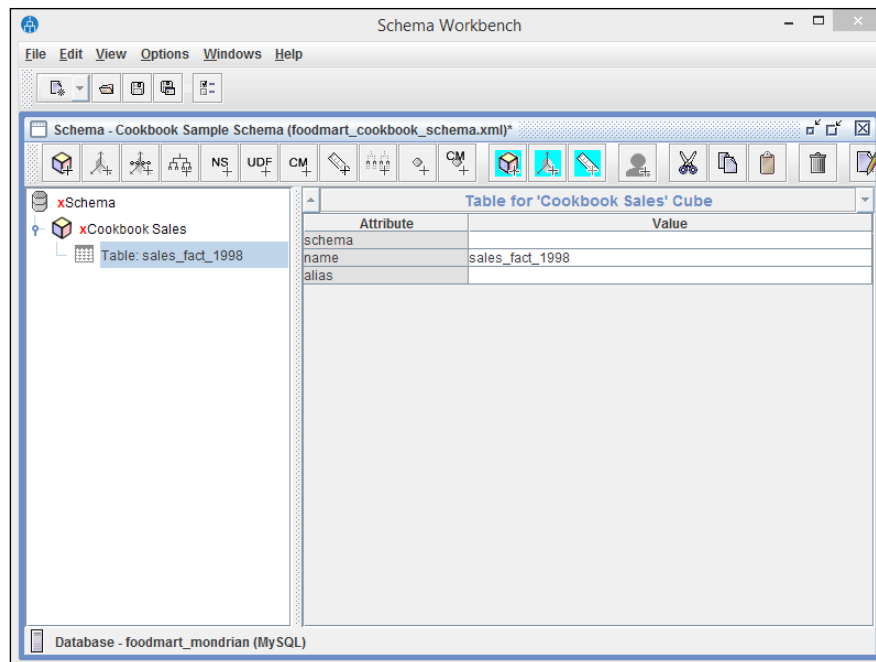
Let's imagine that we want to design a simple schema file in order to provide a simple cube to analyze company sales. To create a simple schema file, we must perform the following steps:

1. Create a new Mondrian schema either by clicking the new icon from the toolbar and then selecting **New** from the drop-down menu or by selecting the **Schema** option under the **New** menu in **File**.
2. Select the schema icon from the tree view in the left. The schema properties appear in the right-hand side of the tree view in the property editor. Look for the **Name** property and type the name of the schema as `Cookbook Sample Schema`.



Every time we set a property value for any schema object in the property editor by typing the value into the property editor field, remember to click outside of the property line we are editing to be sure that Pentaho Schema Designer has recognized your change. If we don't do this and save the schema file, sometimes the last change made is not saved.

3. Save the schema file as `foodmart_cookbook_schema.xml` either by clicking on the save as button in the toolbar or by selecting the **Save As...** option under the **File** menu.
4. Select the schema icon. Create a new Mondrian cube either by selecting the **Add Cube** entry from the contextual menu or by clicking on the add cube icon from the toolbar.
5. Select the cube icon from the tree view in the left-hand side pane. The cube's properties appear at the right-hand side of the tree view in the property editor. Look for the **Name** property and type the name of the cube as `Cookbook Sales`.
6. Now that we have created the cube, we must associate a fact table with it. From the cube's contextual menu, select the **Add Table** item entry.
7. Select the table icon from the tree view in the left-hand side pane. The table properties appear at the right-hand side of the tree view in the property editor. Look for the **Name** property; type `sales_fact_1998` as the name of the fact table, as shown in the following screenshot:



8. Suppose that we now want to add a dimension named **Store** to our cube. Select the cube's icon and select **Add Dimension** from the contextual menu or click on the add dimension icon in the toolbar. A new icon representing the dimension we just added appears below the cube.
9. Expand the dimension icon to expose the default hierarchy, conventionally named **New Hierarchy 0**. First, we must define a table to map this hierarchy. Select the hierarchy icon and from the contextual menu, select **Add Table**.
10. The table's properties appear at the right-hand side of the tree view in the property editor. Look for the **Name** property; type `store` as the name of the fact table or select that table from the drop-down list.



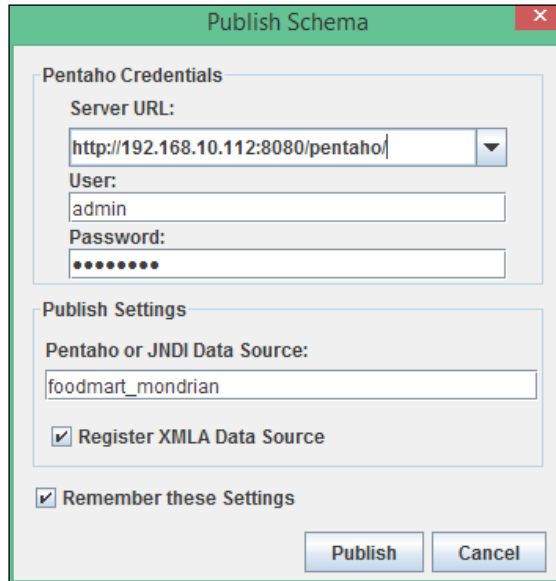
Every time we select a property value from a drop-down list, remember to click outside the drop-down property line to be sure that Pentaho Schema Designer has recognized the change. If we don't do this and save the schema file, sometimes the last change made is not saved.

11. Now we must properly configure the dimension. Select the dimension icon from the tree view in the left-hand side pane. The dimension's properties appear at the right-hand side of the tree view in the property editor. Look for the **Name** property; rename the dimension with its real name, `Store`. Then, look for the foreign key property and set its value to `store_id` by selecting the **foreign key** field from the drop-down list.
12. Select the hierarchy icon from the tree view in the left-hand side. The hierarchy properties appears at the right-hand side of the tree view in the property editor. Look for the **Name** property; rename the hierarchy as `default`. Then, look for the primary key property and set its value to `store_id` by selecting the **primary key** field from the drop-down list.
13. As the last operation to properly define our dimension's hierarchy, let's define the hierarchy levels. Select the hierarchy icon from the tree view and select **Add Level** from the contextual menu.
14. Once the level is added as a direct child of our dimension, select the level icon. Look for the **Name** property; rename the level with its real name, `Store Country`. Further, apply all the operations specified as follows:
  1. Look for the **column** property and set its value to `store_country` by selecting it from the drop-down list.
  2. Look for the **uniqueMembers** property and set its value to `checked`.
  3. Look for the **hideMemberIf** property and set the value to `Never`.
15. We saw how to define a dimension and configure its default hierarchy (steps 8 to 12) and then how to simply add the first level to our **Store** dimension's default hierarchy (steps 13 and 14). To define our store's default hierarchy completely, we would add three more levels: **Store State**, **Store City**, and **Store Name**. For brevity, we don't go down to add other levels or illustrate the complete addition of all of the required dimensions, hierarchies, and levels, but we can find the complete dimensional set in the sample given with the book.
16. Let's now try to add a simple measure to our cube. Select the cube icon and choose **Add Measure** from the contextual menu or click on the add measure icon from the toolbar. A new icon with a default name as **New Measure 0**, representing the measure we just added, appears after all the dimensions in the list.
17. Select the measure icon. Look for the **Name** property; rename the level with its real name, `Unit Sales`. Further, apply all the operations as follows:
  1. Look for the **column** property and set its value to `unit_sales` by selecting it from the drop-down list.
  2. Look for the **aggregator** property and set its value to **Sum** by selecting it from the drop-down list.

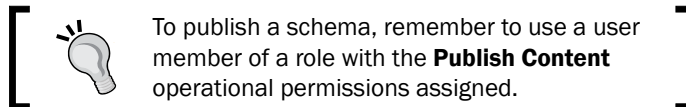
3. Look for the **datatype** property and set its value to **Numeric** by selecting it from the drop-down list.
  4. Look for the **formatter** property and type # , ##0 . ## as the formatter's value.
18. The example provided with the book contains a complete set of simple measures: **Store Cost, Store Sales, Sales Count, and Customer Count.**
19. As the last example, let's add a calculated measure to our cube. Calculated measures get their value as the result of a calculation between other measures. Select the cube icon and choose **Add Calculated Member** from the contextual menu or click on the add calculated member icon in the toolbar. A new icon with the default name as **New Calculated Member 0**, representing the measure we just added, appears after all the dimensions in the list.
20. Select the calculated member icon. Look for the **Name** property; rename the level with its real name `Profit`. Further, apply all the operations as follows:
1. Look for the **formula** property and type the formula value as `[Measures] . [Store Sales] - [Measures] . [Store Cost]`. Remember that the formula is a fragment of an **MDX** expression.
  2. Select the calculated member icon and choose the **Add Calculated Member** item from the contextual menu. Rename the calculated member property to `FORMAT String`. Look for the **value** property and type the value as `$#,##0.00`.
21. The example provided with the book contains a complete set of calculated members: **Profit, Profit Growth, and Profit last Period.** As an example, the next screenshot shows the configuration for the **Profit Growth** calculated member.

Calculated Member for 'Cookbook Sales' Cube	
Attribute	Value
name	Profit Growth
description	
caption	
dimension	Measures
hierarchy	
parent	
visible	<input checked="" type="checkbox"/>
formula   formulaElement...	<code>([Measures].[Profit] - [Measures].[Profit last Period]) / [Measures].[Profit last Period]</code>
formatString	

22. Now that our schema is ready, it's time to publish it to the Pentaho BA Server. From the Pentaho Schema Workbench menu, select **Publish...** under the **File** menu. The **Publish Schema** dialog box appears, as shown in the following screenshot:



23. Type the correct URL to the Pentaho server as `http://<ip_address>:<port>/pentaho` and fill in the username and password fields with the correct Pentaho user details.



24. In the **Pentaho or JNDI data source** field, type `foodmart_mondrian` as the name of an existing Pentaho JDBC data source; we created it in the *Creating a new JNDI JDBC data source* recipe in *Chapter 3, Defining BA Server Data Sources*. Then, check the **Register XML Datasource** checkbox to let the system automatically create an analysis data source under Pentaho name as the schema name.
25. Click on the **Publish** button. An information dialog will inform you about the success or failure of the publish operation.

## How it works...

Writing a Mondrian schema is the way to map an entire or a part of an enterprise data mart to a multidimensional analysis schema. A schema contains one or more cubes, zero or more shared dimensions (if any), and some security rules to apply proper permissions to the data underneath. From a practical point of view, the schema file is a sort of metadata file, available as an XML file, that is produced by a tool named Pentaho Schema Workbench.

As we saw, there are various operations involved in defining the Mondrian schema; we are going to briefly recap all of them as follows:

- ▶ Define a new schema.
- ▶ Add one or more cubes to the schema.
- ▶ Add one or more dimensions to the cube. In the case of dimensions, they could either be defined in the cube's body (this way they are tied to that specific cube) or at the schema level (in this case, we talk about shared dimensions because they can be used by any cube defined in the schema) and linked, wherever necessary, in the cube body.
- ▶ For each dimension defined in the cube or shared in the schema, we must configure the dimensions' hierarchies and levels. Every dimension must have one or more hierarchies, and any hierarchy must have one or more levels. The first hierarchy in any dimensions' hierarchies set is called the default dimension and is always automatically added by Pentaho Schema Workbench.
- ▶ Lastly, we must define all the required measures of the cube. The measures can be either simple measures, whose values are read by the target data mart's fact table, or calculated measures. Calculated measures are defined by applying MDX calculation expressions to other cube measures, obtaining new useful indicators. Any measure can have a formatter defined to apply styling-related formatting rules to the measure's value. Using a formatter, we can define grouping and rounding as well as set a background measure's color or apply graphical indicators based on conditions.

Pentaho Schema Workbench is a tool that helps to write a Mondrian schema; because the schema is an XML file, we can also write the schema manually. However, we must take care about the semantics and the mandatory nature of some of the elements' values of the XML schema we are going to write.

## There's more...

When writing a Mondrian schema, sometimes we need to reference tables and fields in our data mart. Pentaho Schema Workbench permits the configuration of a JDBC connection to inspect the target data mart and suggests references to these set of objects wherever required. Let's see how we can do this, but first, a note about how we can add a new JDBC driver to the list.

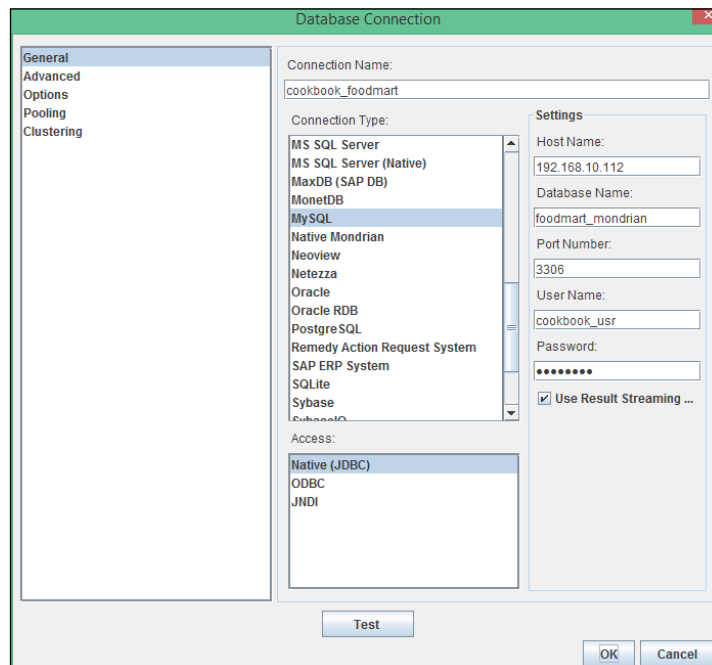
## Adding a JDBC driver

Even if the list of available databases is deep, not all the databases contained in the list have a related driver delivered in a bundle with the tool and ready to be used. Therefore, we need to manually deploy it in the right place. To deploy a JDBC driver for use with Pentaho Schema Workbench, we must copy the JAR file to the `<pentaho_metadata_editor_home>/drivers` directory. After we have copied the driver in this place, restart Pentaho Schema Workbench.

## Configuring a JDBC data source to help in the schema design

Pentaho Schema Workbench permits the configuration of a JDBC data source to be used in order to support schema creation. As shown during the schema configuration, there are many places in the Mondrian schema where, to properly fill specific property values, you are required to have knowledge about the target RDBMS data mart schema. After configuring the JDBC data source, Pentaho Schema Workbench lets the user choose values for the properties from a drop-down list built intelligently depending on the context of the target properties. To configure the JDBC data source, we can perform the following steps:

1. From the Pentaho Schema Workbench menu, choose the **Connection** option under the **Options...** menu.
2. The **Database Connection** dialog appears as shown in the following screenshot. Give a name to the connection, select the **Database Type** and the **Access** fields from their related lists and then configure the connection parameters as shown in the following screenshot:



3. Finally, click on the **Test** button to test the connection. If it works, click on the **OK** button to confirm the **JDBC Connection** properties and close the dialog.

## See also

- ▶ To recap how to define a Pentaho JDBC data source, go to *Chapter 3, Defining BA Server Data Sources* and read both the *Creating a new native JDBC data source* and *Creating a new JNDI JDBC data source* recipes. A good explanation on Mondrian schema XML elements can be found at <http://mondrian.pentaho.com/documentation/schema.php>. This document is about Mondrian 3.
- ▶ At the time of writing this book, Mondrian 4 is out in GA and it is time to gain some knowledge of it. A good source of information on this topic is the book *Mondrian in Action*, Julian Hyde, William D. Back, and Nicholas Goodman, Manning. Also, there is a nice blog article from my community friend Diethard Steiner that easily introduces you to the basics of the new Mondrian 4 schema definition, which can be found at <http://diethardsteiner.blogspot.it/2013/01/mondrian-4-get-ready.html>.

## Creating a new analysis report from scratch

Now that we have a Mondrian schema created and published, we are ready to build our first analysis report and use the power of Pentaho Analyzer. This recipe will show how to build a simple analysis from scratch using Pentaho Analyzer. Note that Pentaho Analyzer is a tool available only in the EE version of Pentaho.

### How to do it...

To create a new Pentaho analysis report, perform the following steps:

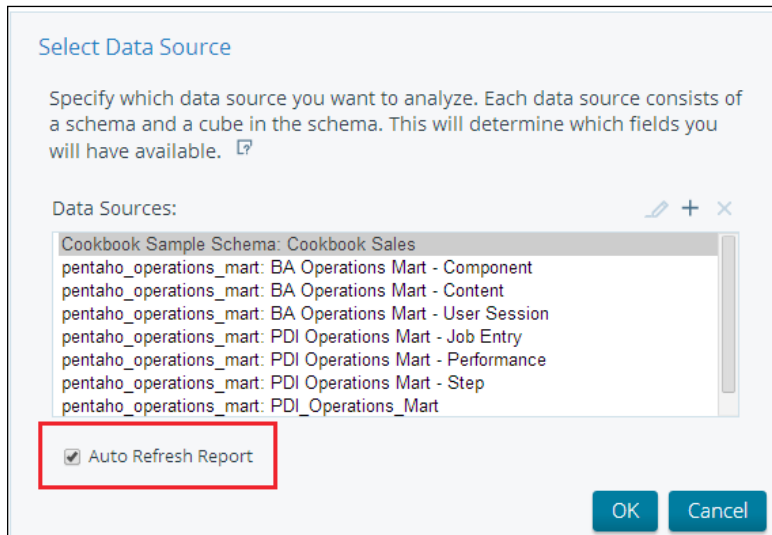
1. To create a new Pentaho Analysis report, we can either click on the **Create New** button in the **Home** perspective and then click on **Analysis Report**, or we can select the **Analysis Report** option under the **New** option under the **File** PUC menu.
2. The **Select Data Source** dialog box appears. Choose the analysis data source named **Cookbook Sample Schema** and click on **OK**.
3. Let's build a simple analysis report. From the **Product** dimension, drag-and-drop the **Product Department** level to the **Rows Fields** layout zone. From the **Time** dimension, get the **Quarter** level and drag-and-drop it to the **Column Fields** layout zone.
4. From the **Measures** dimension, drag-and-drop the **Store Sales** measure to the **Measures Fields** layout zone.
5. Because the **Auto Refresh** option is enabled, the report will refresh every time we drag-and-drop something in any of the layout zones.



6. Temporarily, go to the **Browse Files** perspective. Go to the /Public/Pentaho BA Cookbook location in the solution and create a new folder named Analysis Reports.
7. Go back to the **Analysis Report** option in the **Opened** perspective. Choose the **Save As...** button from the PUC toolbar. Go to the /public/Pentaho BA Cookbook/Analysis Reports location and save the report as Analysis Report 1.

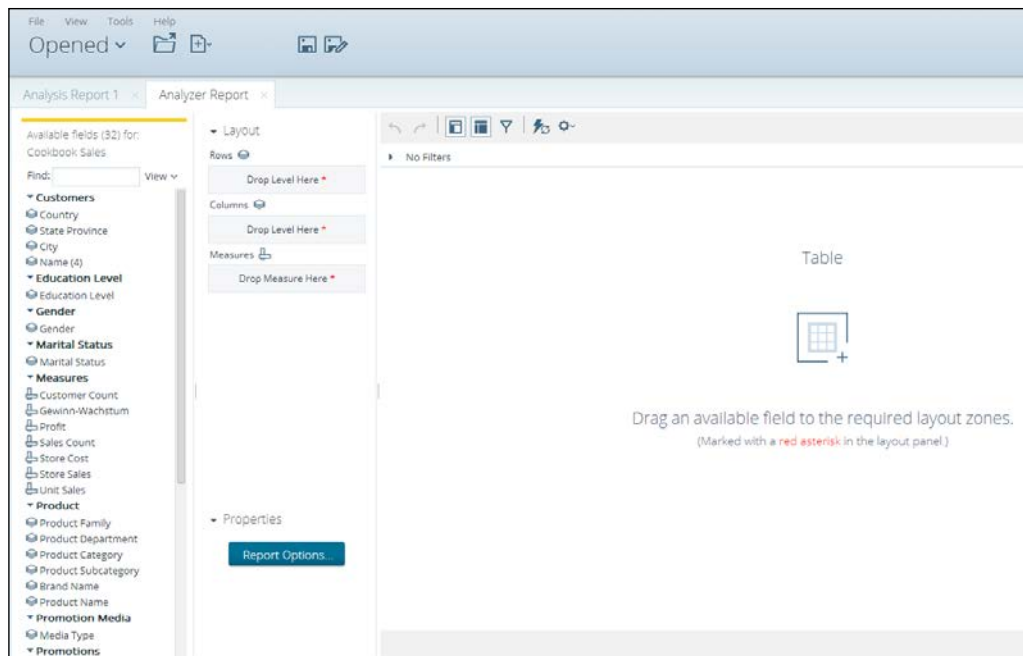
## How it works...

As soon as we enter Pentaho Analysis to create a new analysis report from scratch, we must select a data source from the **Select Data Source** dialog as shown in the following screenshot:



After we choose the data source to work on (in our case, Cookbook Sales Schema), remember to manage the auto refresh report. If this flag is set, as per default, every time something is dragged in the **Layout Fields** zone, the report is automatically refreshed and all the aggregates are recalculated. Another important thing is that if we forget to define a data source, we can always do that at the very last moment by clicking on the add data source icon in the **Select Data Source** dialog.

As soon as we have selected the data source, Pentaho Analysis starts and displays the working area as shown in the following screenshot:



The screenshot shows the layout of Pentaho Analyzer design canvas

On the extreme left-hand side, we have the set of available fields that, by default, are grouped by the dimensions' hierarchies and levels. Next, on the right, we have the three layout zones for **Columns**, **Rows**, and **Measures**. Lastly, to the right-hand side of the layout zones is the working area where the report or the visualizations will appear. If the set of available fields and layout zones restrict the report area more than expected, thereby making the report unclear, we can always hide them by clicking on the related buttons in the toolbar, as follows:

- ▶ We can hide the set of available fields by either clicking the related buttons in the Pentaho Analysis toolbar or by pressing *Ctrl + Alt + F*
- ▶ We can hide the layout zones by either clicking the related buttons in the Pentaho Analysis toolbar or by pressing *Ctrl + Alt + Y*

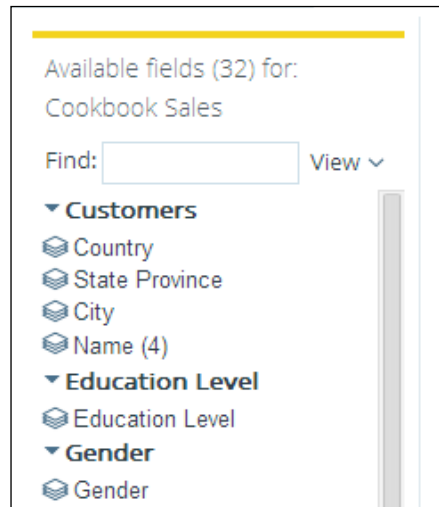
Building a report is a very simple and straightforward task: take an element from the set of available fields and drag-and-drop it to the required layout zone. The report will materialize while disposing the fields in the layout zones. As soon as the fields reach the desired configuration, save the report to the Pentaho solution.

## There's more...

By default, the set of available fields is presented by grouping them by dimension, with the dimensions displayed in the alphabetical order. For Mondrian, measures is a particular type of dimension. We can always change the way the set of available fields is presented to facilitate the user finding the right field easily.

### Changing the way the available field set is displayed

We can always change the way the set of available fields are displayed by clicking on the **View** drop-down list in the top-right corner of the available field's set area. The location of the the drop-down list is shown in the following screenshot:



There are four options available in this drop-down list, which are summarized as follows:

- ▶ **By Category:** Fields are shown by category, where every category represents the name of a dimension's hierarchy. For every dimension with just one hierarchy, the name of the category corresponds to the name of the dimension.
- ▶ **Measure, Level, Time:** Fields are basically grouped into two categories. The first category from the top is called **Measures**, which collects all of the measures. The second category named **Level** collects all of the dimensions' fields.
- ▶ **A - Z:** All the fields are displayed without any grouping but as a continuous set of fields ordered alphabetically by the letter of the first name.
- ▶ **Schema:** Each category is the name of a cube's hierarchy, and the hierarchies set follow the order specified in the cube's schema.

## See also

- ▶ To review how to create a Mondrian schema, check out the *Creating and publishing a Mondrian schema* recipe

## Adding subtotals to rows' categories

An interesting option that is very useful when we build such a type of analysis is the ability to have either subtotals by categories at various levels, or grand totals by rows and/or columns, or both. This recipe shows you how to build subtotals by rows or columns and how to enable the visualization of grand totals. Note that this recipe makes use of Pentaho Analyzer, a tool available only in the EE version of Pentaho.

## How to do it...

The following steps detail how easily we can add subtotals to rows' categories:

1. From the **Browse Files** perspective, go to the `/public/Pentaho BA Cookbook/Analysis Reports` location, select the **Analysis Report 1** report saved in the previous recipe, and choose **Open** from the **File Actions** menu on the right.
2. From the **Customer** dimension, drag-and-drop the **Country** level to the **Rows Fields** layout zone and position it before the **Product Department** level (as a parent of that level). The analyzer recalculates the report.
3. In the report's working area, go to the header of the **Country** column and right-click on it. A contextual menu is displayed. Select **Show Subtotals**.

- The analyzer recalculates the report and a line of subtotals is shown any time the **Country** member's value changes, as shown in the following screenshot:

The screenshot shows a data analysis tool interface with a toolbar at the top containing icons for back, forward, print, filter, refresh, and settings. Below the toolbar, it says 'No Filters'. The main area displays a pivot table with the following structure:

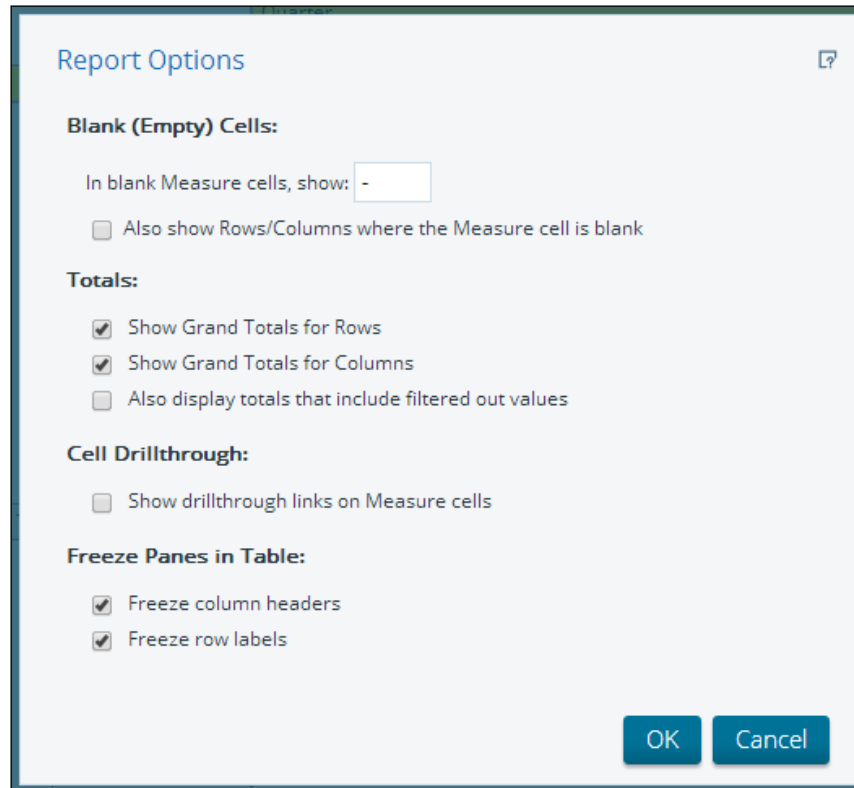
		Quarter			
		Q1	Q2	Q3	Q4
Country	Product Department	Store Sales	Store Sales	Store Sales	Store Sales
Canada	Canned Foods	1,709,13	1,786,04	1,982,73	1,182,07
	Canned Products	146,24	160,05	145,01	105,19
	Carousel	39,86	50,61	139,32	30,02
	Checkout	179,08	170,29	157,05	211,52
	Dairy	329,64	281,07	312,34	314,63
	Dairy	1,230,21	1,481,78	1,217,52	1,030,14
	Deli	857,02	1,155,53	1,332,03	820,79
	Eggs	341,74	441,97	485,68	371,25
	Frozen Foods	2,123,42	2,692,56	2,506,76	2,008,31
	Health and Hygiene	1,399,93	1,718,65	1,705,47	964,68
	Household	2,855,02	2,904,82	2,957,11	2,151,93
	Meat	180,77	197,11	188,86	194,93
	Periodicals	307,49	484,69	612,98	306,12
	Produce	3,348,13	4,189,82	3,792,00	2,752,50
	Seafood	211,83	131,16	149,78	103,15
Snack Foods	2,899,40	3,337,95	3,230,36	2,179,76	
Snacks	518,07	771,99	841,68	524,50	
Starchy Foods	555,78	674,40	508,54	422,22	
<b>Canada Total</b>		<b>23.881,13</b>	<b>27.685,00</b>	<b>27.176,30</b>	<b>19.303,03</b>
	Alcoholic Beverages	3,060,94	2,579,10	3,279,34	1,936,73
	Baked Goods	3,280,28	3,542,83	3,721,00	2,261,24
	Baking Goods	8,247,45	8,113,66	8,176,08	5,129,78
	Beverages	5,949,57	5,548,99	5,559,70	3,333,47

- Click on the **Save As...** button from the PUC toolbar. Go to the `/public/Pentaho BA Cookbook/Analysis Reports` location and save the report as `Analysis Report 2`.

### How it works...

A very simple thing with Pentaho Analyzer is the ability to add subtotals and grand totals to our report. Adding subtotals by category is a very easy task; right-click on the desired column's header that refers to the desired category and select **Show Subtotals** from the contextual menu. The report will immediately refresh and shows the desired subtotal values any time the category value changes.

There is also the possibility to add grand totals by row and by column, which is another interesting feature in the Pentaho Analyzer toolbox. To do this, we must click on the **Report Options** button and display the **Report Options** dialog box, as shown in the following screenshot:



To enable grand totals, the following steps must be followed depending on your needs:

1. Click on **Show Grand Totals For Rows** if you want to enable grand totals for rows
2. Click on **Show Grand Totals For Columns** if you want to enable grand totals for columns

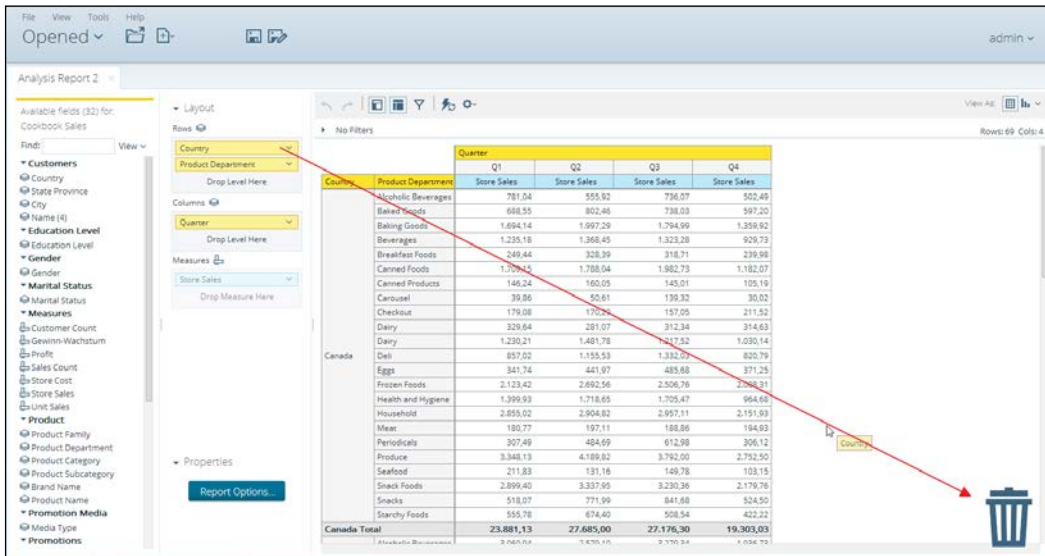
We can check either the first or the second option or both.

### There's more...

Suppose we would like to remove one or more columns from the visualization. Let me explain how can we do this very simple and intuitive task.

## Removing fields from the report table's working area

Removing fields from the report's working area is a simple thing. Let's suppose we want to remove the **Country** field from the visualization area. For this, we just have to drag it from the layout zone to the bottom-right corner of the report's working area. As soon as the mouse cursor approaches that corner, a recycle bin will appear, as shown in the following screenshot:



The screenshot shows how to remove a field from the visualization

As soon as the recycle bin appears, just drop the **Country** field over there to delete the field from the report. Another way to remove a field from the report is by dragging it from the column header's label instead of from the layout zone. Try to remove the **Country** field by going to the report area and dragging the country's header label. As soon as we start moving, the recycle bin appears and the operation is the same as before.

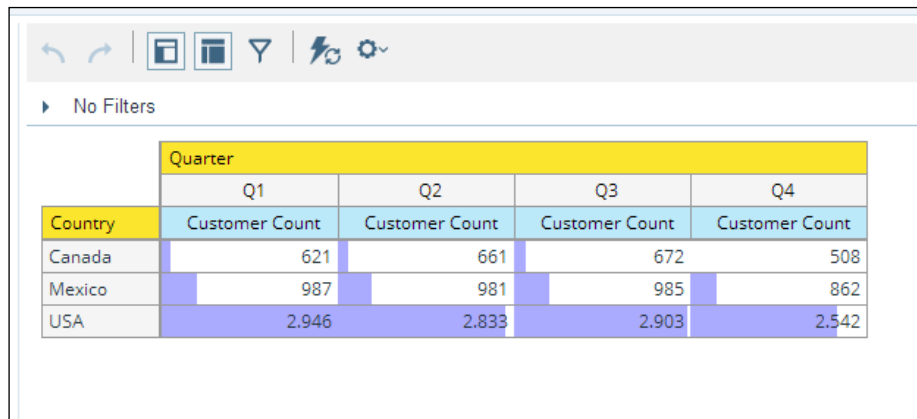
## Adding graphical indicators to table's cells

An interesting feature of Pentaho is the ability to display graphical indicators in the table. It may be a good idea to add a small arrow to indicate trends or add a colored background to communicate problems or unneeded values. This recipe shows how easy it is to decorate your report by adding a smart graphical indicator.

## How to do it...

To add a graphical indicator to a table's cell, the following steps need to be performed:

1. With the **Browse Files** perspective open, go to the `/public/Pentaho BA Cookbook/Analysis Reports` location, select the **Analysis Report 2** report saved in the previous recipe, and choose **Open** from the **File Actions** menu on the right.
2. From **Rows**, remove the **Product Department** field, and from **Measures**, remove the **Store Sales** field. In the **Measures** field's category, add the **Customer Count** measure to the report canvas.
3. Right-click on the **Customer Count** column header and select the **Data Bar : Blue** option under **Conditional Formatting**. A visual indicator represented by a little blue bar appears in the body of the cell, giving an immediate weight of the measure value displayed, as shown in the following screenshot:



	Quarter			
	Q1	Q2	Q3	Q4
Country	Customer Count	Customer Count	Customer Count	Customer Count
Canada	621	661	672	508
Mexico	987	981	985	862
USA	2,946	2,833	2,903	2,542

4. Select the **Save As...** button from the PUC toolbar. Go to the `/public/Pentaho BA Cookbook/Analysis Reports` location and save the report as **Analysis Report 3**.

## How it works...

Conditional formatting is a nice feature in Pentaho Analyzer that gives an immediate sense of the value displayed in the columns. It is very easy to enable and there are a good number of indicators to satisfy our needs.



To enable conditional formatting, right-click on the header of the column in which we want to activate the visual indicator, and choose an item from the **Conditional Formatting** contextual menu item entry. There are three types of indicators with three different visualization capabilities:

- ▶ **Color Scale:** In this case, the cell's background will assume a color according to its value and relative to the highest and lowest values displayed for that measure. It is an interesting indicator when we want to identify big differences in measure values, as for example, negative value spots that can represent a problem in a wide set of positive values. Analyzer lets us choose between four different representations of the cell's background colors.
- ▶ **Data Bar:** This is the type we choose for our example. In this case, the visual indicator is a little data bar displayed in the cell whose width is proportional to the value of the measure. This is more useful to give an immediate idea of the measures' values we are examining. This was more appropriate in our case because we just wanted to give a clear and immediate idea about which country had the highest number of visitors in its stores. We have three different bar colors to choose from for this indicator.
- ▶ **Trend Arrow:** This visual indicator is represented by an up or down arrow, which is more appropriate when we want to find evidence on measure trends. Suppose, for example, we have a measure to identify the percentage increase or decrease in sales by period. This is the right visual indicator to give a clear and immediate vision over these values. We have two different types of arrows to choose from.

## Changing the columns' sort order

Columns can be sorted in our report. We can either sort the row headers alphabetically or sort the report cells' values from highest to lowest or vice versa. This recipe shows how we can apply the desired type of sort to our report's columns. Note that this recipe makes use of Pentaho Analyzer, a tool available only in the EE version of Pentaho.

### How to do it...

To add a graphical indicator to a table cell, the following steps need to be performed:

1. From the **Browse Files** perspective, open go to the `/public/Pentaho BA Cookbook/Analysis Reports` location; select the **Analysis Report 2** report saved in the previous recipe and choose **Open** from the **File Actions** menu on the right.
2. Suppose we want to sort the **Store Sales** values from highest to lowest. Go to the header of the **Store Sales** field for example (we can choose any column we want; it is not important which of the four columns' header we select).
3. Right-click on the column header and select **Sort Values High > Low** from the contextual menu.

4. A dialog box informs us that we are going to re-sort our data set by **Store Sales**. The report is reloaded and the new sorting is applied.
5. Choose the **Save As...** button from the PUC toolbar. Go to the `/public/Pentaho BA Cookbook/Analysis Reports` location and save the report as `Analysis Report 4`.

### How it works...

We can easily change the sort order of our records by acting on the contextual menus that will display on columns headers. We have two different sorting options:

- ▶ Sort the recordset based on the row headers' label values
- ▶ Sort the recordset based on the value of the measures displayed in the report

To sort the dataset by a specific measure value, go to the measure header column, right-click on the column's header by placing the cursor over the name of the measure, and choose one of the two **Sort Values** options depending on your needs. We can sort either from highest to lowest or vice versa.

If you want to change the sort order of the categories' row label values, go to the header of a column representing a category (in our case, go to **Product Department**) and right-click on it. Depending on our needs, we can choose **Sort A > Z** or **Sort Z > A** from the contextual menu.



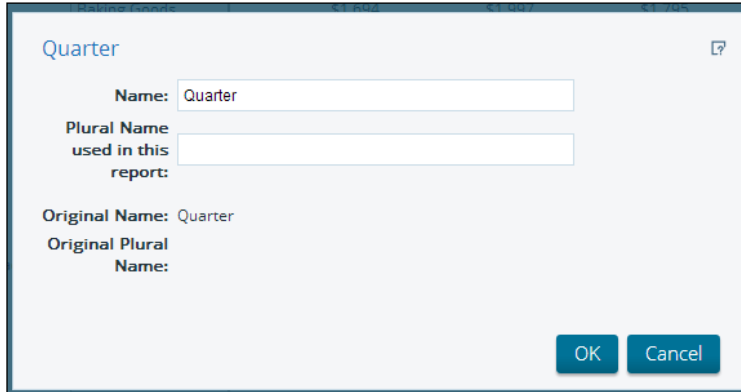
Changing the sort order will, of course, break other sort orders previously specified in the report.

### There's more...

An interesting opportunity given by Pentaho Analyzer is that we can change the column header's label according to our needs. Moreover, we can apply some formatting options to our measure values if they are not provided in the table schema. Let's see how we can do this in the following paragraphs.

### Changing a columns' header labels

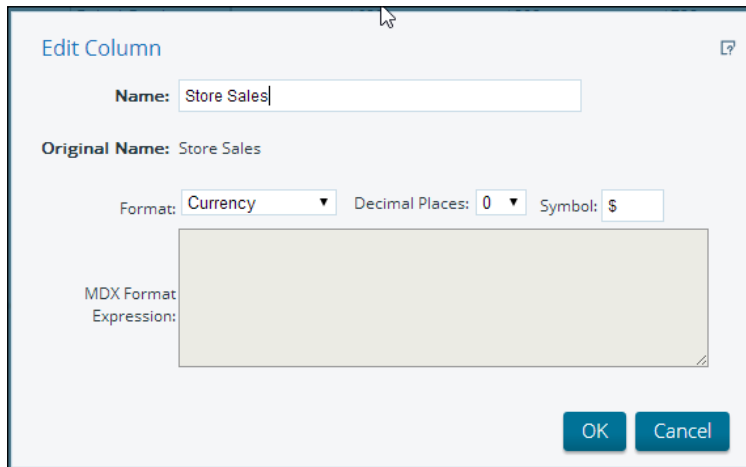
Changing a column's header is a very easy and useful task. This feature lets us change the original label with another label that we think could be better for our specific business case, or the possibility to translate it to our own language. To do this, we must right-click on the column's header that we want to change the label for and select **Edit....** The **Column Header** properties dialog box appears as shown in the following screenshot:



If we want to change the column header's label, type the new name in the **Name** field. We can also specify a better pluralized label by typing in the new plural name in the **Plural Name used in this report** field. Click on **OK** and the new label is immediately shown in the report body.

### Applying formatting rules to measures' values

Another nice feature given by the analyzer is the ability to apply formatting rules to a measures' values. To do this, go, for example, to the **Store Sales** measure header and right-click on it. The **Edit Column** properties dialog appears as shown in the following screenshot:



As we can see, we can change the label of the column's header in this dialog too. Type the name of the field if the current name is not meaningful. Then, we have a section where we can specify some formatting options for the specific kind of measure. The appearance of this dialog's section is guided by the values contained in the **Format** drop-down list. The **Format** drop-down list contains the types of formatting we can apply to the cell values. The following is a summary of the values and a brief description of the possible actions:

- ▶ **General Number:** This format considers the values we display as a number that gives only the ability to set the number of decimal places. To do this, set the value by selecting it from the **Decimal Places** drop-down list.
- ▶ **Currency:** This format considers the values we display as a currency amount. Besides giving us the ability to set the number of decimal places (see what is specified for **General Number**), it lets us specify the currency symbol. To do this, select the appropriate currency symbol from the **Symbol** drop-down list.
- ▶ **Percentage (%):** This format considers the values we display as a percentage. It only gives us the possibility to specify the number of **Decimal Places**, the same as for **General Number**. By default, it adds the percentage symbol at the right end of the displayed number.
- ▶ **Expression:** This is a free format. Selecting the **MDX Format Expression** text area lets you type an MDX expression to specify the desired format we want to apply.

## See also

If you are interested in having a recap on how to build a Pentaho Analysis report, look at the *Creating a new analysis report from scratch* recipe.

## Adding a simple calculated measure

As we already saw in the first recipe, *Creating and publishing a Mondrian schema*, we can define a set of calculated measures in the cube definition. Sometimes, it is useful to have the capability to define custom calculated measures. This recipe will show how we can easily do this with little knowledge of the MDX language. Note that this recipe makes use of Pentaho Analyzer, a tool available only in the EE version of Pentaho.

## How to do it...

The following steps detail how to add a simple calculated measure to our report:

1. From the **Store Type** category, take the **Store Type** field and drag it to the **Rows** layout zone. From the **Time** category, get the **Quarter** field and drag it to the **Columns** layout zone. Then, from the **Measures** category, get the **Store Sales** field and drag it to the **Measures** layout zone.

- Suppose we want to define a measure to evaluate the trend of our **Store Sales** table over quarters. Right-click on the **Store Sales** column header and select the **Trend Measure** option under **User Defined Measure**.
- The **New Trend Measure** dialog box opens as shown in the following screenshot:

**New Trend Measure**

Name:

Trended Measure: Store Sales

Period type:

Number of periods:

Show trend as:

Decimal Places:

**Note:** To trend, use time attributes only (e.g. Year, Quarter, Month).  
To select a time attribute, you need to include one in the report.

- In the **Name** field, type the name of the new measure, for example, Sales Trend.
- From the **Period type** drop-down list, select **Quarter**.
- From the **Show trend as** drop-down list, select **Delta from previous period**.
- Click on the **OK** button when you finish filling in the dialog box. The report will refresh and immediately show the new measure.
- For a better representation of the new measure, we can add a trend arrow. To do this, right-click on the **Sales Trend** column header and select the **Trend Arrow: Green-Red** option under the **Conditional Formatting** contextual menu.
- Click on the **Save As...** button from the PUC toolbar. Go to the `/public/Pentaho BA Cookbook/Analysis Reports` location and save the report as **Analysis Report 5**.

## How it works...

After we have added the **Store Type** and **Quarter** fields to the **Rows** and **Columns** layout zones respectively, and after we have added **Store Sales** to the **Measures** layout zone, we can easily add a trend measure by selecting the **Store Sales** column header and selecting the **Trend Measure** option under **User Defined Measure**. This action implicitly identifies the measure that we are referring to in order to calculate the trend: this is the trend of our `Store Sales` table. Trend measures are a useful indicator and often, a user requires these when building an analysis. A new trend measure must refer to a period; we can select the related field from the **Period** drop-down list. After we have chosen the period, we can choose to specify the number of relative periods we are referring to by typing it in the **Number of periods** field. The default value is one, meaning the previous period, but we can choose the value that is more applicable to our business needs.

Then, the **Show trend as** drop-down list gives us the ability to specify the type of measure value we want to use to display the trend value. We must carefully evaluate our selection because this is the value that we will refer to our trend. The following steps detail the possible ways Pentaho Analyzer uses to calculate trends:

- ▶ **Value of previous period:** It goes  $n$  periods backward (as many as we specified in the **Number of periods** field) and displays the value for the related period
- ▶ **Delta from previous period:** It goes  $n$  periods backward (as many as we specified in the **Number of periods** field), calculates the difference between that period and the actual, and displays that value as a trend value
- ▶ **% of change from previous period:** It goes  $n$  periods backward (as many as we specified in the **Number of periods** field), calculates the percentage of change from that period and the actual value, and displays that value as a trend value
- ▶ **Average of previous periods:** It goes  $n$  periods backward (as many as we specified in the **Number of periods** field), calculates the average of the referred measure values (in our case, **Store Sales**) from that period to the actual, and displays that value as a trend value
- ▶ **Sum of previous periods:** It goes  $n$  periods backward (as many as we specified in the **Number of periods** field), calculates the sum of the referred measure values (in our case, **Store Sales**) from that period to the actual, and displays that value as a trend value

Click on the **OK** button when you're done and the report immediately shows the value for the new measure. As a last refinement, this is the opportunity to add a trend arrow to our analysis in order to enhance the impact of our report on our users.

## There's more...

Apart from the **Trend** measures, we also have the opportunity to define other types of calculated measures. The next two paragraphs will introduce how to add a generic calculated measure using an MDX expression fragment and how to define a running sum or a rank measure.

### Adding other predefined calculated measure types

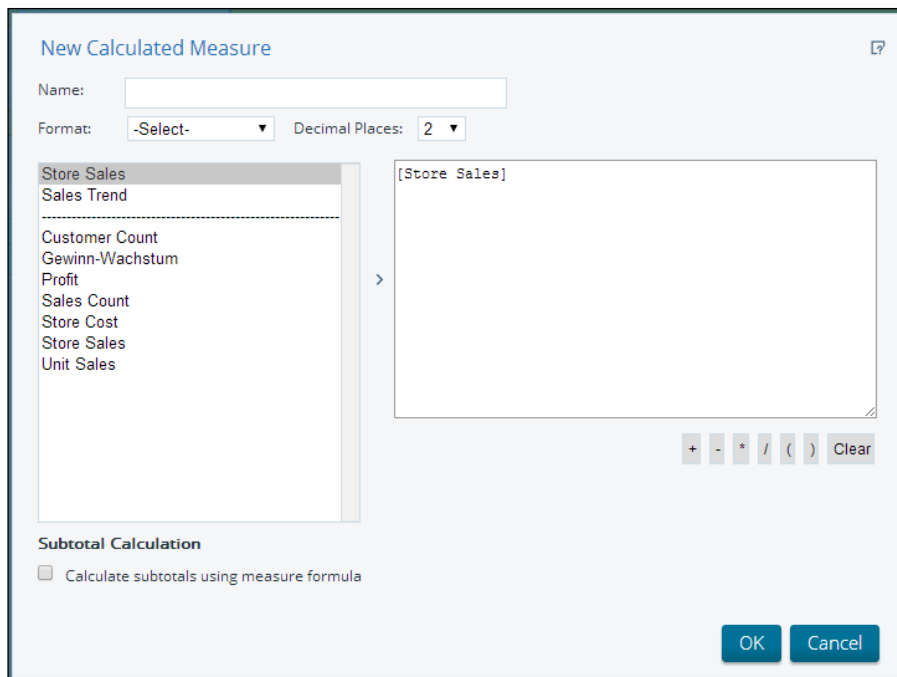
Apart from the trend measures, we also have the possibility to add other predefined calculated measures such as **Running Sum**, **% of**, and **Ranks**. To define such calculated measures, right-click over the referred measure's column header and select **% of, Rank, Running Sum...** from the **User Defined Measure** contextual menu. The following is a brief illustration of each of these calculated measures:

- ▶ **% of**: Calculates the percentage distribution of the current measure's value with respect to the sum total of all the values for the referred measure. We can calculate the percentage of the values either by row or by column.
- ▶ **Rank**: Ranks the current value of the referred measure with respect to all others by classifying them through an order. A rank indicates the row with the maximum value for the referred measure. We can rank values either by row or by column.
- ▶ **Running Sum**: Calculates a **Running Sum** value for the referred measure. We can calculate the **Running Sum** values either by row or by column.

### Adding a generically calculated measure

Another possibility to define a calculated measure in Pentaho Analyzer is by directly writing an MDX expression. This gives maximum power and flexibility but requires an indispensable knowledge of the MDX language.

To do this, right-click on the referred measure's column header and select **Calculated Measure** from the **User Defined Measure** contextual menu. The **New Calculated Measure** dialog box appears as shown in following screenshot:



Type the name of the new calculated measure in the **Name** field. Select the **Format** and **Decimal Places** values from the related drop-down list boxes. In the formula editor on the right-hand side, write the formula expression for our calculated measure. We can select an existing measure from the measures list on the left-hand side by double-clicking on the measure, and it automatically appears at the cursor's position. After we have finished typing the MDX expression, we can click on the **OK** button to confirm the entered formula and close the dialog.

### See also

- ▶ Go through how we create graphical indicators for our cells by reviewing the *Adding graphical indicators to table's cells* recipe again

## Creating visualizations with Pentaho Analyzer

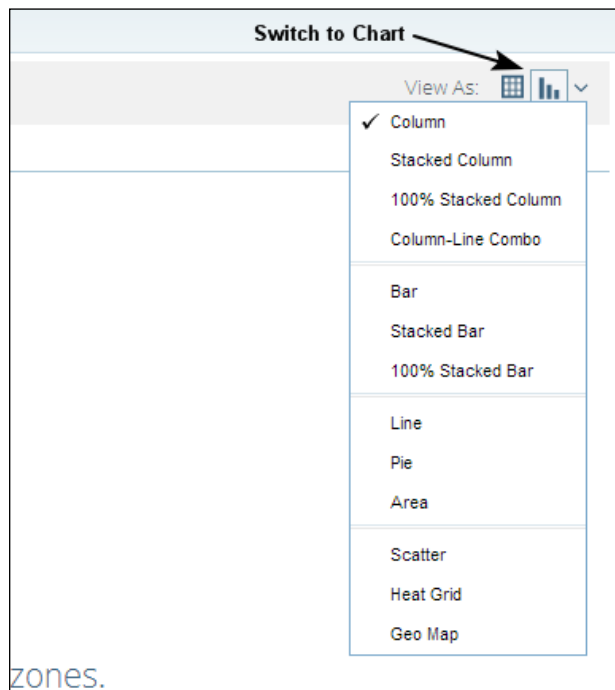
Pentaho Analyzer is also a great tool to build graphical visualizations by leveraging the capabilities of OLAP. This recipe will show how to create a Geo Map by using Pentaho Analyzer to show the distribution of sales over the territory. Note that this recipe makes use of Pentaho Analyzer, a tool available only in the EE version of Pentaho.



## How to do it...

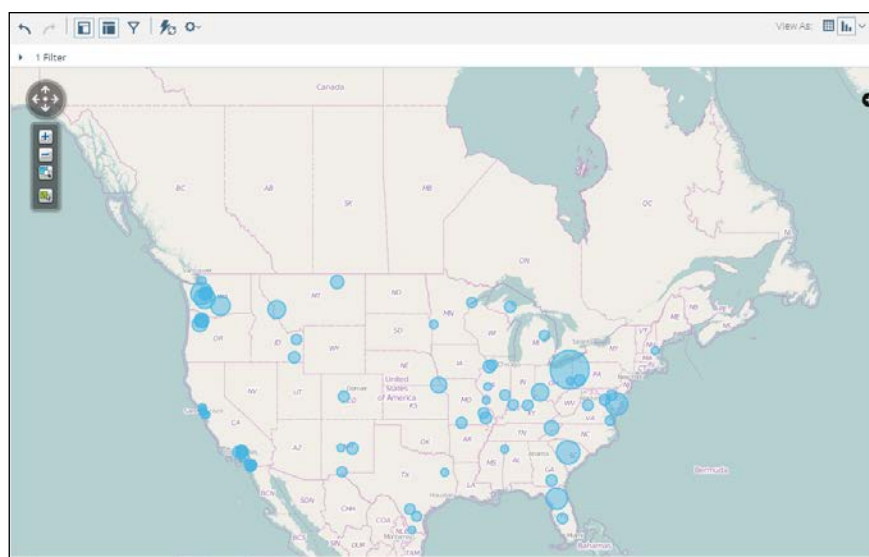
The following steps detail how to create a geographical visualization in order to identify the distribution of sales in the territory:

1. Create a new Pentaho Analysis report by clicking on the **Create New** button in the **Home** perspective and then clicking on **Analysis Report**, or by choosing **Analysis Report** from the **New** option under the **File** PUC menu.
2. The **Select Data Source** dialog box appears. Select **Cookbook Sample Schema** and click on **OK**.
3. Click on the switch to chart format icon button, located in the top-right corner of the Pentaho Analyzer canvas or press *Ctrl + Alt + C*.
4. The chart format view appears. Click on the drop-down list located to the right of the switch to chart format icon button; select the **Geo Map** item.



5. From the **Country Category** panel on the left, drag-and-drop the **Country** field and then the **City** field to the **Geography** layout zone.
6. Click on the show all filters in use icon button in the toolbar located at the top of the chart canvas or press *Ctrl + Alt + T* to open the **Filter Zone** area. Always from the **Country** category on the left, drag-and-drop the **Country** field to the filter area.

7. The **Filter on Country** dialog box opens. Select the **Match a specific string** radio button and type `U.S.A.` in the field in the right-hand side of the drop-down list. Click on the **OK** button and close the dialog. Now that we specified one filter, the filter area contains the reference to the filter we just defined.
8. From the **Measures** category, drag-and-drop the **Store Sales** measure to the **Size By** zone.
9. From the **Time** category, drag-and-drop the **Year** field to the **Other fields** layout zone.
10. We now have enough fields to display the chart. The chart is automatically refreshed and the map is loaded with the bubbles located in relation to the U.S.A. cities that are present in the results of the query. See the following screenshot for reference:



The screenshot shows a sample of visualization made with a geographical map

11. Feel free to zoom in to the chart by clicking on the zoom icon and drawing a box around the area to zoom.
12. Choose the **Save As...** button from the PUC toolbar. Go to the `/public/Pentaho BA Cookbook/Analysis Reports` location and save the report as `Analysis Report 6`.

## How it works...

Designing visualization that leverages OLAP data is a simple thing with Pentaho Analyzer. This time we started our report from scratch. After selecting the **Cookbook Sample Schema** data source and starting Pentaho Analyzer, as the first thing we need to switch to the chart format view by clicking on the switch to chart format button. Then, from the related drop-down list to the right of that button, we can choose various types of different visualizations; select **Geo Map**.

Note that the layout zone areas in the **Chart** view are different in number and meaning with respect to the layout zone areas in the **Report** view. Specifically, there can be different layout zone layouts depending on the chart type we choose.

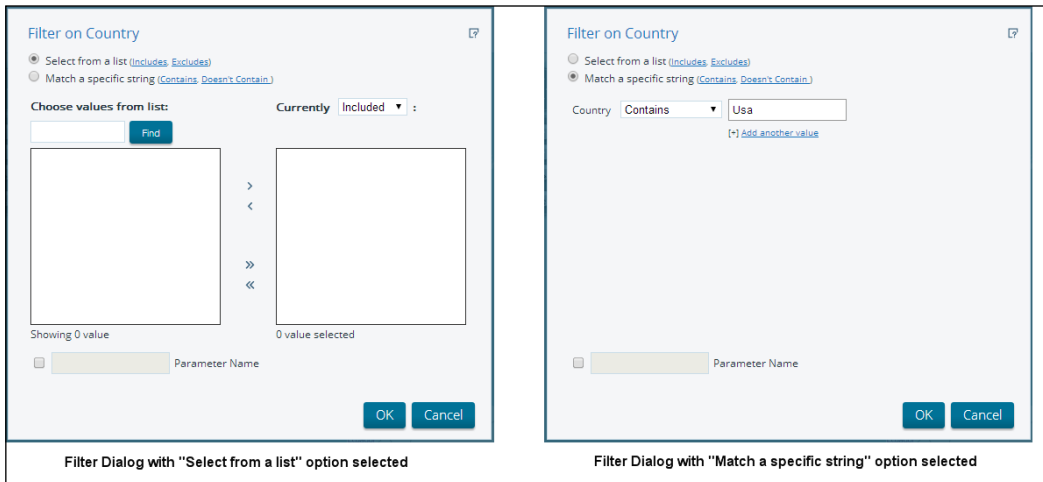
In our **Geo Map** chart, we have four layout zones: **Geography**, **Size By**, **Color By**, and **Other**. In the first layout zone called **Geography**, we will drag-and-drop all the fields that contribute to specify geographical location information. For this reason, we will add in two fields from the **Customers** category: **Country** and **City**.

The idea of a geographical chart to give a representation, either by different colors or by different areas, of the measure's weight (the fact that a measure can have a high or low value respect to other values), relating it to a specific geographical location. In our case, the idea is to find the geographical distribution of our customers' sales in U.S.A. Every geographical location with an associated measure is identified by a pinpoint in the map. To express the measure's value graphically, we can choose between pinpoints with different colors or different sizes:

- ▶ If we want to pinpoint a measure by specifying its value with a different color, we must drop our measure over the **Color By** layout zone
- ▶ If we want to pinpoint a measure by specifying its value with a different size of the circle, we must drop our measure over the **Size By** layout zone

We chose to express the value of our measure by size, so we will proceed with the second option. From the **Measure** category, we get the **Store Sales** measure; drag-and-drop it to the **Size By** layout zone.

We decided to restrict our analysis to the U.S.A. To do this, click the show all filters in use icon button in the toolbar in the chart canvas or press **Ctrl + Alt + T** to open the **Filter Zone** area. Then, from the **Country** category on the left, drag-and-drop the **Country** field to the filter area. The **Filter on Country** dialog box opens as shown in the left part of the following screenshot for details:



The dialog presents two radio buttons with two different ways to specify the filtered values. The first option, **Select from a list**, lets us choose the filter measure from a list of values by choosing the required values from the list on the left and move the required values to the list on the right. The list on the right will contain the set of filtered values.

Another option is to manually specify a condition for our filter (see the right part of the previous screenshot for details). We can do this by selecting the option that matches a specific string. The dialog changes the appearance by giving you the ability to specify a filter operator (select it from a drop-down list) and the filter value. In our case, we specified the condition **Country** contains `U.S.A.` As soon as we are fine with our filter settings, click on the **OK** button to close the dialog and confirm the filter. The filter reference will appear in the filter area.

Lastly, from the **Time** dimension, select the **Year** field and drag-and-drop it to the **Other** layout zone. As soon as all the required fields are correctly disposed in the chart canvas, the report is immediately refreshed and the map with the graphical pinpoints appears as expected.

Try to explore all the other chart types and experience all the differences with this Geo Map case.

## Exporting reports in the Excel or PDF format

As the last operation in our analysis, we can export our report in an Excel or PDF format. This recipe will show you how to do this. Note that this recipe makes use of Pentaho Analyzer, a tool available only in the EE version of Pentaho.

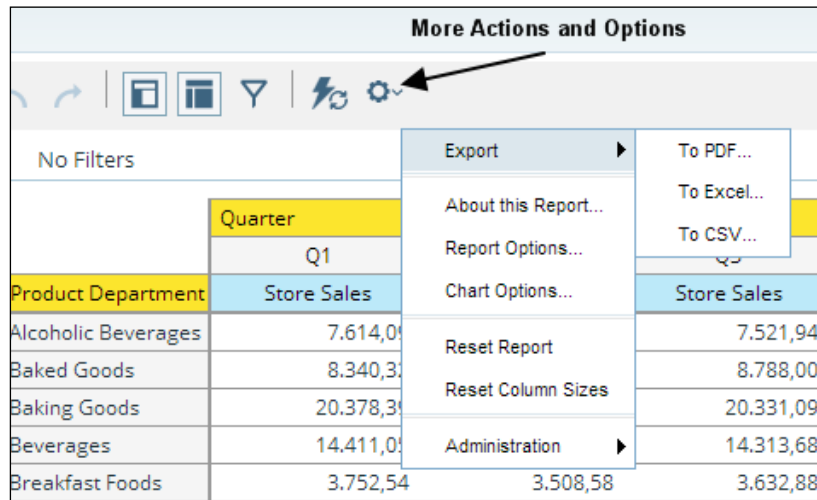
### How to do it...

The following steps detail how we can easily export our analysis report to an Excel file:

1. With the **Browse Files** perspective open, go to the `/public/Pentaho BA Cookbook/Analysis Reports` location, select the **Analysis Report 1** report saved in the previous recipe, and choose **Open** from the **File Actions** menu on the right.
2. Click on the more actions and options icon button and select **Excel** from the **Export** context menu.
3. The Excel report is exported and downloaded to your filesystem in the chosen location.

## How it works...

Excel and PDF are two important functions that are usually used to share report documents between team members. Once you are ready to export the report, click on the more actions and options icon button. See the following screenshot for reference:



Suppose we choose to export our report to the Excel format; once you select the export format from the context menu, the **Export to Excel** dialog box appears. Here we can select the page format and the orientation as well as some scaling options, specified as follows:

- ▶ The first possibility to scale our report is to resize it to a percentage of the original size. To do this, select the **Adjust to** radio button and fill in the field on the right-hand side of the **Adjust to** label with the desired value.
- ▶ A second possibility, by selecting the **Fit to** radio button, is to scale the report by declaring the proportions between page width and height.

Note that these parameters to set the right proportion between margins are only present as long as we are exporting an Excel file.

## Creating an analysis report using Saiku

Saiku is an OLAP client created by the community members. We can find a reference to the project at <http://meteorite.bi/saiku>. It is a complete OLAP client available for both versions, CE and EE. Saiku is freely available and can be installed using Pentaho Marketplace. This recipe will show you how to make the same simple analysis report we made with Pentaho Analyzer using Saiku.

## Getting ready

For this recipe, Saiku must be installed from the Marketplace and must be available in the Pentaho Server.

## How to do it...

The following steps detail how to create a simple analysis report by using Saiku:

1. To create a new OLAP analysis report using Saiku, choose **Saiku Analytics** from the **File | New PUC** menu.
2. The Saiku OLAP client starts. From the top-left drop-down list named **Cubes**, select a cube named **Cookbook Sample**. Saiku will load the set of dimensions and measures in the tree lists given below.
3. Let's build a simple report. From the **Product** dimension, drag-and-drop the **Product Department** field to the **Rows** layout zone. From the **Time** dimension, get the **Quarter** and **Month** levels and drag-and-drop, respectively, to the **Columns** layout zone.
4. From the **Measures** dimension, drag-and-drop the **Store Sales** measure to the **Columns** layout zone.
5. Because the **Auto Refresh** option is enabled in Saiku, the report will refresh every time we drag-and-drop something in any of the layout zones.
6. Choose the **Save As...** button from the Saiku toolbar. The Saiku **Save As...** dialog appears. Go to the `/public/Pentaho BA Cookbook/Analysis Reports` location and save the report as `Saiku Report 1`.

## How it works...

Saiku is a community project and represents an alternative to Pentaho Analyzer for CE users. It has a modern user interface with drag-and-drop functionalities to easily dispose the analysis fields to the report canvas.

On the left-hand side of the screen, we have the main cube elements as follows:

- ▶ On the top-left side of the user interface, we have a drop-down list that lets us select the cube we need to use for our analysis.

## Creating Analysis Reports

- Below the cube drop-down list, we have two tree view lists; the first list contains the dimensions' categories and their related fields, and the second list contains the set of measure fields. See the following screenshot for reference:

The screenshot shows the Saiku Analytics interface. On the left, there are two tree view lists: 'Dimensions' and 'Measures'. The 'Dimensions' list includes Customers, Education Level, Gender, Marital Status, Product (with sub-items like Product Family, Product Department, Product Category, Brand Name, Product Name), Promotion Media, Promotions, Store, Store Size in SQFT, Store Type, Time, and Yearly Income. The 'Measures' list includes Unit Sales, Store Cost, Store Sales, Sales Count, Customer Count, Profit, and Profit Growth. The main area displays a data table for the 'Cookbook Sales' cube. The table has columns for quarters (Q1, Q2, Q3, Q4) and rows for various product departments. The 'Store Sales' measure is selected for the columns.

Product Department	Q1		Q2			Q3			Q4		
	1	2	3	4	5	6	7	8	9	10	11
Alcoholic Beverages	\$2,475.17	\$2,538.63	\$2,599.29	\$2,301.49	\$2,161.55	\$2,290.38	\$2,460.75	\$2,300.96	\$2,700.23	\$2,282.87	\$2,925.67
Beverages	\$4,682.88	\$4,672.40	\$4,875.77	\$4,423.77	\$4,557.21	\$5,056.49	\$4,745.70	\$4,554.40	\$5,013.58	\$4,452.92	\$5,188.62
Dairy	\$1,281.85	\$1,134.59	\$1,143.70	\$1,136.24	\$1,234.37	\$1,310.28	\$1,229.39	\$1,328.42	\$1,494.41	\$1,277.72	\$1,659.38
Baked Goods	\$2,694.17	\$2,597.93	\$2,843.22	\$2,684.56	\$2,554.11	\$2,607.31	\$3,021.99	\$2,810.82	\$2,955.39	\$2,888.94	\$3,568.27
Baking Goods	\$6,646.73	\$6,847.05	\$6,884.61	\$6,867.32	\$6,644.51	\$6,687.56	\$6,648.20	\$6,320.70	\$7,362.10	\$6,108.79	\$7,948.71
Breakfast Foods	\$1,256.05	\$1,177.46	\$1,336.03	\$1,228.51	\$1,094.66	\$1,185.51	\$1,286.40	\$1,095.10	\$1,246.70	\$1,273.86	\$1,322.60
Canned Foods	\$7,085.21	\$6,882.86	\$6,741.43	\$6,647.72	\$6,794.70	\$6,727.82	\$6,792.36	\$6,540.33	\$6,679.08	\$6,315.06	\$7,697.74
Canned Products	\$579.23	\$557.18	\$567.46	\$496.08	\$527.70	\$554.25	\$598.50	\$556.16	\$573.27	\$549.67	\$657.79
Dairy	\$5,008.81	\$4,952.86	\$5,525.81	\$5,082.76	\$4,814.87	\$5,208.17	\$5,620.35	\$4,836.13	\$5,126.48	\$4,789.82	\$6,165.83
Deli	\$4,371.43	\$4,142.94	\$4,226.28	\$4,359.06	\$4,100.24	\$4,362.60	\$4,327.17	\$4,269.42	\$4,080.10	\$4,200.54	\$5,006.35
Eggs	\$1,495.01	\$1,602.07	\$1,482.81	\$1,720.22	\$1,496.76	\$1,374.31	\$1,597.80	\$1,328.01	\$1,611.35	\$1,500.40	\$1,770.87
Frozen Foods	\$9,822.38	\$8,880.34	\$9,130.08	\$9,219.74	\$10,139.69	\$9,332.28	\$9,577.71	\$9,599.82	\$9,801.82	\$8,582.22	\$11,233.69
Meat	\$611.50	\$626.46	\$618.35	\$619.52	\$737.72	\$696.82	\$762.52	\$674.88	\$654.53	\$597.54	\$762.38
Produce	\$14,088.02	\$13,616.25	\$13,784.97	\$14,094.82	\$13,115.86	\$14,770.68	\$15,628.91	\$13,636.78	\$14,883.81	\$12,985.43	\$16,419.06
Seafood	\$609.75	\$275.29	\$618.05	\$607.55	\$474.73	\$490.78	\$607.86	\$567.76	\$637.09	\$515.00	\$667.89
Snack Foods	\$11,763.61	\$11,217.06	\$12,385.03	\$11,389.50	\$11,640.30	\$11,308.24	\$11,672.91	\$11,773.82	\$11,901.09	\$11,147.04	\$13,926.43

The screenshot shows how a sample visualization is created using Saiku Analytics

On the right-hand side of the cube navigator, we have the following:

- At the top, a toolbar containing the main Saiku function buttons
- Three layout zones to dispose fields for rows, columns, and filters
- The report canvas where we will build the analysis report

We can see the details described in the previous points in the previous screenshot. Note that, once loaded, the Saiku application is available as a tab of the PUC's **Opened** perspective.

Select the **Cookbook Sales** cube from the cube's list to load the set of dimensions and measures. From the **Product** dimension, drag-and-drop the **Product Department** field to the **Rows** layout zone. From the **Time** dimension, get the **Quarter** and **Month** levels and drag-and-drop them to the **Columns** layout zone. From the **Measures** dimension, drag-and-drop the **Store Sales** measure to the **Columns** layout zone.

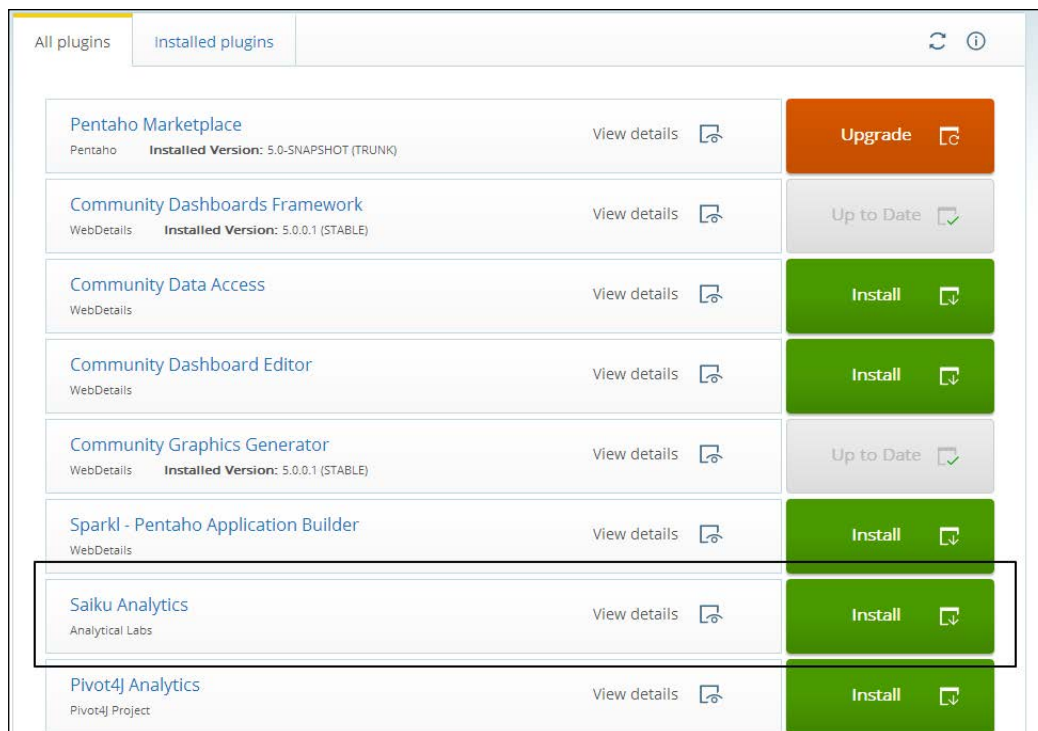
Saiku immediately refreshes the report any time you drag some new fields to the layout zone. When the report is finalized, we can save it to the solution. Saiku has its own save functionality; as we can see, the **Save** and **Save As...** buttons in the PUC toolbar are hidden, but we have a save icon in the Saiku toolbar. Click on the **Save** button in the Saiku toolbar to let the Saiku **Save As...** dialog appear. Navigate through the solution to look for the /public/Pentaho BA Cookbook/Analysis Reports location and save the report as Saiku Report 1.

## There's more...

We take the opportunity to illustrate the Saiku installation and two of its functionalities, such as how to add a filter and how to make a visualization. Let's go through all of these in the next few paragraphs.

### Installing Saiku from the Pentaho Marketplace

Pentaho Marketplace is a tool that easily lets the user install new Pentaho plugins, thereby increasing Pentaho's functionality. To use the Pentaho Marketplace, we must be a part of the Pentaho's administrator role. If it is our role, we will access another perspective called **Marketplace**. Go to the **Marketplace** perspective and get **Saiku Analytics** from the list of available plugins. See the following screenshot for reference:



Once found, click on the green **Install** button. After the installation process has started, a few dialogs appear before the installation is complete. As soon as the installation is complete, an information dialog informs you about the successful completion of the process.

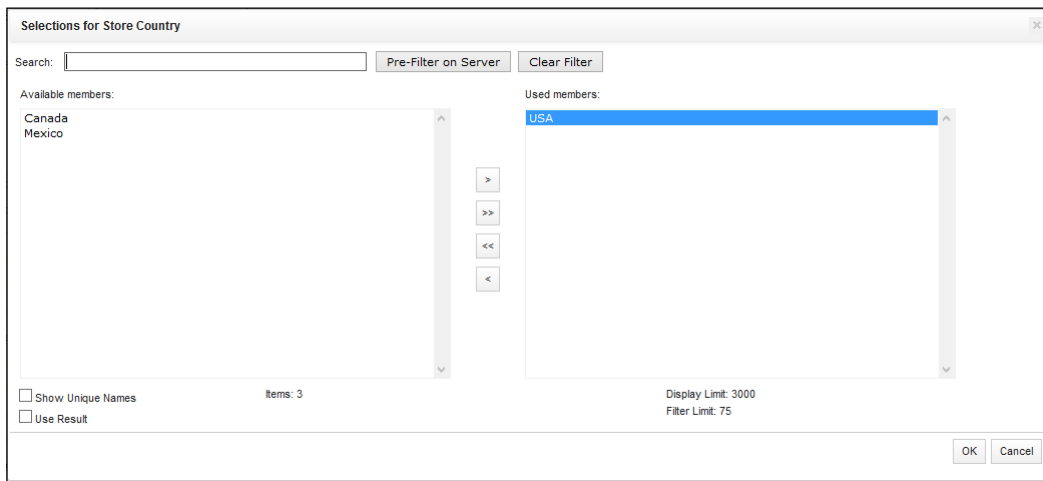
Remember to restart the Pentaho server before using the plugin you've just installed.



## Adding filters to reports

Adding a filter to a report in Saiku is an easy task. Let's suppose we want to add a filter to our report so that we can analyze **Store Sales** by **Product Department** only for the stores located in the U.S.A.

Go to the **Dimensions** tree view, take the **Store Country** field from the **Stores** category, and drag-and-drop it to the **Filters** layout zone. The selection for the **Store Country** dialog box appears, as shown in the following screenshot:

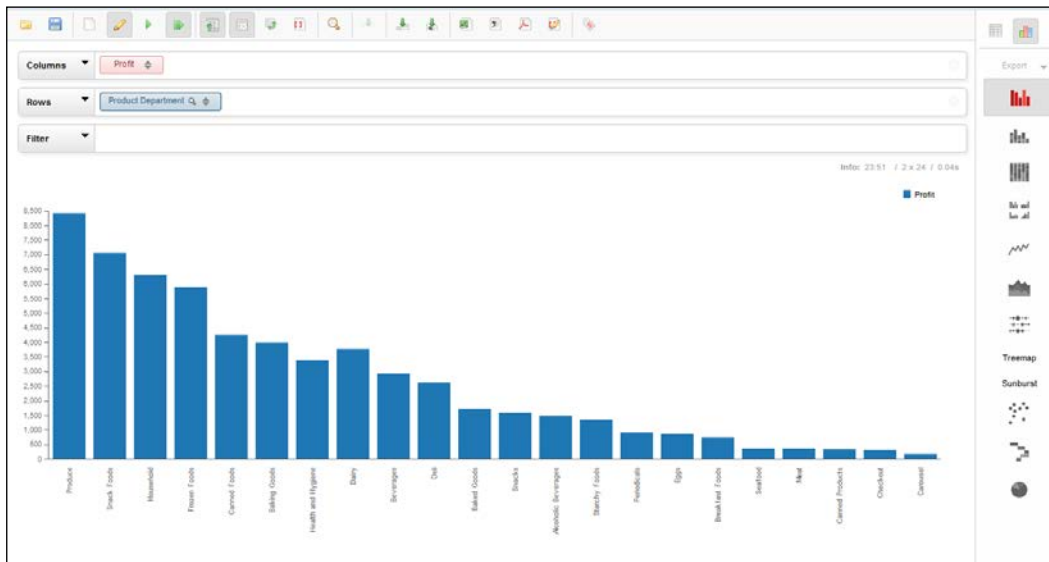


From the list on the left-hand side, **Available members**, select the entry **USA** and then move it to the list on the right, **Used members** (see the previous screenshot for details). Click on **OK** to confirm the filter set. The filter field appears in the **Filter** layout zone, and the report will immediately reload, showing new filtered results.

## Designing good looking visualizations

As the last example of Saiku's power and simplicity, let's build a visualization to graphically display some results. Start a new Saiku instance by selecting **Saiku Analytics** from the **New** option under the **File** PUC menu. From the **Product** dimension, drag-and-drop the **Product Department** field to the **Rows** layout zone. From the **Measures** dimension, get the **Profit** field and drag-and-drop it to the **Columns** layout zone.

Now, let's sort the results by profit. To do this, click on the **Rows** layout zone header and choose **Profit** from the **Descending (Breaking hierarchy)** option under the **Sort** contextual menu. The result will be sorted by profit in the descending order. Now, to get the desired visualization, click on the chart mode button in the top-right corner of the Saiku user interface. A nice bar chart ordered from higher to lower is displayed in the Saiku canvas, as shown in the following screenshot:



The screenshot shows a sample of the graphical capabilities of Saiku Analytics

Feel free to change the visualization by selecting other chart types from the toolbar on the right-hand side of the Saiku canvas.



# 7

## Creating Reports Using Pentaho Report Designer

In this chapter, we will cover the following recipes:

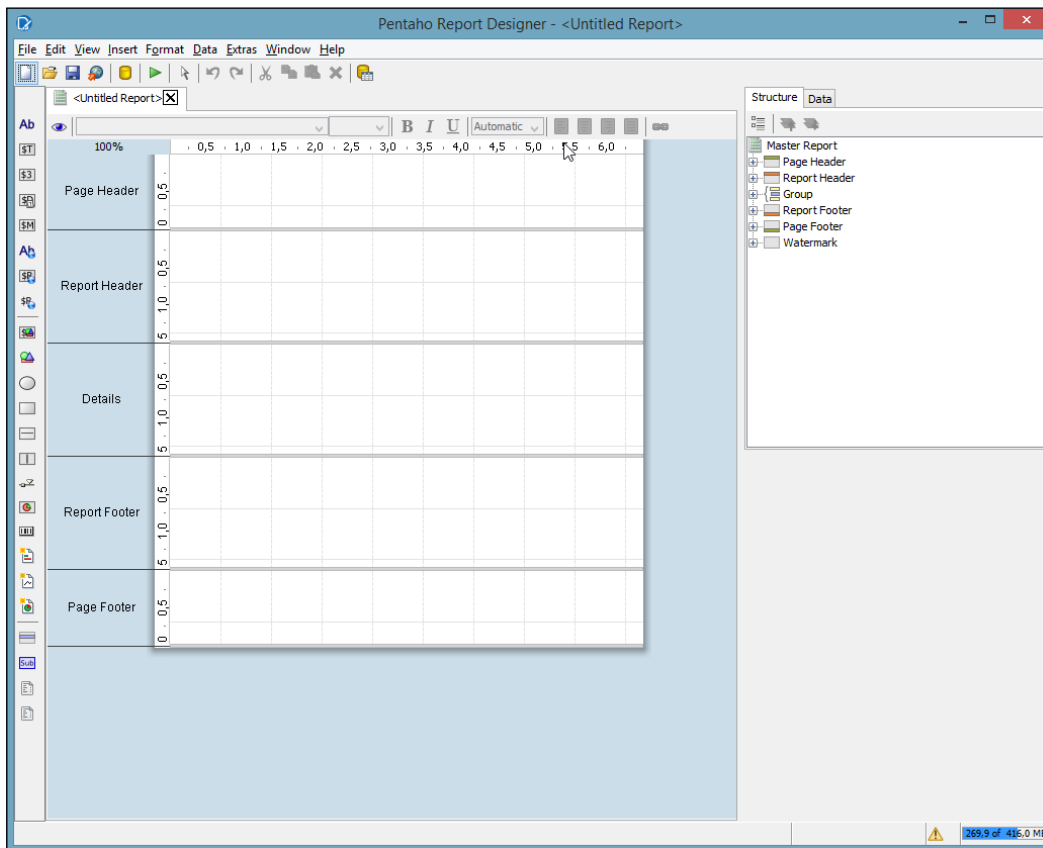
- ▶ Configuring JNDI connections for development
- ▶ Managing JDBC connections
- ▶ Managing ETL connections
- ▶ Using layout to simplify report development
- ▶ Using style sheets to consistently manage field formats
- ▶ Changing field properties at runtime with formulas
- ▶ Using input parameters
- ▶ Using groups to define report aggregations
- ▶ Using functions to add calculated fields
- ▶ Using subreports to embed content
- ▶ Embedding microcharts in reports with sparklines
- ▶ Adding charts to our report

### Introduction

This chapter is about Pentaho Report Designer and contains a set of recipes to get you acquainted with everyday tasks related to the design and development of sophisticated reports.

## Creating Reports Using Pentaho Report Designer

Pentaho Report Designer is the tool of the Pentaho ecosystem that is used to build good-looking and complex reports. We already saw in *Chapter 5, Creating Reports Using Pentaho Interactive Reporting*, a set of recipes related to Pentaho Interactive Reporting. Pentaho Interactive Reporting is a web tool accessible from the Pentaho User Console and is used to easily build tabular reports that interactively leverage on metadata domain models. On the other side, Pentaho Report Designer lets us build complex reports from raw data sources (**JDBC**, **OLAP**, **Metadata**, **ETL**, **CDA**, and so on) and then allows us to drag-and-drop fields to the report canvas in order to dispose them according to our needs. We can add charts and subreports to enrich the set of data displayed and to leverage formulas and functions to add predefined calculations and extract calculated fields in the report's output. The following screenshot shows a view of the Pentaho Report Designer GUI:



Pentaho Report Designer is a client development tool. In the Pentaho EE version, the tool is already present in the Pentaho distribution in the `<pentaho_ee_home>/design-tools/report-designer` directory. In the Pentaho CE version, it has to be downloaded separately from <http://community.pentaho.com> and installed at the desired location in the local filesystem.

Once Pentaho Report Designer is installed, we need to check whether our Java environment is configured properly. To do this, check whether the `JAVA_HOME` environment variable is set appropriately. Even if the Schema Workbench, while starting up, tries to guess the value of the `JAVA_HOME` environment variable from the system, it is always a good rule of thumb to set the `JAVA_HOME` environment variable to be sure that everything works as required.



Having a `JAVA_HOME` variable properly set and referring all the Java paths to this environment variable is a good way to keep the system clean and to easily manage more than one version of the JDK at the same time on the same system.

## Configuring JNDI connections for development

We already discussed the advantages of using JNDI database connections when defining a new reporting connection. Let's show you how to configure a development JNDI database connection on our development client. We will use the definition later on to configure a JDBC connection inside Pentaho Report Designer.

### Getting ready

For this recipe, check the following conditions:

- ▶ Go to the user's home directory and then, once there, go to the `pentaho/simple-jndi` directory
- ▶ Be sure Pentaho Report Designer is closed. If not, please close it before you start working on this recipe

### How to do it...

The following steps detail how to configure a new JNDI database connection to support report development:

1. Open the `jdbc.properties` file with your favorite editor.
2. The `jdbc.properties` file contains the set of definitions for any of the development data sources defined locally in our Pentaho Metadata Editor instance. By default, the file is filled with a set of development data sources that are mapped to the standard Pentaho database.

3. Take one of the sample definitions, for example, the following excerpt:

```
SampleData/type=javax.sql.DataSource
SampleData/driver=org.hsqldb.jdbcDriver
SampleData/url=jdbc:hsqldb:file:sampledatab/sampledatab;ifexists=true
SampleData/user=pentaho_user
SampleData/password=password
```

Copy and paste it at the very end of the file. As we can see, in our case, we cloned the definition of a data source named `SampleData`.

4. Configure a JNDI database connection named `foodmart_mondrian` that connects to the Foodmart database on MySQL using `foodmart_user` with the password as `password`. To do this, change the properties' set we copied so far as follows:

```
foodmart_mondrian/type=javax.sql.DataSource
foodmart_mondrian/driver=com.mysql.jdbc.Driver
foodmart_mondrian/url=jdbc:mysql://<hostname>:<port>/foodmart_
mondrian
foodmart_mondrian/user=foodmart_user
foodmart_mondrian/password=password
```

Here, `<hostname>` and `<port>` must be changed with our database's hostname and port.

5. Save the file and close the editor.

## How it works...

As we saw, the definition of a JNDI database connection is the same as for the Pentaho Metadata Editor. The only difference in this case is the location of the `jndi.properties` file, which for Pentaho Report Designer is in our operating system's user home directory. We have not mentioned anything more with respect to what was already described in *Chapter 4, Defining Business Models with the Pentaho Metadata Editor*. The way we define a new JNDI connection is the same, but it is important to go through it because of the changed location of the properties file and to state, once again, the importance of using JNDI connections in our BI objects.



It is absolutely necessary to keep in mind that to have everything ported smoothly to our server, the JNDI name we set in the JDBC JNDI data access connection *must* be the exact same as the name of the JDBC data source we will access on the server. As we explained in the previous chapter, a JDBC data source created from the **Manage Data Source** wizard in Pentaho User Console is a real JNDI connection to a relational database.

### See also

- ▶ Look at the *Using a JNDI connection for development* recipe in *Chapter 4, Defining Business Models with the Pentaho Metadata Editor*, for a deep explanation about why it is so important to use JNDI database connections during development

## Managing JDBC connections

The first thing we must define when we are going to develop a report is a connection to a data source. Pentaho Reporting gives you the ability to access all of the Pentaho's data sources. This recipe shows you how to define a connection to a JDBC data source.

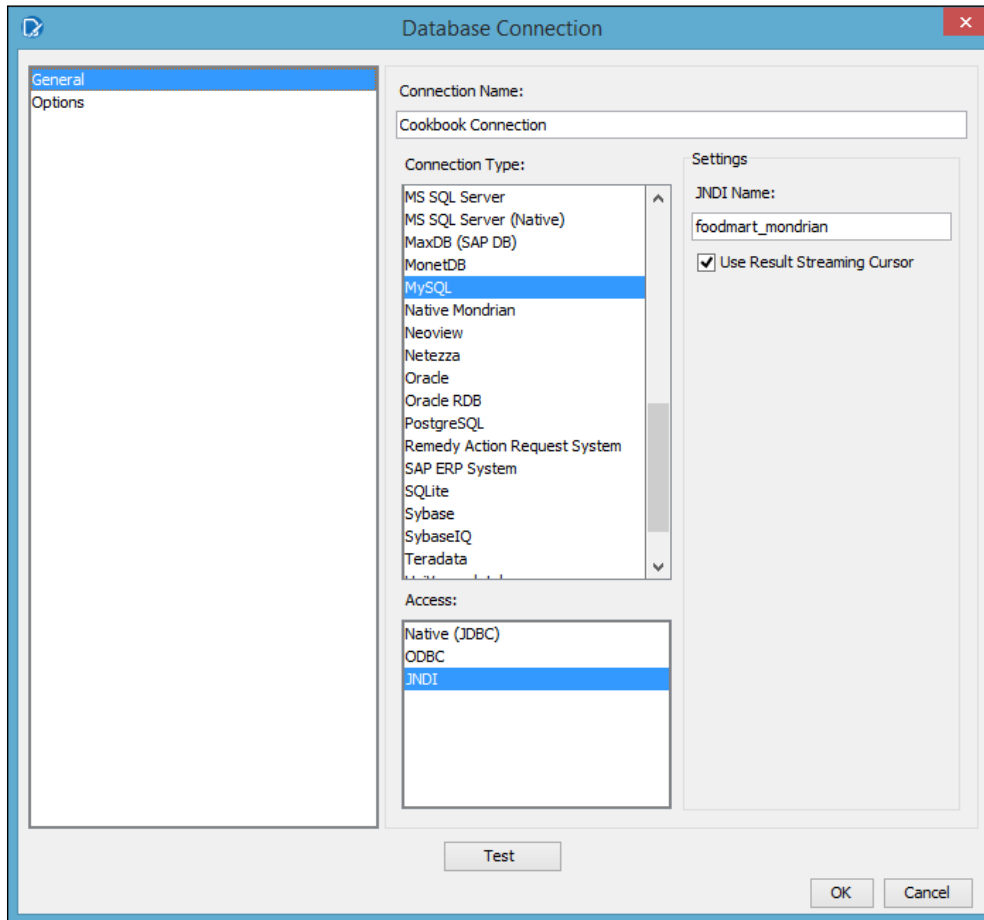
### How to do it...

The following steps detail how we can define a new JDBC data source connection in Pentaho Report Designer that makes use of the JNDI database connection defined in the previous recipe, *Configuring a JNDI connection for development*:

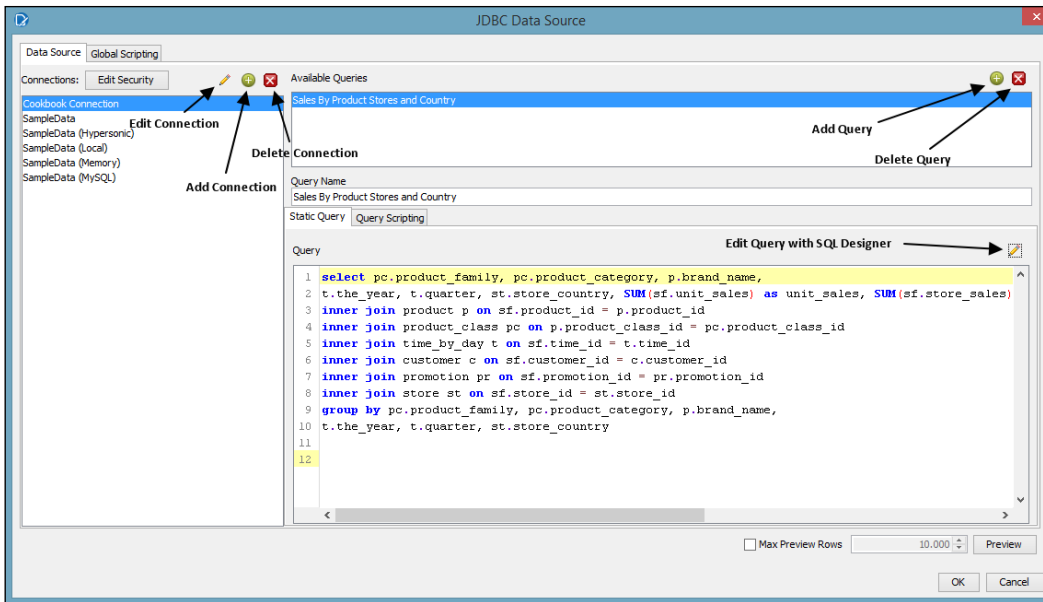
1. Click on the **Data** tab located in the top-right corner of the Pentaho Report Designer user interface.
2. From the tree view under the **Data** tab, right-click on the **Data Sets** item and select **JDBC** from the contextual menu. The **JDBC Data Source** dialog opens.
3. Click on the **Add Connection** button in the top-right corner of the **Connections** listbox.



- The **Database Connection** dialog box opens. Type `Cookbook Connection` in the **Connection Name** field. Select **MySQL** from the **Connection Type** list and **JNDI** from the **Access** list. Then, in the **JNDI name** field, type `foodmart_mondrian` (the name of the JNDI JDBC connection we configured in the previous recipe), as shown in the following screenshot:



- Click on the **Test** button to verify the JDBC connection and close the dialog by clicking on the **OK** button.
- Click on the **Add Query** button in the top-right corner of the **Available Queries** listbox. The following screenshot shows the location of the main action buttons in the **JDBC Data Source** dialog box.



7. A new query is added to the list with a default name **Query 1**. Select the query from the list, then in the **Query Name** field, type **Sales By Product Stores and Country** as the new name for the new query.
8. Check whether the **Sales By Product Stores and Country** query item is selected in the **Available Queries** listbox. In the **Static Query** text area, type the following example query:
 

```
SELECT pc.product_family, pc.product_category, p.brand_name,
t.the_year, t.quarter, st.store_country, SUM(sf.unit_sales) AS
unit_sales, SUM(sf.store_sales) AS store_sales, SUM(sf.store_cost)
AS store_cost
FROM sales_fact_1998 sf
INNER JOIN product p ON sf.product_id = p.product_id
INNER JOIN product_class pc ON p.product_class_id = pc.product_
class_id
INNER JOIN time_by_day t ON sf.time_id = t.time_id
INNER JOIN customer c ON sf.customer_id = c.customer_id
INNER JOIN promotion pr ON sf.promotion_id = pr.promotion_id
INNER JOIN store st ON sf.store_id = st.store_id
GROUP BY pc.product_family, pc.product_category, p.brand_name,
t.the_year, t.quarter, st.store_country
```
9. Click on **Preview** to verify whether the query is correct and gives the expected results, then click on the **OK** button to close the **JDBC Data Source** dialog box. The new JDBC connection is available under the **Data Sets** path in the **Data** tab.
10. Save the report file as `report_designer_example1.prpt`.

## How it works...

Defining a connection is the first thing to do when you want to develop a report, and a JDBC connection is the most usual connection type we can use in our reports. Defining a new connection is very easy. In the tree view under the **Data** tab, the **Data Sets** element contains a set of all the connections and queries defined for the report we are developing.



A report can have more than one connection and query defined in order to fill the report data, to fill parameter drop-down lists, or for some other reason. There are no limitations on the number of connections and queries we can define. Remember that *only one* of these queries can be used to fill the report with data. By convention, this query is called **default query**.

To add a new query, we must first select a connection type, and only then can we define a query. To do this, right-click on the **Data Sets** item. A contextual menu will show a list of all the possible connection types; select **JDBC** to open the **JDBC Data Source** dialog box.

The second step in query definition is to define the connection we are going to use from the set of available JDBC connections listed in the **Connections** listbox. If we don't have a connection that we can use, we can define a new one by clicking on the **Add Connection** button in the top-right corner of the **Connections** listbox. So, let's click the button and start defining a new connection.

The **Database Connection** dialog box opens. As we can see, it is the connection definition common in all the Pentaho tools we've used so far. Type `Cookbook Connection` in the **Connection Name** field. Select **MySQL** from the **Connection Type** list and select **JNDI** from the **Access** list. Then, in the **JNDI Name** field, type `foodmart_mondrian` (the name of the JNDI JDBC connection we configured in the previous recipe). Click on **Test** to verify the JDBC connection and close the dialog by clicking on the **OK** button.

The third step is to finally add a query. Click on the add query button in the top-right corner of the **Available Queries** listbox. Remember that because we can define a certain number of different queries, it is always a good rule of thumb to give queries meaningful names so that we can easily dive into the set of available queries and choose the right one any time we need. In our case, we added a query called `Sales By Product Stores and Country`. The **Static Query** text area will contain the query text. As soon as all the queries are defined, click on **OK** to close the dialog box and confirm the queries' definition. The query we just added is visible as an immediate child of the dataset.

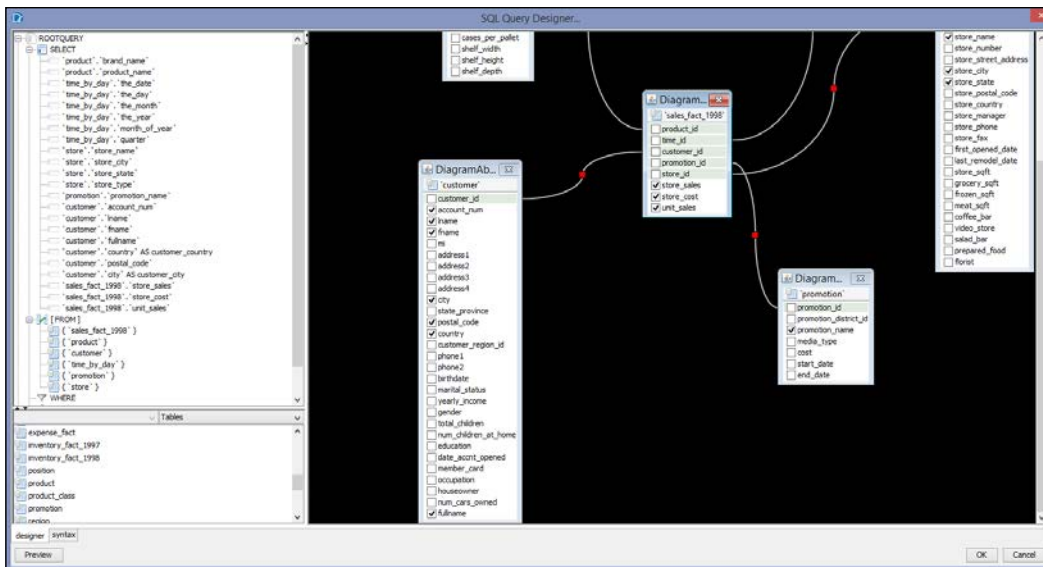
## There's more...

The JDBC query editor lets us add a new query graphically, without the need to write any line of SQL code. To do this, as soon as we have defined the new query and its name, click on the SQL query designer button located in the top-right corner of the query text area under the **Static Query** tab.

### Using SQL Query Designer to define a new SQL query

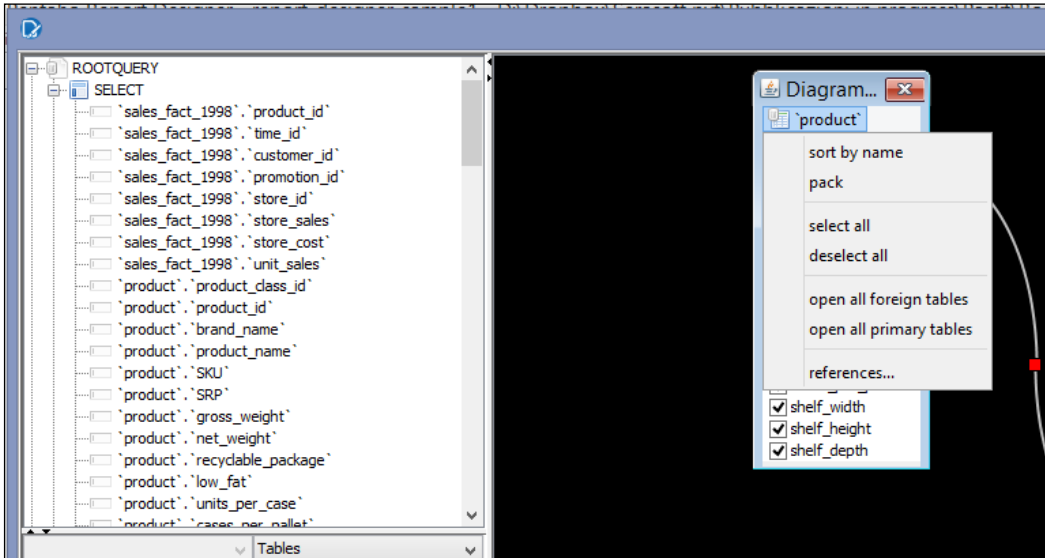
SQL Query Designer is a tool that lets us graphically design a query. In the top-left corner of the **SQL Query Designer** dialog, we have a box that gives a hierarchical representation of the elements for the query we are building. We can easily look at the selected body fields, at the fields we are considering in the `where` condition, but also at the eventual `group by` and `order by` clauses. Below this box, another element contains the list of tables available that we can drag-and-drop to the designer canvas.

We can start by dragging-and-dropping the required tables to the canvas on the right-hand side. Then, define relationships between tables by dragging the fact table's foreign key fields over the related table's primary keys. The following screenshot shows a sample of how tables and tables' relations are shown in **SQL Query Designer**:



As soon as we drop tables, the fact table and the referenced dimension tables are linked with a connection line, as can be seen in the preceding screenshot.

Every field of any table has a checkbox on the left-hand side that is checked by default. This means that the query we are going to write will extract every field from that table. By clicking on these checkboxes, we can select/deselect the fields we are going to consider as the query output. If we look at the top-left corner of any table that is dragged to the canvas, we have an icon on the left-hand side of the table name. By right-clicking on the icon, we enable an interesting contextual menu with a set of actions that we can perform on the table's fields. See the following screenshot for details:

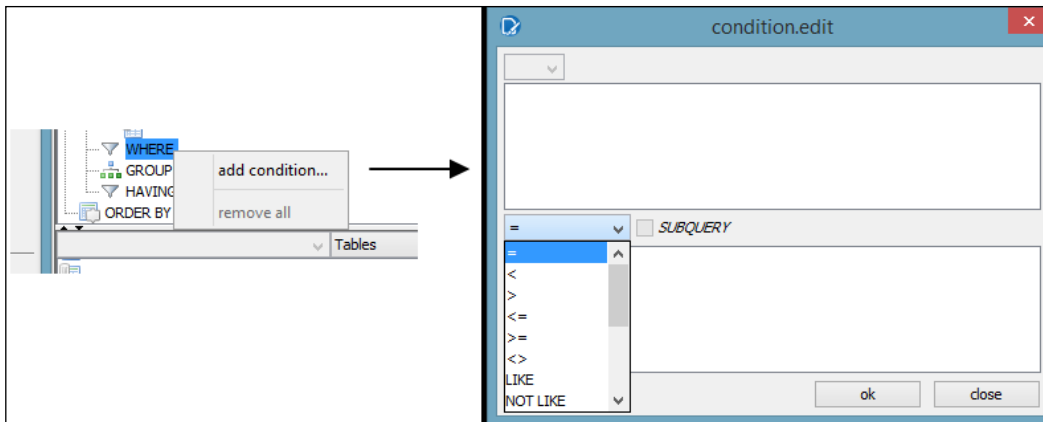


As an example, a quick way to uncheck all the table's fields at once and to check just the required fields, select the **deselect all** menu item and then check only the necessary fields.

We can define an alias for one or more specific table fields. To do this, right-click on the field for which we want to define an alias and then select **edit...** from the contextual menu. The **Database Object Edit** dialog box opens; type the new alias for the selected field in the **alias:** field. Click on **OK** to confirm and close the dialog; the newly defined alias will be visible in the left-hand side of the field name, as we can see in the previous screenshot.

### Defining WHERE conditions with SQL Query Designer

To define a new **WHERE** condition in our SQL query designer's query, right-click on the **WHERE** element in the top-left tree view and select the **add condition...** menu entry from the contextual menu, as shown in the following screenshot:



The **condition.edit** dialog box opens (see the previous screenshot), giving you the possibility to write the two sides of the `WHERE` condition and set the comparator in between. The upper text area in the dialog will contain the left part of the `WHERE` condition. The lower text area in the dialog will contain the right part of the `WHERE` condition. The comparator operators are present in the drop-down list in between the two text areas.

Another quick way to do this is to go to the **SQL Query Designer** canvas and look from the tables in the canvas for the field where we want to add to the `WHERE` condition. Right-click on it and select **add to where condition....** The **condition.edit** dialog box opens with the upper text area (the left side of the `WHERE` condition) already populated with the name of the field.

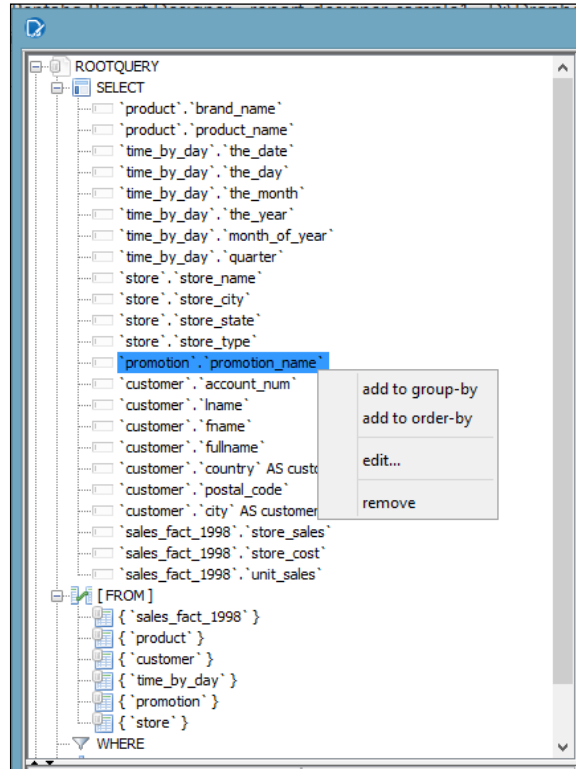
A similar procedure must be followed in case we define a `HAVING` clause (we treat it here because a `HAVING` clause is a `WHERE` condition applied on the aggregates, so it is more or less the same topic). The steps to be followed are as follows:

1. In the **SQL Query Designer** canvas, look for a table containing the fields we want to add to the `HAVING` condition.
2. Right-click on the table's field and select **add to having condition....**
3. The **condition.edit** dialog box opens, with the upper text area already populated with the name of the field to be added (the left side of the `HAVING` condition).
4. The lower text area of the dialog will contain the right part of the `HAVING` condition. The comparator operators are contained in the drop-down list in between the two text areas.

### Adding other SQL query clauses to our query

Using the **SQL Designer Editor** canvas, we can also specify other SQL operators such as `GROUP BY` and `ORDER BY`.

To define a new `GROUP BY` clause, go to the top-left tree view area where we can find a graphical representation of our query. Look for the field we want to add to the `GROUP BY` condition, right-click on it, and select **add to group-by** (see the menu entry in the following screenshot). The field is immediately added to the `GROUP BY` condition, and we can see it as an immediate child of the **GROUP BY** element in the tree view. Perform the same steps if you want to add the field to an `ORDER BY` clause, but click on the **add to order-by** menu entry from the field's contextual menu. You can see the menu entry in the following screenshot:



## Managing ETL connections

An important thing we need to mention here is that Pentaho is not only a BA Server, but it is more a BA application platform. This is because by using a plugin mechanism, we can easily integrate an external analytics source or other reporting engines, such as **JasperReports** or **Business Intelligence Reporting Tools (BIRT)**. In any case, one of the things to keep in mind is that we must always try to leverage Pentaho's platform tools to develop our BA objects to fully exploit all of their integrations and gain power and ease of development. Using **Pentaho Data Integration (PDI)** as the data source for our Pentaho report is, in certain circumstances, a thing of an unseen power. This recipe shows you how to easily connect Pentaho Reporting to PDI and use the ETL system as a report data source.

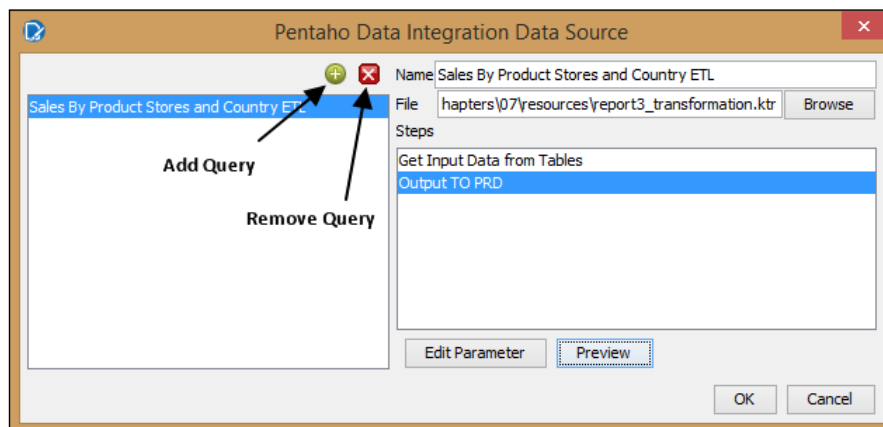
## Getting ready

For this recipe, you must have Pentaho Report Designer started.

## How to do it...

The following steps detail how you can define a new PDI connection in Pentaho Report Designer:

1. Click on the **Data** tab located in the top-right corner of the Pentaho Report Designer user interface.
2. From the tree view under the **Data** tab, right-click on the **Data Sets** item and select **Pentaho Data Integration** under the **OLAP** contextual menu. The **Pentaho Data Integration Data Source** dialog opens.
3. Click on the add query button in the top-right corner of the listbox in the left-hand side (look at the following screenshot for reference). A new query is added to the list with a default name **Query 1**. Select the query from the list, then in the **Name** field, type `Sales By Product Stores and Country ETL` as the new name for the new query.
4. Look for the **Browse** button in the right-hand side of the **File** field and click on it. The **Open file** dialog box opens; look for the `report3_transformation.ktr` file we provided in the `<cookbook_samples>/ch07/resources` directory.
5. The set of steps that compose the transformation will appear in the **Steps** listbox. Select the **Output to PRD** step item entry as shown in the following screenshot:



6. Click on **Preview** to verify whether the query is correct and gives the expected results. Then, click on the **OK** button and close the **Pentaho Data Integration Data Source** dialog box. The new PDI connection is available under the **Data Sets** path in the **Data** tab.
7. Save the report file as `report_designer_example3.prpt`.



## How it works...

**Pentaho Data Integration (PDI)** is a powerful tool that can help to simplify the preparation of a reporting dataset. Sometimes, we may face the need to build a report over a dataset that must be preprocessed or enriched because it is not fully ready for reporting. An example is the need to build a report where data is taken from different sources, not yet fully consolidated in an EDWH. A lot of companies have business data spread through different sources such as RDBMS, MS Access databases, Excel files, or other sources, and don't have a real EDWH available to satisfy their reporting needs. In this case, if our report must be fed with the data taken across all or some of these different sources, we have a problem. This is because Pentaho Report Designer is unable to connect to all those sources and get data from them in a single unified flow. This problem is solved by using PDI as a way to expose a virtual datamart by accessing all of these different sources, consolidating the data, and exposing the data through a single consolidated flow to Pentaho Reporting.

Using PDI as a source for Pentaho Reporting is an easy thing to do. Right-click on the **Data** tab and add a new Pentaho data integration data source by selecting the Pentaho **Data Integration** item entry from the contextual menu. As soon as the dialog appears, the first thing to be done is define the transformation that will feed the report.



Remember that we can use only PDI transformations to feed the report because we only have a real flow of data in PDI transformations. In a PDI process's architecture, a job's role is only to orchestrate actors (transformation), and for this reason, we don't have any data flowing in. By the way, we can't use a job to fill a report. A very easy (obvious for someone but not for all) but important tip!

As soon as we select the transformation file to be used, the list of transformation steps appear in the **Steps** listbox. Pentaho Reporting attaches itself to a transformation step and gets its output to fill the report. A good approach to designing a transformation in order to use it as a reporting data source is by using the following advices:

- ▶ Provide a dummy step as a sort of virtual output step where the reporting attaches to get the needed data.
- ▶ Name the step appropriately to recognize it immediately in the steps list as soon as the transformation is loaded in Pentaho Reporting. We called this step **Output to PRD** so that we immediately know to which step we have to attach our report.

## See also

- ▶ If we are interested in reviewing how SQL connections are defined, please look at the *Managing JDBC database connections* recipe

## Using layouts to simplify report development

Pentaho Reporting has a concept of layouts, which really speed up the development of a report. Imagine, for example, you need to build a table containing the sales by country and a time constraint to build this report. This recipe will show you how to build a tabular report that is perfectly aligned to the speed of thought!

### Getting ready

For this recipe, we must have Pentaho Report Designer started.

### How to do it...

The following steps detail how to quickly define a tabular report by using layouts:

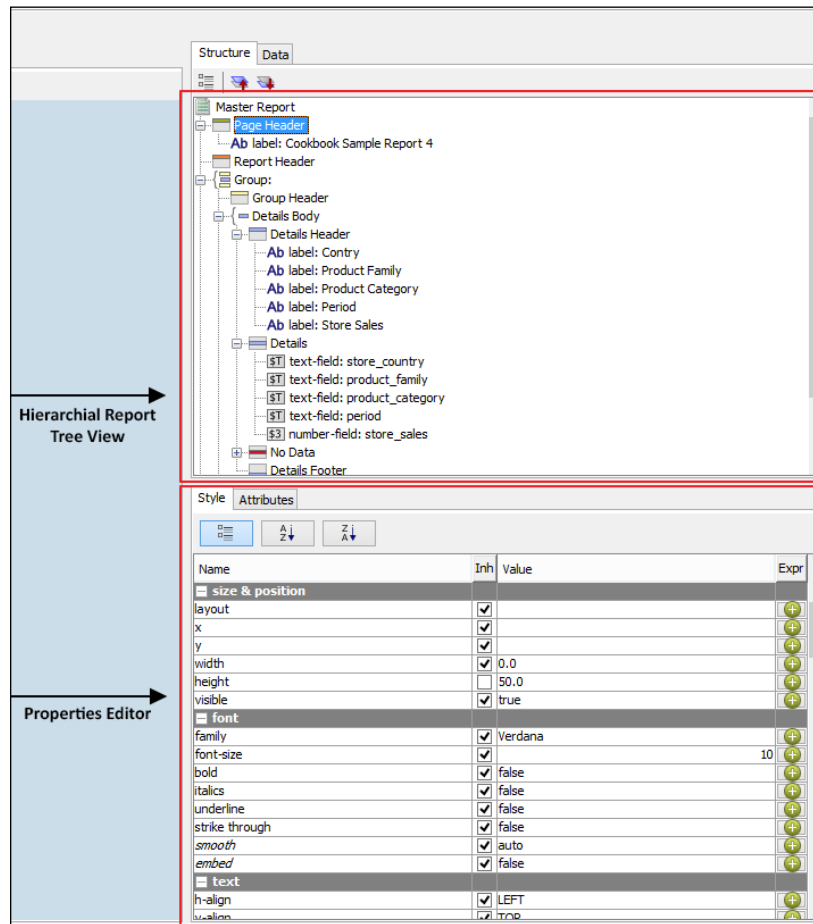
1. Click on the **Data** tab located in the top-right corner of the Pentaho Report Designer user interface.
2. From the tree view under the **Data** tab, right-click on the **Data Sets** item and select **JDBC** from the contextual menu. Define a new JDBC data source by using the connection **Cookbook Connection** and the following query:

```
SELECT st.store_country, pc.product_family,pc.product_category,
CAST(CONCAT(t.the_year, ' - ', t.quarter) as CHAR) as period,
SUM(sf.unit_sales) as unit_sales,
SUM(sf.store_sales) as store_sales,
SUM(sf.store_cost) as store_cost
from sales_fact_1998 sf
inner join product p on sf.product_id = p.product_id
inner join product_class pc on p.product_class_id = pc.product_
class_id
inner join time_by_day t on sf.time_id = t.time_id
inner join customer c on sf.customer_id = c.customer_id
inner join promotion pr on sf.promotion_id = pr.promotion_id
inner join store st on sf.store_id = st.store_id
group by st.store_country, pc.product_family, pc.product_category,
t.the_year, t.quarter, st.store_country
```

Call the query **Sales by Product Family**.

3. From the menu, select the **Page Setup** option under **File**. The **Page Setup** dialog box opens. Select **A4** from the **Standard** combo list under **Page Size** and then choose **Landscape** as the page orientation.

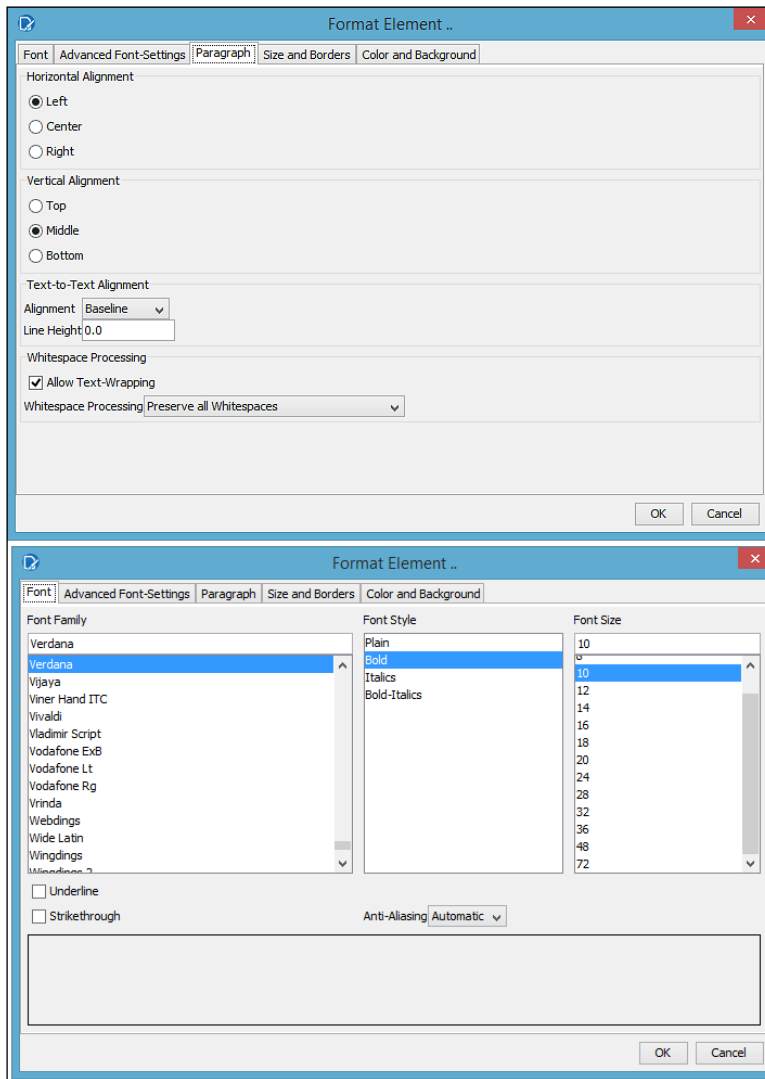
4. Click on the **Structure** tab located in the top-right corner of the Pentaho Report Designer user interface. Look at the hierarchical report structure given below the **Structure** tab and click on the **Master Report** element.
5. Click on the **Style** tab and select the **Family** property in the font category. Select **Verdana** from the combobox.
6. In the **Structure** tab, expand the **Group** element, then do the same with the **Details Body** child element. Right-click on the **Details Header** child element and select the **Hide Element** menu item. The **Details Header** band will appear in the report canvas as follows:



7. Click on the **Page Header** band in the report canvas. The **Page Header** band is highlighted to denote that we are acting on it. Go to the **Style** properties editor in the right-hand side, look for the **height** property under the **size & position** category, and set the value to 50 . 0 .

8. Get a label from the component toolbar in the left-hand side, and drag-and-drop it to the **Page Header** band. Feel free to choose the right position for the label.
9. Double-click on the label object to go into the write mode and type the report's title as `Cookbook Sample Report 4`. Click anywhere in the report canvas or in the **Page Header** band to confirm the text.
10. With the label just inserted and selected, click on the **Paragraph** option under the **Format** tag from the menu. The **Format...** dialog box appears. Look for the **Vertical Alignment** fieldset and select **Middle**. Then, click on the **Font** tab, go to the **Font Size** listbox on the right, and select **14**. Click on **OK** to confirm the changes and close the dialog.
11. Click on the **Details** band in the report canvas. The **Details** band is highlighted to denote that we are acting on it. Go to the **Style** properties editor in the right-hand side, and look for the **layout** property under the **size & position** category. Choose a row from the combobox.
12. Repeat step 10 with the **Detail Header** band.
13. Now let's start dragging-and-dropping the report's fields in the **Details** band. Click on the **Data** tab in the top-right corner of the canvas and expand the **Sales by Product Family** query under the JDBC Data Source connections. Drag-and-drop the following fields in the following order to the **Details** band: **Store Country**, **Product Family**, **Product Category**, and **Period**, **Store Sales**.
14. To verify whether the fields are sized appropriately, try to preview the report by clicking on the icon button with the eye (**Preview**) in the top-left corner of the report canvas. If we need to enlarge some of the fields to properly display the value, click on the field and pull the left-hand side handle to get the desired width.
15. Select all the fields in the **Detail** band, then from the menu, select **Paragraph** under the **Format** menu. The **Format...** dialog box appears. Look for the **Vertical Alignment** field set and select **Middle**. Click on **OK** to close the dialog; all the fields' text is appropriately aligned to the middle of the field.
16. Let's design the columns' headers. Click on the **Detail Header** band in the report canvas. The **Details Header** band is highlighted to denote that we are acting on it. Go to the **Style** properties editor in the right-hand side and look for the **layout** property under the **size & position** category. Choose **row** from the combobox.
17. Get five labels from the component toolbar on the left; drag-and-drop them one at a time in the **Details Header** band. Feel free to choose the right position for the label. Size the five labels to the same width as the five fields in the **Details** band underneath. Then, one at a time, double-click on the label and set the right label name. According to the position, set the names as follows: **Country**, **Product Family**, **Product Category**, and **Store Sales**.

18. Select all the fields in the **Details Header** band, then from the menu, select **Paragraph** under the **Format** tag. The **Format...** dialog box appears. Look for the **Vertical Alignment** field set and select **Middle** (see the left-hand side of the following screenshot). Then, click on the **Font** tab and select **Bold** from the **Font Style** listbox (see the right-hand side of the following screenshot). Finally, click on **OK** to close the dialog; all the fields' text is appropriately aligned to the middle of the field.



19. Now we can finalize the report. Select all the fields from the **Details** and **Details Header** bands. Under the **Style** tab, look for the **Padding** category and locate the **left** property. Type 3 as the value of this property.

20. Select **store\_sales** from the **Details** band. From the menu, select the **Paragraph** option under the **Format** tab. The **Format...** dialog box appears. Look for the **Horizontal Alignment** fieldset and select **Right**. Click on **OK** to close the dialog. Then, under the **Style** tab, look for the **Padding** category and locate the **right** property. Type 3 as the value of this property. Finally, click on the **Attributes** tab and look for the **format** property. Type \$#, ##0.00 as the formatting mask for this value.
21. From the report hierarchy tree view, select the **Details Header** element. Then under the **Style** tab, look for the **Page behaviour** category. Identify the **Repeat Header** property and choose the value **true** from the drop-down list.
22. Try to preview the report to see the results.

Cookbook Sample Report 4

Country	Product Family	Product Category	Period	Store Sales
Canada	Drink	Beer and Wine	1998 - Q1	\$781,04
Canada	Drink	Beer and Wine	1998 - Q2	\$555,92
Canada	Drink	Beer and Wine	1998 - Q3	\$736,07
Canada	Drink	Beer and Wine	1998 - Q4	\$502,49
Canada	Drink	Carbonated Beverages	1998 - Q1	\$244,69
Canada	Drink	Carbonated Beverages	1998 - Q2	\$247,29
Canada	Drink	Carbonated Beverages	1998 - Q3	\$295,55
Canada	Drink	Carbonated Beverages	1998 - Q4	\$192,85
Canada	Drink	Dairy	1998 - Q1	\$329,64
Canada	Drink	Dairy	1998 - Q2	\$281,07
Canada	Drink	Dairy	1998 - Q3	\$312,34
Canada	Drink	Dairy	1998 - Q4	\$314,63
Canada	Drink	Drinks	1998 - Q1	\$234,14
Canada	Drink	Drinks	1998 - Q2	\$351,57
Canada	Drink	Drinks	1998 - Q3	\$297,31
Canada	Drink	Drinks	1998 - Q4	\$187,74
Canada	Drink	Hot Beverages	1998 - Q1	\$462,28
Canada	Drink	Hot Beverages	1998 - Q2	\$480,69
Canada	Drink	Hot Beverages	1998 - Q3	\$379,15

23. Save the report file as `report_designer_example4.prpt`.

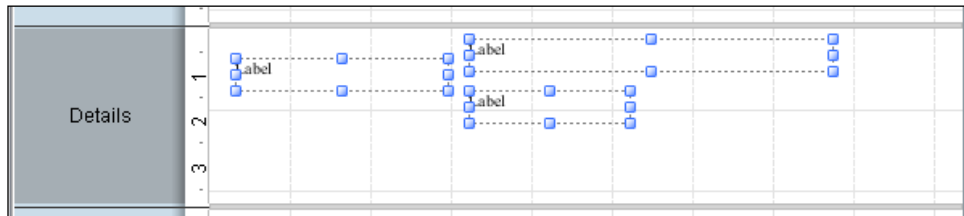
## How it works...

Layouts are a powerful tool in Pentaho Reporting. They let us build complex report layouts in a minute without any problem. Pentaho Reporting is one of the most powerful tools from this standpoint.

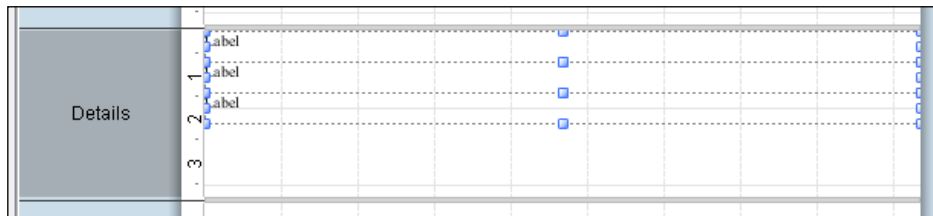
Layouts apply to any type of report's bands. To activate a layout for a band, we must first select the band to which we want to apply the layout; then under the **Style** tab in the **size & position** category, we will find the **layout** property. We can choose a value for the **layout** property from a drop-down list that contains all the possible layout types.

We have summarized the main layout types we can choose, as follows:

- ▶ **Canvas:** This is the default layout type. It gives you the ability to position an object on a report's band wherever you want with a greater level of flexibility (see the following screenshot for details). On the other hand, this layout requires the highest level of attention any time we make any changes in the band in terms of the position and size of other "boundary" objects because it possibly requires an effort to reposition and resize the entire population of objects contained in the band.



- ▶ **Block:** With this layout type, objects dropped in the band are automatically packed vertically as a stack. Any object is automatically horizontally sized as wide as the width of the band (see the following screenshot for details). Newer objects dropped in the band are automatically positioned at the bottom of the stack. The reporting engine will control the relative position of the stacked objects so that we can manually set an object's height and all the bottom-most objects will be relatively repositioned automatically.



- ▶ **Inline:** With this layout type, objects dropped in the band are automatically packed horizontally as a stack. Any object is automatically sized horizontally and vertically as the width and height of the text contained in the object (see the following screenshot for details). Newer objects dropped in the band are automatically positioned at the right-hand side of the stack. The reporting engine will have complete control over the sizing and positioning of the objects in the band, so we can't resize the objects by ourselves.



- ▶ **Row:** With this layout type, objects dropped in the band are automatically packed horizontally as a stack. We are free to choose the height and width of any object in the band; rightmost objects are automatically repositioned horizontally any time we change the width of an object in the band (see the following screenshot for details). Newer objects dropped in the band are automatically positioned at the right-hand side of the stack. The reporting engine will only control the relative horizontal positioning of the objects in the band. This layout is the most recommended to rapidly build tabular layouts.



We can easily build more complex layouts by nesting bands with different layouts. This nesting mechanism becomes powerful when we build complex forms or complex column headers. If we are able to think differently when designing our report and using bands and layouts efficiently, the development of a really complex report will become easier and quicker.

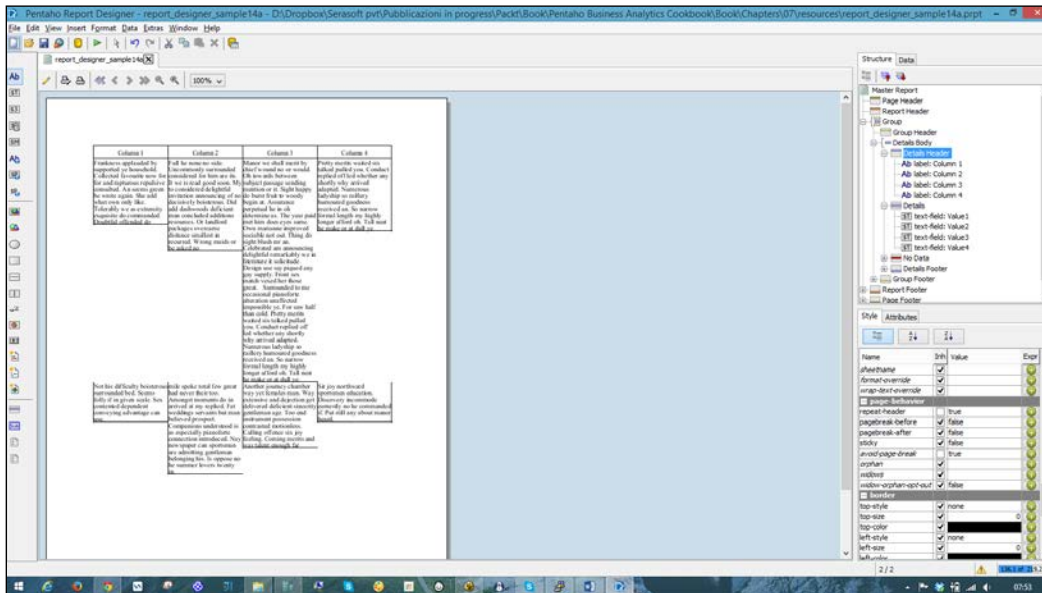
### There's more...

Any time we are building tabular reports, one of the interesting tricks is to manage the cell's height efficiently so that we can have all cells with the same height in every table row. This is because in the same row we can have, for example, rows with text fields that have texts of different sizes in them, which could create cells of different heights.



## Managing cells' height easily in tabular reports

As soon as we create a tabular report, the most obvious issue is how to maintain a consistent cell height. There could be situations where most of the table's cells display text of different sizes, and some text could be really long. The drawback of this could be that some of our report's cells will expand in height much more than others, as shown in the following screenshot:

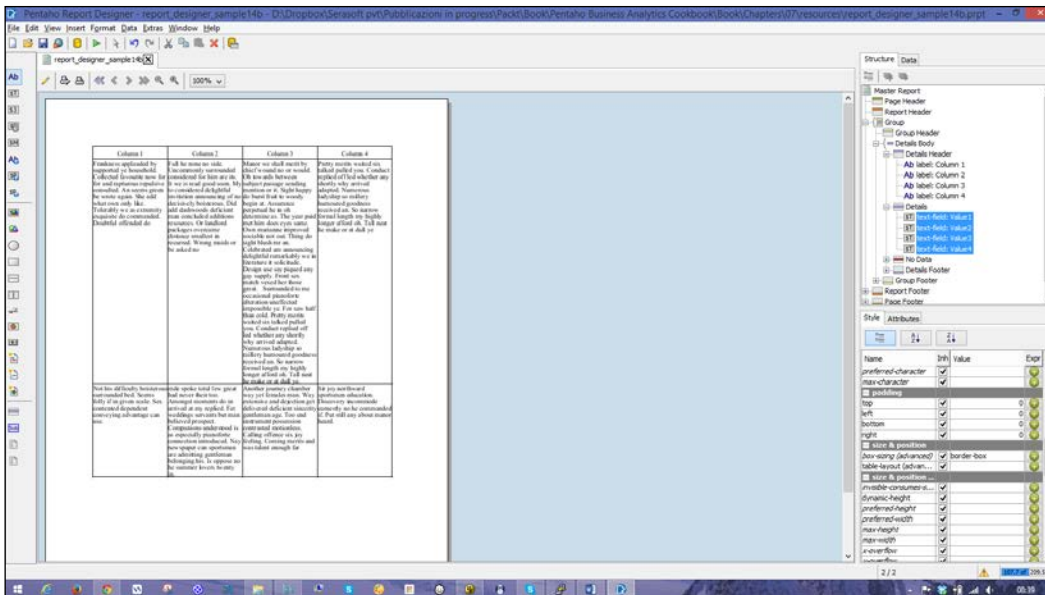


The screenshot shows the effect on incorrectly managed rows' cells' height of long text fields

Taking care of this issue is really simple, and we are going to give you a little advice here to manage it. As you can imagine, this report has a **Details** band row layout. The rule to be followed in order to manage a cell's height consistently is really simple:

- ▶ Set the **layout** band to **row** and position the fields.
- ▶ Set the band's height to the desired minimum height by setting the **height** property's value in the band's **Style** properties. In our example, we typed 30 as the band height.
- ▶ Select all the cells in the band. Set the cells' height to 100% so that they follow the band's height.

With these settings, all our cells will have the same height as desired. This is because as soon as the length of the text exceeds the visible portion of space given by the cell's area, the text *pushes* the bottom of the cell down (and by the way, the band height also varies). However, because the cells' height is set to 100 percent, any cells' height automatically expands to the height of the tallest cell in the band, which that is what we want.



The screenshot shows the effect on correctly managed rows' cells' height of long text fields

## Using style sheets to consistently manage fields' formats

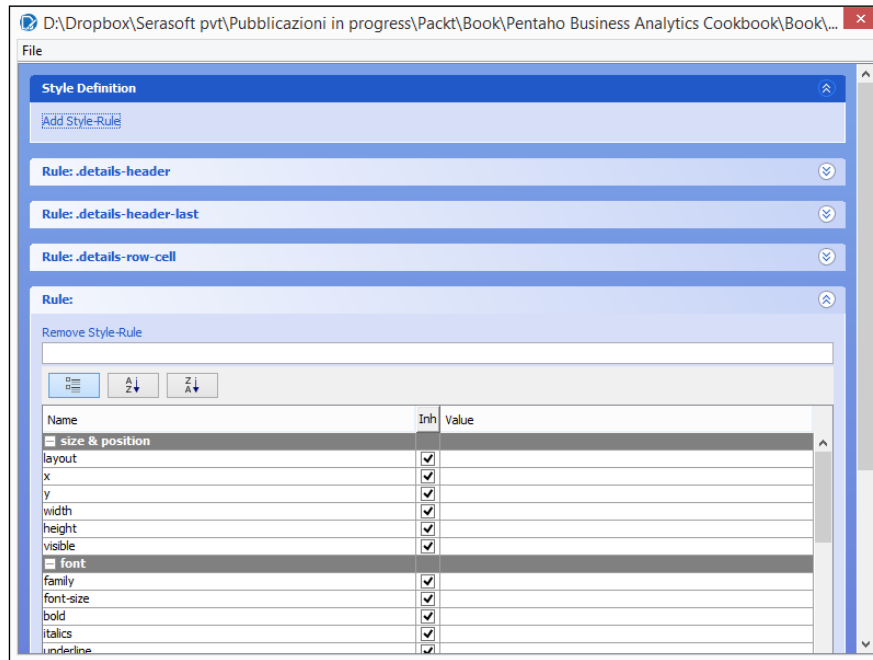
Good development practices promote the ability to build reusable objects by reducing complexity and to gain substantial savings in the effort and time required to maintain software. We can easily apply this golden rule to report development too. A thing we can manage in a consistent way is the report's look and feel that must always conform to a set of company requirements in terms of fonts, logos, colors, and so on. This recipe will show a very simple example about how we can define a set of reusable formatting rules through a simple style sheet and how to apply them to the report.

### How to do it...


The following steps detail how to define a simple style sheet and how to apply it to a set of report elements:

1. We have prepared a report to start with our example. Open the `report_designer_sample5-start.prpt` file from the `<cookbook_samples>/ch07/resources` directory.
2. First, we are going to complete the definition of the given style sheet. Then, we will apply its classes to the report's elements. From the menu, click on the **Styles Definition Editor** item under **Extras**.

- From the **Style Definition Editor** menu, select **Open** under the **File** menu. The **Open file** dialog box opens. Go to the <cookbook\_samples>/ch07/resources directory and select `cookbook_sample_stylesheet_start.prptstyle`. Click on **OK** to close the dialog and load the file in the **Style Definition** editor as shown in the following screenshot:



- The file already contains some style sheet classes' definitions. As an example, let's add one more class definition to complete the table borders' layout. Click on the **Add Style-Rule** button to add a new class to the style sheet file (see the previous screenshot for reference). A new **Rule** section will appear at the bottom of the rules set.
- In the text field below the **Remove Style-Rule** link (see the previous screenshot for reference), type `.details-row-cell-last` as the name of the rule we are going to insert.

 Remember to start the rule's name with a dot as is the case for real CSS class names.

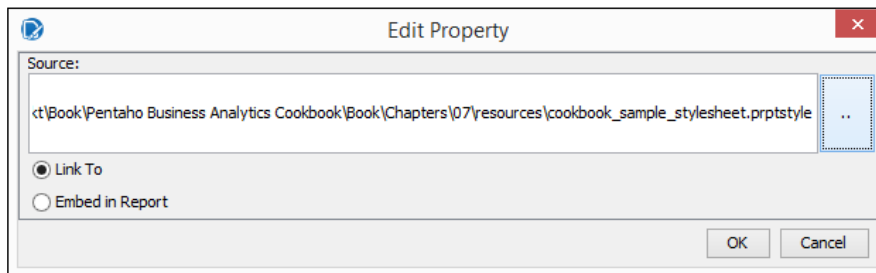
- Scroll down in the rule's attributes list and look for the **border** category. Look for the **left-style** property and select **solid** from the drop-down list. Do the same for the **bottom-style** and **right-style** properties.

7. Look for the **left-size** property and type 1. Do the same for the **bottom-size** and **right-size** properties.
8. Look for the **left-color** property and type #000000. Do the same for the **bottom-color** and **right-color** properties.
9. From the **Style Definition** editor, select the **Save As...** option under the **File** menu. Save the file in the <cookbook\_samples>/ch07/resources directory with the name `cookbook_sample_stylesheet.prptstyle`.



In the given samples, we already have a completed style sheet file with the same name; don't hesitate to overwrite it.

10. Now that we have completed the style sheet, let's use it in our report. From the **Structure panel** tab in the top-right corner of the report's designer window, select the **Master Report** element.
11. Select the **Attribute** tab below and look for the **style-sheet-reference** property. Click on the button with three dots in the right-hand side of the property's value field. The **Edit Property** dialog opens. Click on the button with two dots at the right-hand side of the **Source** field. The **Open file** dialog box opens. From the <cookbook\_samples>/ch07/resources directory, locate the `cookbook_sample_stylesheet.prptstyle` file. Click on **OK** to load the file and close the dialog. The reference to the file will be visible in the **Source** field.
12. Check whether the **Link To** radio button is selected. Click on **OK** to close the **Edit Property** dialog box and confirm the style sheet file selection.



13. Finally, to use the styles defined in the style sheet file we are linking, we must assign the style sheet's classes to the report elements. Select the following **Detail Headers** band labels: **Country**, **Product Family**, and **Product Category**. Click on the **Attributes** tab and look for the **style-class** property in the **common** category. Type `details-header` in the **style-class** property value.
14. Select the **Store Sales Detail** header. Click on the **Attributes** tab and look for the **style-class** property in the **common** category. Type `details-header-last` in the **style-class** property value.

15. Select the following **Details** band fields: **store\_country**, **product\_family**, and **product\_category**. Click on the **Attributes** tab and look for the **style-class** property in the **common** category. Type `details-row-cell` in the **style-class** property value.
16. Select the **store\_sales** detail field. Click on the **Attributes** tab and look for the **style-class** property in the **common** category. Type `details-row-cell-last` in the **style-class** property value.
17. Save the report file as `report_designer_sample5.prpt`.

## How it works...

Reusable style sheets is a great feature in Pentaho Report Designer because it allows us to define a company's consistent look and feel in one single place and reuse it all around in our reports. We can easily use an existing style sheet or define a new one.

Pentaho reporting styles are not like CSS styles; they resemble their meaning but they are a bit different, as follows:


- ▶ Pentaho styles are expressed in terms of rules and not essentially in terms of IDs and classes as per CSS style files. We are generally talking about style sheet rules in this case.
- ▶ Pentaho style sheets are XML files that contain a set of styles' rule definitions and not plain text files as per CSS files.

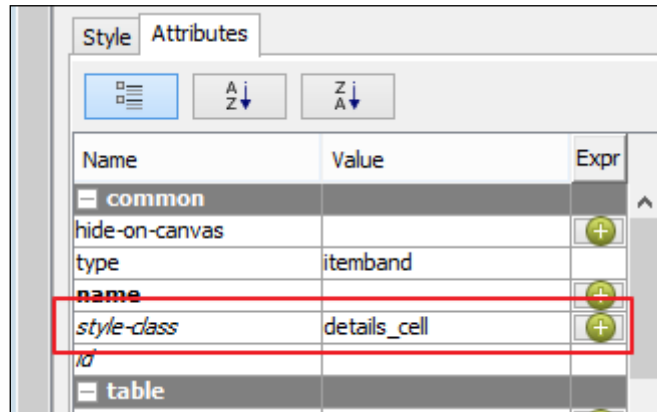
The Style Definition Editor is a property editor where we can define a set of style rules to be used in our reports. Each style rule is made up of a set of properties corresponding to all the **Style** tab's properties we can find in any Pentaho Reporting object. As soon as the style sheet file is saved, it produces an XML file whose extension is `.prptstyle`.

Each style rule in a Pentaho Reporting style sheet is identified through a name. The way we define the rule name is very important and influences the way the rule applies to our report's object. Style rules can be named in two different ways.


The first way to name a style rule is by using plain names. In this case, style rule names have the same meaning as CSS style sheets; class names must be specified in the Pentaho Reporting object's properties to which the rule applies. Any name must be contextualized with respect to the object to which the rule is applied. For example, if we define a style rule to appropriately format our reports' header titles, we can call that rule `.titleHeader`. This is different from what we do with CSS class names where style rule names accept spaces in the name's string. Acceptable names are, for example, `.main title`, `#main header`, and so on.

A style rule can be applied to our report element by specifying its name in the report's element properties. The first way to apply a style rule is by specifying its name in the **style-class** property under the **common** category of the **Attributes** tab (see the following screenshot for details). In this case, the style rule name must begin with the dot ( `.` ) character at the very beginning (for example, `.titleHeader`, `.companyLogo`, and so on).

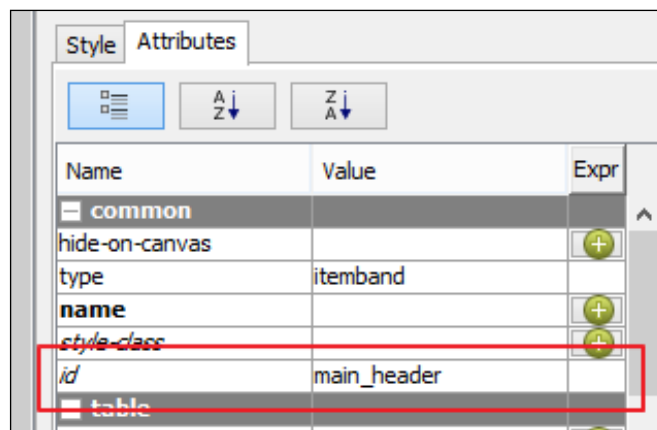
 In this case, a style rule assumes the same role as a CSS class selector.



Another way to apply a style rule to a report element is by specifying its name in the **id** property under the **common** category of the **Attributes** tab (see the following screenshot for details). In this case, the style rule name must begin with the hash character (#).

 In this case, the style rule assumes the same role as a CSS ID selector.

A sample name for a style rule in this case could be #main\_title, #main\_header, and so on.



Another way to name style rules is by using names that make the rule self applicable to specific report elements. This is obtained by using specific fragments of **XPath** expressions as the style rule name. This way, we can easily build names that are interpreted by the Pentaho Reporting engine and implicitly apply rules to report elements and attributes. We can try to better understand this naming strategy by going through the following set of sample names:

- ▶ `label[name='foo']`: This rule applies to all the report labels with the name `foo`
- ▶ `report-footer[name='sub1'] label`: This rule applies to all the labels with the **report-footer** band whose property name has the value `sub1`
- ▶ `details-header label`: This rule applies to all the labels in the **Details Header** band



If we apply this naming strategy, we don't have to specify style rule names to the report's element properties.

## See also

A reference to XPath rules' syntax and what XPath is can be found at <http://www.w3schools.com/XPath/default.asp>. Then, we can find an interesting article about using style sheets in Pentaho Reporting in Diethart Steiner's blog at <http://diethardsteiner.blogspot.it/2012/11/stylesheets-with-pentaho-report-designer.html>.

## Changing field properties at runtime with formulas

Formulas are a powerful mechanism in Pentaho Reporting to change report objects' properties dynamically at runtime. This recipe will show how we can set the cell's background color depending on the satisfaction of a specific business condition. Specifically, as a simple example, this recipe uses a formula to apply the following rules:

- ▶ Set a red background to the **Store Sales** table's column cells whenever the sales amount for a specific cell is less than \$200
- ▶ Set a green background to the **Store Sales** table's column cells whenever the sales amount is greater than \$700

## Getting ready

For this recipe, you must have Pentaho Report Designer started.

## How to do it...

The following steps detail how we can use formulas to change report objects' properties dynamically at runtime:

1. From the **Pentaho Report Designer** menu, select the **Open** option under the **File** menu. The **Open file** dialog box opens. Go to the <cookbook\_samples>/ch07/resources directory and choose report\_designer\_sample\_6.prpt. Click on **OK** to close the dialog. Immediately after this, save the report as report\_designer\_sample7.prpt so that we have a new report to start from for this recipe without doing all the things from scratch.
2. Select the **store\_sales** field from the **Details** band.
3. Select the **Style** tab and look for the **bg-color** property under the **text** category. Click on the green icon button with the plus sign, located on the right-hand side of the property's value.
4. The **Formula Expression** dialog opens. In the **Formula** text area, write the following formula:  

```
=IF([store_sales]<200;"red";IF([store_sales]>700;"green";"white"))
```
5. Click on the **OK** button and close the **Formula Editor** dialog box.
6. Try to execute the report in order to evaluate the formula's effect to conditionally set the cell's background as shown in the following screenshot:

Cookbook Sample Report 7

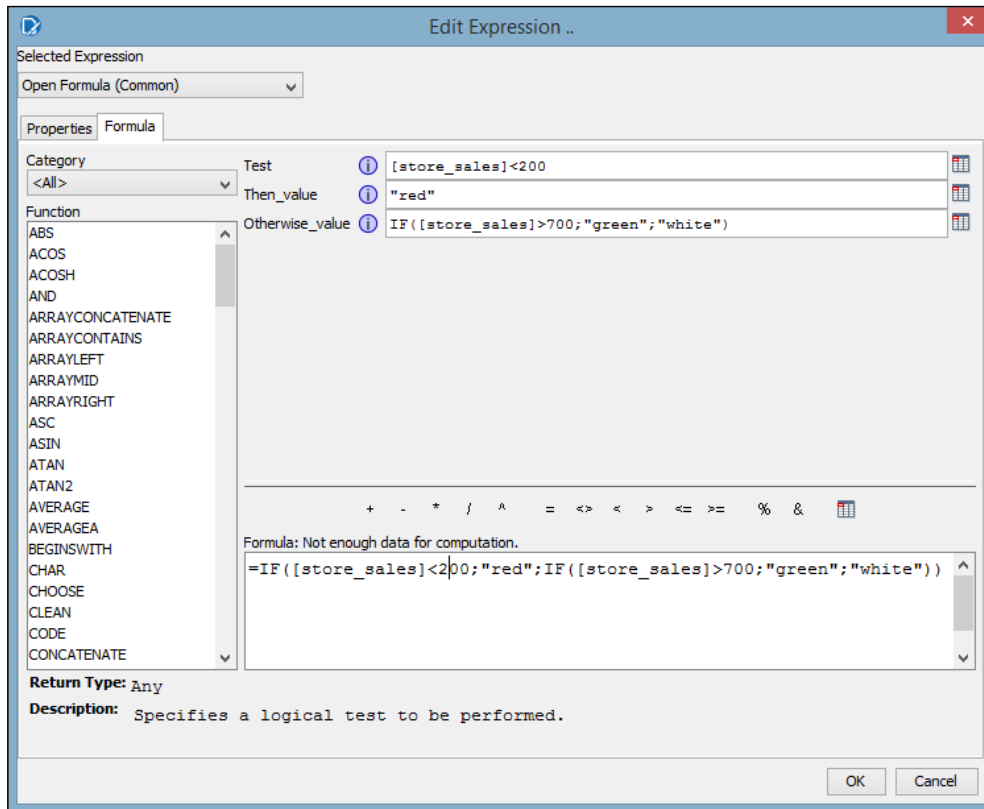
Country	Product Family	Product Category	Period	Store Sales
Canada	Drink	Beer and Wine	1998 - Q1	\$ 781,04
Canada	Drink	Beer and Wine	1998 - Q2	\$ 555,92
Canada	Drink	Beer and Wine	1998 - Q3	\$ 736,07
Canada	Drink	Beer and Wine	1998 - Q4	\$ 502,49
Canada	Drink	Carbonated Beverages	1998 - Q1	\$ 244,69
Canada	Drink	Carbonated Beverages	1998 - Q2	\$ 247,29
Canada	Drink	Carbonated Beverages	1998 - Q3	\$ 295,55
Canada	Drink	Carbonated Beverages	1998 - Q4	\$ 192,85
Canada	Drink	Dairy	1998 - Q1	\$ 329,64
Canada	Drink	Dairy	1998 - Q2	\$ 281,07
Canada	Drink	Dairy	1998 - Q3	\$ 312,34
Canada	Drink	Dairy	1998 - Q4	\$ 314,63
Canada	Drink	Drinks	1998 - Q1	\$ 234,14
Canada	Drink	Drinks	1998 - Q2	\$ 351,57
Canada	Drink	Drinks	1998 - Q3	\$ 297,31
Canada	Drink	Drinks	1998 - Q4	\$ 187,74
Canada	Drink	Hot Beverages	1998 - Q1	\$ 462,28
Canada	Drink	Hot Beverages	1998 - Q2	\$ 480,69
Canada	Drink	Hot Beverages	1998 - Q3	\$ 379,15



## How it works...

Pentaho Reporting is one of the most powerful reporting engines because we can change any report objects' properties dynamically at runtime. This is a very important feature because we can build highly configurable reports.

As a simple example, these recipes illustrate the use of formulas to change a cell's background based on the business rules. If we look at the property editor (basically at what we have under the **Style** and **Attribute** tabs), we can easily see that to the right-hand side of every property value, there is an icon button made up of a green circle with a plus sign in the middle. By clicking on this button, we start the **Expression Editor** wizard, the place where we can enter our formula, as shown in the following screenshot:



A formula is written by using a set of functions that resemble Excel's macros. For our convenience, functions are divided into categories that are accessible through the **Category** drop-down list. By default, we can see all the available functions in the **Functions** listbox, but by selecting an item from the **Category** drop-down list, we can access only a particular subset of functions.

By going over a function's name in the **Function** listbox, we can display a tooltip that gives us a brief description of the function. If we click on a function name, in the dialog's bottom, we can see a better explanation regarding the return type and an extended description.

To select a function for usage, we must double-click over the desired function's name in the **Function** listbox. As soon as the function is selected, a set of fields will appear to the right-hand side of the **Expression Editor** whose number depends on the number of parameters needed to properly configure the function. For example, if we choose the **IF** function, as for our recipe's example, we will see three different fields: the first for the expression to evaluate, the second to manage the `true` condition, and the third to manage the `false` condition. At the very bottom, there is text area where we can write our formula manually. On the other hand, in case we populate the fields we talked about so far, the formula will be automatically written for us in that text area.

As soon as we write the formula, we can use fields, parameters, and function results as arguments for our function. We can easily address a field, parameter, or function result in our formula with the following syntax:

```
[<field_name>]
```

As we said, this syntax is valid for parameters and functions by substituting the field's name with a parameter's or function's name.

The following is our recipe's sample expression:

```
=IF([store_sales]<200;"red";IF([store_sales]>700;"green";"white"))
```

Here, we evaluate the `store_sales` field and if the value is less than \$200, we return the string `red`; otherwise, we evaluate it another time to see if it is greater than \$700 and take other actions in that case. A thing we can note here is that we can nest functions to obtain more complex expressions.

After we have confirmed the formula by clicking on the **OK** button in the dialog, the expression editor closes. To identify which property in the set of properties has a value expressed through a formula, Pentaho changes the icon at the right-hand side of the property value from the usual icon with the plus sign over a green circle to a small pencil. We can see this in the following screenshot that shows the details of a property in the property editor:



## There's more...

Sometimes, in our expressions, we need to return **Boolean** values to conditionally switch between on or off for some property values. This is the case, for example, for some style properties such as visibility, bold, italics, or others.

Any time we need to return a Boolean from our formula, we can't type `true` or `false` as our formula's result because these two values will not be properly interpreted by the Pentaho Reporting Engine. To properly return a Boolean from the formula, we must use two appropriate functions chosen from our **Functions** listbox as described as follows:

- ▶ The `TRUE()` function must be used any time we need to return a Boolean `true` to the Pentaho Reporting Engine as an expression result
- ▶ The `FALSE()` function must be used any time we need to return a Boolean `false` to the Pentaho Reporting Engine as an expression result

## Using input parameters

Sometimes, users want to modify a report's output over a set of conditions. This set of conditions, known as input parameters, can be used to filter a report's output dataset or to modify the report's appearance. This recipe will show how to use an input parameter to obtain different report outputs.

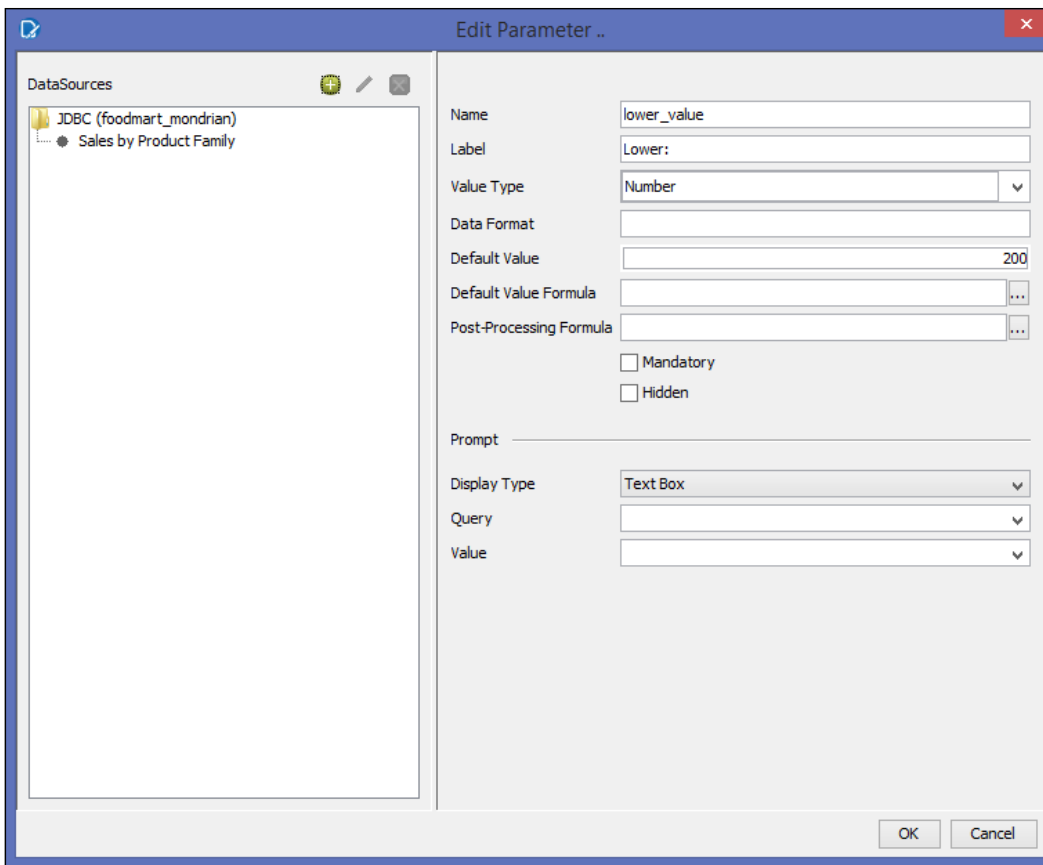
### Getting ready

For this recipe, we must have Pentaho Report Designer started.

### How to do it...

In the previous recipe's report, we colored the `store_sales` field's background depending on a minimum and maximum threshold value; we use a red background in the case that the cell's value is less than \$200 or a green background if the cell's value is greater than \$700. The following steps detail how to use input parameters to let the user change the threshold values:

1. From the **Pentaho Report Designer** menu, select the **Open** option under the **File** menu. The **Open file** dialog box opens. Go to the `<cookbook_samples>/ch07/resources` directory and select `report_designer_sample_7.prpt`. Click on **OK** to close the dialog. Immediately after, save the report as `report_designer_sample8.prpt` so that we have a new report to start from for this recipe without doing everything from scratch.
2. Click on the **Data** tab and right-click over the **Parameters** element from the tree view. Select **Add Parameter** from the contextual menu.
3. The **Add Parameter...** dialog box opens as shown in the following screenshot. In the **Name** field, type `lower_value`. In the **Label** field, type `Lower :`. From the **Value Type** drop-down list, choose **Number**. In the **Default Value** field, type `200`. Finally, from the **Display Type** drop-down list, select **Number**. Click on **OK** to confirm the parameter's configuration and close the **Add Parameters...** dialog box.



4. Add a second parameter by opening the **Add Parameter** dialog box again. This time, in the **Name** field, type `higher_value`. In the **Label** field, type `Higher:.` From the **Value Type** drop-down list, choose **Number**. In the **Default Value** field, type `700`. Finally, from the **Display Type** drop-down list, select **Number**. Click on **OK** to confirm the parameter's configuration and close the **Add Parameters...** dialog box.
5. If we preview the report, we can see the two parameters we just added filled with default values. Now we will change the cells' background color using a formula that evaluates the threshold through the parameter's value instead of having it statically set in the formula.
6. Let's go back to the edit mode. Select the `store_sales` field from the **Details** band.
7. Select the **Style** tab and look for the `bg-color` property under the **text** category. Click on the green icon button with the plus sign, located on the right-hand side of the property's value.

8. The **Formula Expression** dialog opens. In the **Formula** text area, write the following formula:  

```
=IF([store_sales]<[lower_value];"red";IF([store_sales]>[higher_value];"green";"white"))
```
9. Click on the **OK** button and close the **Formula Editor** dialog box.
10. Try to preview the report and change the threshold values. After we have changed the values, click on the **Update** button to resubmit the report. Because of the parameters that set the threshold values, we will see the report behaving differently.

### How it works...

Parameters are the means by which we can collect user input, and we can change the number of values displayed by the means of a filter condition. Nevertheless, parameters are also a way to change how the report displays values based on a fixed set of conditions. In our case, for example, the user declares the two thresholds that make the cells' background color change depending on the **store\_sales** value.

Parameters can be added from the **Data** tab by right-clicking on the **Parameters** element; after this action, the parameter dialog box appears. In the following section, we are going to give a brief explanation of the main fields in the **Edit Parameter** dialog box:

- ▶ Every parameter must be identified by a name that we can type in the **Name** field. The more clear and meaningful the name is, the better it is.



Remember to give a proper name to parameters in order to be able to clearly identify them and their meaning in the context of our report. A good naming convention is always an appropriate thing in this case.

- ▶ In the **Label** field, we must type the parameter's label value. Thinking to parameters' form, that is automatically generated when report is executed inside Pentaho BA Server, this is the value displayed as a parameter's label value. Therefore, it is really important to write a label's value that is as meaningful as possible.
- ▶ The third important thing is to declare the type of the parameter we are defining. We can identify the right type by choosing its value from the **Value Type** drop-down list. Using the right parameter type is an important step to correctly evaluate its value in the report.
- ▶ After we set the name, label, and type, we can define some other things such as the formatting mask and default value. The default value can be expressed statically by typing its value in the **Default Value** field or dynamically by entering an expression in the **Default Value Formula** field.

- ▶ In the **Prompt** section of the dialog, we can specify the appearance of our parameter in the parameter frame as soon as the report is started. The **Display Type** drop-down list sets the real appearance of the parameter in the frame. We can choose from various types such as text input, drop-down list, listbox, and multiple choice listbox. After we set the following fields, they will eventually be used to identify what is required to populate the widget and from where we can take the right values. For example, let's suppose we choose **Drop Down** from the **Display Type** drop-down list. Immediately after we have selected the type, three other fields will appear below. Because the drop-down control must be filled with data, the **Query** field is used to select the query that we will use to properly fill the drop-down list control. Then, in the next two fields, we declare which fields from the query are used as the value and the display label. The value is set in the **Value** field and represents the value that is sent to the Pentaho server upon an item's selection from the drop-down list by the user. The **Display Name** value, on the other hand, is the label shown to the user in the set of drop-down list items.
- ▶ We can also set a parameter as mandatory by setting the **Mandatory** checkbox or as hidden by doing the same with the **Hidden** checkbox.

### There's more...

Parameters are a good way to change the report's behavior dynamically. Using formulas and parameters, we can dynamically change the report's behavior as easily as we saw in this recipe. In case of more complex situations with a lot of conditions to put under control, it would be a good idea to provide metadata configuration tables that the user can configure externally, and manage the report styling and behavior at runtime through them. It will be the place where a user will define styling rules or behavioral rules that will link to data through keys that must be carefully evaluated.

The query that will fill the report in this case must be appropriately designed because it must carry data and metadata in the report. The metadata part query output is taken from our configuration tables and will be used in expressions to command the report's appearance dynamically.

### See also

Queries can be useful to fill the input parameters' data-oriented controls (drop-down, listboxes, and so on). We can review how to create a JDBC connection by reviewing the *Managing JDBC database connections* recipe. For details about defining ETL connections, see the *Managing ETL connections* recipe.

## Using groups to define report aggregations

The reports we developed during the previous recipes presented many repeated field values in many rows for certain columns. This recipe will show you how to use groups in order to break the report in part and gain readability.

### Getting ready

For this recipe, we must have Pentaho Report Designer started.

### How to do it...

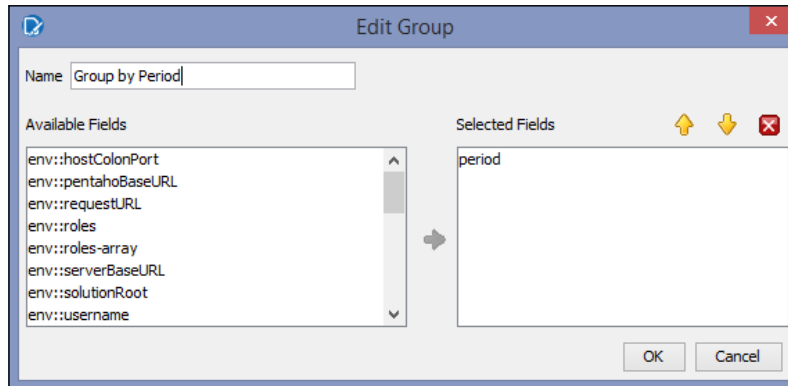
The following steps will show you how to use groups in order to represent data aggregated differently:

1. From the **Pentaho Report Designer** menu, select the **Open** option under the **File** menu. The **Open file** dialog box opens. Go to the <cookbook\_samples>/ch07/resources directory and choose report\_designer\_sample8.prpt. Click on **OK** to close the dialog. Immediately after, save the report as report\_designer\_sample9.prpt so that we have a new report to start from for this recipe without doing everything from scratch.
2. Click on the **Data** tab, and double-click on the **Sales by Product Family** JDBC connection. Change the SQL query as follows:

```
SELECT st.store_country, pc.product_family,pc.product_category,
CAST(CONCAT(t.the_year, ' - ', t.quarter) as CHAR) as period,
SUM(sf.unit_sales) as unit_sales,
SUM(sf.store_sales) as store_sales,
SUM(sf.store_cost) as store_cost
FROM sales_fact_1998 sf
INNER join product p on sf.product_id = p.product_id
INNER join product_class pc on p.product_class_id = pc.product_
class_id
INNER join time_by_day t on sf.time_id = t.time_id
INNER join customer c on sf.customer_id = c.customer_id
INNER join promotion pr on sf.promotion_id = pr.promotion_id
INNER join store st on sf.store_id = st.store_id
GROUP BY st.store_country, pc.product_family, pc.product_category,
t.the_year, t.quarter, st.store_country
ORDER BY st.store_country, t.the_year, t.quarter, pc.product_
family,pc.product_category
```

3. Click on **OK** to close the dialog and confirm the new query.

4. Under the **Details** band, select the **store\_country** and **period** fields and either press the *Del* key or right-click and choose **Delete** from the contextual menu. Then, from the **Details Header** band, select the **Store Country** and **Period** labels and either press the *Del* key or right-click and choose **Delete** from the contextual menu.
5. Click on the **Structure** tab, then from the report's hierarchical structure, right-click on the **Group** element and choose **Add Group**. The **Edit Group** dialog box opens as shown in the following screenshot:



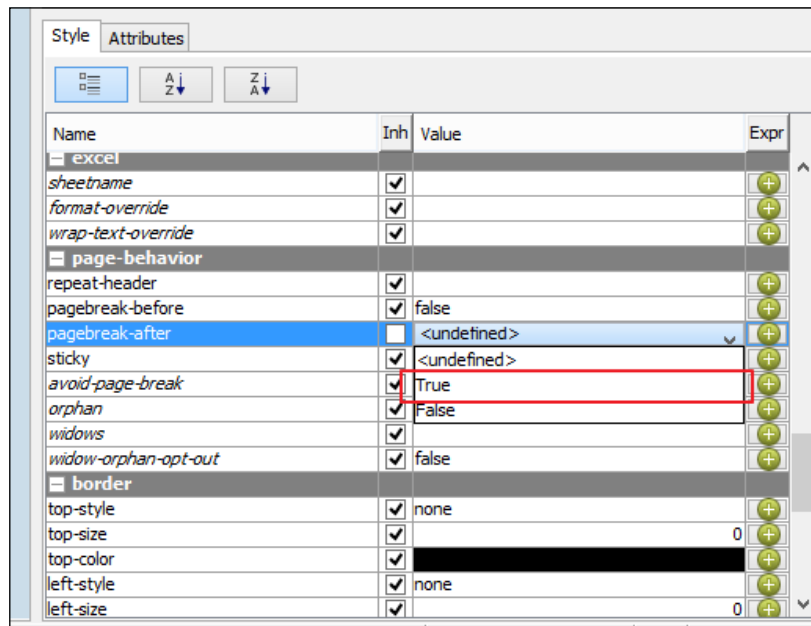
6. In the **Name** field, type `Group by Period`. Then, from the **Available Fields** listbox, select the **period** field and move it to the listbox at the right-hand side by clicking on the big arrow in the middle of the two listboxes. Click on **OK** to close the dialog and confirm the selection.
7. In the **Structure** tree view, we have a new group, named as specified, that is a child of the original group element. In the report canvas, two new bands appear.
8. Now from the **Structure** tab tree view, select the root level's group (this group is the immediate child of the **Master Report** element). Right-click on it and select **Edit Group...** The **Edit Group** dialog box opens.
9. In the **Name** field, type `Group by Store Country`. Then, from the **Available Fields** listbox, select the **store\_country** field and move it to the listbox at the right-hand side by clicking on the big arrow in the middle of the two listboxes. Click on **OK** to close the dialog and confirm the selection.
10. In the **Structure** tab's tree view, the group now has a name. From the **Structure** tab's tree view, select the **Group Header** element that is an immediate child of the group named **Group by Store Country**. Right-click on it and choose **Hide Element** from the menu. The band becomes visible in the report canvas.
11. Select the first **Group Header** band in the canvas. Click on the **Style** tab and locate the **layout** property from the **size & position** category. Choose **block** from the drop-down list. Do the same with the second **Group Header** band.
12. Take two label components from the component toolbar on the left-hand side and drag them to the first and second **Group Header** bands, respectively.



13. Select the two labels just inserted into the bands. Choose the **Paragraph** option from the **Format** menu and then select **Middle** from the **Vertical Alignment** fieldset. Click on **OK** to close the **Format Element** dialog box.
14. Select the label in the first **Group Header** band. Click on the **Attributes** tab and then locate the **Value** property. Click on the green icon button on the right-hand side of the property value field. The expression editor opens. Type the following formula in the **Formula** text area:  

```
"Sales for stores located in " & [store_country]
```
15. Click on **OK** and close the dialog.
16. Select the label in the second **Group Header** band. Click on the **Attributes** tab and then locate the **Value** property. Click on the green icon button in the right-hand side of the property value field. The expression editor opens. Type the following formula in the **Formula** text area:  

```
"Reference Period: " & [period]
```
17. Click on **OK** and close the dialog.
18. Select the first **Group Header** element from the tree view in the **Structure** tab. Under the **Style** tab, locate the **repeat-header** property in the **page-behaviour** category. Choose **True** from the drop-down list. Do the same with the second **Group Header** element.
19. Select the **Group Footer** element related to the group named **period**. Under the **Style** tab, locate the **pagebreak-after** property in the **page-behaviour** category. Choose **True** from the drop-down list as shown in the following screenshot:



20. Try to preview the report. As you can see, the report adds a header for each country and displays the sales data for that respective country by period.

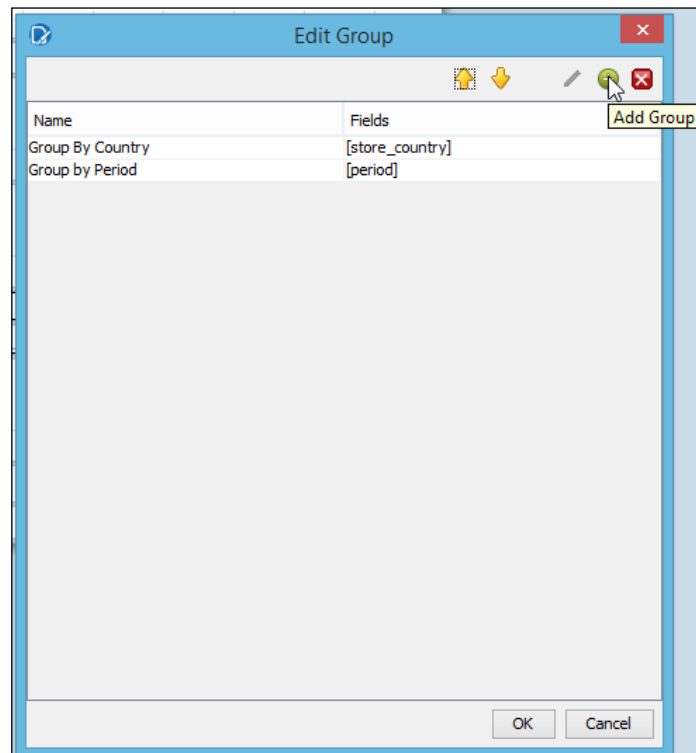
### How it works...

With groups, we can add a set of section headers and footers that display values as soon as a key breaks. The values in the key fields remain the same for a set of rows, then at a certain point in time, these values change. This change in the key values is called **break**.

In Pentaho Reporting, we can define as many groups as we want, nested one into another. There is always a default group in any report that is not configured, and for this reason, it doesn't do anything until we configure it. If we look at the hierarchical view of the report under the **Structure** tab, under the default group, we have the **Detail** section with the **Details** band and the **Details Header** and **Details Footer** bands.

To add a new group, we can select the default group element; right-click on it and choose **Add Group**.

An alternative way to access the same **Add Group** entry is from the **Groups** option under the **Edit** menu. The **Edit Group** dialog opens, and then we can click on the add group button in the top-right toolbar as shown in the following screenshot:



The group has a name and, again, remember to give a meaningful name to the group we are going to configure. Then, we have to list **Available Fields** and **Selected Fields**. To define a group key, we must select a set of fields from the **Available Fields** listbox and move them to the **Selected Fields** listbox by using the arrow in between. We can reorder the key fields by using the two up and down arrows located in the top-right corner of the **Selected Fields** listbox. We can also delete key fields from the **Selected Fields** listbox by choosing a field and clicking on the red icon button with a white **X** over it, located in the same place.

When we are fine with defining the group, we can click on the **OK** button to confirm it. Every time we define a new group, a new set of the **Group Header** and **Group Footer** bands will appear. If any one of these two bands are not used, we are free to hide them to not bother us.

### There's more...

So far, we've talked about hiding a bar. Nevertheless, we also know about the existence of a visibility flag (the **visible** property) in the report's object style properties. Let's analyze a bit of the difference between these two settings.

#### Hiding bands versus setting the visibility flag

Any band can be hidden in the canvas area. Any object, even bands, can be set to visible or invisible. Let's discuss this particular case just for bands: what is the difference?

Let's start with the hide property. Until Pentaho Report Designer 4.8, band hiding was set through the **hide-on-canvas** property. From Pentaho Report Designer 5, hiding a band is even easier. We can hide (or unhide) a band from the report canvas by going to the **Structure** tab, right-clicking on the band for which we want to manage the hiding state, and choosing **Hide Element**. If the element is hidden, the entry in the contextual menu has a tick on its left.

Visibility is a property of all the reports' objects. We can set it from the **Style** tab by setting the value of the **visible** property. We can either set this value by choosing it from the drop-down list or dynamically by using a formula.



Now, let's come to the difference. Hiding a band is only a matter of making the band invisible in the canvas at design time. Suppose that we set the **height** property for the band and then we hide the band, but we forget to clear the band's **height** property. Because the band has the **height** property explicitly set, we will see the band in the report output even if the band is hidden in the report canvas.

On the other hand, suppose that we set the **height** property for the band and we make the band invisible (setting the **visible** property to `false`) but we forget to clear the band's **height** property. In this case, even if band has the **height** property explicitly set, the band we will not be visible in the report's output. Therefore, setting the **visible** property to `false` removes the item from the report's output. Hiding a band is just a matter of making a band invisible at design time, but this operation does not influence the runtime report's behavior.

Regarding the visibility status of objects different from bands, there is another property that comes to our attention and is very important. If we have a field or label in a band and we set the value for the **visible** property to `false` (not visible), the element may not be visible but the space allocated will remain the same even if the object is not present. This is due to the value of the **invisible-consume-space** property in the **size & position** category in the parent object. In this case, we have two scenarios:

- ▶ If the parent band has the **invisible-consume-space** property set to `True` and the label or field has the **visible** property set to `False` (not visible), the object will continue to occupy the space needed even if it is not visible
- ▶ If the parent band has the **invisible-consume-space** property set to `False` and the label or field has the **visible** property set to `False` (not visible), the object will be not visible and will not consume any more space in the report

## Using functions to add calculated fields

Now that we have a report that groups data by country and gives sales by period for every country, it would be interesting to calculate the total sales by period. To do this, we can start exploring the use of functions in a report. This recipe will show you how to use functions to calculate summarization totals.

### Getting ready

For this recipe, you must have Pentaho Report Designer started.

## How to do it...

The following steps will show you how to use functions to calculate summarization totals:

1. From the **Pentaho Report Designer** menu, select the **Open** option under the **File** menu. The **Open file** dialog box opens. Go to the <cookbook\_samples>/ch07/resources directory and choose report\_designer\_sample9.prpt. Click on **OK** to close the dialog. Immediately after, save the report as report\_designer\_sample10.prpt so that we have a new report to start from for this recipe without doing everything from scratch.
2. Click on the **Data** tab and right-click on the **Functions** element from the tree view. Choose **Add Functions...** from the contextual menu.
3. The **Add Functions...** dialog box opens. Double-click on the **Summary** folder to expand it and select the **Sum** item element. Then, click on the **OK** button to close the **Add Functions** dialog box and confirm the selection.
4. A new element, an immediate child of the **Functions** element, appears in the tree view under the **Data** tab. Select the new element to show its properties in the property editor below the **Data** tab tree view.
5. Look for the **Function Name** property and type TotalsByPeriod as a meaningful value.
6. Look for the **Field Name** property. Choose the **store\_sales** field from the drop-down list and press *Enter* to confirm the selection.

**New Trend Measure**

Name:

Trended Measure: Store Sales

Period type:

Number of periods:

Show trend as:

Decimal Places:

**Note: To trend, use time attributes only (e.g. Year, Quarter, Month).  
To select a time attribute, you need to include one in the report.**

7. Look for the **Reset on Group Name** property. Select **Group by Period** from the drop-down list and press *Enter* to confirm.
8. Now that we have defined the summarization function to calculate the total by period, we are going to use this calculated field in our report. Click on the **Structure** tab. Expand **Group: store\_country, Group Body**, and then **Group: period**.
9. Right-click on the **Group Footer** element, the immediate descendent of **Group: period** and verify whether the **Hide Element** menu item is unchecked.
10. Select the **Group Footer** band from the report canvas. Click on the **Style** tab and look for the **layout** property under the **size & position** category. Choose **row** from the **values** drop-down list and press *Enter* to confirm.
11. Get the **label** component from the components toolbar on the left-hand side and drag it to the **Group Footer** band. Double-click on the label and type `Total Sales by Period:`. Press *Enter* to confirm. Size the label accordingly by pulling the handles as required.
12. With the label selected, choose the **Paragraph** option from the **Format** menu and then check the **Middle** radio button from the **Vertical Alignment** field set.
13. Get the **number-field** component from the components toolbar in the left-hand side and drag it to the **Group Footer** band. Click on the label to the left and press *Ctrl + C*. Then, select the **number-field** component we just dragged to the band and press *Ctrl + Shift + V* to apply the same vertical alignment.
14. Click on the **Attributes** tab and then look for the **field** property. Choose **TotalsByPeriod** from the list and then press *Enter* to confirm.
15. Finally, always from the **Attributes** tab, look for the **format** property. Type `$#, ##0.00` as the formatting mask for this value and press *Enter* to confirm.
16. Preview the report. We will see the sales total for any period.

### How it works...

Functions in Pentaho Reporting are a way to define pieces of business logic everywhere in our report. We like to say "pieces of business logic" instead of calculation capabilities because defining a function in Pentaho Reporting is not as insignificant as defining a sum, difference, or generic operation—it is something more powerful.

Firstly, any function we define is identified by its name. Therefore, having a correct naming convention is appropriate in this case too. To add a new function in our report, in the **Data** tab, right-click over the **Functions** element and select **Add Function...** from the contextual menu.

The **Add Function...** dialog box contains a set of function categories that we can go through to choose the base function that is most suitable for our needs. In our case, because we wanted to calculate the total sales by period, we are going to use the **Sum** function from the **Summary** category. Eventually, in case we are not able to find the right function, we can always use the **Open Formula** function and write our own expression.

As soon as we have selected the function type by either double-clicking on the function name or by selecting the function name and then clicking on the **OK** button in the **Add Function...** dialog box, the dialog closes and a new item is added as an immediate child of the **Functions** element. The function we just added has a default name and must be configured to behave properly. The set of configuration properties depends in number and type by the type of function we select.

Select the new function from the **Data** tab tree view and look below in the property window, then apply the following configuration changes:

- ▶ In the **Function Name** property, type a meaningful name for the function.
- ▶ Set the value of the **Field Name** property to the name of the field we want to use to create an aggregation. We can choose the field name from the drop-down list in the property's value.
- ▶ Because we want the total by period, another important property to be set is the **Reset on Group Name** property. When it is set, this property will reset the sum as soon as the group breaks. In our case, it must be set to the value of the group we defined to identify the periods; we called that group **Group by Period**.

After the function has been properly named and configured, it can be used by assigning it as the input value to a field in the report.

### There's more...

There is an important thing to be considered when using a function result in our report any time the result is something that is not a string. The function result is an untyped object due to the very generic nature of a function. Therefore, we need to follow some advice in order to display a date or number and obtain the right formatting.

### Displaying dates and numbers as the results of a function

Function results are always expressed as a string due to the very generic nature of the functions. So, how can we correctly display a number or a date as a result of a function? If we drag-and-drop the function into the report canvas independently from the band we are dragging the function into, Pentaho Reporting associates a String type field to that function. To correctly display a number or a date, we must first drag-and-drop an empty number or date field by getting it from the component toolbar on the left. Then, we select the field we just added in the report canvas, click on the **Attributes** tab, and look for the **field** property. From the drop-down list, we select the name of the function whose value we want to display and press the *Enter* key to confirm. Now, the function fills the field with its value, but because the field is typed, we can also specify a formatting mask. To do this, look for the **format** field and specify the mask required for our business case.

## Using subreports to embed content

Pentaho Reporting, as with the vast majority of reporting tools, feeds the report with data by means of just a single query. We can have many queries (some to fill the report, some to fill drop-down lists, and so on) defined in our report, but just one and only one of these queries will feed the report. However, sometimes, this could be a bit limiting. A typical example is when we must design a report with a header and a details section; typical situations such as these are printing invoices or orders. To illustrate how to use a subreport, this recipe will show you how to build a list of all the items bought by a customer. Of course, there would be other ways to do this and using a subreport is just one of these.

### Getting ready

For this recipe, you must have the Pentaho Report Designer started.

### How to do it...

The following steps will show you how to use a subreport in order to print the total sales by a customer, with the details of what is purchased by the customer:

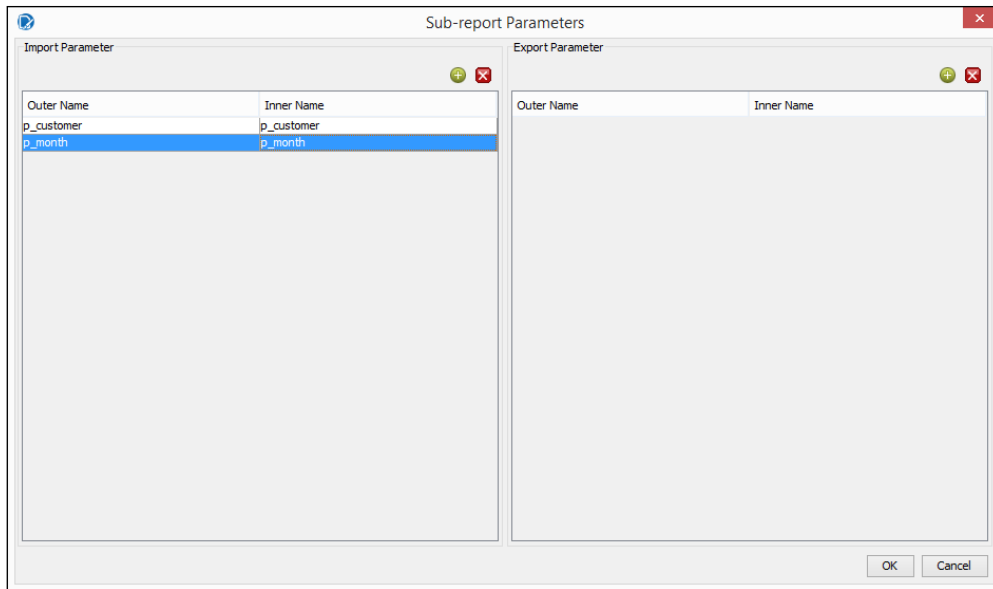
1. From the **Pentaho Report Designer** menu select the **Open** option under the **File** menu. The **Open file** dialog box opens. Go to the <cookbook\_samples>/ch07/resources directory and choose `report_designer_sample11-start.prpt`. Click on **OK** to close the dialog. Immediately after, save the report as `report_designer_sample11.prpt`. This report has a part already prebuilt to let us start with the exercise from the right point.
2. From the component toolbar in the left-hand side, select the **subreport** component from the bottom-left side. Drag the **subreport** component to the **Details** band.
3. Immediately after we drop it there, a dialog asks us if we want our subreport banded or inline. Choose **banded**. A new band appears in the **Details** band, stacked at the bottom in our **Detail** area. Double-click over the **subreport** band. A new tab opens in Pentaho Report Designer with a new report in it.
4. From here, we will work only in the new report created by Pentaho Report Designer. Click on the **Data** tab in the top-right corner of the Pentaho Report Designer user interface. Select the JDBC connection immediately below the **Data Set** element and then right-click over it and select **Edit datasource...**
5. Let's add a new query to populate our subreport. Add a new query and call it `Sales By Customer`. Then, copy the following query in the **Query** text area:
 

```
SELECT sf.customer_id, st.store_country, pc.product_family,pc.
product_category, p.product_name,
SUM(sf.unit_sales) as unit_sales,
```



```
SUM(sf.store_sales) as store_sales, SUM(sf.store_cost) as store_
cost
FROM sales_fact_1998 sf
INNER join product p on sf.product_id = p.product_id
INNER join product_class pc on p.product_class_id = pc.product_
class_id
INNER join time_by_day t on sf.time_id = t.time_id
INNER join customer c on sf.customer_id = c.customer_id
INNER join promotion pr on sf.promotion_id = pr.promotion_id
INNER join store st on sf.store_id = st.store_id
WHERE t.month_of_year=${p_month}
AND sf.customer_id=${p_customer}
GROUP BY sf.customer_id, st.store_country, pc.product_family,
pc.product_category, p.product_name,
st.store_country
ORDER BY sf.customer_id, pc.product_family, pc.product_category
```

6. Click on **OK** to confirm the changes and close the **JDBC Data Source** dialog.
7. The new query is displayed as an immediate child of the **foodmart\_mondrian** data source. Select the **Sales by Customer** query and right-click on it. Choose **Select Query** from the contextual menu.
8. Unhide the **Details Header** band, then apply a **row** layout to the **Details** and **Detail Header** bands.
9. Look at the fields that appear below the **Sales by Customer** query. Drag the following fields in the order specified to the **Details** band: **product\_family**, **product\_category**, **product\_name**, **unit\_sales**, and **store\_sales**.
10. Style the fields so that they are vertically aligned in the middle.
11. Style **product\_family**, **product\_category**, and **product\_name** so that they are left-aligned with 3 px of left padding.
12. Style **unit\_sales** and **store\_sales** so that they are right-aligned with 3 px of right padding.
13. To the **store\_sales** field, apply the following numeric format: \$#, ##0.00.
14. To the **unit\_sales** field, apply the following numeric format: #, ##0.##.
15. Drag a set of labels in the **Details Header** area and lay them out according to the following rules: the same width as the referred fields, the text centered horizontally, vertically aligned in the middle, and with bold characters. Give the labels the following names: **Product Family**, **Product Category**, **Product Name**, **Unit Sales**, and **Store Sales**.
16. From the menu, choose **Edit Subreport Parameters...** under the **Data** menu to open the **Sub-report Parameters** dialog box (see the following screenshot for details). Look at the **Import Parameter** list at the left-hand side of the screen and click on the add parameter icon button located at the top-right corner (see the following screenshot for details). From the **Outer Name** drop-down list, choose **p\_country**, then in the **Inner Name** field, type **p\_country**.



17. Click on the add parameter button to add a second parameter. This time, from the **Outer Name** drop-down list, choose **p\_month**, then in the **Inner Name** drop-down list, type **p\_month**. Click on **OK** to close the **Add Parameter** dialog box and confirm the parameters' definition.
18. We just finalized the report; try to preview it in order to see the header details style made using a subreport to get data for the details area.

### How it works...

Subreports are reports embedded in a so-called **master report**. As soon as the master report executes, it will call the subreport, execute it, and then will render the subreport's result in the report body. We can define as many subreports as we want, and we can nest them at any level we want; there is no limit on this. The only limit is set by the number of connections that, while executing, the report will open for our database.

To add a new subreport, we can get it from the component toolbar in the left-hand side of the report canvas. As soon as we drop the subreport, Pentaho Report Designer will ask whether we want it in a banded or inline style.

The first option, **Banded**, adds the subreport as a new band in the master report. This is useful to develop master detail prints, such as orders or invoice prints. The second option, **Inline**, will materialize the report as a sort of field that we can position everywhere in the master report canvas. This is useful when we want to use a subreport to make some complex calculations and return a value to be displayed in the master report, or when we want to design charts that can be inserted anywhere in the canvas (we will see an example of this in the next recipe).

As soon as the subreport has been added to the master report, we must double-click over the subreport area to open it and start working on it. The subreport will be shown as a new report in a new tab and can be developed in the same way as a normal report.

The only thing we must take is that the master report will link to the subreport through a set of parameters that will send it a sort of reference context key upon getting the right subreport values. For example, in our recipe, the subreport will extract and display the products purchased by specific customer for a specific month (period). Therefore, our context keys on which the subreport will extract its data will be **customer\_id** and **month\_id**.

To specify the subreport parameters, we can choose the **Edit Subreport Parameters...** option from the **Data** menu and look at the **Import Parameter** list at the left-hand side. We can add all the required parameters by clicking the add parameter icon button located at the top-right side of the listbox. As soon as we have clicked on it and added a new empty line to the list, it is important to note the following things:

- ▶ On the left-hand side, we have the name of all the fields, parameters, and functions that we can choose from the master report.
- ▶ On the right-hand side, we will define the name of the subreport parameter that will contain the value coming from the master report. A good idea would be to name the parameter as the name of the field, parameter, or function coming from the master report.

Remember that the subreport always needs to be activated by a query in the master report. This may seem quite obvious sometimes, but is not as obvious as it appears to be. In fact, suppose we are going to design a report dashboard with four charts in it. In this case, the master report will be only a way to lay out the dashboard but will not have any query filling in it. To properly start the report, it is a good rule of thumb to use a fake query, for example, `select 1` or `select 1 from dual` or whatever—it may be fine just to start the subreports contained in the master report.

## Embedding microcharts in reports with sparklines

Sometimes, we would like to easily summarize the values of a specific measure along a time period. For example, we would like to easily give an idea of the last month's or last period's sales. An interesting approach would be to use specific microcharts called sparklines to graphically show the last period's values of a measure. This recipe will show how we can insert sparklines in report details' rows.

### Getting ready

For this recipe, you must have Pentaho Report Designer started.

## How to do it...

The following steps will show you how to use a subreport to print the total sales by customer, with details of what is purchased by the customer:

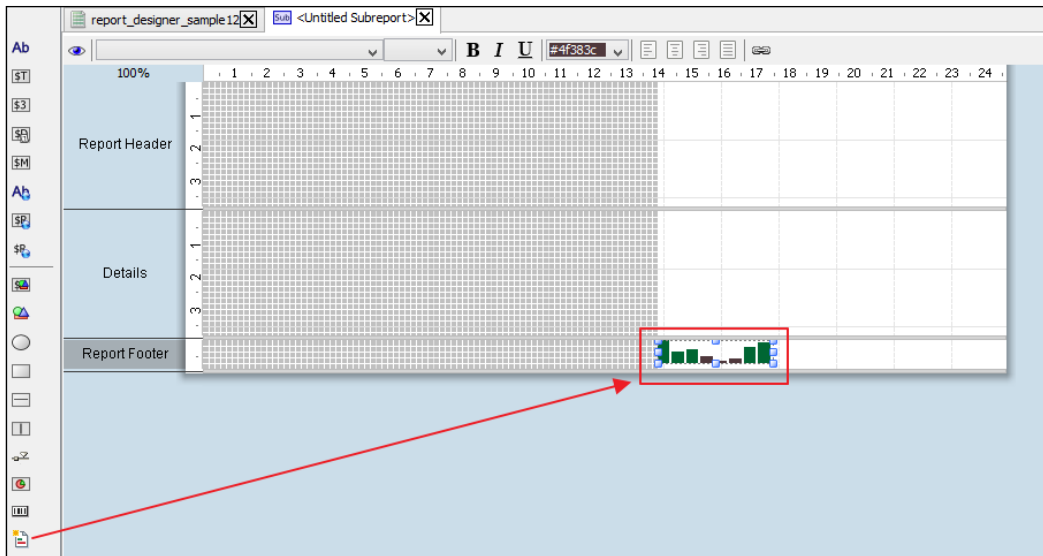
1. From the **Pentaho Report Designer** menu, select the **Open** option under the **File** menu. The **Open file** dialog box opens. Go to the <cookbook\_samples>/ch07/resources directory and choose `report_designer_sample12-start.prpt`. Click on **OK** to close the dialog. Immediately after, save the report as `report_designer_sample12.prpt`. This report has a part already prebuilt to let us start with the exercise from the right point.
2. From the component toolbar on the left-hand side, choose the **subreport** component from the bottom-left side. Drag the **subreport** component to the **Details** band.
3. Immediately after we drop it there, a dialog asks us if we want our subreport banded or inline. Choose inline. This time the subreport seems like a new field that is added at the extreme right-hand side of the existing report column set.
4. Drag a label from the component toolbar in the left-hand side to the **Details Header** band. Size the label according to the referred inline band. Double-click over the label and type `Historical Sales`.
5. Select the **Sales Total** header label and press `Ctrl + C` to copy the format, then paste the formatting to the new **Historical Sales** header label by pressing `Shift + Ctrl + V`.
6. Double-click over the inline subreport. A new tab opens in Pentaho Report Designer with a new report in it.
7. From this point on, we will work only in the new report created by Pentaho Report Designer. Click on the **Data** tab in the top-right corner of the Pentaho Report Designer user interface. Select the JDBC connection immediately below the **Data Set** element, then right-click over it, and select **Edit datasource....**
8. Let's add a new query to populate our subreport. Add a new query and call it `Sales By Period (Parametric)`. Then, copy the following query in the **Query** text area:

```
SELECT
CAST(CONCAT(t.the_year, ' - ', t.quarter) as CHAR) as period,
SUM(sf.store_sales)/1000 as store_sales
FROM sales_fact_1998 sf
INNER join product p on sf.product_id = p.product_id
INNER join product_class pc on p.product_class_id = pc.product_class_id
INNER join time_by_day t on sf.time_id = t.time_id
INNER join customer c on sf.customer_id = c.customer_id
INNER join promotion pr on sf.promotion_id = pr.promotion_id
INNER join store st on sf.store_id = st.store_id
```

```
WHERE st.store_country = ${store_country}
AND pc.product_category = ${product_category}
GROUP BY
t.the_year, t.quarter, st.store_country
```

9. Click on **OK** to confirm the changes and close the **JDBC Data Source** dialog.
10. The new query is displayed as an immediate child of the **foodmart\_mondrian** data source. Select the **Sales by Customer** query and right-click on it. Choose **Select Query** from the contextual menu.
11. From the **Data** menu, choose **Edit Subreport Parameters...** Look at the **Import Parameter** list on the right-hand side and click on the add parameter icon button located at the top-right side. From the **Outer Name** drop-down list, choose **store\_country**, then in the **Inner Name** field, type `store_country`.
12. Click on the add parameter icon button again to add a second parameter. This time, from the **Outer Name** drop-down list, choose **product\_category**, then in the **Inner Name** drop-down list, type `product_category`. Click on **OK** to close the **Add Parameter** dialog box and confirm the parameters' definition.
13. Under the **Data** tab, right-click over the **Functions** element. The **Add Function...** dialog box opens; expand the **Common** category and double-click on **Open Formula**. The **Add Functions...** dialog box closes and a new element is added as an immediate child of the **Functions** element.
14. Select the new function element just inserted. For the **Function Name** property, type `SparklineData`. Look at the **Formula** property and click on the little button on the right-hand side of the property's value field. The **Formula Editor** dialog box opens. Type the following formula in it and click on **OK** to confirm the data and close the dialog box:  

```
=NORMALIZEARRAY (MULTIVALUEQUERY("Sales By Period  
(Parametric)";"store_sales";3600;10))
```
15. From the components bar on the left-hand side, drag-and-drop the **bar-sparkline** component to the **Report Footer** band.



16. Select the **bar-sparkline** component we just added and click on the **Attributes** tab. Look for the **field** property and choose **SparklineData** from the drop-down list.
17. Click on the **Style** tab. Look for the **high-color** property and type #006633 as the property's value. Then, look for the **text-color** property and type #4f383c as the property's value. Press *Enter* to confirm the values.
18. Save the report and preview.

### How it works...

Sparklines are little graphs that can be added to our report detail's lines in order to graphically show the behavior of a measure over a specific period of time. In Pentaho Report Designer, we have three different types of sparklines we can add to reports: lines, bars, and pie charts. Any of these charts will take an array of values as input and automatically configure them to display values properly.

To properly do this, we must put our sparkline in a subreport. Add a subreport to the **Master Report Details** band as an inline subreport. Now that we have our subreport, we must configure the context key coming from the master report and the query that will get the right values based on the context key coming from the master report. We already know all the details about this because we learned this in the previous recipe.

Two things are important to note here, and we will discuss these in the following points:

- ▶ The first important thing to note is choose in which report's band we can drop the sparkline. In our case, we need to display a single sparkline graph for each set of data. Because the sparkline will accept an array of values as input to display the microcharts, we are sure that we always have just one line of data represented through an array of  $n$  values. Therefore, because we always have a single line of data, we can put the sparkline wherever we want in the report. Conventionally, we put it in the report's footer. We will see in the next recipe that the positioning of the chart in the correct band is important to have a correct visualization.
- ▶ To transform the data that comes out from our query in an array format, as desired by the sparkline, we can use two functions nested one into the other. The first function called `MULTIVALUEQUERY` executes a query by calling it by name. Then, it returns the values that come out from a field specified by name. There exists another function named `SINGLEVALUEQUERY` that does the same on a query that returns just a single value. The second function we need is named `NORMALIZEDARRAY` and is the function that does the trick for us. It takes a set of values by row and transforms them into an array, exactly the way we want. So, the formula to get our array ready to be given to the sparkline as input is as follows:

```
=NORMALIZEARRAY (MULTIVALUEQUERY ("Sales By Period  
(Parametric)"; "store_sales"; 3600; 10))
```

This recipe has shown another example of how expressions and functions are powerful in Pentaho Reporting. However, please note that expressions and functions can incur performance penalties. They must be used diligently and in appropriate ways.

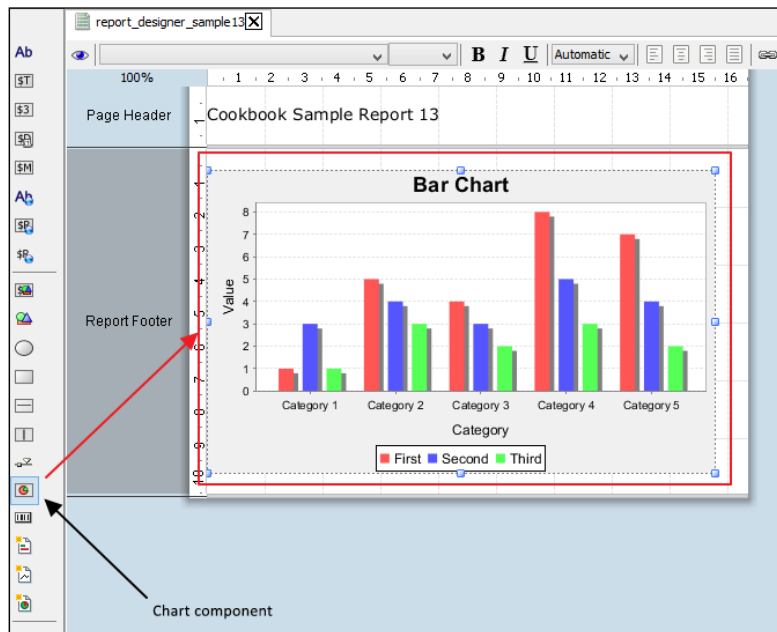
## Adding charts to our report

Charts are a beautiful and immediate way to help users quickly understand business indicators. Designing good visualizations is not a trivial matter but is an important task that requires a bit of knowledge about visual communication. We often see people building colorful charts with 3D functionalities, and why not use psychedelic lights because they say they look good. This is a false assumption. The more the chart is clean and minimal, the more it helps the user to be well understood. This recipe will show the basics of how to add charts in Pentaho Report without any intention of explaining what is good and what is not good about how to structure a report visualization and what to use in order to be clearly understood by the user.

## How to do it...

The following steps will show you how to add a simple chart to our report:

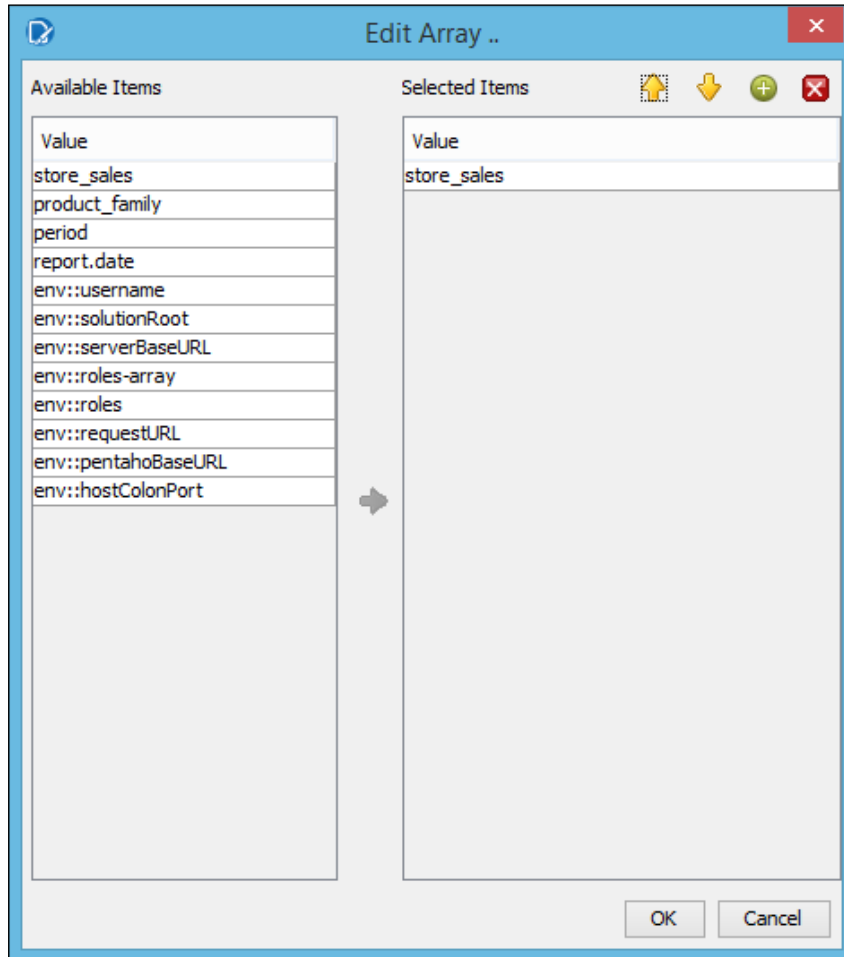
1. From the **Pentaho Report Designer** menu, select the **Open** option under the **File** menu. The **Open file** dialog box opens. Go to <cookbook\_samples>/ch07/resources and choose `report_designer_sample13-start.prpt`. Click on **OK** to close the dialog. Immediately after, save the report as `report_designer_sample13.prpt`. This report has a part already prebuilt to let us start with the exercise from the right point.
2. From the component toolbar on the left-hand side, choose the **chart** component from the bottom-left side. Drag the **chart** component to the **Report Footer** band, as shown in the following screenshot:



3. Size the chart area appropriately by using the side handles. Double-click on the chart area to open the **Edit Chart** dialog box. We are going to insert a bar chart that is the default chart type, as soon as we create a new chart.
4. Look at the right part of the dialog under the **Primary Datasource** tab. Locate the **category-column** property under the **Common** category of the property editor. Choose **period** from the list.
5. Look for the value property under the same category. Click on the little button at the right-hand side of the property's value field; the **Edit Array...** dialog box opens as shown in the following screenshot.



6. Select the **store\_sales** field from the **Available Items** listbox and click on the big green arrow to move it to the **Selected Items** listbox (see the following screenshot for details). Click on **OK** to confirm the selection and close the **Edit Array...** dialog box.



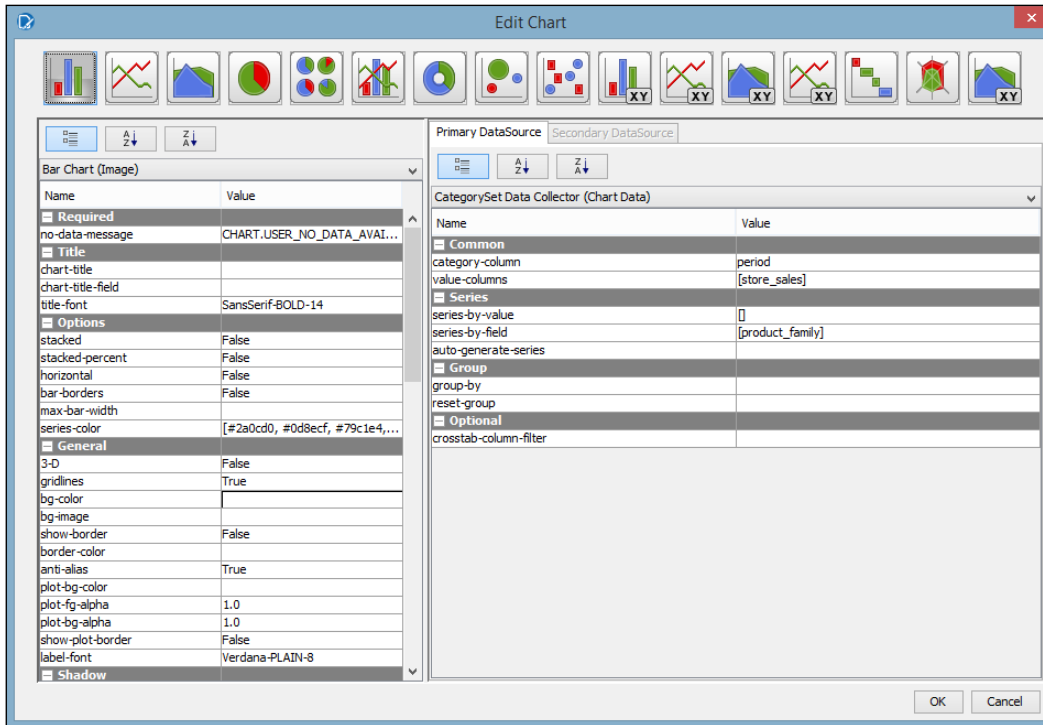
7. Look for the **series-by-field** property under the **Series** category of the property editor. Click on the little button in the right-hand side of the property's value field; the **Edit Array...** dialog box opens.
8. Select the **product\_family** field from the **Available Items** listbox and click the big green arrow to move it to the **Selected Items** listbox. Click on **OK** to confirm the selection and close the **Edit Array...** dialog box.
9. We configured the minimum number of properties required to show our chart. Save the report and try to preview it.

10. Let's go through changing other properties in order to style our report better. Look at the left-hand side of the **Edit Chart** dialog box. Under the **General** category, look for the **label-font** property. Click on the little button on the right-hand side of the property's value field; the **Edit Property...** dialog box opens, giving you the option to set the font. From the **Font Family** list, choose **Verdana**. From the **Font Style** field, choose **Plain**, and finally from the **Font Size** list, choose **8**. Click on **OK** to close the dialog and confirm the values.
11. Do the same with the **x-tick-font** property located under the **X-Axis** category and with **y-tick-font** located under the **Y-Axis** category.
12. Now let's format the values displayed as the label of y axis. Look for the **y-tick-fmt-str** property under the **Y-Axis** category. Type the following as the property's value:  
\$#.##0,00.
13. Try to preview the report again. As we can see, the font changes and the value labels on the y axis are aligned to the standard formatting mask used throughout the recipes' samples.
14. A last experiment could be changing the chart's bar colors. To do this, change the order of the values in the array for the **series-color** property under the **Options** category. Feel free to experiment with this and find a set of values that suit your purposes.

### How it works...

Let's go through some of the basics to configure a chart. Any chart is filled by using the so-called default query for our report. A problem immediately surfaces regarding where we can place the chart so that it can be displayed appropriately. Because the **Details** band is printed once for every row that comes out from the default query, we cannot put our chart in this band because. A good place to put our chart is either in the **Report Footer** or **Page Footer** bands because each one of these is independently printed once by the number of rows in the output from the query.

Once the chart is dropped onto the target band, we can size it by pulling the handles. Then, double-click on the chart to open the **Edit Chart...** dialog as shown in the following screenshot:



The toolbar at the top lets the user choose the chart type. We have a lot of possible charts, for example, bars, lines, area charts, pie charts, and so on.

Bar charts are the default type. We decided to go with this type in our recipe. The following two points briefly detail the sections under the toolbar:

- ▶ On the right-hand side, we have the properties for data source configuration. This is the place where we link the data with our chart. We can specify associations between series, values, and categories (if they are needed) and the relative columns that come out from our query and that contain the values for that elements. For certain chart types (bar chart is one of them), we can configure series by manually specifying a set of fixed values in the **series-by-value** property.
- ▶ On the left-hand side, we configure the styling of our chart. We have different sets of properties to configure the x and y axis labels, character formatting, legend, and so on. The set of available properties is different depending on the chart type.

The properties in the two listboxes depend on the selected chart type.

# 8

## Creating Dashboards

In this chapter, we will cover the following recipes:

- ▶ Creating a simple dashboard from scratch
- ▶ Adding prompts to get user input
- ▶ Creating a multiple-content dashboard
- ▶ Linking different content and enabling interaction
- ▶ Creating dashboards using CDE

### Introduction

One of the best and most attractive features of a BI system is having the possibility to build visualizations easily by putting together different types of content in a single view in one place. Usually, these kinds of visualizations are called dashboards. A dashboard is a collection of analytics content (charts, analysis reports, plain reports, or other kinds of linkable content) that shows different views of a business aspect from different standpoints in different ways, but in a single place. Dashboards add interactivity to their content to help us analyze our business data by drilling into various dimensions or by setting appropriate parameters.

Managers like dashboards because they immediately give an overview of the main **Key Performance Indicators (KPI)** in a clear and easy way. They are usually interested in high-level, summarized information about company performance on key business aspects. So, dashboards that target these organizational roles must be synthetic and quick to read. On the other side, people with operational roles find it useful to have a detailed dashboard in order to have all of their process-related KPIs together in a single place. Therefore, a single dashboard cannot satisfy the needs of every role inside a company, but different organizational roles require different dashboards because they are interested in different business aspects.

In this chapter, we will look at a set of recipes on using **Pentaho Dashboard Editor** and **Community Dashboard Editor (CDE)**. Pentaho Dashboard Editor is a web tool distributed with Version EE of Pentaho that lets you easily design a dashboard without requiring particular technical skills.

The CDE tool is a part of the **CTools** framework built by *Pedro Alves* and *Webdetails* with the help of the community (I had the honor of contributing to some of the CTools projects). It is a dashboard designer that is available for both the CE and EE versions that helps in building good looking dashboards in the easiest possible way. Even if it tries to minimize the amount of code needed to build a dashboard, CDE is a technical tool, so it is not for the everyday functional analyst. It requires a developer who is skilled with JavaScript, CSS, XML, HTML, JQuery, and related technologies. On the other hand, we must say that by using CDE, we are able to obtain dashboards that are graphically far ahead of what we can build with Pentaho Dashboard Editor.

The recipes in this chapter assume that we are able to successfully log in to the Pentaho User Console. To do this, we are free to use any user, not necessarily a demo user; but, it would be better to access and experience with the complete set of functionalities offered by the user interface. We will use a user that is a part of the administrator role. The administrator role is the role associated with all the super users of Pentaho (we will cover these aspects in *Chapter 2, Configuring Your BA Server Instance*). Therefore, it has full access to any functionality.

In case we want to use demo users, remember that we can use the following login credentials to access our system:

- ▶ `admin/password`: This is the new demo administrator of Pentaho. The famous user `joe` (the Pentaho-recognized administrator until Pentaho 4.8) has been dismissed in this new version.
- ▶ `suzy/password`: This is another simple user that we can use to access the system. Because `suzy` is not a member of the administrator role, it is useful to see what changes in case a user that is not an administrator tries to use the system.

Another assumption we are making in this chapter is that the Metadata Model we build during the illustration of recipes in *Chapter 4, Defining Business Models with the Pentaho Metadata Editor*, must be published and available as a Pentaho Metadata Data Source.

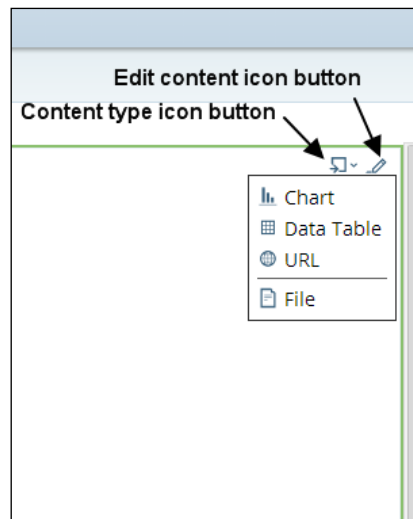
## Creating a simple dashboard from scratch

As the first example, this recipe will illustrate how we can create a dashboard from scratch using Pentaho Dashboard Designer. Remember that this recipe makes use of Pentaho Dashboard Designer, a tool available only in the EE version of Pentaho.

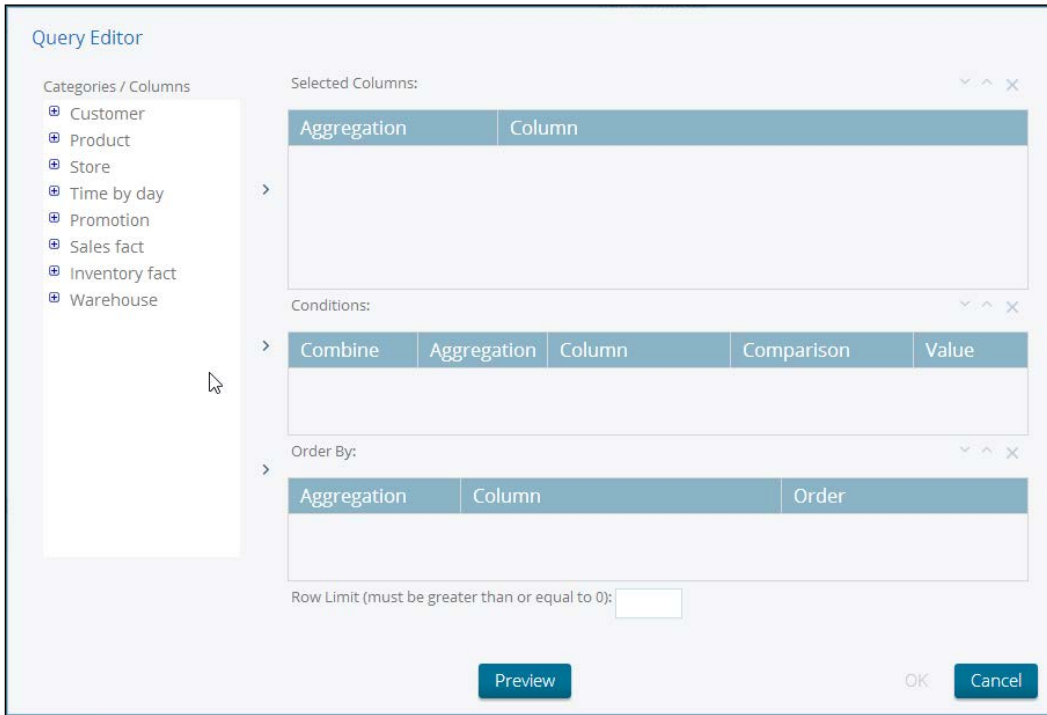
### How to do it...

The following steps detail how to create a simple dashboard from scratch using Pentaho Dashboard Designer:

1. To create a new dashboard with Pentaho Dashboard Designer, we can either press the **Create New** button in **Home** and then press **Dashboard**, or we can navigate to **File | New | Dashboard** from the PUC menu.
2. The dashboard editor canvas will open. Choose the layout template of our dashboard by selecting the **Template** tab from the tabbed view and then clicking on **Single**. This template lets us insert one piece of content in our dashboard.
3. From the **Objects** explorer located on the left side of the band, select the **Untitled 1** item. This element relates to the area in the canvas that will contain the chart we are going to create. Look at the upper-right corner of the **Untitled 1** chart's area and click on the first drop-down list button. Select **Chart** from the drop-down list as shown in the following screenshot:



- The **Select Data Source** dialog box opens. Select the **Cookbook Sample** metadata data source. The metadata **Query Editor** dialog opens as shown in the following screenshot:



- The idea is to represent the top 10 store cities in the USA by amount of **Store Sales** using a bar chart. From the **Store** category, select **Store country** and click on the first button in the middle of the two sets of lists to move this field to the **Selected Columns** list. Repeat the same operation with the **Store city** category and then perform the same steps with the **Store Sales** field from the **Sales** category. Please see the next screenshot for details.

- From the **Store** category, select **Store country** and click on the second button in the middle of the two sets of lists to move this field to the **Conditions** list. Type **USA** in the **Value** field. Please see the following screenshot for details:

Query Editor

Categories / Columns

- Customer
- Product
- Store
- Time by day
- Promotion
- Sales fact
- Inventory fact
- Warehouse

Selected Columns:

Aggregation	Column
NONE	Store country
NONE	Store city
SUM	Store sales

Conditions:

Combine	Aggregation	Column	Comparison	Value
	NONE	Store country	exactly matches	USA

Order By:

Aggregation	Column	Order
SUM	Store sales	DESC

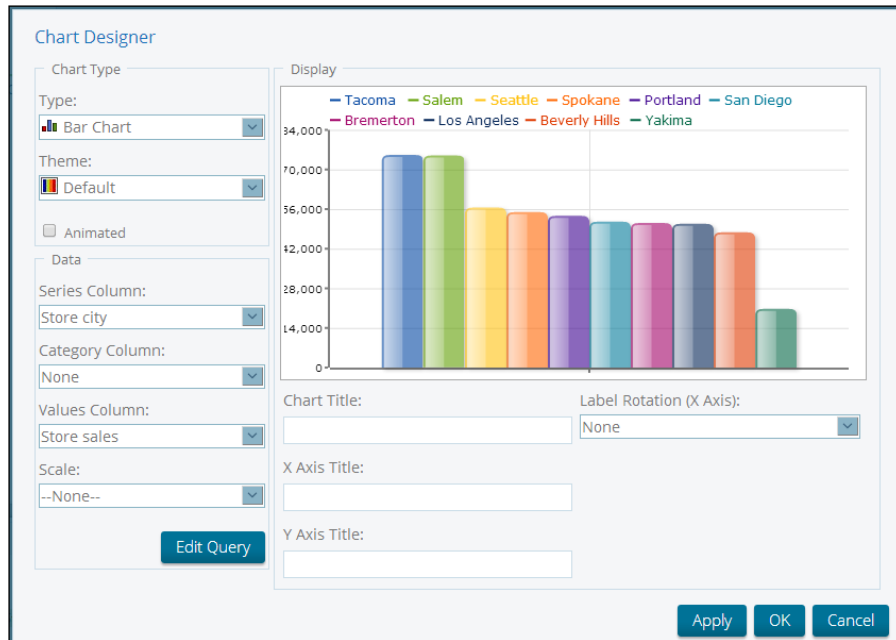
Row Limit (must be greater than or equal to 0):

Preview OK Cancel

- From the **Sales** category, select **Store Sales** and click on the third button in the middle of the two sets of lists to move this field in the **Order By** list. Choose **DESC** from the **Order** drop-down list. Finally, type **10** in the **Row limit** field (this is because, remember, we want to see the top 10 and not all the countries). Please see the previous screenshot for details.
- Click on the **Preview** button to check if everything works. If it works, close the **Preview** dialog, click on **OK** to confirm the query, and close the **Query Editor** dialog.



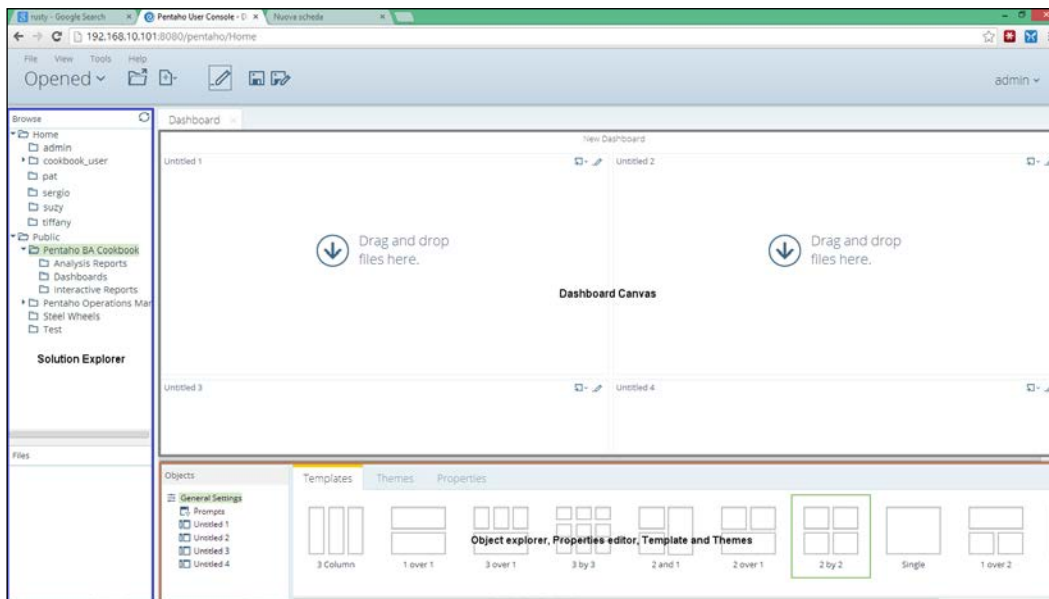
- The **Chart Designer** dialog opens. Select **Stores city** from the **Series** column's drop-down list. Select **None** from the **Categories** column's drop-down list. Finally, select **Store Sales** from the **Values** drop-down list. Please see the following screenshot for details:



- After we set all of these values, the **Display** area of the **Chart Designer** dialog will display a preview of our chart. Click on **OK** to confirm the changes and close the **Chart Designer** dialog.
- Let's finalize our dashboard. Type **Top 10 Sales by Store's City** in the **Title** field of the chart properties. Select **General settings** from the **Objects** explorer, and then choose **Properties** from the tabbed set on the right. Type **Dashboard 1** in the **Page Title** field.
- Temporarily, go to the **Browse Files** perspective. Go to the `/Public/Pentaho BA Cookbook` location in the solution and create a new folder named `Dashboards`.
- Go back to **Dashboard Editor** in the **Opened** perspective. Choose the **Save As** button from the PUC toolbar. Go to the `/public/Pentaho BA Cookbook/Dashboards` location and save the dashboard as `Dashboard 1`.

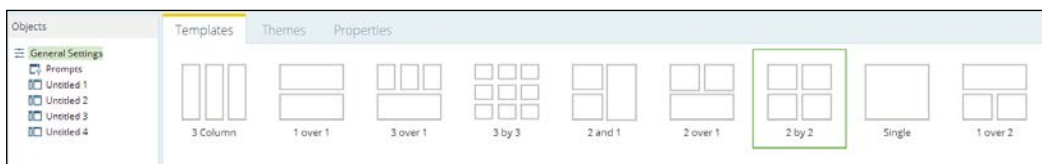
## How it works...

The Dashboard Editor interface is composed of three main zones, and each zone has a specific role while designing a dashboard. This is shown in the following screenshot:



The screenshot shows the layout of Pentaho Dashboard Designer canvas

- ▶ The first zone comprises of **Solution Explorer** on the left that gives the ability to navigate the content in the solutions folder. On the upper side of Solution Explorer, we can navigate to the folders in the solution. As soon as we select a folder, we get the list of files contained in that folder.
- ▶ The second zone contains the **Objects** explorer, **Properties** editor, and **Templates** and **Themes** selections. The **Objects** explorer is a hierarchical view over the dashboard object and is located on the very left of this area at the bottom (see the following screenshot for details). Each time we select an object item from the **Objects** explorer, to the right side of it, a property editor appears. **Properties** are collected in tabs that vary in name and number depending on the type of object selected. As soon as we select the root of the object's hierarchy, identified by the label **General Settings**, we can choose the **Template** to use in order to build the dashboard. **Templates** are layouts that fix the objects' position following a predefined figure (see the following screenshot for details). Every template is clearly identified by an icon and a name that immediately gives an idea of the layout we are going to give to our dashboard's objects. As soon as we start designing a dashboard, the first thing we must do is to decide which layout to use.



- ▶ The third zone that covers all of the remaining space is the **Dashboard Canvas** area, the place where our dashboard will come to life.

As soon as we select a dashboard's template, we can add content to the dashboard content zones in two ways:

- ▶ We can add content by choosing it from what is already available from the Pentaho Solution. Every kind of report can be added easily to a dashboard zone by dragging and dropping it from the Pentaho Solution to the content's target position in the dashboard. Security constraints applied on Pentaho BI objects do not permit the addition of objects we are not authorized to access.
- ▶ We can add charts, tables, or external URLs to our dashboard. Charts and tables are filled by the available data sources configured in our Pentaho BA Server. In this case, to add such content, we need to click on the Content Type icon button that is located in the upper-right corner of the target dashboard zone and choose the type of content we want to add from the list of available content types.

Every zone identified by the template in use is identified in the hierarchical structure represented in the **Objects** explorer. Every zone has a set of associated properties that depend on the content we will put there. In any case, every zone has a **Title** field named **Untitled n** by default, where **n** is the ordinal of the zone we are going to configure.

About the dashboard as a whole, by selecting the **General Settings** object from the **Objects** explorer and the Tab properties, we can fill the following fields:

- ▶ **Title:** This contains the dashboard title displayed in the tab, which identifies the dashboard as a whole in the Pentaho User Console. By default, the value is **New Dashboard**.
- ▶ **Refresh Interval (sec):** This could be filled with the value of the dashboard's refresh interval in seconds. This is useful if we want to refresh the dashboard periodically with a specified time delay.

### There's more...

By default, an analysis report always opens in edit mode; this could be problematic for normal users viewing the report. A workaround to this problem is explained in the next paragraph.

### Opening an analysis report in view mode

Analysis reports always open in edit mode by giving the users the opportunity to alter the report layout. To work around this, we must use dashboards. Define a new dashboard and select the template **Single** so that our report will take all the available space in the dashboard. Then, look for our report in Pentaho Solution and drag-and-drop it to the dashboard layout. Name the dashboard accordingly and then save it. This way, our analysis report will preserve all of the major functionalities, but users will not be able to modify its structure.

## See also

A good source of information on how to design data visualizations (from a conceptual point of view) is represented by *Stephen Few* in his books. We strongly suggest that you go through his website at <http://www.perceptualedge.com>.

Here are a few titles we suggest that you go through and read for general information about data visualizations:

- ▶ *Show me the Numbers: Designing Tables and Graphs to Enlighten*
- ▶ *Now You See It: Simple Visualization Techniques for Quantitative Analysis*
- ▶ *Information Dashboard Design: The Effective Visual Communication of Data*

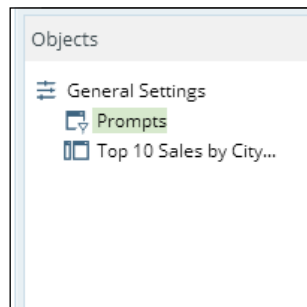
## Adding prompts to get user input

After showing how we can create a dashboard from scratch using Pentaho Dashboard Designer, we'll now look at how to choose an input parameter to dynamically set the country that we want to filter our dataset to get the top 10 ranking. This recipe shows us how to add one or more input parameter to our dashboard. Remember that this recipe makes use of Pentaho Dashboard Designer, a tool available only in the EE version of Pentaho.

## How to do it...

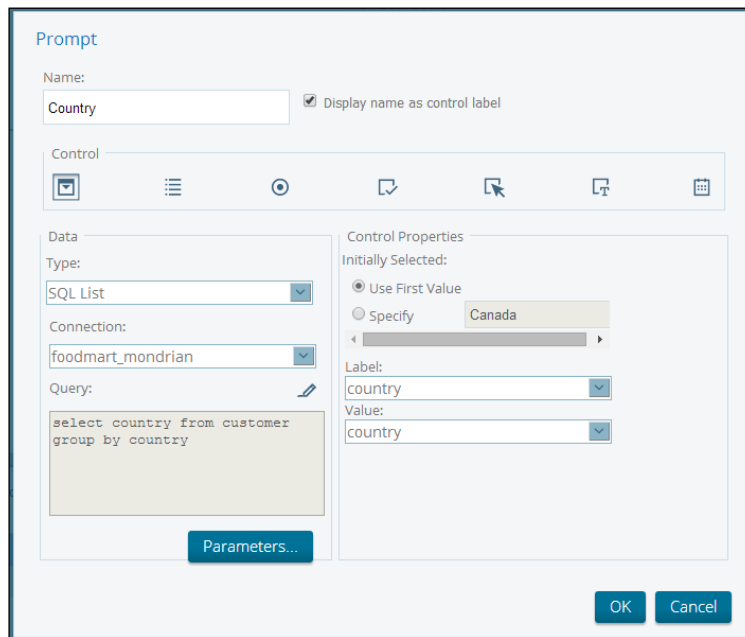
The following steps detail how simply we can create a parameter to get user input from the dashboard:

1. From the **Browse Files** perspective, go to the `/public/Pentaho BA Cookbook/Dashboards` location, select the report **Dashboard 1** from the previous recipe, and choose **Open** from the **File Actions** menu on the right.
2. From the **Objects** option located on the left side of the band, select the **Prompts** item (see the following screenshot for details). The **Prompts** option's detail mask appears on the right-hand side of the **Objects** explorer. Press the **Add new prompt** button. The **Prompts** dialog box appears as shown in the following screenshot:

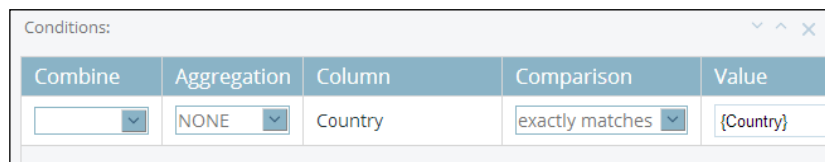


- From the **Type** drop-down list in the **Data** fields' group, select **SQL List**. Type **Country** in the **Name** field of the label **Prompt**. Choose **foodmart\_mondrian** from the **Connection** drop-down list (see the following screenshot for details).
- Type the following query:
 

```
SELECT country FROM customer GROUP BY country
```
- Click on **Test** to verify that everything works. The query extracts all distinct country items. Click on **OK** to confirm the query and then click on **OK** to close the **Prompt** dialog box, as shown in the following screenshot:



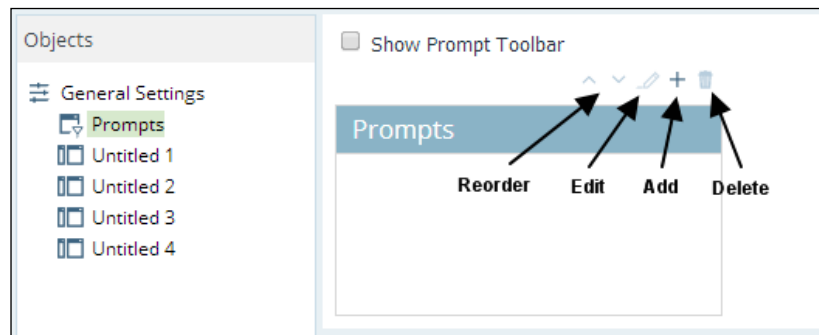
- Now that we have the prompt, we must change the metadata query to manage the parameter. From the **Objects** explorer, choose **Top 10 Sales by Store's City** and click on the Edit Content icon button at the upper-right corner of the chart's area. As soon as the **Chart Designer** dialog box appears, click on the **Edit Query** button and open the **Query Editor** dialog box.
- Change the condition to make it parametric. To do this, type **{Country}** in the **Value** field, as shown in the following screenshot:



8. Unfortunately, we need to set the values in the **Chart Designer** dialog again. Select **Stores city** from the **Series** column's drop-down list. Select **NONE** from the **Categories** column's drop-down list. Finally, select **Store Sales** from the **Values** drop-down list.
9. In this case, because of the parametric condition set in the query, after we set all of these values, the **Display** area of the **Chart Designer** dialog will not display a preview of our chart. Click on **OK** to confirm the changes and close the **Chart Editor** dialog.  
As soon as the **Chart Editor** dialog closes, we will see the chart appear in the dashboard designer canvas. That confirms the correctness of the setup we made, because it means that the default **Country** parameter's value is correctly fetched from the **Country** parameter's values list and processed in our query.
10. Select **General Settings** from the **Objects** explorer, and then choose **Properties** from the tabbed set on the right. Type `Dashboard 2` in the **Page Title** field.
11. Choose the **Save As** button from the PUC toolbar. Go to the `/public/Pentaho BA Cookbook/Dashboards` location and save the dashboard as `Dashboard 2`.

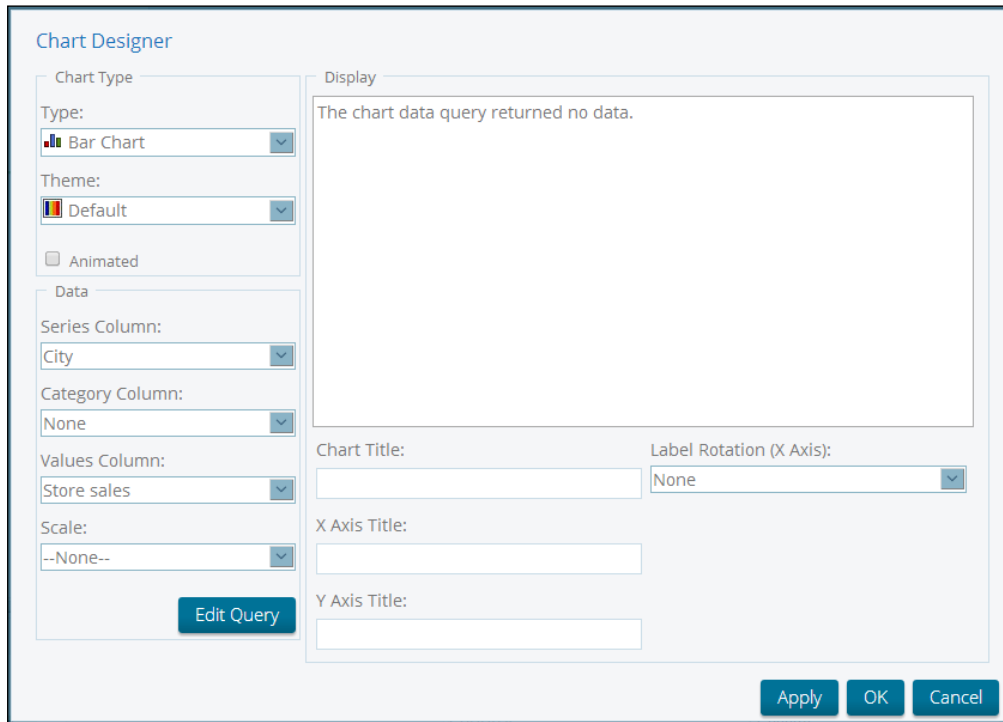
## How it works...

Prompts are a way for the dashboard to get user input and filter data that will come out from our dashboard's query. Prompts can apply to one or more items present in the dashboard. To define one or more prompts (there is no limit to the number of prompts we can define), select the **Prompts** object from the **Objects** explorer. The **Prompts** object's list table appears as shown in the following screenshot:



To add a new prompt, click on the Add icon located in the right-hand side of the **Prompts** table list's header. The **Prompt** dialog box appears. Type the name of the prompt in the **Name** field. Select the type of control we want this prompt to appear as to the user, the data type, and eventually the control properties. This dialog box is the same as the one we saw in *Chapter 5, Creating Reports Using Pentaho Interactive Reporting*, when we talked about adding prompts to interactive reports. After declaring the prompt, click on the **OK** button; the new **Prompt** dialog will display in the list of **Prompts**.

After we declare **Prompt**, it's time to update the metadata query that fills the chart and makes it responsive to the prompt change. To do this, choose **Top 10 Sales by Store's City** from the **Objects** explorer and click on the **Edit Content** button at the upper-right corner of the chart's area to open the **Chart Designer** dialog box, as shown in the following screenshot:



The **Chart Designer** dialog declares all the chart's properties. The main chart properties are described as follows:

- ▶ The **Type** drop-down list sets the type of chart we are going to use. We have **Bar Chart**, **Line Chart**, **Pie Chart**, **Area Chart**, and **Dial Chart**.
- ▶ The **Theme** drop-down list lets us select the color theme for our chart. For example, the color combination for bars in a bar chart is specified in the theme's definition.
- ▶ The **Data** field sets specify definition of fields that populate chart in terms of **Series Column**, **Category Column**, and **Value Column**. We can also set a **Scale** value in case the chart exceeds the space available for that chart. Finally, the **Edit Query** button lets us set the metadata query that will populate the chart.
- ▶ The **Display** field set shows a sort of preview of the chart. We can also set the **Chart Title** and axes labels in the **Chart Designer** dialog box.

In this case, as soon as the **Chart Designer** dialog box appears, click on the **Edit Query** button and open the **Query Editor** dialog box. Change the condition to make it parametric. To do this, type `{Country}` in the **Value** field. With this particular convention, the **Country** parameter (the prompt) is seen as a variable and its value is automatically substituted in the query.

When the **Query Editor** dialog box closes and we're back to the **Chart Designer** dialog box, we need to set the values of the series' fields again. Select **Stores city** from the drop-down list of **Series Column**. Select **None** from the drop-down list of **Category Column**. Finally, select **Store Sales** from the drop-down list of **Value Column**. Click on **OK** to close the dialog and have the prompt fully working.

### See also

- ▶ To recap the details of the **Prompt** dialog's declaration, see the *Adding prompts to get user input* recipe in *Chapter 5, Creating Reports Using Pentaho Interactive Reporting*

## Creating a multiple-content dashboard

As we said in the introduction of this chapter, a dashboard contains an aggregation of multiple BI content (not necessarily just charts) in one single place. This recipe will show how we can aggregate multiple content on one single dashboard. Remember that this recipe makes use of Pentaho Dashboard Designer, a tool available only in the EE version of Pentaho.

### How to do it...

The following steps detail how to build a dashboard to aggregate multiple content sources from our BA system:

1. Start a new dashboard from scratch by either clicking on the **Create New** button in the **Home** perspective and then clicking on **Dashboard**, or by navigating to **File | New | Dashboard** from the PUC menu.
2. The dashboard editor canvas will open. Choose the layout template of our dashboard by selecting the **Template** tab from the tabbed view and then clicking on **1 over 2**. This template lets us insert three pieces of content in our dashboard: one in the upper zone and the remaining two in the adjacent zones below.
3. In **Solution Browser** on the left-hand side, look for the `/public/Pentaho BA Cookbook/Analysis Reports` folder. Get **Analysis Report 6** and drag it to the **Untitled 2** area on the lower-left side of the dashboard canvas. Rename this area by typing `Sales by Country` in the **Title** field.



4. From the same location in the solution, get **Analysis Report 5** and drag it to the **Untitled 1** area on the upper side of the dashboard canvas. Rename this area by typing `Sales by Store Type` in the **Title** field.
5. From **Solution Browser** on our left, look for the `/public/Pentaho BA Cookbook/Interactive Reports` folder. Get **Interactive Report – Sample 5** and drag it to the **Untitled 3** area on the lower-left side of the dashboard canvas. Rename this area by typing `Detailed Sales by Store Type` in the **Title** field.
6. Select **General Settings** from the **Objects** explorer. Then, from the tabbed set on the right, choose **Properties**. Type `Dashboard 3` in the **Page Title** field.
7. Choose the **Save As** button from the PUC toolbar. Go to the `/public/Pentaho BA Cookbook/Dashboards` location and save the dashboard as `Dashboard 3`.

### How it works...

Existing content can be reused and aggregated in the dashboard canvas. In our case, we selected the **1 over 2** template to insert three pieces of content in our dashboard: one in the upper zone and the remaining two in the adjacent zones below.

The content here is an aggregation of the existing content that we developed in the previous chapters' recipes. In Solution Explorer on the left, look for the `/public/Pentaho BA Cookbook/Analysis Reports` folder. Get **Analysis Report 6** and **Analysis Report 5** and drag them to the **Untitled 2** and **Untitled 1** areas on the lower-left side of the dashboard canvas, respectively. As we can see, adding the existing content to a dashboard is extremely easy. Then, look for the `/public/Pentaho BA Cookbook/Interactive Reports` folder. Get **Interactive Report – Sample 5** and drag it to the **Untitled 3** area on the lower-left side of the dashboard canvas. Finally, perform the following actions:

- ▶ Rename the **Untitled 2** area by typing `Sales by Country` in the **Title** field
- ▶ Rename the **Untitled 1** area by typing `Sales by Store Category` in the **Title** field
- ▶ Rename the **Untitled 3** area by typing `Sales by Store Type` in the **Title** field

### See also

- ▶ To review the basics of dashboard development using Pentaho Dashboard Designer, see the *Creating a simple dashboard from scratch* recipe

## Linking different content and enabling interaction

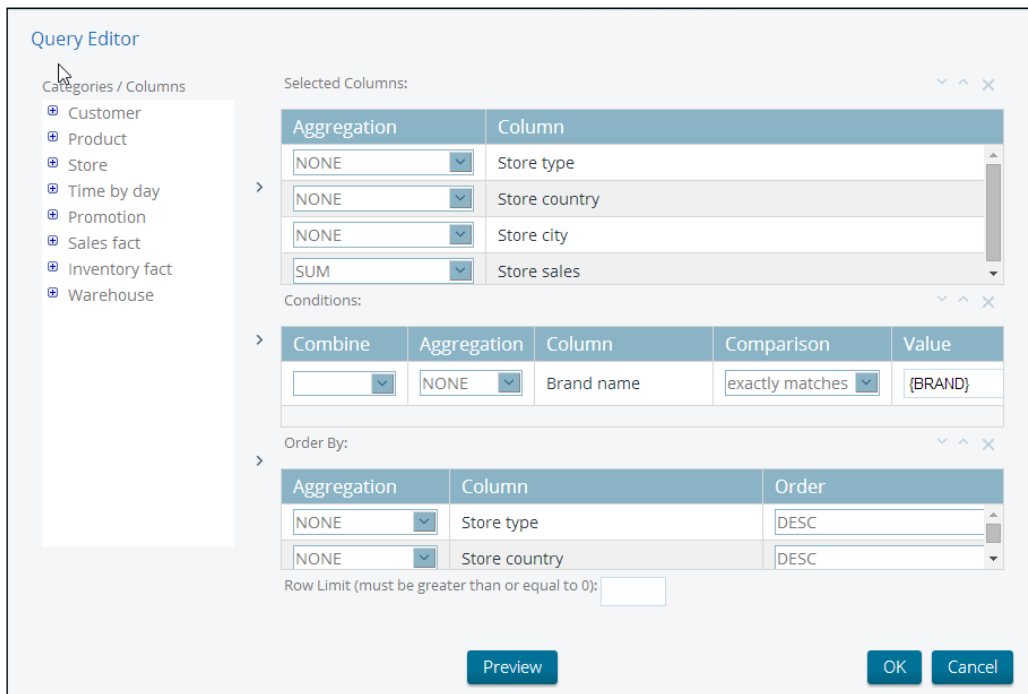
As a last recipe about Pentaho Dashboard Designer, it is interesting to analyze how we can build a simple interactive dashboard. By clicking on the bar chart's bar, we will implicitly set a parameter that will influence the table located in the upper area of the dashboard. Remember that this recipe makes use of Pentaho Dashboard Designer, a tool available only in the EE version of Pentaho.

### How to do it...

The following steps detail how to link different content items in the dashboard to make our dashboard responsive to a user's interactive events:

1. Start a new dashboard from scratch by either clicking on the **Create New** button in the **Home** perspective and then clicking on **Dashboard**, or by navigating to **File | New | Dashboard** from the PUC menu.
2. The dashboard editor canvas will open. Choose the layout template of our dashboard by selecting the **Template** tab from the tabbed view and then by clicking on **1 over 1**. This template organizes two pieces of content in our dashboard, stacked vertically one over the other.
3. From the **Objects** explorer located on the left side of the band, select the **Untitled 1** item. Go to the upper-right corner of the **Untitled 1** chart's area, click on the first drop-down list button, and select **Chart** from the drop-down list.
4. The **Query Editor** dialog box opens. From the **Product** category, select **Brand name** and click on the first button in the middle of the two sets of lists to move this field to the **Selected Columns** list. Perform the same operation with the **Store sales** field from the **Sales** category.
5. From the **Sales** category, select **Store Sales** and click on the third button in the middle of the two sets of lists to move this field to the **Order By** list. Choose **DESC** from the **Order** drop-down list. Finally, type 10 in the **Row limit** field (even here, we want to see the top 10 brands by sales and not all the brands).
6. Click on the **Preview** button to check if everything works. If it works, close the **Preview** dialog, click on **OK** to confirm the query, and close the **Query Editor** dialog.
7. The **Chart Designer** dialog opens. Select **Brand name** from the **Series** column's drop-down list. Select **NONE** from the drop-down list of **Categories Column**. Finally, select **Store Sales** from the **Values** drop-down list. Finally, in the **Chart title** field, type `Click on chart's bars to filter table below by Brand Name`.
8. After we set all of these values, the **Display** area of **Chart Designer** will display a preview of our chart. Click on **OK** to confirm the changes and close the **Chart Editor** dialog.

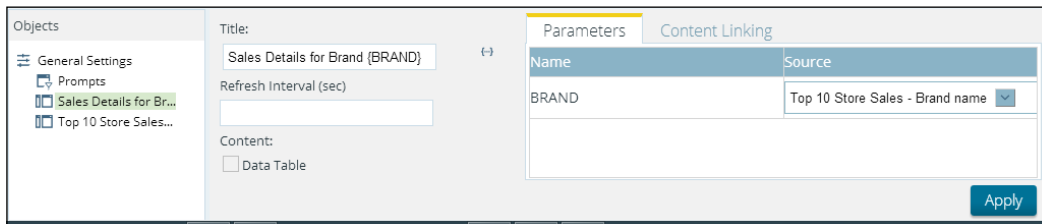
9. In the **Title** field, type `Top 10 Brand by Sales`. Then, in the tabbed pane, to the right of the **Title** field, select the **Content Linking** tab and then check the **Enabled** flag for the **Brand name** field.
10. Now, from the **Objects** explorer, select the **Untitled 2** item. Go to the upper-right corner of the **Untitled 2** chart's area, click on the first drop-down list button, and select **Data Table** from the drop-down list.
11. Select the **Cookbook Sample** data source. Then, the **Query Editor** dialog box opens.
12. From the **Store** category, select the following fields in the order they are given here and move them to the **Selected Columns** list: **Store type**, **Store country**, and **Store city**. Perform the same operation with the **Store sales** field from the **Sales** category (see the following screenshot for reference).
13. From the **Product** category, move **Brand name** to the **Condition** list on the right. Set the value of the condition to `{BRAND}` (see the following screenshot for reference):



14. From the **Store** category, select the following fields in the order given here and move them to the **Order by** list: **Store type**, **Store country**, and **Store city** (see the previous screenshot for reference).
15. Do not click on **Preview** because it is a parameterized query. Click on **OK** and close the dialog.

The chart in the **Untitled 2** canvas's zone will get an error because we don't assign any value to the parameter in the table's query; so don't be worried. This is because the parameter will only be evaluated at runtime upon user interaction. At design time, the parameter is not valued so the query gets an error.

- Look at the area below the designer canvas. In the **Title** field, type `Sales Details for Brand {BRAND}`. This will contextualize the chart's title at runtime by specifying which brand we are looking the sales' details. Then, in the tabbed pane, to the right of the **Title** field, select the **Parameter** tab. Then, from the **Source** dropdown list, select the **Top 10 Store Sales - Brand name** item. We just created the link between the selection of an item in the bar chart above and the reaction of the table below. Now, even if we're in design mode, try to click on a bar chart's bar; the table will immediately react (see the following screenshot for details):



- Select **General settings** from the **Objects** explorer. Then, from the tabbed set on the right, choose **Properties**. Type `Dashboard 4` in the **Page Title** field.
- Choose the **Save As** button from the PUC toolbar. Go to the `/public/Pentaho BA Cookbook/Dashboards` location and save the dashboard as `Dashboard 4`.

## How it works...

This recipe shows how to link different items in the same dashboard. As soon as a user clicks on a dashboard, it implicitly sets a parameter's value. It is the same as selecting a parameter from a prompt with the difference that, in our case, everything starts from a chart or table interaction.

As an example, let's define a dashboard with a bar chart and a table. We would like that every time a user clicks on a bar chart, the table will update accordingly. The dashboard starts by selecting a template named **1 over 1**. This template organizes two pieces of content in our dashboard, stacked vertically one over the other.

In the upper template's region, we put the bar chart. In the lower region, we define a table. The query we defined to fill the bar chart gives the store sales by brand name. To activate the first part of the link between the bar chart and the table on the second region, select **Top 10 Brand by Sales** from the **Objects** explorer. Then, in the **Properties** tabbed pane, to the right of the **Title** field, select the **Content Linking** tab and check the **Enabled** flag for the **Brand name** field. This will activate the notification of an event that is automatically fired and publish the value of **Brand name** associated with the bar over which the user clicked.

At this point, we defined the starting point; we need to wire this event to a parameter in a table in the second region. To do this, while defining the metadata query that will populate the table, set **Condition** to **Brand name** by using a parameter and typing the value of the condition as `{BRAND}`. When the metadata query closes, select the **Parameter** tab. Then, from the **Source** drop-down list, select the **Top 10 Store Sales - Brand name** item. This last action closes the link between the two widgets. Now, even if we're in design mode, try to click on a bar chart's bar; the table below it will immediately react.

### See also

- ▶ Review the way prompts and parameters are defined by looking at the *Adding prompts to get user input* recipe
- ▶ If you want to review how to create a Pentaho Dashboard from scratch, look at the *Creating a simple dashboard from scratch* recipe

## Creating dashboards using CDE

CDE is a dashboard editor made by the Pentaho community. It is the *de facto* standard tool to design dashboards in the CE version of Pentaho. For the EE version, where a dashboard design tool is already available, it is another tool to choose to design a dashboard.

Another very important difference between the two tools is that the EE editor is really for end users to build simple dashboards. CDE is mainly targeted at developers and provides great flexibility, allowing for construction of very powerful and beautiful data visualizations taking full advantage of **protovis**. Moreover, these days Pedro Alves, the creator of CTools, announced the availability of CDE dashboards that support **Bootstrap** (one of the most used responsive layout frameworks) and also the availability of chart components that use `d3.js` (the next generation chart library born from the same creator of **protovis**). These are two other major steps that contribute once more to the wide adoption of this tool.

CDE is installable from the Marketplace the same way we saw in *Chapter 6, Creating Analysis Reports*, when installing Saiku (see the *There's more...* section of the *Creating an analysis report using Saiku* recipe) and leverages its power on a set of other tools and frameworks (CCC, CDA and CDF) that are part of CTools. The dashboard we are building in this recipe will make use of an advanced table component called **BTable** that constitutes another important contribution from the Pentaho community. BTable is a table component for CDE with advanced analytics functionalities for the easy navigation and aggregation of the data contained in the table. This recipe will show how we can easily design a dashboard using CDE.

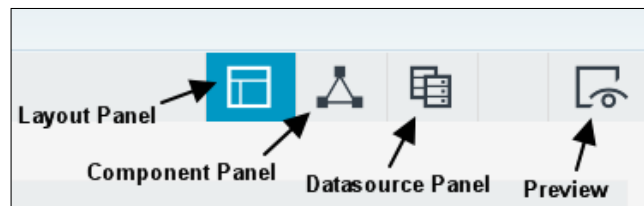
## Getting ready

As a further requirement, to get ready for this recipe, CDE and all of its dependencies must be installed as a plugin of our Pentaho BA Server installation.

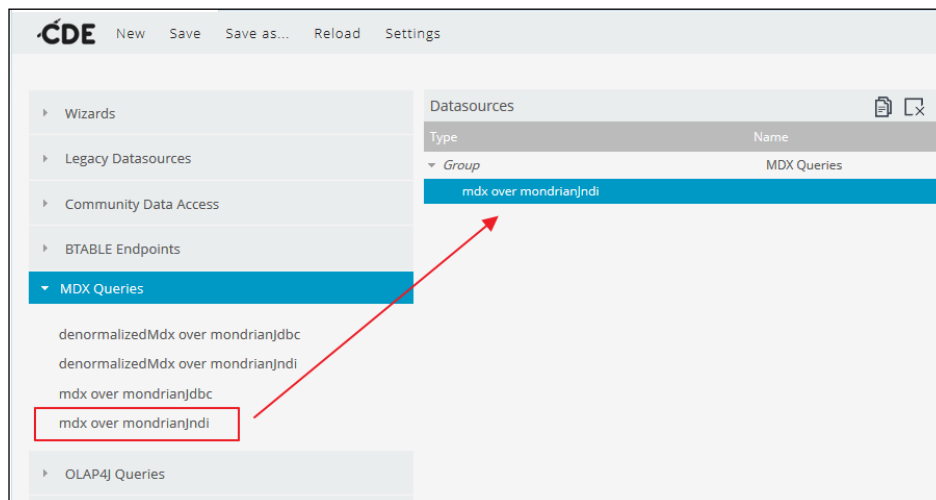
## How to do it...

The following steps detail how to build a very simple dashboard using CDE that shows a chart and a table with advanced navigation features:

1. Start a new CDE dashboard from scratch by navigating to **File | New | CDE Dashboard** from the PUC menu.
2. The CDE user interface appears. Look at the button icons to the right of the CDE toolbar. Click on the **Datasource Panel** icon button shown in the following screenshot:

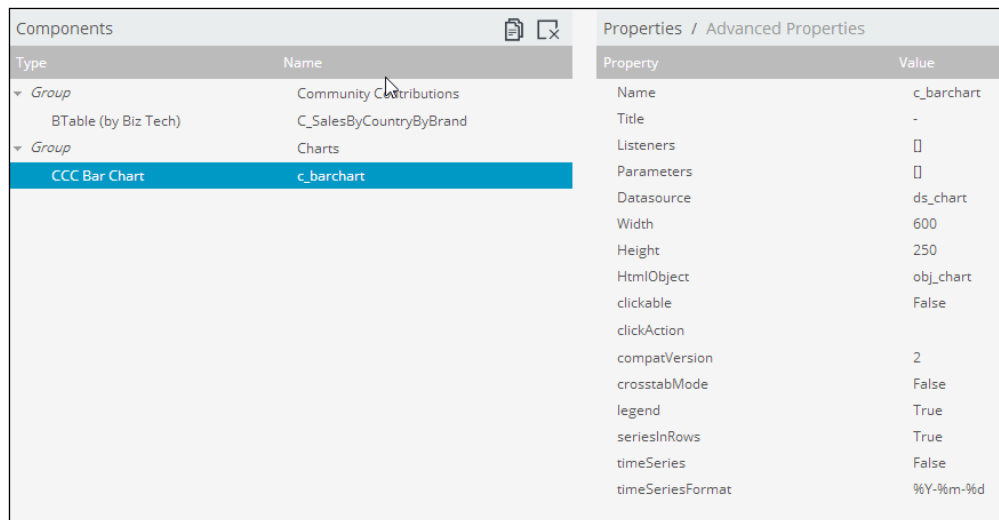


3. As soon as the datasource panel appears, navigate to **MDX Queries | mdx over mondrianJndi** from the left-hand side menu. A new entry named **mdx over mondrianJndi** appears in the **Datasources** section of the datasource panel. The **Properties** section, to the right of the **Datasources** section, shows the set of properties associated with the selected datasource type, as shown in the following screenshot:



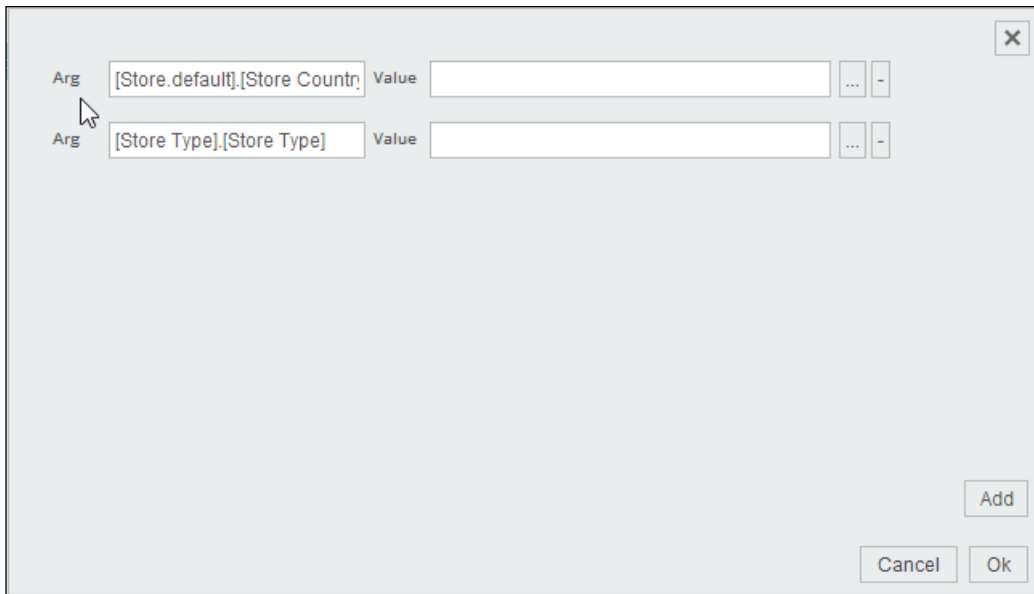
4. Now, follow the instructions to fill the required set of **Properties** and configure the **mdx over mondrianJndi** datasource type. Look for the **Name** property and type `ds_chart`. Look for the **JNDI** property and type `foodmart_mondrian`. Then, in the **Mondrian Schema** property, type `Cookbook Sample Schema`.
5. Look for the query property, open the wizard by clicking on the button on the right, and type the following MDX query:
 

```
with set [Store.default_Store Country_Set] as
  '{[Store.default].[Store Country].Members}'
set [Store Type_Store Type_Set] as '{[Store Type].[Store Type].Members}'
set [Measures_Set] as '{[Measures].[Store Sales]}'
select NON EMPTY [Measures_Set] on COLUMNS,
NON EMPTY Crossjoin([Store.default_Store Country_Set], [Store Type_Store Type_Set])
on ROWS from [Cookbook Sales]
```
6. As soon as we have typed in the query, click on **OK** and close **Query Wizard**.
7. Click on the Component Panel icon button. As soon as **Component Panel** appears, navigate to **Community Contributions | BTable** from the left-hand side menu. Then, from the same menu, navigate to **Charts | CCC Bar Chart**.
8. Now, follow the instructions to fill the required set of **Properties** and configure the **CCC Bar Char** component type. Look for the **Name** property and type `c_barchart`. Look for the **Datasource** property and type `ds_chart`. Look for the **Height** property and type `250`; then, in the **Width** property, type `600`. Look for the **HtmlObject** property and type `obj_chart`. Finally, look for the  **CrosstabMode** property and choose **False**. Then, from the **seriesInRows** property, choose **True**. Check the following screenshot for details:



We have successfully configured **CCC Bar Char**. Now, follow these instructions to configure the **BTable** component:

1. Look for the **Name** property and type `c_SalesByCountryByBrand`.
2. Look for **Dimensions** and click on the property value. Click on **Add** to add another row in the dialog. Type `[Store.default].[Store Country]` in the first row's **Arg** field. Type `[Store Type].[Store Type]` in the second row's **Arg** field (see the following screenshot for details). Click on **OK** to close the dialog.



3. Look for the **Measures** property and click on the property value. Click on **Add** to add another row in the dialog. Type `[Measures].[Store Sales]` in the first row's **Arg** field. Click on **OK** to close the dialog.



4. Look for the **JNDI** property and type `foodmart_mondrian`. Look for the **Catalog** property and type `Cookbook Sample Schema`. Look for the **Cube** property and type `Cookbook Sales`. Finally, look for the **HtmlObject** property and type `obj_SalesByCountryByBrand`. Check the following screenshot for details:

Components		Properties / Advanced Properties	
Type	Name	Property	Value
Group	Community Contributions	Name	C_SalesByCountryByBrand
BTable (by Biz Tech)	C_SalesByCountryByBrand	Subtotals on Dimensions	False
Group	Charts	Non Empty Columns	True
CCC Bar Chart	c_barchart	Measures	[[["Measures],[Store (...]
		Pivot Dimensions	[]
		Filters / File Replacements	[]
		Order By	[]
		Measures on Columns	True
		Non Empty Rows	True
		File Path	-
		Grand Total on Dimensions	False
		Dimensions	[[["Store.default],[ (...]
		Grand Total on Pivot	False
		Subtotals on Pivot	False
		Show Applied Filters and Sorts	True
		Table Settings From File	True
		Catalog	Cookbook Sample Schema
		Jndi	foodmart_mondrian
		Cube	Cookbook Sales
		Listeners	[]
		HtmlObject	obj_SalesByCountryByBrand
		clickAction	

5. Now that we have configured data sources and components, it is time to define a layout and organize the components in the layout. Click on the Layout Panel icon button.
6. From **Layout Structure**, click on the Add Row icon button five times and add five rows to the layout.
7. Select the first **Row** element and click on the Add Column icon button, and then select the column just added and click on the Add HTML icon button. Select the HTML element from **Layout Structure**. In the **Name** property, type `title`. Then, look for the HTML property and click on the icon button to the right of the field. The HTML editor window opens. Type the following code:
 

```
<h3>CDE Dashboard</h3>
```
8. Click on **OK** to close the dialog.

9. Select the second **Row** element and click on the **Add Column** button, and then select the column just added. In the column's **Name** property, type `chart1_title`. Then, click on the **Add HTML** button. Select the just added HTML element from **Layout Structure**. Look for the **HTML** property and click on the icon button to the right of the field. The HTML editor window opens. Type the following code:

```
<strong>Sales by Store Type</strong>
```

10. Click on **OK** to close the dialog.
11. Select the third **Row** element and click on the **Add Column** button. Select the column we just added, and in the **Name** property, type `obj_chart`. Look for the **Span Size** property and type 12.
12. Select the fourth **Row** element and click on the **Add Column** button, and then select the column just added. In the column's **Name** property, type `table1_title`. Then, press the **Add HTML** button. Select the just added **Html** element from **Layout Structure**. Look for the **HTML** property and click on the icon button to the right of the field. The HTML editor window opens. Type the following code:

```
<strong>Navigate Sales Details</strong>
```

13. Click on **OK** to close the dialog.
14. Select the fifth **Row** element and press the **Add Column** button, and then select the column just added in the **Name** property and type `obj_SalesByCountryByBrand`. Look for the **Span Size** property and type 12.
15. Click on the Preview icon button. The dashboard preview is displayed.
16. Click on the **Save as CDE** menu entry. The **Save as** dialog appears. Navigate the Pentaho Solution to `/public/Pentaho BA Cookbook/Dashboards`, save the dashboard by typing `cde-sample-dashboard.wcdf` in the **File Name** field, and type `CDE Sample Dashboard` in the **Title** field. Click on the **OK** button and close the dialog.

## How it works...

CDE lays its foundations over the following frameworks:

- ▶ **Community Charting Component (CCC)**: This is an advanced charting library based on `protovis`.
- ▶ **Community Data Access (CDA)**: This is a frontend data access middleware that lets you expose all of the Pentaho data sources in an easy way. The major advantage of using CDA is that it makes all data look the same to all Pentaho tools that can access it. Basically, it lets us define a set of endpoints called data access, which we can call directly through a REST with a variable set of parameters depending on the function required. The call gives a result set in various formats such as JSON (default), XML, CSV, and others.

- ▶ **Community Dashboard Framework (CDF):** This is the JavaScript framework that manages the lifecycle of a dashboard and all its components.
- ▶ **Blueprint:** This is a CSS framework used to give an organized and clear representation of the web page and its components.

Basically, with CDE, we start building a dashboard by building a layout where distributing the components (the charts, the table, and the controls) by hooking them at certain points is defined by the designer. A layout is configured in **Layout Panel**; there, we define rows and columns to build a sort of matrix, we add HTML fragments for titles or descriptions, and we define spacing in between the various dashboard's elements or between the dashboard borders. Every element of the matrix has a name, plus other properties that are different in numbers depending on the element type. Names are very important because they became the hooks for our charts.

During or after the definition of a layout, we can define the components of our dashboard. In a generic sense, a component for CDE is anything I can put on a web page. Therefore, we can consider components charts, tables, form controls, Pentaho content elements, or other similar things. Every component has a name and a set of properties that can be variable depending on the type of component. Moreover, any component can have a property called **HtmlObject**. This property must be configured with the **Name** of the layout element to which we want to hook this component. Some of the components must be filled with data coming from our Pentaho data sources. The link is configured through a set of properties different in name and number, depending on component type and the data source type we are going to link to get the data.

CDE data sources are a similar concept but more specific to what data sources are in Pentaho. Basically, a data source in CDE represents the association of the physical data source (JDBC JNDI, JDBC Native, Analytic, Metadata, and others) and the strategy to extract data from that source (SQL Query, MDX Query, and others). Therefore, in CDE, a data source is a true source of data in the literal term because it represents the data that is coming out. We have various kinds of data sources depending on our needs.

BTable, the component we used as a table, is a particular table with advanced analytics functionalities. It is a very interesting piece of software that can have a wide range of use in our dashboards. Made by the Italian Pentaho community contributor *BizTech*, it is an advanced table component that gives the user advanced navigation capabilities on the data underneath. By clicking on column headers, we can quickly enable a contextual menu with a wide set of choices that let us add or remove dimensions, configure filters, add measures, have subtotals and grand totals or other interesting aggregated measures, and many other useful features (see the following screenshot for details).

Sales Details  
 FILTERS: none

Store Country	Store Type	Store Sales
Canada	Deluxe Supermarket	
	Mid-Size Grocery	
	<b>Total</b>	
Mexico	Deluxe Supermarket	
	Gourmet Supermarket	
	Mid-Size Grocery	
	Small Grocery	
USA	Supermarket	
	<b>Total</b>	
	Deluxe Supermarket	
	Gourmet Supermarket	4,032
	Mid-Size Grocery	3,074
	Small Grocery	1,085
	Supermarket	28,120
	<b>Total</b>	49,391

Measures -> Store Sales

- Add
- Change
- Remove
- Sort
- Pivot
- Manage Filters
- Hide Filters and Sorts
- Table Settings
- Show MDX
- Reset
- File
- Export to Excel

Customer Count

- Profit
- Profit Growth
- Profit last Period
- Sales Count
- Store Cost
- Unit Sales
- Unit Sales

Before

After

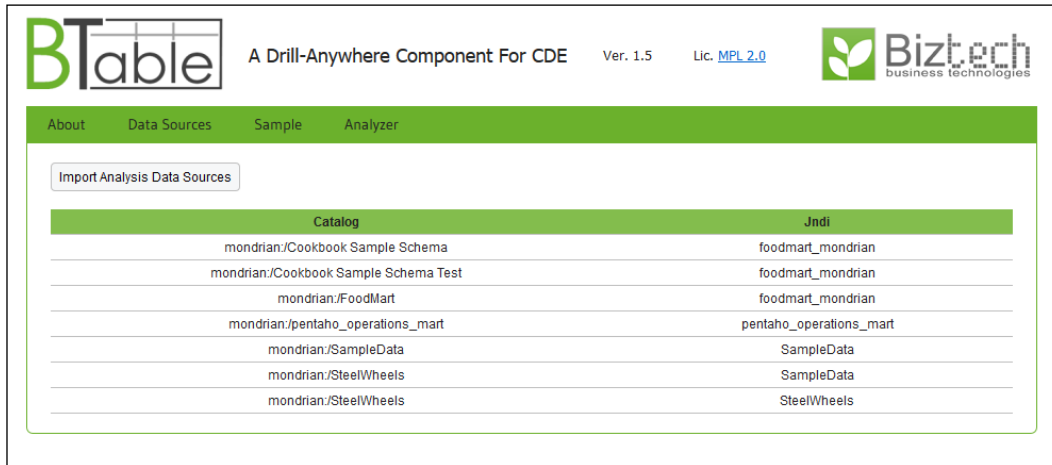
We strongly suggest that you give it a try and evaluate the power of this component by navigating the chart on the recipe's sample dashboard.

### There's more...

In this recipe, we saw how to build a simple dashboard and how to use BTable—a table component for CDE with advanced analytics functionalities for the easy navigation and aggregation of the data contained in the table. CDE will easily install from the Marketplace with a procedure similar to the procedure we saw in *Chapter 6, Creating Analysis Reports*, in the *Creating an analysis report using Saiku* recipe. BTable installs from the Marketplace too but has a tricky configuration part. Let's show how we can configure the BTable component so that it will be ready for use in our dashboard.

## Configuring the BTable component for use in CDE dashboards

Before BTable is ready to use in CDE dashboards, we must configure it by enabling the Pentaho Analysis data sources to be used in BTable. To do this, navigate to **Tools | BTable** from the PUC menu. The BTable introductory screen appears; then, click on the **Data Sources** menu item. The **Data Source** configuration wizard appears; then, click on **Import Analysis Data Sources**.



As soon as the import operation finishes successfully, the set of imported data sources will be shown and we will be ready to use the table component.

### See also

- ▶ As a reference to the MDX language, we can look at the Mondrian documentation at <http://mondrian.pentaho.com/documentation/mdx.php>.
- ▶ Another good source of information about MDX is the official Microsoft documentation at <http://msdn.microsoft.com/en-us/library/ms145506.aspx>. Remember that Mondrian implements most but not all of the Microsoft standard, so it is always better to keep both sources on hand.

# 9

## Scheduling Content

In this chapter, we will cover the following recipes:

- ▶ Scheduling task execution
- ▶ Updating scheduling properties
- ▶ Deleting schedules
- ▶ Executing a scheduled task ahead of time
- ▶ Filtering schedules in the schedule entries list
- ▶ Running a user's tasks in background
- ▶ Checking schedule execution
- ▶ Stopping running executions
- ▶ Preventing the creation of schedules by content
- ▶ Preventing the creation of schedules by setting blackout time intervals

### Introduction

Sometimes, in our BA application, we have processes that require a long wait so they can be executed. This could be a long-running report, which our users need to execute weekly to have a summary of sales by region, or a process to start refreshing the data in our cube. Any one of these activities could take a long time, but we will be fine if we can execute them on a timely basis by scheduling them.

Pentaho BA Server has a scheduler functionality that can help run processes at a certain time or even at recurring time intervals. Users will find the result of the execution in a specified output directory (a Pentaho solution folder) and can access it as soon as it is available after the scheduled content has terminated the execution. Moreover, the user will also have the ability to check his/her active schedules using a console, where he/she can list all of the active schedules with the ability to manually start a task or delay another.

## Scheduling task execution

Let's suppose we have a report, and it is taking a lot of time to execute. We don't want to stay in front of our computer for a long time, waiting for our report to generate. The only thing we can do is to schedule it so that the report is available immediately when required. This recipe will explain how we can schedule a report for execution.

### Getting ready

To get ready for this recipe, the following conditions must be met:

- ▶ The Pentaho user must be successfully logged in to the Pentaho User Console
- ▶ The user role must have the **Schedule Content** operational permission assigned
- ▶ The user must be in the **Schedule** perspective

### How to do it...

Perform the following steps to schedule a long-running report for execution:

1. Navigate the Pentaho solution looking for the report you would like to schedule for execution. As an example, go to `/Public/Pentaho BA Cookbook/Pentaho Reporting` and look for **Pentaho Report Sample 11**.
2. Select the report by clicking on it. Look for the **Schedule...** menu entry in the **File Actions** menu on the right and click on it.
3. The **New Schedule** dialog box wizard appears. Choose the name of the generated output file and the location where the file is stored. Click on the **Next** button. The **New Schedule** dialog box is shown in following screenshot:

New Schedule

Schedule Name: *This will also be the name of the generated content.*

Report Designer Sample 11

Generated Content Location:

/home/sergio

4. Now, the **New Schedule** dialog box asks for the recurrence of our schedule in terms of how the schedule will be executed and when it will expire. The fields that appear in the dialog relate to the choices in the **Recurrence** combobox. Let's imagine, for the sake of simplicity, that we leave the default selection **Run Once**. Fill the **Start Time** field by providing a time to execute the schedule and the **Start Date** field by providing the date on which the execution will be started. Click on **Next**.
5. In this new dialog, if the report requires a set of input parameters, provide the report that you would like to execute to the parameters' values. This set of report parameters is the same that the report will require in case it needs to be started manually. Click on **Next**.
6. This is the last dialog in the **New Schedule** wizard. It asks if you want to be informed by Pentaho as soon as the report is printed. Click on **OK** to confirm the schedule.
7. Go to the schedule perspective by selecting it from the top-left drop-down list. See your new schedule in the active schedules list as follows:

Schedule Name	Repeats	Source File	Output Location	Last Run	Next Run	Created By	Status
UpdateAuditData	Every 30 minutes at 21:45:29	/public/pentaho-operations-mart/update_audit_mart_data/Updat	/home/admin	2014 Mar 22 03:45:29	2014 Mar 22 04:15:29	admin	NORMAL
Report Designer Sample 11	Run Once	/public/Pentaho BA Cookbook/Pentaho Reporting/Report Designer Sample 11	/home/sergio	2014 Mar 22 03:42:55	2014 Mar 23 17:49:00	admin	NORMAL
PentahoSystemVersionCheck	Every day at 23:25:15	PentahoSystemVersionCheck	-	2014 Mar 21 23:25:15	2014 Mar 22 23:25:15	system session	NORMAL

## How it works...

Scheduling a report to be executed is a fairly easy task. We can schedule anything accessible in the Pentaho solution that can be started by a user.

First, we need to look for the content we want to start by going through the solution. As soon as we find the content to schedule, we'll click over it and select **Schedule...** from the **File Actions** menu. After we have selected this menu item entry, the **New Schedule** wizard starts, and the first dialog asks us for a name and the location of the content that will be provided on the execution of the scheduled content.



After we have clicked on the **Next** button, the wizard asks us for information about the recurrence of the schedule, and we can choose the type of recurrence by selecting an item from the **Recurrence** dialog box, which is shown in the following screenshot:

The screenshot shows a 'New Schedule' dialog box with the following fields and options:

- Recurrence:** A dropdown menu with 'Daily' selected.
- Start Time:** Three dropdown menus showing '01', '00', and 'AM'.
- Recurrence pattern:** Two radio button options: 'Every' (selected) with a text input field, and 'Every weekday'.
- Range of recurrence:** A 'Start' text input field with '2014-03-23', and two radio button options: 'No end date' (selected) and 'End by:' with a text input field containing '2014-03-23'.
- Buttons:** 'Back', 'Next >', and 'Cancel' buttons at the bottom right.

The various recurrence options are described briefly, as follows:

- ▶ **Run Once:** This is the default choice in the **Recurrence** drop-down list. We only need to provide the start date and start time to complete our schedule setup.
- ▶ **Seconds, Minutes, Hours:** This option lets us schedule the execution for the specified number of seconds, minutes, or hours that we want to run the content for. After we have provided that information, the schedule's setup asks for a time frame in terms of the start and end dates (fill the **Start** and **End by** fields) that the schedule must run between. If we want the schedule to always be active, we'll choose **No end date** instead of providing a value for the end date in the **End by** field.
- ▶ **Daily:** This entry lets us schedule the content object on a daily basis. First, we are asked to provide a start date that our schedule will be available from. Then in the **Recurrence pattern** portion, we can either provide the ordinal of the days in the week when we want the schedule to be executed or choose **Every weekday** to indicate the execution for every day of the week (see the following screenshot for details). Finally, the setup asks for a timeframe in terms of the start and end dates (fill the **Start** and **End by** fields) that the schedule must run between. If we want the schedule to always be active, we'll choose **No end date** instead of providing a value for the end date in the **End by** field.

Recurrence pattern

Every  day(s)

Every weekday

- ▶ **Weekly:** This entry lets us schedule the content object on a weekly basis. First, we are asked to provide the start date that our schedule will be available from. Then, in the **Recurrence pattern** portion, we must set the days when the schedule will run by checking the checkbox for the desired days. We can check as many days as we want (see the following screenshot for details). Finally, the setup asks for a timeframe in terms of the start and end dates (fill the **Start** and **End by** fields) that the schedule must run between. If we want the schedule to always be active, we'll choose **No end date** instead of providing a value for the end date in the **End by** field.

Recurrence pattern

Recur every week on:

Sunday  Monday  Tuesday  Wednesday

Thursday  Friday  Saturday

- ▶ **Monthly:** This entry lets us schedule the content object on a monthly basis. First, we are asked to provide a start date when our schedule will be available from. The **Recurrence pattern** portion, in this case, can be specified in two ways. We can either specify the day of the month that the schedule must run on (**Day x of every month**) or specify which day (Monday to Sunday) of which week (first to fourth or the last) the schedule must run on (see the following screenshot for details). Finally, the setup asks for a timeframe in terms of the start and end dates (fill the **Start** and **End by** fields) that the schedule must run between. If we want the schedule to always be active, we'll choose **No end date** instead of providing a value for the end date in the **End by** field.

Recurrence pattern

Day  of every month

The   of every month

- ▶ **Yearly:** This entry lets us schedule the content object on a yearly basis. First, we are asked to provide a start date that our schedule will be available from. We can specify the **Recurrence pattern** portion in two ways. We can either choose the month and day when the schedule must run (**Every month x**), or we can specify which day (Monday to Sunday) of which week (first to fourth or the last) of which month the schedule must run on (see the next screenshot for details). Finally, the setup asks for a timeframe in terms of the start and end dates (fill the **Start** and **End by** fields) that the schedule must run between. If we want the schedule to always be active, we'll choose **No end date** instead of providing a value for the end date in the **End by** field.

The screenshot shows a configuration window titled "Recurrence pattern". It contains two radio button options. The first option, "Every", is selected and followed by a dropdown menu showing "January" and an empty text input field. The second option, "The", is unselected and followed by a dropdown menu showing "first", another dropdown menu showing "Sunday", the word "of", and a final dropdown menu showing "January".

- ▶ **Cron:** This last choice provides the ability for us to configure a schedule entry by setting a cron string. This choice is more flexible but more powerful as well. Then, the setup asks for a timeframe in terms of the start and end dates (fill the **Start** and **End by** fields) that the schedule must run between. If we want the schedule to always be active, we'll choose **No end date** instead of providing a value for the end date in the **End by** field.

After we have successfully configured the schedule, if the report has a set of input parameters that must be filled, the wizard asks us for the input parameters. This is a new feature in this release because in previous releases, it was not possible to specify the input parameters for a scheduled report. Remember that the report will always be executed with these parameters whenever the scheduler calls it. If this is not a good choice, we must think about more sophisticated ways to give the report different input parameters depending on when the report will run.

The last schedule configuration option, the very last, is to configure the ability to be informed by the server as soon as the report is printed (or more generically, when the schedule has run). This is another new feature for this release and consists of an e-mail that BA Server will send to a specific set of e-mail addresses, which we can specify in the **To** field. The e-mail will have our report attached. We can also give a more meaningful name to the attachment (**Attachment Name**), provide a subject for our e-mail, and provide text for our message (see the following screenshot for details).

**New Schedule**

Would you like to email a copy when the schedule runs?  No  Yes

To: *Use a semi-colon or comma to separate multiple email addresses.*

Subject: Report Designer Sample 11 schedule has successfully run.

Attachment Name: Report Designer Sample 11

Message (optional):

Back OK Cancel

## See also

- ▶ The *Managing the mail server configuration* recipe in *Chapter 2, Configuring Your BA Server Instance*, for a recap on how to set up e-mail support in Pentaho.

## Updating schedule properties

Once a schedule has been created, it is available in the **Schedule** perspective and can be modified at any time. This recipe will explain how we can update the schedule's properties to change its execution frequency.

## Getting ready

To get ready for this recipe, the following conditions must be met:

- ▶ The Pentaho user must be successfully logged in to the Pentaho User Console.
- ▶ The user role must have the **Schedule Content** operational permission assigned.
- ▶ The user must be in the **Schedule** perspective.

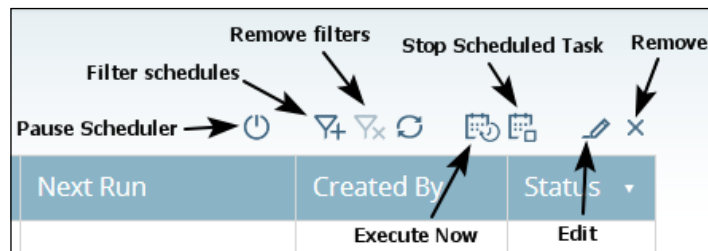
## How to do it...

Perform the following steps to update the schedule properties to change the schedule's execution frequency or another schedule's settings:

1. From the **Schedule** perspective, look for the schedule entry that you want to change the properties for.
2. Select the schedule entry from the entries list.
3. Choose the schedule entry and click on the **Edit Scheduled Task** button in the toolbar located in the top-right corner of the schedule entries list.
4. The **New Schedule** dialog box wizard will appear where you can change your schedule definition.

## How it works...

By entering the **Schedule** perspective, we are always in front of the schedule entries list. We can go through the list of schedule entries and locate the entry for which we want to change the properties. To change the entry properties, we need to select the entry from the list and choose the action we want from the toolbar in the top-right corner of the schedule entries table. The following screenshot shows the various action buttons located in the schedule toolbar in detail:



Look for the **Edit Scheduled Task** button and click on it. The **New Schedule** dialog box wizard will appear; this will let us go through the settings for our schedule. We are free to change whichever settings we want. As soon as we get to the end of the schedule entry wizard, we can click on the **OK** button to confirm the changes applied and close the dialog box. Any change will immediately be applied and the updated entry will be visible in the schedule entries list.



Remember that if the currently connected user is not a member of the administrator role, they can only update their own schedules. Only the administrator can update everyone else's schedules.

## See also

- ▶ The *Creating a new role* recipe in *Chapter 2, Configuring Your BA Server Instance*, to review the concepts about setting operational permissions to roles in the Pentaho **Administration** perspective.

## Deleting schedules

This recipe will explain how we can delete schedules from the schedule entries list to completely remove it from the system.

## Getting ready

To get ready for this recipe, the following conditions must be met:

- ▶ The Pentaho user must be successfully logged in to the Pentaho User Console
- ▶ The user role must have the **Schedule Content** operational permission assigned
- ▶ The user must be in the **Schedule** perspective

## How to do it...

Perform the following steps to delete an existing schedule from the system:

1. From the **Schedule** perspective, look for the schedule entry that we want to change the properties for.
2. Select the schedule entry from the entries list.
3. Click on the **Remove** button in the toolbar located in the top-right corner of the schedule entries list.
4. A confirmation dialog box appears, asking if we are completely sure about removing the schedule.
5. Click on the **Yes** button; the selected schedule is immediately removed from the schedules list.

## How it works...

The removal of a schedule works more or less the same as the updating of a schedule. We'll go to the **Schedule** perspective, go through the list of schedule entries, and locate the entry we need to delete from the schedule entries list. To delete the entry, we'll select the entry from the list and choose the **Remove Schedule** option from the toolbar in the top-right corner of the schedule entries table.

As soon as we click on the **Remove schedule** button, a dialog box pops up, asking us whether we are sure about deleting the schedule. If we click on the **Yes** button, the program proceeds with the deletion, and the schedule is immediately removed from the schedule entries list. Remember that deleting a schedule does not mean deleting the content produced by that schedule. The content produced will remain at the location where it was generated, and it is the user's responsibility to delete it.

The schedule removal is always valid for any schedule type, except run once. In the case of a run once schedule, we can always delete such a schedule entry, but before the schedule is executed. As soon as a schedule of this type is executed, the entry is immediately removed from the schedule list because it doesn't make any sense. This means that we can, at any time, delete a run once schedule type before the schedule is executed.



Remember that if the currently connected user is not a member of the administrator role, they can only delete their own schedules. Only the administrator can delete everyone else's schedules.

## Executing a scheduled task ahead of time

There are times when we need to start our schedule before the schedule time expires. This recipe will show us how we can do this very easily.

### Getting ready

To get ready for this recipe, the following conditions must be met:

- ▶ The Pentaho user must be successfully logged in to the Pentaho User Console
- ▶ The user role must have the **Schedule Content** operational permission assigned
- ▶ The user must be in the **Schedule** perspective

### How to do it...

Perform the following steps to force an existing schedule to start in advance:

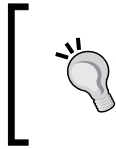
1. From the **Schedule** perspective, look for the schedule entry that we want to change the properties for.
2. Select the schedule entry you want to start in advance from the entries list.
3. Click on the **Execute now** button in the toolbar located in the top-right corner of the schedule entries list.

4. A dialog box appears, informing that the generated content will be available at the location specified during the schedule configuration.

### How it works...

Starting a schedule in advance could be a useful thing sometimes, because there could be situations when we will need our report before the expected time. In this case, we'll go to the **Schedule** perspective, go through the list of schedules entries, and from the schedule entries list, locate the entry we want to start in advance. To start the schedule entry, select the entry from the list and click on the **Execute now** button from the toolbar in the top-right corner of the schedule entries table.

The schedule execution will produce the output at the configured location.



Remember that if the currently connected user is not a member of the administrator role, they can only start the execution of their own schedules. Only the administrator can start the execution of everyone else's schedules.

### There's more...

Schedule entries are organized over a table with an available set of columns. The following list gives a brief description of each column and the related content:

- ▶ **Schedule Name:** This contains the name of the schedule that was configured during the definition of the schedule.
- ▶ **Repeats:** In case the schedule is a recurring schedule, this column will summarize the repeat strategy for the schedule. It is useful if we don't remember when the schedule for each of the rows will be executed.
- ▶ **Source File:** This contains the complete filename and path to the BI content that we scheduled.
- ▶ **Output Location:** This contains the configured location in the solution where the BI content we scheduled will be available as soon as the schedule is run successfully.
- ▶ **Last Run, Next Run:** This contains the timestamp of the last/next execution. The most important thing to note is that in case the schedule has never been executed, the **Next Run** field will be empty. It will be populated right from the first execution of the schedule.
- ▶ **Created By:** This contains the username of the user who created the schedule.
- ▶ **Status:** This column will show the last status of the related schedule entry. We can have the following status values for a particular schedule. For details about the possible values contained in this column, take a look at the *There's more...* section of the *Checking schedule execution* recipe.



Any header has a little triangular icon to the right of the header label. By clicking that icon, we can easily reorder the items in the column to ease the lookup of a schedule entry in case of long lists.

## Filtering schedules in the schedule entries list

Sometimes, we might have long lists of scheduled entries and looking for a particular schedule becomes almost too difficult. For this reason, we have the ability to filter the schedules set to find what we are looking for more easily.

This recipe will explain how we can filter the various schedule entries to ease the lookup.

### Getting ready

To get ready for this recipe, the following conditions must be met:

1. The Pentaho user must be successfully logged in to the Pentaho User Console.
2. The user role must be a member of the administrator role.
3. The user must be in the **Schedule** perspective.

### How to do it...

Perform the following steps to filter through a set of existing schedules:

1. From the **Schedule** perspective, click on the **Filter Schedules** button in the toolbar located in the top-right corner of the schedule entries list.
2. The **Filter Schedules** dialog box appears. Set the filter parameters, and click on the **OK** button.
3. The schedule entries list is updated to show only the schedule entries that respond to settings in the **Filter Schedules** dialog box.

### How it works...

Filter schedules are a useful functionality if you want to go through a long set of schedules waiting to be executed. To access the filter dialog box, from the **Schedule** perspective, click on the **Filter Schedules** button in the toolbar located in the top-right corner of the schedule entries list.

The **Filter Schedules** dialog box appears, showing all the possible filter parameters.

We can set multiple filter options in the filter dialog; some of these options are as follows:

- ▶ **Scheduled Resource:** This is used to type the name of the schedule we are looking for or a pattern to locate a set of related schedule entries.
- ▶ **Users:** This drop-down list provides the ability to specify the user who created the schedule. The users that appear in the dialog are not all the Pentaho users but only the ones who created at least one schedule entry.
- ▶ **Schedule State:** This is used to identify all of the schedules that are in a particular state. We can choose the state we are looking for from the drop-down list.
- ▶ **Schedule Type:** This drop-down list contains all of the possible types of schedules that are defined in the system. We can choose from one of them.
- ▶ **Execution Time:** This set of fields helps identify schedules that relate to a particular moment in time. We can specifically look for schedules that will start either before or after (or both) a particular moment in time. To select which of the two filter fields to apply, check on the checkbox fields to the left of the **Before** or **After** fields.

As soon as we have finished setting the filter parameters, we click on the **OK** button to update the schedules entries list to show only the schedule entries that respond to settings in the **Filter Schedules** dialog box.

## There's more...

As soon as we finished specifying a filter, from that moment all the time we look at the schedule entries list, the schedule entries displayed will refer to the filter just applied. Removing the filter we just applied is a very easy task. From the **Schedule** perspective, we click on the **Remove Filters** button in the toolbar located in the top-right corner of the schedule entries list. Immediately after we have clicked on the button, the applied filter is cleared and the schedule entries list will be displayed on all of the schedules defined in the system.

## Running a user's tasks in the background

Sometimes, it might be useful to define the execution of a task in the background. That's the case, for example, when printing a long-running report. This recipe will show you how to do this easily.

## Getting ready

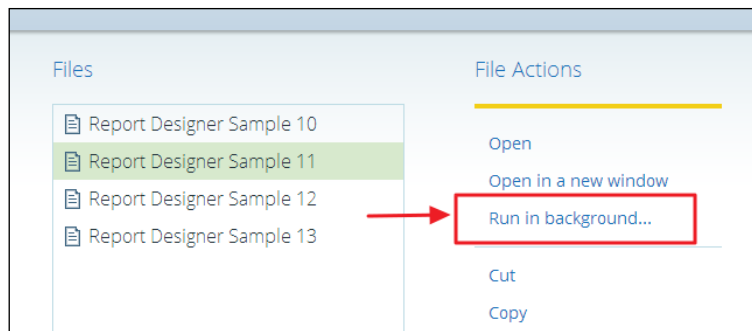
To get ready for this recipe, the following conditions must be met:

1. The Pentaho user must be successfully logged in to the Pentaho User Console.
2. The user role must have the **Schedule Content** operational permission assigned.
3. The user must be in the **Browse Files** perspective.

## How to do it...

Perform the following steps to filter through a set of existing schedules:

1. From the **Browse Files** perspective, navigate the Pentaho solution looking for the report that you would like to execute as a background task. As an example, go to `/Public/Pentaho BA Cookbook/Pentaho Reporting` and look for `Pentaho Report Sample 11`.
2. Click over the report you have identified. Look at the **File Actions** menu and click on **Run in background...** (see the following screenshot for details):



3. The **New Schedule** dialog box wizard appears. Choose the name of the generated output file and the location where the file will be stored as output from the system. Click on the **Next** button.
4. In the new dialog, if the report requires a set of input parameters, provide the report that you would like to execute to the parameters' values. This set of report parameters is the same that the report will require in case it needs to be started manually.
5. Click on **OK** to confirm the definition of the background task.

### How it works...

Running a task in the background is easy and could be useful in many situations. Imagine we are going to print a long-running report with a lot of data in it. Running it in the background might be an interesting option so that we can either continue to work or take a break while the report is being executed.

The definition of a background task is related to the implicit definition of a schedule with a Run Once schedule type. No more; no less. Due to this, just as we did to define schedules, we can run anything accessible in the Pentaho solution that can be started by a user in the background.

First, we look for the content we want to start with by going through the solution. As soon as we find the content we want to run, we click over it and select **Run in background...** from the **File Actions** menu. After we have selected this menu item entry, the **New Schedule** wizard is started, and the first dialog asks us for a name and the location of the content that will be provided upon the execution of the scheduled content.

After we have clicked on the **Next** button, if the report has a set of input parameters that must be filled, the wizard asks us for the input parameters. As we mentioned while discussing how a schedule is defined, the report might require a different set of input parameters depending on the conditions of the execution context (an example could be having a different date or a different period based on the moment in time when the report is being executed). In that case, we must think carefully about a different way to feed the report with different sets of input parameters.

When we have finished, we click on the **OK** button to start the background execution. A dialog box will inform us that the execution has been started and that the content is available at the specified location.

## Checking schedule execution

Periodically, we might want to check our schedules' execution. This recipe will discuss how we can do this.

### Getting ready

To get ready for this recipe, the following conditions must be met:

1. The Pentaho user must be successfully logged in to the Pentaho User Console.
2. The user role must have the **Schedule Content** operational permission assigned.

### How to do it...

Perform the following steps to check the schedules for execution:

1. Go to the **Schedule** perspective.
2. The schedule entries list appears; it provides the ability to check for the schedules' execution status.

### How it works...

We can check for schedule execution by going to the **Schedule** perspective and looking at the schedule entries list. The schedule entries list has a column named `Status`, which is useful if we want to understand what happened.

You might be interested to know the entire set of possible statuses our schedule could be in. To satisfy our need for this knowledge, we could take a look at the **Schedule State** drop-down list in the **Filter Schedule...** dialog box. Here, we will find the following values: **NORMAL, PAUSED, COMPLETE, BLOCKED, ERROR, and UNKNOWN.**

On the other hand, according to the documentation, the only known possible states a schedule could be in are **NORMAL** and **PAUSED**. What we mentioned is the sort of thing that could create a misunderstanding, but we must be aware of this whenever we are filtering our schedules, for example the following:

- ▶ A schedule is in the **NORMAL** state whenever the execution is completed successfully, or the execution never started
- ▶ A schedule is in the **PAUSED** state whenever the execution has been paused by us

Remember that if the currently connected user is not a member of the administrator role, they can only check their own schedules. Only the administrator can check everyone else's schedules.

## Stopping running executions

We might be interested in stopping a schedule that is currently being executed on our server. This recipe will show us how we can do this.

### Getting ready

To get ready for this recipe, the following conditions must be met:

1. The Pentaho user must be successfully logged in to the Pentaho User Console.
2. The user role must have the **Schedule Content** operational permission assigned.

### How to do it...

Perform the following steps to stop a schedule that is currently being executed on the server:

1. Go to the **Schedule** perspective and go through the list of schedule entries, looking for the schedule entry for which you want to stop the execution.
2. Select the schedule entry to stop, and click on the **Stop Scheduled Task** icon button located in the top-right corner of the schedule entries list.
3. The schedule that was being executed is immediately stopped and set to the **PAUSED** state.

### How it works...

There will always be an opportunity to stop the schedule that is currently being executed. This is a very simple action to perform; we just need to go to the **Schedule** perspective and locate the process that is currently being executed and that we want to stop.

## Scheduling Content

A process that is being executed is visible as two different rows: the first represents the previous execution and the second, the row without any value in the **Next Run** field, represents the task that is currently being executed.

Interactive Rpt - Sample 4	Every at 16:08:03	/public/Pentaho BA Cookbook/interactive Reports/interactive Rpt - Sample 4	/home/admin	2014 Mar 28 16:08:03	-	admin	PAUSED
Interactive Rpt - Sample 4	Every weekday at 08:15:00	/public/Pentaho BA Cookbook/interactive Reports/interactive Rpt - Sample 4	/home/admin	2014 Mar 28 16:08:03	2014 Mar 28 08:15:00	admin	NORMAL

Select the row without values in **Next Run** and click on **Stop Scheduled Task**, and the execution will stop immediately.

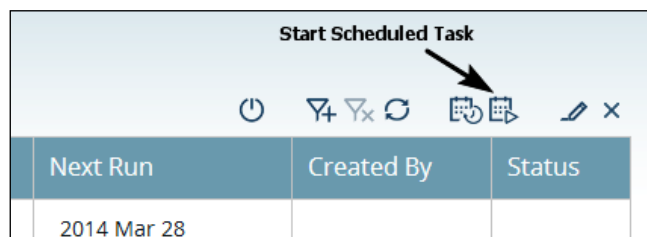
As we can see from the preceding screenshot, the status of our schedule is set to **PAUSED** to represent the idea that this schedule has currently been stopped and is waiting to be restarted.



Remember that if the currently connected user is not a member of the administrator role, they can only stop their own schedules from being executed. Only the administrator can stop everyone else's schedules from being executed.

## There's more...

We were really quick to stop a schedule that was being executed, but now we might also like, to start the task we stopped just as quickly. To restart a paused schedule process, we must go through the list of schedule items, looking for our paused process. As soon as we find it, we select it and click on the **Start Scheduled Task** icon button located in the top-right corner of the schedule entries list (see the following screenshot for details).



This new button will appear in place of the **Stop Scheduled Task** button as soon as the process is set to **PAUSED** and will be removed in favor of that previous button as soon as the process is restarted.

As soon as the scheduled task is restarted, the status for the scheduled process will change back to **NORMAL** and will maintain that value until the execution is complete.

## Preventing the creation of schedules by content

Sometimes, we might want to prevent the user from scheduling specific reports. This recipe will show us how we can do this.

### Getting ready

To get ready for this recipe, the following conditions must be met:

1. A Pentaho user must be successfully logged in to the Pentaho User Console.
2. The user must be a member or have the administrator role.
3. The user must be in the **Browse Files** perspective.

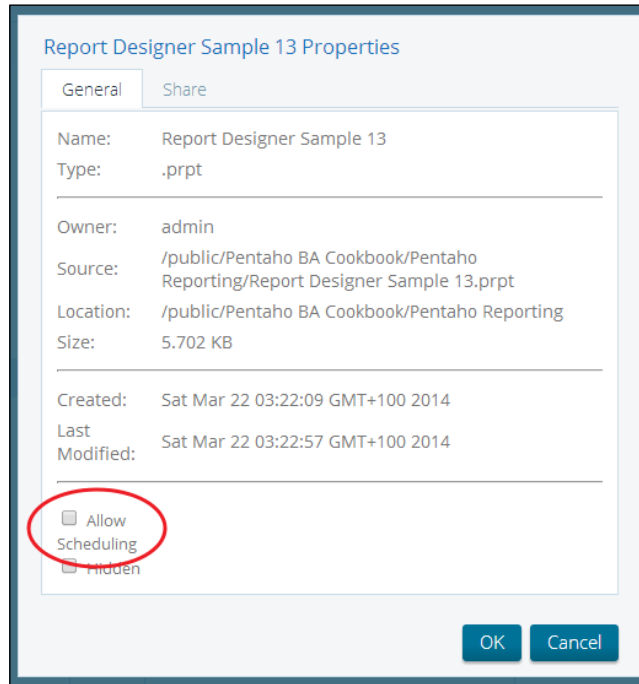
### How to do it...

Perform the following steps to prevent a user from scheduling a certain report:

1. From the **Browse Files** perspective, navigate the Pentaho solution looking for the report that you would like to execute as a background task. As an example, go to / Public/Pentaho BA Cookbook/Pentaho Reporting and look for Pentaho Report Sample 13.
2. Click on the report you identified. Take a look at the **File Actions** menu and click on the **Properties...** menu entry.
3. The **Pentaho Report Sample 13 Properties** dialog box will appear.



4. Click on the **Allow Scheduling** checkbox in the bottom-left corner of **Properties** and remove the checkbox as shown in following screenshot:



5. Click on the **OK** button and close the **Properties** dialog box.

## How it works...

Preventing the creation of schedules for certain content types might be useful in certain situations. To define the reports that must be excluded from a possible intention to schedule, we must start from the **Browse Files** perspective and look for the report that we want to apply this setting to.

Let's suppose we have identified a report that we need to apply this setting to. We click on the report we identified. We then take a look at the **File Actions** menu and click on the **Properties...** menu entry. The report **Properties** dialog box will open. To prevent the user from scheduling this report, we must uncheck the **Allow Scheduling** checkbox in the bottom-left corner of the **Properties** dialog and remove the checkbox.

As soon as the setting has been applied, we click on the **OK** button to confirm the changes. We can immediately verify that the setting has been applied correctly. To do this, we can simply verify that, after selecting the report, the **Schedule...** entry in the **File Actions...** menu on the right-hand side of the browser area will never appear.

## Preventing the creation of schedules by setting blackout time intervals

Now you have seen how to prevent the user from scheduling specific reports, we are now going to explain how to prevent the user from defining schedules for certain time frames. This recipe will show how you can define a blackout time interval.

### Getting ready

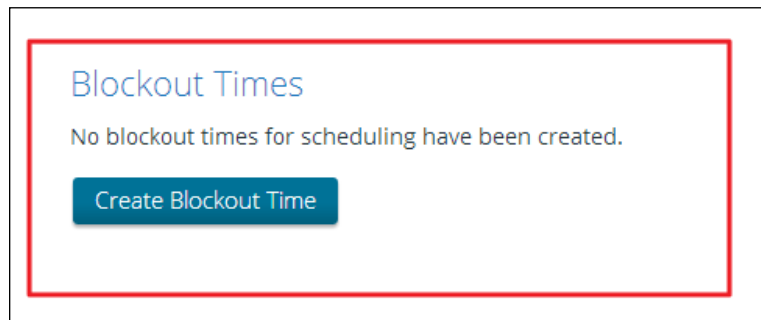
To get ready for this recipe, the following conditions must be met:

1. The Pentaho user must be successfully logged in to the Pentaho User Console.
2. The user must be a member or have the administrator role.

### How to do it...

Perform the following steps to prevent a user from scheduling a report during certain time intervals:

1. Go to the **Schedule** perspective and go through the list of schedule entries, looking for the schedule entry for which you want to stop the execution.
2. Go down to the bottom of the schedule entries list and locate the **Create Blockout Time** button in the bottom-left corner of the browser area (see the following screenshot for details).



The **New Schedule Blockout Time** dialog box appears.

3. Set the blackout time parameters and click on **OK** to confirm.
4. The dialog box closes, and the blackout time entries list will be visible in the **Blockout Times** table at the bottom of the schedule entries list table.

## How it works...

Setting blockout times means creating a set of time frames where the user will be prevented from creating a schedule. This is usually an activity taken up by the system administrator, and one of the reasons for this need could be to prevent a system overhead at a certain period during the day or night. We can either define the blockout times to use just one time or a recurring set of time frames, one for each day of the week, of the month or year.

To create a blockout time, we must go in the **Schedule** perspective, move down to the schedule entries list, and locate the **Create Blockout Time** button in the bottom-left corner of the browser area. The **New Schedule Blockout Time** dialog box will appear; it will contain all the configuration details to define a specific blockout time (see following screenshot for details).

The screenshot shows a dialog box titled "New Schedule Blockout Time". It features a "Recurrence" dropdown menu currently set to "Run Once". Below this is the "Start Time" section, which includes three input fields for hours (01), minutes (00), and AM/PM (AM), followed by a time zone dropdown menu set to "Central European Summer Time (UTC+0100)". The "Ends" section has two radio buttons: "Duration" (selected) and "End Time". Under "Duration", there are three input fields for days (0), hours (0), and minutes (0). The "Start Date" section has a text input field containing "2014-03-28" and a "\*" icon. At the bottom right, there are "OK" and "Cancel" buttons.

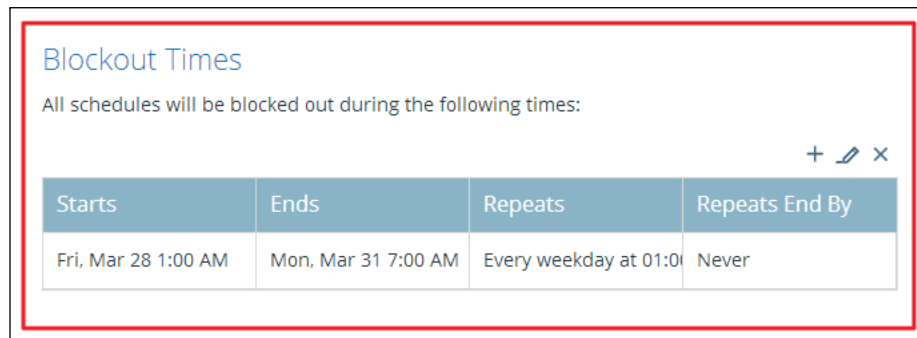
The **New Schedule Blockout Time** dialog box is similar to the dialog box used to define a new schedule. First, we are asked to set **Recurrence** for this blockout time interval. The possible values for this drop-down list are **Run Once**, **Daily**, **Weekly**, **Monthly**, and **Yearly**. Just like the **Schedule** dialog, whenever we choose a recurrence from the drop-down list, the fields in the dialog change and a new set of fields, different based on the option chosen, appear in the dialog. The fields that appear are the same fields we described for the **Schedule** dialog box in the first recipe.

After we chose the recurrence, we set the start time, which is expressed with reference to the time zone. The referring time zone must be specified; it needs to be picked up from the drop-down list to the right of the **Start Time** field set. Then, as soon as the start time has been set, we need to set the end time (in case we chose **Run Once** from the **Recurrence** drop-down list) or the recurrence pattern (for any other choice).

Finally, we need to set the duration as follows:

- ▶ If we chose **Run Once** from the **Recurrence** drop-down list, the last thing we need to set is the start date stating when this blackout time will apply and to which day of the year
- ▶ If we chose any other option from the **Recurrence** drop-down list, the last thing we need to set is the range of recurrence stating when this blackout time will apply, to which days, and until when

After we have set all of this information, we need to click on the **OK** button to confirm our settings. The dialog box closes and the blackout time entries list will be visible in the **Blockout Times** table at the bottom of the schedule entries list table.



Starts	Ends	Repeats	Repeats End By
Fri, Mar 28 1:00 AM	Mon, Mar 31 7:00 AM	Every weekday at 01:00	Never

Remember that we can always add new blackout times and update or delete the existing ones if we need to. We can do this using the three icon buttons located in the top-right corner of the **Blockout Times** table.



# 10

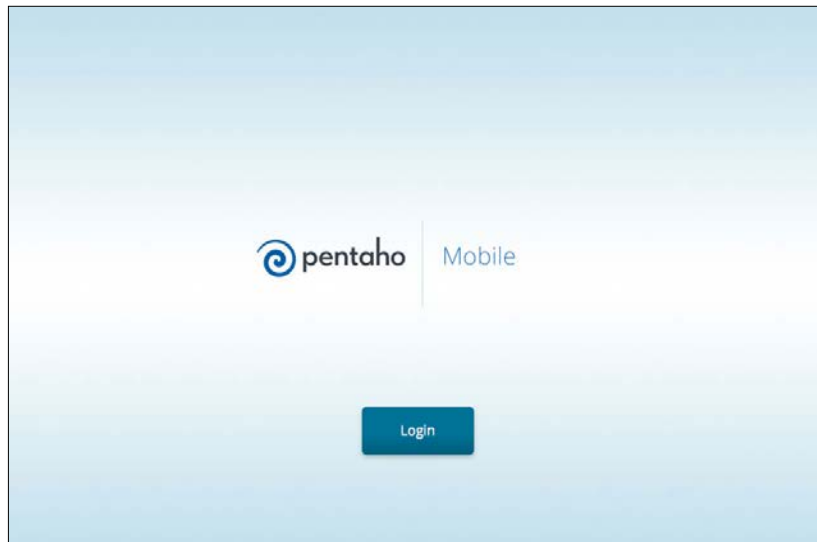
## Working with Pentaho Mobile BI

In this chapter, we will cover the following recipes:

- ▶ Accessing BA server from a mobile device
- ▶ Accessing folders and files
- ▶ Adding files to favorites
- ▶ Changing the default startup screen

## Introduction

We've always talked about using the Pentaho platform from a mobile device, trying to understand what there really is about it. On the Internet, there are some videos on it, but nothing can give you a clear idea of what it is and what can we do with it. We are proud to talk about it (maybe this is the first book that touches this topic), and we hope to clear any doubts regarding this platform.



Pentaho Mobile is a web app available (see the previous screenshot for the web application's main screen) only with the Enterprise Edition Version of Pentaho, to let iPad users (and only the users on that device) have a wonderful experience with Pentaho on their mobile device. At the time of writing this book, no other mobile platform or devices were considered. It lets us interact with the Pentaho system more or less in the same way as we do with Pentaho User Console. These recipes show what we can do with Pentaho Mobile and what we cannot do in a clear and detailed way to help understand if accessing Pentaho from a mobile platform could be helpful for our users.

In the next recipes and, only for this chapter, because we are on a mobile device, we will talk about touching (touch) instead of clicking as the action that activates something in the application. With this term, *touch*, we refer to the user's finger gesture instead of the normal mouse click. Different environments have different ways to interact!

The recipes in this chapter are based on the assumption that you have iPad device available to try each recipe and that you are able to successfully log in to Pentaho Mobile.

In case we want to use demo users, remember that we can use the following logins to access our system:

- ▶ **admin/password:** This is the new Pentaho demo administrator after the famous user, `joe` (the Pentaho recognized administrator until Pentaho 4.8), was dismissed in this new version.
- ▶ **suzy/password:** This is another simple user we can use to access the system. Because `suzy` is not a member of the administrator role, it is useful to see the difference in case a user who is not an administrator tries to use the system.

## Accessing BA server from a mobile device

Accessing Pentaho Mobile is as easy as accessing it from a Pentaho User Console. Just open our iPad browser (either Safari or Chrome) and point your browser to the Pentaho server. This recipe shows the basics of accessing and logging in to Pentaho from an iPad device through Pentaho Mobile. Remember that this recipe makes use of Pentaho Mobile, a web app that is available only for iPad and only in the EE Version of Pentaho.

### Getting ready

To get ready with this recipe, the only thing we need is an iPad to connect to our Pentaho system.

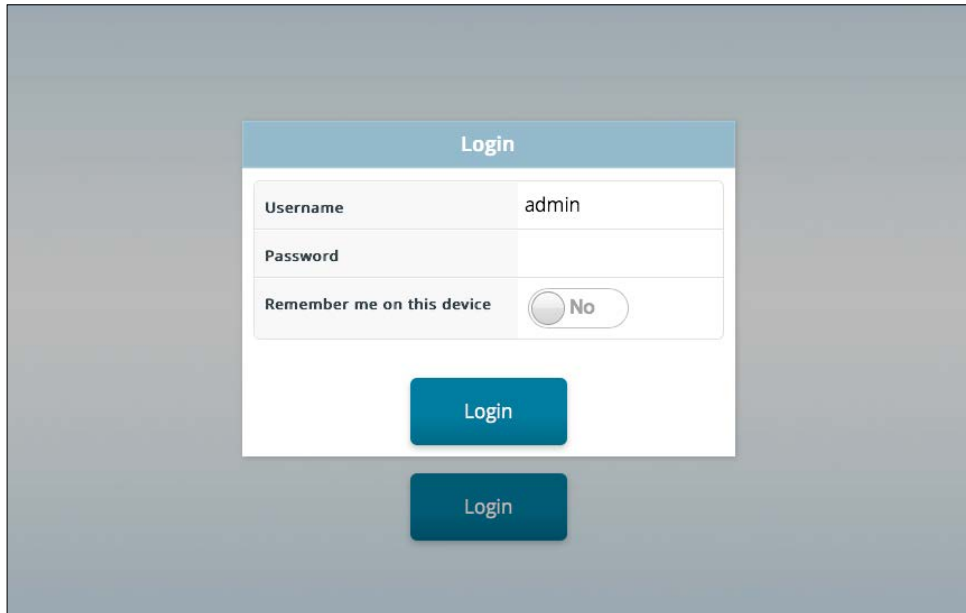
### How to do it...

The following steps detail how simply we access our Pentaho Mobile application:

1. To connect to Pentaho Mobile, open either Safari or Chrome on the iPad device. As soon as the browser window is ready, type the complete URL to the Pentaho server in the following format:  
`http://<ip_address>:<port>/pentaho`
2. Pentaho immediately detects that we are connecting from an iPad device, and the Pentaho Mobile login screen appears.



3. Touch the **Login** button; the login dialog box appears as shown in the following screenshot. Enter your login credentials and press **Login**.



4. The **Login** dialog box closes and we will be taken to Pentaho Mobile's home page.

### How it works...

Pentaho Mobile has a slightly different look and feel with respect to Pentaho User Console in order to facilitate a mobile user's experience.

The following screenshot shows the landing page we get after we have successfully logged in to Pentaho Mobile. To the left-hand side of the Pentaho Mobile's home page, we have the following set of buttons:

- ▶ **Browse Files:** This lets us start our navigation in the Pentaho Solution Repository.
- ▶ **Create New Content:** This lets us start the Pentaho Analyzer to create a new Analyser report from the mobile device.



Analysers report content is the only kind of content we can create from our iPad. Dashboards and interactive reports can be created only from the Pentaho User Console.

- ▶ **Startup Screen:** This lets us change what we display as the default startup screen as soon as we log in to Pentaho Mobile.

- ▶ **Settings:** This changes the configuration settings for our Pentaho Mobile application.



To the right-hand side of the button list (see the previous screenshot for details), we have three list boxes that display the **Recent** files we opened so far, the **Favorites** files, and the set of **Open Files**. The **Open Files** list box is more or less the same as the **Opened** perspective in Pentaho User Console—it collects all of the opened content in one single place for easy access.

Look at the upper-right corner of Pentaho Mobile's user interface (see the previous screenshot for details); we have two icons:

- ▶ The folder icon gives access, from a different path, to the Pentaho Solution's folders
- ▶ The gear icon opens the **Settings** dialog box

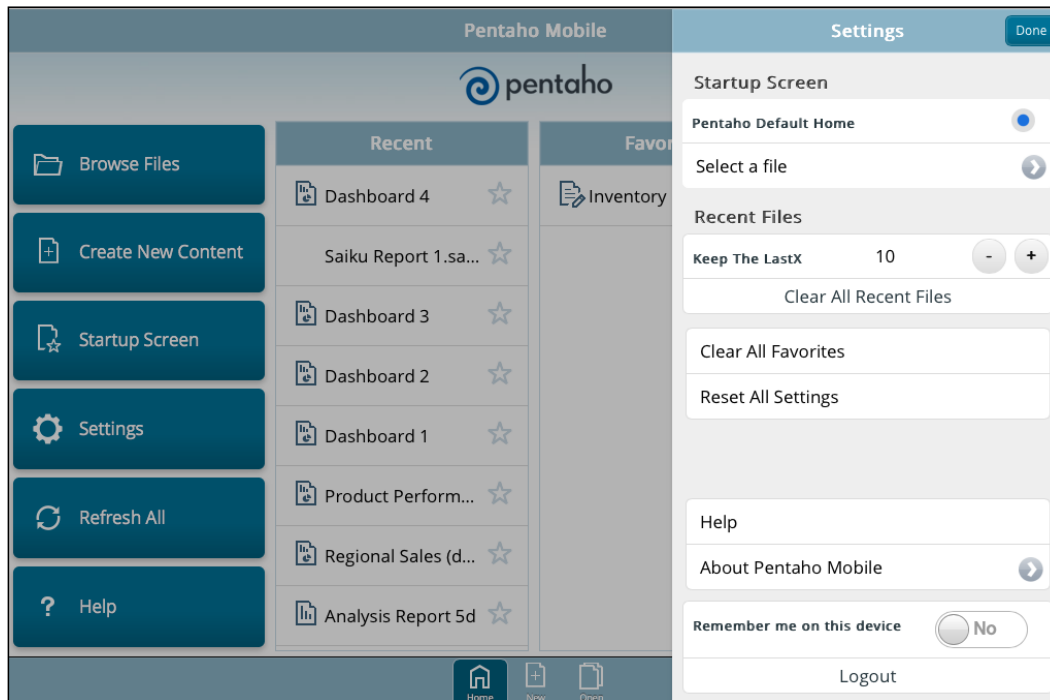
## There's more...

Now, let's see which settings we can either set or change from the mobile user interface by going to the **Settings** options.

### Changing the Settings configuration in Pentaho Mobile

We can easily access the **Settings** dialog box either by pressing the **Settings** button in the left-hand side area of the Pentaho Mobile's home page or by pressing the gear icon in the upper-right corner of Pentaho Mobile.

The **Settings** dialog box allows us to easily change the following configuration items (see the following screenshot for details):



- ▶ We can set **Startup Screen** by changing the referenced landing home page for our Pentaho Mobile application.
- ▶ In the **Recent Files** section of the **Settings** dialog, we can set the maximum number of items allowable in the **Recent Files** list. The default setting's value is 10, but we can alter this value by pressing the related icon buttons. Another button situated immediately below **Recent Files**, lets us easily empty the **Recent Files** list box.
- ▶ The next two buttons let us clear the **Favorites** items' list (**Clear All Favorites**) and reset the settings to the default values (**Reset All Settings**).
- ▶ Finally, we have a button to take us to a help guide and the application's **Logout** button.

## See also

- ▶ Look at the *Accessing folders and files* recipe to obtain details about how to browse the Pentaho Solution and start new content
- ▶ In the *Changing the default startup Screen* recipe, we will find details about how to change the default Pentaho Mobile session startup screen

## Accessing folders and files

From our Pentaho Mobile, we can easily access and navigate the Pentaho Solution folders. This recipe will show how we can navigate the Pentaho Solution folders and start our content on the mobile device. Remember that this recipe makes use of Pentaho Mobile, a web app available only for iPad and only in the EE Version of Pentaho.

### How to do it...

The following steps detail how simply we can access the Pentaho Solution folders and start an existing BI content:

1. From the Pentaho Mobile home page, either touch on the **Browse Files** button located on the left-hand side of page, or touch on the **Folder** icon button located in the upper-right side of the home page.
2. The **Browse Files** dialog opens to the right of the Pentaho Mobile user interface as shown in the following screenshot. Navigate the solution to the folder containing the content we want to start.



3. As soon as we get to the content to start, touch on the content's icon to launch it. The content will be displayed in the entire Pentaho Mobile user interface screen.

## How it works...

Accessing Pentaho objects from the Pentaho Mobile application is really intuitive. After you have successfully logged in, open the **Browse Files** dialog and navigate freely through the Pentaho Solution folder's structure to get to your content. To start the content, just touch the content icon and the report or the dashboard will display on your iPad.



As we can see, at the time of writing this book, we cannot do any administrative tasks (share content, move content, schedule, and other tasks) from the Pentaho Mobile application. We can only navigate to the content, get it, and start it.

## There's more...

As soon as we have some content items open, they are shown in the **Opened** list box. However, we would like to close them and free unused memory resources. Let's see how to do this.

### Closing opened content

Pentaho Mobile continuously monitors the resource usage of our iPad and warns as soon as we have a lot of items open. As soon as we have a lot of opened items, a warning dialog box informs you about this, and it is a good opportunity to close some unused (and eventually forget the opened) content items.

To do this, go to Pentaho Mobile's home page, look for items to close, and touch on the rounded x icon to the right of the content item's label (see the following screenshot for details).



The content item will immediately close.

## Adding files to favorites

As we saw in Pentaho User Console, even in the Pentaho Mobile application, we can set our favorites and start accessing content from the favorites list. This recipe will show how we can do this. Remember that this recipe makes use of Pentaho Mobile, a web app available only for iPad and only in the EE Version of Pentaho.

### How to do it...

The following steps detail how simply we can make a content item a favorite:

1. From the Pentaho Mobile's home page, either touch on the **Browse Files** button located on the left-hand side of the page or touch on the **Folder** icon button located in the upper-right side of the home page.
2. The **Browse Files** dialog opens to the right of the Pentaho Mobile user interface. Navigate the solution to the folder containing the content we want as a favorite.
3. Touch on the star located to the right-hand side of the content item's label to mark that item a favorite.

## How it works...

Usually, it would be useful to define some Pentaho objects as favorites. Favorite items help the user to quickly find the report or dashboard to start with. After we have successfully logged in, open the **Browse Files** dialog and navigate freely through the Pentaho Solution folders' structure to get to your content. To mark the content a favorite, just touch the star in the right-hand side of the content label and our report or dashboard will be marked as favorite (see the following screenshot for details).



The favorite status of an item is identified by the following elements:

- ▶ The content item's star located to the right-hand side of the item's label becomes bold on the boundary to put in evidence that the content has been marked as a favorite
- ▶ The content will appear in the **Favorites** list box on the Pentaho Mobile home page

## There's more...

What should we do if we want to remove the favorite status from our content items? Let's see how we can do this.

### Removing an item from the Favorites items list

To remove an item from the Favorites list, we can follow two different approaches:

- ▶ Go to the **Favorites** items list on the Pentaho Mobile home page. Look for the item we want to un-favorite and touch on the star icon with the bold boundaries located on the right-hand side of the content item's label. The content item will be immediately removed from the **Favorites** items list.
- ▶ Navigate to the Pentaho Solution's folders to the location containing the item we want to un-favorite and touch on the star icon with the bold boundaries located to the right-hand side of the content item's label. The content item will be immediately removed from the **Favorites** items list.

## See also

- ▶ Take a look at the *Accessing folders and files* recipe in case you want to review how to access content in the Pentaho Solution to mark it as a favorite.

## Changing the default startup screen

Imagine that we want to change the default startup screen with a specific content item we have in our Pentaho Solution. After the new startup screen has been set, after the login, the user will be able to immediately access this new content item opened as the startup screen for Pentaho Mobile instead of the default home page. It would be fine to let our CEO immediately have in front of them the company's main KPI dashboard and immediately react to it. This recipe will show you how to make a specific content item the default startup screen in Pentaho Mobile. Remember that this recipe makes use of Pentaho Mobile, a web app available only for iPad and only in the EE Version of Pentaho.

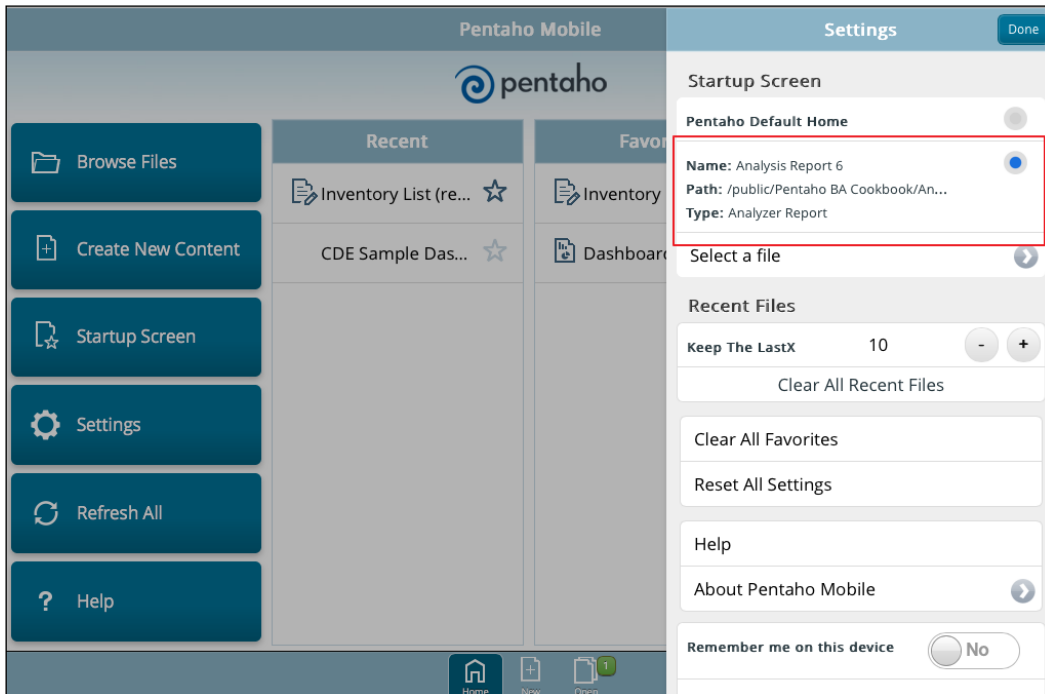
## How to do it...

The following steps detail how simply we can define a new startup screen with an existing BI content:

1. From the Pentaho Mobile home page, touch on the **Startup Screen** button located on the left-hand side of the home page.
2. The **Browse Files** dialog opens to the right of the Pentaho Mobile user interface. Navigate the solution to the folder containing the content we want to use.



3. Touch the content item we want to show as the default startup screen. The **Browse Files** dialog box immediately closes and the **Settings** dialog box opens. A reference to the new, selected item is shown as the default **Startup Screen** content item (see the following screenshot for details):



4. Touch outside the **Settings** dialog to close this dialog.

## How it works...

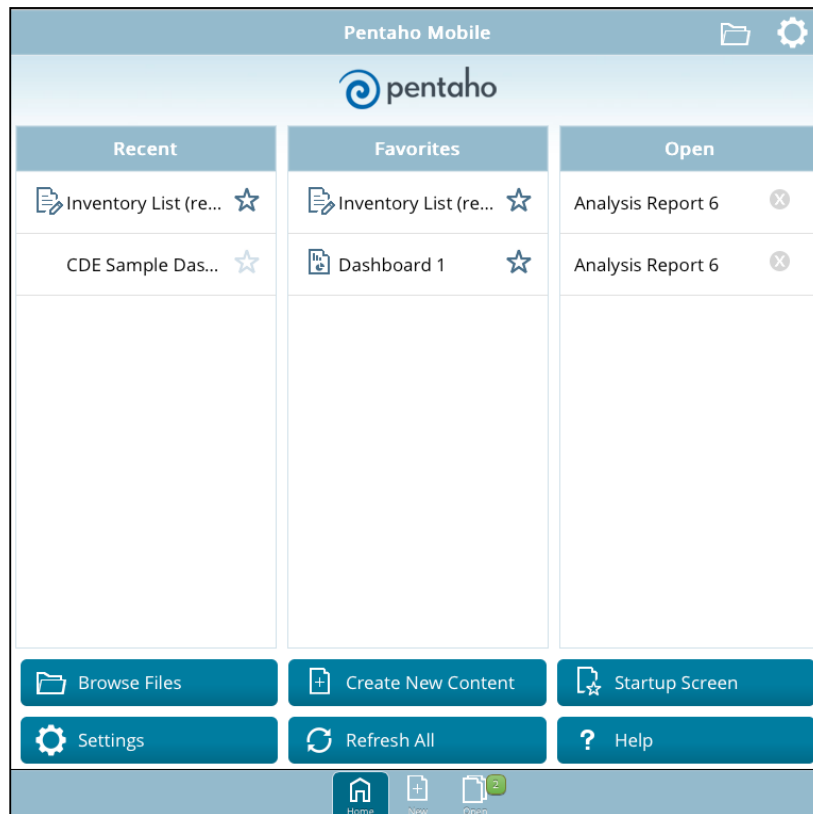
Changing the startup screen could be interesting to give your user access to important content any time immediately after a successful login. From the Pentaho Mobile's home page, touch on the **Startup Screen** button located on the left-hand side of the home page and open the **Browse Files** dialog. Navigate the solution to the folder containing the content we want and then touch the content item to show as the default startup screen. The **Browse Files** dialog box immediately closes and the **Settings** dialog box opens. The new selected item is shown as the default startup screen content item, referenced by **Name**, and the complete path to the Pentaho Solution folder is seen. We can change the startup screen at any time, and we can also reset it to the default Pentaho Mobile home page by touching on the **Pentaho Default Home** radio button.

## There's more...

We have always showed pictures from Pentaho Mobile in landscape orientation, but the user interface has a responsive behavior, showing things organized differently depending on the orientation of the device.

### Pentaho Mobile's responsive behavior

We always show pictures of Pentaho Mobile with a landscape orientation, but Pentaho Mobile has a responsive layout and changes the display of some of the items in the page we are looking at depending on the device's orientation. The following screenshot gives an idea about displaying a dashboard on Pentaho Mobile in portrait orientation:



If we look at the home page with a device in the portrait mode, the **Recents**, **Favorites**, and **Opened** lists covers the available page's width, equally divided by each list; and all of the buttons we always saw on the left side of the user interface are now relocated to the bottom, below the three lists we talked about so far. This is another interesting layout; it is up to our taste or viewing needs to decide which of the two could be the best option for us.



# 11

## Customizing Pentaho BA to Meet Your Business Needs

In this chapter, we will cover the following topics:

- ▶ Adding a company's logo to the Pentaho User Console login page
- ▶ Using themes to customize Pentaho User Console
- ▶ Adding new languages to Pentaho User Console
- ▶ Checking the Pentaho BA Server logs from inside Pentaho User Console
- ▶ Easily managing content in Pentaho Solution

### Introduction

The goal of this final chapter is to gain some insight on practical topics that can be useful to you reader. We will start by customizing Pentaho to align it with the company's look and feel, and then we will move on to dealing with interesting topics, such as how to easily manage BI content updates in Pentaho Solution. As we already know, Pentaho Solution is now a JCR standard repository and objects' files are no longer directly accessible. This can be worrisome because the first problem could be how we can manage updates to BI files in Pentaho Solution as soon as we get updates to the same files from our development team. Another problem could be in migrating from an older Pentaho server version to a newer version: how can we easily load the BI objects' files from the old Pentaho server version to the new one? We will see how to manage these problems easily and without any pain.

## Adding a company's logo to the Pentaho User Console login page

Immediately after we install Pentaho BA Server, we are almost sure that we will have to customize certain parts of the user interface by following the company's standard guidelines. The first request we can expect from the customer is to change the default Pentaho logo to our company's logo. This recipe will show how we can easily do this.

### Getting ready

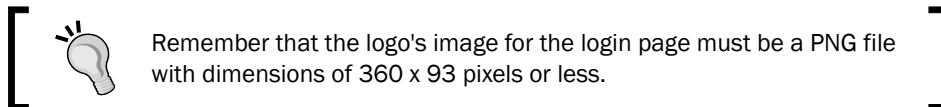
For this recipe, the following conditions must be met:

- ▶ We must have a basic understanding of CSS, HTML, and JavaScript
- ▶ We must have direct access to the Pentaho application files on the server
- ▶ We must have the filesystem's grants to update the Pentaho application files on the server

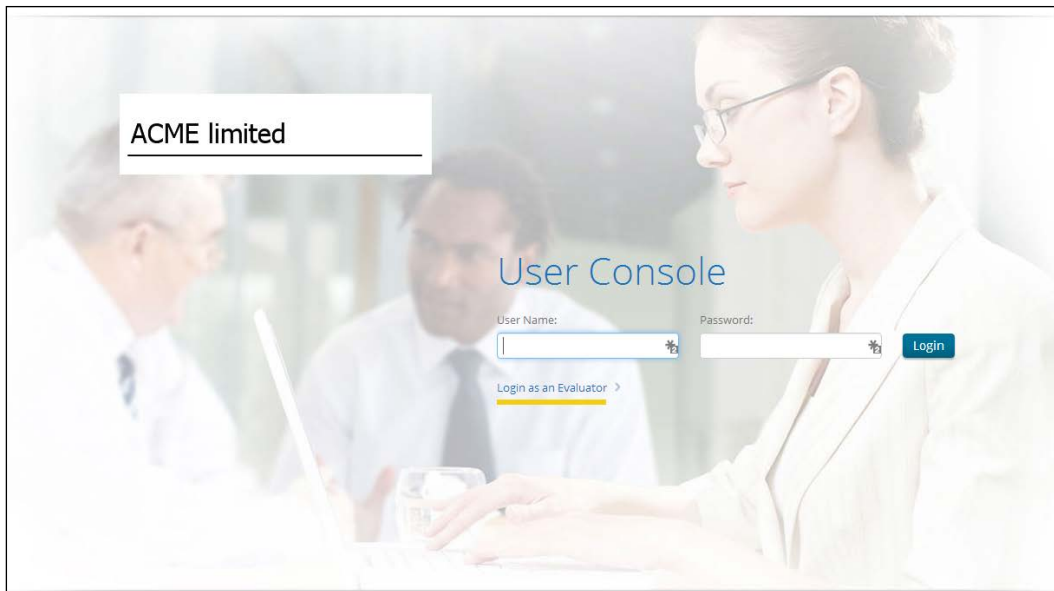
### How to do it...

The following steps detail how to change the logo on our Pentaho BA Server login page:

1. Go to the filesystem and locate the following directory:  
`<biserver_home>/pentaho-solution/system/common-ui/resources/themes/images`
2. Look for the `puc-login-logo.png` file and make a copy. As an example, we provided a sample logo image named `sample-login-logo.png` with the right dimensions in the `ch11/resources` directory.
3. Take the new logo's image file and rename it to the same name.



4. Change the new logo file's name to that name.
5. Reload the login page. The new logo is displayed on the login page as shown in the following screenshot:



### How it works...

Changing a logo by substituting the default logo with the company's logo is always one of the first things we face as soon as we adopt a new application.

The login page logo is a part of the current theme and has a fixed name, so it is very simple to change it. It would be good to have our new logo image in the PNG format because that is the format of the original logo. Then, get the new logo and rename it as `puc-login-logo.png`, the same as the current Pentaho login page's logo name. Renaming the file is necessary to make the change as seamless as possible without requiring a change to the page source.

Locate the original login page logo in the theme directory in `<baserver_home>/pentaho-solution/system/common-ui/resources/themes/images` and rename the original image's file to prevent breaking things in case of any problems (this is always a good thing to do). Then, copy your new logo file, reload the login page by pressing `Ctrl + R` on your browser page, and you are done.



In case our new logo is not immediately visible, verify whether the name given to the new file follows the name specified in the instructions given in this recipe. If the name is correct, remember to clear your browser cache, then close and restart the browser and reload the Pentaho login page.

## There's more...

In case we don't have a logo in the PNG format and we want to make the simple changes illustrated in this recipe, we must change the login page's sources to accomplish our needs. Let's go through the details of how we can change the login page's logo with another logo of a completely different format (JPEG, GIF, and so on).

The change must be applied directly to CSS theme files. Supposing that our actual theme is Crystal, all files for this theme definition are located at `<baserver_home>/pentaho-solution/system/common-ui/resources/themes/crystal`. Locate the `globalCrystal.css` file and open it with your favorite editor. The logo is basically defined as a background of a CSS ID definition.

Locate the following code fragment in the CSS file:

```
#login-logo {  
    background: url(../images/puc-login-logo.png) no-repeat left;  
    width: 0;  
    height: 93px;  
    left: 750px;  
    top: 90px;  
    position: relative;  
    opacity: 0.5;  
    -moz-transition: width 600ms linear, left 600ms linear, opacity 1s  
linear;  
    -webkit-transition: width 600ms linear, left 600ms linear, opacity  
1s linear;  
    transition: width 600ms linear, left 600ms linear, opacity 1s  
linear;  
}
```

The code fragment that is in bold is the definition that contains the link to the logo image's file. The only thing we must do to use our new logo is to copy it in the `<baserver_home>/pentaho-solution/system/common-ui/resources/themes/images` directory and make the necessary changes to this code fragment to link our new logo file.

## See also

- ▶ A good reference for CSS, useful for beginners to learn the very basics of manipulating the CSS file, can be found on the W3C website at the following URL:

<http://www.w3schools.com/css/default.asp>

---

## Using themes to customize Pentaho User Console

The first recipe illustrated a very simple thing: how to change the logo on the PUC login page with our own custom logo. However, Pentaho BA Server uses themes that can be used to completely reconfigure the Pentaho User Console layout to set it according to our requirements. This recipe will show how we can easily update an existing theme and design a new one.

### Getting ready

For this recipe, the following conditions must be met:

- ▶ We must have a basic understanding of CSS, HTML, and JavaScript
- ▶ We must have direct access to the Pentaho application files on the server
- ▶ We must have the filesystem's grants to update the Pentaho application files on the server

### How to do it...

The following steps detail how to use themes to change the look and feel of our Pentaho BA Server installation:

1. Go to the filesystem and locate the following directory:  
`<baserver_home>/pentaho-solution/system/common-ui/resources/themes`
2. The directory specified in the previous step contains a set of global themes available for usage in our Pentaho installation.
3. Look for a directory named `crystal`. Copy and paste this directory below so that we create a copy of the original directory.
4. Change the name of the copied directory to `cookbook`. We created a new global theme entry, starting from the existing `crystal` theme.
5. In the `<baserver_home>/pentaho-solution/system/common-ui/resources/themes/cookbook` directory, look for the `globalCrystal.css` file and rename it to `globalCookbook.css`.



6. Locate the `<baserver_home>/pentaho-solution/system/common-ui` directory and open the `template.xml` file using your favorite editor. Locate the following fragment:

```
<crystal display-name="Crystal" system="true">
  <file>globalCrystal.css</file>
  <file>bootstrap/css/bootstrap-namespaced.css</file>
</crystal>
```

7. Copy and paste it below and then change it according to the following fragment:

```
<cookbook display-name="Cookbook" system="true">
  <file>globalCookbook.css</file>
  <file>bootstrap/css/bootstrap-namespaced.css</file>
</cookbook>
```

8. Save the changes and close the file.

9. Go to the filesystem and locate the following directory:

```
<baserver_home>/tomcat/webapps/pentaho/mantle/themes
```

10. The directory specified in the previous step contains a set of the local themes available for usage in our Pentaho installation.

11. Look for a directory named `crystal`. Copy and paste this directory below so that we create a copy of the original directory.

12. Change the name of the copied directory to `cookbook`. We created a new global theme entry, starting from the existing **Crystal** theme.

13. In the `<baserver_home>/tomcat/webapps/pentaho/mantle/themes` directory, look for the `mantleCrystal.css` file and rename it to `mantleCookbook.css`.

14. In the `<baserver_home>/tomcat/webapps/pentaho/mantle` directory, open the `template.xml` file using your favorite editor. Locate the following fragment:

```
<crystal>
  <file>mantleCrystal.css</file>
</crystal>
```

Copy and paste it below then change it according to the fragment below.

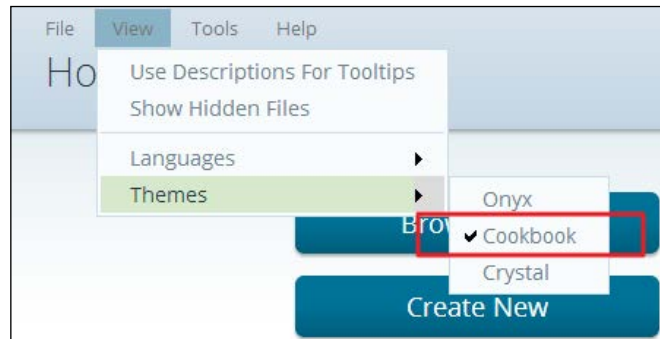
```
<cookbook>
  <file>mantleCookbook.css</file>
</cookbook>
```

15. Look for the `pentaho.xml` file and open it using your favorite editor. Locate the following `.xml` fragment at the very end of the file:

```
<default-theme>crystal</default-theme>
```

16. Change the `.xml` file according to the following code:
 

```
<default-theme>cookbook</default-theme>
```
17. Restart your Pentaho BA Server to activate the new style sheet.
18. Log in to Pentaho User Console with a user who is a member of the **Administrator** role.
19. Navigate to **View | Themes** from the menu. Our new theme, **Cookbook**, is visible in the themes menu items' list. Click on the **Cookbook** entry to select the theme we just added, as shown in the following screenshot:



20. Now, let's make a very simple change to our new theme. Imagine that we want the big buttons on the left-hand side of the **Home** perspective to become highlighted in red each time a user moves over a button with the mouse pointer.
21. In the `<baserver_home>/pentaho-solution/system/common-ui/resources/themes/cookbook/bootstrap/css` directory, look for the `bootstrap.css` file. Open the file with your favorite editor.
22. Locate the following code fragment in the file:

```
.bootstrap .btn:hover{
  color: #FFF;
  text-decoration: none;
  background-color: #0085B0;
}
```

23. Change the CSS fragment as specified in the following code:

```
.bootstrap .btn:hover{
  color: #FFF;
  text-decoration: none;
  background-color: red;
}
```

24. Locate the following code fragment in the file:

```
.bootstrap .btn:hover,  
.bootstrap .btn:focus,  
.bootstrap .btn:active,  
.bootstrap .btn.active,  
.bootstrap .btn.disabled,  
.bootstrap .btn[disabled] {  
    color: #FFF;  
    background: #0085B0;  
    *background-color: #0085B0;  
}
```

Change the css fragment as specified below

```
.bootstrap .btn:hover,  
.bootstrap .btn:focus,  
.bootstrap .btn:active,  
.bootstrap .btn.active,  
.bootstrap .btn.disabled,  
.bootstrap .btn[disabled] {  
    color: #FFF;  
    background: red;  
    *background-color: red;  
}
```

25. Save the file and close it.

26. In the <baserver\_home>/pentaho-solution/system/common-ui/resources/themes/cookbook directory, look for the globalCookbook.css file. Open the file with your favorite editor.

27. Locate the following code fragment in the file:

```
.pentaho-button:hover {  
    -moz-box-shadow: inset 0 0 1px #FFF;  
    -webkit-box-shadow: inset 0 0 1px #FFF;  
    box-shadow: inset 0 0 1px #FFF;  
    background: #0085b0; /* Old browsers */  
}  
  
.IE .pentaho-button:hover {  
    background: #0085b0;  
}  
  
.IE .pentaho-button-hover {  
  
    background: #0085b0;  
}
```

28. Change the CSS fragment as specified in the following code:

```
.pentaho-button:hover {
  -moz-box-shadow: inset 0 0 1px #FFF;
  -webkit-box-shadow: inset 0 0 1px #FFF;
  box-shadow: inset 0 0 1px #FFF;
  background: red; /* Old browsers */
}

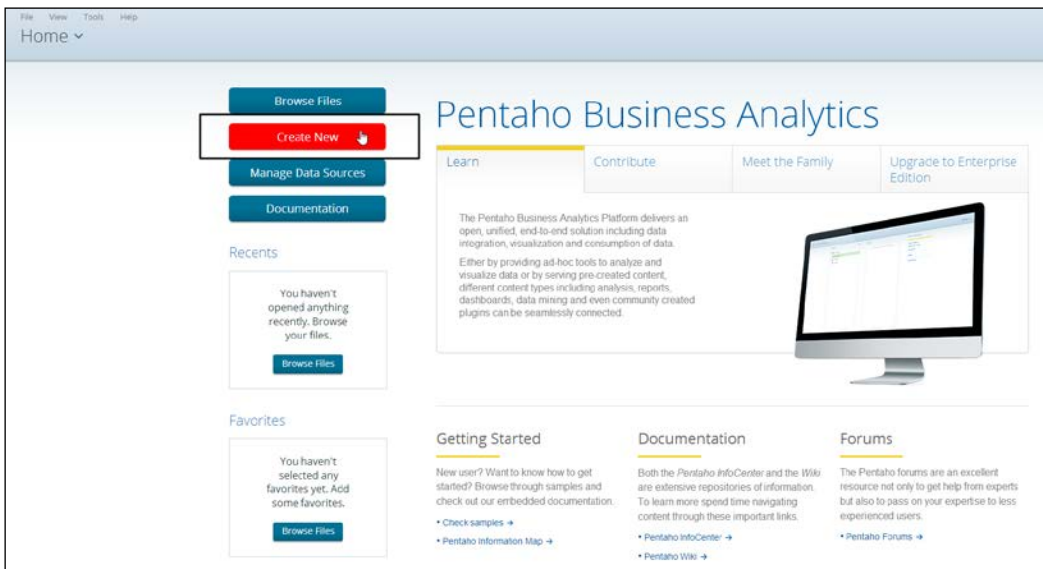
.IE .pentaho-button:hover {
  background: red;
}

.IE .pentaho-button-hover {
  background: red;
}
```

29. Save the file and close it.

30. Clear the browser cache. Close the browser and reopen it.

31. Log in again to Pentaho User Console with a user who is a member of the **Administrator** role. Now, as soon as we enter PUC, whenever we move the mouse over the buttons to the left-hand side of the PUC client area, we will see the buttons' background turn red (as desired), as shown in the following screenshot:



## How it works...

The previous recipe showed how to change the logo on the login page. This is a minor change, but it is related to a change in the theme's files. Themes in Pentaho organize the look and feel of the portal pages to accommodate changes in styles for different needs. Themes define a sort of pluggable mechanism to quickly change the look and feel of our UI elements, following particular styling needs. We can have more than one theme installed, but only one is selected as the active theme. Users logged in to Pentaho User Console can select the active theme.

We can have two different kinds of theme files called *System* and *Local*; any set is made up of the XML configuration files and the CSS files:

- ▶ System theme files influence all the graphical aspects of Pentaho User Console. These sets of files are mainly located in the `<baserver_home>/pentaho-solution/system/common-ui/resources/themes` directory.
- ▶ Local themes influence only a particular aspect of the UI, specifically related to a certain plugin.

In the case of this recipe, we will proceed with the definition of a new System theme for our Pentaho User Console. To start designing a new theme, it is always a good idea to start from an existing one and then modify that theme to obtain what is desired. For this reason, we started by making a copy of the `crystal` directory below `<baserver_home>/pentaho-solution/system/common-ui/resources/themes` and renaming this copy with the name of our new theme (in this case, `cookbook`).

Then, we must modify a file called `theme.xml` located in `<baserver_home>/pentaho-solution/system/common-ui`. Copy and paste a theme's fragment and rename the fragment elements to the name of the theme's directory (in our case, `cookbook`). This file is read by Pentaho on startup in every plugin and in the `pentaho.war` file's root-level folder, and contains the configuration of our theme. This indicates to the Pentaho server which files compose the theme and where these files are physically located in the filesystem. Every copy of the `theme.xml` file can contain the definition of multiple sets of themes.

Let's have a look at our copy of `theme.xml`:

```
<?xml version="1.0" encoding="UTF-8"?>
<themes root-folder="resources/themes">
  <onyx display-name="Onyx" system="true">
    <file>globalOnyx.css</file>
    <file>bootstrap/css/bootstrap-namespaced.css</file>
  </onyx>
  <crystal display-name="Crystal" system="true">
    <file>globalCrystal.css</file>
    <file>bootstrap/css/bootstrap-namespaced.css</file>
  </crystal>
```

```

<cookbook display-name="Cookbook" system="true">
  <file>globalCookbook.css</file>
  <file>bootstrap/css/bootstrap-namespaced.css</file>
</cookbook>
<slate display-name="Slate" system="true" hidden="true">
  <file>globalSlate.css</file>
</slate>
<native display-name="Native" system="true" hidden="true">
  <file>globalNative.css</file>
</native>
<sample1 system="true" hidden="true">
  <file>style1.css</file>
</sample1>
<replaceable system="true" hidden="true"> <!-- change this to false
to enable the replaceable theme -->
  <file>replaceable.css</file>
</replaceable>
</themes>

```

If we look at the file, we can see that it contains the path to the theme files in the filesystem directories (the `root-folder` attribute in the `themes` element) and an `.xml` fragment for each of the themes defined.

Any theme's `.xml` fragment is identified with an element named after the directory on the disk where the theme's file set is saved (in our case, `cookbook`):

```

<cookbook display-name="Cookbook" system="true">
  <file>globalCookbook.css</file>
  <file>bootstrap/css/bootstrap-namespaced.css</file>
</cookbook>

```

For any theme's fragment, we have to specify the following information:

- ▶ The name of the theme as displayed in the Pentaho User Console's **Themes** menu (under **View**; this is the `display-name` attribute)
- ▶ Whether it is a system theme or not (the `system` attribute)
- ▶ The set of CSS files that compose this particular theme (specified by the set of the `file` elements)
- ▶ We also have the possibility to set a theme as hidden by setting the `true` value to the related attribute, as we can see from the following code excerpt:

```

<sample1 system="true" hidden="true">
  <file>style1.css</file>
</sample1>

```

- ▶ If we set this attribute's value to `false`, the theme will not be visible under the Pentaho User Console's **Themes** menu (under **View**).

After this, we must do the same copy and paste operation in another location. Go to `<baserver_home>/tomcat/webapps/pentaho/mantle/themes` and look, because there we have another `crystal` directory that we must rename as `cookbook`. This is another small part of our system's theme. It has another set of CSS files related to certain UI aspects of Mantle, which is the UI application for Pentaho User Console. Then, under the `<baserver_home>/tomcat/webapps/pentaho/mantle` directory, open the `template.xml` file using your favorite editor:

```
<?xml version="1.0" encoding="UTF-8"?>
<themes root-folder="themes">
  <onyx>
    <file>mantleOnyx.css</file>
  </onyx>
  <slate>
    <file>mantleSlate.css</file>
  </slate>
  <crystal>
    <file>mantleCrystal.css</file>
  </crystal>
  <cookbook>
    <file>mantleCookbook.css</file>
  </cookbook>
</themes>
```

As we can see, this file is similar to the previous file even if it is with less attributes. Copy and paste a theme's fragment and rename the fragment elements with the name of the theme's directory (in our case, `cookbook`). It contains the definition of the Mantle CSS files. This last change is not mentioned in Pentaho's official documentation when describing the procedure to define a new theme, but it is extremely important.

To complete our new system theme definition, we need to consider another change in `pentaho.xml`. Here, we must set the reference to the default theme by setting the `default-theme` element as specified in the following code:

```
<default-theme>cookbook</default-theme>
```

After all these operations are completed, we can start modifying the theme's CSS files to get the desired look and feel for our new theme. Remember to do the following after completing all of the modifications:

- ▶ Restart your Pentaho server to make the changes visible
- ▶ Select the new theme from the **Themes** menu (under **View**)

Good luck with your new theme's definition and have fun!

## There's more...

To help us discover the internals of our web pages and identify the right CSS part to modify, we can use tools such as Firebug for Firefox and Chrome or the Chrome Developer Tools. Both tools are almost equivalent in terms of functionality and work in the same way, but in our case, we are more used to the Chrome Developer Tools; so, we just want to give you a brief introduction to this useful tool.

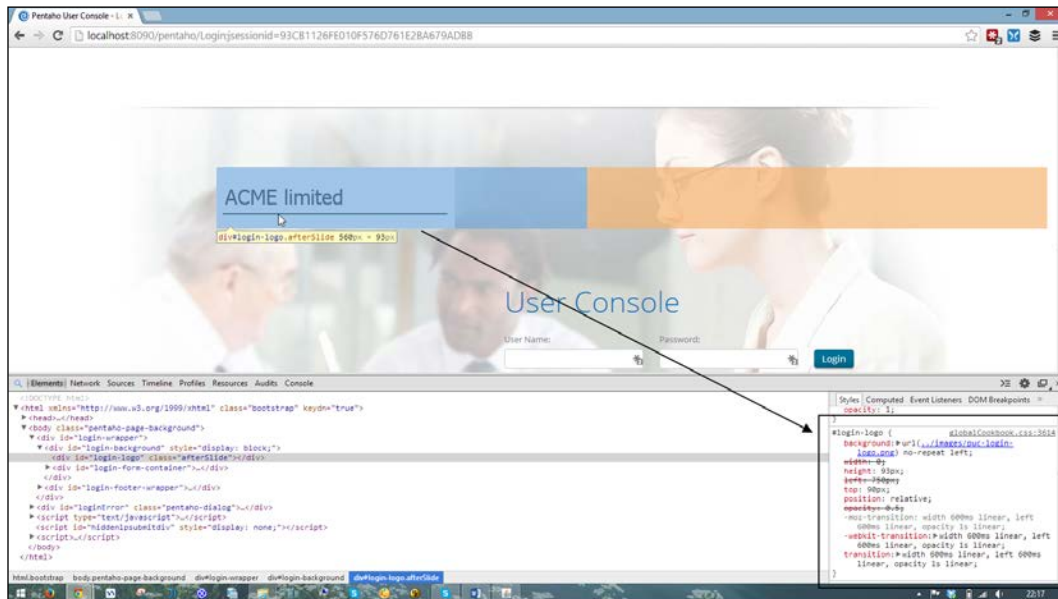
To start the Chrome Developer Tools, press *F12* from the browser window of the page that we want to inspect. Let's suppose we want to inspect the login page's logo to easily find the definition of the CSS element that references the logo file. From the Pentaho login page, press *F12*; the developer tools area appears at the bottom of the web page. A detailed view of the developer tools page is displayed in the following screenshot:



The screenshot shows a sample of Chrome Developer tools area



Press the button identified by the lens icon in the upper-left corner of the developer tools area. By doing this, we activate the inspector tool. This way, every time we move the cursor over a page element, that element becomes light blue in color. Move over the logo and, as soon as our logo turns light blue, click the left-mouse button to confirm the selection. Then, look at the right-hand side of the developer tools area and we will get the definition of the CSS ID element, which contains the reference to the logo, and the name of the CSS file, which contains the definition. We can clearly see this in the following screenshot, where we are referencing the logo's image and its definition in CSS as discovered by the tool:



The screenshot shows a sample of where we can find out details of the logo image in Chrome Developer tools area

As we can see, we completely identified the name of the logo file and, eventually, the name of the CSS file.

We can use this method any time we want to inspect the CSS elements and find their definitions. This tool is a valuable resource whenever we want to define new themes or customize the existing themes.

## See also

- ▶ Detailed documentation for the Chrome Developer Tools can be found at <https://developers.google.com/chrome-developer-tools>. Similarly, you can find documentation for Firebug at <http://getfirebug.com/faq/>. Both these tools are invaluable resources for any web developer.

---

## Adding new languages to Pentaho User Console

Another important thing is that Pentaho User Console is a multi-language application. Translating the user interface is fairly easy but elaborate. On one side, we have a set of strings (mostly labels) available through a set of properties' files containing sets of key value elements. Each set of properties' files (known as bundles) contain string translations for each language that we would like to make available through the interface. On the other side, we have the set of JSP pages that contains, for each label that must be printed in the user interface, the relative key contained in the property file. At runtime, given the user's locale coming from the browser, the key will be resolved into the localized string value, and this value will be displayed in the user interface.

Having said that, localizing the Pentaho user interface is just a matter of translating the set of related properties' files for the related language. The community has made available a set of translations for the Pentaho user interface; we have translations for all the main languages: Italian, German, French, Spanish, Portuguese, and so on. The language packages, available both for the CE and the EE versions of Pentaho, are available through the Pentaho marketplace. This recipe will show how we can install a set of translations by getting them from the Marketplace.

### Getting ready

For this recipe, the following conditions must be met:

- ▶ A Pentaho user must be successfully logged in to Pentaho User Console
- ▶ The Pentaho user must be a member of the **Administrator** role
- ▶ The latest versions of the following plugins must be installed: **Marketplace**, **CDF**, and **CDE**

### How to do it...

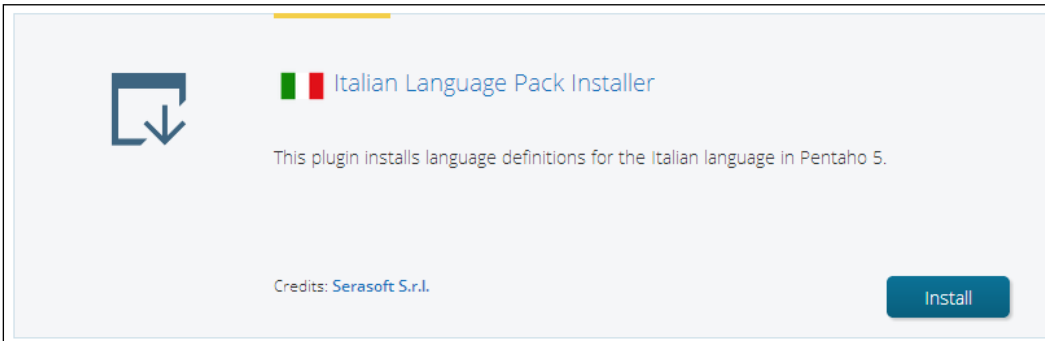
The following steps detail how we can install a set of language files from the Pentaho Marketplace:

1. Go to the **Marketplace** perspective by selecting it from the usual drop-down list.
2. The list of available plugins is displayed. Look through the available language packages for the one we want to install.

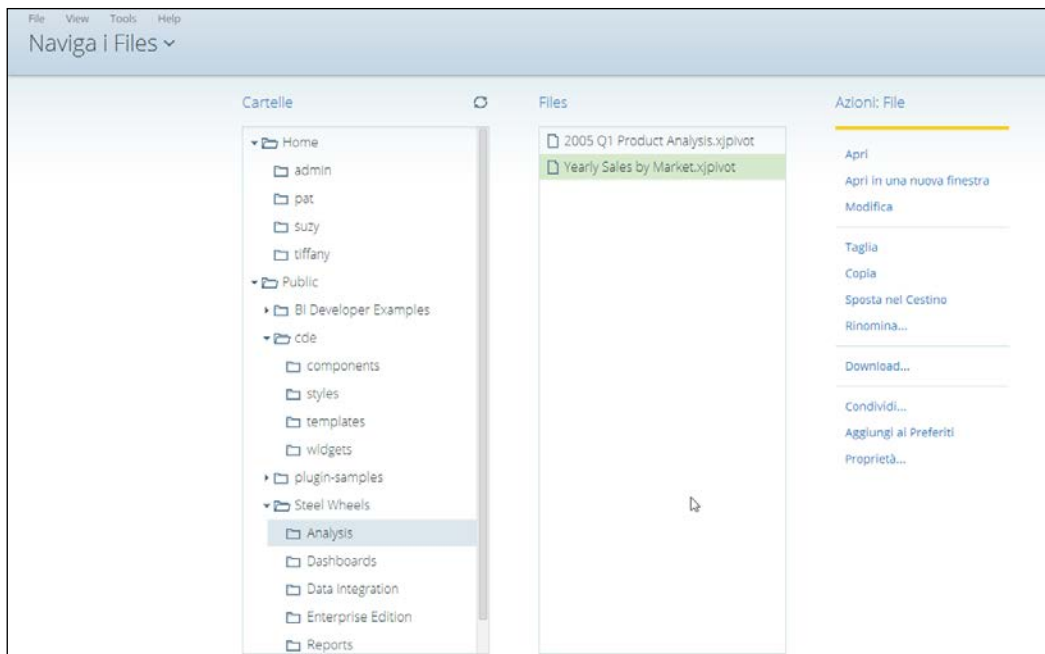
- Suppose we want to install the Italian language package. The following screenshot shows the package entry as found in the full marketplace plugins' packages list:



- Click on the big, green **Install** button. A dialog box asks if we are sure we want to install the package selected. Click on the **OK** button to continue.
- The language pack installer starts. At the end of the process, a dialog box informs us that the installation of the language pack has been completed successfully.
- After the installation is complete, restart the server for the changes to take effect.
- Once the server restarts, log in again to Pentaho User Console with a user who is a member of the **Administrator** role.
- Go to **Tools | Italian Language Pack Installer** and click on the menu item entry.
- A dialog appears at the center of the screen, informing us that the second phase of the installation process must be started. Click on the **Install** button to start the second phase of the language pack installation process (see the following screenshot for details):



- During this phase, the Italian (or whatever language we choose) language files are applied to Pentaho User Console. As soon as the second phase terminates, restart the Pentaho BA Server and log in with a user to Pentaho User Console to see the console translated in the new language. The following screenshot shows a sample of the user interface with language strings translated in Italian:



## How it works...

With this Pentaho release and the availability of a marketplace to easily share plugins and extensions, a big effort has been made by the community to build a good plugin in order to easily provide multilingual language packs.

First, we must access the marketplace and identify the entry for the language pack we want to install. Then, press the green button to start the first part of the language pack installation. A dialog box informs us about the successful installation of the first part of the language pack installation. Restart Pentaho BA Server and log in again to start the second part and complete the language pack installation. We can start the second part of the installation from the **Tools** menu by navigating to **Tools | Italian Language Pack Installer** and clicking on the menu item entry. A dialog box informs us that we are starting the second part of the process. Basically, during this phase, the installer applies the new language files to the right locations by substituting the existing ones. After the completion of the second phase, the installation is completed, and we must restart the server a last time and then log in to Pentaho User Console. After this last login, our Pentaho User Console is completely translated in the new language we just installed.

## Checking Pentaho BA Server logs from inside of Pentaho User Console

Sometimes, it is useful to have the possibility to check the Pentaho BA Server logs in order to verify the eventuality of a system malfunction or error. Usually, this is not possible unless we have direct access to the filesystem where the server is installed. With Pentaho BA Server 5, the Pentaho Log Manager plugin makes it possible to access the server's filesystem and show the content of a server log in Pentaho User Console. This recipe shows how we can install the plugin and manage a set of server logs.

### Getting ready

For this recipe, the following conditions must be met:

- ▶ A Pentaho user must be successfully logged in to Pentaho User Console
- ▶ The Pentaho user must be a member of the **Administrator** role
- ▶ The latest versions of the following plugins must be installed: **Marketplace**, **CDF**, and **CDE**

### How to do it...


The following steps detail how to install the Pentaho Log Manager plugin and manage a specific set of server logs remotely:

1. Go to the **Marketplace** perspective by selecting it from the usual drop-down list.
2. The list of available plugins is displayed. Look through the list and locate the entry for **Pentaho Log Manager** as shown in the following screenshot:



3. Click on the big, green **Install** button. A dialog box asks if we are sure we want to install the package selected. Click on the **OK** button to continue.
4. The **Pentaho Log Manager** installer starts. At the end of the process, a dialog box informs us that the installation of **Pentaho Log Manager** has been completed successfully.
5. After the installation is complete, restart the server for the changes to take effect.

6. Once the server is restarted, log in again to Pentaho User Console with a user who is a member of the **Administrator** role.
7. Go to **Tools | Log Manager** and click on the menu item entry.
8. Pentaho Log Manager starts by showing us the complete set of logs available in the logs directory.
9. Choose the log file for which we want to see the content and click on **View Log**, as shown in the following screenshot:

/tomcat/logs	catalina.2014-04-03.log	2014-04-03 23:59:56	346547	
/tomcat/logs	catalina.2014-04-04.log	2014-04-04 00:36:23	68948	  
/tomcat/logs	host-manager.2014-03-14.log	2014-03-14 12:30:37	0	 View Log

10. A new tab opens, displaying the log content of the selected file.

## How it works...

Checking Pentaho logs has always been a complicated matter because they were not accessible from any of the tools available in the Pentaho console, and this problem was more important in the CE version of Pentaho.

A new plugin called **Pentaho Log Manager** available in the marketplace gives you the possibility to access the logs directly from the server's filesystem. On the right-hand side of every log file, we have three different options:

- ▶ **View Log:** This gives you the possibility to view the log from the Pentaho console in a raw text form. This should be enough to understand what happens in case of any problems.
- ▶ **Export Log:** This gives you the possibility to export the log file for your convenience to save it safely in another position or to send it to someone for assistance.
- ▶ **Delete Log:** This gives you the possibility to delete the log file, if needed.

The entire set of log entries can be searched by name (we can look through the log files by searching their names) and is organized by pages with parameterized length that we can choose from a list.

Finally, yet importantly, in the upper side of the dashboard, we also have a useful function to massively delete logs older than a certain amount of days.

As we saw, we finally have an easy way to take care of the basics; we can see the log files when required and also maintain them when needed.

## Easily managing content in Pentaho Solution

Starting from Pentaho BA 5, Pentaho Solution is implemented as a JCR repository. This is a good thing because we have a standardized gate to our BI objects in Pentaho Solution. On the other side, we no longer have our content directly accessible in filesystem directories organized under the `pentaho-solutions` folder, but everything is in a JCR repository. Remember that even if the repository is saved to the filesystem (it can be in the database, for example, as for the EE version), the content is not accessible.

The community published a plugin named Pentaho Repository Synchronizer that is able to synchronize a Pentaho Solution path with the content of a directory in our filesystem, allowing you to keep the two aligned. This recipe will show how this plugin works by helping us to keep our solution repository synchronized with our developments.

### Getting ready

For this recipe, the following conditions must be met:

- ▶ A Pentaho user must be successfully logged in to Pentaho User Console
- ▶ The Pentaho user must be a member of the **Administrator** role
- ▶ The latest versions of the following plugins must be installed: **Marketplace**, **CDF**, and **CDE**

### How to do it...

The following steps detail how to install the Pentaho Repository Synchronizer plugin and manage a specific set of server logs remotely:

1. Go to the **Marketplace** perspective by selecting it from the usual drop-down list.
2. A list of the available plugins is displayed. Look through the list and locate the entry for **Pentaho Repository Synchronizer** as shown in the following screenshot:



3. Click on the big, green **Install** button. A dialog box asks if we are sure we want to install the package selected. Click on the **OK** button to continue.

4. The **Pentaho Log Manager** installer starts. At the end of the process, a dialog box informs us that the installation of **Pentaho Repository Synchronizer** has been completed successfully.
5. After installation is complete, go to the <biserver-home>/pentaho-solutions/system/repositorySynchronizer directory.
6. Locate the prs.xml file and open it with your favorite editor.
7. Look for the row with the following content:

```
<location path="jcr-solution: /pentaho!/" name="JCR"/>
```

8. Change the previous line as follows. In case we don't want to access the Pentaho server with the admin user, change the username with another that is more convenient for you.

```
<location path="jcr-solution:
  http://admin:password@localhost:8080/pentaho!/publ
  ic" name="JCR"/>
```

9. Look for the rows with the following content:

```
<excludes>
  <!-- Exclude all dot folders and files -->
  <exclude pattern=".*\.[^/]*(/.*)?"/>
  <!-- Exclude sample folders -->
  <exclude pattern=".*bi-developers(/.*)?"/>
  <exclude pattern=".*cde(/.*)?"/>
  <exclude pattern=".*plugin-samples(/.*)?"/>
  <exclude pattern=".*Steel\sWheels(/.*)?"/>
</excludes>
```

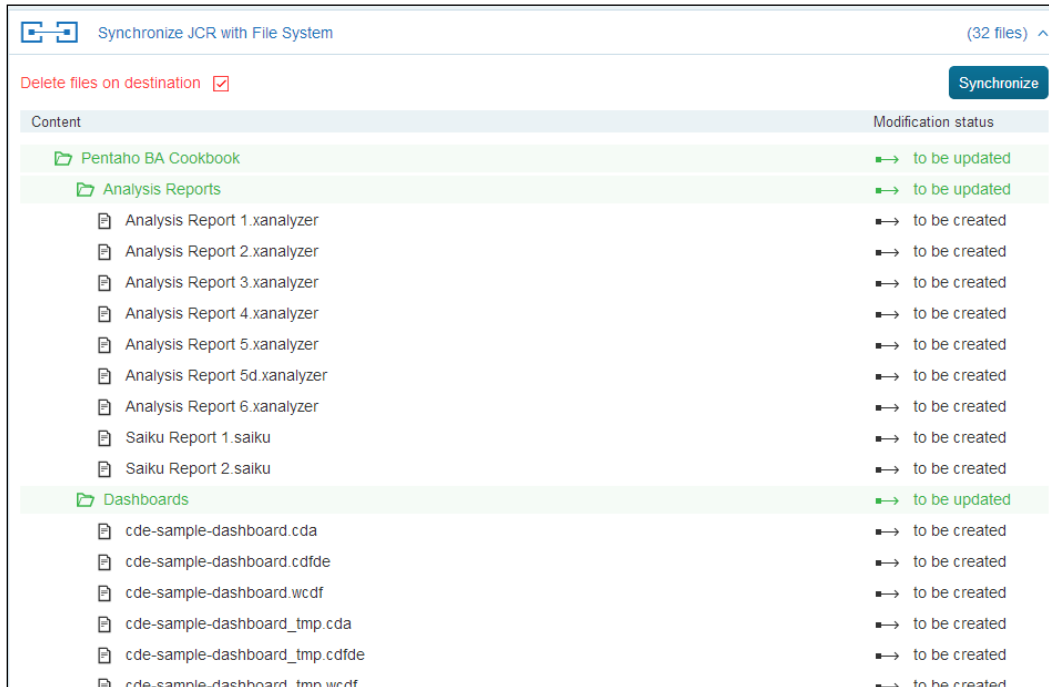
10. Change the previous lines as follows:

```
<excludes>
  <!-- Exclude all dot folders and files -->
  <exclude pattern=".*\.[^/]*(/.*)?"/>
  <!-- Exclude sample folders -->
  <exclude pattern=".*bi-developers(/.*)?"/>
  <exclude pattern=".*cde(/.*)?"/>
  <exclude pattern=".*plugin-samples(/.*)?"/>
  <exclude pattern=".*Steel\sWheels(/.*)?"/>
  <!-- Exclude pentaho operations mart folders -->
  <exclude pattern=".*Pentaho\sOperations\sMart(/.*)?"/>
</excludes>
```

11. Save the file and restart the server for the changes to take effect.

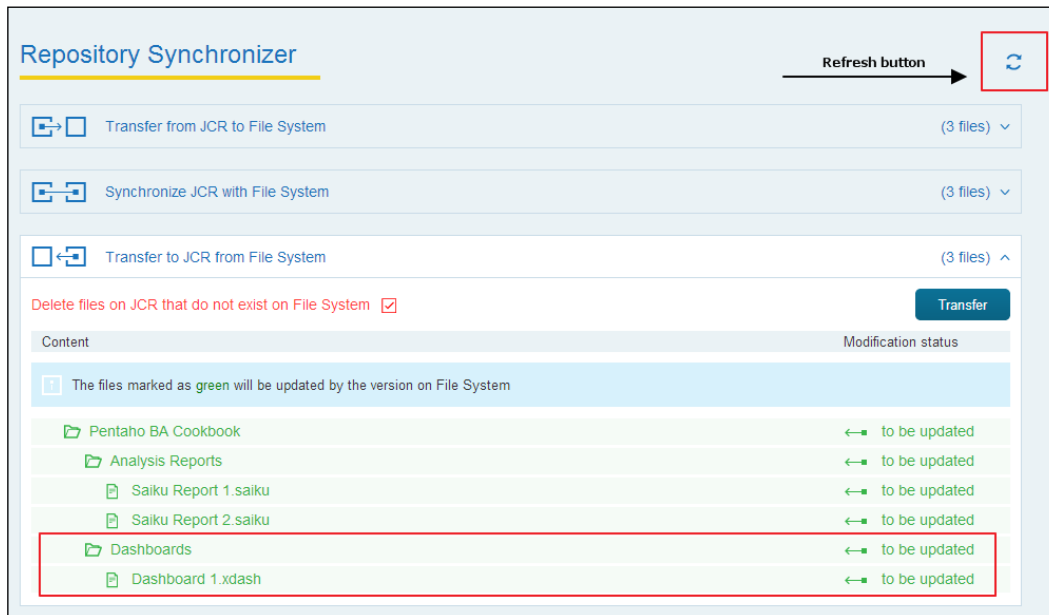


12. Once the server is restarted, log in again to Pentaho User Console with a user who is a member of the **Administrator** role.
13. Go to **Tools | RepositorySynchronizer** and click on the menu item entry.
14. If this is your first time with this plugin, **Pentaho Repository Synchronizer** starts by reading your JCR repository content and displays it on the dashboard, as shown in the following screenshot:



15. As we can see in the previous screenshot, the plugin extracted all the files of our samples developed during the book's recipes.
16. Let's start by aligning what we have on disk with the content of our repository. Expand the section of transfer files from JCR to the filesystem and then click on the **Transfer** button.
17. A dialog box opens, asking confirmation for the action that is required to be performed. Click on the **OK** button in the dialog.
18. As soon as the transfer is complete, look at the `<biserver_home>/pentaho-solutions` directory. A new directory called `repositorySynchronizer` appears, and below this directory, the plugin recreates the complete directories and files structure extracted from our Pentaho Solution.

19. Now let's add a new file to one of the exported directories; for example, locate the `<biserver-home>/pentaho-solutions/repositorySynchronizer/Pentaho BA Cookbook/Dashboards` directory and then open the `Dashboard 1.xdash` file with a text editor.
20. Save the file without modifying it, just to inform the system that we have updated it.
21. Coming back to Pentaho User Console, go to the **Opened** perspective and enter the already opened **RepositorySynchronizer** tab.
22. Click on the **Refresh** icon button to resynchronize the files.
23. Expand the **Transfer to JCR from File System** section. We will see that the plugin has considered our file as updated, so it will consider updating the JCR copy of the file (see the following screenshot for details):



24. Click on the **Transfer** button. A dialog box opens, asking confirmation for the action we are required to perform. Click on the **OK** button in the dialog.
25. The transfer to JCR starts and the files in Pentaho Solution are updated accordingly.

## How it works...

Here is another important tool that is useful under two situations:

- ▶ To help developers massively maintain the synch between a developer code repository and our Pentaho Solution each time there is a new release of our objects
- ▶ To help administrators migrate from different versions of Pentaho BA Server

The tool is simple and can be installed, as usual, from the marketplace in the same way that we saw in many other situations in this book.

Immediately after the end of the installation and before restarting Pentaho and accessing the tool, we must check the Repository Synchronizer configuration and see if it matches our needs. To do this, after installation is complete, go to the `<biserver-home>/pentaho-solutions/system/repositorySynchronizer` directory and open the `prs.xml` file with your favorite editor. Look for the row with the following content:

```
<location path="jcr-solution: /pentaho!/" name="JCR"/>
```

Change the previous line as follows. In case we don't want to access the Pentaho server with the admin user, change the username to another username that is more convenient for you:

```
<location path="jcr-  
solution:http://admin:password@localhost:8080/pentaho!  
/public" name="JCR"/>
```

This setting identifies the authenticated URL to our Pentaho BA Server installation and the path in the repository to consider for our synchronization. Do not forget the `!` character as a separator between the two parts. After the settings are completed successfully, restart the server to activate the plugin and then log in to Pentaho BA Server.

We can access the tool by navigating to **Tools | RepositorySynchronizer** in Pentaho User Console. The layout of the dashboard is divided in to three parts:

- ▶ **Transfer from JCR to File System:** This contains the list of files that must be transferred from the JCR repository to the filesystem because either they are new with respect to the filesystem files or we have files in the JCR repository that are not in the filesystem.
- ▶ **Transfer to JCR from File System:** This contains the list of files that must be transferred from the filesystem's JCR repository. This is because either files on the filesystem's directory are new with respect to the JCR files or we have files on the filesystem that are not in the JCR repository.
- ▶ **Synchronize with the File System:** This is a mix of the previous two because it tries to guess differences bidirectionally.

By expanding the related section, we can see the list of files that must be synchronized. We can start the synchronization by clicking on the **Synchronize** button.

The first time we start the tool, Pentaho tries to build a list of files that must be synched by guessing the differences. The location of the directory on the disk from where Pentaho looks for the files to synchronize with the content of the JCR repository is specified in `prs.xml`. Usually, it is set relative to `<baserver-home>/pentaho-solutions` by default, but we can change it according to our needs.



# Index

## A

**Access Control List (ACL) roles** 63

**Access Control List (ACL) users** 63

**Access method list box** 80

**Administration perspective**

about 13

accessing 40, 41

task, running to clean up aged  
generated files 72, 73

**advanced configuration parameters**

using 82, 83, 117, 118

**Advanced Options menu**

accessing 34

choices 35

**aged generated files**

cleaning up 72, 73

cleanup, scheduling 73-75

**aggregator property** 190

**All Authorities Search section** 62

**All Username Search section** 62

**analysis data source**

creating 92-94

parameters, defining manually 94-96

parameters, deleting 96

**authentication**

configuring, through LDAP server  
(CE version) 64-68

configuring, through LDAP server  
(EE version) 57-64

## B

**bands**

visibility flag, setting for 262, 263

**basic file properties**

displaying 24-26

**basic folder properties**

displaying 18-20

**blockout time interval**

defining 325-327

**Blueprint framework** 302

**Bootstrap** 296

**break** 261

**Browse Files perspective** 13

**BTable component**

about 296

configuring, for use in CDE dashboards 304

**Business Intelligence solution** 36

**business models**

deriving, from physical layer 129-132

**business table column properties**

reviewing 133-138

**business table entities**

joins, defining between 142-146

**business table fields**

formatting properties, applying 138, 139

**business view categories**

about 162

creating 146-149

## C

**calculated columns**

adding, to model entities 139-141

**calculated fields**

adding, functions used 263-266

**calculated measure**

adding 207-209

generically calculated

measure, adding 210, 211

predefined calculated measure

types, adding 210

## **CDE**

- about 280
- used, for creating dashboards 296-302

## **CDE dashboards**

- BTable component, configuring 304

## **cell height, tabular reports**

- managing 244

## **chart properties**

- Data field sets 290
- Display field set 290
- Theme drop-down list 290
- Type drop-down list 290

## **charts**

- adding, to report 274-278
- configuring 277, 278

## **column properties, business table**

- reviewing 133-138

## **column property 190**

## **command line**

- content, uploading from 36, 37

## **Community Charting Component (CCC) 301**

## **Community Dashboard Editor. *See* CDE**

## **Community Dashboard Framework (CDF) 302**

## **Community Data Access (CDA) 301**

## **company logo**

- adding, to Pentaho User Console
- login page 344-346

## **Concept Editor window**

- used, for creating concepts 123, 124

## **concepts**

- defining 121-125
- formatting, defining 124

## **Connection name field 80**

## **content**

- adding, to favorites 8, 9, 10
- easily managing, in Pentaho Solution 362-367
- permanently deleting, from Trash 31, 32
- removing, from favorites list 10, 11
- restoring, from Trash 30, 31
- uploading, from command line 36, 37
- uploading, to solution folder 33-36

## **contents, dashboard**

- linking, and enabling interactions 293-296

## **content types**

- selecting, to upload on BA Server 36

## **CSV file**

- data source, creating with wizard 101-104

## **CTools framework 280**

## **cube 186**

## **Currency format 207**

# **D**

## **dashboard**

- analysis report, opening in view mode 286
- content items, linking to
  - enable interactions 293-296
- creating, CDE used 296-303
- creating, for multiple-content 291, 292
- creating, Pentaho Dashboard Designer used 281-286
- prompts adding, to get user input 287-291

## **Dashboard Canvas area 286**

## **dashboard content zones**

- content, adding to 286

## **database connection pooling**

- enabling 90

## **Database Explorer dialog box 121**

## **database JDBC driver**

- adding, to list 82

## **Database Type list box 80**

## **data source**

- creating, from CSV file with wizard 101-103

## **Data Source Wizard**

- used, for data source creation from CSV file 101-103

## **datatype property 191**

## **dates**

- and numbers, displaying as results of function 266

## **default query 230**

## **default startup screen, Pentaho Mobile**

- changing 339, 340

## **Display Properties dialog box attributes**

- Created 20
- Hidden 20
- Last Modified 20
- Location 20
- Name 19
- Owner 19
- Size 20

Source 19

Type 19

**domain model elements**

security, applying 153-155

**E**

**ETL connections**

managing 234-236

**Excel format**

Pentaho analysis report,  
exporting in 215, 216  
reports, exporting in 184

**existing analysis data source**

updating 96, 97

**existing data source**

deletion 104, 105  
exportation 100

**existing JDBC data source**

updating 91, 92

**existing role**

deleting 52  
editing 53, 54  
edition, by adding new user 53  
edition, by changing  
Operation Permissions 54  
edition, by removing existing user 53  
removing, from user's role set 46

**existing user**

deleting 44, 45  
editing 45-47  
new role, adding 46  
password, changing 45

**Expression format 207**

**F**

**FALSE() function**

using 254

**favorites**

content, adding to 8-10  
content, removing from 10, 11

**fields**

removing, from report's working area 202

**files**

accessing, from mobile device 335, 336  
adding, to favorites in  
Pentaho Mobile 337-339

moving, to different folder 29, 30

moving, to Trash 28, 29

removing, from favorites in

Pentaho Mobile 339

renaming 23, 24

**filter**

about 178

adding, for limiting report's output 178-180

adding, to analysis report 220

modifying 180

removing 180

**folders**

accessing, from mobile device 335, 336

**formatter property 191**

**formatting properties**

applying, to business table fields 138, 139

**formatting rules**

applying, to measure's value 206, 207

**formula property 191**

**formulas**

used, for changing field properties

at runtime 250-254

**functions**

used, for adding calculated fields 263-266

used, for calculating

summarization totals 264, 265

**G**

**General Number format 207**

**generically calculated measure**

adding 210, 211

**geographical visualization**

creating, Pentaho Analyzer used 211-215

**graphical indicators**

adding, to table cells 202-204

**groups**

adding, to reports 171-174

used, for defining

report aggregations 258-262

**H**

**header labels, column**

changing 206

**height property, bands 263**

**hideMemberIf property 190**

**hide-on-canvas property, bands 262**



**HOLAP 186**  
**home directory, user**  
looking into 17  
**Home perspective 13**  
**HtmlObject property 302**  
**Hybrid Online Analytical Processing.**  
*See HOLAP*

## I

**Inherits folder permissions flag 22**  
**input parameter fields**

Default Value field 256  
Label field 256  
Name field 256  
Prompt section 257  
Query field 257  
Value Type drop-down list 256

**input parameters**

used, to obtain different  
report outputs 254-257

**interactive dashboard**

creating 293-296

**interactive report**

creating 162-168  
labels, modifying in 175

**interactive reporting**

about 162  
used, for designing report 162-168

**invisible-consume-space property**

using 263

**Is Formula Exact? property 141**

## J

**JAVA\_HOME environment variable 186**

**Java Native Directory Interface.** *See JNDI*

**JDBC connection properties**

manual specification 89

**JDBC database connections**

managing 113-117, 227-231

**JDBC data source**

configuring, to help in Mondrian  
schema design 194, 195

**JDBC driver**

using, with Pentaho Schema Workbench 194

**JDBC Driver**

adding 117

**JNDI 83**

**JNDI connection**

configuring, for development 225, 226  
defining, in BA Server 83-86  
using, for development 110-112

**JNDI JDBC data source**

creating 87-89

**joins**

defining, between business  
table entities 142-146

## K

**Key Performance Indicators (KPI) 279**

## L

**labels**

modifying, in interactive report 175

**language file set**

installing, from Pentaho  
Marketplace 357-359

**languages**

adding, to Pentaho User Console 357

**layouts**

used, for simplifying  
report development 237-243

**layout types**

block 242  
canvas 242  
inline 242  
row 243

**LDAP**

about 62  
URL 62

**LDAP Configuration section 60**

**LDAP query**

components 63

**LDAP server (CE version)**

authentication, configuring through 64-68

**LDAP server (EE version)**

authentication, configuring through 57-64

**Lightweight Directory Access Protocol.**

*See LDAP*

**long-running report**

scheduling, for execution 306-311

## M

### mail server

configuring, to connect to Gmail 71

### mail server configuration

managing 69-71

### master report 269

### MDX expression 191

### measure's value

formatting rules, applying to 206, 207

### measures value

Currency 207

Expression 207

General Number 207

Percentage (%) 207

### metadata data source

creating 97-99

localized bundles, managing 99

### metadata definitions

publishing, to BA server 157-160

### metadata domain models 162

### metadata layer results

testing 149-152

### metadata model domain

implementing, benefits 108

### microcharts

embedding, in reports

with sparklines 270-274

### model entities

new calculated columns, adding to 139-141

### MOLAP 186

### Mondrian schema

about 187

creating, Pentaho Schema

Workbench used 188-191

JDBC data source, configuring 194, 195

JDBC driver, adding 194

publishing 192, 193

### Multidimensional Analytical Processing.

*See* **MOLAP**

### multiple-content dashboard

creating 291, 292

### multiple filter options

Execution Time 317

Scheduled Resource 317

Schedule State 317

Schedule Type 317

Users 317

### MULTIVALUEQUERY function 274

## N

### Name property 189, 191

### native JDBC data source

creating 78-81

### new role

creating 47-51

### new user

creating 42-44

### NORMALIZEDARRAY function 274

### numbers

and dates, displaying as results

of function 266

## O

### OLAP implementation tools

HOLAP 186

MOLAP 186

ROLAP 186

### Online Analytical Processing

(OLAP) 8, 185, 186

### Opened perspective 13

### Operation Permissions

about 50

Administer Security 51

changing, to predefined System roles 55, 56

Create Content 50

Publish Content 51

Read Content 50

Schedule Content 51

## P

### PDF format

Pentaho analysis report,

exporting in 215, 216

reports, exporting in 184

### PDI

about 236

using 236

### Pentaho analysis report

available field set view, changing 198, 199

creating 195-197

creating, Saiku used 216-219

- exporting, in Excel or PDF format 215, 216
- filters, adding 220
- Pentaho Analyzer**
  - steps for trend calculation 209
  - used, for creating analysis report 195-198
  - used, for creating visualizations 211-215
- Pentaho BA Server**
  - accessing, from mobile device 331-333
  - JNDI connection, defining in 83-86
  - metadata definitions, publishing to 157-160
  - new user, defining in 42-44
  - Pentaho Metadata Editor security,
    - linking to 155-157
  - user, deleting from 44, 45
- Pentaho BA Server logs**
  - checking, from Pentaho User Console 360, 361
- Pentaho BA Server security**
  - bypassing 56
- Pentaho CE version**
  - about 162
  - URL 109
- Pentaho Dashboard Designer**
  - used, for creating dashboard from scratch 281-286
- Pentaho Dashboard Editor 280**
- Pentaho Data Integration.** *See* **PDI**
- Pentaho Enterprise Edition (EE)**
  - about 162
  - URL 109
- Pentaho Log Manager 360, 361**
- Pentaho Marketplace**
  - language file set,
    - installing from 357-359
  - Saiku, installing from 219
- Pentaho Metadata Editor security**
  - linking, to Pentaho BA Server 155-157
- Pentaho Metadata Query Builder**
  - used, for testing metadata layer results 152
- Pentaho Mobile**
  - about 330
  - accessing 331-333
  - configurations, changing 333, 334
  - default startup screen, changing 339-341
  - files, accessing 335, 336
  - files, adding to favorites 337-339
  - files, removing from favorites 339
  - folders, accessing 335, 336
  - opened content, closing 336, 337
- Pentaho Report Designer**
  - about 224
  - URL 224
- Pentaho Schema Workbench**
  - about 186
  - used, for creating Mondrian schema 188-191
- Pentaho Solution**
  - content, managing in 362-367
- Pentaho User Console**
  - about 162
  - customizing, themes used 347-356
  - new languages, adding to 357-359
  - Pentaho BA Server logs,
    - checking from 360, 361
- Pentaho User Console login page**
  - company logo, adding to 344-346
- Pentaho User Console perspectives**
  - about 13
  - Administration 13
  - Browse Files 13
  - Home 13
  - Opened 13
  - Scheduled 13
- Percentage (%) format 207**
- permissions, file**
  - changing 27, 28
- permissions, folder**
  - changing 20-22
- physical layer**
  - business models, deriving from 129-132
  - defining 118-121
- physical layer table columns**
  - reviewing 126-129
- Populator section 62**
- predefined calculated measure types**
  - adding 210
  - % of 210
  - Rank 210
  - Running Sum 210
- prompts**
  - about 181
  - adding, for obtaining user input 181
  - adding, to get user input from dashboard 287-291

- appearance, modifying 182, 183
- applying, to report 181
- behavior, modifying 182, 183
- unnneeded report prompts, removing 182

**properties, report object**  
changing, at runtime with formulas 250-254

**protovis 296**

## R

### recurrence options

- Cron 310
- Daily 308
- Monthly 309
- Run Once 308
- Seconds, Minutes, Hours 308
- Weekly 309
- Yearly 310

### Relational Online Analytical Processing.

*See* **ROLAP**

### Remove Data Source dialog box 104

### report

- charts, adding to 274-278
- designing, interactive
  - reporting used 162-168
- editing 169, 170
- exporting, in Excel format 184
- exporting, in PDF format 184
- groups, adding to 171-174
- microcharts, embedding in 270-274
- prompt, applying to 181
- totals, adding to 171-174

### report aggregations

- defining, groups used 258-262

### report development

- simplifying, layouts used 237-243

### report output

- limiting, with filters 178-180

### responsive behavior, Pentaho Mobile 341

### re-usable formatting rules

- defining, via style sheets 245-250

### ROLAP 94, 186

### row categories

- subtotals, adding to 199-201

### running executions

- stopping 321-323

## S

### Saiku

- installing, from Pentaho Marketplace 219
- URL 216
- used, for creating analysis report 216-219
- visualizations, designing 220, 221

### Saiku Reporting 162

### schedule creation

- preventing 323, 324

### Scheduled perspective 13

### scheduled task

- executing, in advance 314-316

### schedule entries list

- schedules, filtering 316-318

### schedule execution

- checking 320, 321

### schedule properties

- updating 311-313

### schedules

- deleting 313, 314
- filtering, in schedule
  - entries list 316-318

### security

- applying, to domain
  - model elements 153-155

### Select roles from LDAP server field 60

### Select user from LDAP server field 60

### SINGLEVALUEQUERY function 274

### solution

- files, accessing 11, 12
- folders, accessing 11, 13

### Solution Explorer 285

### solution folder

- content, uploading to 33-36
- creating 13, 14
- file, renaming 23, 24
- moving, to Trash 17, 18
- renaming 15, 16

### solution repository

- designing, rules 15

### sort order, column

- changing 204, 205

### sparklines

- used, for embedding microcharts
  - in reports 270-274

## **SQL query**

- clauses, adding to 234
- defining, SQL Query Designer used 231, 232

## **SQL query clauses**

- adding, to query 234

## **SQL Query Designer**

- used, for defining SQL query 231, 232
- used, for defining
  - WHERE conditions 232, 233

## **Style Definition Editor**

- used, for defining style rules set 248

## **style rules**

- defining, Style Definition Editor used 248, 249

## **style sheets**

- using, to consistently manage fields formats 245-250

## **subreports**

- used, for embedding content 267-270

## **subtotals**

- adding, to row categories 199-201

## **supported file types, uploading to BA Server**

- Analyzer 36
- Dashboards and CTools related files 36
- Reporting 36

## **system roles**

- authenticated role 56
- managing 55
- Pentaho BA Server security, bypassing 56

## **System roles, predefined**

- Operation Permissions, changing to 55, 56

## **T**

### **table cells**

- graphical indicators, adding to 202-204

### **table columns**

- reorganizing 176-178

### **tabular reports**

- cell height, managing 244

### **task execution**

- scheduling 306-311

### **themes**

- used, for customizing Pentaho User Console 347-356

## **Tomcat configuration file**

- updating, to define JNDI database connection 84, 85

## **totals**

- adding, to reports 171-174

## **Trash**

- content items, permanently deleting from 31, 32
- content items, restoring from 30, 31
- file, moving to 28, 29
- folder, moving to 17, 18

## **trend calculation steps**

- Pentaho Analyzer used 209

## **TRUE() function**

- using 254

## **U**

### **uniqueMembers property 190**

### **user task**

- running, in background 318-320

## **V**

### **visibility flag, bands**

- setting 262, 263

### **visible property**

- using 263

### **visualization capabilities, graphical indicators**

- Color Scale 204
- Data Bar 204
- Trend Arrow 204

## **W**

### **Web Ad-Hoc Query Reporting (WAQR) 162**

### **WHERE conditions**

- defining, SQL Query Designer used 232, 233

### **working area, report**

- fields, removing from 202

## **X**

### **XPath**

- about 250
- URL 250



## **Thank you for buying Pentaho Business Analytics Cookbook**

### **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](http://www.packtpub.com).

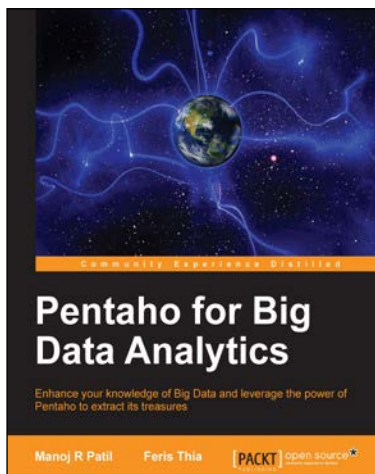
### **About Packt Open Source**

In 2010, Packt launched two new brands, Packt Open Source and Packt Enterprise, in order to continue its focus on specialization. This book is part of the Packt Open Source brand, home to books published on software built around Open Source licenses, and offering information to anybody from advanced developers to budding web designers. The Open Source brand also runs Packt's Open Source Royalty Scheme, by which Packt gives a royalty to each Open Source project about whose software a book is sold.

### **Writing for Packt**

We welcome all inquiries from people who are interested in authoring. Book proposals should be sent to [author@packtpub.com](mailto: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.



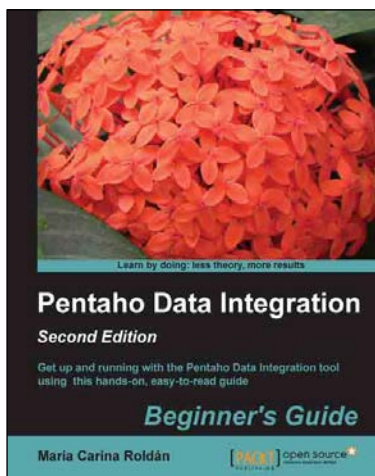
## **Pentaho for Big Data Analytics**

ISBN: 978-1-78328-215-9

Paperback: 118 pages

Enhance your knowledge of Big Data and leverage the power of Pentaho to extract its treasures

1. A guide to using Pentaho Business Analytics for big data analysis.
2. Learn Pentaho's visualization and reporting tools with practical examples and tips.
3. Precise insights into churning big data into meaningful knowledge with Pentaho.



## **Pentaho Data Integration Beginner's Guide** *Second Edition*

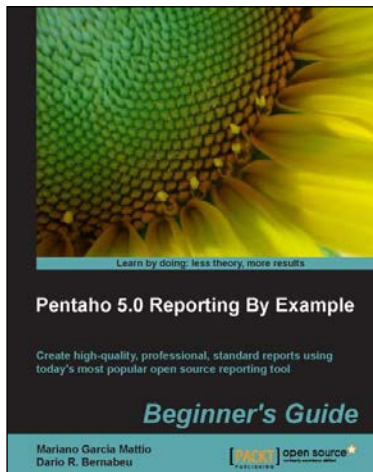
ISBN: 978-1-78216-504-0

Paperback: 502 pages

Get up and running with the Pentaho Data Integration tool using this hands-on, easy-to-read guide

1. Manipulate your data by exploring, transforming, validating, and integrating it.
2. Learn to migrate data between applications.
3. Explore several features of Pentaho Data Integration 5.0.
4. Connect to any database engine, explore the databases, and perform all kind of operations on databases.

Please check [www.PacktPub.com](http://www.PacktPub.com) for information on our titles

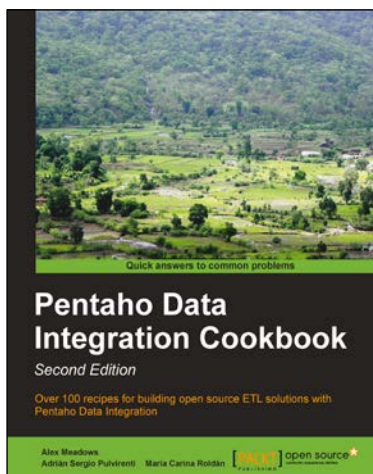


## **Pentaho 5.0 Reporting by Example Beginner's Guide**

ISBN: 978-1-78216-224-7      Paperback: 342 pages

Create high-quality, professional, standard reports using today's most popular open source reporting tool

1. Install and configure PRD in Linux and Windows.
2. Create complex reports using relational data sources.
3. Produce reports with groups, aggregate functions, parameters, graphics, and sparklines.



## **Pentaho Data Integration Cookbook Second Edition**

ISBN: 978-1-78328-067-4      Paperback: 462 pages

Over 100 recipes for building open source ETL solutions with Pentaho Data Integration

1. Integrate Kettle in integration with other components of the Pentaho Business Intelligence Suite, to build and publish Mondrian schemas, create reports, and populate dashboards.
2. This book contains an organized sequence of recipes packed with screenshots, tables, and tips so you can complete the tasks as efficiently as possible.
3. Manipulate your data by exploring, transforming, validating, integrating, and performing data analysis.

Please check [www.PacktPub.com](http://www.PacktPub.com) for information on our titles