

Professional Expertise Distilled

# Getting Started with Dynamics NAV 2013 Application Development

A simple and practical guide to creating a relevant application for  
your company using Dynamics NAV 2013

Alex Chow

**[PACKT]** enterprise   
PUBLISHING professional expertise distilled

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

# Getting Started with Dynamics NAV 2013 Application Development

A simple and practical guide to creating a relevant application for your company using Dynamics NAV 2013

**Alex Chow**



BIRMINGHAM - MUMBAI

# Getting Started with Dynamics NAV 2013 Application Development

Copyright © 2013 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: May 2013

Production Reference: 1160513

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

ISBN 978-1-84968-948-9

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

Cover Image by Marcus Grandon ([marcusgrandon@mac.com](mailto:marcusgrandon@mac.com))

# Credits

**Author**

Alex Chow

**Project Coordinator**

Shiksha Chaturvedi

**Reviewers**

Daniel Rimmelzwaan

David Roys

Bill Warnke

Eric Wauters (waldo)

**Proofreader**

Paul Hindle

**Indexer**

Tejal Soni

**Acquisition Editor**

Joanne Fitzpatrick

**Graphics**

Abhinash Sahu

**Lead Technical Editor**

Neeshma Ramakrishnan

**Production Coordinator**

Pooja Chiplunkar

**Technical Editor**

Worrell Lewis

**Cover Work**

Pooja Chiplunkar

**Copy Editors**

Aditya Nair

Alfida Paiva

Sajeev Raghavan

Laxmi Subramanian

# About the Author

**Alex Chow** has been working with Microsoft Dynamics NAV, formerly Navision, since 1999. Over the years, he has conducted hundreds of implementations across multiple industries. His customers range from \$2-million-a-year small enterprises to \$500-million-a-year multinational corporations.

Over the course of his Dynamics NAV career, he has often been designated as the primary person responsible for the success and failure of a Dynamics NAV implementation. His extensive career in the Dynamics NAV business is evidence of his success rate and expertise.

With a background in implementing all the functions and modules in and outside of Microsoft Dynamics NAV, Alex has encountered and resolved the most practical to the most complex requirements and business rules. Through these experiences, he has learned that sometimes you have to be a little crazy to have a competitive edge.

Believing that sharing these experiences and this knowledge would benefit the Dynamics NAV community, Alex writes about his journey at [www.dynamicsnavconsultant.com](http://www.dynamicsnavconsultant.com). He also founded AP Commerce, Inc. ([www.apcommerce.com](http://www.apcommerce.com)), a full service Dynamics NAV service center, in 2005.

Alex lives in Southern California with his beautiful wife and two lovely daughters, and he considers himself the luckiest man in the world.

# About the Reviewers

**Daniel Rimmelzwaan** was born and raised in the Netherlands, and moved to the USA at the end of 1999 to be with his new American wife. In Holland, he worked as a Microsoft Access and VBA developer. While looking for a job as a VB developer in the USA, he was introduced to Navision by a “VB Recruiter” and was intrigued by the simplicity of its development tools. He decided to accept a job offer as a Navision Developer with the firm intention to continue looking for a “real” developer job.

More than 13 years later, having been involved with all aspects of NAV implementations, Daniel is still working with NAV. He currently owns his own business (RIS Plus), where he does business analysis and solution design and is enjoying his career more than ever.

Ever since he started working with NAV, Daniel has been an active member of the online communities, such as `mibuso.com` and `dynamicsuser.net`, and online forums managed by Microsoft. For his contributions to these communities, Daniel received his first of eight consecutive Microsoft Most Valuable Professional Awards in July 2005, which was just the second year that the award existed for NAV. Microsoft gives the MVP award to independent members of technology communities around the world, and recognizes people who share their knowledge with other members of the community.

Daniel has also worked as a reviewer for *Microsoft Dynamics NAV 2009 Application Design*, Mark Brummel, Packt Publishing and *Microsoft Dynamics NAV 2009 Professional Reporting*, Steven Renders, Packt Publishing.

Daniel lives with his wife and two kids in Michigan, USA.



**David Roys** works as a Dynamics NAV consultant and programmer for Intergen, and in his spare time he writes novels about people who work with computers. He co-authored one of the first Packt Publishing books on Dynamics NAV and has gone on to read and review a number of their publications. David has his suspicions that no one reads the biographies of technical reviewers, so to prove him wrong, you should go and like his Facebook page <http://facebook.com/DavidRoysAuthor>. You can read David's blog on NAV at <http://teachmenav.com/blogs/dave>.

**Bill Warnke** has been working with Microsoft Dynamics NAV as an IT Administrator tasked with creating an integration between NAV and an existing mobile invoicing solution since 2006. He was hooked on the rapid development, all-in-one environment, and the integration capabilities of NAV. Since then, he's worked exclusively with Dynamics NAV.

He works for ABC Computers, Inc., a NAV reseller and Microsoft Gold Certified partner based out of Waupaca, WI. He is part of the ERP delivery team, and works on new NAV implementations, upgrades, and integrations. Find out more about what they have to offer at [www.abc-computers.com](http://www.abc-computers.com).

He tries to maintain a blog presence at [www.billwarnke.com](http://www.billwarnke.com), but his wife and daughter (and dog too) have slowed that down.

**Eric Wauters (waldo)** is one of the founding partners of iFacto Business Solutions ([www.ifacto.be](http://www.ifacto.be)). With his 11 years of technical expertise, he is an everyday inspiration to its development team. As Development Manager, he continually acts on iFacto's technical readiness and guarantees that he and iFacto are always on top of the latest Microsoft Dynamics NAV developments.

Apart from that, he is also very active in the Microsoft Dynamics NAV community, where he tries to solve technical issues and strives to share his knowledge with other Dynamics NAV enthusiasts. Surely, many among you will have read some of Eric's posts on [Mibuso.com](http://Mibuso.com), [Dynamicsusers.net](http://Dynamicsusers.net), or on his own blog ([www.waldo.be](http://www.waldo.be)), which he invariably signs with "waldo". A few years ago, he co-founded the Belgian Dynamics Community, a platform for all Belgian Dynamics NAV users, consultants, and partners, enabling knowledge sharing and networking. His proven track record has entitled him to be awarded the Microsoft Most Valuable Professional (MVP) for Microsoft Dynamics NAV consecutively since 2007.

# 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.

## Instant Updates on New Packt Books

Get notified! Find out when new books are published by following @PacktEnterprise on Twitter, or the *Packt Enterprise* Facebook page.





# Table of Contents

|  |           |
|--|-----------|
| <b>Preface</b>   | <b>1</b>  |
| <b>Chapter 1: Getting Dynamics NAV 2013 on Your Computer – For (Almost) Free</b>       | <b>7</b>  |
| <b>Getting your free copy</b>  | <b>8</b>  |
| <b>Installing the software</b>   | <b>10</b> |
| Installing Visual Studio Web Developer 2010 Express                                    | 12        |
| A quick overview of the additional contents of the installation files for Dynamics NAV | 12        |
| <b>A look at what is installed</b>   | <b>15</b> |
| Microsoft Dynamics NAV 2013 Administration Shell                                       | 16        |
| Microsoft Dynamics NAV 2013 Development Environment                                    | 17        |
| Microsoft Dynamics NAV 2013 Windows Client   | 18        |
| Microsoft Dynamics NAV Administration  | 18        |
| The SQL Server 2012 folder   | 19        |
| The SQL Server 2012 report builder   | 19        |
| <b>Getting your license</b>  | <b>19</b> |
| Demo license   | 20        |
| MSDN license   | 20        |
| A full On-Premise license  | 21        |
| The cloud license  | 22        |
| <b>Summary</b>   | <b>23</b> |
| <b>Chapter 2: Getting Familiar with Dynamics NAV 2013</b>                              | <b>25</b> |
| <b>But first, a little history</b>   | <b>26</b> |
| <b>The Windows Client (WC) interface</b>   | <b>29</b> |
| Exploring the role center page   | 30        |
| Page search  | 32        |
| Accessing other functional areas   | 33        |
| <b>Exploring the list page</b>   | <b>35</b> |

|   |           |
|---|-----------|
| Personalizing the list page   | 38        |
| Searching your data based on filters  | 42        |
| Exploring the card page   | 45        |
| Personalizing the card page   | 47        |
| Exploring the document page   | 47        |
| Exploring the rest of the RTC environment   | 49        |
| The Development Environment interface   | 49        |
| Summary   | 52        |
| <b>Chapter 3: Exploring the Data Structure and Basic<br/>Layout of Dynamics NAV</b> | <b>53</b> |
| Exploring the different departments   | 54        |
| Drilling across modules and departments   | 54        |
| Sales and marketing   | 54        |
| Going into the inventory  | 57        |
| Continuing on to the vendor   | 58        |
| Drilling down to the detailed transactions  | 59        |
| Keep drilling   | 61        |
| Creating a sales order  | 62        |
| Drill it on your own  | 71        |
| Summary   | 72        |
| <b>Chapter 4: Determining a Task List</b>   | <b>73</b> |
| Who you are   | 74        |
| Your company  | 74        |
| Identifying the major pains in the company  | 76        |
| Understanding the current operation   | 76        |
| Listing out all of the problems   | 77        |
| Defining the non-negotiable must-haves  | 78        |
| Designing the solution  | 79        |
| Summary   | 80        |
| <b>Chapter 5: Finding Similar Functions for Inspiration</b>                         | <b>81</b> |
| A closer look at the requirements   | 81        |
| Where have you seen similar behavior?   | 83        |
| A closer look at the Sales Header<br>table (36)                                     | 90        |
| Data types in Dynamics NAV  | 92        |
| Primary key and indexes   | 94        |
| Looking at C/AL behind the table<br>(the code)                                      | 94        |
| Table relations   | 96        |
| A closer look at the Sales Order page (42)  | 98        |

|  |            |
|--|------------|
| Looking at the properties                                      | 101        |
| A closer look at the Sales Order Subform page (46)             | 102        |
| Looking at C/AL on the page (the code)                         | 103        |
| Summary  | 103        |
| <b>Chapter 6: Creating the Application – Tables</b>            | <b>105</b> |
| Creating the table and identifying the primary key(s)          | 106        |
| Saving, compiling, and running our table                       | 108        |
| Primary keys   | 110        |
| Checking our requirements list                                 | 110        |
| Adding new fields to the tables                                | 111        |
| Defining table relations in fields                             | 112        |
| Creating the Complaint Line table                              | 115        |
| Creating a conditional table relationship                      | 118        |
| Adding a composite primary key                                 | 122        |
| Adding the Complaint Comments table                            | 124        |
| Summary  | 125        |
| <b>Chapter 7: Creating the Application – Pages and Reports</b> | <b>127</b> |
| Creating the Product Complaint page                            | 128        |
| Linking FactBoxes  | 132        |
| Creating the Product Complaint subpage                         | 134        |
| The AutoSplitKey property                                      | 135        |
| Creating the Product Complaint Comments page                   | 137        |
| Linking the pages together                                     | 138        |
| Create an analysis report using wizards                        | 143        |
| Summary  | 152        |
| <b>Chapter 8: Extending Our Application</b>                    | <b>153</b> |
| A quick look at our user requirements                          | 154        |
| A quick test of our application                                | 154        |
| Generate unique document numbers automatically                 | 156        |
| Creating a number series for our application                   | 157        |
| Programming our table for the number series                    | 160        |
| Put our code in the table                                      | 165        |
| Defaulting fields using code                                   | 167        |
| Defaulting fields using FlowFields                             | 171        |
| Defaulting an item description on the line table               | 173        |
| Changing the properties of the decimal values                  | 173        |
| Creating a separate screen for closed complaints               | 174        |
| Creating the list page   | 176        |

*Table of Contents*

---

|   |            |
|---|------------|
| <b>Data clean up</b>  | <b>180</b> |
| <b>Adding the application to the RTC menu</b>   | <b>182</b> |
| <b>Testing our application</b>  | <b>185</b> |
| <b>Last check of our requirement list</b>   | <b>189</b> |
| <b>Summary</b>  | <b>189</b> |
| <b>Chapter 9: Dynamics NAV Modules to Address the Specific Needs of Your Business</b> | <b>191</b> |
| <b>Exploring the Help tool</b>  | <b>192</b> |
| <b>Exploring the Warehouse Management functionality</b>                               | <b>195</b> |
| <b>Exploring the Manufacturing functionality</b>                                      | <b>197</b> |
| <b>Exploring the Jobs functionality</b>   | <b>198</b> |
| <b>Exploring the Service Management functionality</b>                                 | <b>200</b> |
| <b>Summary</b>  | <b>202</b> |
| <b>Appendix: Additional Resources and Conclusion</b>                                  | <b>203</b> |
| <b>Official online resource</b>   | <b>203</b> |
| Connect online  | 204        |
| CustomerSource  | 204        |
| MSDN site   | 205        |
| Microsoft Dynamics Community  | 205        |
| <b>Unofficial online resource</b>   | <b>205</b> |
| Online forums   | 205        |
| Blogs   | 206        |
| <b>Dynamics NAV add-ons</b>   | <b>206</b> |
| <b>Dynamics NAV solution center finder</b>  | <b>207</b> |
| <b>Summary</b>  | <b>207</b> |
| <b>Index</b>  | <b>209</b> |

---

# Preface

Let me start out by saying congratulations on your decision to work with Dynamics NAV. When I started working with Dynamics NAV (formerly known as Navision) back in 1999, Dynamics NAV was nothing more than an accounting system out of Denmark. After a couple of releases, acquisition by Microsoft, and a couple more releases, Dynamics NAV has become a full Enterprise Resource Planning (ERP) software with rich functionality. With every release, we see technical and functional improvements. And they're not yet done.

At the time of writing, the Dynamics NAV installation base was 94,000 companies (<http://www.microsoft.com/en-us/dynamics/erp-nav-overview.aspx>). No other ERP software for the small and mid-market comes close to that number. In addition, Dynamics NAV has a wide range of add-on solutions that are available. Most of these add-ons are built directly within the Dynamics NAV environment with the same user interface. So, by using these add-ons, your company would not need to learn any other new software. One of the main selling points of Dynamic NAV from the very beginning is the ability to customize it exactly the way you run your business. Because of its flexibility, you can find a lot of tutorials and explanations on how to develop specific tasks, but not a lot of tutorials on how to create a project from scratch.

To get acquainted with the Dynamics NAV environment, it's important for the user to create an entire project from start to finish, not just a specific element, but everything from understanding the business problem, designing it, developing it, and integrating it to Dynamics NAV. By doing so, the user can understand the power (and the danger) of customization and speak on the same terms when consultants are making recommendations on customization.

Your company has made a wise decision to use Dynamics NAV as its main business software. But what good is the world's most powerful software if you don't know how to use it? Dive in and learn the software from a development standpoint and take the initial steps to unlock the software's potential.



This book will walk you through creating an application from start to finish. Once you know how to create a working application that the users can use, you will have the knowledge and the resources needed to create other applications and fill in the blanks with additional resources that are published on the Web.

## What this book covers

*Chapter 1, Getting Dynamics NAV 2013 on Your Computer – For (Almost) Free*, teaches you where and how to download free copies of Dynamics NAV. We will walk through the installation and configuration of Dynamics NAV on your computer so that you can start playing around with it. We will also explore using Dynamics NAV in the cloud environment to bypass the installation process altogether.

*Chapter 2, Getting Familiar with Dynamics NAV 2013*, will go through and highlight some areas to begin navigating around the system. We will learn the design concepts for Dynamics NAV and how to personalize the settings for our session.

*Chapter 3, Exploring the Data Structure and Basic Layout of Dynamics NAV*, explores how the data flows within the most commonly used modules in Dynamics NAV. We will look up, drill down, and drill across the application to find the information we're looking for.

*Chapter 4, Determining a Task List*, will look at some of the requirements for our fictitious company and show you how to create a task list for the project.

*Chapter 5, Finding Similar Functions for Inspiration*, explores the power of Dynamics NAV to customize a solution to fit our needs. There is not much customization that you will do that has not been already done before. In this chapter, we will examine the requirements from the users and find existing functions in Dynamics NAV to model our solution after.

*Chapter 6, Creating the Application – Tables*, will teach us how to build custom tables for our project to satisfy the task at hand. The tables are the starting point of every custom application that you will build in Dynamics NAV. We will reference the existing Sales Order function as the basis of our custom application.

*Chapter 7, Creating the Application – Pages and Reports*, allows us to add, modify, and delete the information without having to use the table itself. We discover how to create pages for the users to interface with the data. We will also create reports so that we can get meaningful outputs from the data entered.

*Chapter 8, Extending Our Application*, helps us add additional functionality to our tables, pages, and MenuSuite to improve user experience.

*Chapter 9, Dynamics NAV Modules to Address the Specific Needs of Your Business*, explores the additional capabilities of Dynamics NAV as a comprehensive ERP solution for the mid-market. The setup in the demonstration database can be quite overwhelming. This chapter will give us a glimpse of the advanced modules in Dynamics NAV so we can understand the functionality out of the box.

*Appendix, Additional Resources and Conclusion*, will examine the additional resources related to Dynamics NAV. This book will help you get started so you can quickly get up to speed in Dynamics NAV. If Dynamics NAV is something you're interested in, and you're hungry for more in-depth information, you will find some places you can go here.

## What you need for this book

For this book, you will need the following:

- Microsoft Dynamics NAV 2013:
  - Microsoft Dynamics NAV Windows Client
  - Microsoft Dynamics NAV Development Environment
  - SQL Server Express 2008 or higher
- Visual Studio Web Developer 2010 Express:
  - Visual Studio Web Developer 2010 Express or newer
- Microsoft Visual Studio 2010 Shell (Integrated) Redistributable Package
- Miscellaneous tools:
  - Java Client (for the cloud environment)

## Who this book is for

This book is for any user whose company has bought or has plans to buy Dynamics NAV as their main business software. This is also for any developer, who may be familiar with another accounting software, but who wants to get into the Dynamics NAV field.

## 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, and Twitter handles are shown as follows: “The built-in `Cmdlet` allows the administrator to configure and troubleshoot permissions and connection problems on a local or remote computer.”

A block of code is set as follows:

```
IF "No." = '' THEN BEGIN
    TestNoSeries;
    NoSeriesMgt.InitSeries(GetNoSeriesCode,xRec."No. Series","Posting
Date","No.,""No. Series");
END;
```

**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: “After the installation finishes, if you click on your **Start** icon in Windows, you’ll notice a few new icons.”

 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.

## 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 Dynamics NAV 2013 on Your Computer – For (Almost) Free

*"Try them, try them, and you may! Try them and you may, I say."  
– Dr. Seuss, Green Eggs and Ham*

This chapter will walk you through downloading and installing Dynamics NAV on your computer so that you can try out the software without having to hire external consultants or piece together online information.

Whether you're getting into Dynamics NAV because you're interested in a profession in this field, or because your company is interested in using Dynamics NAV as their ERP system, trying the software before you make the commitment will ensure you're making the best decision for you and your company.

There are three components that are needed for you to get into the development environment in Dynamics NAV. They are:

- The Dynamics NAV installation software
- Visual Studio 2012 Express
- The license file

The installation files can be downloaded for free. The installation files come with the full development environment. They will install almost everything you need to work with Dynamics NAV. The files are the same for a single-user installation and a multinational corporation.



To create and modify reports in Dynamics NAV, you will need to have Visual Studio installed on your computer. Dynamics NAV uses the RDLC reporting method, which means the reports do not get processed on the SQL server; rather, it's processed on the server where the middle tier is installed. The last part of going into the development environment is getting the proper license. Yes, you can download the software for free, but the license will cost you some money in terms of an MSDN subscription.

We will go through the online resources to download your copy of Dynamics NAV to be installed on your computer. We will also explore ways of trying out the development environment in Dynamics NAV by signing up for a free trial using the cloud service that is available.

## **Getting your free copy**

Before we go through the trouble of downloading the software, make sure the computer you're working with has the proper specifications in order to do a full installation. For a list of the hardware requirements, take a look at the following link:

[http://msdn.microsoft.com/en-us/library/dd301054\(v=nav.70\).aspx](http://msdn.microsoft.com/en-us/library/dd301054(v=nav.70).aspx)

Once we've verified that our hardware is good, we can start our journey and become familiar with the Dynamics NAV development environment by getting a copy of the software.


Microsoft has the installation files available for download; however, you have to be signed up as a Microsoft partner, have MSDN access, or already be a Microsoft Dynamics customer with access to download from the Microsoft CustomerSource portal.


The download for the full software can also be found on the following links:

- <http://www.mibuso.com/dlinfo.asp?FileID=1495>

Home | Forum | Downloads | Business Directory | Product Directory | How Tos | Blogs | Jobs | Feedback


### Downloads

Name: [Microsoft Dynamics NAV W1 2013](#) 

Author: Microsoft Business Solutions 

Date: 08/10/2012

Size: 590.41 MB

Downloads: 262 

MDS Digest: 4C4798F335029AEB70A5DBE36457B963


On DVD-ROM: No

Archive format: [Z-Zip](#)

Category: Commercial Demos & Trial Versions


Description: Microsoft Dynamics NAV 2013 is a landmark release, building on a strong product foundation to deliver new levels of customer and partner value through faster, more efficient implementations, compelling new and enhanced application functionality, and unparalleled levels of customer choice on how to deploy and access the solution.


**We strongly advise you to use a Download Manager (for example <http://www.freedownloadmanager.org/>) to download this large file.**


Avg. rating: No rating available      Popularity: 


[Be the first to discuss this download in the forum](#)

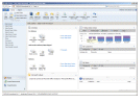
[Find all downloads from this author](#) (661 downloads)

 [SMS this title to a friend](#)

 [Mail this title to a friend](#)

 [report broken link](#)


 **YYZR**



(Click screenshot to enlarge)


- <http://dynamicsuser.net/media/p/312076.aspx>


Previous | Next | View all files | View Slideshow

 **Microsoft Dynamics NAV 2013 W1 - Product DVD (33781)**

Filed under: [W1](#), [Product DVD](#), [NAV 2013](#), [Download](#)

The world wide version (ENU language only) of the product DVD of Microsoft Dynamics NAV 2013 release (build 33781 - RTM).

 Download

posted by Erik P. Ernst  
Mon, Nov 12 2012  


[Add file to favorites](#)

[Add gallery to favs](#)

Downloads: 66  
File size: 644.8MB  
Views: 169

Don't worry, these links are legitimate. Luc (owner of [mibuso.com](http://mibuso.com)) and Erik (owner of [dynamicsuser.net](http://dynamicsuser.net)) are good people whom I've had the pleasure of interacting with.

There are other sites that you can download the software installation files from, which you can try at your own risk.


To be able to modify and create reports in Dynamics NAV, you will also need a copy of Visual Studio 2010 installed on your computer. Fortunately, all you need is the Express version, which is free. The link to download this directly from Microsoft is as follows:

<http://www.microsoft.com/visualstudio/eng/products/visual-studio-express-products>

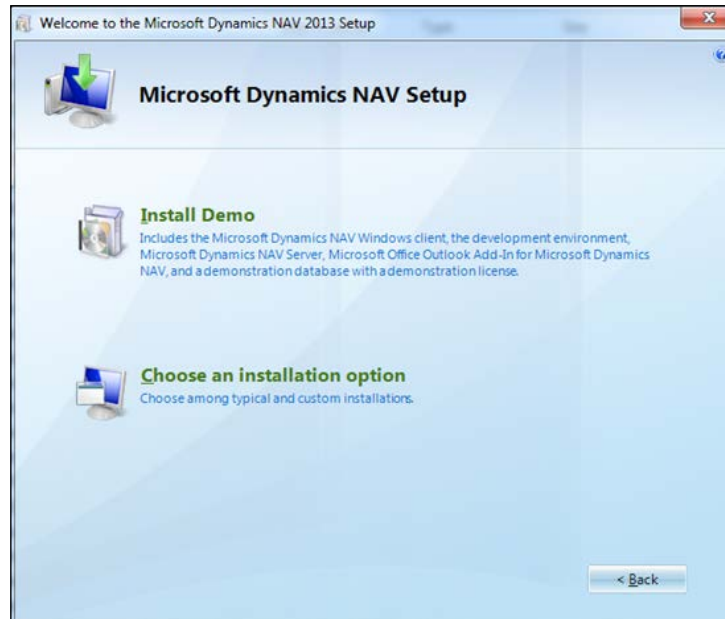
## Installing the software

Once you've extracted the downloaded file, the folder where you extracted the file should look something like this:

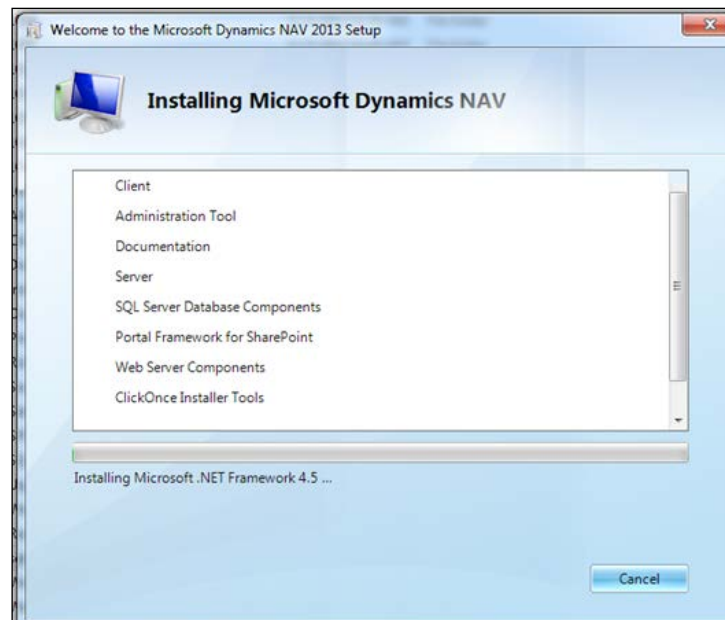
| Name                                    | Date modified      | Type          | Size   |
|---|--------------------|---------------|--------|
| 1030                                    | 11/5/2012 11:10 AM | File folder   |        |
| 1031                                    | 11/5/2012 11:10 AM | File folder   |        |
| 1033                                    | 11/5/2012 11:10 AM | File folder   |        |
| 1034                                    | 11/5/2012 11:10 AM | File folder   |        |
| 1036                                    | 11/5/2012 11:10 AM | File folder   |        |
| 1040                                    | 11/5/2012 11:10 AM | File folder   |        |
| 1043                                    | 11/5/2012 11:10 AM | File folder   |        |
| 1053                                    | 11/5/2012 11:10 AM | File folder   |        |
| ADCS                                    | 11/5/2012 11:10 AM | File folder   |        |
| ClickOnceInstallerTools                 | 11/5/2012 11:10 AM | File folder   |        |
| Documentation                           | 11/5/2012 11:10 AM | File folder   |        |
| Installers                              | 11/5/2012 11:13 AM | File folder   |        |
| Outlook                                 | 11/5/2012 11:13 AM | File folder   |        |
| Prerequisite Components                 | 11/5/2012 11:14 AM | File folder   |        |
| RoleTailoredClient                      | 11/5/2012 11:14 AM | File folder   |        |
| ServiceTier                             | 11/5/2012 11:15 AM | File folder   |        |
| Setup                                   | 11/5/2012 11:15 AM | File folder   |        |
| SPC Server                              | 11/5/2012 11:15 AM | File folder   |        |
| SQLDemoDatabase                         | 11/5/2012 11:15 AM | File folder   |        |
| UpgradeToolKit                          | 11/5/2012 11:15 AM | File folder   |        |
| WebClient                               | 11/5/2012 11:15 AM | File folder   |        |
| ReadMe.htm                              | 5/15/2012 8:29 AM  | HTML Document | 11 KB  |
| setup.exe                               | 9/13/2012 5:52 PM  | Application   | 755 KB |
| WebClientDependencyInstaller.exe        | 9/7/2012 2:26 AM   | Application   | 34 KB  |
| WebClientDependencyInstaller.exe.config | 8/23/2012 11:30 PM | CONFIG File   | 1 KB   |

 If you have any prior versions of Dynamics NAV (or Navision) installed, please make sure you uninstall them before you run the installation.

Click on the `setup.exe` file and run through the installation wizard.



Click on the **Install Demo** option and go grab some coffee. We'll wait for you.



## **Installing Visual Studio Web Developer 2010 Express**

After Dynamics NAV is installed, run the Visual Studio Web Developer 2010 Express installation. Again, this is needed for creating and modifying Dynamics NAV reports.

There's an additional component that you will need to install on your computer in order to properly modify Dynamics NAV reports using the Visual Studio Web Developer Express. You will need to install the free version of Microsoft Visual Studio 2010 Shell (Integrated) Redistributable Package. The link is as follows:

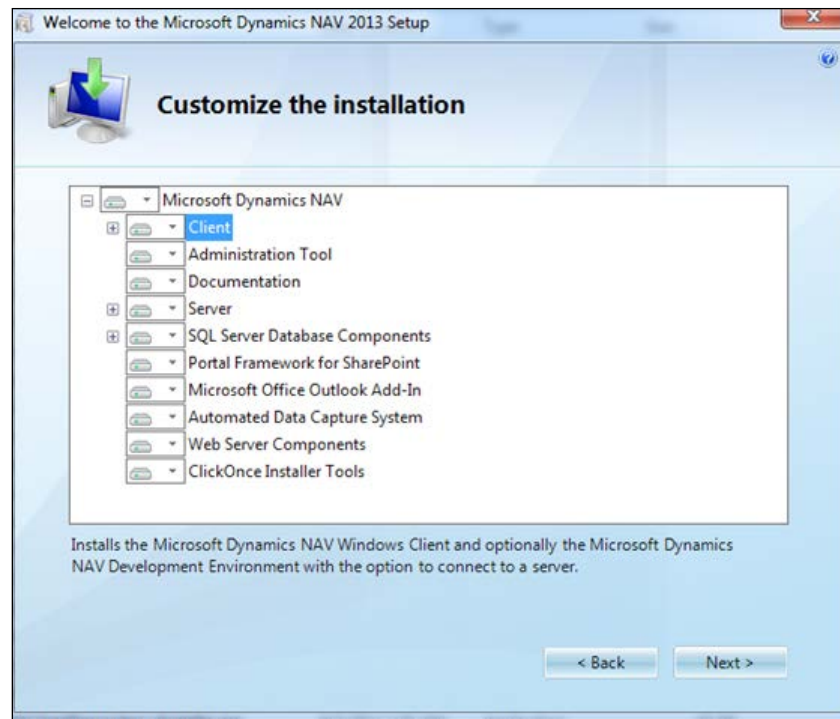
<http://www.microsoft.com/en-us/download/details.aspx?displaylang=en&id=115>

For this book, any additional contents on the installation files will not be needed. Running through the installation wizard will be more than enough for what we need to do.

## **A quick overview of the additional contents of the installation files for Dynamics NAV**

Suppose you're a curious person like myself and you click on the **Choose an installation option** instead; you'll instead be presented with a list of the components that you can install.

Here's a quick rundown of the components that can be installed individually. If you click on an icon, you will get a description of what it is. I will go through these options in simple English.



The components you will see are as follows:

- **Client:** This is the Role Tailored Client, or as Microsoft calls it, the "Windows Client". If you wish to deploy this for your company, you will at least need to install this component.
- **Administration Tools:** This is the snap-in console that allows you to configure the Windows services related to Dynamics NAV. This is an interface that allows you to be able to, for example, change the port for a Dynamics NAV connection without having to mess with DOS prompts or the registry.



- **Documentation:** If you want the documentation or **Help** to be available for the NAV client, you should probably install this.
- **Server:** Dynamics NAV is a three-tier system. The middle tier is where the business logic is executed. So any device, web service, or client software will use the middle tier to get and write data into DynamicsNAV.
- **SQL Server Database Components:** Choose this option for installing SQL Server Express on your computer. Do this only if you have a version below SQL Server 2008 installed on your computer. Microsoft recommends that you should use at least SQL Server 2008 R2. Dynamics NAV 2013 will only run on a SQL Server database. Sorry!
- **Portal Framework for SharePoint:** Dynamics NAV is integrated with SharePoint. This will allow you to build SharePoint web applications in Dynamics NAV. For this to work, you will need SharePoint 2010 installed.
- **Microsoft Office Outlook Add-in:** Believe it or not, Dynamics NAV has a built-in CRM solution. This option allows you to install a component to Microsoft Outlook to synchronize contacts, tasks, and calendar items with Dynamics NAV.
- **Automated Data Capture System:** ADCS is the acronym that you'll find if you do a search online. This option allows the warehouse staff to pick and put away inventory to/from the warehouse bins in real time using handheld devices.
- **Web Server Components:** Dynamics NAV 2013 is the first version in NAV history to have an out of the box web client. This option allows you to use a web client instead of the Windows client.
- **ClickOnce Installer Tools:** If you're an in-house IT guy, you know that installing software on each and every computer is a pain. ClickOnce technology allows you to deploy Dynamics NAV through a web link. You can preconfigure the setup for each user so they can do the installation themselves.

If any of the preceding components catch your eye and you would like additional information on them, Microsoft provides detailed explanations on each of the components and how to deploy them in your organization. This detailed information can be found at:

[http://msdn.microsoft.com/en-us/library/dd301130\(v=nav.70\).aspx](http://msdn.microsoft.com/en-us/library/dd301130(v=nav.70).aspx)

If you want more information on these additional components and how they can benefit your business, I would highly recommend you contact your local Dynamics NAV partner and get them involved. Nothing is more frustrating than installing something and not knowing how it works.

## A look at what is installed

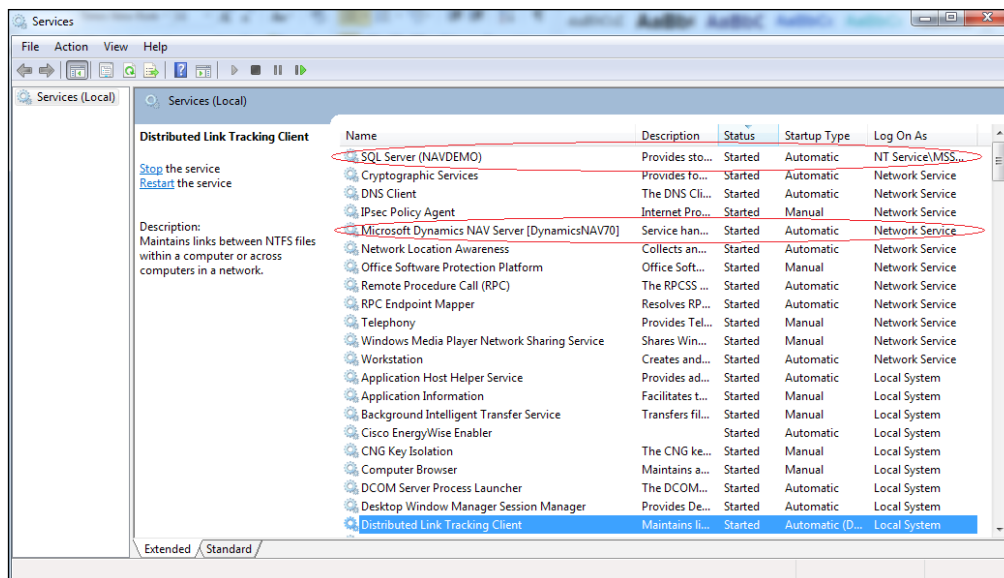
After the installation finishes, if you click on your **Start** icon in Windows, you'll notice a few new icons. They are:

- Microsoft Dynamics NAV 2013 Administration Shell
- Microsoft Dynamics NAV 2013 Development Environment
- Microsoft Dynamics NAV 2013
- Microsoft Dynamics NAV Administration
- Microsoft SQL Server 2012 folder
- Microsoft SQL Report Builder folder

In addition, there will be two services that will be running. They are:

- Microsoft Dynamics NAV Server [DynamicsNAV70]
- SQL Server (NAVDEMO)

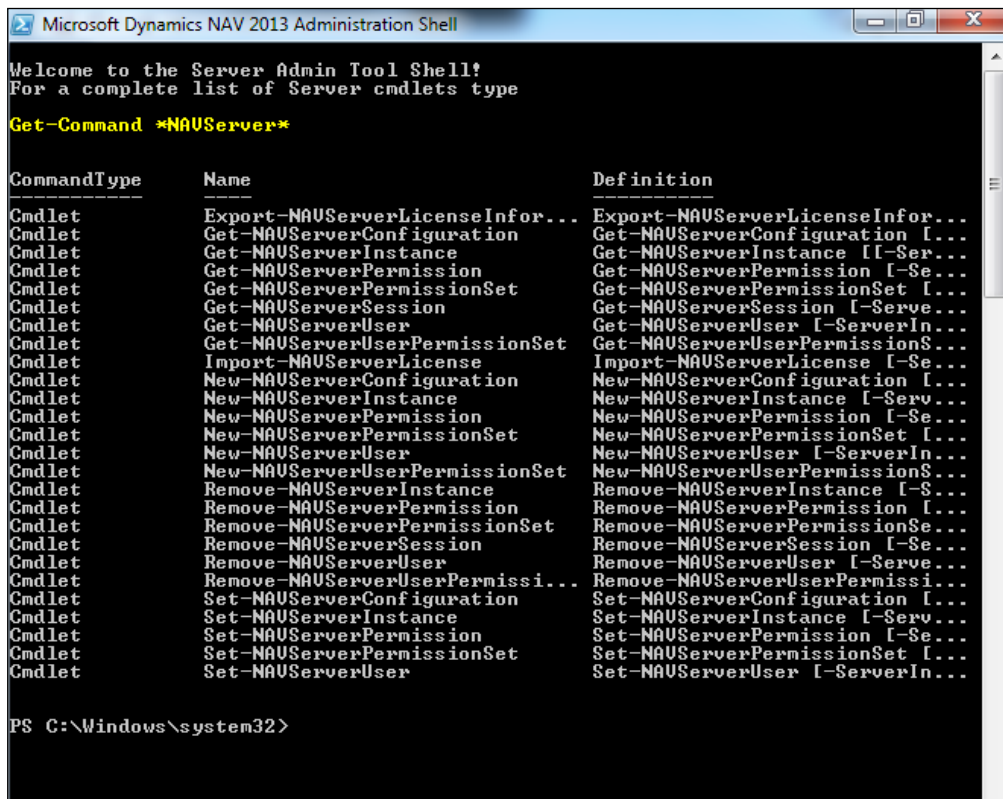
The SQL Server (NAVDEMO) is the service for the SQL Server instance where the demo database resides. **Microsoft Dynamics NAV [DynamicsNAV70]** is the middle tier that the client connects to. Ensure both of these have their status as **Started**.



## Microsoft Dynamics NAV 2013 Administration Shell

The Microsoft Dynamics NAV 2013 Administration Shell allows you to run scripts implemented using PowerShell 2.0. There are predefined commands (called `Cmdlet`) that the user can use right away. The built-in `Cmdlet` allows the administrator to configure and troubleshoot permissions and connection problems on a local or remote computer. Also worth mentioning is that `Cmdlet` should always be run as an administrator.

This tool will come in very handy if you're deploying Dynamics NAV to remote locations or in your own private cloud.



```
Microsoft Dynamics NAV 2013 Administration Shell
Welcome to the Server Admin Tool Shell!
For a complete list of Server cmdlets type
Get-Command *NAUServer*

CommandType      Name                                     Definition
-----
Cmdlet           Export-NAUServerLicenseInfor...        Export-NAUServerLicenseInfor...
Cmdlet           Get-NAUServerConfiguration            Get-NAUServerConfiguration I...
Cmdlet           Get-NAUServerInstance                Get-NAUServerInstance [I-Se...
Cmdlet           Get-NAUServerPermission              Get-NAUServerPermission [Se...
Cmdlet           Get-NAUServerPermissionSet           Get-NAUServerPermissionSet I...
Cmdlet           Get-NAUServerSession                 Get-NAUServerSession [Serve...
Cmdlet           Get-NAUServerUser                    Get-NAUServerUser [I-ServerIn...
Cmdlet           Get-NAUServerUserPermissionSet       Get-NAUServerUserPermissionS...
Cmdlet           Import-NAUServerLicense               Import-NAUServerLicense [I-Se...
Cmdlet           New-NAUServerConfiguration           New-NAUServerConfiguration I...
Cmdlet           New-NAUServerInstance                New-NAUServerInstance [I-Serv...
Cmdlet           New-NAUServerPermission              New-NAUServerPermission [I-Se...
Cmdlet           New-NAUServerPermissionSet           New-NAUServerPermissionSet I...
Cmdlet           New-NAUServerUser                    New-NAUServerUser [I-ServerIn...
Cmdlet           New-NAUServerUserPermissionSet       New-NAUServerUserPermissionS...
Cmdlet           Remove-NAUServerInstance              Remove-NAUServerInstance [I-S...
Cmdlet           Remove-NAUServerPermission           Remove-NAUServerPermission [I...
Cmdlet           Remove-NAUServerPermissionSet        Remove-NAUServerPermissionSe...
Cmdlet           Remove-NAUServerSession              Remove-NAUServerSession [I-Se...
Cmdlet           Remove-NAUServerUser                 Remove-NAUServerUser [I-Serve...
Cmdlet           Remove-NAUServerUserPermissionSet    Remove-NAUServerUserPermissi...
Cmdlet           Set-NAUServerConfiguration           Set-NAUServerConfiguration I...
Cmdlet           Set-NAUServerInstance                Set-NAUServerInstance [I-Serv...
Cmdlet           Set-NAUServerPermission              Set-NAUServerPermission [I-Se...
Cmdlet           Set-NAUServerPermissionSet           Set-NAUServerPermissionSet I...
Cmdlet           Set-NAUServerUser                    Set-NAUServerUser [I-ServerIn...

PS C:\Windows\system32>
```

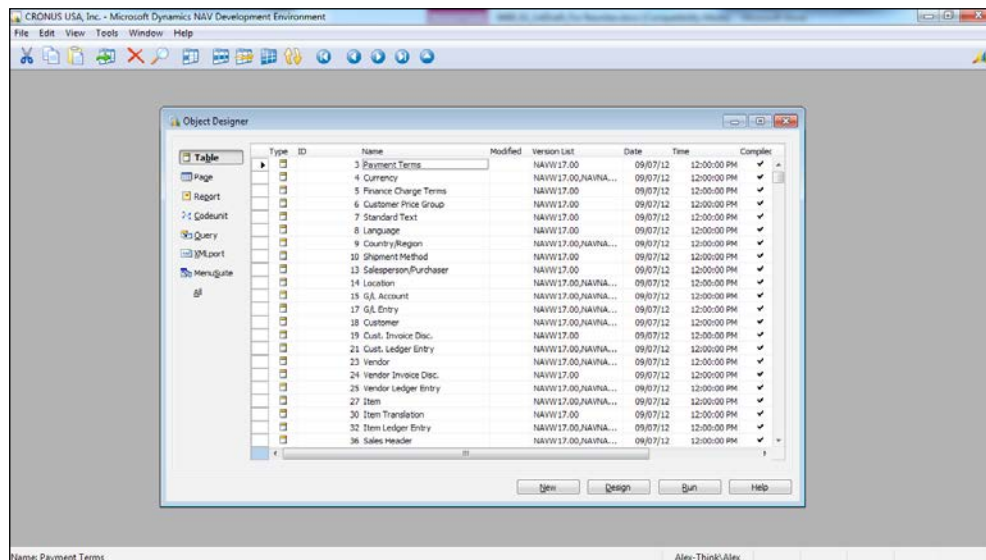
# Microsoft Dynamics NAV 2013 Development Environment

As the name suggests, this is where NAV developers come to create all sorts of wonderful things for Dynamics NAV. This is the main environment where developers work; it is called the **Client/Server Integrated Development Environment (C/SIDE)**. Within C/SIDE, you will be developing using a language called **Client/server Application Language (C/AL)**.

All of the objects are contained within this environment and stored in the SQL Server database, so you do not have to go anywhere else to create or modify applications for the end users. Even the development of report layouts, which uses Visual Studio and not C/SIDE, is tightly integrated and is launched from within the development environment.

The development environment is also where the user can update the license either on the server or for their particular session.

When you start this application, it will look very empty. For Dynamics NAV developers, the majority of their time will be spent here, so having a nice clean space is a very good start.

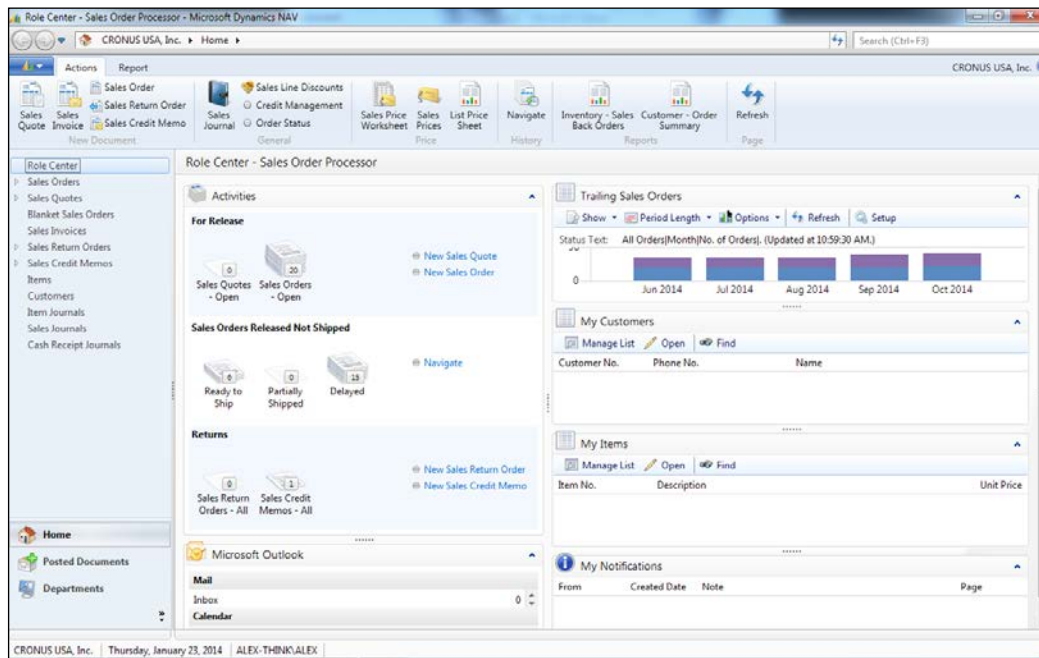


## Microsoft Dynamics NAV 2013 Windows Client

Microsoft Dynamics NAV 2013 Windows Client is the actual client application that the end users will be using to transact their daily operations. As mentioned earlier, the client application is called the **Role Tailored Client (RTC)**, or Windows Client.

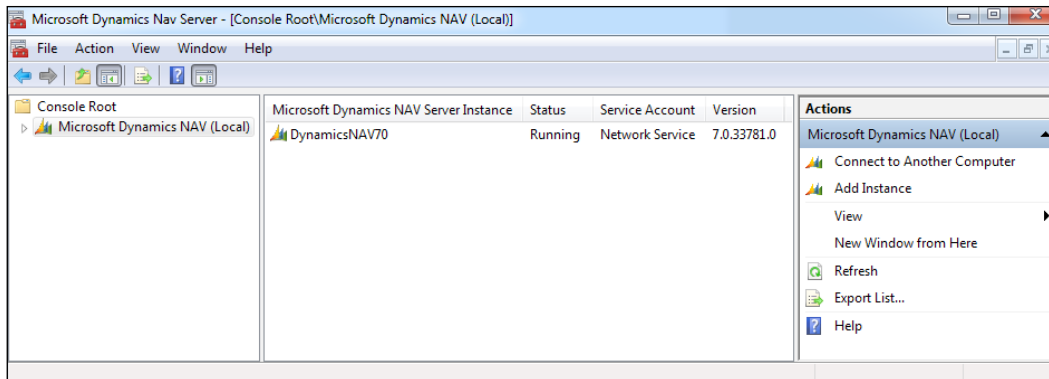
By default, when you start the Windows Client, you will assume the role of a sales-order processor. This is okay, because you can still access any part of the system as long as you have permission to do so.

Any changes we've made in the development environment will be reflected in the client application through the middle tier. If you're using any other interface, such as a mobile interface or a web interface, the changes will be reflected there as well.



## Microsoft Dynamics NAV Administration

Do not confuse this with the previously mentioned Administration Shell. The Microsoft Dynamics NAV Administration program allows you to manage the Dynamics NAV services that are installed both on your local computer and on the server. You can also manage the services without using this program using the command prompt, but that wouldn't be very efficient, or fun.



## The SQL Server 2012 folder

There's a tool in the SQL Server 2012 folder called Import/Export Data. As a general rule, when working with Dynamics NAV or any ERP software out there, do not ever try to import data directly into the SQL tables in your ERP software. The reason is that these import/export programs do not validate against any business logic that's built in place. By doing these imports in an external program, you risk undermining the integrity of the data in your ERP software.

## The SQL Server 2012 report builder

The SQL Server 2012 report builder is installed primarily for the API to upgrade RDLC 2005 for Dynamics NAV 2009 to RDLC 2008 in Dynamics NAV 2013.

Of course, you can use the report builder to create beautiful reports for Dynamics NAV if you do not wish to use the reporting tool within C/SIDE.

The reports that you will be modifying or creating in C/SIDE will automatically be linked to the appropriate objects. However, the reporting in C/SIDE will not be done using the Report Builder application.

## Getting your license

Now that we have all the software installed, you will need proper licensing in order to do some development. There are a couple of different licenses that you, the end user, will have access to.

## **Demo license**

When you install Dynamics NAV, the demonstration license is installed by default. The demonstration license allows you to access every module in Dynamics NAV. However, the areas you're able to develop are severely restricted. The demo license is intended for you to click around and test a few transactions with certain date ranges. It's not really meant for learning development, which is the reason why you're reading this book.

In case you're already familiar with Dynamics NAV object numbers, here are the properties of the Dynamics NAV 2013 demo license from the Microsoft website:

- Start up to two simultaneous client sessions on any platform.
- Create up to two companies.
- Support an unlimited number of web users.
- Run in any supported language.
- Use all application functionalities, including add-on products, local extensions, and customizations in current and previous versions. This means that you can run, but not modify, all object types within the range 1 to 99,999,999.
- Run and modify table 18, 2000000061 and 2000000064 through 2000000200, pages 21 and 22, report 101, and XML ports 99,008,503 and 99,008,510. Each object in Dynamics NAV is assigned an ID, so when we say we can modify table 18, it means we can make modifications to the table with ID 18.
- Run, modify, and create fields 99,990 to 99,999, page 99,998 and 99,999, report 99,999, Query 99,9999, and MenuSuite 90.

The database data restriction is as follows:

- Enter transactions in months other than November, December, January, and February

## **MSDN license**

If you're looking for more development capability, you can also subscribe to MSDN to get a Dynamics NAV license with more capability than the out of the box demo license.

At the time of writing, Microsoft had not released the MSDN license for Dynamics NAV 2013. If you want the MSDN license, check the MSDN site regularly.

The MSDN license for Dynamics NAV is meant for you to learn development in Dynamics NAV; however, you will not be able to use this license to run your company. The license has the same restrictions as the demo license, in that you can only enter transactions in a certain date range.

If you're already familiar with Dynamics NAV object numbers, here are the properties of the MSDN licensing:

- Table 18 can be modified. Fields 99,990 through 99,999 can be inserted into table 18.
- Pages 21 and 22 can be modified. Forms 99,998 and 99,999 can be inserted.
- Report 101 can be modified. Report 99,999 can be inserted.
- MenuSuite 90 can be inserted.
- XML Port 99,999 can be inserted.
- Create new objects in the object range 123,456,700 through 123,456,799.

The restriction on the database data is as follows:

- Enter transactions in months other than November, December, January, and February

## **A full On-Premise license**

The pricing for an MSDN subscription and a full user license is comparable. In fact, buying a full Dynamics NAV user license is cheaper than an MSDN subscription.

If buying Dynamics NAV is what your company is going to do anyway, it may be better to purchase the Starter Pack without any additional users. This will allow you to use the license and the database that you will use when Dynamics NAV is implemented for your company. If you choose to buy the full license, you will be able to follow the majority of exercises in this book. However, without buying the developer license, you will not be able to modify the coding that's covered in the later chapters of the book.

You will also not get to use the more advanced modules, such as manufacturing or warehouse management. However, this book will not be getting into these advanced granules.

The On-Premise license will allow you to modify everything except the following:

- Code units
- Code behind the pages
- Code behind the tables

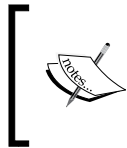


In addition, the On-Premise license comes with the following:

- 10 custom tables you can create
- 100 custom pages you can create
- 100 custom reports you can create

In order to buy the On-Premise license, you will need to find your local Dynamics NAV Solution Provider. Thankfully, Microsoft provides a directory for partners that provide service and software, as follows:

<http://www.microsoft.com/en-us/dynamics/partner-information.aspx>



When you go on this site, make sure the company also provides services for Dynamics NAV. I would caution you against buying Dynamics NAV from a company that does not provide the service for it. Finding the right partner is a science in itself.

## The cloud license

Part of the big push with the release of Dynamics NAV 2013 is the ability to move the product into the cloud environment. There are a few companies that have cloud offerings.

The company that has graciously allowed us to use their environment for you to log in and follow along with the book is called Data Resolution, Inc. Coincidentally, they were named Hosting Partner of the Year by Microsoft in 2012.

You can get a 30-day free trial at <http://navappdev.erpclouds.com/>. Thirty days should be more than enough for you to follow through the chapters in this book. Unfortunately, Data Resolution does not offer a longer free trial.

The licensing available when you sign up would be the same as buying the On-Premise Dynamics NAV license with full development capabilities for your company.

Again, you will not get to use the more advanced modules, such as manufacturing or warehouse management. This book will not be getting into these advanced granules; if you're interested in these by the end of this book, you can ask your local Dynamics NAV solution provider to discuss using this solution for your company.

If you wish to continue following through the book and continue doing examples after the 30-day trial, you will need to sign up. So, use your time with this book wisely.

## **Summary**

In this chapter, we've gone over finding your copy of Dynamics NAV and the installation process. There are localized versions of Dynamics NAV databases, for example, the US Dynamics NAV database will be a little different than the Indian Dynamics NAV database. For the purpose of learning development, which is described in this book, it does not matter which database you use.

To maximize the experience you get out of this book, I would highly recommend you utilize the 30-day free trial on the cloud. Even if you need more than 30 days, the cost involved in that additional 30 days would be significantly lower than if you were to get a MSDN subscription or buy the On-Premise Starter Pack.

Getting the software itself is the easiest part. Finding the right license that you will need is actually more of a challenge.



# 2

## Getting Familiar with Dynamics NAV 2013

*"You can learn new things at any time in your life if you're willing to be a beginner. If you actually learn to like being a beginner, the whole world opens up to you."  
- Barbara Sher*

After getting the software installed with the right licensing in place, it's time to explore the software. One of the key productivity enhancers in Dynamics NAV is the consistent user interface since its first release in Version 1.0. The user interface allows any user to handle work from other departments, if there's ever a need, without having to learn how to navigate within the particular module.

In addition to Windows Client, the developer will need to get familiar with the Development Environment as well. As mentioned in the previous chapter, the Development Environment is the place where the developer will create the applications for the users to interact with.

In this chapter, we will dive into navigating around both the Windows Client interface as well as the Development Environment's interface. It's important for the Dynamics NAV developer to get accustomed to both areas. One of the most common mistakes a developer can make is not learning how the standard system works. By not understanding how the standard system or the out-of-the-box functionality works, there's a good possibility that you will create applications and modifications that are not really necessary.

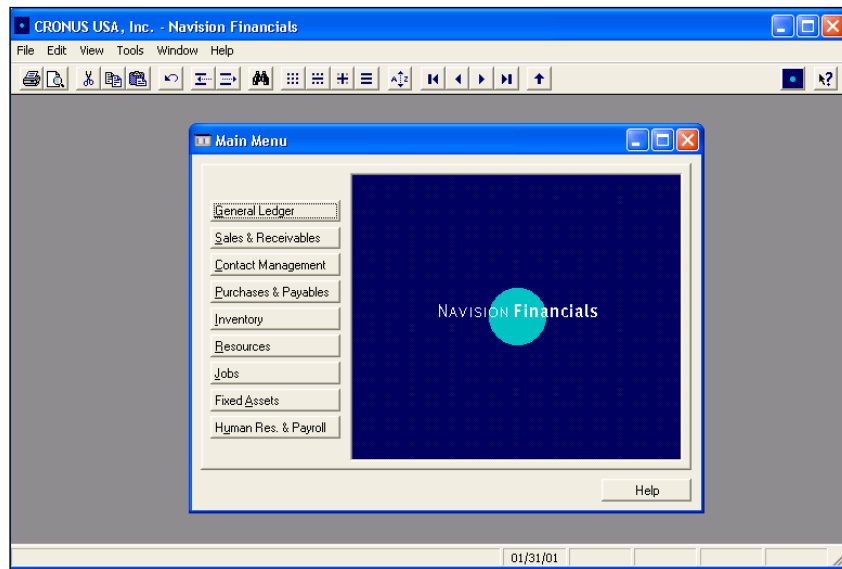
In Windows Client, we'll be looking through the design concepts for the pages you'll be accessing. We'll also go through how to navigate between the different modules even if you're logged in with a specific role.

In the Development Environment, we'll be looking at how to access the application objects and how to run and design these objects without having to manually go to the RTC.

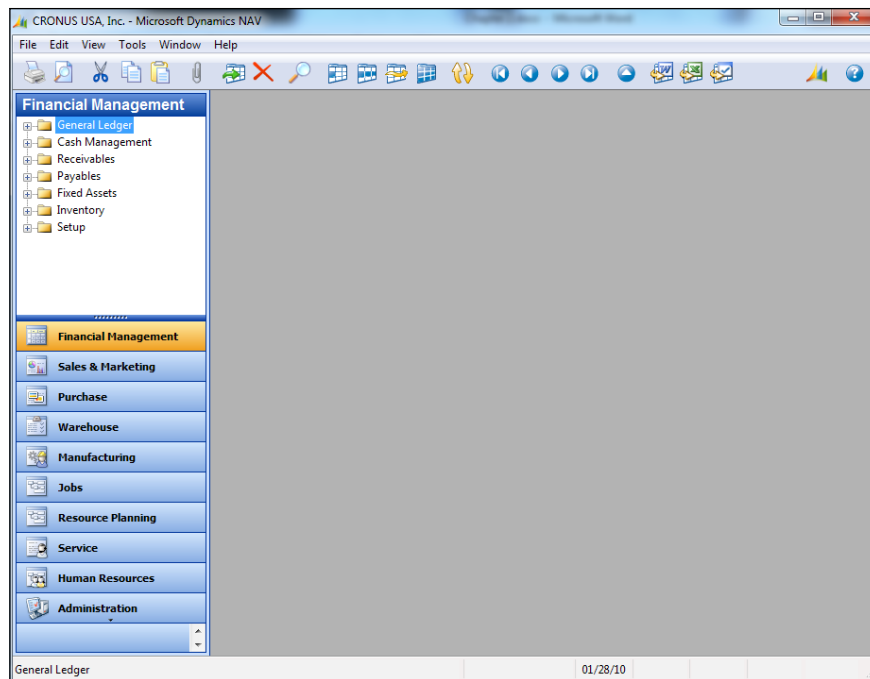
## But first, a little history

Prior to Dynamics NAV 2009, the Development Environment was actually the primary end user interface before Microsoft revamped the user interface that we now call the RTC.

One of the greatest technological breakthroughs with the original Navision (the name before it was called Dynamics NAV) was that the application-programming objects, the user interface, and the database all resided together in one file! Back in the late 1990s and early 2000s, no other software came close to having a design as efficient as this. This was the main menu for Navision Financials Version 2.0:



We're now more than a decade past 2000, and technology has changed quite a bit. Dynamics NAV has kept up to date with the latest technology that has the best impact for businesses. However, most of these improvements and updates are in the backend. This is an important reason why Dynamics NAV has never faded away. There were a couple of user interface improvements that largely look and feel very much the same. This is the main menu for Dynamics NAV 5.0:

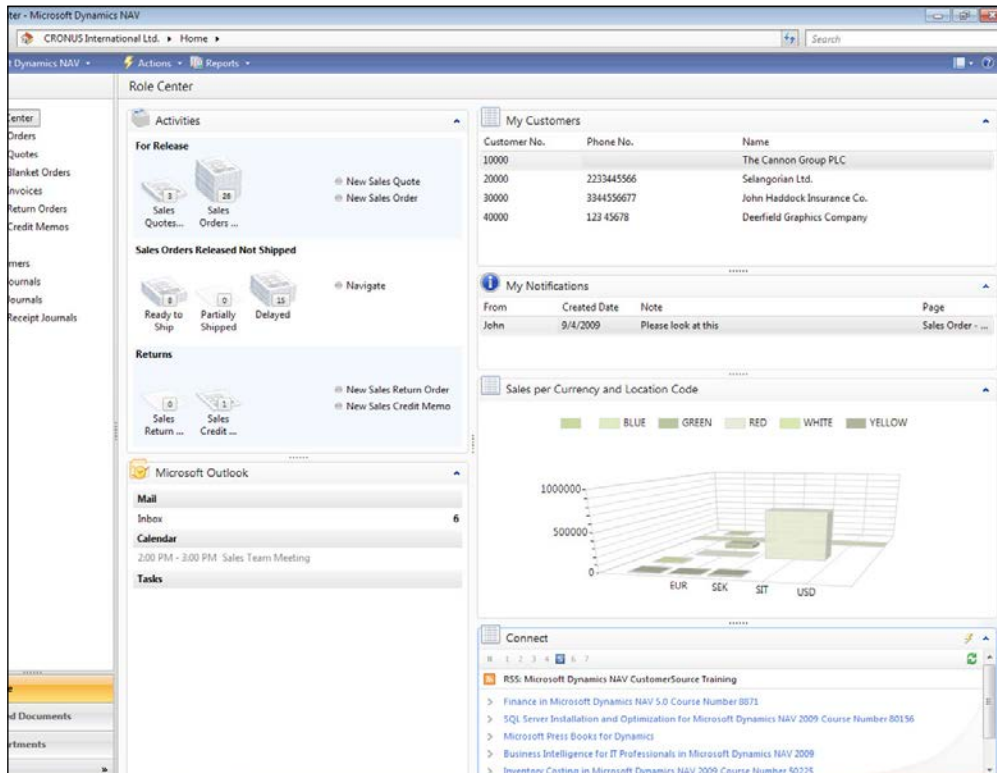


Then something happened. With the rise of a company called Apple, people started paying more attention to the aesthetics and the overall interface of the technology they were using. People demanded not just powerful software with a strong backend, but they wanted an elegant design with a simple and intuitive user interface as well.

Because of this shift in user perception, what was once the greatest innovation in accounting software since Sliced Bread, had become, not obsolete, but dated.

When you pit the old interface (called Classic Client) against some of the newer applications, even though the backend was light years better, the Classic Client was the ugly one. And we all know somebody who has made a terrible decision based only on looks and not really on what's inside.

So when NAV 2009 was introduced, they released the RTC, which is the interface you see when you install Dynamics NAV for end users. NAV 2009 was unique in that it allowed both Classic Client and RTC to coexist. This was largely to appease the existing NAV gurus and users that did not want to learn the new interface. This is what NAV 2009 in the RTC looked like:



At first glance, NAV 2009 and NAV 2013 in the RTC environment do not look too different. You will have to believe me when I say that there were significant user interface and usability changes. I can list out these changes, but if you're not already familiar with Dynamics NAV (or Navision), I'll most likely put you to sleep. If you still want to know for the sake of knowing, you can read the improvements here:

[http://msdn.microsoft.com/en-us/library/hh173994 \(v=nav.70\) .aspx](http://msdn.microsoft.com/en-us/library/hh173994 (v=nav.70) .aspx)

That grace period expired when NAV 2013 was released and the Classic Client user interface was completely removed. Microsoft basically renamed the Classic Client as the Development Environment. For the foreseeable future, it looks like the Development Environment and the Windows Client environment will remain separated.

## The Windows Client (WC) interface

As mentioned in the beginning of this chapter, one of the key user interface design considerations for Dynamics NAV is consistency. The consistent user interface is designed to make it easier for existing users to teach new hires how to use Dynamics NAV. This will allow the company to gain productivity from new hires more efficiently.

Additionally, the design concepts of Dynamics NAV fit in with the "look and feel" of other Microsoft products. Users familiar with using Windows will find it comfortable to use Dynamics NAV.

Microsoft is serious about this consistency, and even published a guideline for third-party developers that wish to create add-ons for Dynamics NAV. So even if you purchase any add-ons along with your solution, you can safely assume that the user will not notice that they're not using base Dynamics NAV. To learn a little more about the User Experience Guideline, go to the following site:

[http://msdn.microsoft.com/en-us/library/jj128065\(v=nav.70\).aspx](http://msdn.microsoft.com/en-us/library/jj128065(v=nav.70).aspx)

Note that this is just a simple guideline and not the stringent one Dynamics NAV add-on developers have to adhere to. We will not be getting into the terminology within the Microsoft standards too much. There are basically three types of pages that you will need to look out for that will be relevant to what we are doing:

- List
- Card
- Document

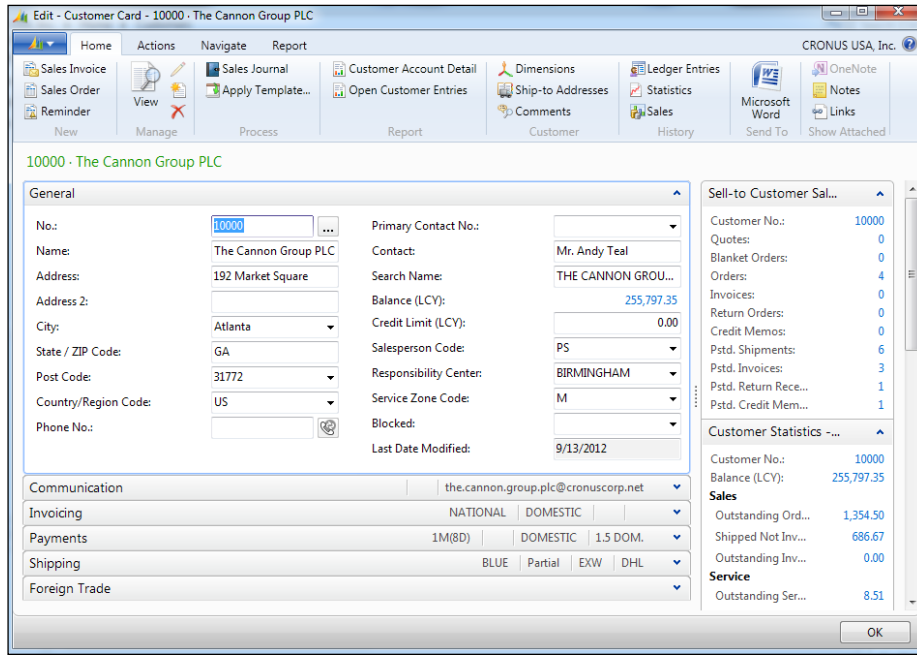
I'm going to assume that you know what a role center is. You'll have the urge to make whatever modifications you're working on look like standard Dynamics NAV (remember we spoke about the ease of use and consistency in the beginning?).

We will learn about the terms for each part as we go along, but if you want to be cool and learn all the terminology beforehand, here's the link to all sorts of different page types in Dynamics NAV:

[http://msdn.microsoft.com/en-us/library/jj651618\(v=nav.70\).aspx](http://msdn.microsoft.com/en-us/library/jj651618(v=nav.70).aspx)

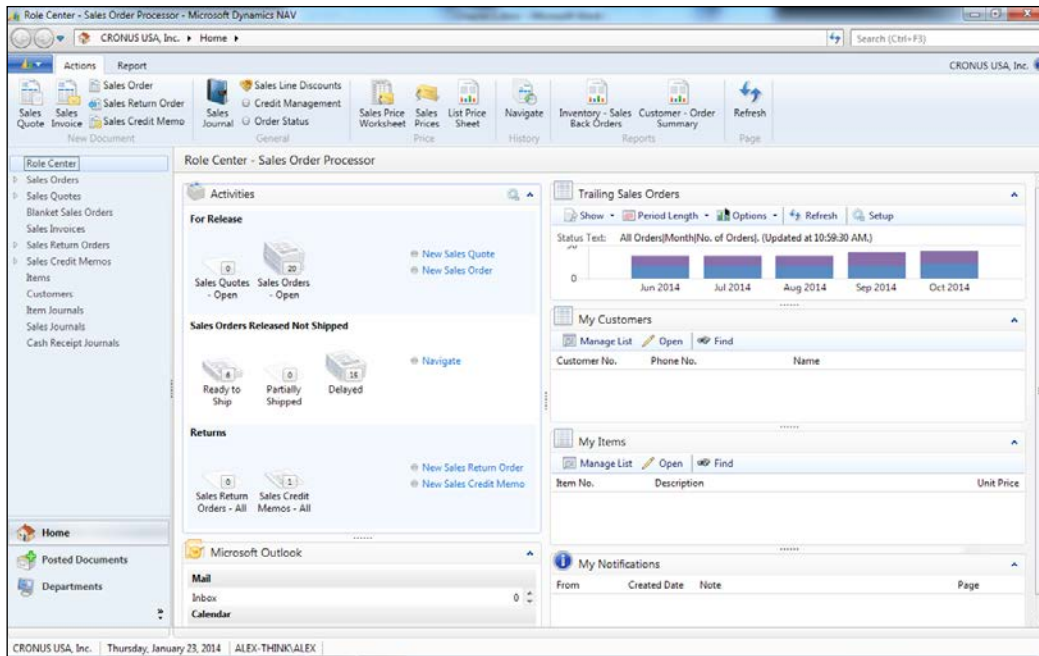


We're also going to assume that when you see a standard Dynamics NAV page similar to the following screenshot:



## Exploring the role center page

When you first start Dynamics NAV, you will by default be logged in with the **Sales Order Processor** role, as shown in the following screenshot:

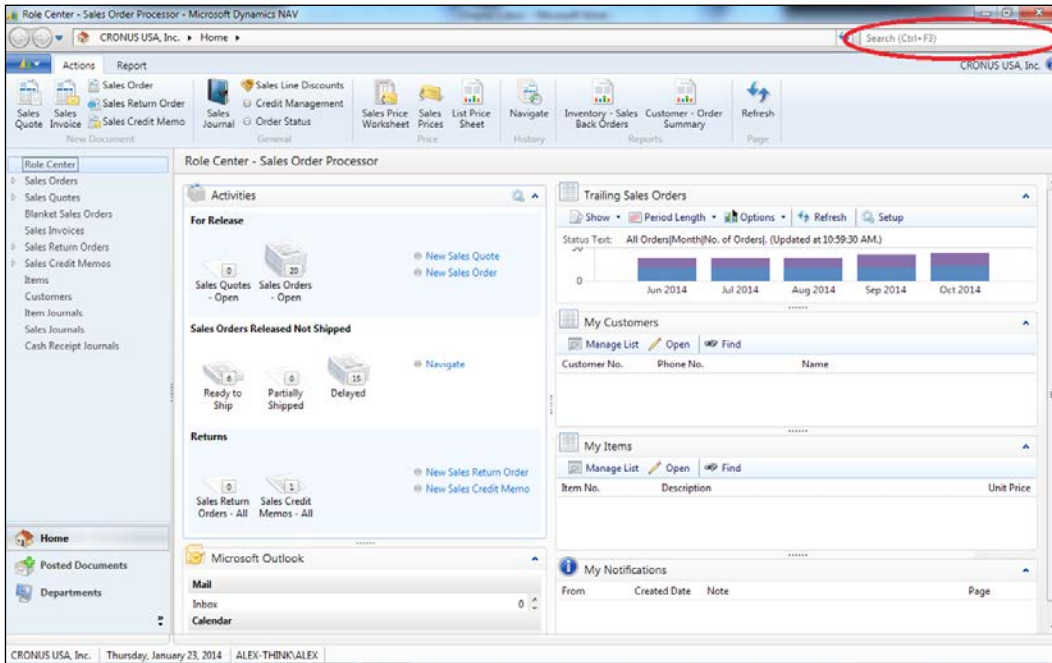


You may or may not be the sales order processor in the company; this is just the default that the administrator can change. Regardless of which role you're logging in as, the role center will have a similar layout. The concept of the role center is to allow the users to get information on what they need to do, when they need to do it, and how they're going to do it without having to click through a ton of screens.

There are a couple of parts of the role center that are critical for the NAV developer and will allow us to move between modules if necessary.

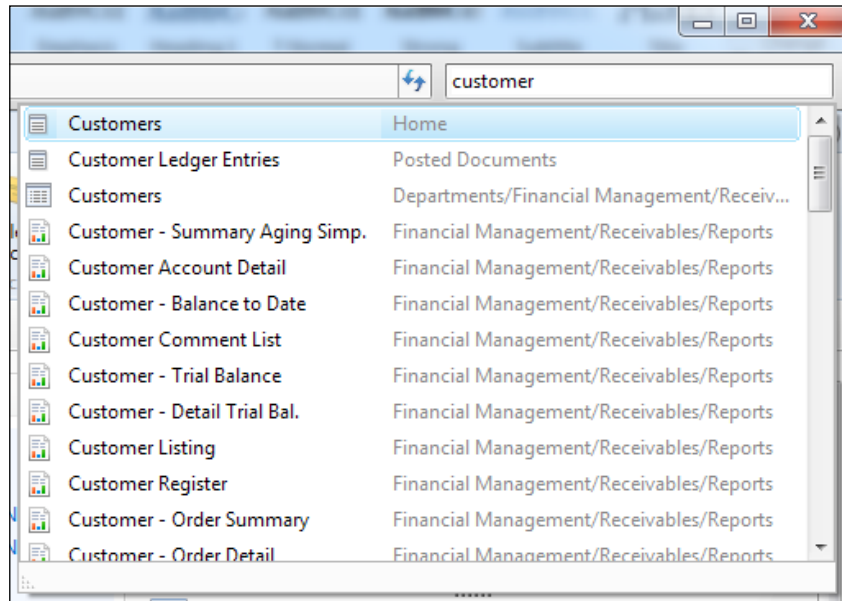
## Page search

One of the most important parts of the role center is the search. It's conveniently located at the top-right portion of the role center.



There are thousands of objects in Dynamics NAV, and it's sometimes hard to know exactly where a particular report or page is located. If you know the name of the report or page that a developer included on the MenuSuite, you can type that in the search; not only will it find it for you, but it'll tell you exactly how to access it. This is one of the most useful tools for Dynamics NAV consultants and developers addressing support issues.

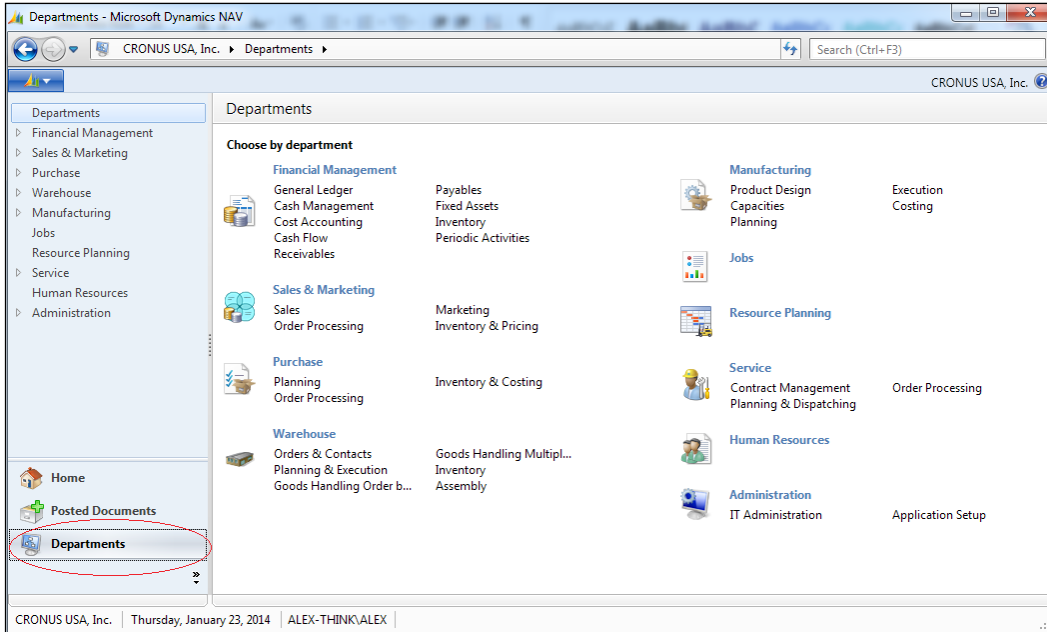
For example, if you wanted to search for a report or page that has the word "customer" in it, just type in `customer` and a list will display showing you all of the options available with the word "customer" in it.



## Accessing other functional areas

You may or may not be a sales order processor in the organization you're working for, but we also know you're a developer, and you will need access to every module in Dynamics NAV because you need to customize, administer, and support these modules.

Fortunately, there's a place for us to see everything. This is the other important area a Dynamics NAV developer will use quite often.

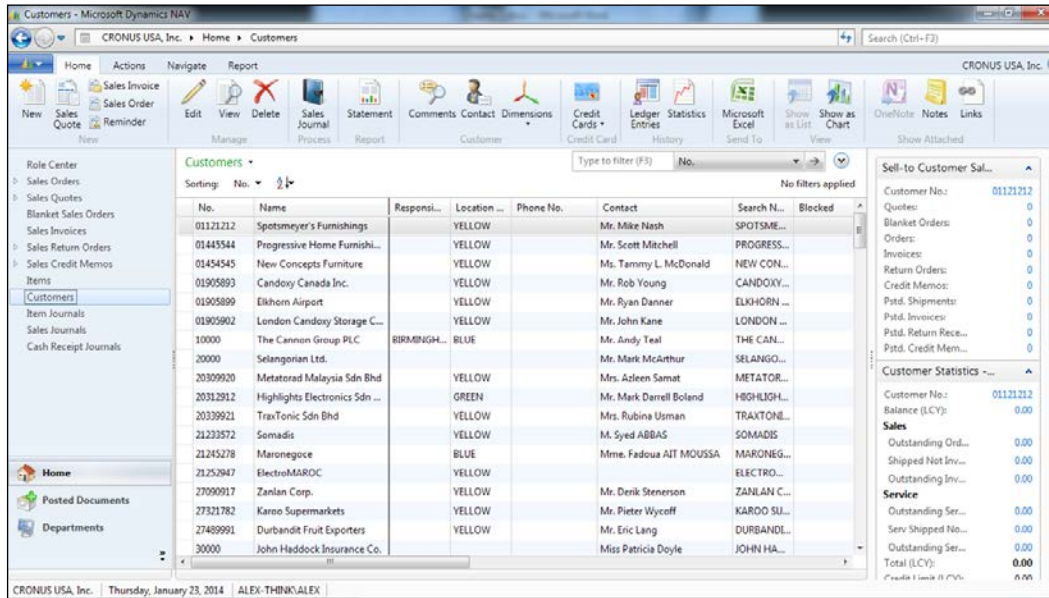


The **Departments** screen allows you to have access to every module in Dynamics NAV that you have license for. This is why it's not that important for the Dynamics NAV developer to be assigned a specific role, because the developer can see everything.

Oh, by the way, all of the roles assigned to users have the **Departments** screen.

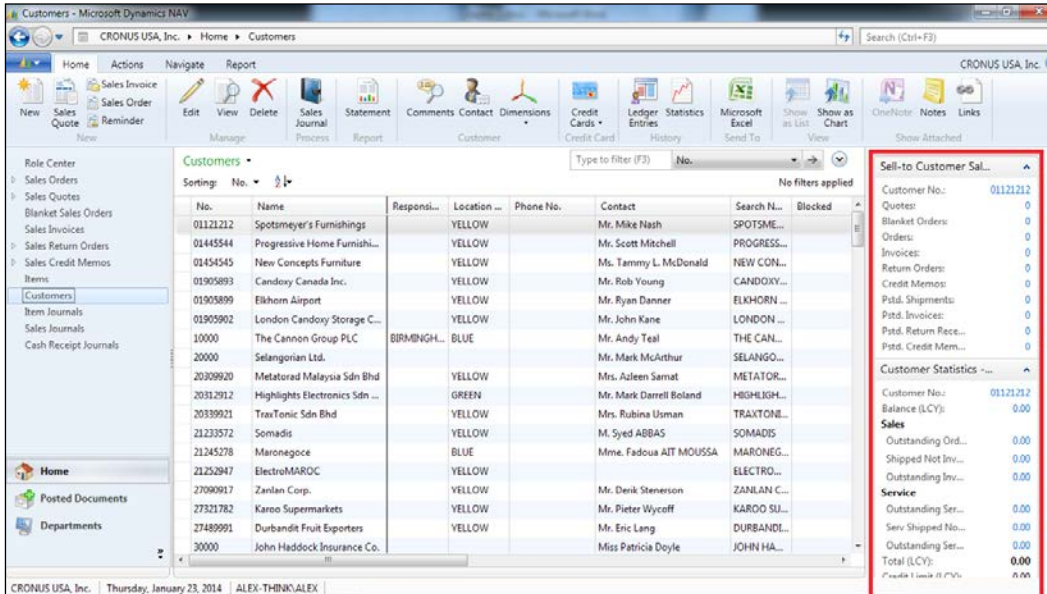
## Exploring the list page

The first type of page you'll see in Dynamics NAV is the list page. You can see this by clicking on the **Customers** tab shown here:



The list page displays a number of records—as many as the screen allows—in a particular table. The list page is typically used for searching for a particular record that you want to work with.

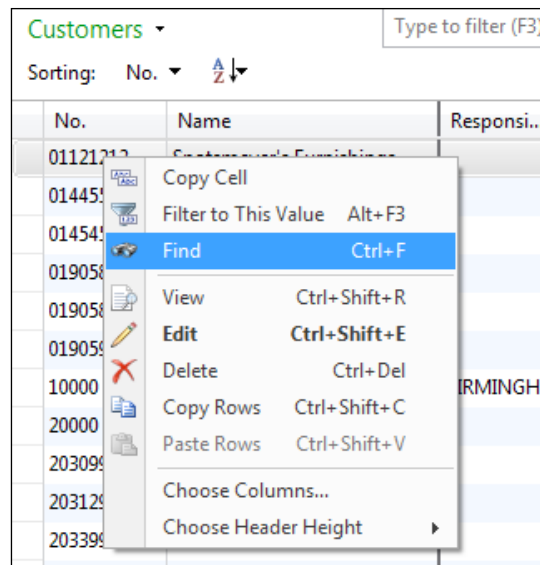
On most list pages, if there is relevant data associated with the particular record, you'll find it to the right of the list page. These "frames" of information are called FactBoxes.



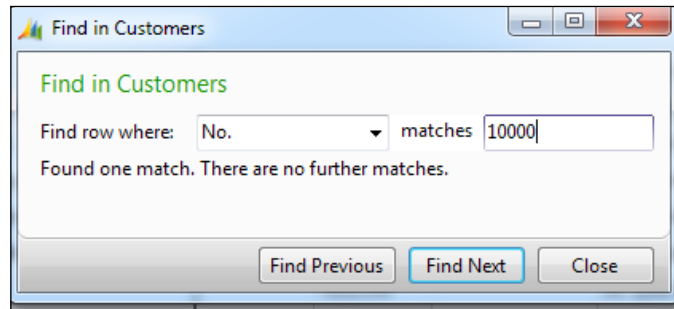
FactBoxes allow the user quick access to the information without having to bring up any additional icon. The nice thing about FactBoxes is that they are customizable and reusable if you're creating a new page related to, in this case, the customer table.

Typically, numbers are shown in the FactBoxes. If you want further details on, for instance, the outstanding sales orders for a particular customer, all you need to do is click on that number and a new list page will be displayed showing you how that number is derived.

Go ahead and find customer **10000, The Cannon Group PLC**. You can bring up the **Find** screen by right-clicking on the **No.** column and clicking on **Find** or pushing **Ctrl + F**.



Enter what you want to find and click on **Find Next**.



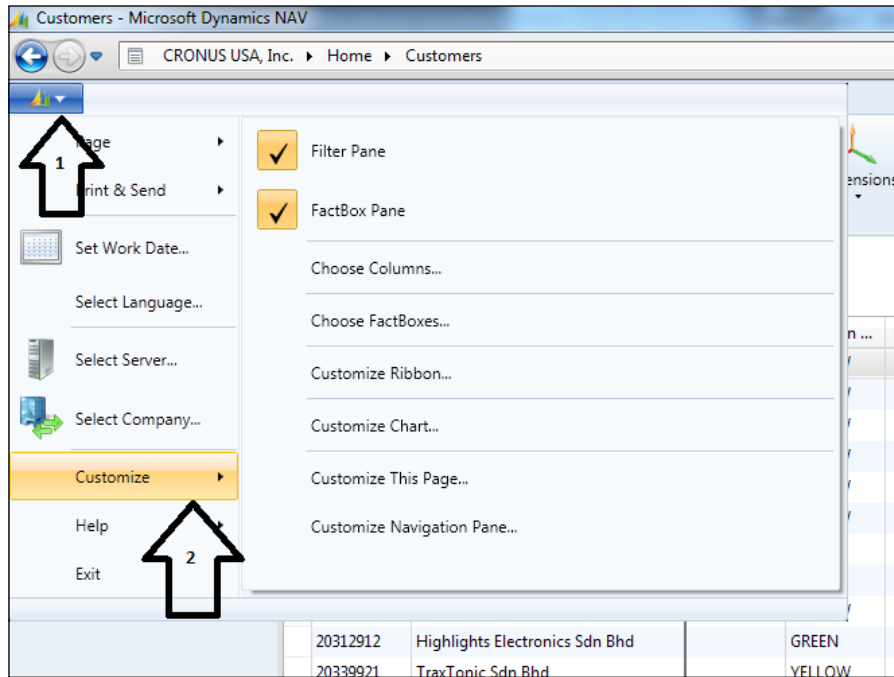
Once you've found customer 10000, you can double-click on it or press the *Enter* key to bring up the card page.



## Personalizing the list page

On most list pages that you'll encounter, there will most likely be more fields than what's displayed on the screen. In addition, there will most likely be more of those little information boxes to the right of the list screen (called FactBoxes).

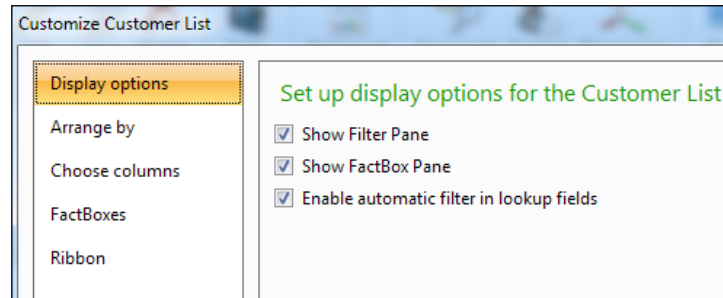
To access the customization menu, click on the icons as shown in the following screenshot:



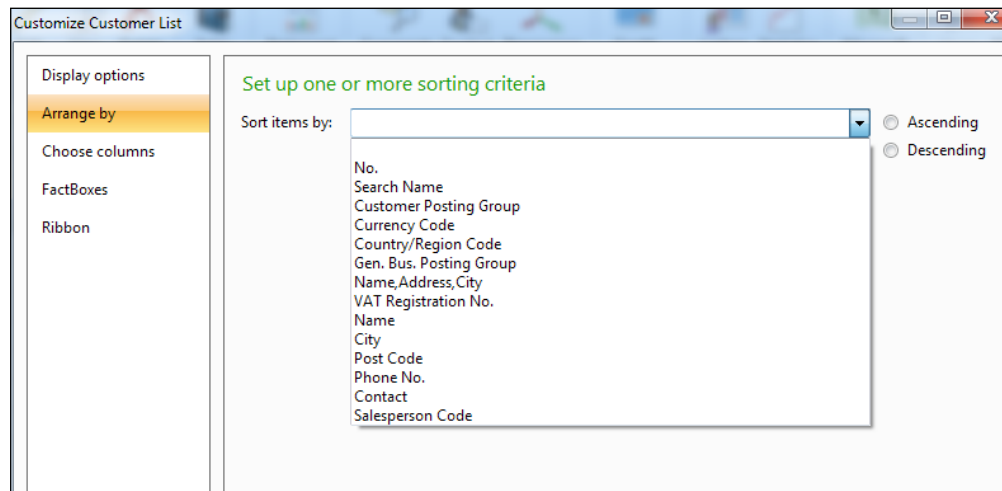
From this screen, you can customize the columns, FactBoxes to the right, and the Ribbons on the top. Go ahead and click on **Customize This Page...** to bring up the customization options.

The customization options that you will see are as follows:

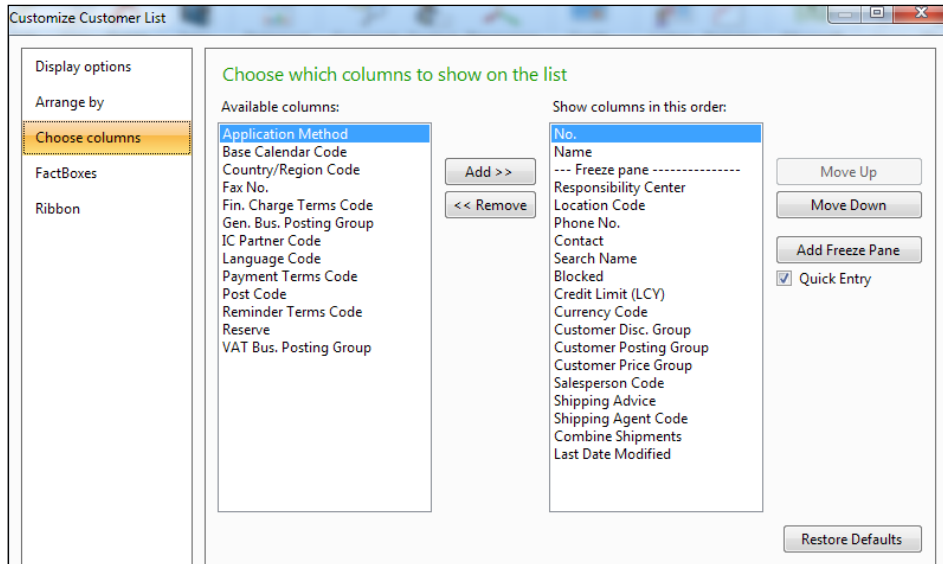
- **Display options:** This screen allows you to control the behavior of the list page. You can choose to hide or show the Filter Pane and the FactBoxes. The **Enable automatic filter in lookup fields** option allows you to toggle the autocomplete feature.



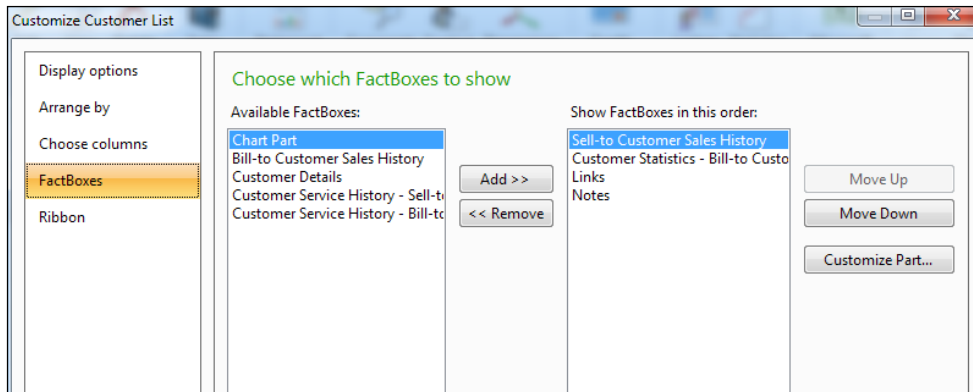
- **Arrange by:** This allows you to sort the list page by the predefined keys that are set for this table. You can sort the list screen based on the ascending or descending order. Unlike Excel, you will not be able to sort on every field; it has to be predefined by the NAV developer in the table keys.



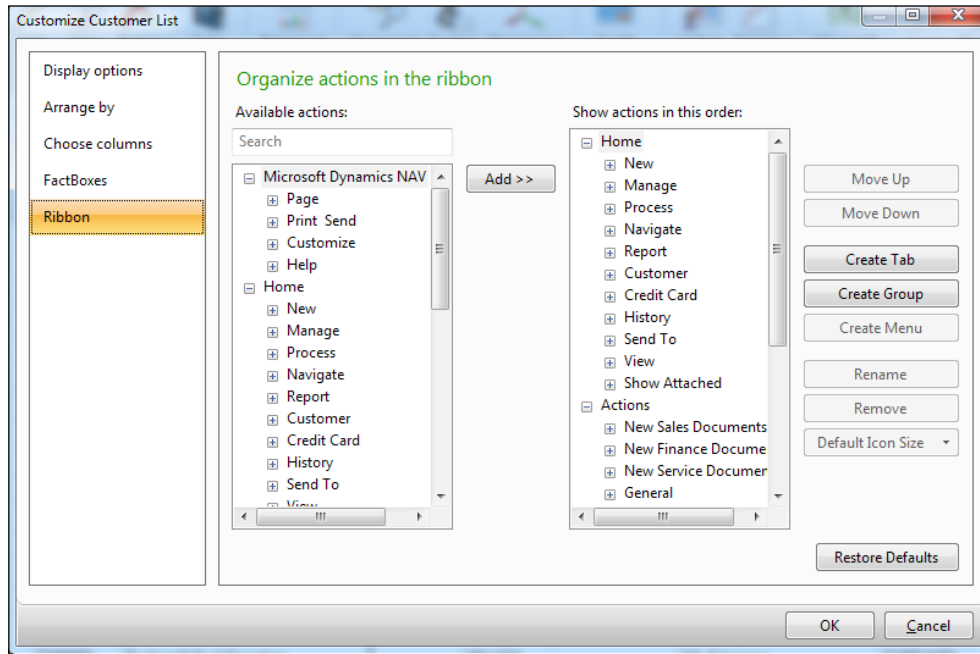
- **Choose columns:** Not every field is displayed by default. In this screen, you can choose what columns you want to display or hide.



- **FactBoxes:** This screen is similar to choosing the columns; there are additional FactBoxes with the list page that are not being displayed. You can show and hide the FactBoxes according to their relevance to their respective job tasks.



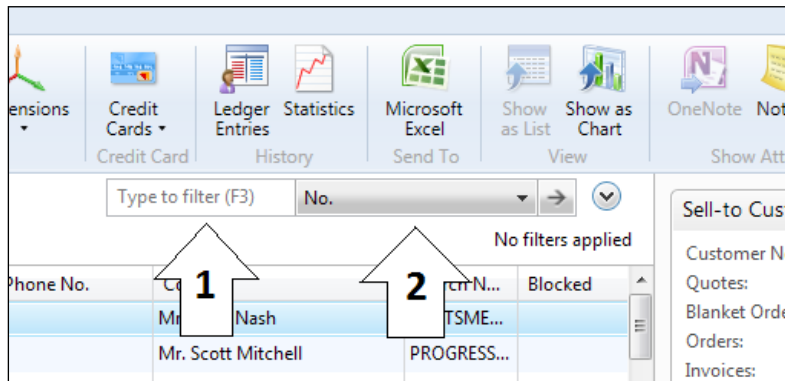
- **Ribbon:** Ribbons are a quick way to access the functions that are associated with the list page. You can edit your ribbon and make the functions you run most often a part of, for instance, the **Home** tab.



Once you've made your changes, as soon as you click on **OK**, the screen will be refreshed with the customizations you've made. This customization will only be applied to you, so any other users that log in to Dynamics NAV using their login credentials will not be affected. This means that the other users can also customize Dynamics NAV to their liking.

## Searching your data based on filters

On any of the list-type pages, you will be able to filter a specific value to generate a specific list of data. The filter option is usually located on the top-right corner of the screen.



The area that the first arrow is pointing to in the preceding screenshot is where the user can click to select the field to filter. Any field that's displayed on the page can be selected for filtering.

The area that the second arrow is pointing to is where the user can enter the filtering criteria. The values that are entered can be specific or wild cards. The value is also dependent on the data type, so this means that if you're filtering on the date value, you must enter data in the date format; if you're filtering an integer value, you will not be able to enter decimals. Here's the format that you can enter filters in:

| Enter   | This includes                  |
|---------|--------------------------------|
| 10000   | Everything equal to 10000      |
| >10000  | Greater than 10000             |
| >=10000 | Greater than or equal to 10000 |
| 10000.. | Greater than or equal to 10000 |
| <10000  | Less than 10000                |

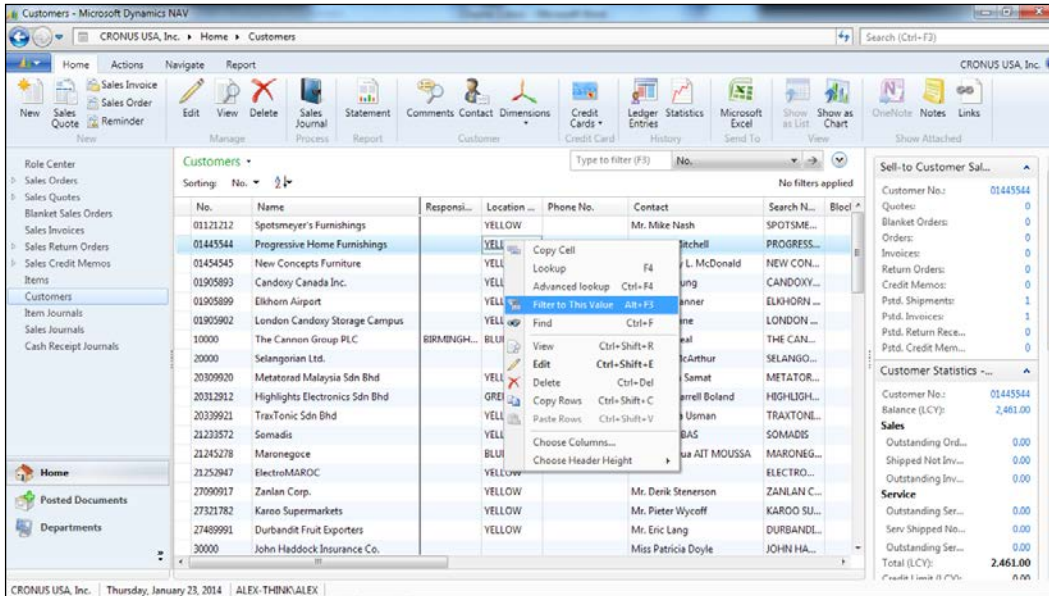
---

| <b>Enter</b>     | <b>This includes</b>  |
|------------------|---|
| <=10000          | Less than or equal to 10000   |
| ..10000          | Everything up to and including 10000  |
| <>10000          | Not equal to 10000  |
| 10000..40000     | From 10000 to 40000   |
| <40000&>10000    | Less than 40000, and greater than 10000                                     |
| 10000 40000      | 10000 and 40000 but nothing in between (an OR statement)                    |
| *00              | Ending in 00 (10000, 1100, 12000, and so on)                                |
| 10000..20000&*00 | From 10000 to 20000 that end in 00 (& is an AND statement)                  |
| P8               | The eighth accounting period in the fiscal year (in the <b>Date</b> field)  |
| *Co*             | Text items that contain Co  |
| Hans?n           | For one unknown character, for example, text items such as Hansen or Hanson |

---

Put your cursor on the green box and type in 10000 . . 40000 and press *Enter* to filter the **No.** field. You'll see that the values displayed are all between 10000 and 40000. To clear the filter, just remove the value and press *Enter*.

In addition to using the filtering box for setting filters, you can also set filters to a specific value that's displayed on the screen. While you're on the **Customers** list, right-click on the value **YELLOW** on the **Location Code** column and then click on **Filter to This Value**. This will also filter this list to, in our case, all the **YELLOW** locations.



## Exploring the card page

The card page gives you greater detail on a particular record. The card page is mainly used to insert, delete, or modify a record. This is just as with the list page described previously; on the right side of the card you'll see the FactBoxes related to this particular record. You can click on the numbers in the FactBoxes to bring up the list page to see how the number was derived.

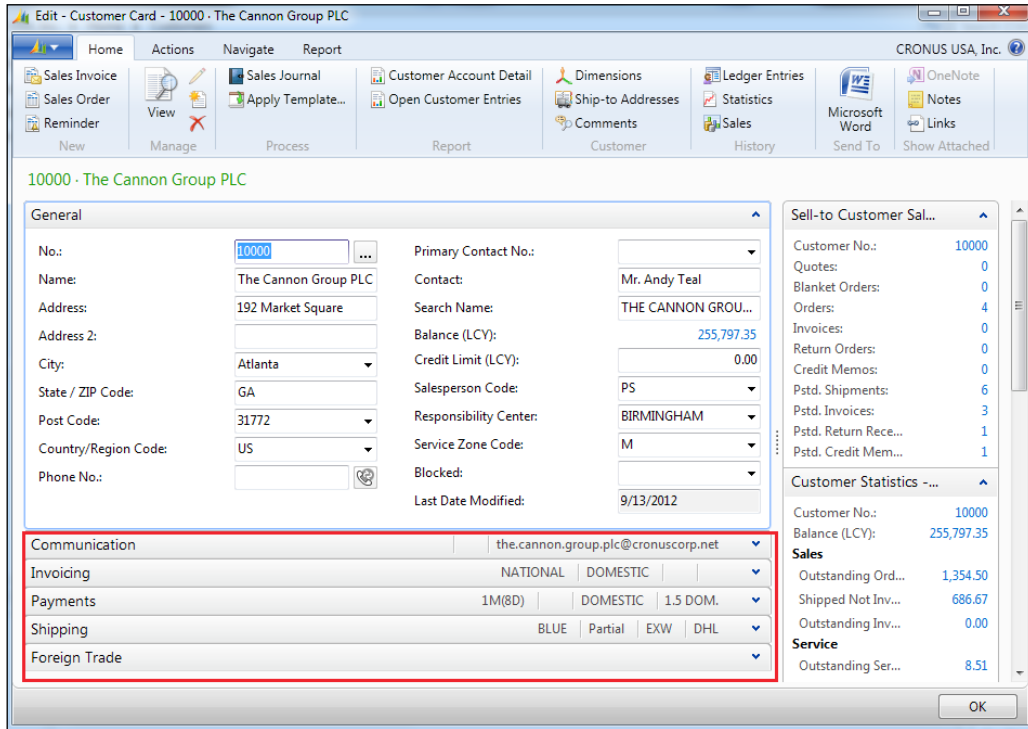
The screenshot displays the 'Edit - Customer Card - 10000 - The Cannon Group PLC' window. The interface includes a ribbon with tabs for Home, Actions, Navigate, and Report. The main content area is divided into several sections:

- General:** Fields for No. (10000), Name (The Cannon Group PLC), Address (192 Market Square), City (Atlanta), State / ZIP Code (GA 31772), Country/Region Code (US), Primary Contact No. (Mr. Andy Teal), Search Name (THE CANNON GROU...), Balance (LCY) (255,797.35), Credit Limit (LCY) (0.00), Salesperson Code (PS), Responsibility Center (BIRMINGHAM), Service Zone Code (M), and Last Date Modified (9/13/2012).
- Communication:** Email address: the.cannon.group.plc@cronuscorp.net
- Invoicing:** NATIONAL | DOMESTIC
- Payments:** 1M(8D) | DOMESTIC | 1.5 DOM.
- Shipping:** BLUE | Partial | EXW | DHL
- Foreign Trade:** (Dropdown menu)
- Sell-to Customer Sal...:** Summary statistics including Customer No. (10000), Balance (LCY) (255,797.35), and Sales/Service metrics.
- Customer Statistics -...:** Summary statistics including Customer No. (10000), Balance (LCY) (255,797.35), and Sales/Service metrics.

The right side of the window features a vertical scroll bar and an OK button at the bottom right.



Good real estate is scarce, especially if that real estate is your screen. The Dynamics NAV product team at Microsoft understands that everyone wants everything displayed on their screens. However, your screen is only so big. You can increase your screen resolution, but your eyes may hurt. In order to organize the large amount of data Dynamics NAV can capture on a single card page, data is grouped using FastTabs.



If you expand the FastTabs, you'll see the relevant information related to the respective FastTab. In our case, since we're looking at the customer card, the **Communication** FastTab displays information such as the customer's phone number and e-mail. If you expand the **Payments** FastTab, you'll see information related to their payment terms, payment methods, and so on.

These FastTabs can be expanded and collapsed. If you have really good eyes and can handle high resolutions, you can expand everything to see everything.

You will typically see the card page used for master records, such as customer, vendor, item, salespeople, and so on. The card page displays more detail for a single record out of the list of records.

## Personalizing the card page

To access the customization screen on the card page, the steps are the same as accessing it from the list page.

What is missing from the card page customization is the ability to choose which columns to hide and display. The reason for this is that the card page does not have columns; they have fields that are predefined by the NAV developer.

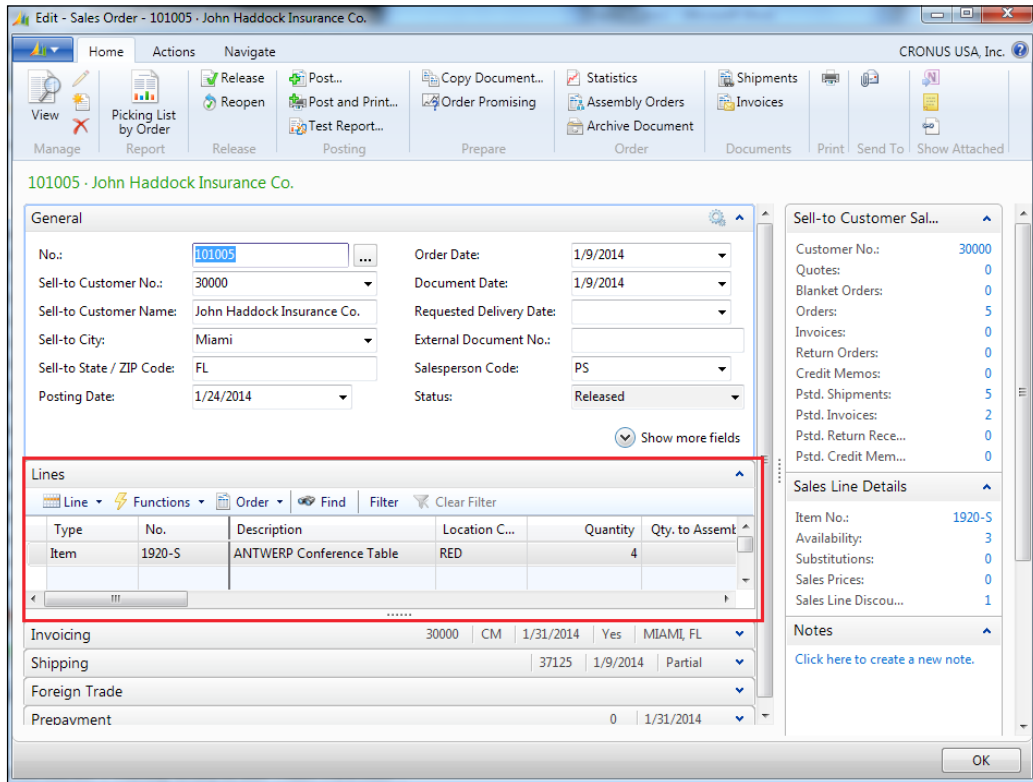
## Exploring the document page

The last page that we need to pay attention to is the document page. An example of the document page is the **Sales Orders** page. Go back to the home page and click on **Sales Orders**; then double-click on a sales order to see an example of a document page.

The screenshot shows the 'Edit - Sales Order - 101005 - John Haddock Insurance Co.' window. The interface includes a ribbon with tabs for Home, Actions, and Navigate. The main area is divided into several sections:

- General:** Fields for No. (101005), Order Date (1/9/2014), Sell-to Customer No. (30000), Document Date (1/9/2014), Sell-to Customer Name (John Haddock Insurance Co.), Requested Delivery Date, Sell-to City (Miami), External Document No., Sell-to State / ZIP Code (FL), Salesperson Code (PS), Posting Date (1/24/2014), and Status (Released).
- Lines:** A table with columns for Line, Type, No., Description, Location C..., Quantity, and Qty. to Assemk. The first line is Item 1920-5, ANTWERP Conference Table, RED, with a quantity of 4.
- Summary:** Fields for Invoicing (30000, CM, 1/31/2014, Yes, MIAMI, FL), Shipping (37125, 1/9/2014, Partial), Foreign Trade, and Prepayment (0, 1/31/2014).
- Sell-to Customer Sales:** A summary of sales statistics including Customer No. (30000), Quotes (0), Blanket Orders (0), Orders (5), Invoices (0), Return Orders (0), Credit Memos (0), Pstd. Shipments (5), Pstd. Invoices (2), Pstd. Return Rece... (0), and Pstd. Credit Mem... (0).
- Sales Line Details:** Fields for Item No. (1920-S), Availability (3), Substitutions (0), Sales Prices (0), and Sales Line Discou... (1).
- Notes:** A section with a link to 'Click here to create a new note.'

The usual layout is consistent to that of a card page. You have the FactBoxes to the right and the FastTabs to group the data. One thing unique to the document page is the lines within the page, which are called subpages.



This allows for the entry of detailed information in respect to the header record. In this case, since we're looking at the sales order page, the header is where the sales order number is generated. This is where we put the customer number, the customer purchase order number, and the shipping location for the order. **Lines** is where we put which items and services the customer will be ordering from us.

You will typically find the document page on any order screen (purchase and sales), a few master records such as the Production **Bill of Material (BOM)**, and fixed assets. When there is a header/line relationship between two related tables, a document page will typically be used to display the data.

## Exploring the rest of the RTC environment

There are a lot of other types of pages in the RTC, but it really just boils down to the three types that I have listed previously. These three terms are all that's necessary for you to speak intelligently with a Dynamics NAV consultant when talking about modifications that you need. If you need to know the proper naming convention for the different types of pages, they can be found on this link: [http://msdn.microsoft.com/en-us/library/jj651618\(v=nav.70\).aspx](http://msdn.microsoft.com/en-us/library/jj651618(v=nav.70).aspx).

I would encourage you to click around the RTC environment and get used to the "look and feel". As I've mentioned earlier, there's no way to become an awesome developer if you cannot find where you need to go. Being lost is not awesome.

There are also some simple walkthroughs that will take you through some of the modules. I highly recommend these walkthroughs as they will help you understand more about the capabilities of Dynamics NAV. They will also help you understand what is provided out of the box and what needs to be modified when you start using Dynamics NAV for your company.

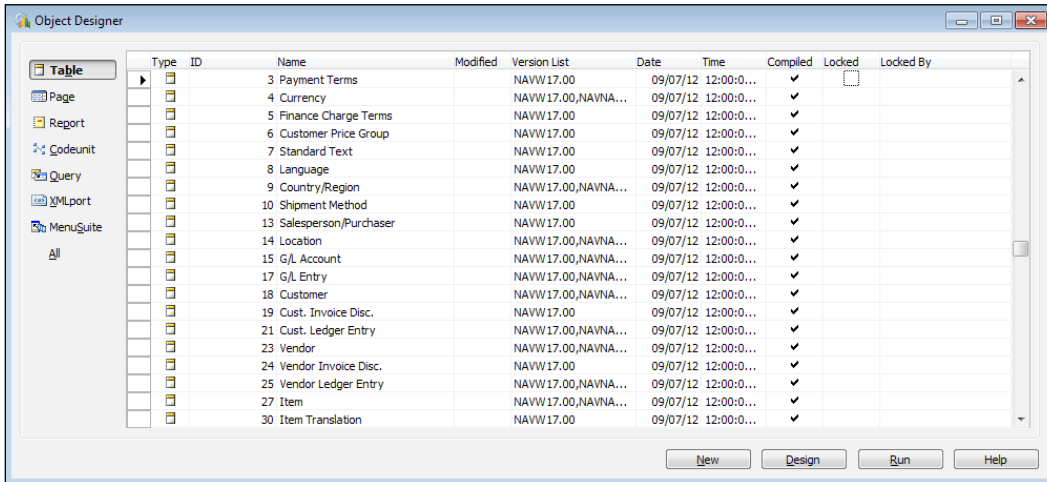
The link to the walkthroughs is as follows:

[http://msdn.microsoft.com/en-us/library/hh997462\(v=nav.70\).aspx](http://msdn.microsoft.com/en-us/library/hh997462(v=nav.70).aspx)

## The Development Environment interface

The Development Environment is the place where any code and any of the visual aspects of Dynamics NAV are programmed and designed. The look and feel of the Developer Environment is distinctly different from the Windows Client.

All the programming and visual design elements of the Development Environment are located in objects accessed through a screen called **Object Designer**. An object represents things like a single table that stores data, or a report that presents data as a sales order or invoice. Objects are explained in more depth later. If **Object Designer** is not already open when you start the Development Environment, go to **Tools | Object Designer**.



There are seven types of objects in Dynamics NAV. They are:

- **Tables:** Tables are where the data is stored. In Dynamics NAV, most of the business rule validation coding occurs at the table level as the users are entering the data. The reason for this design is so that you can create many pages and other interfaces that enter data into tables.
- **Page:** Pages are where information from the tables are displayed. The pages are used to input data and get outputs from the table for the users. The end user will not interface with the tables directly; they will always need to go through a page or other intermediaries. Every page in Dynamics NAV that is of any importance to the developer and user will have a table associated with it.
- **Report:** Reports are run in a looping fashion, meaning it will go through all of the records on the table(s) you specify based on the filtering criteria you set. There are two types of reports, a **regular report** that prints to Excel, PDF, or paper; and a **process-only report**.

Regular reports are generally used to only output data. There are certain regular reports that will modify data in tables, but they are usually done in the backend.

Process-only reports are exactly what they sound like. They're just processes to loop through the data and make changes to them. These are very useful for Dynamics NAV developers to mass update data in a particular table. This tool, although powerful, can prove quite dangerous. The last thing you want is to have a novice developer doing mass changes to the data without knowing the consequences.

- **Codeunit:** Codeunits are basically C/AL code containers. Codeunits are where other objects will make calls to execute some logic. There's no interface for the user's input or output. Codeunits are the backbone of Dynamics NAV because this is where the financial posting routines happen. For example, if you post a sales order from a sales order page, it will have to use codeunits to validate and move the data to the proper ledger tables (that is, from the sales order tables to the financial ledger tables).

Codeunits are closely guarded in Dynamics NAV, mainly to protect you from yourself. Using the standard license, you will not be able to modify or create codeunits. You will need to buy the Application Builder license in order to modify and create codeunits. Even with the Application Builder license, you will not be able to modify the codeunits and the tables that store the financial ledger data.

In order to do that, in addition to the Application Builder license, you will need to buy the Solution Developer license. Again, most companies will never need the Application Builder or Solution Developer license. It's much more cost efficient to have professional Dynamics NAV developers create complex solutions and have you modify it than spending time to create it yourself.

- **Query:** Query is used to group fields from different tables into one area so that you can create pages and reports of it. Before the Query object came out, to combine calculation data on fields from different tables, you would've had to create a report or put them on a form as calculations, either as functions or variables. If you wanted to re-use these functions, you would have had to recreate them from scratch, as it was not easy to copy and paste. With Query objects, you can predefine the objects to a "table", and then create pages and reports of these Query objects.
- **XMLport:** XMLport is where you can export data out of and import data into tables in Dynamics NAV using XML files. It is also used in conjunction with web services to pass XML documents. Instead of creating the XML data by hand, XMLport allows you to quickly get the format and link them to the appropriate fields and tables.

In addition to files in the XML format, you can also use XMLport to import and export data in any delimited flat files; for example, a comma delimited file (CSV).

- **MenuSuite:** When you click on the departments in the RTC to access all of the modules, the listed contents are in the MenuSuite object. If you've created any reports or pages that you want to be searchable from the RTC, you will need to edit the MenuSuite and add this object into the MenuSuite object.

## Summary

In this chapter, we've described the differences between the Windows Client and the Development Environment. The WC is the main interface where the end users enter and retrieve data for their day-to-day operations. The Development Environment is the main interface where the Dynamics NAV developers make changes to the interface, to the table structure, and to the RTC.

Understanding how to navigate within the interface will help us as we go through the development of the application because we will know exactly where to go to make these changes appear for the end users. In addition, being familiar with the interface allows us to "copy and paste" similar functions that we want to replicate.

If you would like more information on the Dynamics NAV Windows Client user interface design, and on the different pages that are available, visit this site:

[http://msdn.microsoft.com/en-us/library/jj128065\(v=nav.70\).aspx](http://msdn.microsoft.com/en-us/library/jj128065(v=nav.70).aspx)

In the next chapter, we will take a closer look at the data structure of Dynamics NAV. We will look at the master and transactional data and how we can look up and drill down the information within the Windows Client.

# 3

## Exploring the Data Structure and Basic Layout of Dynamics NAV

*"I hear and I forget. I see and I remember. I do and I understand."  
– Confucius*

In the previous chapter, we looked at general navigation within Dynamics NAV. Now we will go through and explore the data structure and layout within the base product.

So why is the title of the chapter exploring the *basic* layout? There are some concepts within Dynamics NAV that are reused throughout the software (refer back to the consistency of the software). The design of the tables is no exception.

In this chapter, we will go through some of the tables that are commonly used for Dynamics NAV implementations. The tables that we will go through will cover the following departments:

- Sales
- Purchasing
- Inventory

Within each department, we will explore the basic tables, such as:

- Master data: Customer, vendor, items, and so on
- Reference data: Shipping agent, payment terms, posting groups, and so on
- Transaction data: Orders, journal lines, ledger entries, and so on



The data structure within Dynamics NAV may be similar to other ERP software packages or the one that you're coming off of; however, in order for us to be able to understand how to modify our application, we must first be able to know where to modify.

We will also go through an example of entering and posting a sales order. After the order is posted, we will analyze what tables are affected so that we can better understand how the data flows.

## **Exploring the different departments**

From the beginning, Dynamics NAV was designed to be used as an integrated system. It started off with the basic General Ledger functionality, and from there it has been extended to different functions and modules to what Dynamics NAV is today.

One of the main selling points of Dynamics NAV is that it is a fully integrated ERP software. The data structure is built like a spider web; you can easily drill down and drill across data from different modules anywhere in the system. This integrated data approach is in sharp contrast to the best-of-breed or the Frankenstein approach used by other ERP software. This approach requires the modules to be "bolted on" by a developer, and they need to be reconciled periodically by the end user.

The end result with the Dynamics NAV approach is a seamless ERP system without the different interfaces and data integration errors. Not to mention the ability to navigate on a certain transaction and have every related transaction shown instantly.

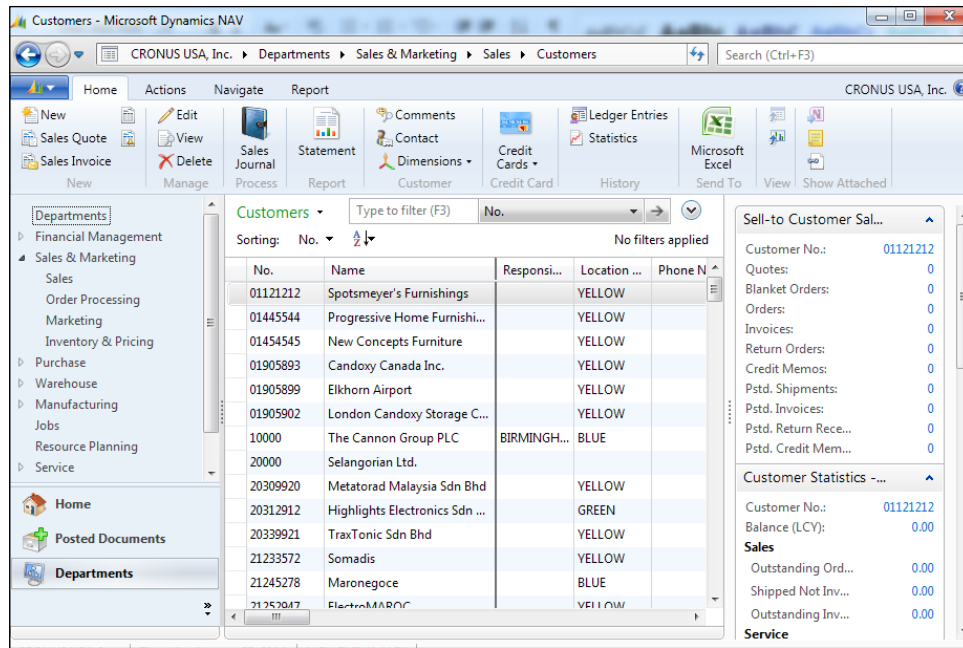
## **Drilling across modules and departments**

Almost every department or module in Dynamics NAV will have some master data. The master data are records including, but not limited to, customers, vendors, and items. They're typically referenced when you process transactions within the database.

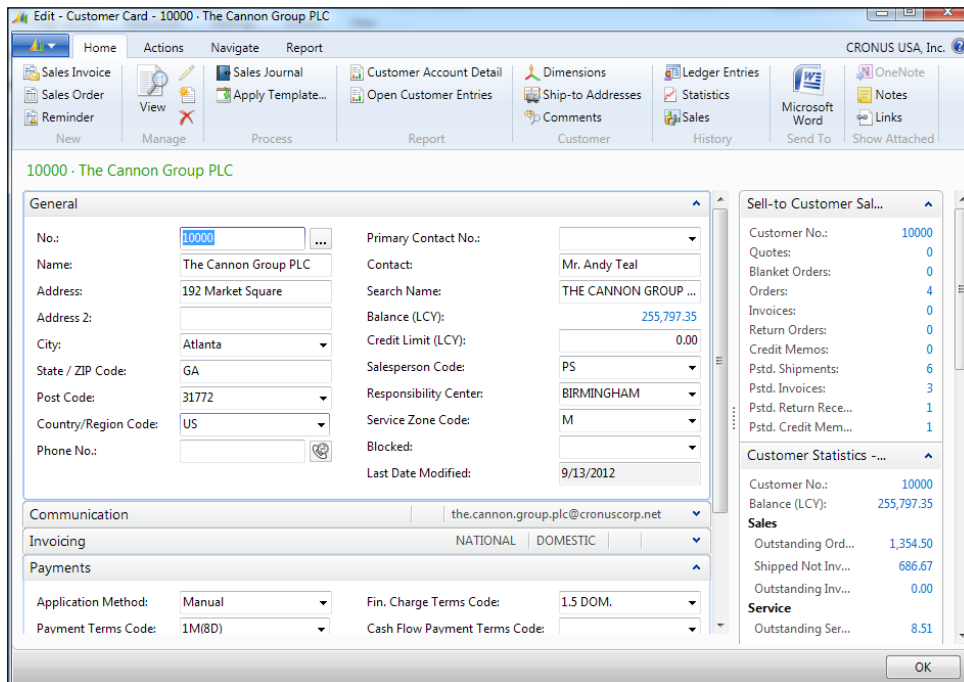
From here, we will drill across different modules or departments without having to click out of the screen. Because Dynamics NAV is built on an integrated, or spider-web, approach, you can drill down and drill across to find the transaction you're looking for in Dynamics NAV.

## **Sales and marketing**

From the WC, go to **Departments | Sales & Marketing | Sales | Customers**. You can also access the list of customers by using page search, as described in *Chapter 2, Getting Familiar with Dynamics NAV 2013*.



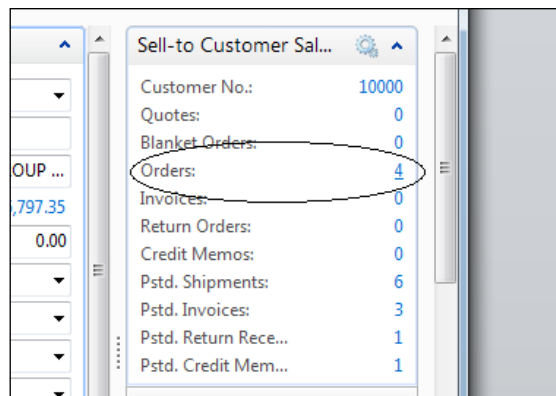
Double-click on customer 10000, The Cannon Group PLC, to bring up the card page.



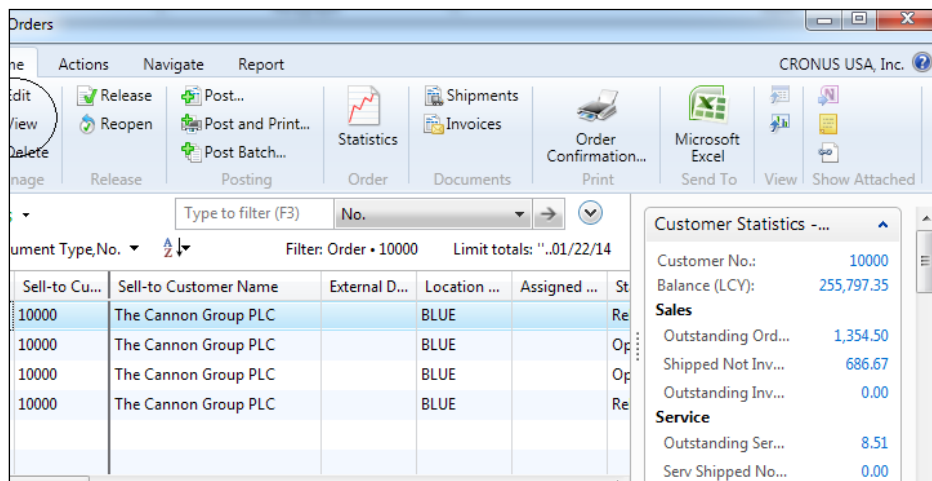
As you may have guessed, the Customer master data holds attributes related to your customers, such as the name, address, and payment terms.

Along with the information on the customer, you can see the outstanding balance in the **General** FastTab; you can also see any outstanding orders and the amount of orders from the FactBoxes to the right.

From the customer card, let's take a look at the outstanding sales orders entered for this customer. On the **Sell-to Customer Sales History** FactBox, click on **4** next to **Orders:**. This will give you a list of the outstanding orders associated with this customer.

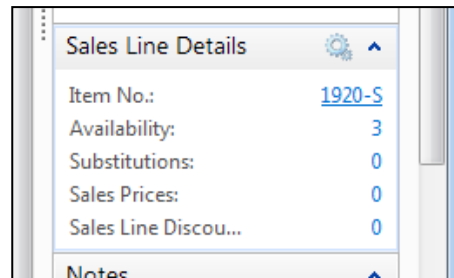


From the list of orders for this customer, click on the **View** or **Edit** button to see the full sales order for order **101016**.

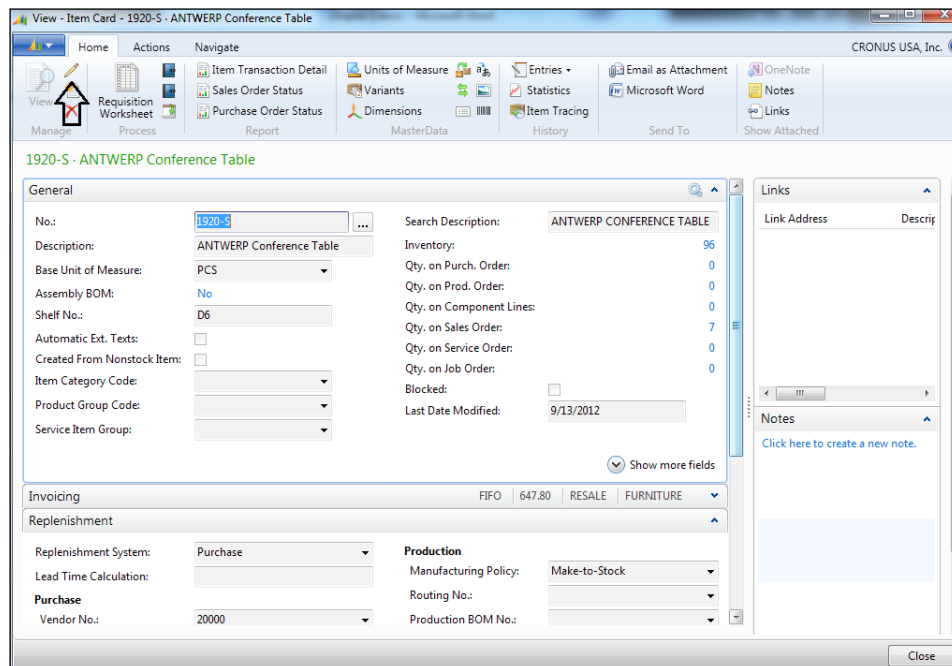


## Going into the inventory

In the **Sales Orders** screen, you'll notice item number **1920-S**. Let's say you want to take a look at the item details for **1920-S**; go over to the **Sales Line Details** FactBox and click on **Item No.** to bring up the **Item Card** screen.



In the **Item Card** screen, we're able to see the details for item **1920-S**. You can also edit the item information for **1920-S** by going into the **Edit** mode. In case you don't know how to go into the **Edit** mode, it is done by clicking on the **Edit** option directly on the page.



## Continuing on to the vendor

Let's focus on the **Replenishment** tab in the item card. From here, we can see the vendor that supplies this item in the **No.** field. Click on the down arrow in the **No.** field to get a short list of vendors.

| Group Code:                | No.      | Name                         | City        | Post Code | Phone No. | Contact               |
|----------------------------|----------|------------------------------|-------------|-----------|-----------|-----------------------|
| Item Group:                | 01254796 | Progressive Home Furnishings | Columbia    | 27136     |           | Mr. Michael Sean Ray  |
|                            | 01587796 | Custom Metals Incorporated   | Birmingham  | 35242     |           | Mr. Peter Houston     |
|                            | 01863656 | American Wood Exports        | New York    | 11010     |           | Mr. Jeff D. Henshaw   |
|                            | 01905283 | Mundersand Corporation       | Thunder Bay | P7A 4K8   |           | Mr. Mike Hines        |
|                            | 01905382 | NewCaSup                     | Toronto     | M5E 1G5   |           | Mr. Toby Nixon        |
|                            | 01905777 | OakvilleWorld                | Oakville    | L6J 3J3   |           | Mr. Sean P. Alexander |
| Replenishment System:      | 10000    | London Postmaster            | Atlanta     | 31772     |           | Mrs. Carol Philips    |
| Replenishment Calculation: | 20000    | AR Day Property Management   | Chicago     | 61236     |           | Mr. Frank Lee         |

Click on **Advanced** to get the full list.

| No.      | Name                          | Responsibil... | Location C... | Phone No. | Contact               |
|----------|-------------------------------|----------------|---------------|-----------|-----------------------|
| 01905283 | Mundersand Corporation        |                |               |           | Mr. Mike Hines        |
| 01905382 | NewCaSup                      |                |               |           | Mr. Toby Nixon        |
| 01905777 | OakvilleWorld                 |                |               |           | Mr. Sean P. Alexander |
| 10000    | London Postmaster             | NEW YORK       |               |           | Mrs. Carol Philips    |
| 20000    | AR Day Property Management    | NEW YORK       | YELLOW        |           | Mr. Frank Lee         |
| 20300190 | Malay-Dan Export Unit Sdn Bhd |                | YELLOW        |           | Mr. Fabrice Perez     |
| 20319939 | KDHSL99 Sdn Bhd               |                |               |           | Mr. Toh Chin Theng    |
| 20323323 | Tengah Butong Sdn Bhd         |                |               |           | Mrs. Anisah Yoosoof   |
| 21201992 | Texpro Maroc                  |                |               |           | M. Charaf HAMZAOUI    |

In the **Vendor List** screen, we can click on the **View** or **Edit** button to bring up the card to view or edit the detailed information.

As with any list screen that you'll find throughout Dynamics NAV, you can see any related information that's associated with the record you're working with on the FactBoxes to the right.

We've started from the customer in the Sales and Marketing department and drilled across to the Inventory, and then to the Purchases department. We've done this without exiting the customer master record that we started from.

## Drilling down to the detailed transactions

Most of the master records in Dynamics NAV will have detailed transactions associated with them. For example, the Customer will have an associated Customer Ledger, Item will have an Item Ledger, Vendor will have a Vendor Ledger, and so on.

When we drill down to the detailed transactions, we will still be able to find related information and drill across information.

Without exiting the **Vendor List** screen, take a look at the **Vendor Statistics** FactBox.

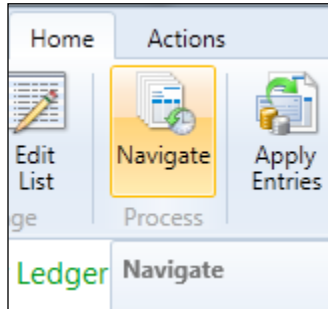
| Vendor Statistics    |                 |
|----------------------|-----------------|
| Vendor No.:          | 20000           |
| Balance (LCY):       | <u>3,580.92</u> |
| Outstanding Ord...   | 0.00            |
| Amt. Rcd. Not Inv... | 0.00            |
| Outstanding Invo...  | 0.00            |
| Total (LCY):         | 3,580.92        |
| Overdue Amount...    | -1,893.78       |

Click on the amount across **Balance (LCY):**. This will drill down to the **Vendor Ledger Entries** screen, which is a table where the posted transactions for the vendor are stored.

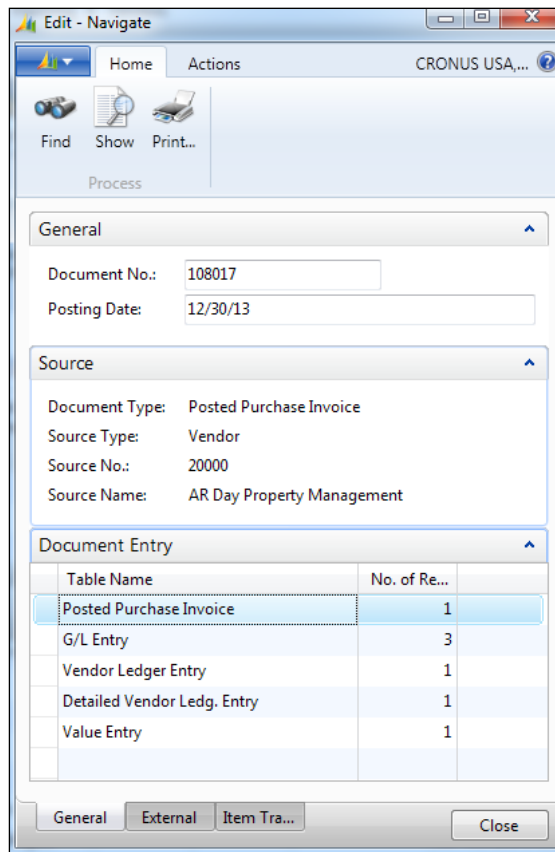
| Posting Date | Document Type | Document No. | External Docume... | Vendor No. | Description  | Original Amount | Amount    | Remaining Amount | Due Date   | Pmt. Discou... |
|--------------|---------------|--------------|--------------------|------------|--------------|-----------------|-----------|------------------|------------|----------------|
| 12/30/2013   | Invoice       | 108017       | 5755               | 20000      | Order 106001 | -1,893.78       | -1,893.78 | -1,893.78        | 12/31/2013 | 12/30/2013     |
| 1/20/2014    | Invoice       | 108025       | 5966               | 20000      | Order 106007 | -1,687.14       | -1,687.14 | -1,687.14        | 1/31/2014  | 1/20/2014      |

On the outstanding balances for vendor **20000, AR Day Property Management**, there is an outstanding invoice, **108017**, that was due on **12/31/2013**. Notice that the **Remaining Amount** column still shows what we owe this vendor.

To look at the details of this past-due invoice, click on the **Navigate** button.



The **Navigate** button is available on all of the pages that display transaction-level data. This feature in Dynamics NAV will allow the system to find all the related transactions of the record we're navigating; in this case, vendor invoice **108017**.



This **Navigate** screen tells us that entries in these tables were created when this document was posted.

For each of these entries, you can click on **Show** to see the detailed transactions. Click on **Show** while you have the **Posted Purchase Invoice** table name highlighted. This will show you the details of this vendor invoice.

108017 - AR Day Property Management

**General**

No.: 108017      Posting Date: 12/30/2013  
 Buy-from Vendor No.: 20000      Document Date: 12/30/2013  
 Buy-from Contact No.:      Quote No.:  
 Buy-from Vendor Name: AR Day Property Management      Order No.: 106001  
 Buy-from Address: 100 Day Drive      Pre-Assigned No.:  
 Buy-from Address 2:      Vendor Order No.:  
 Buy-from City: Chicago      Vendor Invoice No.: 5755  
 Buy-from State / ZIP Code: IL      Order Address Code:  
 Buy-from Post Code: 61236      Purchaser Code: RL  
 Buy-from Contact: Mr. Frank Lee      Responsibility Center:  
 No. Printed: 0

**Lines**

| Type | No.    | Description             | Quantity | Unit of Mea... | Direct Unit Cost... |
|------|--------|-------------------------|----------|----------------|---------------------|
| Item | 1964-S | TOKYO Guest Chair, blue | 14       | PCS            | 150.30              |

Notes  
[Click here to create a new note.](#)

Close

## Keep drilling

In the **Posted Purchase Invoice** screen, you can look up item **1964-S** and do the following:

- Bring up the advanced list
- Bring up the item card
- Drill down the inventory to bring up the **Item Ledger Entries** screen
- Navigate to one of the item ledger transactions
- Show the transactions on one of the navigated entries
- Continue drilling to your heart's content



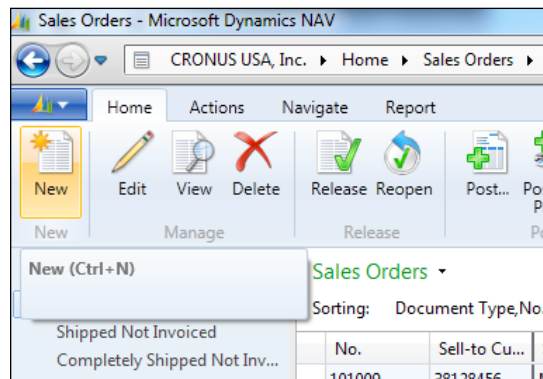
Knowing how to navigate in Dynamics NAV will be important for you to understand the ins and outs of the software. It allows us to know what tables are affected by what transactions so we can make modifications intelligently.

## Creating a sales order

Now that we have understood how to navigate within Dynamics NAV, we will enter an actual transaction and see the data in action. In our example, we will enter a sales order and post the sales order to convert it into an invoice.

Come back to the main Role Center page and go to the **Sales Orders** screen (you should know how to access it. Hint: use page search!).

In the **Sales Orders** screen, click on **New** to create a new order.



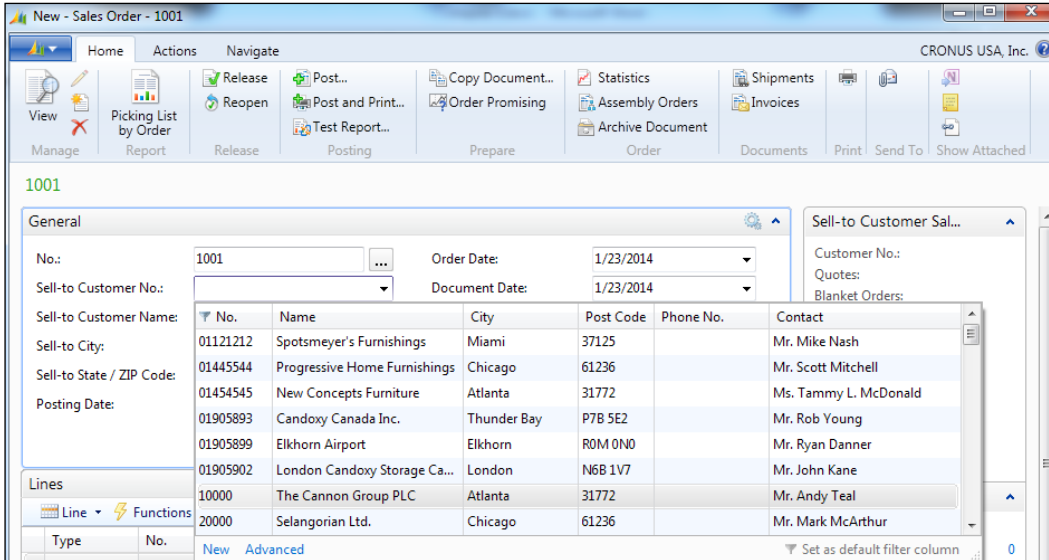
This will bring up a blank **Sales Order** screen.

The screenshot shows the 'Sales Order' window with the following details:

- Customer Information:** Customer No., Customer Name, City, State / ZIP Code, Date.
- Order Information:** Order Date, Document Date, Requested Delivery Date, External Document No., Salesperson Code, Status (Open).
- Sell-to Customer Sales:** Customer No., Quotes, Blanket Orders, Orders, Invoices, Return Orders, Credit Memos, Pstd. Shipments, Pstd. Invoices, Pstd. Return Rece..., Pstd. Credit Mem...
- Sales Line Details:** Item No., Availability (0), Substitutions (0), Sales Prices (0), Sales Line Discou... (0).
- Notes:** Click here to create a new note.
- Table:** Columns: No., Description, Location C..., Quantity, Qty. to Assemb...
- Bottom Section:** g, g, Trade, ment.

Hit *Enter* in the **No.** field and a unique sales order number will be generated based on a number series.

Click on the down arrow on the **Sell-to Customer No.** field to look up and select the customer number for this sales order. Select customer **10000, The Cannon Group PLC**.



When you select the customer, the customer information will be automatically populated for you. You can expand the **Shipping** FastTab and change the shipping information if you wish.

Going straight to the **Lines** FastTab, fill in the three lines with the following information:

| Type     | No.   | Quantity |
|----------|-------|----------|
| Item     | 70000 | 30       |
| Item     | 70001 | 20       |
| Resource | LIFT  | 1        |

The **Lines** FastTab is where we enter the items the customer wants to buy. Note that a lot of the information will be present by default as you enter the lines,, including the **Unit Price** value. For this order, you will only need to fill in the fields shown in the preceding table.

Your sales order should look something similar to the following screenshot:

**1001 - The Cannon Group PLC**

**General**

No.: 1001 Order Date: 1/23/2014  
 Sell-to Customer No.: 10000 Document Date: 1/23/2014  
 Sell-to Customer Name: The Cannon Group PLC Requested Delivery Date:  
 Sell-to City: Atlanta External Document No.:  
 Sell-to State / ZIP Code: GA Salesperson Code: PS  
 Posting Date: 1/23/2014 Status: Open

**Lines**

| Type     | No.   | Description     | Location C... | Quantity | Unit Price Excl. ... | Unit of... | Line Amount Ex... |
|----------|-------|-----------------|---------------|----------|----------------------|------------|-------------------|
| Item     | 70000 | Side Panel      | BLUE          | 30       | 47.30                | PCS        | 1,419.00          |
| Item     | 70001 | Base            | BLUE          | 20       | 62.10                | PCS        | 1,242.00          |
| Resource | LIFT  | Charge for Lift | BLUE          | 1        | 450.00               | HOUR       | 450.00            |

**Sell-to Customer Sales**

Customer No.: 10000  
 Quotes: 0  
 Blanket Orders: 0  
 Orders: 5  
 Invoices: 0  
 Return Orders: 0  
 Credit Memos: 0  
 Pstd. Shipments: 6  
 Pstd. Invoices: 3  
 Pstd. Return Rece...: 1  
 Pstd. Credit Mem...: 1

**Sales Line Details**

Item No.: 70000  
 Availability: 2,172  
 Substitutions: 0  
 Sales Prices: 0  
 Sales Line Discou...: 0

**Notes**

[Click here to create a new note.](#)

To get a subtotal of the order, click on the **Navigate** menu at the top and click on **Statistics**, as shown in the following screenshot:

**Edit - Sales Order - 1001 - The Cannon Group PLC**

**Navigate**

Statistics Card Dimensions Approvals Assemblies

**1001 - The Cannon Group PLC**

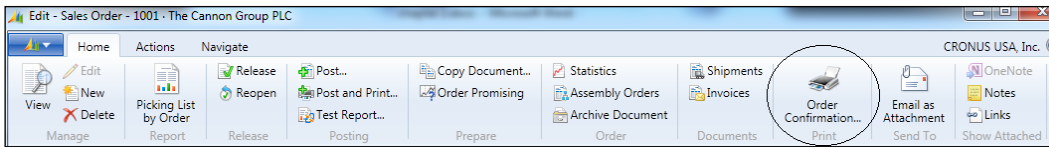
**General**

The statistics screen will also break down any discounts and sales tax if applicable. If you're not using the database localized to North America, your statistics screen will look a little different.

| General                  |          |
|--------------------------|----------|
| Amount Excl. VAT:        | 3,111.00 |
| Inv. Discount Amount:    | 0.00     |
| Total Excl. VAT:         | 3,111.00 |
| Tax Amount:              | 146.55   |
| Total Incl. VAT:         | 3,257.55 |
| Sales (\$):              | 3,111.00 |
| Profit (\$):             | 1,754.00 |
| Adjusted Profit (\$):    | 1,754.00 |
| Original Profit %:       | 56.4     |
| Adjusted Profit %:       | 56.4     |
| Quantity:                | 51       |
| Parcels:                 | 0        |
| Net Weight:              | 125      |
| Gross Weight:            | 143.9    |
| Volume:                  | 2.4      |
| Original Cost (\$):      | 1,357.00 |
| Adjusted Cost (\$):      | 1,357.00 |
| Cost Adjmt. Amount (\$): | 0.00     |
| Sales Tax Breakdown:     |          |
| ATLANTA, GA:             | 146.55   |
| No. of VAT Lines:        | 6        |

You can expand the **Invoicing** and **Shipping** FastTabs to see the amount in each shipment if you were to partially ship the order.

Once you've confirmed the total, go back to the order. To print a copy of the sales order, go to the **Home** tab on the top and click on **Order Confirmation....**



Click on **Print** or **Preview** on the sales order to print a copy for the customer.

**Print Preview**

**Sales Order**

1 of 1 | Whole Page | Find | Next

**SALES ORDER**  
Page: 1

Sales Order Number: 1001  
Sales Order Date: 1/23/2014

**Sold To:** The Cannon Group PLC  
Mr. Andy Teal  
192 Market Square  
Atlanta, GA 31772  
USA

**Ship To:** The Cannon Group PLC  
Mr. Andy Teal  
192 Market Square  
Atlanta, GA 31772  
USA

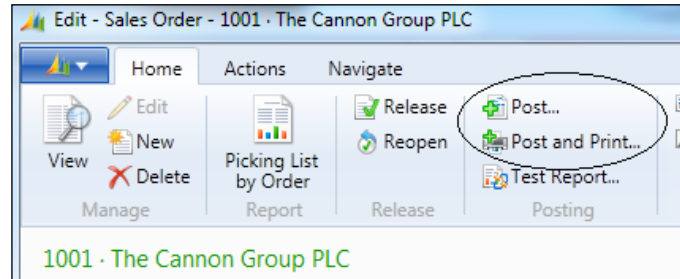
Tax Ident. Type: Legal Entity  
Ship Via: Ex Warehouse  
Ship Date: 1/23/2014  
Terms: 1 Month/2% 8 days

Customer ID: 10000  
P.O. Number:  
P.O. Date: 1/23/2014  
SalesPerson: Peter Sadow

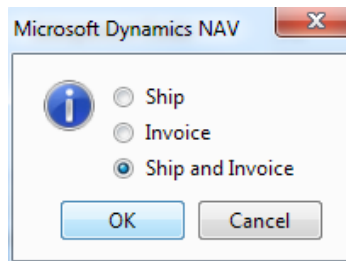
| Item No.                     | Description     | Unit  | Quantity | Unit Price       | Total Price     |
|------------------------------|-----------------|-------|----------|------------------|-----------------|
| 70000                        | Side Panel      | Piece | 30       | 47.30            | 1,419.00        |
| 70001                        | Base            | Piece | 20       | 62.10            | 1,242.00        |
| LIFT                         | Charge for Lift | Hour  | 1        | 450.00           | 450.00          |
| Amount Subject to Sales Tax  |                 |       |          | 3111.00          |                 |
| Amount Exempt from Sales Tax |                 |       |          | 0.00             |                 |
|                              |                 |       |          | <b>Subtotal</b>  | <b>3,111.00</b> |
|                              |                 |       |          | Invoice Discount | 0.00            |
|                              |                 |       |          | Total Sales Tax  | 146.55          |
|                              |                 |       |          | <b>Total</b>     | <b>3,257.55</b> |

In a normal operation where inventory quantities are involved, we would create and print the pick ticket for the warehouse staff to do their work. For the purpose of this walkthrough, we will bypass those steps and go directly into posting this order into a sales invoice.

Go back to the **Sales Order** screen and click on **Post...** or **Post and Print...** (if you want a sales shipment and a sales invoice printout).



The system will prompt you to choose either **Ship**, **Invoice**, or **Ship and Invoice**. Select the **Ship and Invoice** option and click on **OK**.



After the order is done posting, you will be brought back to the blank **Sales Order** screen. We're done here; go ahead and close the order screen.

Let's see the posted transactions that we've done and navigate to see what ledger entries were created as a result.

We want to go into the **Posted Sales Invoice** screen. The easiest way to do this is to use search. You can also access it from the main RTC by clicking on the **Posted Documents** tab, and then on **Posted Sales Invoice**.

When we bring up the **Posted Sales Invoice** screen, we will get a list of the posted invoices in the system. The invoice we posted will be the very last one.

| No.    | Sell-to Cu... | Sell-to Customer Name         | Currency ... | Amount     | Amount Incl... | Location ... | Electronic... | No. Printed |
|--------|---------------|-------------------------------|--------------|------------|----------------|--------------|---------------|-------------|
| 103001 | 10000         | The Cannon Group PLC          |              | 11,433.25  | 11,776.25      | BLUE         |               | 2           |
| 103002 | 20000         | Selangorian Ltd.              |              | 9,741.71   | 10,033.96      |              |               | 0           |
| 103003 | 30000         | John Haddock Insurance Co.    |              | 8,383.00   | 8,634.49       |              |               | 0           |
| 103005 | 10000         | The Cannon Group PLC          |              | 10,195.02  | 10,806.72      | BLUE         |               | 0           |
| 103006 | 42147258      | BYT-KOMPLET s.r.o.            | CZK          | 58,518.63  | 58,518.63      | RED          |               | 0           |
| 103007 | 43687129      | Designstudio Gmunden          | EUR          | 3,112.13   | 3,112.13       | RED          |               | 0           |
| 103008 | 20000         | Selangorian Ltd.              |              | 970.78     | 1,019.32       |              |               | 0           |
| 103009 | 20000         | Selangorian Ltd.              |              | 266.26     | 279.57         |              |               | 0           |
| 103010 | 32656565      | Antarcticopy                  | EUR          | 3,218.14   | 3,218.14       | YELLOW       |               | 0           |
| 103011 | 49633663      | Autohaus Mielberg KG          | EUR          | 1,159.58   | 1,159.58       | GREEN        |               | 0           |
| 103012 | 46897889      | Englunds Kontorsmöbler AB     | SEK          | 7,841.00   | 7,841.00       | YELLOW       |               | 0           |
| 103013 | 01445544      | Progressive Home Furnishin... |              | 2,461.00   | 2,461.00       | YELLOW       |               | 0           |
| 103014 | 20000         | Selangorian Ltd.              |              | 1,412.13   | 1,482.74       |              |               | 0           |
| 103015 | 47563218      | Klubben                       | NOK          | 115,966.31 | 115,966.31     | YELLOW       |               | 0           |
| 103016 | 35963852      | Heimilispyrdi                 | ISK          | 200,615.42 | 200,615.42     | YELLOW       |               | 0           |
| 103017 | 35451236      | Gagn & Gaman                  | ISK          | 86,949.84  | 86,949.84      | YELLOW       |               | 0           |
| 103018 | 10000         | The Cannon Group PLC          |              | 5,057.00   | 5,334.85       | BLUE         |               | 0           |
| 103019 | 40000         | Deerfield Graphics Company    |              | 1,638.10   | 1,736.39       | BLUE         |               | 0           |
| 103020 | 50000         | Guildford Water Department    |              | 822.00     | 822.00         | BLUE         |               | 0           |
| 103021 | 30000         | John Haddock Insurance Co.    |              | 1,061.30   | 1,114.37       | BLUE         |               | 0           |
| 103022 | 10000         | The Cannon Group PLC          |              | 3,111.00   | 3,257.55       | BLUE         |               | 0           |

Go to the very last record on this screen by pushing *Ctrl + End* on your keyboard or by pushing *Page Down* until the last record is seen. Then, double-click on the last record to bring up the **Posted Sales Invoice** document to see the detailed contents of this invoice.

Edit - Posted Sales Invoice - 103022 - The Cannon Group PLC

Home Actions Navigate

View Edit Delete Navigate Statistics Comments Dimensions Approvals

Manage Process Invoice Credit Cards Transaction Log Entries Credit Card Print Refresh Go to Previous Next OneNote Notes Links Show Attached

103022 - The Cannon Group PLC

General

No.: 103022 Posting Date: 1/23/2014

Sell-to Customer No.: 10000 Document Date: 1/23/2014

Sell-to Contact No.: CT000007 Quote No.:

Sell-to Customer Name: The Cannon Group PLC Order No.: 1001

Sell-to Address: 192 Market Square Pre-Assigned No.:

Sell-to Address 2: External Document No.:

Sell-to City: Atlanta Salesperson Code: PS

Sell-to State / ZIP Code: GA Responsibility Center: BIRMINGHAM

Sell-to Post Code: 31772 No. Printed: 0

Sell-to Contact: Mr. Andy Teal

Notes

[Click here to create a new note.](#)

Lines

| Type     | No.   | Description     | Quantity | Unit of Mea... | Unit Price Excl. ... |
|----------|-------|-----------------|----------|----------------|----------------------|
| Item     | 70000 | Side Panel      | 30       | PCS            | 47.30                |
| Item     | 70001 | Base            | 20       | PCS            | 62.10                |
| Resource | LJET  | Charge for Lift | 1        | HOUR           | 450.00               |

Invoicing: 10000 1M(8D) 2/23/2014

Shipping: 31772 1/23/2014

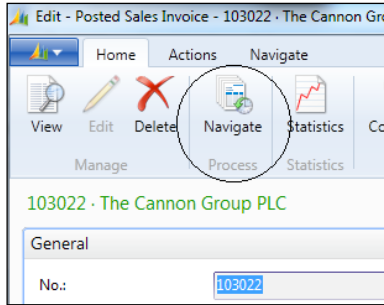
Foreign Trade

Electronic Invoice

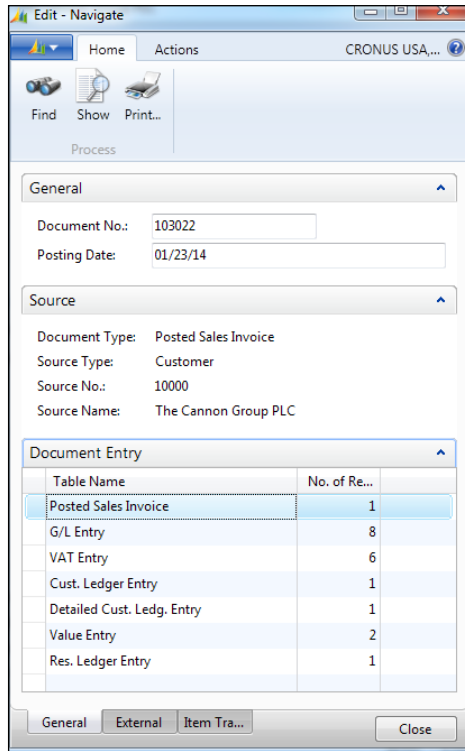
OK



Click on the **Navigate** button to see what entries were created from this posting. Do not confuse this with the **Navigate** tab on the ribbon!

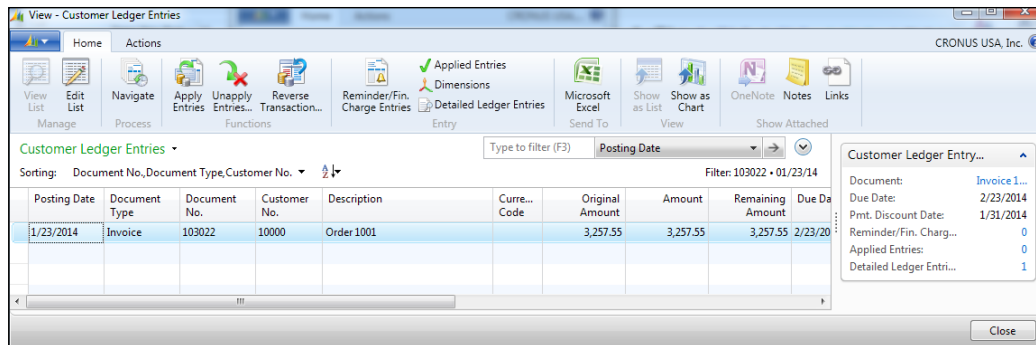


When the **Navigate** screen comes up, you'll see that we've hit several tables by posting the sales order.



If you click on the **G/L Entry** line and click on **Show**, you will see the G/L transactions that are created based on the system setup.

For our purposes, we will take a deeper dive into **Cust. Ledger Entry**. Click on the **Cust. Ledger Entry** line, and then click on **Show** to show this entry.



The customer ledger holds all of the customer-related transactions such as payments, credits, and in this case, invoices.

## Drill it on your own

Now that you know how to enter the sales order, follow the same steps to enter a purchase order. Enter an order with items only (there is no Resource option available on purchase orders), post it, and see what entries are affected.

Drill across and drill down. The more you move around with in the system, the better you will understand how the data flows within Dynamics NAV.

## Summary

In this chapter, we learned that Dynamics NAV is an integrated system, so it's easy for us to navigate between modules without having to leave what we were doing. We were also able to drill down deep into the detailed transactions and pull up the supporting documents for every posted transaction in Dynamics NAV.

From the **Navigate** screen, we can see which ledger entries are affected when an entry is posted. This will help accounting or support personnel to isolate problem entries without having to pull their hair out.

More importantly, understanding how the system flows is the key to becoming a good developer. As mentioned before, an excellent developer in Dynamics NAV is a person that works like a ninja. You barely notice he's there, but he will have a dramatic impact on your operations.

In the next chapter, we will tie what we've learned back to your business. A business application that does not solve any business problems is just a waste of time and resources. We are going to pretend you're a worker in a fictitious company that has been asked to create a software to solve a business problem. In the next chapter, we will explain who you are and what you do.

# 4

## Determining a Task List

*"Most people spend more time and energy going around problems than in trying to solve them."*

*– Henry Ford*

The goal of using any kind of business software is to solve business problems. Some of the more specific reasons for companies buying and using or switching business software are, but not limited to, the following:

- **Efficient record keeping:** Instead of having a hundred Excel spreadsheets that track a specific piece of information saved on your server somewhere, you can incorporate this information into the system.
- **Automation:** Some of the processes that are done manually can be automated. What used to work using pen and paper may not be necessary with the right software in place. This frees up human resources to focus on improving your business in other areas.
- **Compliance:** There may be some authority or customer that demands you to comply with some processes in order to pass the audit or do business. These can get the proper lot control in place if you're a food manufacturer, or if you require **Electronic Data Interchange (EDI)** for the customer to place orders with you.
- **Process optimization:** This is similar to the automation point mentioned previously. The first rule of business process optimization is: "Don't automate, obliterate!" It means that there are certain processes in your business that are not really necessary in the digital age. Using a new business software allows you an opportunity to examine existing business rules, assumptions, and processes to determine whether they're really needed.

- **Visibility:** There are lots of businesses that use their business software mainly as a record-keeping software for year-end tax filing. The analysis is separated out by exporting the data to Excel and running some weird formulas and pivot tables. This is fine, until you want more details on a specific set of numbers. It's very likely that you will need a person to spend a few hours or days drilling down and gathering the answer you're looking for. It's also likely that by the time you get the answer, you would've forgotten what the original question was. New technologies in business software allow you this visibility, that is, to get the answers to your questions efficiently and effectively.
- **Competition is moving:** Hiding in a rabbit hole and ignoring the outside world is good if you're a rabbit. Not so much if you're running a business. When your competitor is using technology to make better decisions, improve customer service, and monitor costs while you're not, it's a problem.
- **Obsolete software:** If the system you're working with is no longer supported, and the programmers are either retiring or switching to another field, it's a good indication that you need to move to a new system.

Again, these are just some of the many reasons why a company would want to consider using business software. In this chapter, we'll examine our hypothetical business and the problem we're trying to solve using Dynamics NAV.

## Who you are

You work for Cronus, USA. Being aware of your many talents and being very cost conscious, the management of Cronus has put you in charge as the operations manager, and you can also double up as the IT administrator.

Your role within the company is not a junior level one, but is quite the opposite; your position in the company requires you to deal with many people across departments. You've been with the company for some time, so you're aware of the political and interpersonal dealings within the company.

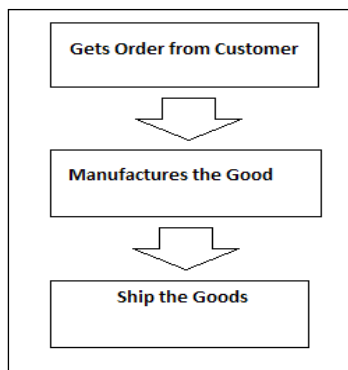
## Your company

Cronus, USA manufactures and sells bicycles, office furniture, and computer equipment. The company has been using an antiquated accounting system for some time. Even though the current legacy system leaves more to be desired, through your persistence, you've managed to get by and get the system to a place where people are not complaining too much.

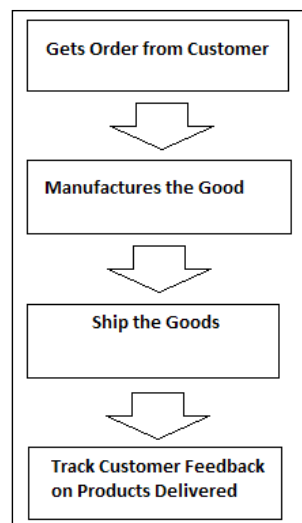
The company has created many Excel sheets and bought/created many small programs to patch the needs of the company. Even though the setup is crazy, you still manage to make it all work somehow.

One day, the management of the company comes to you and asks you to develop a system to record complaints for the items Cronus manufactures. This is to ensure that Cronus has a process in place to record feedback from customers on their products, and continues to deliver products of the highest quality.

Thinking about this process carefully and how it can be applied to your business, the current manufacturing sales process looks like the following:



After discussing with the management, the manufacturing shop floor manager, and the product delivery team, the order fulfillment process will look something like the following:



Of course, there are some actions that need to be performed when the feedback is received in order to make continuous improvements to the products delivered. Providing a place in the system to keep track of the data will be the challenge for you. This will require some changes in the way Cronus operates, and you will need some way to make these changes, such as using an off-the-shelf program, creating a spreadsheet in Excel, or developing a custom module in the current business software.

With users increasingly hating the way things are working, the time it takes for you to manage and troubleshoot, and the overtime you're putting into operations, this will be the last straw. It is time for a system to incorporate all of your existing business processes and have the capability to incorporate any additional changes or additions in the process in the future.

Fortunately, you and your company made the right decision in choosing Dynamics NAV as your ERP software of choice.

## Identifying the major pains in the company

Before presenting any large project to the management for approval, there needs to be proper supporting proof that such an investment is needed in order for the company to grow. Unfortunately, in most cases, just saying that the system is old and antiquated will garner some nods, but not much action.

Even before the selection of the software and the vendor, you've already identified the major pains in the company, so you can have a meaningful conversation with the Dynamics NAV solution center.

There are a lot of different ways to do business analysis and identify major problems within the organization. Any method is fine as long as you can justify the proper **Return on Investment (ROI)**. There's no one proper way to do an ROI, and the key measures for how you determine an ROI is a science in itself.

## Understanding the current operation

The people that are actually doing the work need to be interviewed thoroughly to understand what they're currently doing and why they are doing what they're doing. There are a lot of intricacies that are involved in every business process. Some of the business processes that are in place may not make much sense, but it may be derived from years and years of history. Some of this history may predate your time within the company.

The management of the company needs to be interviewed, because we need to understand what they would consider a success to this system implementation project. Without understanding what the management wants from you as an IT manager, you can pour your heart out and work your fingers to the bones; the project may still be deemed a failure because the objectives of the management were not reached. Make sure to get their objectives in writing!

The goal of the analysis is not for you to solve the problems within the company. The goal is to fully understand your current business process.

The pitfalls of trying to find solutions to your problems prior to starting any implementation are that you may have some preconception of what the solution is, but if the software consultant comes in and proposes a better and more efficient plan of action, all of the work you've done may go waste.

If the software consultant proposes a lesser plan of action, you will know right away. Why? Because you understand the cause and effect of every operation in the company and can instantly know the requirement gaps.

## Listing out all of the problems

After the analysis, there should be a list of requirements for each department. The first version of the list does not have to make sense, but it does have to capture what the users are complaining about. A sample form listing business problems can look similar to the following screenshot:

|    | A                                  | B   | C                 | D                   |
|----|------------------------------------|---|-------------------|---------------------|
| 1  | <b>Improvements List</b>           |   |                   |                     |
| 2  | <b>Problems and Inefficiencies</b> |   |                   |                     |
| 3  | <b>Priority (1 - 10)</b>           | <b>Problem</b>  | <b>Department</b> | <b>Requested By</b> |
| 4  | 4                                  | Takes too long to find payment applications                     | A/P               | Jim                 |
| 5  | 7                                  | No easy way to find out the tracking number for shipped package | Shipping          | Owen                |
| 6  | 10                                 | Need a system to keep track of Customer Feedback                |                   | Alex                |
| 7  | 3                                  | Need to be able to e-mail invoices                              | Invoicing         | Jessica             |
| 8  | 6                                  | No way to easily find out when ordered goods will come in       | Purchasing        |                     |
| 9  |                                    |   |                   |                     |
| 10 |                                    |   |                   |                     |
| 11 |                                    |   |                   |                     |
| 12 |                                    |   |                   |                     |



Every request is important! However, with limited time and resources, the list of problems and challenges should be graded in the order of importance based on your understanding of your current business process.

Another benefit from writing down a list of the problems you're facing is that you will have a guideline on whether the software solution presented to you will fit the needs of your company.

## **Defining the non-negotiable must-haves**

Once the list of the problems has been compiled, the next step is to define the non-negotiable must-haves when you move to Dynamics NAV.

It is important to summarize the important areas that need immediate attention from the management of the company. What's even better is if you can quantify the monetary value for the important areas. Translating the business problems into monetary values will allow you to better calculate the ROI for the management to get the project adopted and approved.

When you begin your software implementation, these non-negotiable must-haves should either be a standard out-of-the-box feature, through an add-on, or should be available via customization without costing an arm and a leg. The non-negotiable must-haves must be clearly communicated with your Dynamics NAV partner.

Having a list of non-negotiable must-haves will also help you stay on track in addressing the business problems when you're presented with all types of different solutions.

After doing the business analysis, you've identified the major non-negotiable must-haves for your company, which are as follows:

- You need a **Quality Management System (QMS)** as the management wants to track the overall quality of the goods delivered from customer feedback.
- You need a faster response time to customers' inquiries on item availability and delivery. This has already proven to be a high priority because Cronus had lost some business because the customer could not get a straight answer on when they would receive their product from customer service.
- You need on-the-fly reporting and business intelligence for management without involving IT or exporting data into Excel. During management meetings, there are a lot of strategic decisions that were held up because of a lack of proper supporting data.

## Designing the solution

Having done your homework, and with your incredible intelligence and wisdom, you know that Dynamics NAV handles most of your non-negotiable must-haves out of the box. The only area where customization will be required is the quality management system, which keeps a track of your unique needs.

Working with your Dynamics NAV partner, you begin to define the detailed specifications for the QMS module that's built into NAV. Being confident of your own abilities (and the frugality of the management), you insisted to your Dynamics NAV partner that you would like to create the QMS system on your own and only involve them if there are some areas you cannot access.

On further analysis of the QMS system, the main functionalities required are obtained. They are as follows:

- Every incident needs a unique number so it can be tracked
- The customer needs to be associated with every incident
- The vendor that supplied the goods, if applicable, needs to be tracked
- The date of the interaction must be tracked
- The item number in question
- The quantity of the item that was accepted and rejected
- The source of the production order or purchase order
- Detailed comments that are associated with every QMS entry

In addition, the following attributes will be needed for the QMS system:

- There can be one or more items per incident
- Any QMS document that has been resolved will need to be marked as such
- The open QMS documents and the closed ones should be displayed as a separate screen
- A report needs to be created on the feedback per item

## **Summary**

In this chapter, we've discussed some of the common problems that a company will face, and considered switching to a new system. As an IT manager, one of your responsibilities is to be prepared to understand your current operations. Part of this process is to assemble a task list of the needs of Cronus for the new system.

You will, most likely, have multiple roles within your organization, and the main advantage you have within your organization is that understanding the system will give you a better insight on how your company operates.

Understanding what the problem is and how to go about solving this problem effectively is valuable to any organization. The way to go about understanding the system is to be familiar with the development environment.

Once the scope of the problem is defined with concrete requirements, the development part is easy. In the following chapters, we will dive deeper into Dynamics NAV to draw inspirations and ideas to build our application.

# 5

## Finding Similar Functions for Inspiration

*"If I have seen further it is by standing on the shoulders of giants."  
– Isaac Newton*

In the previous chapter, we defined our requirements for creating the application that would help our company become more efficient.

Once we have the requirements defined, we can then begin to construct this function within Dynamics NAV. We will go into the application and create the necessary tables to store the information, pages to display the information, and reports to print.

If you're looking for an example of how to program *Hello World*, as much as I like saying hello to the world, it will not serve a purpose for addressing business problems too much.

### A closer look at the requirements

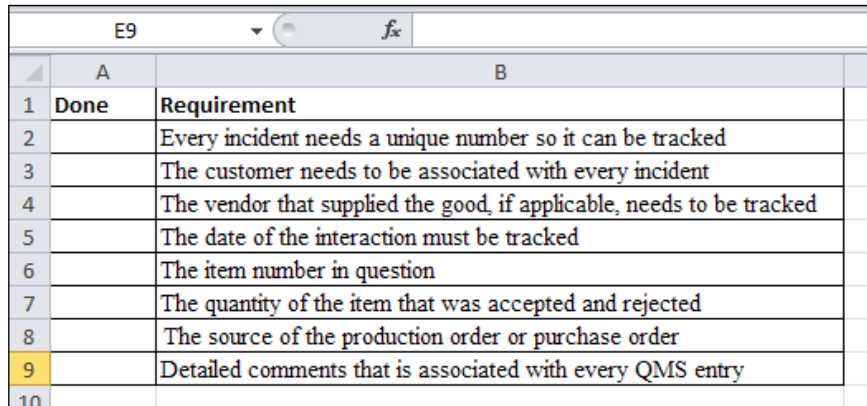
Almost everything you see when you start up Dynamics NAV resides in what we NAV people call **objects**. To access these objects, go ahead and run the Microsoft Dynamics NAV Development Environment and then start up the **Object Designer** by clicking on **Tools | Object Designer** if it's not already open. As described in *Chapter 2, Getting Familiar with Dynamics NAV*, there are seven types of programming objects in Dynamics NAV. In this section, we will be focusing on tables. Without tables, there wouldn't be a place to hold our data.

When creating any new functions or modules in NAV, we always start from the tables. The table is the foundation, as all pages and reports that the users will interact with will be based on a table or multiple tables.

In addition, most of the business validation and rules will be programmed in tables. Why? One of the main reasons is if you put the validation and business rules on the pages, then you would need to replicate that code across any pages that you will develop. This is not a very efficient way of programming in NAV, and in the world of NAV, we're all about efficiency.

Before we start going crazy with our fingers to create an awesome program, we need to make sure that we have a full understanding of the requirements. Nothing is worse than creating an awesome program that the user never asked for.

In the previous chapter, we've defined the requirements for the QMS function that we're going to build. By looking at the requirements in more depth, we can extract the requirements that are related to the table. These requirements are as shown in the following screenshot:



| E9 |      | <i>f_x</i>  |
|----|------|---|
|    | A    | B   |
| 1  | Done | Requirement   |
| 2  |      | Every incident needs a unique number so it can be tracked             |
| 3  |      | The customer needs to be associated with every incident               |
| 4  |      | The vendor that supplied the good, if applicable, needs to be tracked |
| 5  |      | The date of the interaction must be tracked                           |
| 6  |      | The item number in question   |
| 7  |      | The quantity of the item that was accepted and rejected               |
| 8  |      | The source of the production order or purchase order                  |
| 9  |      | Detailed comments that is associated with every QMS entry             |
| 10 |      |   |

It's good to have a checklist of all of the requirements. This way, we can ensure that the program we're creating addresses what the requirements are.

## Where have you seen similar behavior?

After looking at the requirements in the previous screenshot, and assuming you have looked through and spent a few moments clicking around in the system, where have you seen a similar area that allows you to do the following?

- Allows you to have a unique identifying number for the document
- Allows you to enter a customer/vendor number
- Allows you to enter multiple items and quantities

You don't even need to know Dynamics NAV to know this, because every ERP system will have this. The answer is, of course, the sales order and/or the purchase order screens.

The sales and purchase order screens allow you to obtain the order ID to uniquely identify the order of your customer or vendor. It has an area that allows the customer/vendor ID to indicate who the order is for. It also has an area that allows you to enter as many lines of items that you want as well as the associated quantities.

Why are we mentioning this? Well, part of being a great developer is being able to model your solution after what has been done. Look at what the existing program already has, find similar occurrences of what you're trying to create, and either replicate or copy from it.

Since this is a programming book, we will not just copy, paste, and then remove what we don't need. That would be cheating. In this case, there are a lot of features in the `sales_order` function, so by copying and pasting the `sales_order` function, you will spend more time deleting than anything else.

## Finding Similar Functions for Inspiration

This will not stop us from at least looking at how this is structured and replicating it for our project. First, let's take a look at the sales order. Using the skills you learned in *Chapter 2, Getting Familiar with Dynamics NAV 2013*, go ahead and open the **Sales Order** card:

**6001 - Beef House**

**General**

No.: 6001      Order Date: 10/9/2014  
Sell-to Customer No.: 49525252      Document Date: 10/9/2014  
Sell-to Customer Name: Beef House      Requested Delivery Date:  
Sell-to City: Dusseldorf      External Document No.:  
Sell-to State / ZIP Code:      Salesperson Code: JR  
Posting Date: 10/9/2014      Status: Released

**Lines**

| Type | No.    | Description               | Location C... | Quantity | Qty. to Assemb... |
|------|--------|---------------------------|---------------|----------|-------------------|
| Item | 1908-S | LONDON Swivel Chair, blue | GREEN         | 12       |                   |
| Item | 1906-S | ATHENS Mobile Pedestal    | GREEN         | 22       |                   |
| Item | 80100  | Printing Paper            | GREEN         | 20       |                   |

**Sell-to Customer Sal...**

Customer No.: 49525252  
Quotes: 0  
Blanket Orders: 0  
Orders: 2  
Invoices: 0  
Return Orders: 0  
Credit Memos: 0  
Pstd. Shipments: 0  
Pstd. Invoices: 0  
Pstd. Return Rece...: 0  
Pstd. Credit Mem...: 0

**Sales Line Details**

Item No.: 1908-S  
Availability: 90  
Substitutions: 0  
Sales Prices: 0  
Sales Line Discou...: 0

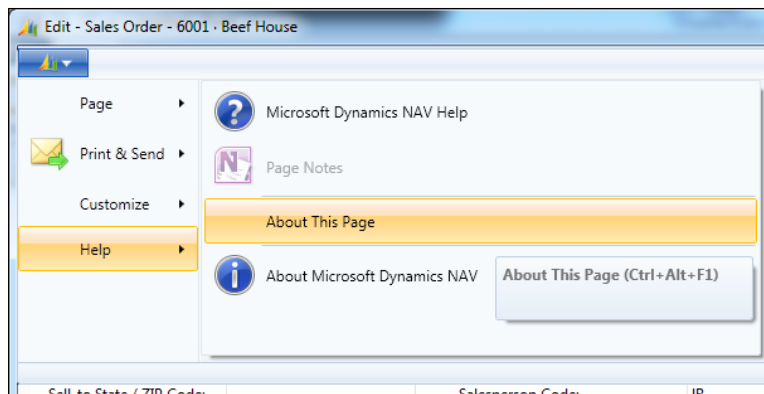
**Notes**

[Click here to create a new note.](#)

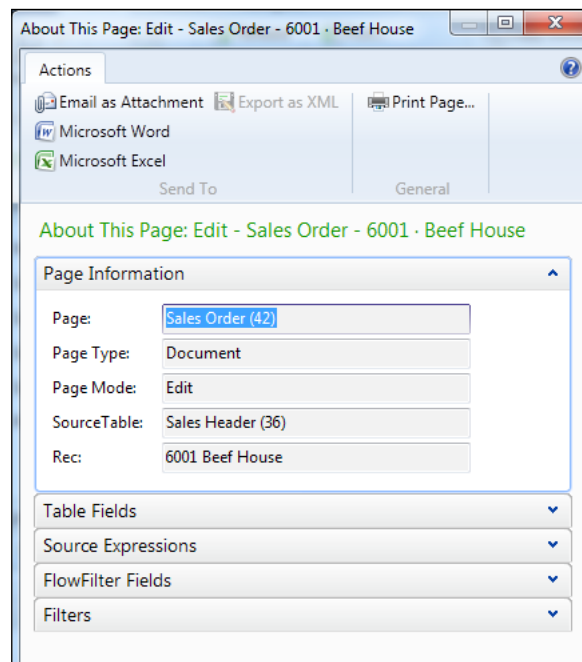
Invoicing: 49525252 | 1M(8D) | 11/9/2014 | No  
Shipping: DE-40593 | 10/9/2014 | Partial  
Foreign Trade: EUR  
Prepayment: 0 | 11/9/2014

OK

Notice that, although not exactly the same, it has pretty much the structure of what we need. To look at the tables associated with the sales order, click on **Help | About This Page**, which is accessible from the Dynamics NAV icon at the upper-left corner, as shown in the following screenshot:



The shortcut for the **About This Page** screen is *Ctrl + Alt + F1*. This screen allows you to see all the details that are within the current page you're working on:



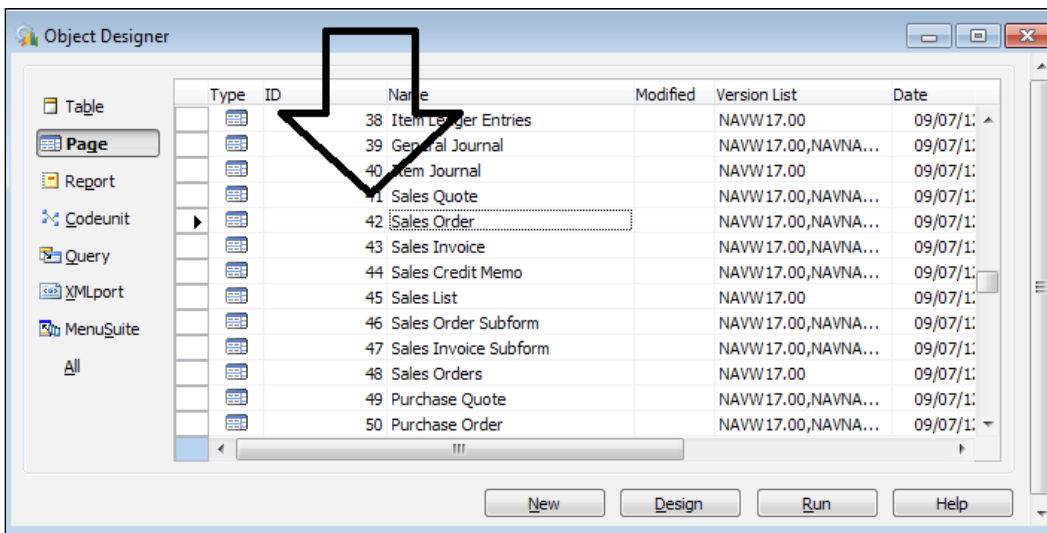
You can expand the **Table Fields**, **Source Expressions**, and other tabs to see what they're about. The main information that we need from this page is the **Page** and **SourceTable** fields.



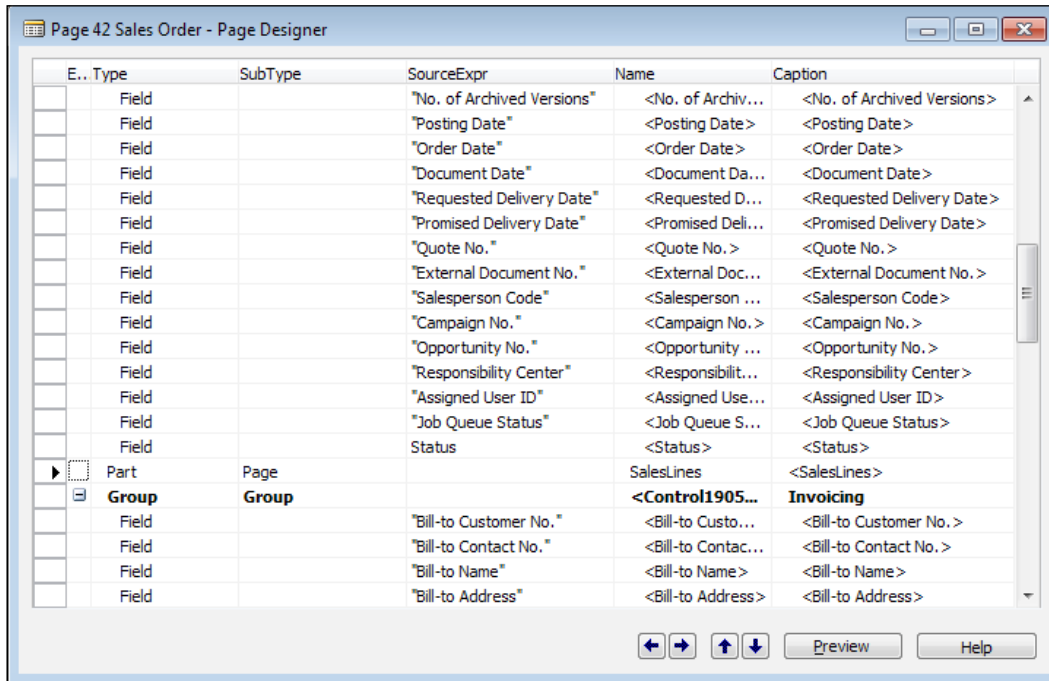
The corresponding IDs next to these values indicate the unique value that is assigned. So, within the **Page** object on the **Object Designer** screen, if you find page **42**, it'll be **Sales Order**. The same goes for **SourceTable**.

The fact that table **36** and page **42** are involved in **Sales Order** is only part of the equation. What about the lines where the item numbers can be entered? The answer lies in page number **42**. You can also press *Ctrl + Alt + F1* on your keyboard to get the sales line information.

Going back to the Dynamics NAV Development Environment, click on **Page** and find the page number **42**:

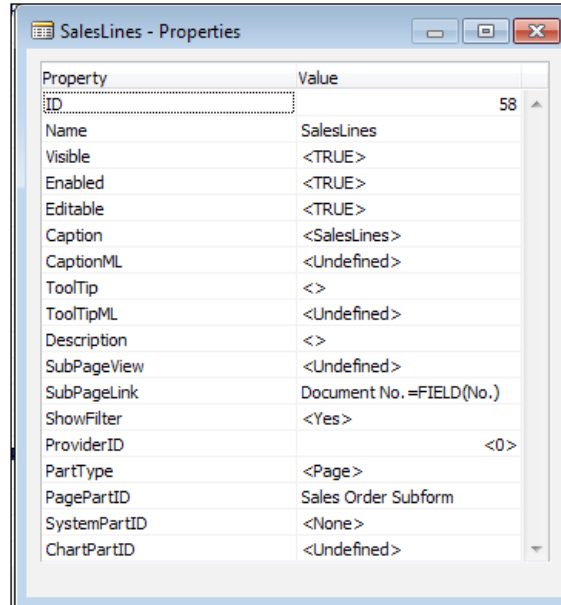


Once you've found it, click on the **Design** button to go to the **Page Designer** screen:

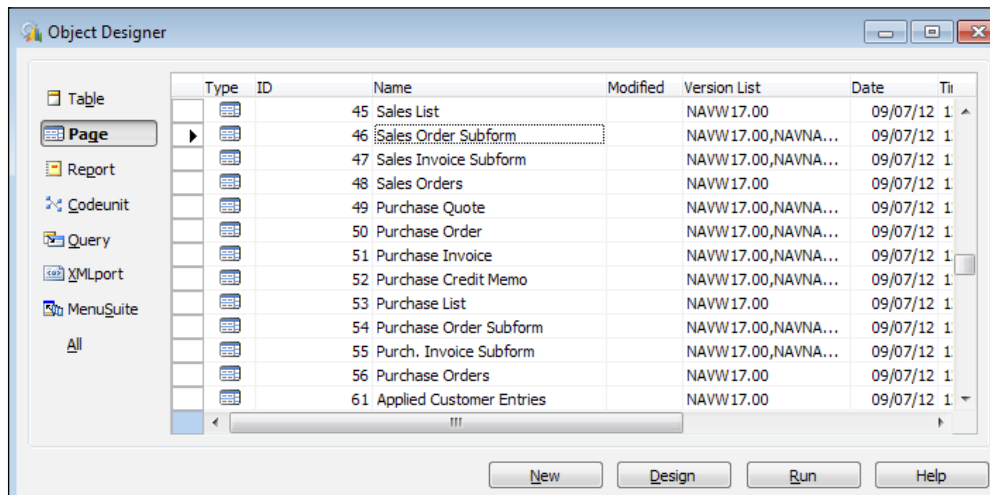


Note that the **Sales Order** page is basically linked to table 36. However, the lines are displayed on this page as a subpage. The subpage is how we get the header and the line behavior on the **Sales Order** page.

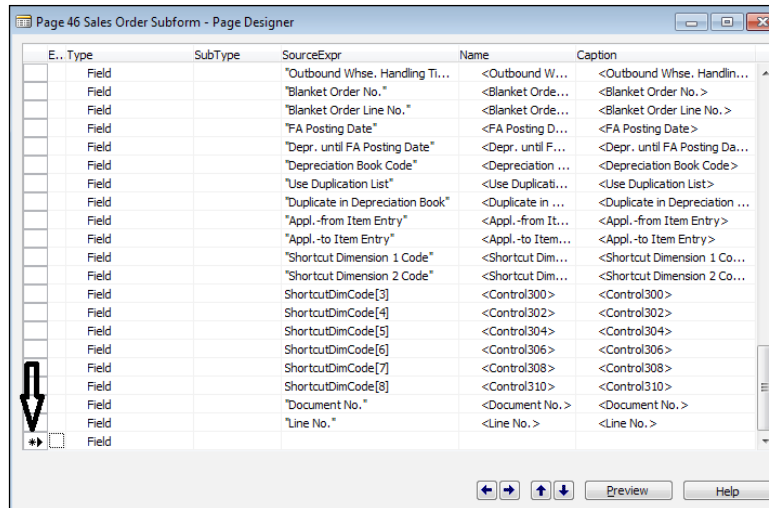
Scroll down to the area where the **Type** column is displayed as **Part**. Go ahead and highlight the line if you want to. To know what page is displaying the line data you have to access the properties for, click on **View | Properties**:



The main information we want from here is the **PagePartID** property. Once we know what the value is, go back to the **Object Designer** screen and find **Sales Order Subform**:

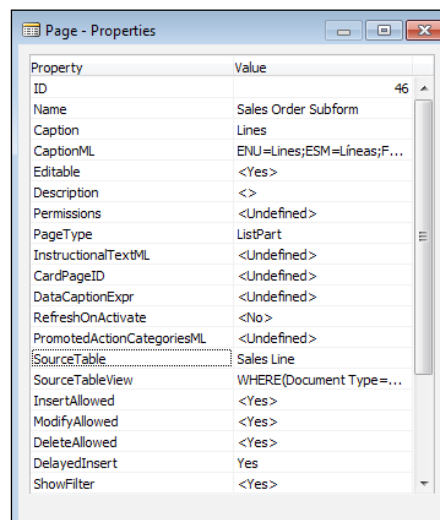


Again, click on **Design** to access the **Page Designer** screen for page **46**. To access the property of this page and not the individual fields, you have to go to the very bottom of the page; basically, keep pressing *Page Down* on your keyboard until you can't go any further:



You'll know you've hit the end when there is an \* sign, as shown in the previous screenshot. The \* sign indicates that it's a new record that has not been inserted or committed into the database.

Click on **View | Properties** and this will give you the properties for the whole page. Next, find the following table behind this page:

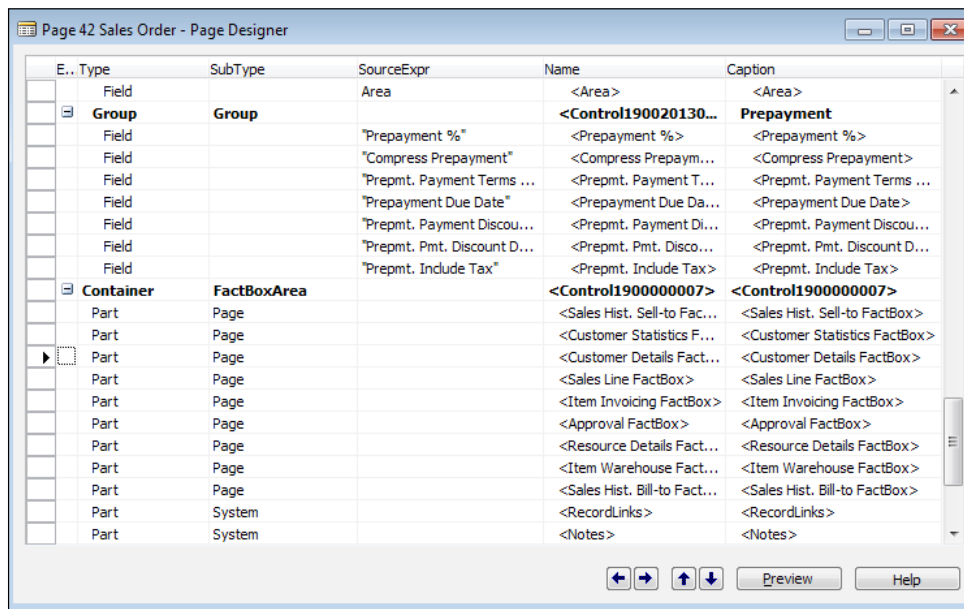


The **SourceTable** property will tell us what table this page is for. Now it's your turn to find the table ID for the **Sales Line** table.

After you find this, you'll know the following objects are responsible for the sales order:

- Table 36 – Sales Header
- Table 37 – Sales Line
- Page 42 – Sales Order
- Page 46 – Sales Order Subform

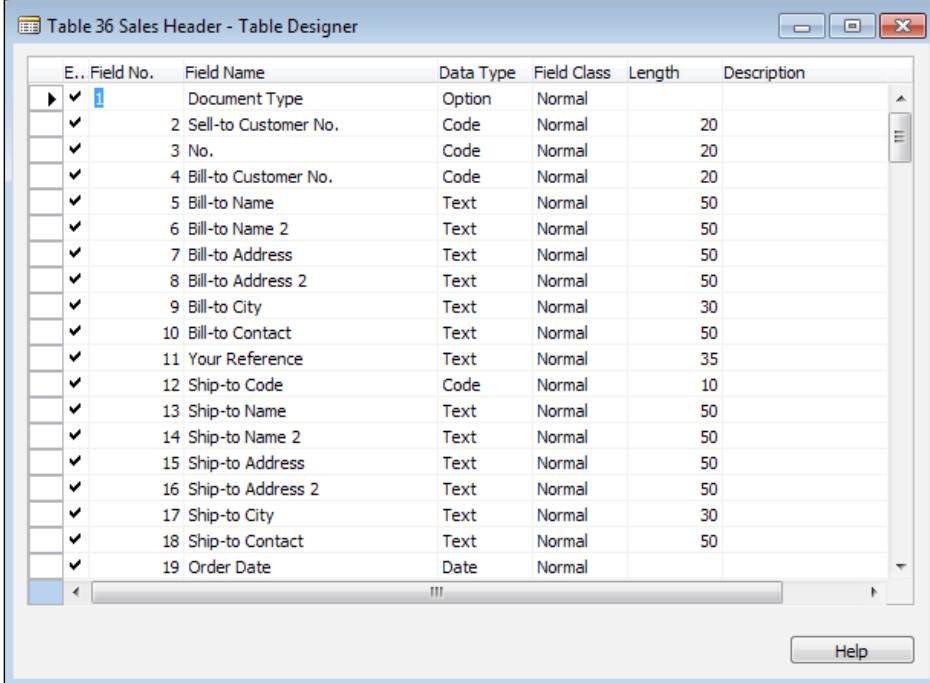
As a bonus, go back to **Sales Order** on page 42 and go down to **FactBoxArea**, and try finding the page ID and the table ID for those FactBoxes:



## A closer look at the Sales Header table (36)

For the main tables that are involved, let's take a look at the relationship between the **Sales Header** table (36) and the **Sales Line** table (37). Don't worry about the tables related to the FactBoxes; as we go through the **Sales Header** and the **Sales Line** tables, we'll see that figuring out the FactBoxes is a piece of cake.

Go back to **Table Designer** for **Table 36** and let's take a deeper look at the composition of this table. Find **Table 36** and click on **Design** to access **Table Designer**:



| E.. Field No. | Field Name           | Data Type | Field Class | Length | Description |
|---------------|----------------------|-----------|-------------|--------|-------------|
| 1             | Document Type        | Option    | Normal      |        |             |
| 2             | Sell-to Customer No. | Code      | Normal      | 20     |             |
| 3             | No.                  | Code      | Normal      | 20     |             |
| 4             | Bill-to Customer No. | Code      | Normal      | 20     |             |
| 5             | Bill-to Name         | Text      | Normal      | 50     |             |
| 6             | Bill-to Name 2       | Text      | Normal      | 50     |             |
| 7             | Bill-to Address      | Text      | Normal      | 50     |             |
| 8             | Bill-to Address 2    | Text      | Normal      | 50     |             |
| 9             | Bill-to City         | Text      | Normal      | 30     |             |
| 10            | Bill-to Contact      | Text      | Normal      | 50     |             |
| 11            | Your Reference       | Text      | Normal      | 35     |             |
| 12            | Ship-to Code         | Code      | Normal      | 10     |             |
| 13            | Ship-to Name         | Text      | Normal      | 50     |             |
| 14            | Ship-to Name 2       | Text      | Normal      | 50     |             |
| 15            | Ship-to Address      | Text      | Normal      | 50     |             |
| 16            | Ship-to Address 2    | Text      | Normal      | 50     |             |
| 17            | Ship-to City         | Text      | Normal      | 30     |             |
| 18            | Ship-to Contact      | Text      | Normal      | 50     |             |
| 19            | Order Date           | Date      | Normal      |        |             |

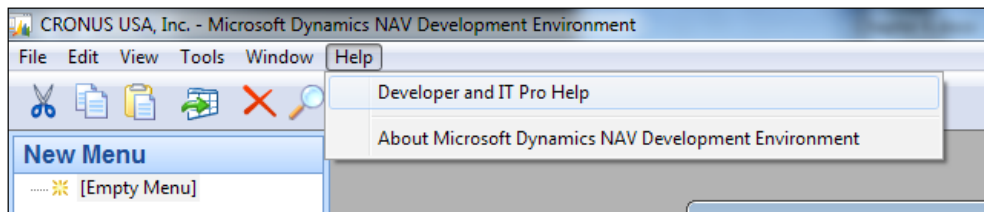
**Table Designer** has the following columns:


- **Enabled:** You can define whether to enable a field or not on the table level.
- **Field No.:** This is a unique ID for this particular field for that particular table.
- **Field Name:** The name you want to give to this field. **DO NOT** assign a "developer" field name. For example, if you want to add a field called Customer Name, don't put it as CustName. Put it as Customer Name. The reason is because the pages and reports will use the field names on the table by default as the caption. Don't double your work!
- **Data Type:** The data type is basically the type of data for this field. For example, if you want to store text in this field, put in **Text**; for values, put in **Date**. There are 17 data types available in Dynamics NAV 2013. We will go through these data types in a while.

- **Field Class:** Dynamics NAV has three different field classes:
  - **Normal:** A normal field.
  - **Flowfield:** A field that's always calculating based on a subtable, typically a ledger table or a table such as `Sales Order Line`. At runtime, it will take a snapshot of the calculated figure.
  - **Flowfilter:** This is a field that will filter the values on the flowfield. On the flowfield calculation, you can define the filters at runtime to be applied to the subtables to give the result. For example, you can set a flowfilter on the posting date, and the results of the flowfield will give you the results for the posting date criteria.
- **Length:** The length of the data type. This only applies to the **Text** and **Code** type fields.
- **Description:** Not used much in standard Dynamics NAV. This is where you can add comments for the fields.

## Data types in Dynamics NAV

As previously mentioned, there are 17 data types in Dynamics NAV. Before you continue, all of these data types are explained in great detail when you go to the **Help** section:



[  Detailed information about the data types in Dynamics NAV can also be found here: [http://msdn.microsoft.com/en-us/library/dd301350\(v=nav.70\).aspx](http://msdn.microsoft.com/en-us/library/dd301350(v=nav.70).aspx) ]

As a matter of fact, you can find any technical terms used in this book in the **Help** section or on the MSDN site, but I will lay these data types out for you as "non-programmer" as possible:

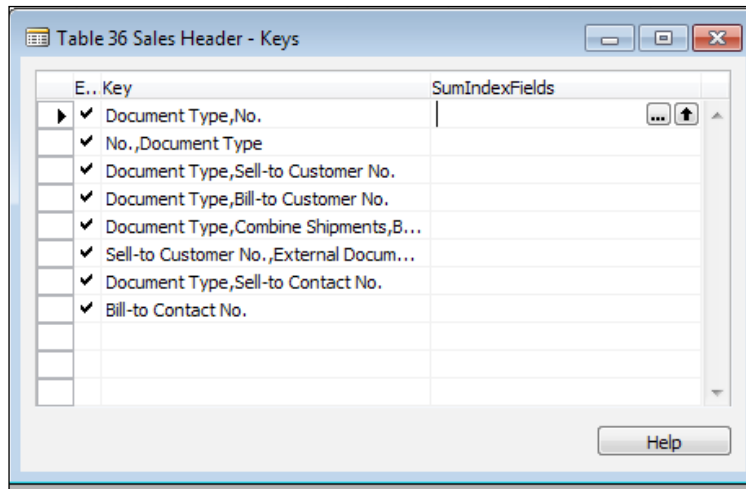
- **BLOB:** Binary Large Objects are typically used to store any files (pictures, Word files, and so on).

- **BigInteger:** Allows you to store numbers from -9,223,372,036,854,775,807 to 9,223,372,036,854,775,807. If your program needs this many numbers, something is wrong.
- **Binary:** Used to store a fixed length of binary data. If you do a quick Google search, this field was discontinued in NAV2009. Going forward, it's best to ignore this data type.
- **Boolean:** This data type has two values; true or false.
- **Code:** This is basically a text field (alphanumeric), but anything you type in here gets converted to uppercase. Essentially, the values you type in here will not be case sensitive.
- **Date:** This data is stored as DATETIME in Microsoft SQL Server. The earliest valid date in SQL Server for a DATETIME data type is 01-01-1753 00:00:00:000.
- **DateFormula:** NAV allows you to input data calculation formulas. So, if you want to store a formula that adds seven days from a particular date, you will put +7D in this field.
- **DateTime:** This stores the date and the time together in one field.
- **Decimal:** A number field that allows decimals.
- **Duration:** Allows you to add or subtract from the **DateTime** data type.
- **GUID:** This gives a unique identifier for a particular object, page, record, and so on. This is based on an algorithm and is guaranteed to be unique.
- **Integer:** Whole numbers. Note that this field should be used in place of **BigInteger** for 99.9999 percent of the time.
- **Option:** This allows you to create a drop-down menu for the user to choose a value that you predefine.
- **RecordID:** This field stores the table ID and the unique identifier or the primary key of the table.
- **TableFilter:** This field shows what filters are being made when you predefine filters at the SQL level.
- **Text:** This is an alphanumeric field. You can basically type anything MS SQL Server allows you to.
- **Time:** A field that stores the time.



## Primary key and indexes

The primary key for a table is defined by going to **View | Keys**:



There are multiple keys or indexes defined here. In Dynamics NAV, the first key defined is automatically assigned as the primary key. In the case of the **Sales Header** table, the **Document Type** and **No.** fields are composite keys. Basically, these two fields are needed to keep the records in the table unique.

All of the subsequent keys that are defined are basically indexes that allow us to sort the content differently.

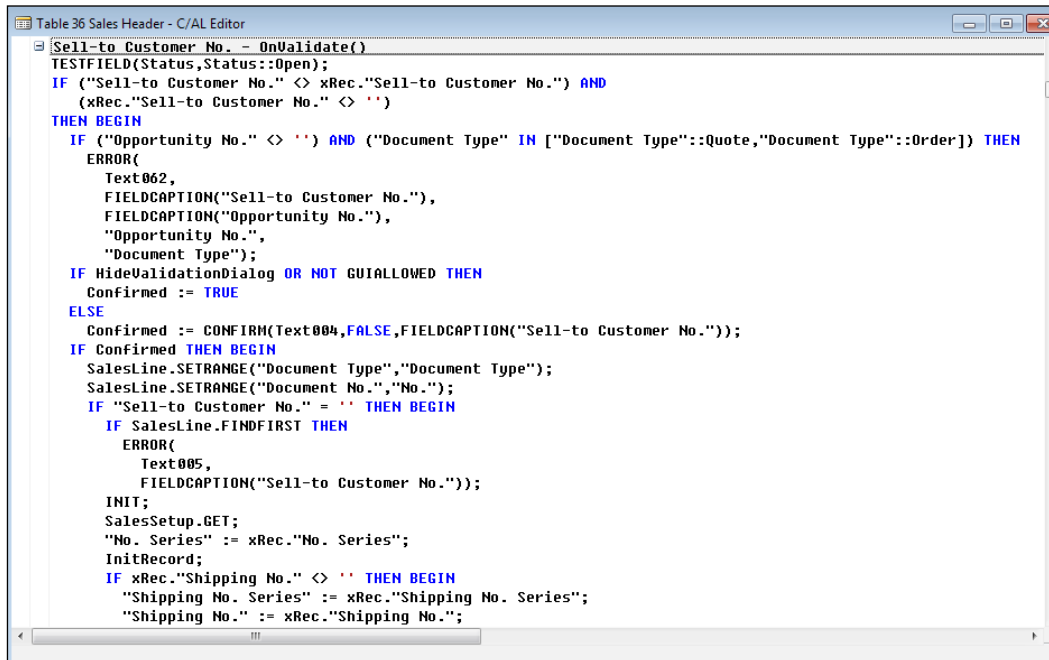
The **SumIndexFields** column allows us to quickly and efficiently calculate the sum of a column. You would typically use this if an index is used for calculation using a flowfield or through code.

## Looking at C/AL behind the table (the code)

When you entered **Sell-to Customer No.** on the sales order, did you noticed how the entire name, address, and other information became populated? Unfortunately, this doesn't happen by magic. There is actually coding behind this behavior.

In order to look at the **C/AL** code behind the table, you will need a developer license or an end user license with the **Application Builder** add-on. The alternative is to use the 30-day free trial on the cloud site described in *Chapter 1, Getting Dynamics NAV 2013 on Your Computer – For (Almost) Free.*

To access the coding in Dynamics NAV, click on the **Sell-to Customer No.** field in **Table Designer** and press **F9**. You can also access it by clicking on **View | C/AL Code**:



```

Table 36 Sales Header - C/AL Editor
Sell-to Customer No. - OnValidate()
TESTFIELD(Status,Status::Open);
IF ("Sell-to Customer No." <> xRec."Sell-to Customer No.") AND
(xRec."Sell-to Customer No." <> '')
THEN BEGIN
  IF ("Opportunity No." <> '') AND ("Document Type" IN ["Document Type"::Quote,"Document Type"::Order]) THEN
  ERROR(
    Text002,
    FIELDCAPIION("Sell-to Customer No."),
    FIELDCAPIION("Opportunity No."),
    "Opportunity No.",
    "Document Type");
  IF HideValidationDialog OR NOT GUIALLOWED THEN
  Confirmed := TRUE
ELSE
  Confirmed := CONFIRM(Text004,FALSE,FIELDCAPIION("Sell-to Customer No.));
  IF Confirmed THEN BEGIN
    SalesLine.SETRANGE("Document Type","Document Type");
    SalesLine.SETRANGE("Document No.,"No.");
    IF "Sell-to Customer No." = '' THEN BEGIN
      IF SalesLine.FINDFIRST THEN
      ERROR(
        Text005,
        FIELDCAPIION("Sell-to Customer No.));
      INIT;
      SalesSetup.GET;
      "No. Series" := xRec."No. Series";
      InitRecord;
      IF xRec."Shipping No." <> '' THEN BEGIN
        "Shipping No. Series" := xRec."Shipping No. Series";
        "Shipping No." := xRec."Shipping No.";

```

Most of the coding that you will use will be on the `OnValidate` section or trigger. Scroll down a couple of pages and you'll see the name and addresses being populated. Most of the business logic is coded in the tables, so tables will be the place where you see the majority of the coding.

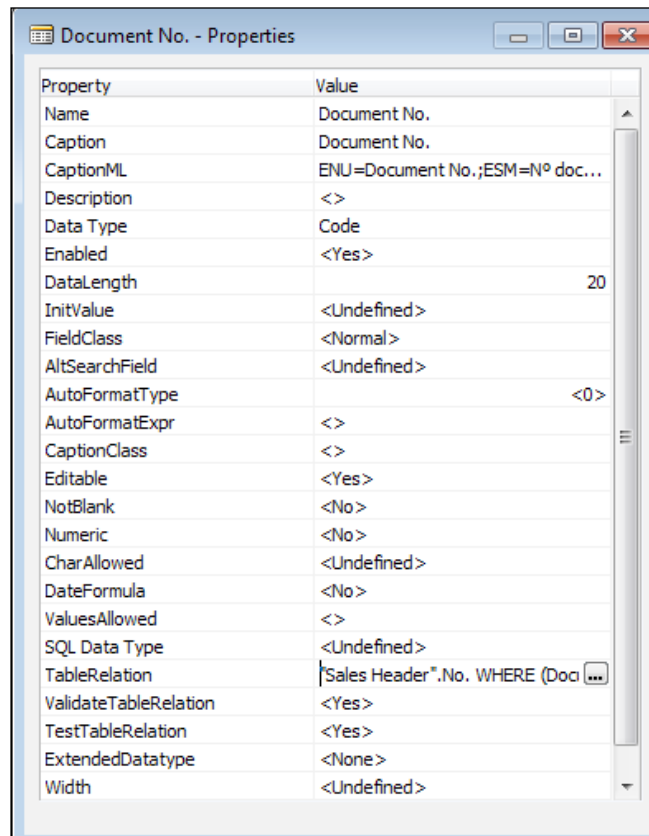
The instruction on coding is a whole book in itself; for what we're trying to do, we will only focus on the code needed to create our project in the subsequent chapters. If you have to learn everything about C/AL (the programming language), here's the link for you to get all the details: <http://msdn.microsoft.com/en-us/library/dd355277.aspx>.

Also important is where you put the C/AL code once you know the syntax. In Dynamics NAV, within each object, an action to tell the system to run the code is called a **trigger**. For the full definition of the triggers available on the table, please go to this link: [http://msdn.microsoft.com/en-us/library/dd354905\(v=nav.70\).aspx](http://msdn.microsoft.com/en-us/library/dd354905(v=nav.70).aspx).

## Table relations

We already know that the `Sales Line` table is related to the `Sales Header` table. But how do we tell Dynamics NAV about the relationship between these two tables?

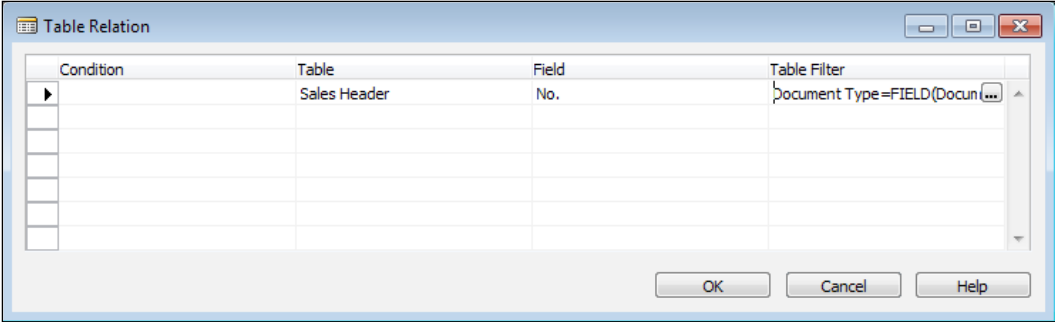
The table relation for the fields is defined in the properties of the field. Go ahead and click on the **Document No.** field in the `Sales Line` table and find the property called **TableRelation**:



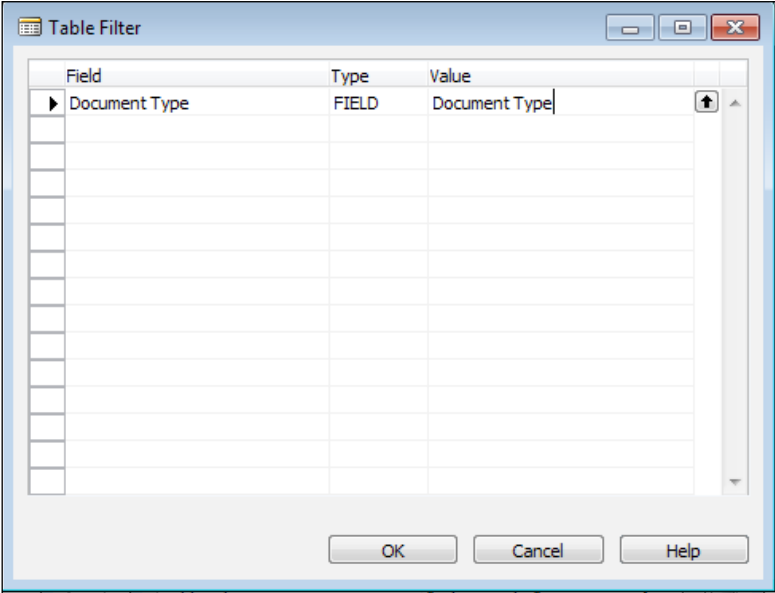
Click on the **AssistEdit** button to bring up the **Table Relation** screen. The **AssistEdit** button is the button that looks similar to three dots:



Clicking on the **AssistEdit** button will tell us the table relation that's associated with that field:



From the **Table Relation** screen, click on the **AssistEdit** button to bring up **Table Filter**; this will tell us the relation criteria:



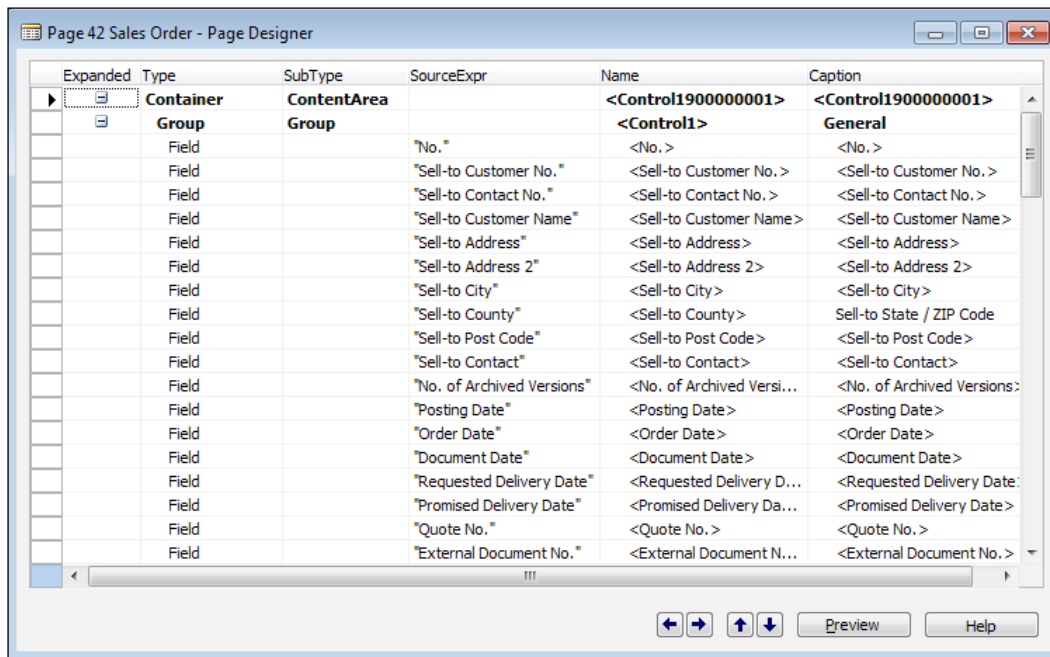
From the previous screenshots, the sentence you would construct would be this: *The Document No. field is related to the Sales Header No. field, where the value on Document Type on the Sales Line table has to equal to the Document Type on the Sales Header table.*

Make sure to take special note of the primary key in the `Sales Line` table. Notice the similarity with the field used as the primary key in the `Sales Header` table? The primary keys are `Document Type`, `Document No.`, and `Line No.`.

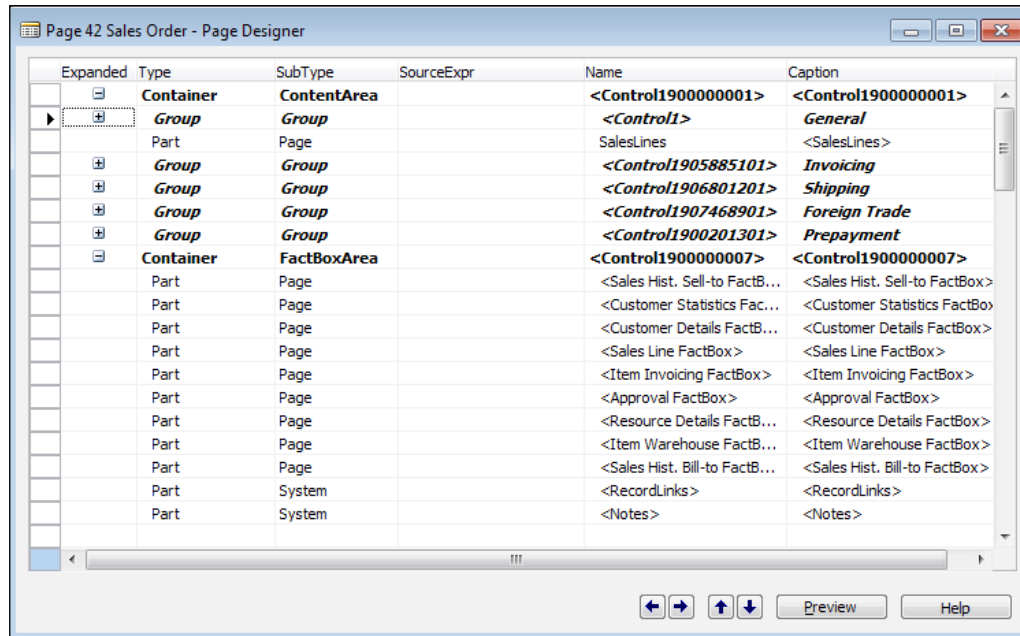
`Line No.` is used to uniquely identify each item in your table. The system will automatically assign a line number; we will take a look at how the system does this when we look at pages.

## A closer look at the Sales Order page (42)

From **Object Designer**, find the `Sales Order` page and click on **Design**:



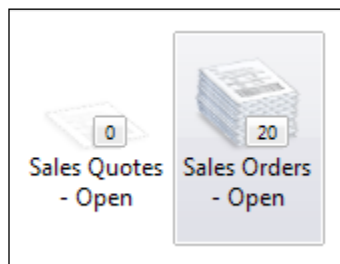
This screen is the expanded view of **Page Designer** for the **Sales Order** page:



When we collapse it, you'll notice that it will be easier for us to see how it's laid out. The columns that you see are as follows:

- **Expanded:** This shows whether the line is expanded or collapsed.
- **Type:** This is the control type. There are four control types in NAV:
  - **Container:** This is basically a place in which all of the controls are contained.
  - **Group:** This is where we group the common data.
  - **Field:** A field on the table or a variable that you define.
  - **Part:** Typically used to show different pages such as FactBoxes, or a sales line.

- **SubType:** Based on the information specified on the type, there will be a number of subtypes you can choose from.
- For **Container**, you can choose from the following subtypes:
  - **ContentArea:** Usually at the start of a page, where you define the main contents of the information being displayed.
  - **FactBoxArea:** A place where you put FactBoxes.
  - **RoleCenterArea:** If you're creating a new role center, you would contain your information using this subtype. An example of a role center you can check out is the page **Order Processor Role Center** (page 9006), which is also the standard role that you're logged in as.
- For **Group**, you can choose from the following subtypes:
  - **Group:** Used to group the data together in FastTabs.
  - **Repeater:** Used to display a list of data. An example of this is in **Sales List** (page 45). This page basically lists all of the sales documents in the system.
  - **CueGroup:** You know that cute stack of paper when you first log in? In this example, this stack shows you the number of open Sales Orders, or the sales orders that has not been processed yet.



An example of the use of **CueGroup** is **SO Processor Activities** (page 9060).

- **FixedLayout:** When you group the field using **FixedLayout**, the columns will be predefined by you and the user will not be able to customize it. An example of this is in **Customer Statistics** (page 151).

- **GridLayout:** By default, the fields you add in a card page will be organized into two columns. Using this subtype, you will be able to organize, for example, the layout display for city, state, and zip code. Unfortunately, there are no examples of this in standard Dynamics NAV; however, it's described in detail here: [http://msdn.microsoft.com/en-us/library/hh168529\(v=nav.70\).aspx](http://msdn.microsoft.com/en-us/library/hh168529(v=nav.70).aspx).
  - For **Part**, you can choose from the following subtypes:
  - **Page:** Basically, this displays the page of your choice.
  - **System:** System objects are **Notes**, **RecordLinks**, **Outlook**, and **MyNotes**.
  - **Chart:** There are a number of predefined charts you can associate with the page you're displaying. You can create your own chart with the step-by-step example here: <http://blogs.msdn.com/b/nav/archive/2008/08/20/nav-2009-how-to-generate-charts-kpis.aspx>. Note that creating your own charts will not be covered by this book.
- **SourceExpr:** If your **Type** is **Field**, then **SourceExpr** is where you define where the information is coming from. The data will come from a table, a variable, or a function.
  - **Name:** The name of the control. This will be referenced when you're coding specifics. Most of the time, you will leave this alone.
  - **Caption:** The caption that's going to be displayed at runtime.

## Looking at the properties

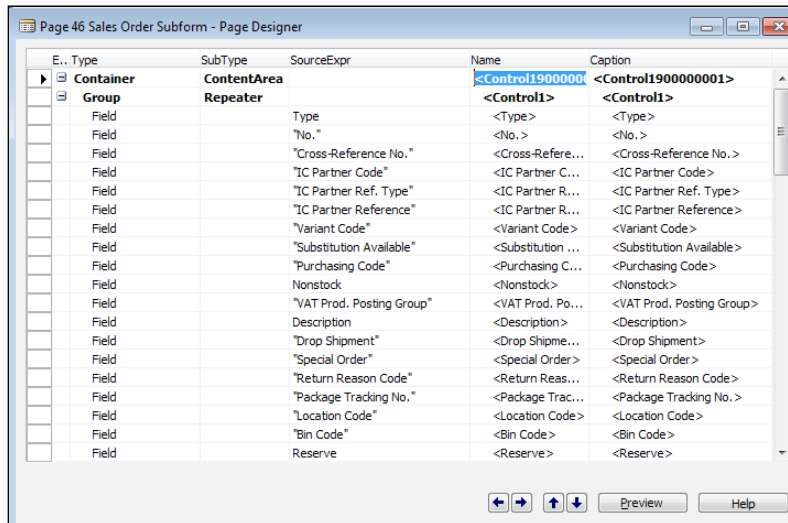
Each of the controls are listed on the **Page Designer** screen; for each control, there are multiple properties. To access the property per control, just click on the control you want and click on **View | Properties**.

We will highlight the properties we will use most often in the next chapter when we're creating a page. For a complete list of the properties on a page, you can go here: <http://msdn.microsoft.com/en-us/library/dd355281.aspx>.



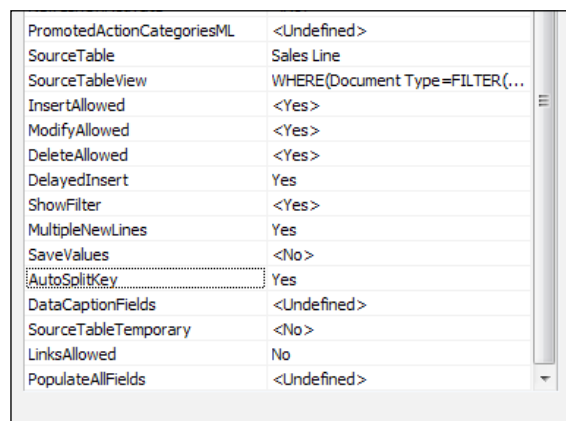
## A closer look at the Sales Order Subform page (46)

Using the same methods we learned in the earlier chapter, go to **Page Designer** for page 46:



Keep pressing the *Page Down* key until you get to an empty line, and then click on **View | Properties**.

One important property of this page that we want to highlight is the **AutoSplitKey** property:



As we discussed earlier, the primary keys for `Sales Line` are `Document Type`, `Document No.`, and `Line No.`. So how does Dynamics NAV know how to assign a number automatically? The answer is in this **AutoSplitKey** property. For any line table where the user will be entering data, this property is typically used.

In order for this property to work, the `Line No.` field (or any integer field) must be the last field on your composite primary key.

## Looking at C/AL on the page (the code)

You can put code in every object in NAV. When you go into the C/AL coding for pages, you'll notice that it's pretty empty. Again, most of the coding and the business logic have been built into the table level. There is some coding on pages to control the behavior of the page, but you will find most of the coding in the tables.

## Summary

Any developer or implementer can create something. The successful ones are the ones that take the existing formula and use it for their own use. Most of the design and leg work has already been done for you. There's really no need to re-invent the wheel.

Another important reason why you should try to find similar existing functions is the design aspects of NAV. When the users are using the system and using the function that you will masterfully create, the trick is not to remind them that this function was "bolted" on. Remember what we say about the consistency in the user design; it absolutely needs to be consistent to speed up training and user adoption.

This chapter goes over how to dig into existing functions and features you would like to replicate into your project. Once you find these similar functions, you will be able to model the solution after what's already been created.



# 6

## Creating the Application – Tables

*"Between stimulus and response there is a space. In that space is our power to choose our response. In our response lies our growth and our freedom."*

*– Viktor E. Frankl*

We've gotten the user requirements solidified. We have found an existing function within Dynamics NAV that we can use as a template for what we need to do to address the user requirements. Now we get to the fun part, the actual creation of the application.

Although creating the application is fun, this step should not be the beginning of your project. Every user requirement that comes through needs to be scrutinized and analyzed before it can be customized. Most developers make the mistake of programming it first and asking questions later. Don't be that guy! Don't waste your own time and other people's time having to rework your programs because the needs are not met.

In our case, we've gone through and identified the business problem, the user requirements (that the users signed-off on), and as a bonus, we found a template from which we can draw our inspiration. Only then can we even begin to talk about designing and creating the application.

There are a couple of steps involved in creating the application for the users:

- Creating the tables to hold our data
- Creating the pages to allow the user to interface with the data
- Creating reports to display the data
- Adding business logic to our application to validate the data

In this chapter, we'll focus on creating the tables to hold our data. Without it, you will not be able to create pages and reports.

## Creating the table and identifying the primary key(s)

For each table, there needs to be a unique identifier, or what's called a primary key. The primary key allows us to differentiate each record so we can get the proper content when we find that unique ID.

Let's bring back our user requirements again and see what we need to do.

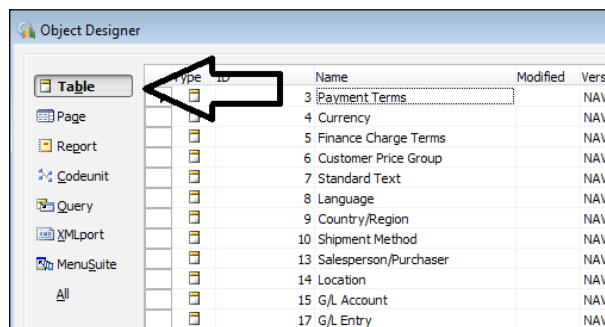
|    | A    | B   |
|----|------|---|
| 1  | Done | Requirement   |
| 2  |      | Every incident needs a unique number so it can be tracked                           |
| 3  |      | The customer needs to be associated with every incident                             |
| 4  |      | The vendor that supplied the good, if applicable, needs to be tracked               |
| 5  |      | The date of the interaction must be tracked   |
| 6  |      | The item number in question   |
| 7  |      | The quantity of the item that was accepted and rejected                             |
| 8  |      | The source of the production order or purchase order                                |
| 9  |      | Detailed comments that is associated with every QMS entry                           |
| 10 |      | There can be 1 or more items per incident   |
| 11 |      | Any QMS document that has been resolved will need to be marked as so                |
| 12 |      | The open QMS documents and the closed ones should be displayed as a separate screen |
| 13 |      | A report needs to be created on the feedback per item                               |

Looking at the requirements, which would you choose to be the unique identifier or the primary key? Every incident needs a unique number so it can be tracked.

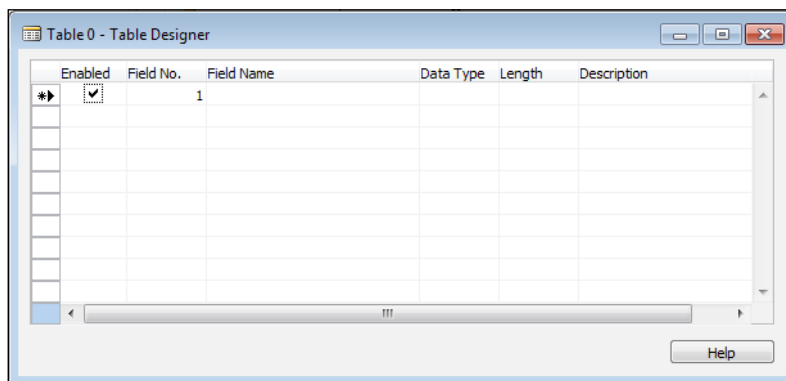
Right away, this requirement should jump out at you saying "USE ME AS A PRIMARY KEY!" By knowing the primary key of the table we're going to create, we can begin to create our table.

Let's do it! On the **Object Designer** screen, do the following:

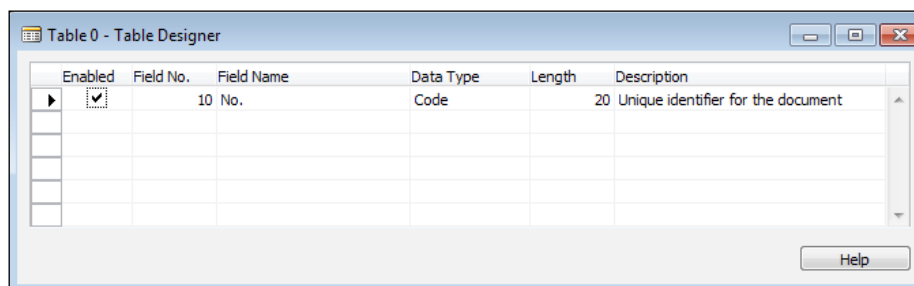
1. Click on **Table**.



2. Click on **New** and you will get the **Table Designer** screen.



For the organization of the table, let's create the fields for our primary keys first. Enter the information as shown in the following example:



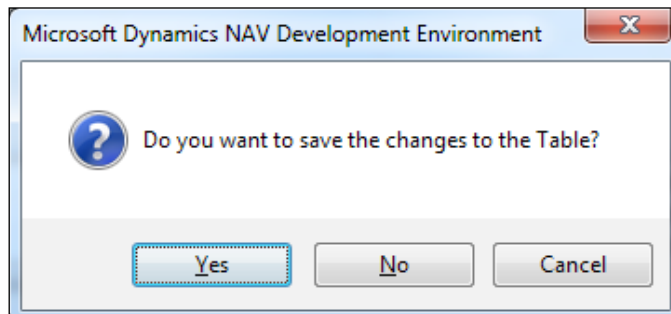
It's good practice to increment **Field No.** by **10** instead of by **1**. The reason is if you want to add a field in between Field No. 10 and Field No. 20, you can add it using Field No. 15, whereas if you used Field No. 1 and Field No. 2, you will not be able to add anything in between.

We will use **No.** as the field for our unique identifier allowing alphanumeric characters with **Length** as **20**. This is very similar to how the `Sales Header` table is set up with the exception of `Document Type`. As our requirement does not require us to separate the document type of the complaint, it's not necessary to create this field.

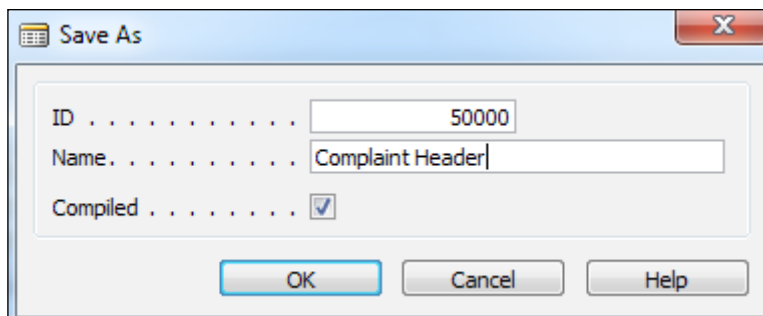
## Saving, compiling, and running our table

Before we continue, the object ID will assume that you're using the cloud environment or that you have an end user Dynamics NAV license.

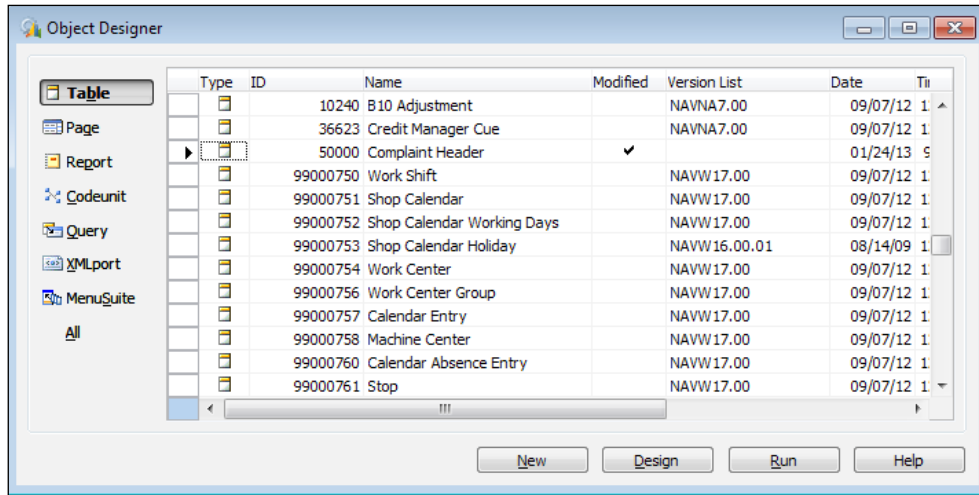
1. Keep pressing the *Esc* key until you see the following screen.  
You can also access this screen by closing **Table Designer** or by navigating to **File | Save As**.



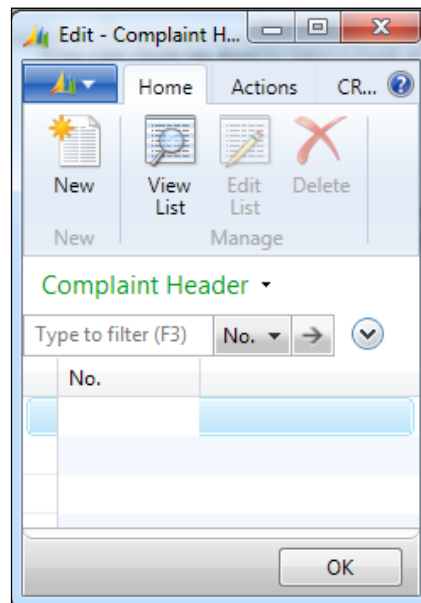
2. Click on **Yes** and you will be prompted to assign a table **ID** and **Name** for this table.



3. After you click on **OK**, when you look at the table objects in **Object Designer**, you will be able to see the table you created.



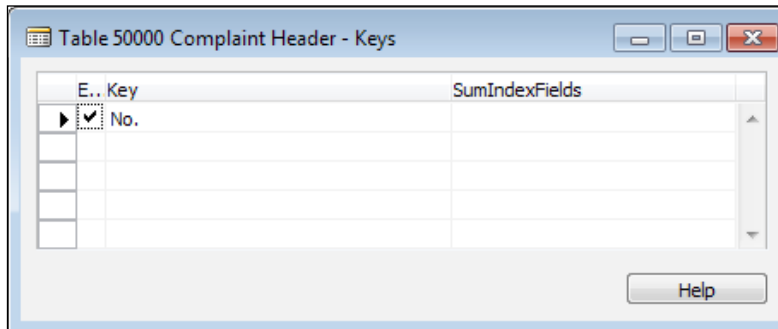
4. You can click on **Run** to take a look at what you've created.



Unfortunately, there is nothing too exciting. What you've basically done is created a table with just the **No.** field. Try entering some data in there and see if you can enter line rows with the same value. You can't.



- Go ahead and exit out of the table to go back to **Object Designer**. Click on **Design** to go to **Table Designer** for table **50000**. If you navigate to **View | Keys**, you'll notice there's a key that's automatically created for you:



## Primary keys

Every table that is created needs a primary key. By default, NAV will assume that the first field that you create for this table is the primary key. So when you save the table, Dynamics NAV will assign the first field, in our case the **No.** field, as the primary key. This is why you'll get an error if you try to enter two rows with the same value.

## Checking our requirements list

Now let's look at our requirements list again. We've satisfied the first condition. Let's mark it off:

|    | A    | B   |
|----|------|---|
| 1  | Done | Requirement   |
| 2  | x    | Every incident needs a unique number so it can be tracked                           |
| 3  |      | The customer needs to be associated with every incident                             |
| 4  |      | The vendor that supplied the good, if applicable, needs to be tracked               |
| 5  |      | The date of the interaction must be tracked   |
| 6  |      | The item number in question   |
| 7  |      | The quantity of the item that was accepted and rejected                             |
| 8  |      | The source of the production order or purchase order                                |
| 9  |      | Detailed comments that is associated with every QMS entry                           |
| 10 |      | There can be 1 or more items per incident   |
| 11 |      | Any QMS document that has been resolved will need to be marked as so                |
| 12 |      | The open QMS documents and the closed ones should be displayed as a separate screen |
| 13 |      | A report needs to be created on the feedback per item                               |

That was easy! Let's take a look at the other requirements and get them knocked-off of our list as well.

## Adding new fields to the tables

Go back to the **Table Designer** screen for our **Table 50000**. The requirement list states that we need the customer and vendor information. We also need the date on which that document is created. So let's do that.

| Enabled                             | Field No. | Field Name    | Data Type | Length | Description                        |
|-------------------------------------|-----------|---------------|-----------|--------|------------------------------------|
| <input checked="" type="checkbox"/> | 10        | No.           | Code      | 20     | Unique identifier for the document |
| <input checked="" type="checkbox"/> | 20        | Customer No.  | Code      | 20     | Looks up to Customer table         |
| <input checked="" type="checkbox"/> | 30        | Name          | Text      | 50     |                                    |
| <input checked="" type="checkbox"/> | 40        | Address       | Text      | 50     |                                    |
| <input checked="" type="checkbox"/> | 50        | Address 2     | Text      | 50     |                                    |
| <input checked="" type="checkbox"/> | 60        | City          | Text      | 30     |                                    |
| <input checked="" type="checkbox"/> | 70        | Post Code     | Code      | 20     |                                    |
| <input checked="" type="checkbox"/> | 80        | County        | Text      | 30     |                                    |
| <input checked="" type="checkbox"/> | 90        | Contact Name  | Text      | 50     |                                    |
| <input checked="" type="checkbox"/> | 100       | Vendor No     | Code      | 20     | Looks up to Vendor table           |
| <input checked="" type="checkbox"/> | 110       | Vendor Name   | Text      | 50     |                                    |
| <input checked="" type="checkbox"/> | 120       | Document Date | Date      | 50     | The date the document is entered   |
| <input checked="" type="checkbox"/> | 130       | Resolved      | Boolean   | 50     |                                    |

**The Data Type and Length** of these fields mimic the `Customer` table. Since this data is related, you have to ensure that **Data Type** and **Length** are the same, or else you will get a lot of errors when you're trying to copy data between these tables and fields. If you go back and look at the `Sales Header` table, you'll see that the **Data Type** and **Length** of the fields are the same as the `Customer` table.



It's generally a very bad idea to try to expand the length or change the data type of the fields in the standard Dynamics NAV tables. The reason is because the standard logic and business rules are all based on the standard data type and field length. Changing them may cause serious damage to your Dynamics NAV database. If, for whatever reason, you need to change the standard fields, consult your Dynamics NAV partner or a highly-experienced Dynamics NAV professional.

Notice that we've purposely set **Field No.** 10 numbers apart. The reason is if we want to add a field in between the **Name** and **Address** fields, we will now be able to do so.

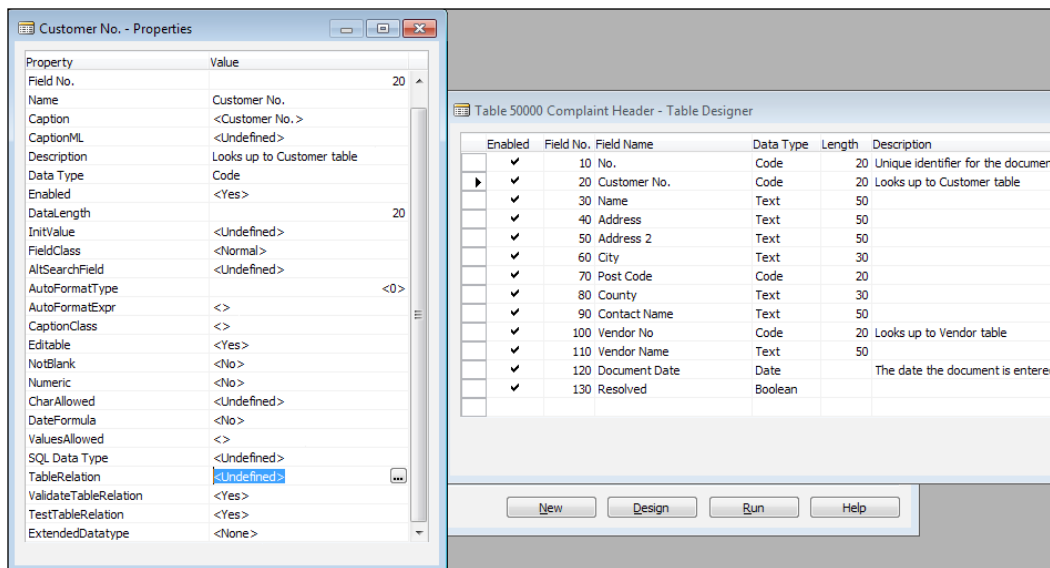
Before we continue, let's save and compile the table. Navigate to **File | Save** and click on **OK** on the confirmation message. Being a developer yourself, you know that saving frequently will prevent keyboard rage.

## Defining table relations in fields

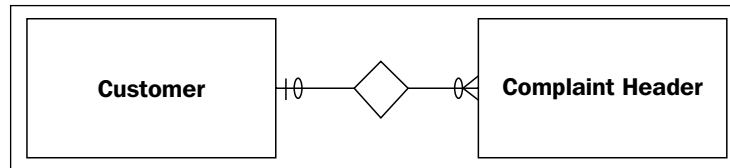
In our requirements described previously, we specified that the program needs to be able to specify the customer and the vendor information. The customer and vendor information must be valid, or else it really defeats the whole purpose of keeping track of the appropriate information.

To ensure that the user enters the relevant customer, we will need the user to be able to select from an existing list of customers that's already defined in the `Customer` table. In order to enforce this, we will need to define the table relationships to the master table data or set up tables within each field.

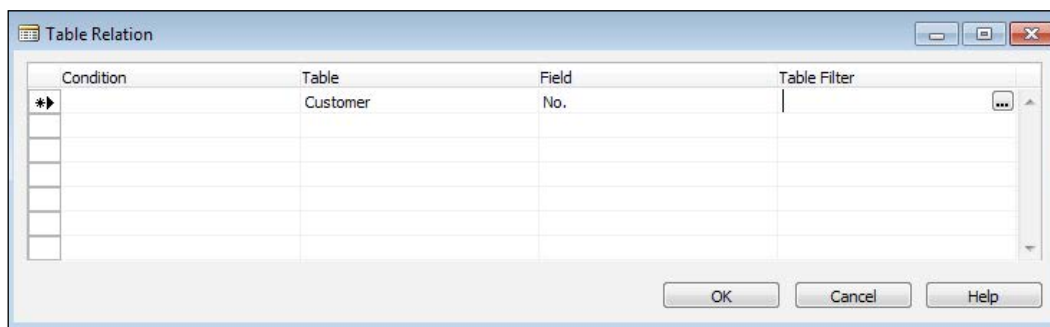
1. Go back to the **Table Designer** screen for our **Table 50000** and click on the **Customer No.** row. Then, navigate to **View | Properties**.



- The table relationship we're looking for will be zero to many. So a customer can occur zero times or more on the `Complaint Header` table and the `Complaint Header` table can have none or one occurrence of a customer number.



- Click on the **AssistEdit** button and bring up the **Table Relation** screen for this field. Again, following our rules, the table that we will be looking up to is the `Customer` table. The field that we're looking for is the **No.** field on the `Customer` table. Your table relation should look like the following screenshot:

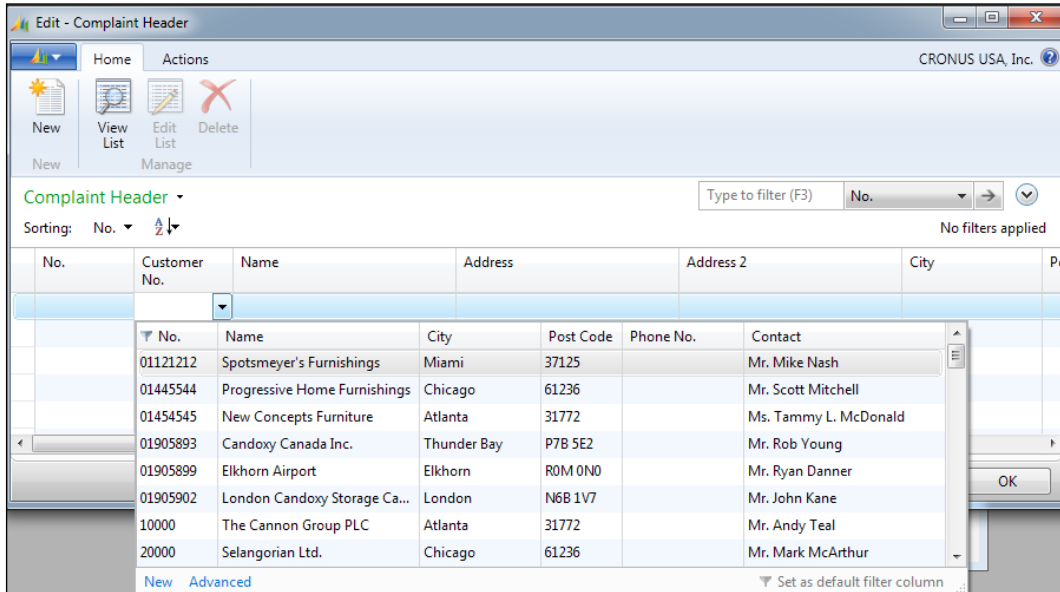


- Click on **OK** and it will change the value of the **TableRelation** property from **<Undefined>** to **Customer.No.** Let's close this screen and do the same for the **Vendor No.** field.

The table relationship we're looking for on the **Vendor No.** field will be zero to many as well. So a vendor can occur zero or more number of times in the `Complaint Header` table and the `Complaint Header` table can have none or one occurrence of a vendor number.

- To verify that our table relation was created successfully, exit the **Table Designer** screen for our table. When it prompts you to save, click on **Yes**.

- Click on **Run** from the **Object Designer** screen to run our table. To check if our table relation is working properly, click on the field that we modified and see if we can look up the relevant information.



Once we've verified that we can do that, let's move on!



For more information on defining table relationships in Dynamics NAV, you can find details at <http://msdn.microsoft.com/en-us/library/dd338639.aspx>.

## Creating the Complaint Line table

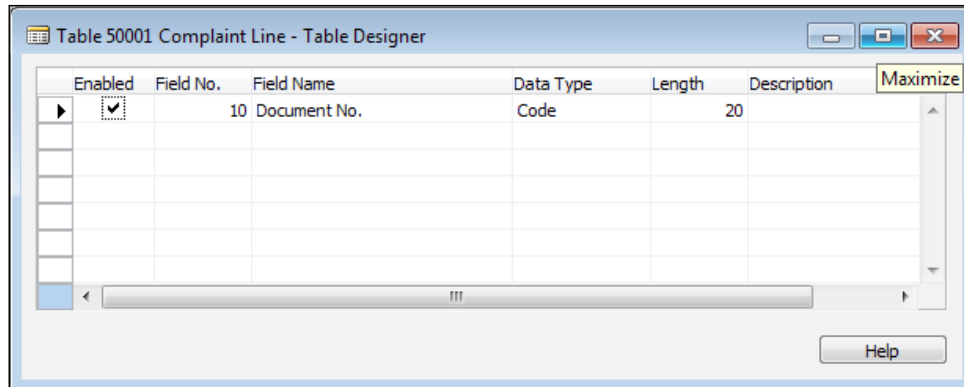
Alright! Let's take a look at our requirement list and see what we've accomplished so we can get them marked off as follows:

|    | A    | B   |
|----|------|---|
| 1  | Done | Requirement   |
| 2  | x    | Every incident needs a unique number so it can be tracked                           |
| 3  | x    | The customer needs to be associated with every incident                             |
| 4  | x    | The vendor that supplied the good, if applicable, needs to be tracked               |
| 5  | x    | The date of the interaction must be tracked   |
| 6  |      | The item number in question   |
| 7  |      | The quantity of the item that was accepted and rejected                             |
| 8  |      | The source of the production order or purchase order                                |
| 9  |      | Detailed comments that is associated with every QMS entry                           |
| 10 |      | There can be 1 or more items per incident   |
| 11 | x    | Any QMS document that has been resolved will need to be marked as so                |
| 12 |      | The open QMS documents and the closed ones should be displayed as a separate screen |
| 13 |      | A report needs to be created on the feedback per item                               |

The next thing on our requirement list will require a different table. How do we know this? Because we've thoroughly looked through the `Sales Header` and `Sales Line` tables and we've seen what information should go where.

The requirement list says that we need to have one or more items per incident. This should be the leading indicator for you to know that the information related to the item belongs in the line table.

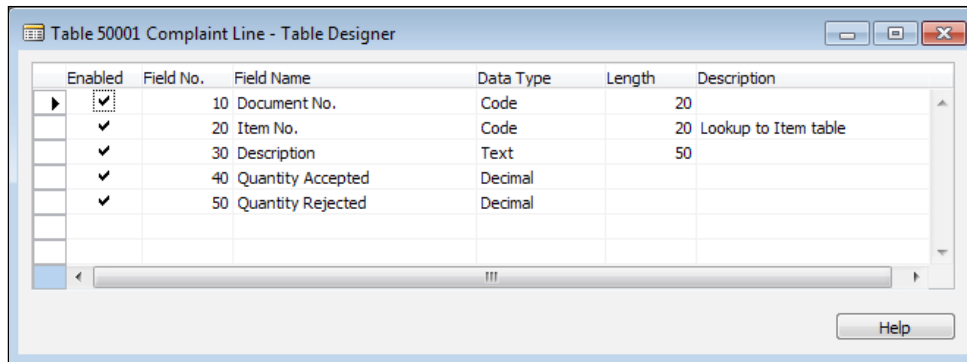
In **Object Designer**, create a new table called **Complaint Line** with the field **Document No.**. The **Data Type** should be **Code** with **Length** as **20**; again, this should be exactly the same as the **No.** field on the **Complaint Header** table. As you may have guessed, these two tables will be related. When you save it, save the table as **50001**.



Set up the table relation to the **Complaint Header** table on the **Document No.** field. The reason we want to do this is to enforce the proper table relationship.

Going by what we've learned, we can also modify our table to satisfy the following additional requirements:

- The item number in question (create the table relation to the **Item** table)
- The quantity of the item that was accepted and rejected



After adding these fields, mark the requirements off of the list.

|    | A    | B   |
|----|------|---|
| 1  | Done | Requirement   |
| 2  | x    | Every incident needs a unique number so it can be tracked                           |
| 3  | x    | The customer needs to be associated with every incident                             |
| 4  | x    | The vendor that supplied the good, if applicable, needs to be tracked               |
| 5  | x    | The date of the interaction must be tracked   |
| 6  | x    | The item number in question   |
| 7  | x    | The quantity of the item that was accepted and rejected                             |
| 8  |      | The source of the production order or purchase order                                |
| 9  |      | Detailed comments that is associated with every QMS entry                           |
| 10 |      | There can be 1 or more items per incident   |
| 11 | x    | Any QMS document that has been resolved will need to be marked as so                |
| 12 |      | The open QMS documents and the closed ones should be displayed as a separate screen |
| 13 |      | A report needs to be created on the feedback per item                               |

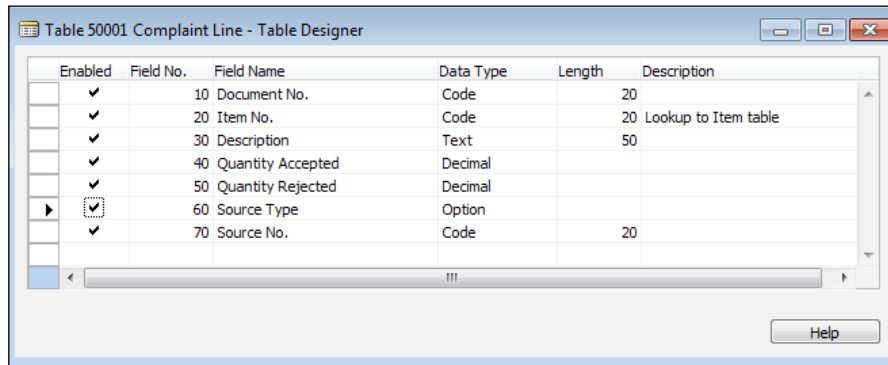


## Creating a conditional table relationship

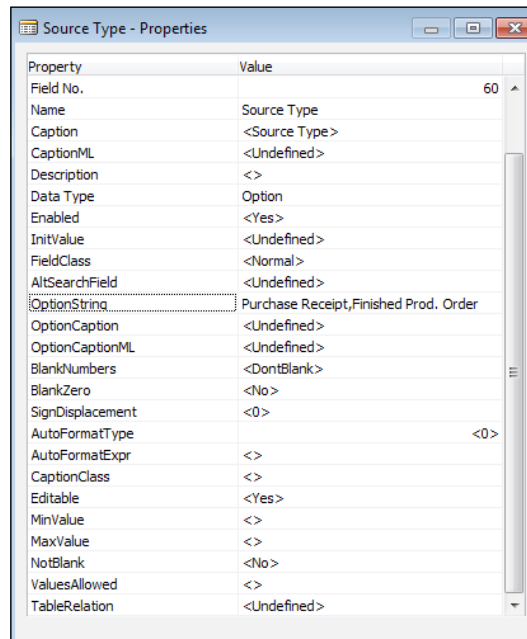
Within the requirement list, we have stated that we want to allow the user to specify whether the item came from a purchase order from a vendor or a production order. We want to allow the user to be able to specify and look up either Purchase Receipt or Finished Production Order.

The same concept is applied to the **Type** and the **No.** field in the Sales Line table. Depending on what you select as **Type**, the **No.** field will look up to the appropriate table. For example, if you choose **G/L Account** as **Type**, the **No.** field will look up to the G/L Account table.

1. To do this, we will need to create two fields, **Source Type** and **Source No.**. The **Source Type** field will allow the user to indicate whether **Source No.** will be a **Purchase Receipt** number or the **Finished Prod. Order** number.

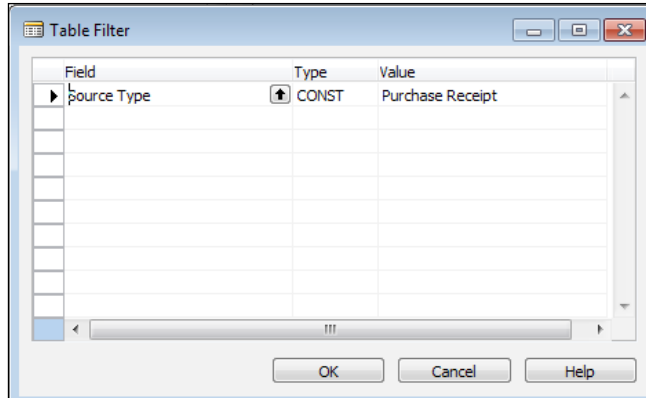


- Go to the property of the **Source Type** field and find the **OptionString** property. In the **Value** column of this property, type in `Purchase Receipt,Finished Prod. Order` with no spaces between the comma. This will give us the option values for our field.



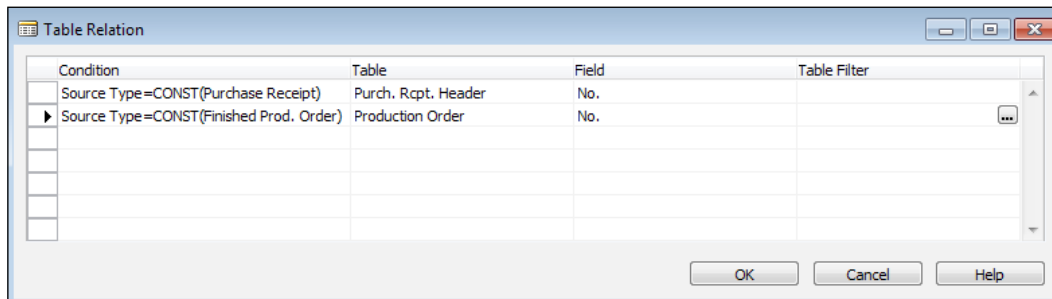
- At this point, close the table designer for table **50001** and save and compile. This step is necessary for us to be able to use the new fields we've created.
- Once that's done, go to the **TableRelation** property of **Source No.** to define our relations. Again, this is a conditional table relation, which means that the table relation depends on what we choose in the **Source Type** field.

- To specify the condition, click on the **AssistEdit** button in the **Condition** column. The field we want to use to determine the table relation is the **Source Type** field. So, if **Source Type** is **Purchase Receipt**, the table relation should be to the **Purchase Receipt Header** table.



- Click on **OK** and continue to define the table relationship. Perform the same steps for **Finished Prod. Order**. Using the same steps of finding what tables are used for what pages, find out the name of the table that holds the Finished Prod. Orders.

When you're done, your table relation for the **Source No.** field should look like the following screenshot:

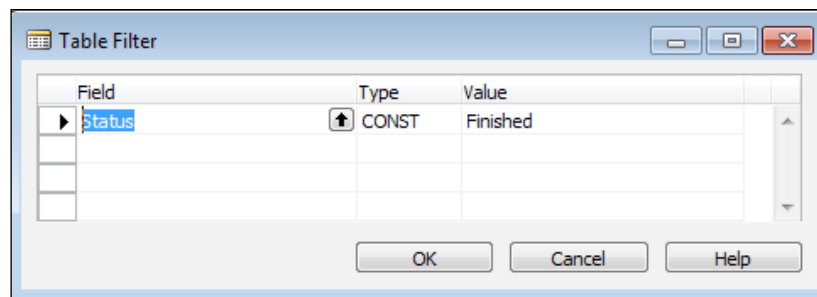


7. There's one additional detail that we need to pay attention to. We should look at the **Finished Prod. Order**, because by definition you can only ship production orders if they're finished.

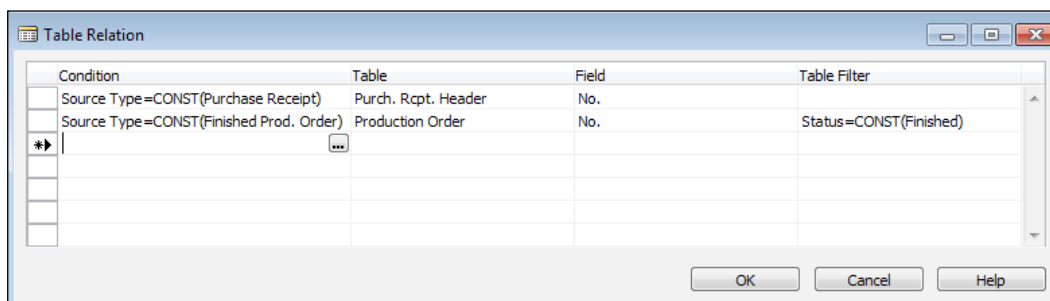


In Dynamics NAV, you can ship partially-produced items in the production order. But for the sake of our exercise, we're going to assume that as a business rule for CRONUS, only production orders that are finished are allowed to be shipped.

8. To specify what the user can see in the **Source No.** field if the user chooses **Finished Prod. Order**, we need to define **Table Filter**. Click on the **AssistEdit** button on the **Table Filter** column and filter on the **Status** field where the value is **Finished**.



9. Click on **OK**. After you're done, your final **Table Relation** for the **Source No.** field should look like the following screenshot:



## Adding a composite primary key

Based on what we have learned, we know that Dynamics NAV will automatically assign the first field on the table as the primary key when the table is created. In our case, **Document No.** will automatically be set as our primary key. However, just using **Document No.** as the primary key will not be enough to satisfy our requirement of allowing us to enter multiple lines. Don't believe me? Try entering two lines with the same document number.

So we need another field as part of the primary key. What do you think it should be?

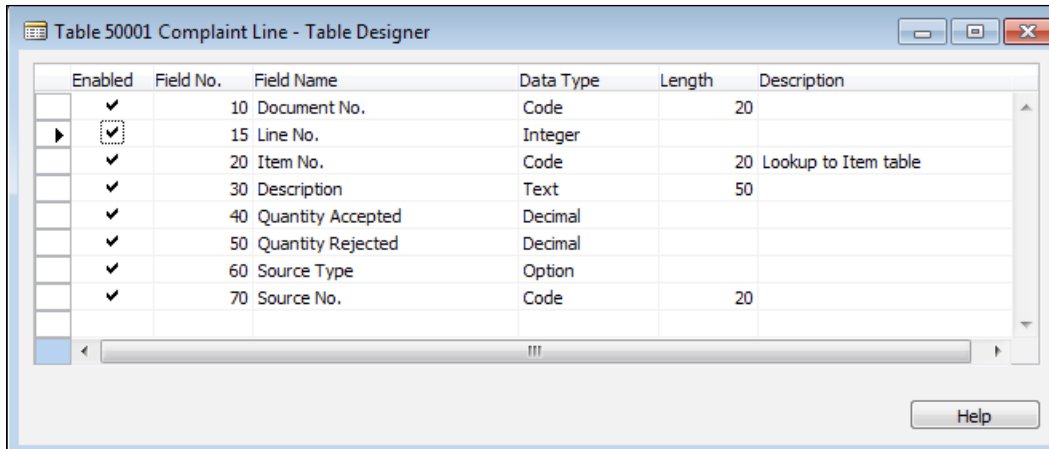
That's right! The answer is to look at what existing functions Dynamics NAV already has. If you guessed anything else, go back to the previous chapter. Again, the function in Dynamics NAV that most resembles what we're trying to do is `Sales Order`; more specifically, the `Sales Line` table is.

1. Go back and look at what the primary key for the `Sales Line` table is.

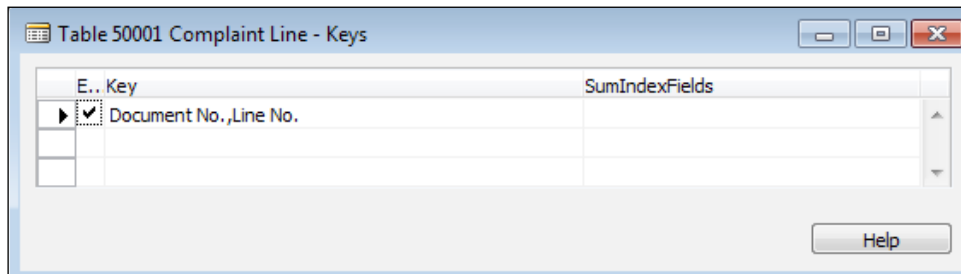
| E.. Key  | SumIndexFields                         |
|--|--|
| Document Type, Document No., Line No.                      | Amount, Amount Including VAT, Outst... |
| Document No., Line No., Document Type                      |  |
| Document Type, Type, No., Variant Code, Drop Shipmen...    | Outstanding Qty. (Base)                |
| Document Type, Bill-to Customer No., Currency Code         | Outstanding Amount, Shipped Not In...  |
| Document Type, Type, No., Variant Code, Drop Shipmen...    | Outstanding Qty. (Base)                |
| Document Type, Bill-to Customer No., Shortcut Dimensi...   | Outstanding Amount, Shipped Not In...  |
| Document Type, Blanket Order No., Blanket Order Line ...   |  |
| Document Type, Document No., Location Code                 |  |
| Document Type, Shipment No., Shipment Line No.             |  |
| Type, No., Variant Code, Drop Shipment, Location Code, ... |  |
| Document Type, Sell-to Customer No., Shipment No.          | Outstanding Amount (LCY)               |
| Job Contract Entry No.                                     |  |
| Document Type, Document No., Qty. Shipped Not Invoi...     |  |
| Document Type, Document No., Type, No.                     |  |

We've already stated that **Document Type** is not needed for what we're trying to do. So that leaves using **Document No.** and **Line No.** as our primary keys. Having the **Line No.** field automatically assigned by the system will allow us to ensure that whatever the users are entering into the lines area is unique.

- So let's add the **Line No.** field to our table. It's good practice to have all of our primary key fields near the top. As a good reader, you already know that it's good practice to have the field ID 10 numbers apart so that we can insert fields in between if we need to.



- After we add the **Line No.** field, save and compile the table so the additional field can be used.
- From the **Table Designer** screen on our table **50001**, navigate to **View | Keys** to specify our primary key. Use the **AssistEdit** button to be able to identify the additional field without overwriting what we already have.



- After we're done, close the **Table Designer** screen and then save and compile our table. The automatic assigning of the **Line No.** field will be taken care of in the next chapter when we create the pages for the user to interact with.

## Adding the Complaint Comments table

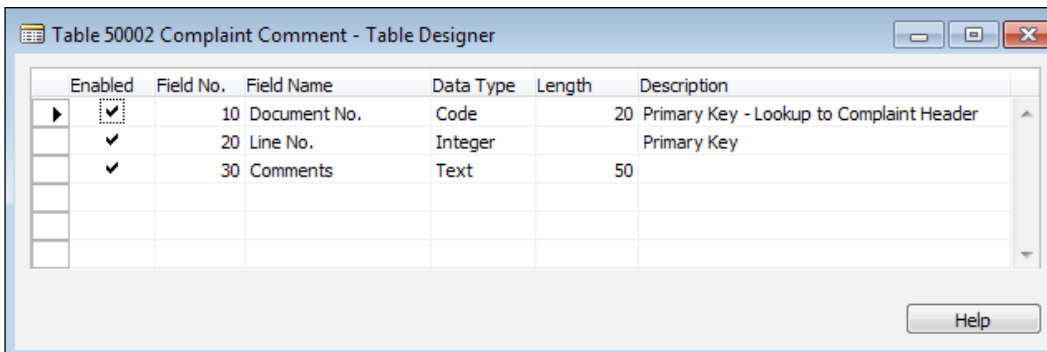
The last requirement that we want to work on is to allow the user to add comments. We can allow the users to enter comments directly to the lines area. For simplicity, we can leave it at that. However, doing that will clump the item data with comments. The users may have to differentiate what's a comment and what's an item if it's a long complaint log.

It's fine if you don't care about messy data. However, as we all know, creating reports based on messy data is not good. We're more professional than that; the comments should be in a specific section for the users to type in. The user can still enter brief comments on the line if they wish, but there's an area for the users to enter whatever drama they've encountered to their heart's content.

Using the existing `Sales Order` structure in Dynamics NAV as our basis, the table we want to model for the comments is the `Sales Comment Line` table. The table ID is **44**.

Similar to the `Complaint Line` table, the user needs to be able to add unlimited lines of comments. Create the `Complaint Comment` table and save it as **50002**.

Remember to add the table relationship and set the proper composite primary keys as well.



| Enabled                             | Field No. | Field Name   | Data Type | Length | Description                              |
|-------------------------------------|-----------|--------------|-----------|--------|--|
| <input checked="" type="checkbox"/> | 10        | Document No. | Code      | 20     | Primary Key - Lookup to Complaint Header |
| <input checked="" type="checkbox"/> | 20        | Line No.     | Integer   |        | Primary Key                              |
| <input checked="" type="checkbox"/> | 30        | Comments     | Text      | 50     |  |
| <input type="checkbox"/>            |           |              |           |        |  |
| <input type="checkbox"/>            |           |              |           |        |  |
| <input type="checkbox"/>            |           |              |           |        |  |

After creating the `Complaint Comment` table, let's get the requirements checked-off of our list.

|    | A    | B   |
|----|------|---|
| 1  | Done | Requirement   |
| 2  | x    | Every incident needs a unique number so it can be tracked                           |
| 3  | x    | The customer needs to be associated with every incident                             |
| 4  | x    | The vendor that supplied the good, if applicable, needs to be tracked               |
| 5  | x    | The date of the interaction must be tracked   |
| 6  | x    | The item number in question   |
| 7  | x    | The quantity of the item that was accepted and rejected                             |
| 8  | x    | The source of the production order or purchase order                                |
| 9  | x    | Detailed comments that is associated with every QMS entry                           |
| 10 | x    | There can be 1 or more items per incident   |
| 11 | x    | Any QMS document that has been resolved will need to be marked as so                |
| 12 |      | The open QMS documents and the closed ones should be displayed as a separate screen |
| 13 |      | A report needs to be created on the feedback per item                               |

## Summary

The first step in creating any application is to start with the tables. Once the initial tables are defined, we can start doing our pages and reports. In addition, defining what data you want to keep track of is the most important step in any application development.

In this chapter, we've created the `Complaint Header`, `Complaint Line`, and `Complaint Comment` tables.

Note that the table you have created is not set in stone. We can always go back to make changes to the table structure if the requirements change. You'll notice that we have not put in any coding in the tables. Right now, you've just created a holding place for the data with no internal business logic, other than setting table relations. In most applications that you'll encounter, that's the majority of the work.

As a general rule for developers starting out in Dynamics NAV, always look for an existing application that does a similar thing to what you're trying to do. There's really not many applications you want to create that don't already have something similar in the standard out of the box Dynamics NAV. For anything that is not similar, you probably do not want to attempt it on your own. Rather, seek professional help from an experienced Dynamics NAV partner.

In the next chapter, we will create the pages that the user will use to interact with the tables. We will also create a report to display our data.





# 7

## Creating the Application – Pages and Reports

*"The aim of art is to represent not the outward appearance of things, but their inward significance."*

*– Aristotle*

In the previous chapter, we created tables and set the table relationships for the appropriate fields. Having tables allows us to have a place to store the data, but it's not very user friendly in terms of data entry.

Imagine if you had to insert a record into the header table; you would have to exit the header table and go to the lines to insert the detail. It's possible, but your job would not be much fun. In addition, running the tables will display all the records within the table, so you would need to filter very carefully to see the details of a complaint. Possible, but not fun.

This is where pages come in. When we created our tables, we built some business logic through the table relationships and primary keys. With pages, we present the data from the table in an effective manner so the user can easily get the information they want quickly and accurately. We're allowed to group similar tables, reports, and functions into pages to enhance what the user can do with the data. It also allows a way for the user to modify and delete data efficiently.

The same concept applies to reports. The difference between a report and a page is that a report is usually output only. It's great for analysis where creating a page for your analysis may not make too much sense.

In this chapter, we will explore creating pages to the complaint tables we created in the previous chapter. We will create a report to analyze and use the `Sales Order` page as the basis to model our `Complaint entry` screen.

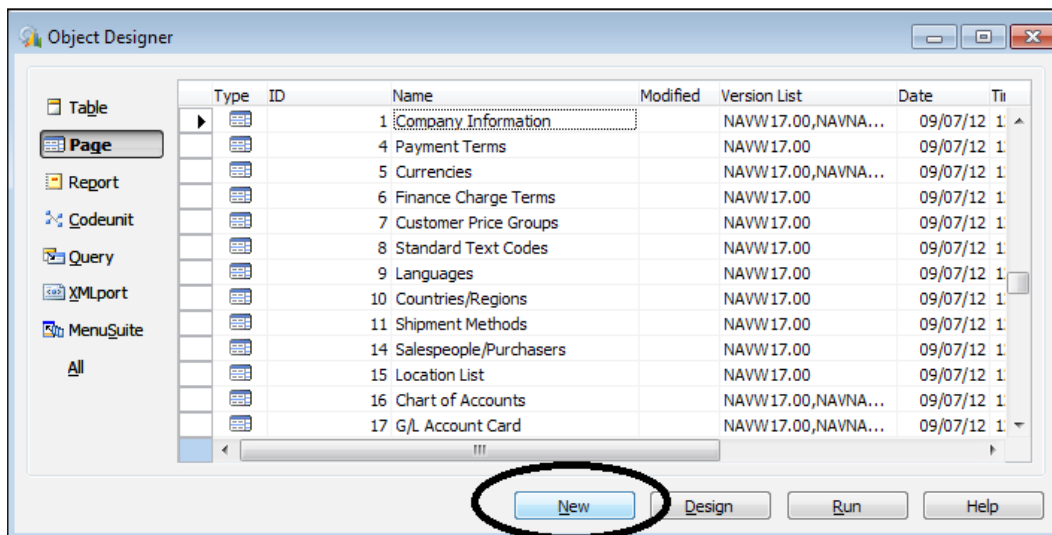
As mentioned in *Chapter 2, Getting Familiar with Dynamics NAV 2013*, there are many different pages you can create in Dynamics NAV. For a full list of the different pages you can create and what they're used for in Dynamics NAV, go here: [http://msdn.microsoft.com/en-us/library/dd301400\(v=nav.70\).aspx](http://msdn.microsoft.com/en-us/library/dd301400(v=nav.70).aspx).

To address what our users need in our project, we will focus on creating the `Document`, `ListPart`, and `List` pages.

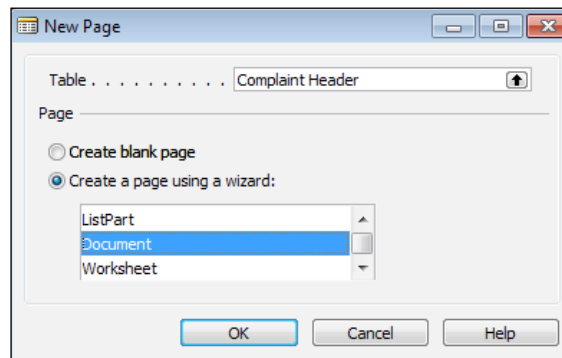
## Creating the Product Complaint page

We will be continuing the momentum we built up while resolving the user requirements; creating pages was not one of them. And why would it be? The user should not care about the relationship between tables and pages. All they should care about is being able to properly do their job with the function that you deliver to them.

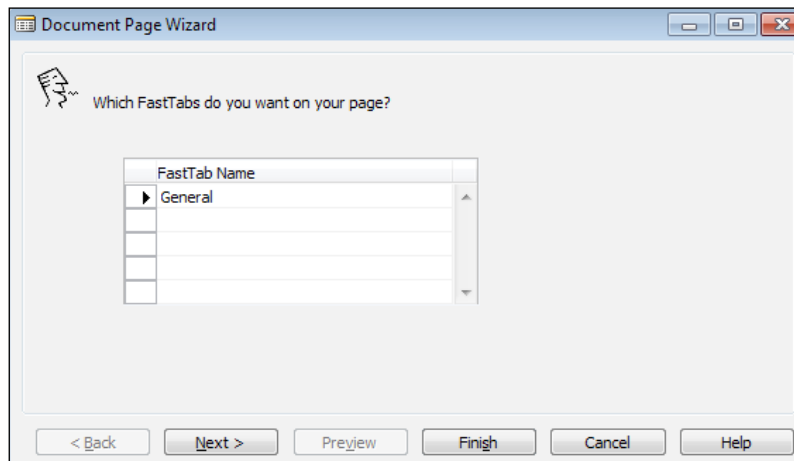
To create a page for the user to interface the data, go to **Object Designer** and click on **Page**. Click on **New**:



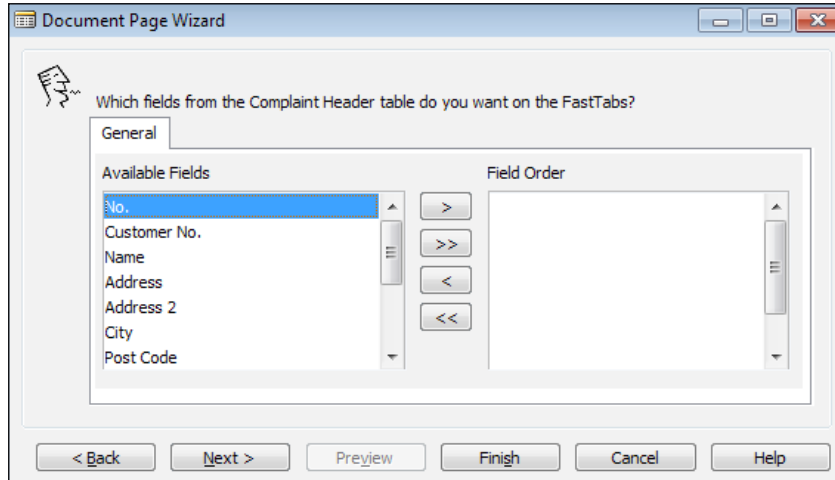
This will bring up the page creation wizard. In the **Table** section, specify **Complaint Header** as the table we will use for the wizard. Since we're creating a page where the user can enter the header and the line information just like `Sales Order` page, we will create the **Document** page. You can see how the `Sales Order` pages are built by going to the page properties as explained in *Chapter 5, Finding Similar Functions for Inspiration*.



When we click on **OK**, the wizard will ask us if we want to add **FastTabs** to our page. Remember, **FastTabs** are ways for us to group fields together:



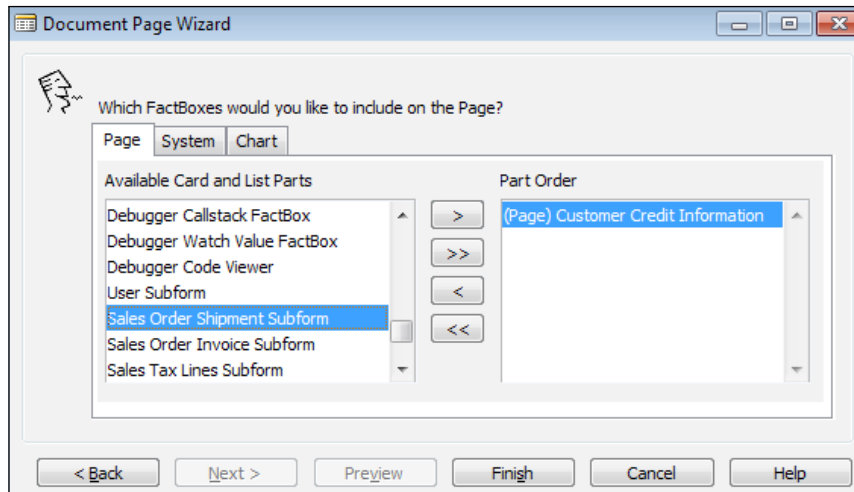
Since we don't have much information to group together, just click on **Next**:



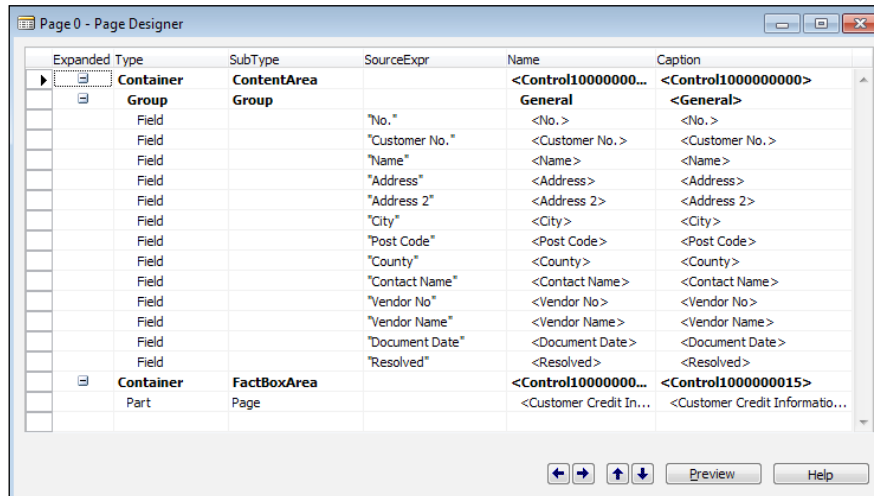
The wizard will now ask us to specify which fields we want to add to the page. Go ahead and click on the right arrow (>>) to move over all of the fields. Click on **Next** when you have moved all of the fields over to the **Field Order** column.

The next part of the **Document Page Wizard** will allow us to add some premade **FactBoxes**. Note that you can make your own **FactBox** pages to be selected and used.

When the users are entering a complaint by the customer, it helps the user to get a quick glance at the customer. Scroll down and find **Customer Credit Information** and move it to the **Part Order** column:



Click on the **Finish** button to complete our page wizard:

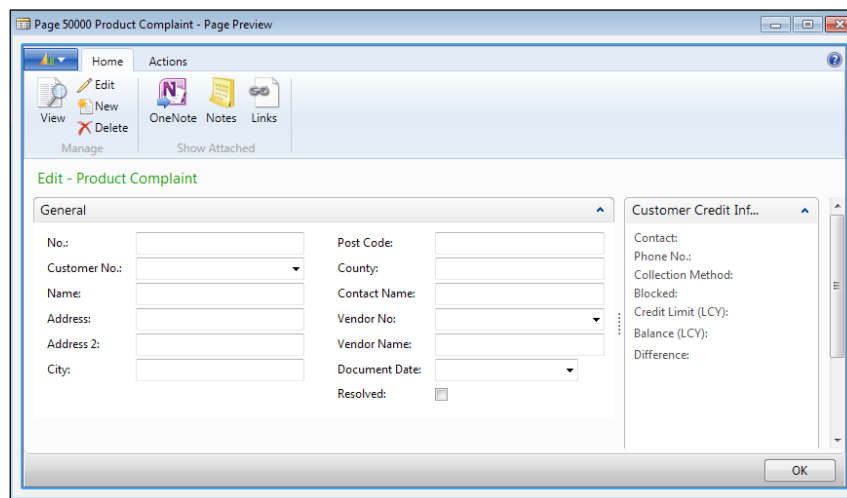


Click on **File | Save**, assign the **ID** field as **50000**, and name it **Product Complaint**.



If you're using an MSDN license to follow along with this book, make sure you save the objects that are allowed in the MSDN license! The contents of the MSDN license are described in *Chapter 1, Getting Dynamics NAV 2013 Installed on Your Computer – For (Almost) Free*.

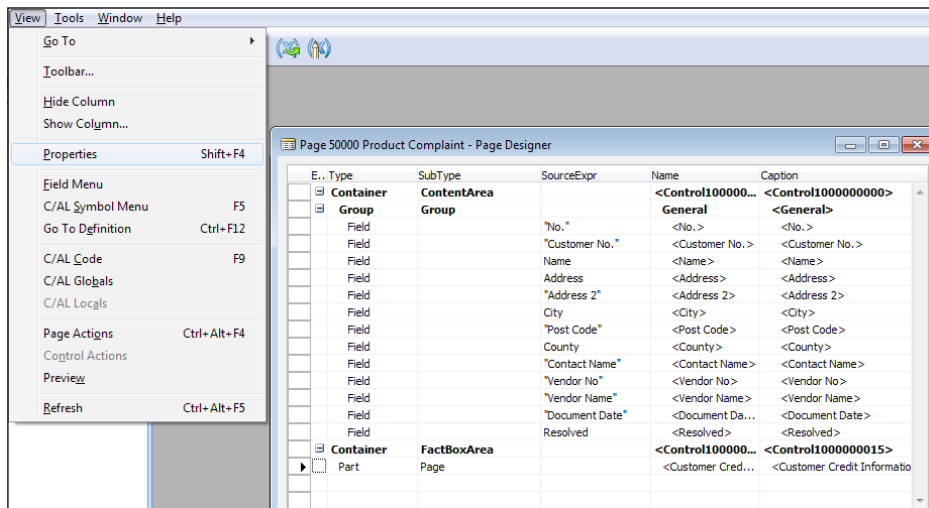
Once the page is saved, you can click on **Preview** to look at what this page looks like:



## Linking FactBoxes

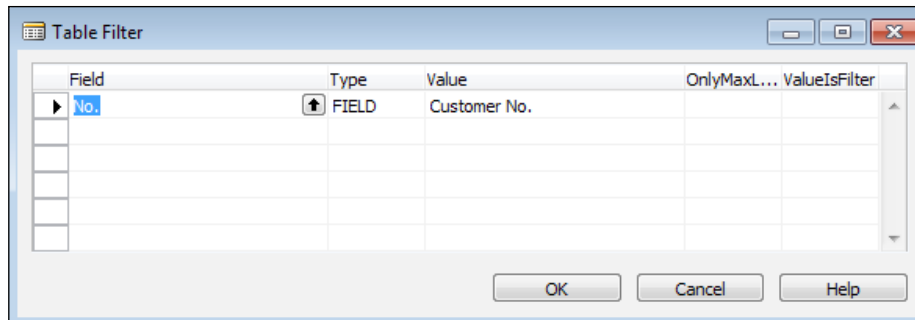
We've added a FactBox to the **Product Complaint** page, but how does Dynamics NAV know which customer to display in the FactBox? Well, we have to tell it to display the right customer information!

Go back to **Page Designer** for the **Product Complaint** page and put your cursor on the **Part** type where we defined our FactBox and click on **View | Properties**:

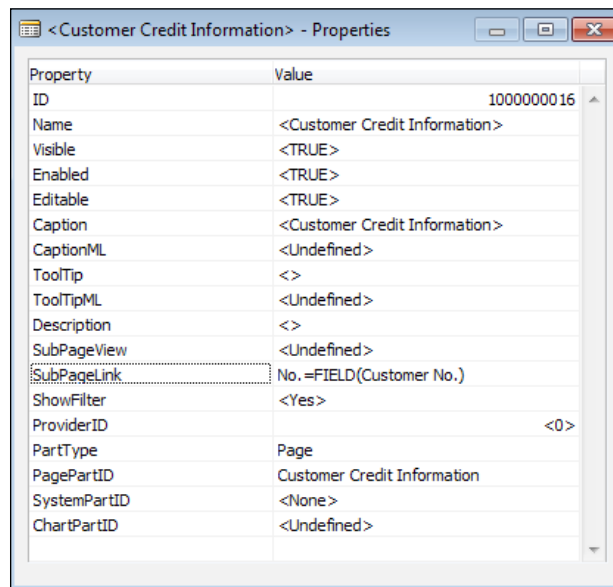


Similar to the previous chapter where we specified the table relationships, we can define the relationships between the tables at the page level. The property we want to use is the **SubPageLink** property.

Click on the **AssistEdit** button and define the form link. This is very similar to defining the field relationship on table fields. The **Customer Credit Information** FactBox uses the **Customer** table as its source table. So, we want to relate the **Customer** table **No.** field to the **Customer No.** field on our **Product Complaint** page:



Click on **OK** to confirm the table relationship settings. The property for the **Part** type for FactBox should be the same as shown in the following screenshot:



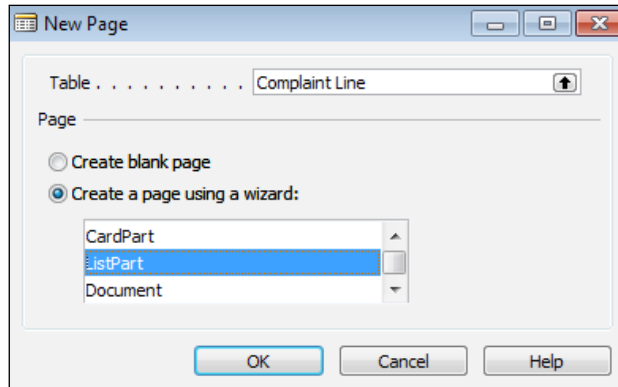
Click on **File | Save** to save our project. Close **Page Designer** for the **Product Complaint** page.

We're done with the **Product Complaint** page, for now.



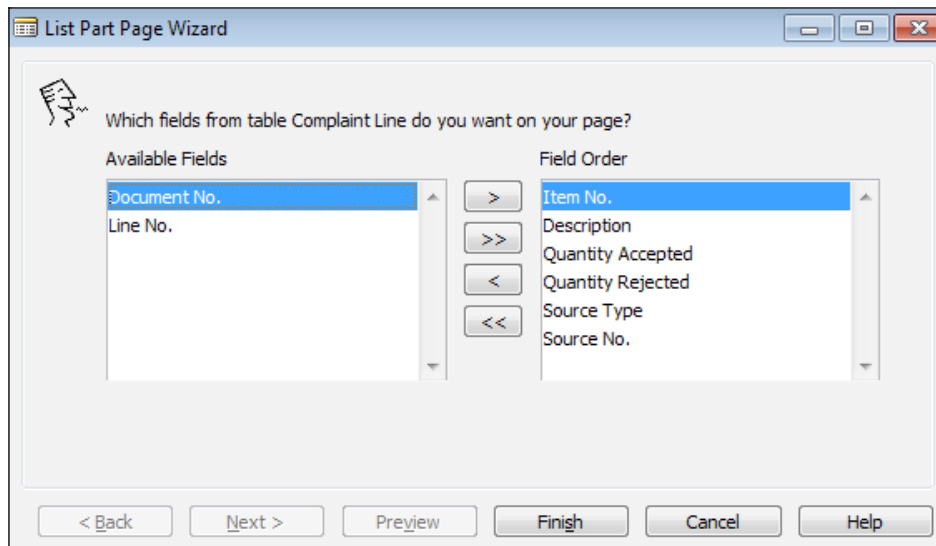
## Creating the Product Complaint subpage

After the **Document** page is created, we can now create the **ListPart** page for the lines in our table. Go back to **Object Designer** and click on **New on Page** to bring up the wizard. This time, chose the **Complaint Line** table and choose the **ListPart** option for our wizard:



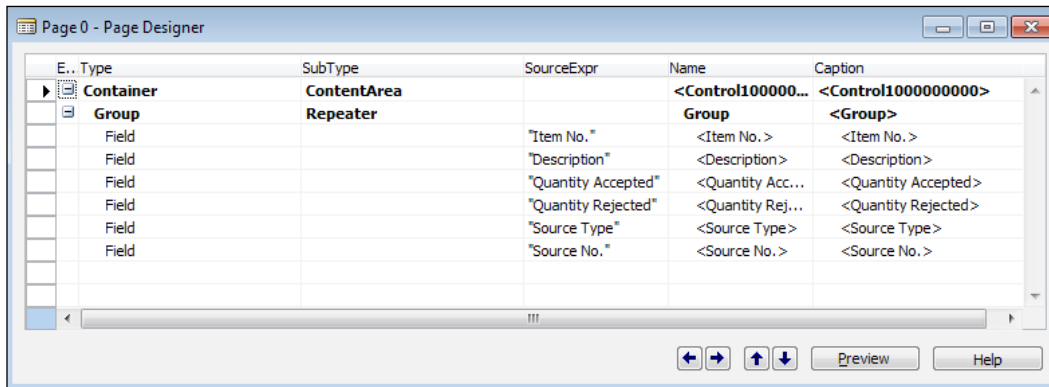
Click on **OK** to bring up the next portion of the wizard.

The next screen will show us what fields we want on the subpage. Bring over the fields as shown in the following screenshot:




The reason why we don't want to bring over the **Document No.** field and the **Line No.** field is because those are automatically assigned by the system. In our set up for the **Complaint Line** table, the **Document No.** field is related to the **No.** field in the **Complaint Header** table. If linked, the **Document No.** field will automatically be populated when we insert a new line. And as we explained earlier, the **Line No.** field is automatically assigned by NAV as well.

Click on **Finish** and the wizard will create the layout for you:



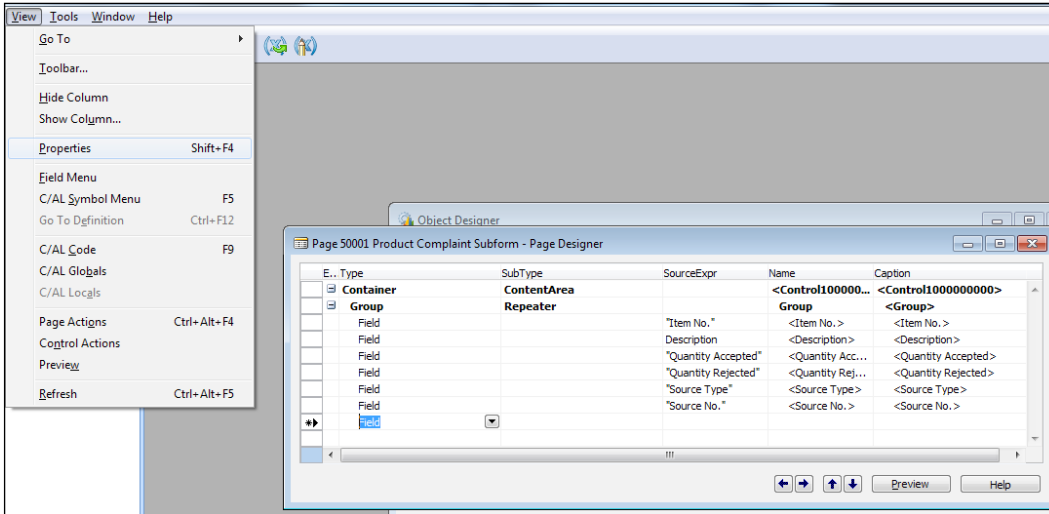
Click on **File | Save**, assign the **ID** as **50001**, and name it **Product Complaint Subform**.


 Even though we're naming this object **Product Complaint Subform**, it really should be called **Product Complaint Subpage**. Forms are used in the older version of Dynamics NAV and have since been replaced by pages. In the interest of following the same naming standards from Dynamics NAV, we will use the term subform.

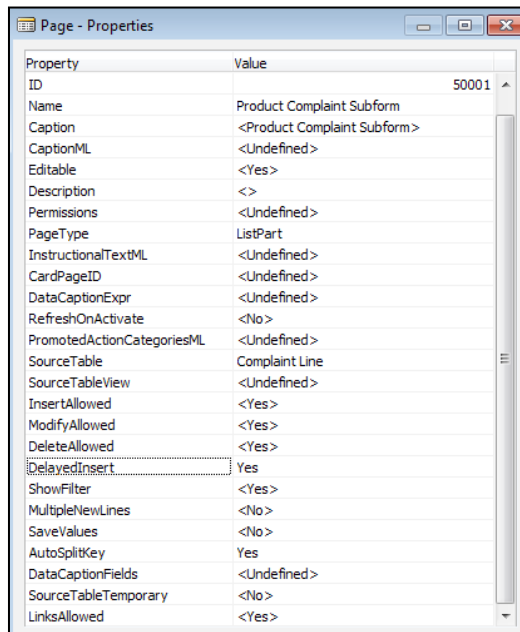
## The AutoSplitKey property

Remember how our primary key on the lines ends with **Line No.**? Remember how we keep stating that the **Line No.** field is automatically generated by the system? Here's how we do this.

The way to do this is not at the table level, but at the page level. From **Page Designer**, move the cursor down until you are at a new line, then click on **View | Properties**:



On the property screen, find the **AutoSplitKey** property and set it to **Yes**. Another property you will need to set is the **DelayedInsert** property. Set this property to **Yes** as well:



The **AutoSplitKey** property can only be set if the last key on your primary key is an integer field. The purpose of **AutoSplitKey** is to have Dynamics NAV automatically assign a unique integer value to keep the record unique.

The **AutoSplitKey** and the **DelayedInsert** properties almost always go together. If you leave the **DelayedInsert** property as **No** (as it is by default), when you go to a new line, it will automatically insert a blank line; the reason is because the primary key would've been initialized. Setting this property to **Yes** will basically delay inserting a record until additional information is filled in.

Once you set this to **Yes**, you're done! Dynamics NAV will do its job.

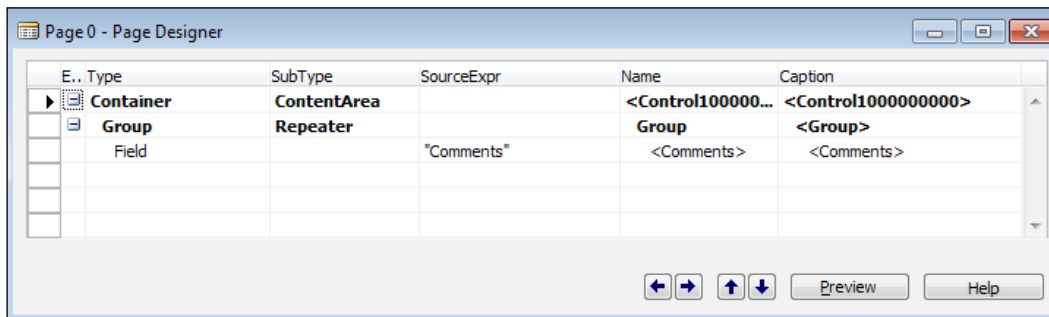
Close the **Page Designer** screen for the **Product Complaint Subform** page. Don't forget to save!

## Creating the Product Complaint Comments page

We will now create the **Product Complaint Comments** page. The page we will create will be a **List** page. Try creating this page using the wizard on your own. Here are a few pointers:

- The table you want to use is **50002, Complaint Comment**
- Make sure you set the **AutoSplitKey** and the **DelayedInsert** properties
- We don't need any FactBoxes for the comments page

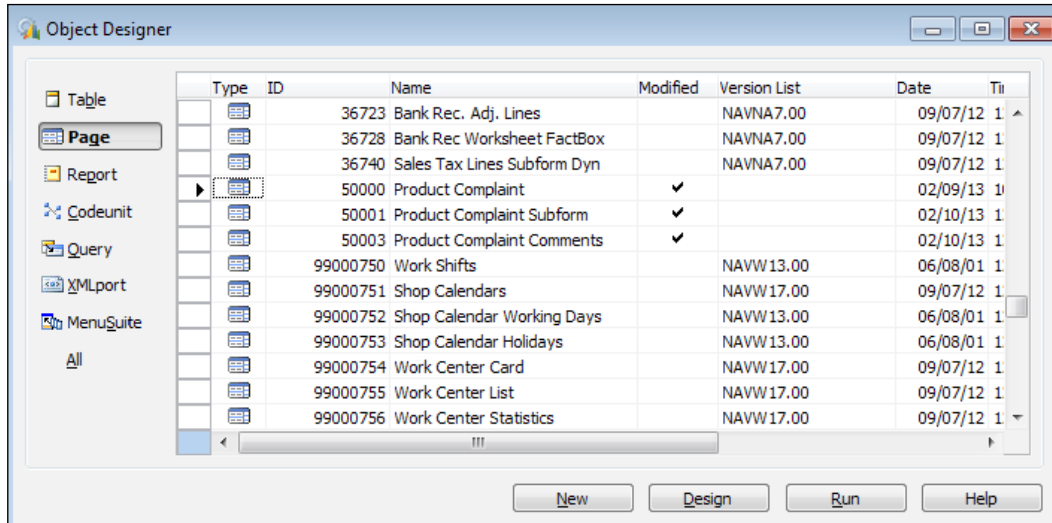
When you're done, **Page Designer** should look as shown in the following screenshot:



Save this page with ID as **50003** and name it **Product Complaint Comments**.

## Linking the pages together

We've now created three pages that are saved in **Object Designer**:

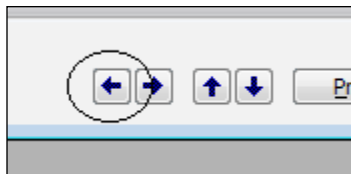


Individually, they are not much to look at. The next step we need to take is to link up the different pages so that they can perform as one cohesive unit for the users.

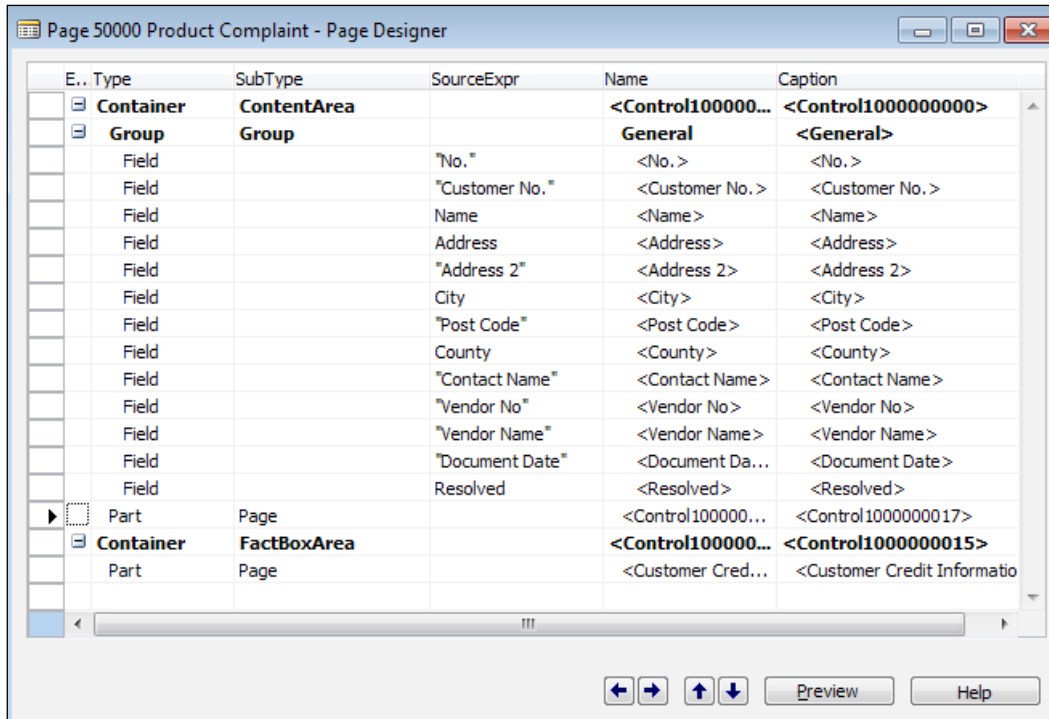
The user will only need to open up the **Product Complaint** page to start recording complaints, so this is the page where we will add the subpage and the comment page we've created.

Go to **Page Designer** for page 50000, **Product Complaint**. The first thing we will do is add page 50001, **Product Complaint Subform**. Using the **Sales Order** page as a sample, we need to add a **Part** type with the subtype of **Page**.

To insert a line to the appropriate area, we can click on **Edit | New** or press **F3**. Fill in the **Type** and the **SubType** fields as we described. On the indentation, we need it to be at the same level as the group. To move **Part** to the left, click on **Edit | Position | Move Left**. You can also click on the **Move Left** icon at the bottom-right side of **Page Designer**:



After aligning the indentation of the parts on the page, your screen should be as shown in the following screenshot:

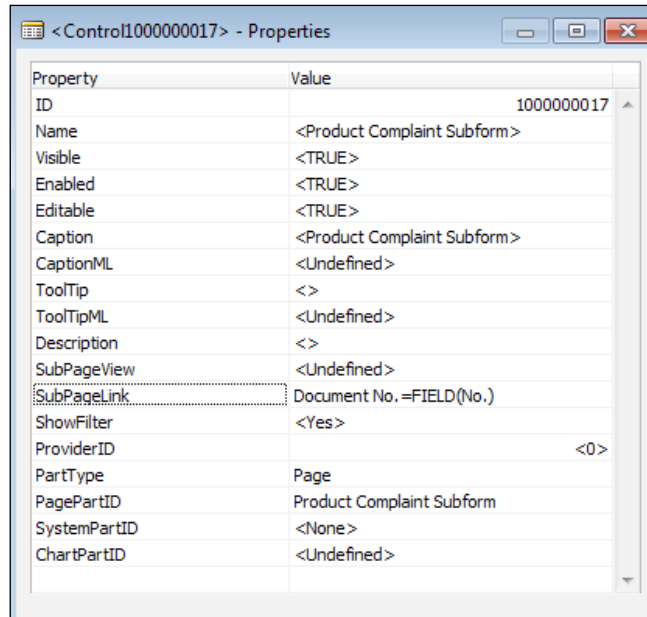


The next thing we need to do is to specify the subpage page ID and link the page. To do this, we go to the property for **Part**. Make sure your cursor is on the **Part** line and click on **View | Properties**.

Specify the **PagePartID** property as **Product Complaint Subform**.

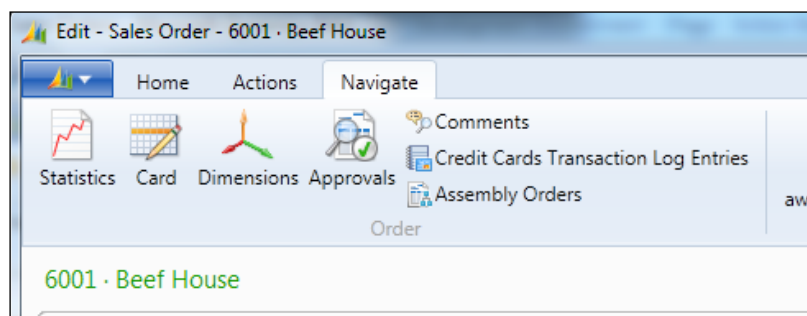
Set the table relation on the **SubPageLink** property so the appropriate link can be established between the **Header** and the **Line** table. In our case, the **Document No.** field on the **Line** table will link to the **No.** field on the **Header** table.

When you're done, the property for this **Part** page should look as shown in the following screenshot:



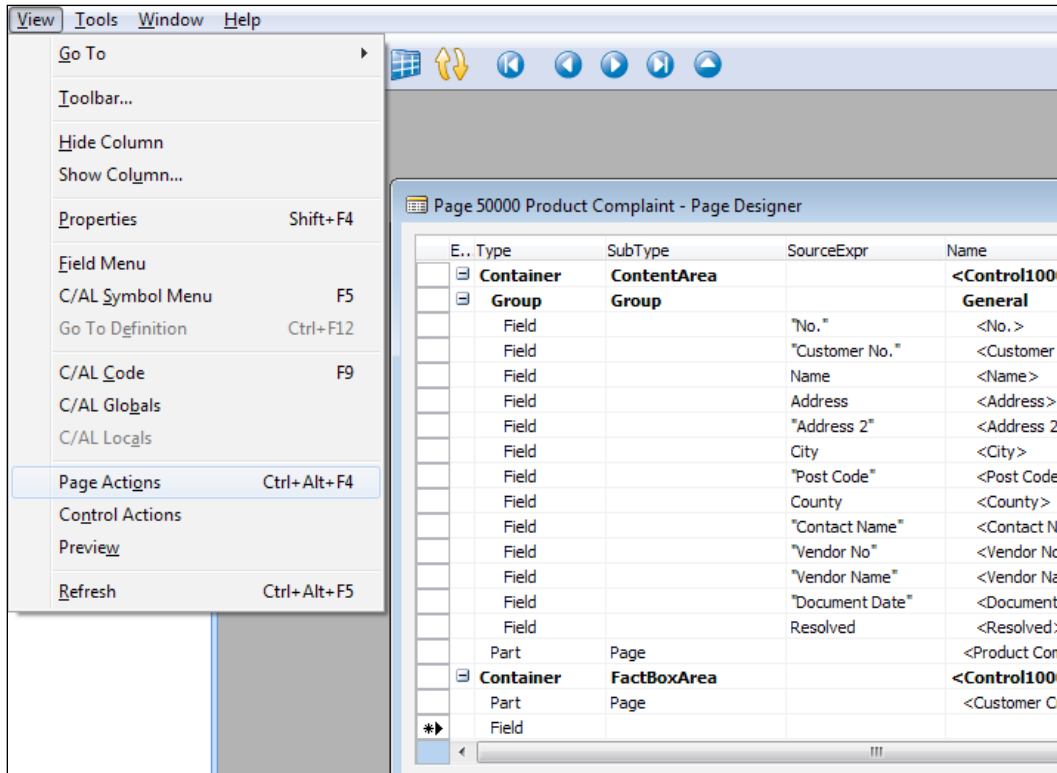
Close the **Properties** screen and save the page.

The **Product Complaint Comment** page will be linked a little differently. Based on the example, when we look at the *Sales Order* function in Dynamics NAV, the comments are not displayed the same way as the data on the lines. Rather, you access the comments by clicking on an icon to bring up a page for data entry:



Of course we can add the comments to look like **Product Complaint Subform**, but we want to enforce the same design principles that we discussed back in *Chapter 2, Getting Familiar with Dynamics NAV 2013*. We will add the comments page as part of the menu at the top.

On **Page Designer**, go to the bottom until you hit a new line and click on **View | Page Actions**:



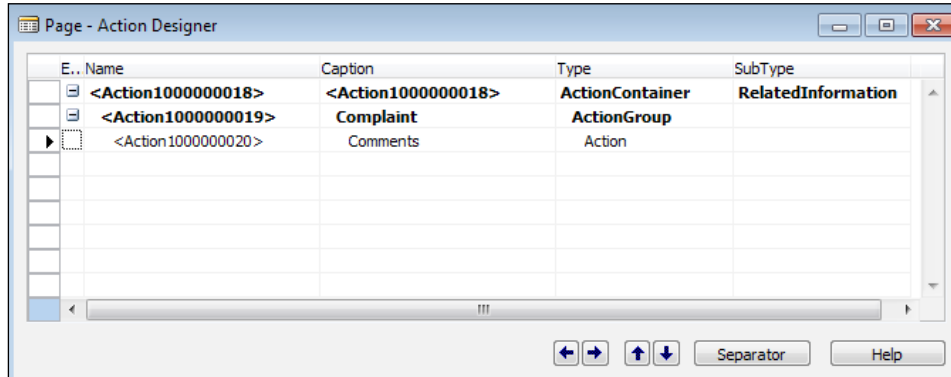
In **Action Designer**, we can link other functions to be accessible directly from the **Product Complaint** page. The only additional function we've created is **Product Complaint Comment**. As you get familiar with Dynamics NAV, you can add additional functions.

To group the functions together, do the following once you get to the **Action Designer** screen:

1. Go to the **Type** column and type in **ActionContainer**.
2. Select **RelatedInformation** as **SubType**.
3. Go to the second line (it should be automatically indented) and fill in the **Complaint** as the **Caption**.
4. Select **ActionGroup** for the **Type** column.
5. Go to the third line (this will be automatically indented) and put in **Comments** for the **Caption** column.



When you're done, it should look as shown in the following screenshot:

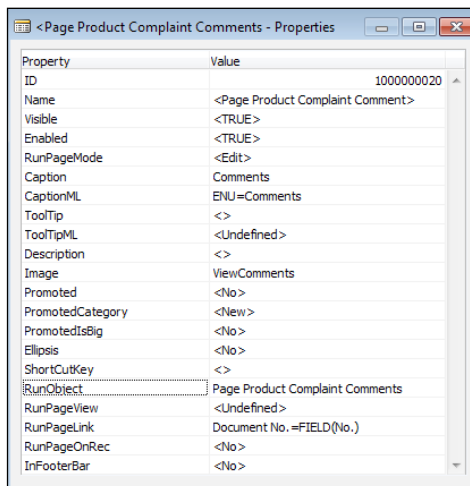


Don't worry about the long integer value after the word **Action**. Those are assigned to keep the controls within the object unique. Do not modify them!

Click on **Comments**, go to the properties, and then specify the following properties:

- **RunObject** – Select the page **Product Complaint Comment**.
- **RunPageLink** – Link **Document No.** field on the **Comment** table to the **No.** field on the **Header** table.
- **Image** – Set it as **ViewComments**. This is a built-in image that can be displayed on the page as our icon. This is for pure aesthetics.

When you're done, your properties page for the **Comments** action should look as shown in following screenshot:



Close **Page Designer** and save!

Now click on **Run** on page 50000 and see what we've created:

The screenshot shows a Dynamics NAV window titled "View - Product Complaint". The window has a ribbon with "Home", "Actions", and "Navigate" tabs. Below the ribbon is a "Comments" section and a "Complaint" section. The main content area is divided into three panes:

- General:** Contains various text boxes and dropdown menus for fields like "No.", "Customer No.", "Name", "Address", "Address 2", "City", "Post Code", "County", "Contact Name", "Vendor No.", "Vendor Name", "Document Date", and "Resolved".
- Product Complaint Subform:** A table with columns: "Item No.", "Description", "Quantity Accep...", "Quantity Reject...", "Source Type", and "Source No.". It includes "Find", "Filter", and "Clear Filter" buttons.
- Customer Credit Information:** A vertical list of fields including "Contact", "Phone No.", "Collection Method", "Blocked", "Credit Limit (LCY)", "Balance (LCY)", "Difference", "Aging as of 01/23/14 (showing days of)", "31-60 Days", "Over 60 D", "Payment Terms Code", "Payment Method Code", "Latest Payment Date", and "Latest Payment Amount".

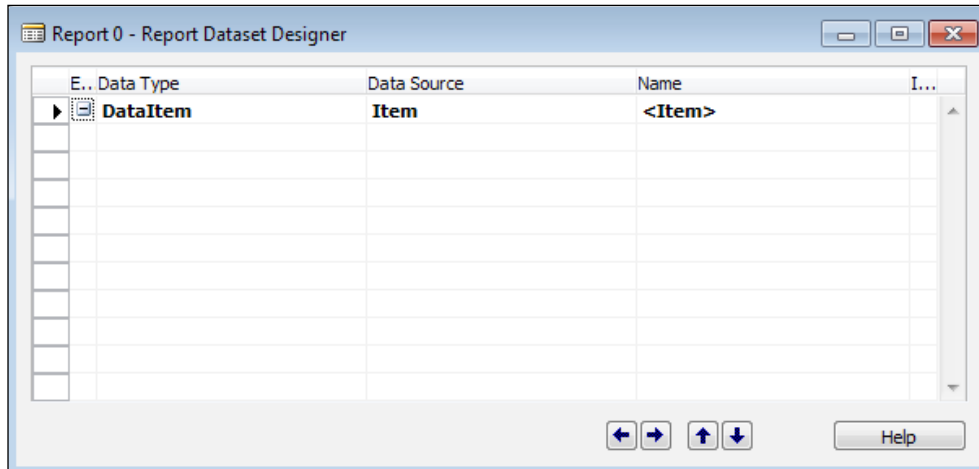
A "Close" button is located at the bottom right of the window.

It's looking a lot like a page that you have opened in Dynamics NAV! Good job!

## Create an analysis report using wizards

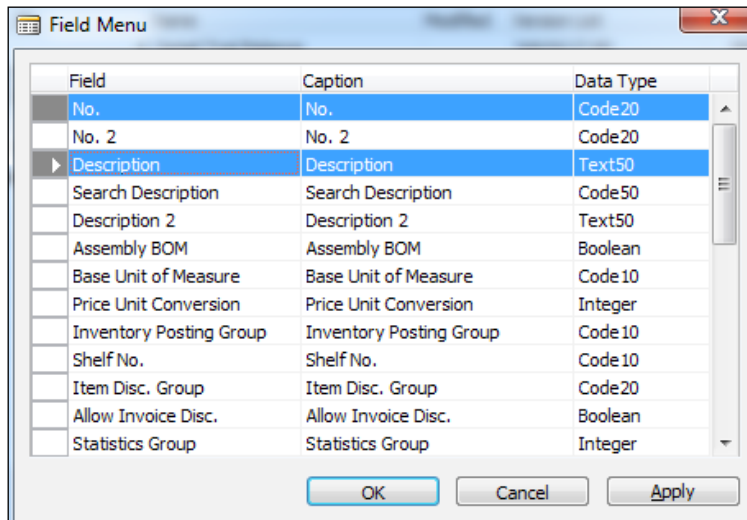
Going back to the requirement list, one of the things that the users want is to be able to create a simple report to display the items that are accepted and rejected. So, the report will display the item, then the complaint lines that are associated with the item.

From **Object Designer**, click on **Reports** and click on **New** to create a new report. For **Data Source**, we want to start off using our **Item** table:

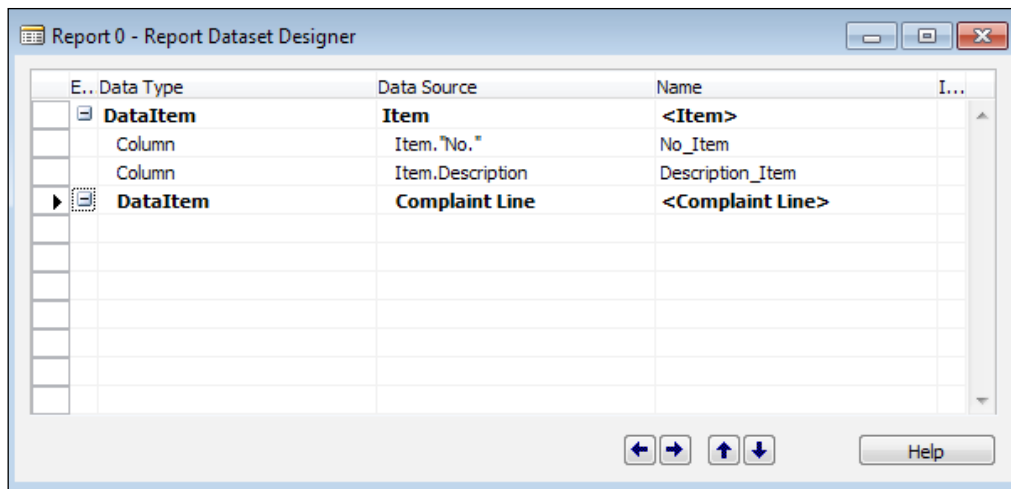


To automatically fill in the fields we want to display on the report, we click on the **View | Field Menu** option. When the **Field Menu** screen appears, simply highlight the fields you want and click on **OK** to have it inserted into **Report Designer**.

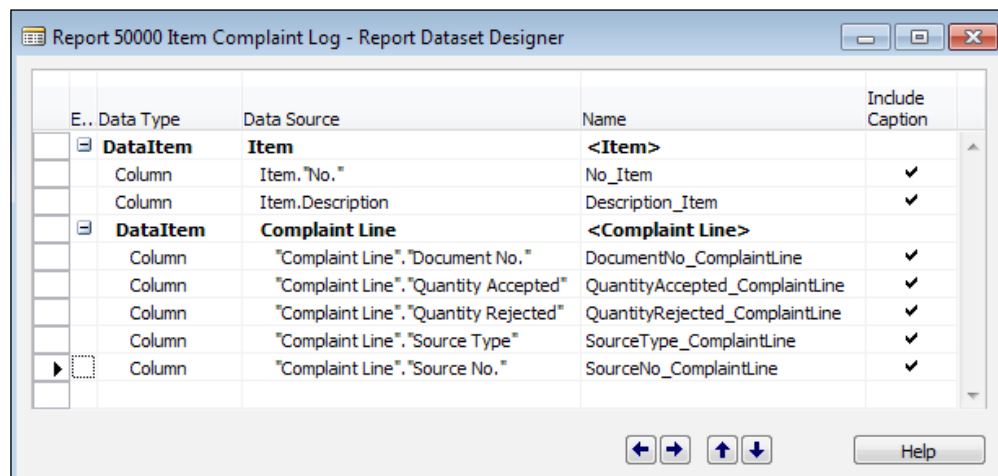
For our report, we will add the **No.** and **Description** fields from the **Item** table:



Now we will add in the second **Data Source** value, which in this case is the **Complaint Line** table:

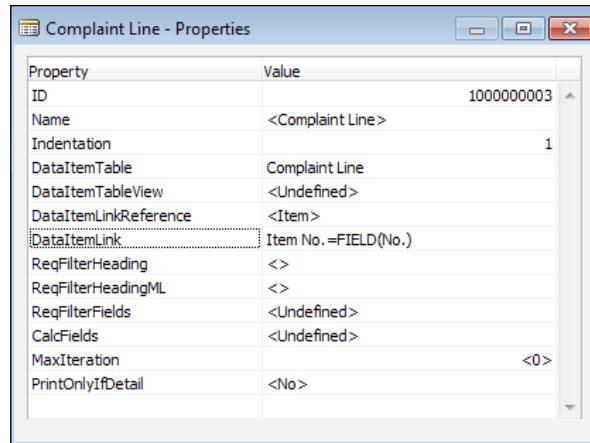


Bring up **Field Menu** and add the fields we want printed from **Complaint Line**. After adding the fields, mark the **Include Caption** checkbox on the last column for the fields you want to display the caption for. In our case, every field that's displayed should have a caption:



We need to link the `Complaint Line` table to the `Item` table, or else we will print every record on the `Complaint Line` table for every item record that's read. To do this, click on the **Complaint Line** data item and then click on **View | Properties**.

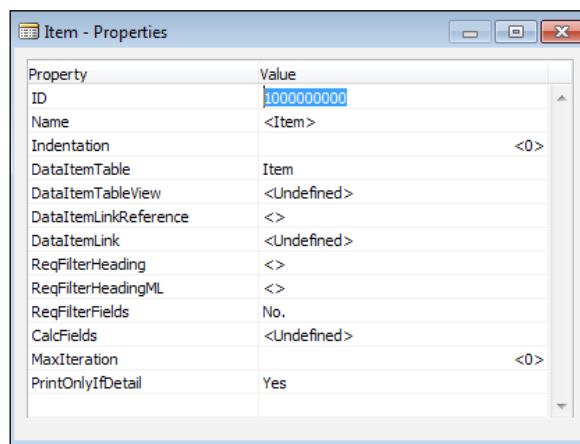
The property we need to set is the **DataItemLink** property. We want to link the **Item No.** field from the **Complaint Line** table to the **No.** field on the **Item** table. We will use the **AssistEdit** button to help us in creating the link. When you're done, the **DataItemLink** property should look as shown in the following screenshot:



In addition, we want to allow the user to filter the item numbers they want to print. To have the filter visible for the user to filter on, we need to go to the property of **Item Data Source**.

The property we want to set is **ReqFilterFields**. Use the **AssistEdit** button and choose **No.** as the field.

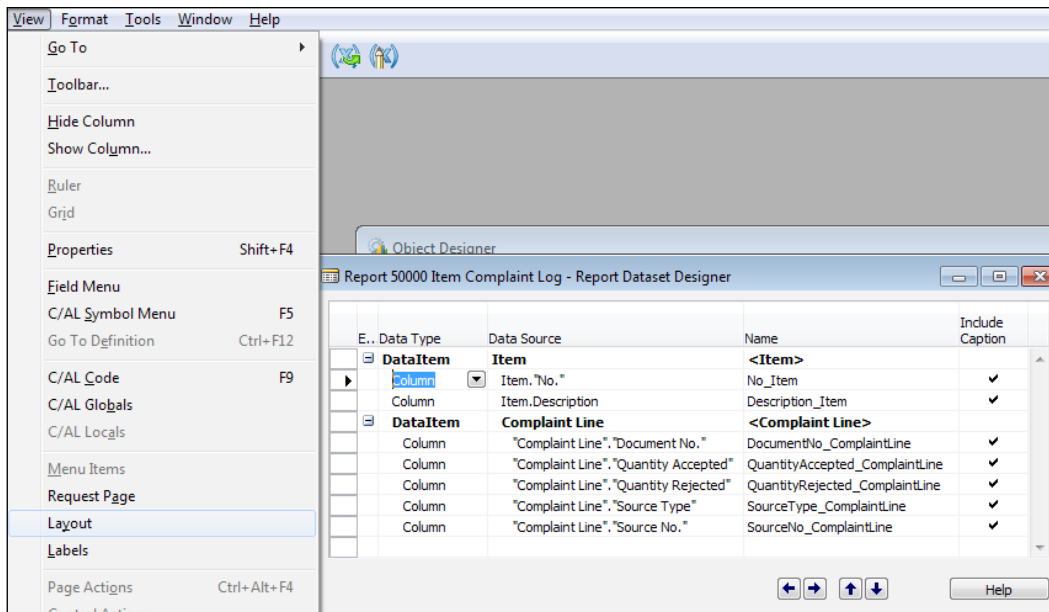
Also on the **Item Data Source** property, we do not want to display the item if there are no details against it. Basically, we don't want to print out items that do not have any complaints against it. To do this, we set the **PrintOnlyIfDetail** property to **Yes**:



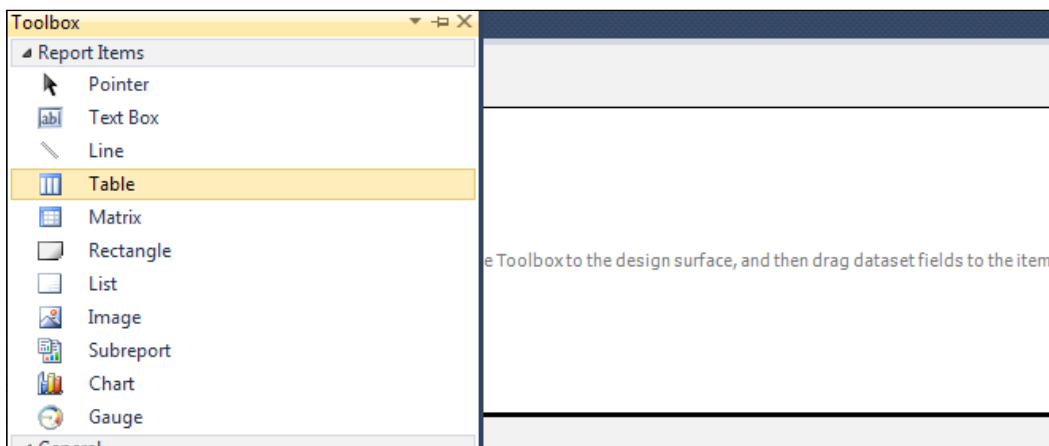
Once you're done, save this report as **50000** and call it **Item Complaint Log**.

Now that we've created **Data Source** and specified the fields that go into the report, the next step is to design how the report will actually print.

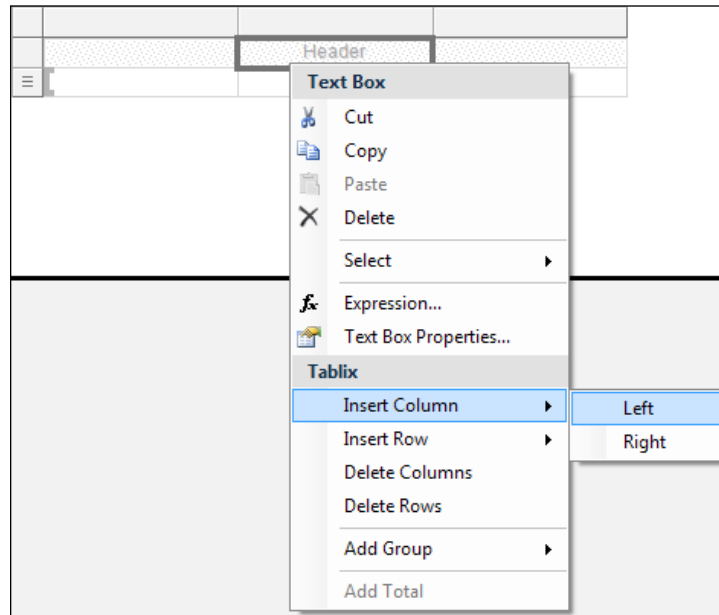
Click on **View | Layout** to access **Visual Studio 2010**:



When Visual Studio is started, you'll get a blank page. Still in Visual Studio 2010, click on **View | Toolbox** and drag the **Table** control to the report layout:



By default, when you drag the **Table** control to the report layout, it'll have three columns. We will need to add four additional columns. Right-click on the **Table** control and click on **Insert Column**. Choose either **Left** or **Right**, it doesn't matter:

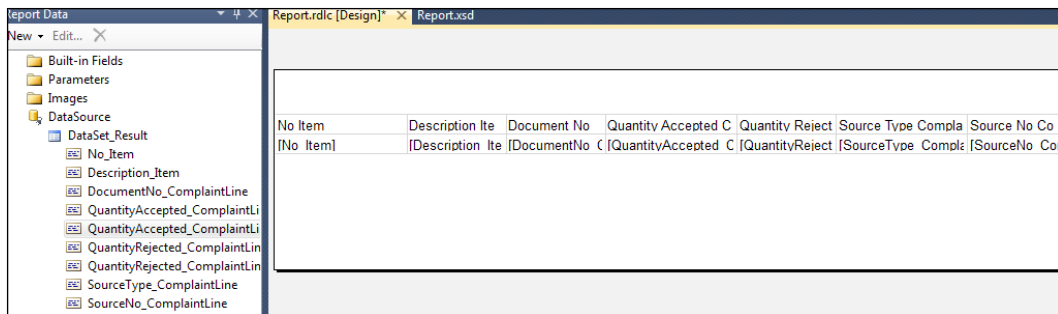


Keep adding columns until you have seven columns.

The next step is to add the dataset into our report. In Visual Studio 2010, click on **View | Report Data**. Drag the dataset into each of the columns. A note about dragging decimal or date fields: you always want to drag the dataset that does not end with a format if it's available. In our example, we have:

- **QuantityAccepted\_CommentLine**
- **QuantityAccepted\_CommentLineFormat**

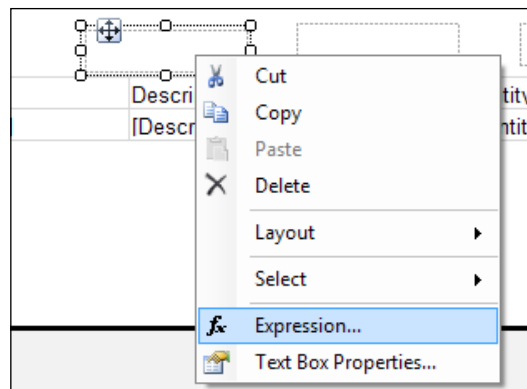
You always want to choose **QuantityAccepted\_CommentLine**. The same description that ends with the word **Format** is used to transfer the format settings from Dynamics NAV to the generated report. For the purpose of this book, we will not define any special format to be displayed.



On the last step of our report, we want to add the title displaying the date, company name, and so on. In other words, format it to be very similar to other out-of-the-box Dynamics NAV reports.

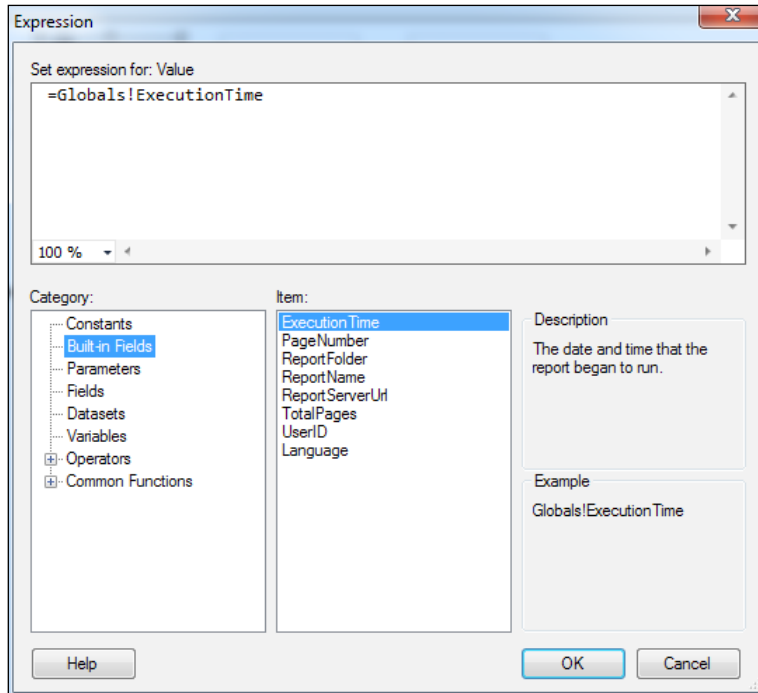
Bring up the **Toolbox** option again from **View | Toolbox**. Drag two textboxes into the report layout above our table.

Right-click on the textbox and select **Expression...**:





Under **Category**, choose **Built-in Fields**, double-click on **ExecutionTime**, and then choose the **OK** button. This expression displays the time the report was executed:

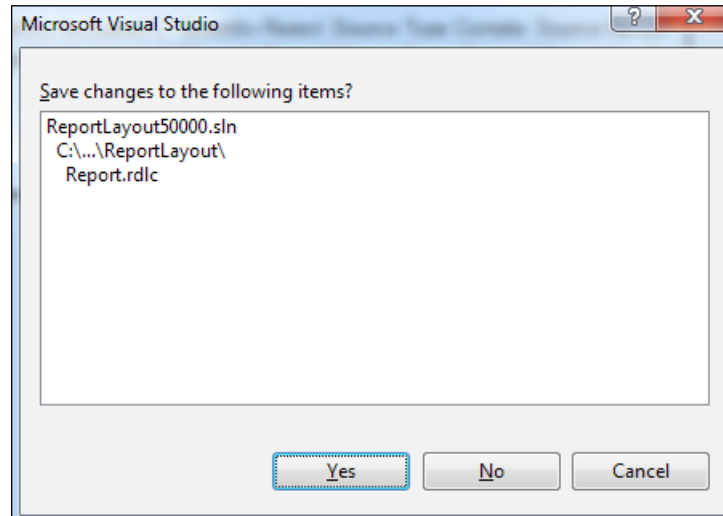


Repeat this for **UserID** on one of the other textboxes. This expression displays the user ID of the user who runs the report.

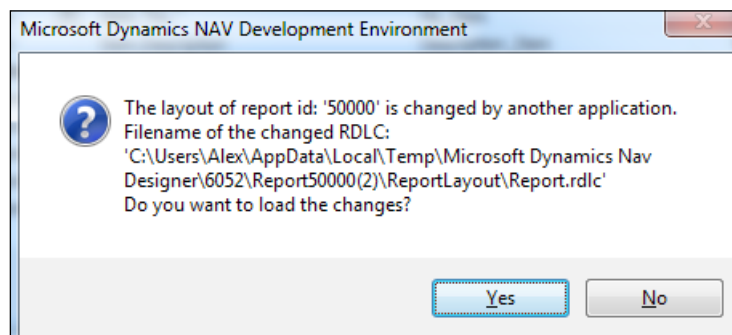
Once you've done that, move the controls on the report around so it's aesthetically pleasing:

|           |                 |             |                     |                 |                   |               |                  |
|-----------|-----------------|-------------|---------------------|-----------------|-------------------|---------------|------------------|
|           |                 |             |                     |                 |                   |               | [&UserID]        |
|           |                 |             |                     |                 |                   |               | [&ExecutionTime] |
| No Item   | Description It  | Document No | Quantity Accepted C | Quantity Reiect | Source Type Compl | Source No Co  |                  |
| [No Item] | [Description It | [DocumentNo | [QuantityAccepted C | [QuantityReiect | [SourceType Compl | [SourceNo Con |                  |

Close Visual Studio 2010 and it will ask you to save the changes to the RDLC layout:



Click on **Yes** to go back to **Report Designer** in Dynamics NAV. When you close this report again, it'll ask you if you want to load the RDLC changes into the report:



Click on **Yes**. At the final prompt, it will ask you if you want to save and compile; click on **Yes** to this also. Congratulations, you've just created your first Dynamics NAV 2013 report. When you run the report now, it will be blank. The reason is that we haven't entered any data for the report to display. We will go through entering data so we can test the printed layouts as we progress in our project.

There are a lot of additional features in the new reporting functionality that are not covered in this book, such as adding links, charting, and visibility toggles.

For more examples on report creation in Dynamics NAV 2013, go here: [http://msdn.microsoft.com/en-us/library/dd338904\(v=nav.70\).aspx](http://msdn.microsoft.com/en-us/library/dd338904(v=nav.70).aspx).

## **Summary**

In this chapter, we've created the user interface for the user to access the data. We've also created a simple analysis report for the user to generate the complaints per item.

We want to pay close attention to the design concepts of the standard Dynamics NAV. If there are any points when you're creating a project and you're not sure how a certain function should look and feel, always reference what's been done in Dynamics NAV.

In the next chapter, we will extend our program by adding some functionalities and usability improvements. We will add C/AL code to our project so some fields can be defaulted for us. We will wrap up our project and make sure all of the requirements are addressed.

# 8

## Extending Our Application

*"If there's a trick to it, the UI is broken."*

*– Douglas Anderson*

We've created a pretty good framework with our application. We created the necessary tables and pages to hold and display the data. We will perform a test with our application to see what other functionality should be added to add to the overall user experience.

As we work with our application, there will be some areas that are not on the task list, but they should still be tweaked so our application is more user friendly. We will extend the functionality of our application with the use of the C/AL code and by changing the property of table fields.

Note that in this chapter, we will be accessing the code behind objects. You will not be able to access the C/AL codes without using an end user license with the Application Builder granule or a Solution Developer license.

If you do not have access to the end user license with the Application Builder on your license and you would like to follow along, please use the Dynamics NAV cloud site described in *Chapter 1, Getting Dynamics NAV 2013 on Your Computer – For (Almost) Free*.

## A quick look at our user requirements

Before we continue, let's take a look at where we left off previously on our requirement list:

|    | A    | B   |
|----|------|---|
| 1  | Done | Requirement   |
| 2  | x    | Every incident needs a unique number so it can be tracked                           |
| 3  | x    | The customer needs to be associated with every incident                             |
| 4  | x    | The vendor that supplied the good, if applicable, needs to be tracked               |
| 5  | x    | The date of the interaction must be tracked   |
| 6  | x    | The item number in question   |
| 7  | x    | The quantity of the item that was accepted and rejected                             |
| 8  | x    | The source of the production order or purchase order                                |
| 9  | x    | Detailed comments that is associated with every QMS entry                           |
| 10 | x    | There can be 1 or more items per incident   |
| 11 | x    | Any QMS document that has been resolved will need to be marked as so                |
| 12 |      | The open QMS documents and the closed ones should be displayed as a separate screen |
| 13 | x    | A report needs to be created on the feedback per item                               |

Looking at this list, it looks like we've covered all of the user requirements that were laid out for us. The only item that's missing is creating a QMS document for the open and closed QMS documents, or documents that have been processed. We will take care of that in this chapter.

## A quick test of our application

Let's take a look at what we've created so far. Click on the **Object Designer** option in the Development Environment, find page 50000, **Product Complaint**, and click on **Run**.

When you do this, a page will be displayed from the Windows Client showing our page. By default, when you run the page from the Object Designer, it will be in the View Only mode. Click on the **New** button to create a new record and enter the following data in the header fields:

- **No.:** Type in 10000
- **Customer No.:** Select customer 10000 - **The Cannon Group**
- **Vendor No.:** Select vendor 10000 - **London Postmaster**
- **Document Date:** Enter today's date

In the **Product Complaint Subform** section, enter the following data:

| Item No. | Quantity Accepted | Quantity Rejected |
|----------|-------------------|-------------------|
| 70000    | 1                 | 2                 |
| 70001    | 3                 | 1                 |
| 70002    | 12                | 1                 |

After you're done entering the data, your complaint screen should look like this:

The screenshot shows the 'Edit - Product Complaint - 10000' window. The 'General' tab is active, displaying the following fields:

- No.: 10000
- Customer No.: 10000
- Name:
- Address:
- Address 2:
- City:
- Post Code:
- County:
- Contact Name:
- Vendor No.: 10000
- Vendor Name:
- Document Date: 2/27/2013
- Resolved:

The 'Product Complaint Subform' table is displayed below the general fields:

| Item No. | Description | Quantity Accep... | Quantity Reject... | Source Type    | Source No. |
|----------|-------------|-------------------|--------------------|----------------|------------|
| 70000    |             | 1.00              | 2.00               | Purchase Re... |            |
| 70001    |             | 3.00              | 1.00               | Purchase Re... |            |
| 70002    |             | 12.00             | 1.00               | Purchase Re... |            |

The 'Customer Credit Information' panel on the right shows the following details:

- Contact: Mr. Andy Teal
- Phone No.:
- Collection Method:
- Blocked:
- Credit Limit (LCY): 0.00
- Balance (LCY): 259,054.90
- Difference: -259,054.90
- Difference: 259,437.76
- Aging as of 01/23/14 (showing days overdue): -382.86
- 31-60 Days: 0.00
- Over 60 Days: 0.00
- Payment Terms Code: 1M(8D)
- Payment Method Code:
- Latest Payment Date: 1/12/2014
- Latest Payment Amount: 104,339.38

Right off the bat, you will notice some glaring usability problems. They are as follows:

- The **No.** field should automatically be generated based on a number series; similar to the sales order
- The name, address, and so on, should automatically be populated when the **Customer No.** is entered
- The **Vendor Name** field should automatically be populated when the **Vendor No.** field is entered

- The **Description** field on the lines should automatically be populated with the item description after the **Item No.** field is entered
- There's no way for the end user using the Windows Client to access the report that we created in the previous chapter
- The **Quantity Accepted** and **Quantity Rejected** fields should display no decimal places if no decimals are entered

These tasks are not necessarily the actual requirements coming from the users; however, they are usability enhancements you should make for an application that the end users will use on a daily basis.

Note these usability problems and add them to our requirements list so that we can keep track of these tasks and cross them off once we've completed them. Once we've updated our list and we know our task, we can begin to knock them out.

## Generate unique document numbers automatically

Let's do a quick check on our requirement list:

| 1  | Done | Requirement   |
|----|------|---|
| 2  | x    | Every incident needs a unique number so it can be tracked   |
| 3  | x    | The customer needs to be associated with every incident   |
| 4  | x    | The vendor that supplied the good, if applicable, needs to be tracked   |
| 5  | x    | The date of the interaction must be tracked   |
| 6  | x    | The item number in question   |
| 7  | x    | The quantity of the item that was accepted and rejected   |
| 8  | x    | The source of the production order or purchase order  |
| 9  | x    | Detailed comments that is associated with every QMS entry   |
| 10 | x    | There can be 1 or more items per incident   |
| 11 | x    | Any QMS document that has been resolved will need to be marked as so  |
| 12 |      | The open QMS documents and the closed ones should be displayed as a separate screen   |
| 13 | x    | A report needs to be created on the feedback per item   |
| 14 |      |   |
| 15 |      | The <b>No.</b> field should automatically be generated based on a number series; similar to the sales order                                   |
| 16 |      | The name, address, etc. should automatically be populated when the <b>Customer No.</b> is entered   |
| 17 |      | The <b>Vendor Name</b> should automatically be populated when the <b>Vendor No.</b> is entered  |
| 18 |      | The <b>Description</b> field on the lines should automatically be populated with the item description if the <b>Item No.</b> field is entered |
| 19 |      | There's no way on this page to access the report that we created in the previous chapter  |
| 20 |      | The <b>Quantity Accepted</b> and <b>Quantity Rejected</b> should display no decimal places if no decimals are entered                         |

To have a record automatically generate a unique number, we will use the existing Dynamics NAV functionality called number series (shortened to **No. Series** in Dynamics NAV). The purpose of the number series functionality is to allow the system to keep track of a unique number for you and save the number used in a table. This function is typically used for primary key fields, as we will use with our application.

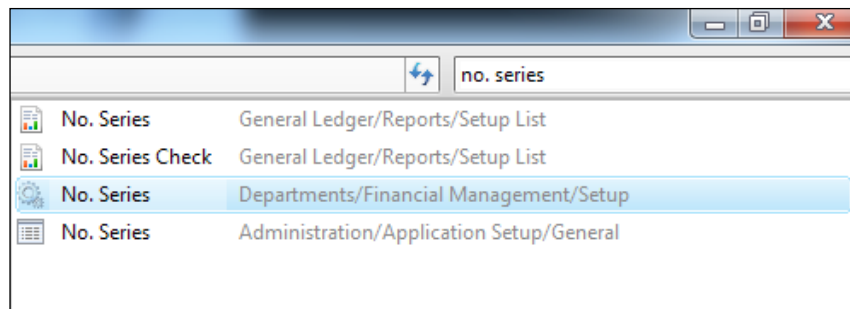
For example, if we set our sales invoice number series to start from INV-100000, Dynamics NAV will automatically increment INV-100000 by one numeric digit when we call this function. In this case, the next value to be given will be INV-100001.

If you want to learn more about the number series function in Dynamics NAV, you can read about it here:

[http://msdn.microsoft.com/en-us/library/hh879485\(v=nav.70\).aspx](http://msdn.microsoft.com/en-us/library/hh879485(v=nav.70).aspx)

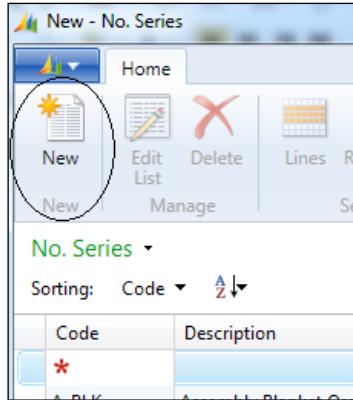
## Creating a number series for our application

The first thing we need to do in order for us to be able to use a number series functionality is to create one. From the Dynamics NAV 2013 client, use the search function to search for **No. Series**. There are a few **No. Series** options; make sure to choose the icon with the gears, as it's an option for the setup:

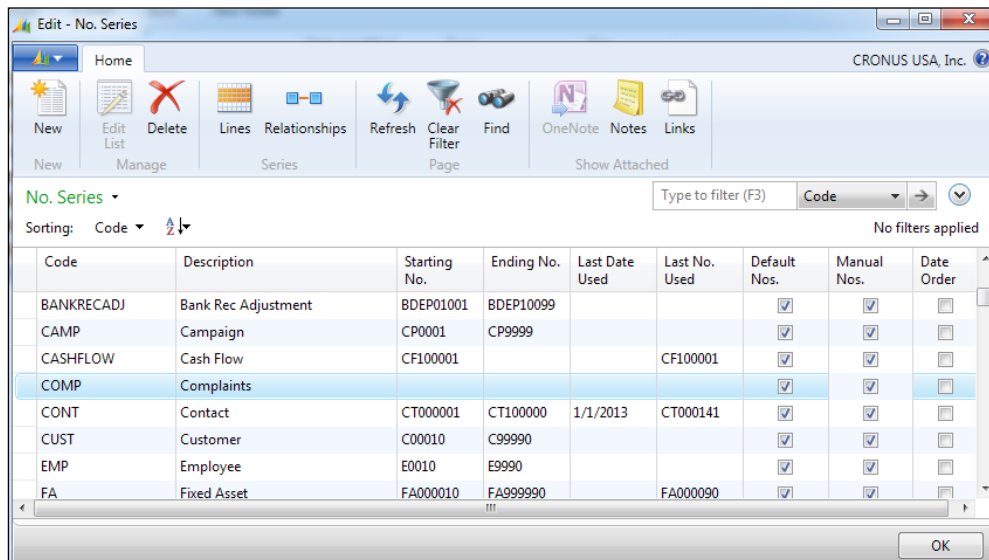




On the **No. Series** page, click on **New** to generate a new number series code for our complaints module:

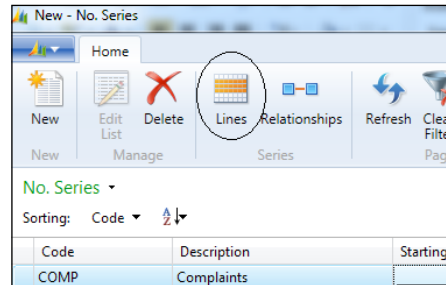


For our number series, put in **COMP** as a unique identifier in the **Code** field, and for the **Description** field, put in **Complaints**:



Make sure to put a check mark on the **Default Nos.** and **Manual Nos.** fields. Checking the **Default Nos.** field indicates that we want the system to assign a number for us automatically. Checking the **Manual Nos.** field allows the user to assign a number manually.

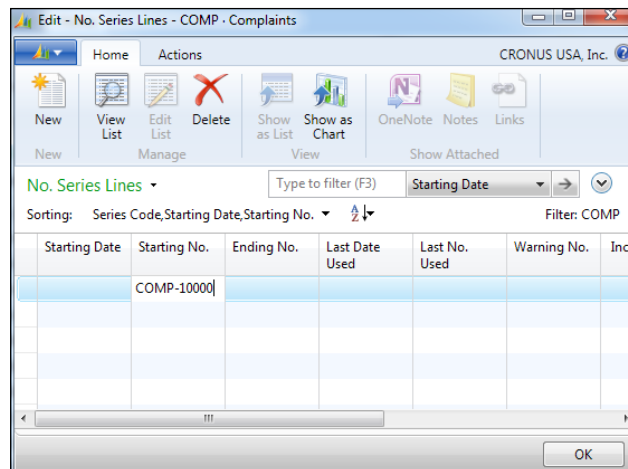
Click on the **Lines** button to bring up the page for us to set up the numbering sequence:



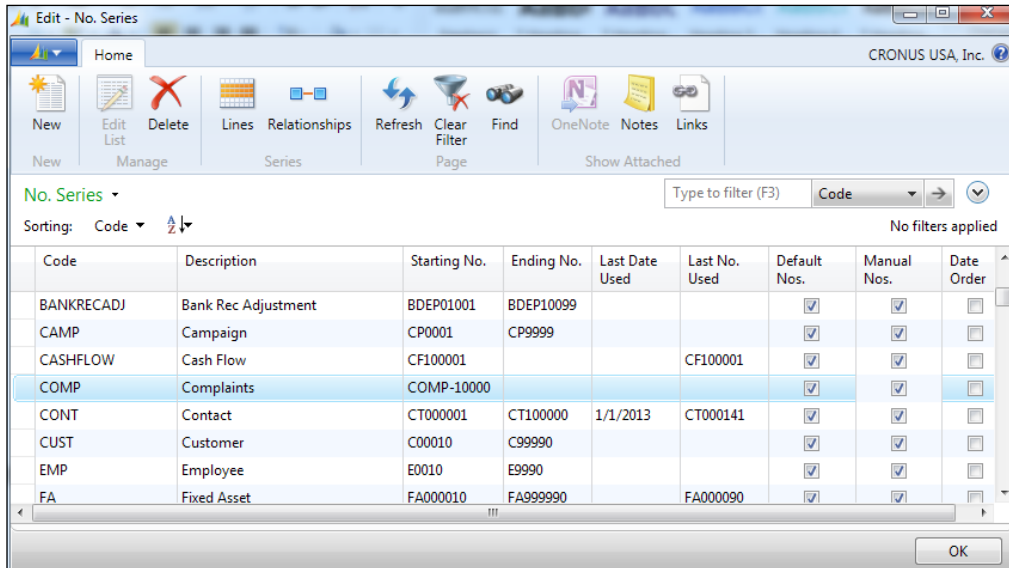
We want to assign a number series that uniquely identifies the record the user is working on. It's not a good idea to assign a generic number like 10000 because, throughout your organization, there will be many numbers that you have to keep track of. There are sales order numbers, invoice numbers, purchase order numbers, customer numbers, item numbers, and so on. There's a number for everything.

When creating a numbering logic for our documents, it's good practice to indicate what the document is for by abbreviating the type of document in the beginning of the number. For example, for sales orders, we define it as SO-10000; SO to indicate that it's a sales order. For purchase orders, we define it as PO-10000; PO to indicate it's a purchase order.

Using this best practice, we will do the same thing with our Complaints numbering. We will assign COMP-10000; COMP to indicate that it's a Complaints record followed by numbers. Put this value into the **Starting No.** field and click on **OK** to close the page:



When taken back to the **No. Series** page, you will notice that the **Starting No.** field on the **No. Series** page is now populated with our starting number:



Click on **OK** to close the screen.

## Programming our table for the number series

So how do we proceed to program this? Before we begin analyzing the code and putting in the codes we want, everything you need to know about C/AL coding in Dynamics NAV 2013 can be found at the following URL:

[http://msdn.microsoft.com/en-us/library/dd338764\(v=nav.70\).aspx](http://msdn.microsoft.com/en-us/library/dd338764(v=nav.70).aspx)

From this link, you will learn constants, variables, operators, functions, keywords, statements, expressions, controls, and all the other things in Dynamics NAV 2013 programming.

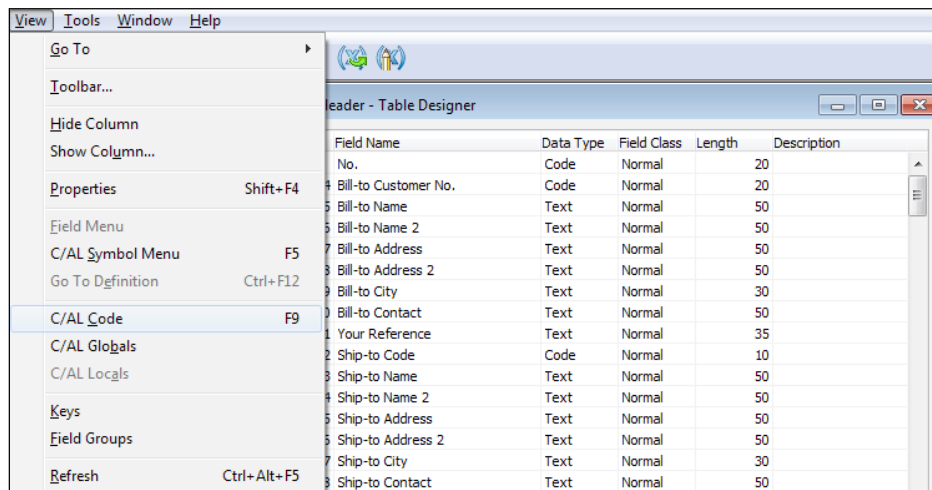
Individual pieces of code are nice, but they don't really have much value if they do not fit our purpose. As you are only starting with programming in Dynamics NAV, it may not be beneficial to go into every little detail about Dynamics NAV 2013 programming; that's not our job yet. Our job now is to create an application that addresses a specific business problem, preferably a business problem that can be resolved through copying and pasting from similar functions.

So, back to our original question: How do we program this? When in doubt, use the existing code as inspiration.

We've been basing the layout of our tables on the **Sales Order** table. When writing code, it will not be different.

Go to the Dynamics NAV 2013 Development Environment and bring up **Table 36 Sales Header** in the design mode through the **Object Designer**.

Click on **View | C/AL Code** to go to the C/AL editor. You can also press *F9*, which is the hot key.



When the C/AL screen comes up, you'll see a bunch of code that looks something like this:

```
OnInsert()
IF "No." = '' THEN BEGIN
    TestNoSeries;
    NoSeriesMgt.InitSeries(GetNoSeriesCode,xRec."No. Series","Posting Date","No.,"No. Series");
END;

InitRecord;
InsertMode := TRUE;

IF GETFILTER("Sell-to Customer No.") <> '' THEN
    IF GETRANGEMIN("Sell-to Customer No.") = GETRANGEMAX("Sell-to Customer No.") THEN
        VALIDATE("Sell-to Customer No.",GETRANGEMIN("Sell-to Customer No."));

IF GETFILTER("Sell-to Contact No.") <> '' THEN
    IF GETRANGEMIN("Sell-to Contact No.") = GETRANGEMAX("Sell-to Contact No.") THEN
        VALIDATE("Sell-to Contact No.",GETRANGEMIN("Sell-to Contact No."));

"Doc. No. Occurrence" := ArchiveManagement.GetNextOccurrenceNo(DATABASE::"Sales Header","Document Type","No.");

OnModify()

OnDelete()
DOPaymentTransLogMgt.ValidateCanDeleteDocument("Payment Method Code","Document Type",FORMAT("Document Type"),"No.");

IF NOT UserSetupMgt.CheckRespCenter(0,"Responsibility Center") THEN
    ERROR(
        Text022,
        RespCenter.TABLECAPTION,UserSetupMgt.GetSalesFilter);

IF ("Opportunity No." <> '') AND
    ("Document Type" IN ["Document Type"::Quote,"Document Type"::Order])
THEN BEGIN
    IF Opp.GET("Opportunity No.") THEN BEGIN
        IF "Document Type" = "Document Type"::Order THEN BEGIN
            IF NOT CONFIRM(Text040,TRUE) THEN

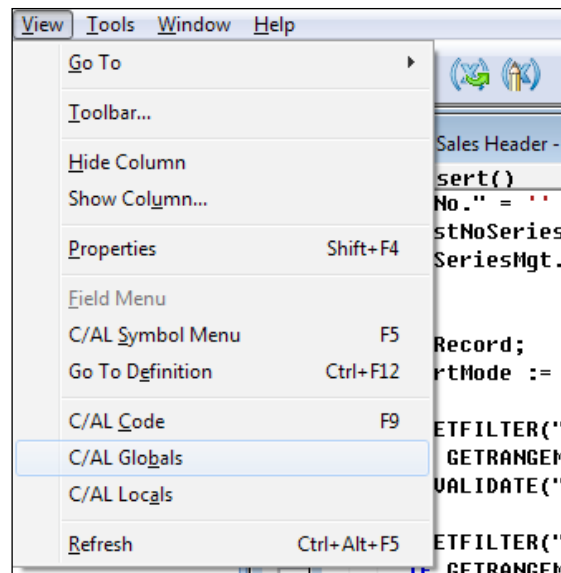
```

The code within Dynamics NAV are grouped by triggers. These triggers are fired when a particular action is taken by the user or another piece of code. In our example, the code in the OnInsert trigger runs when a record is inserted into the table. The OnDelete trigger runs when the record is deleted.

Every object in the C/AL environment will have a different set of triggers. For a full list of triggers for every object type in Dynamics NAV, go to the following link:

[http://msdn.microsoft.com/en-us/library/dd301068\(v=nav.70\).aspx](http://msdn.microsoft.com/en-us/library/dd301068(v=nav.70).aspx)

As with any development environment, there are variables. For our purpose, we will mainly work with global variables. To access the global variables in the object, click on **View | C/AL Globals**. There are also local variables that we can define, but we will keep things simple and define everything on a global level.



When the global variables screen is displayed, we will see all sorts of variables defined. In addition to the variables that are described in *Chapter 5, Finding Similar Functions for Inspiration*, you can also define complex data types such as tables, pages, reports, codeunits, and files as variables. For a complete list of the data types that you can define in the C/AL environment, nobody explains it better than Microsoft; go to:

[http://msdn.microsoft.com/en-us/library/dd338759\(v=nav.70\).aspx](http://msdn.microsoft.com/en-us/library/dd338759(v=nav.70).aspx)

Going back to the C/AL editor, when we press the *Page Up* key to go to the top of the screen, there's a piece of code that's interesting in the *OnInsert* trigger:

```

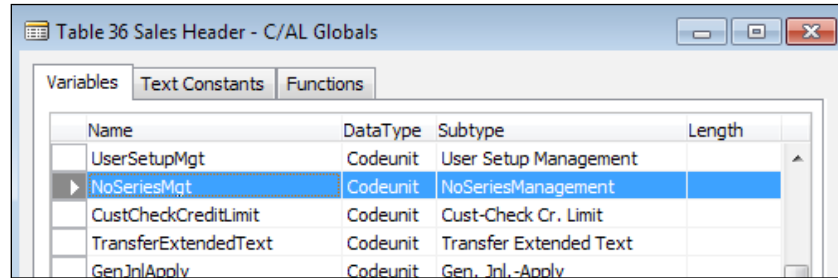
OnInsert()
IF "No." = '' THEN BEGIN
    TestNoSeries;
    NoSeriesMgt.InitSeries(GetNoSeriesCode,xRec."No. Series","Posting Date","No.,""No. Series");
END;

```

Why is this interesting? Here are a few reasons:

- We want the number to be automatically generated when the record is inserted
- We know we want to use the **No. Series** functionality in Dynamics NAV

- The variable is called **NoSeriesMgt**
- Looking at the C/AL global variables, the **NoSeriesMgt** variable is referenced to the **NoSeriesManagement** codeunit



An experienced NAV developer will use these tips to further the investigation on whether this is the correct function to use. To investigate, we would go to the actual codeunit, look at the function being called, and see if it fits what we need.

The skill of going through code will come with time working with Dynamics NAV. For the purpose of this book, we will confirm that the **NoSeriesMgt** variable is the correct function that we need to copy and paste into our table.

```
IF "No." = '' THEN BEGIN
    TestNoSeries;
    NoSeriesMgt.InitSeries(GetNoSeriesCode,xRec."No. Series","Posting
Date","No.,""No. Series");
END;
```

We will analyze this piece of code as follows:

- IF "No." = '' THEN BEGIN ... END;: Basically, if the **No.** field is blank, we will execute this code
- TestNoSeries;: This function tests whether the number series is properly set up in the setup table
- NoSeriesMgt.InitSeries(GetNoSeriesCode,xRec."No. Series","Posting Date","No.,""No. Series");: InitSeries is a function within the NoSeriesMgt codeunit to grab the next number in the sequence based on the date of the document

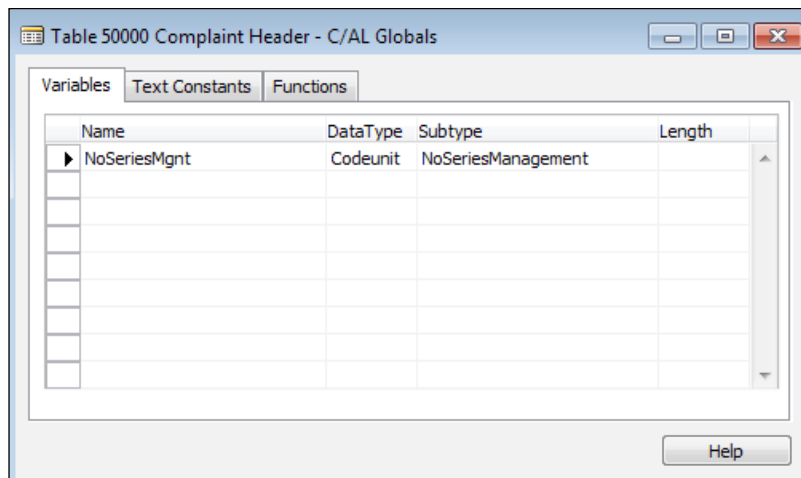
## Put our code in the table

Dynamics NAV has internal functions to perform certain tasks. The list of these internal C/AL functions for tables can be found here:

<http://msdn.microsoft.com/en-us/library/dd355032.aspx>

Now that we know what code to put in and where to put it, go back to the tables in **Object Designer** and go to our table **50000, Complaint Header**. Click on **Design** to enter the design mode.

The first thing we need to do is define the variables in our table. Click on **View** | **C/AL Globals**. Define the **NoSeriesMgt** variable as the **NoSeriesManagement** codeunit, or we can just copy and paste the variable from the **Sales Header** table. Be sure to highlight the whole line and copy and paste, otherwise you may just copy the name or the data type.



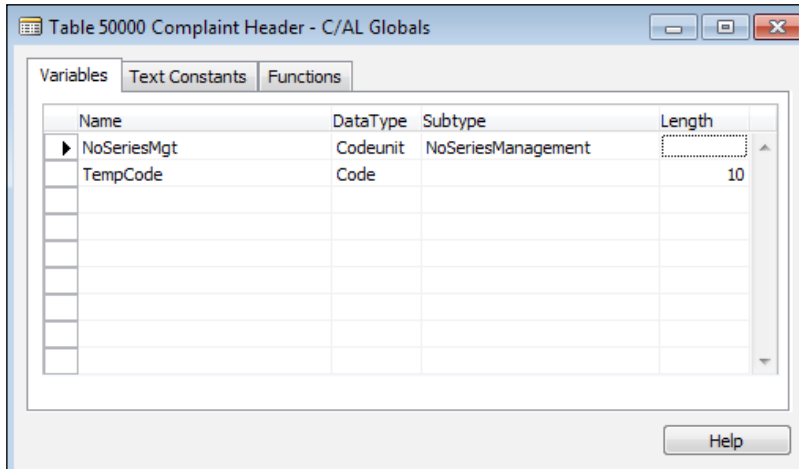
After the variable is defined, close the **C/AL Globals** screen and push **F9** to go into the C/AL code.

We will omit the `TestNoSeries` function because we're defining the number series directly within the function.

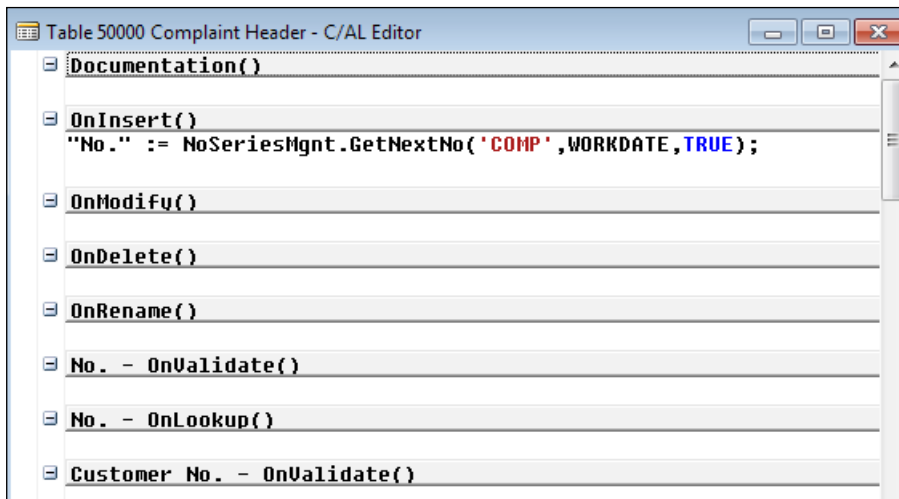


Hardcoding values is not the correct way to do this. The proper way is to create a **Complaint No. Series** field on the setup tables for the users to fill in what number series they wish to use. However, for the sake of moving on with our exercise, we will hardcode the **No. Series** value we want.

In order for this codeunit to work, we will need to define the **TempCode** variable in the **C/AL Globals** screen.



Close the **C/AL Globals** screen; push the *Page Up* or *Page Down* key on your keyboard, find the **OnInsert** trigger, and type in the following code:



Save what we have done so far. Make sure to leave the **Compile** checkbox on when prompted.

## Defaulting fields using code

Before we continue, let's check off what we've done in our requirement list:

|    | A    | B   |
|----|------|---|
| 1  | Done | Requirement   |
| 2  | x    | Every incident needs a unique number so it can be tracked   |
| 3  | x    | The customer needs to be associated with every incident   |
| 4  | x    | The vendor that supplied the good, if applicable, needs to be tracked   |
| 5  | x    | The date of the interaction must be tracked   |
| 6  | x    | The item number in question   |
| 7  | x    | The quantity of the item that was accepted and rejected   |
| 8  | x    | The source of the production order or purchase order  |
| 9  | x    | Detailed comments that is associated with every QMS entry   |
| 10 | x    | There can be 1 or more items per incident   |
| 11 | x    | Any QMS document that has been resolved will need to be marked as so  |
| 12 |      | The open QMS documents and the closed ones should be displayed as a separate screen   |
| 13 | x    | A report needs to be created on the feedback per item   |
| 14 |      |   |
| 15 | x    | The <b>No.</b> field should automatically be generated based on a number series; similar to the sales order                                   |
| 16 |      | The name, address, etc. should automatically be populated when the <b>Customer No.</b> is entered   |
| 17 |      | The <b>Vendor Name</b> should automatically be populated when the <b>Vendor No.</b> is entered  |
| 18 |      | The <b>Description</b> field on the lines should automatically be populated with the item description if the <b>Item No.</b> field is entered |
| 19 |      | There's no way on this page to access the report that we created in the previous chapter  |
| 20 |      | The <b>Quantity Accepted</b> and <b>Quantity Rejected</b> should display no decimal places if no decimals are entered                         |

The next task is to have our program fill in the customer name, address, and so on when the **Customer No.** field is filled in. We'll draw inspiration from our **Sales Header** table. Working with the **Sales Order** table, we know that the customer information is automatically populated when the **Sell-to Customer No.** field is entered.

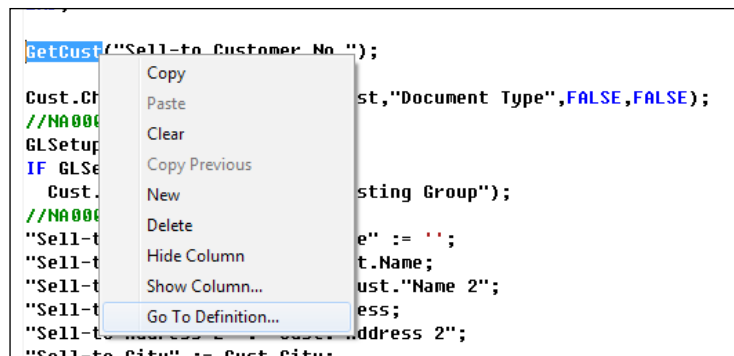
Let's go back to the **Sales Header** table, click on the **Sell-to Customer No.** field, and go to the C/AL code by pressing *F9*. Scroll down (or press *Page Down*) until you find some reference for **Address**.

```
Table 36 Sales Header - C/AL Editor
SalesLine.RESET;
END;

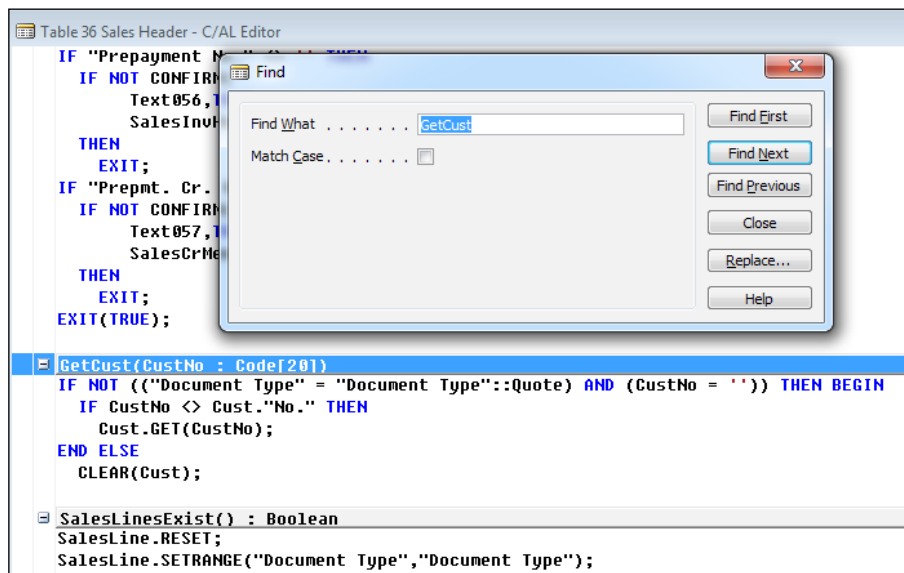
GetCust("Sell-to Customer No.");

Cust.CheckBlockedCustOnDocs(Cust,"Document Type",FALSE,FALSE);
//NA0002.begin
GLSetup.GET;
IF GLSetup."VAT in Use" THEN
    Cust.TESTFIELD("Gen. Bus. Posting Group");
//NA0002.end
"Sell-to Customer Template Code" := '';
"Sell-to Customer Name" := Cust.Name;
"Sell-to Customer Name 2" := Cust."Name 2";
"Sell-to Address" := Cust.Address;
"Sell-to Address 2" := Cust."Address 2";
"Sell-to City" := Cust.City;
"Sell-to Post Code" := Cust."Post Code";
"Sell-to County" := Cust.County;
"Sell-to Country/Region Code" := Cust."Country/Region Code";
IF NOT SkipSellToContact THEN
    "Sell-to Contact" := Cust.Contact;
"Gen. Bus. Posting Group" := Cust."Gen. Bus. Posting Group";
"VAT Bus. Posting Group" := Cust."VAT Bus. Posting Group";
"Tax Area Code" := Cust."Tax Area Code";
"Tax Liabile" := Cust."Tax Liabile";
"Tax Exemption No." := Cust."Tax Exemption No."; // NA0008
"VAT Registration No." := Cust."VAT Registration No.";
"VAT Country/Region Code" := Cust."Country/Region Code";
"Shipping Advice" := Cust."Shipping Advice";
"Responsibility Center" := UserSetupMgt.GetRespCenter(0,Cust."Respo
VALIDATE("Location Code",UserSetupMgt.GetLocation(0,Cust."Location
```

Looking at the code here, you can probably guess that **GetCust** is a function to get the customer information. To verify this, use the Find function (*Ctrl + F*) to search for **GetCust**. You can also highlight the **GetCust** word, right-click, and click on **Go to Definition...**

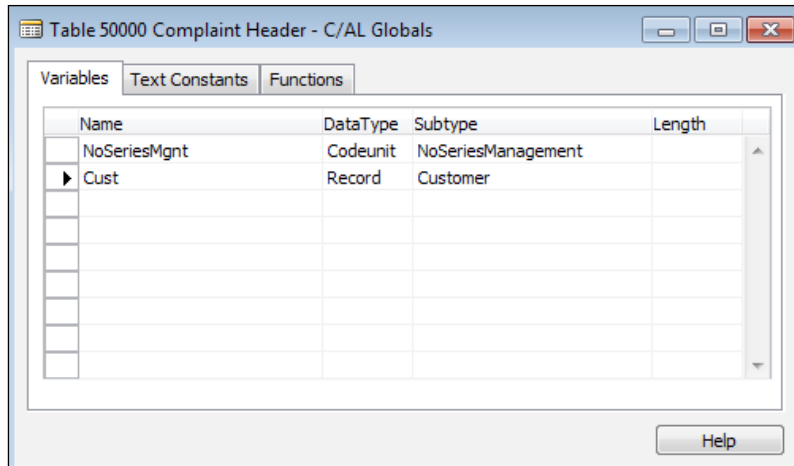


When we use **Go To Definition...**, the function in question will be brought up. If we use the Find (*Ctrl + F*) method, we will have to keep clicking on **Find Next** to get to the function in question.



Checking the C/AL global variables, you'll also find that **Cust** is a variable for the customer record.

Armed with this knowledge, let's put what we learned into our code. Go back to **Table 50000 Complaint Header** and go into the design mode. Go to the C/AL global variables and define the **Cust** variable to reference the **Customer** table in exactly the same way as the **Sales Header** table.



Close the **C/AL Globals** screen, click on the **Customer No.** field, and push **F9** to go into the C/AL code. Type in the following code in the **OnValidate** trigger of the **Customer No.** field:

```
Customer No. - OnValidate()
IF Cust.GET("Customer No.") THEN BEGIN
    Name := Cust.Name;
    Address := Cust.Address;
    "Address 2" := Cust."Address 2";
    City := Cust.City;
    "Post Code" := Cust."Post Code";
    County := Cust.County;
    "Contact Name" := Cust.Contact;
END;
Customer No. - OnLookup()
```

The **GET** function in Dynamics NAV gets the record based on the primary key. For more information on the **GET** function, look here:

<http://msdn.microsoft.com/en-us/library/dd301056.aspx>

## Defaulting fields using FlowFields

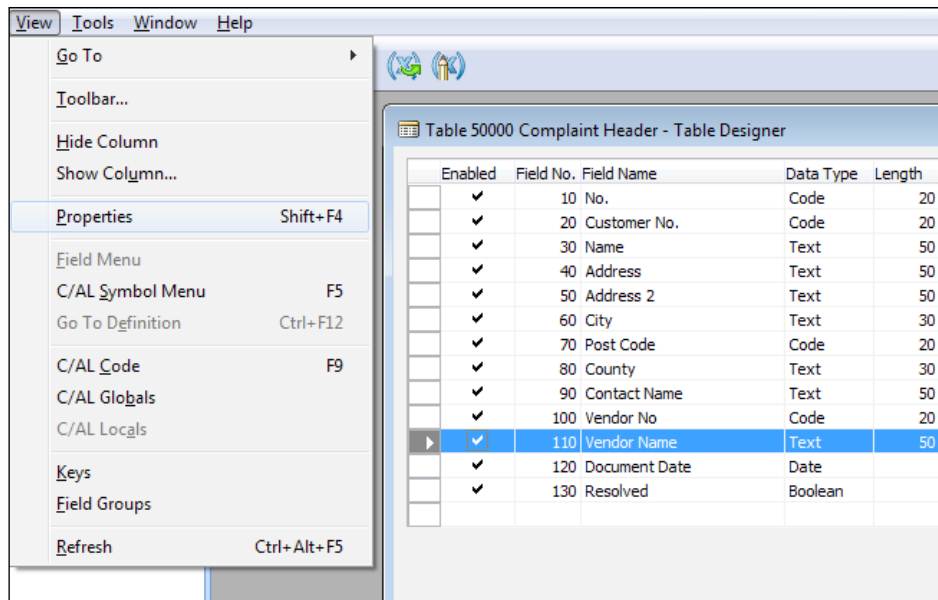
After we default the fields for the customer address information, we will also default the **Vendor Name** when the user enters the **Vendor No.** value.

Instead of using code this time, we will use the FlowFields feature in Dynamics NAV to display this information. For a full and detailed explanation on FlowFields, go here:

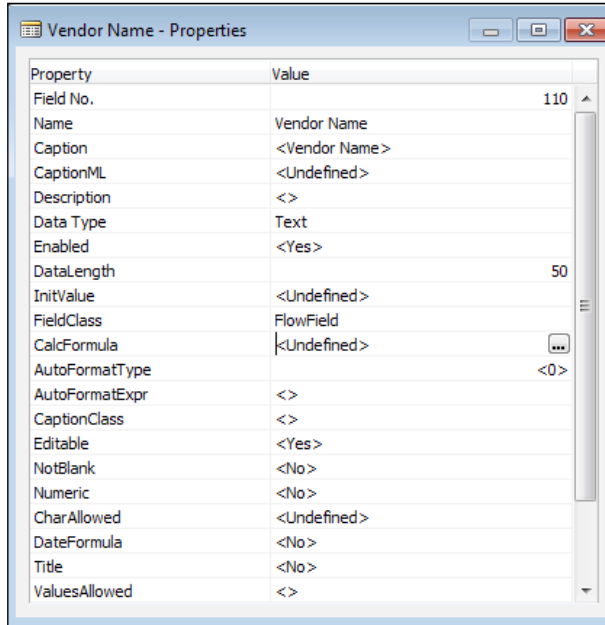
[http://msdn.microsoft.com/en-us/library/dd338766\(v=nav.70\).aspx](http://msdn.microsoft.com/en-us/library/dd338766(v=nav.70).aspx)

FlowFields are basically a way to quickly display calculated information in a field. We will replicate this, but instead of calculating numbers, we will look up information.

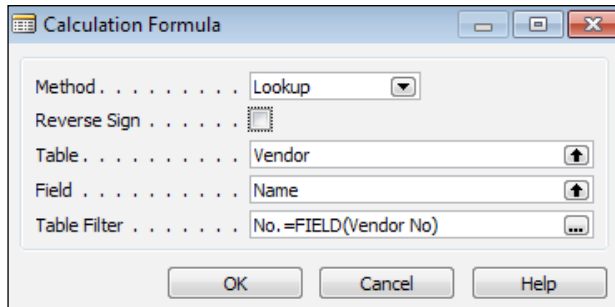
Go to **Properties** for the **Vendor Name** field:



Change the **FieldClass** property to **FlowField**. Click on the **AssistEdit** button in the **CalcFormula** property.



Here you want to put in a formula to **Lookup** the **Vendor** table in the **Name** field where the **No.** field on the **Vendor** table is the same as the **Vendor No.** in the **Complaint Header** table. The following screenshot describes what we should have in place:



Click on **OK** once you've defined your formula. Make sure to change the **Editable** property to **No.** FlowFields are not actual fields that hold data. They are calculated fields, and making them not editable will reduce the confusion for the users.

When you're done, save (and compile) the table. Go back to the **Object Designer**.

## Defaulting an item description on the line table

Using the skills we've gained from defaulting values on the fields, we will now try defaulting the **Description** field from the **Item** table when the user enters a value in the **Item No.** field.

For the coding method, or the method that's more consistent with out of the box Dynamics NAV, the table we want to use is the **Item** table. The field we want is the **Description** field from the **Item** table.

A reason why the FlowFields method would not work in the **Complaint Line** table is that if we set the field to be a FlowField, we will need to set the **Description** field as non-editable. This will not work because the user should be able to type in their own descriptions as they are entering data in the complaint lines.

## Changing the properties of the decimal values

Let's take a quick glance at our requirement list to see what else we have left:

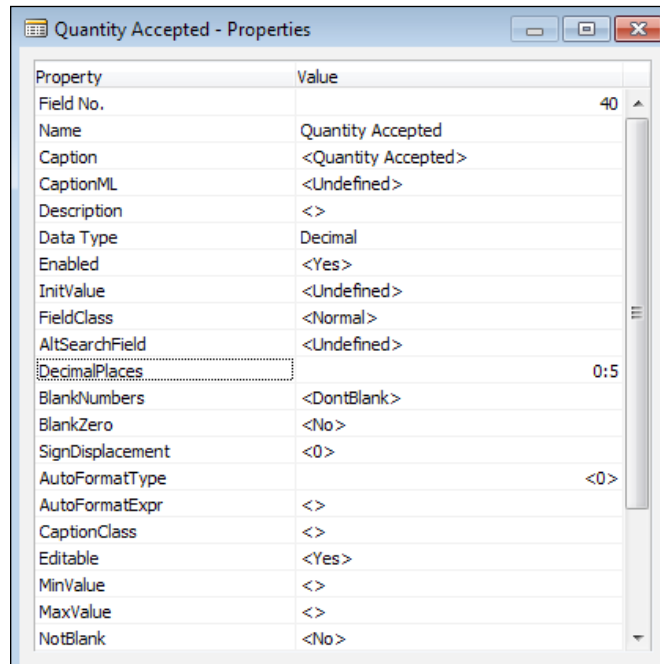
|    | A    | B   |
|----|------|---|
| 1  | Done | Requirement   |
| 2  | x    | Every incident needs a unique number so it can be tracked   |
| 3  | x    | The customer needs to be associated with every incident   |
| 4  | x    | The vendor that supplied the good, if applicable, needs to be tracked   |
| 5  | x    | The date of the interaction must be tracked   |
| 6  | x    | The item number in question   |
| 7  | x    | The quantity of the item that was accepted and rejected   |
| 8  | x    | The source of the production order or purchase order  |
| 9  | x    | Detailed comments that is associated with every QMS entry   |
| 10 | x    | There can be 1 or more items per incident   |
| 11 | x    | Any QMS document that has been resolved will need to be marked as so  |
| 12 |      | The open QMS documents and the closed ones should be displayed as a separate screen   |
| 13 | x    | A report needs to be created on the feedback per item   |
| 14 |      |   |
| 15 | x    | The <b>No.</b> field should automatically be generated based on a number series; similar to the sales order                                   |
| 16 | x    | The name, address, etc. should automatically be populated when the <b>Customer No.</b> is entered   |
| 17 | x    | The <b>Vendor Name</b> should automatically be populated when the <b>Vendor No.</b> is entered  |
| 18 | x    | The <b>Description</b> field on the lines should automatically be populated with the item description if the <b>Item No.</b> field is entered |
| 19 |      | There's no way on this page to access the report that we created in the previous chapter  |
| 20 |      | The <b>Quantity Accepted</b> and <b>Quantity Rejected</b> should display no decimal places if no decimals are entered                         |



Things are looking good! We're almost complete with our project! There are three additional points that we need to change, after which we'll be ready for testing.

Open the **Object Designer** in the Dynamics NAV 2013 Development Environment. Bring up the **Table Designer** screen for table **50001, Complaint Line**.

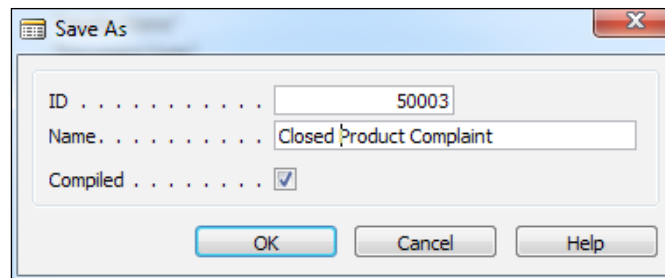
Click on the **Quantity Accepted** field and click on **View | Properties**. Change the **DecimalPlaces** property to **0:5**. This means that the table will display **0** decimals at the minimum and **5** decimals at the maximum.



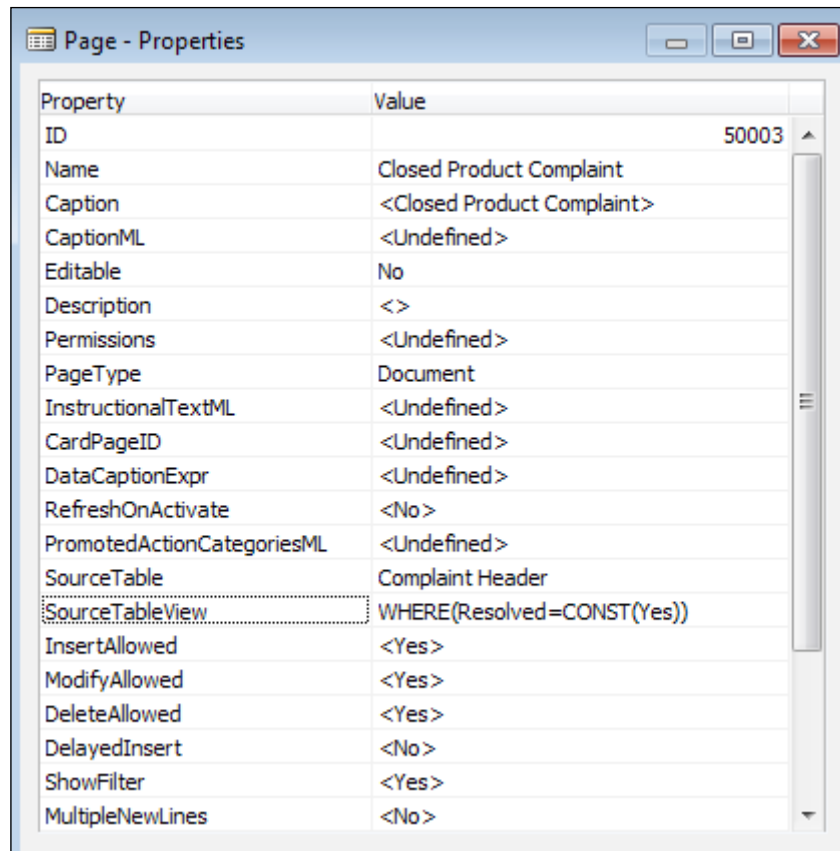
For a complete list of the field properties and what they mean, go to the following URL: <http://msdn.microsoft.com/en-us/library/dd301103.aspx>.

## Creating a separate screen for closed complaints

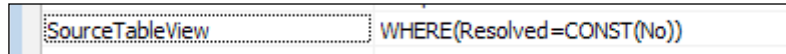
One thing that's been outstanding for a long time in Dynamics NAV is having a screen for closed complaints. Let's take care of that one. Remember, the best tool a programmer can use is to look at existing functions and draw inspiration from them. In the **Object Designer**, click on **Page** and then go into the design mode for page **50000**. Click on **File | Save As**. Save the new page as **50003** and name it **Closed Product Complaints**.



We do not want the users to change any information on closed product complaints. Go to the very bottom of the **Page Designer** and click on **View | Properties**. Change the **Editable** property to **No**. In addition to that, we only want to see the transactions that are resolved. We will change the property on **SourceTableView** where resolved is **Yes**.



Close and save this page. Go back to page **50000** and specify the same condition for the **SourceTableView** in the **Page** property, except this time, we will only look at complaints that are not resolved.

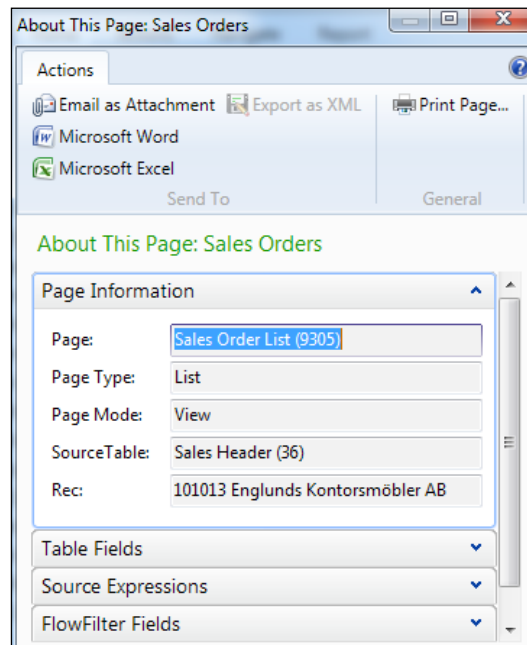


Close and save the page.

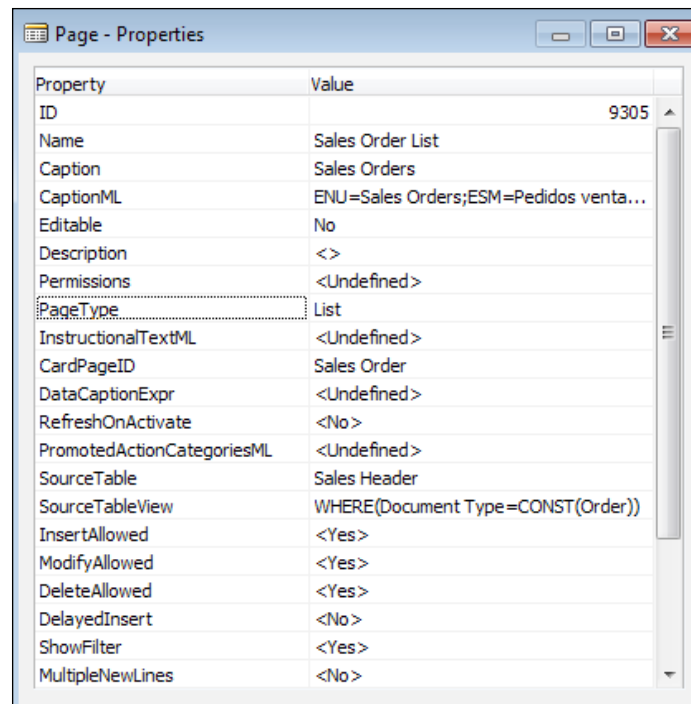
## Creating the list page

Looking through the RTC, you'll notice that when you click on **Sales Order**, the first thing that appears is the list of the orders instead of the document page. From there, you can click on **New** to create a new order or **Edit** to edit an order and only then would the document page be displayed.

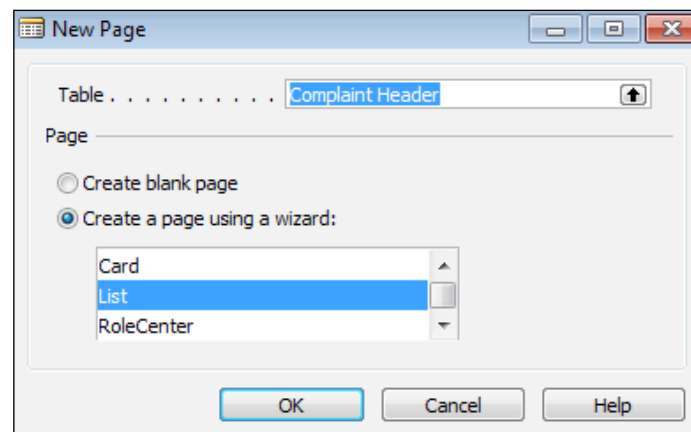
To look at how the list page is created, we need to find the page ID in the **Object Designer**. Using the skills that we've gained from *Chapter 5, Finding Similar Functions for Inspiration*, we will use the **About This Page** screen to find the page ID.



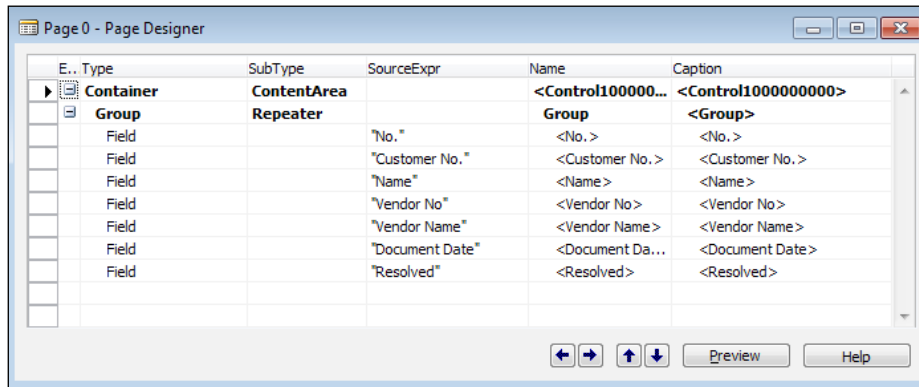
Go back to the **Object Designer** and bring up the **Page - Properties** window for page **9305**. Verify the **PageType** and **SourceTable** properties so that we can model it to create the **Product Complaints** list page.



Close the **Page Designer** for 9305. We now know that we need to create a list page with the **Source** table as the header. From the **Object Designer**, click on **New** to create a new page. Enter `Complaint Header`, table 50000, and use the wizard to create the list page.

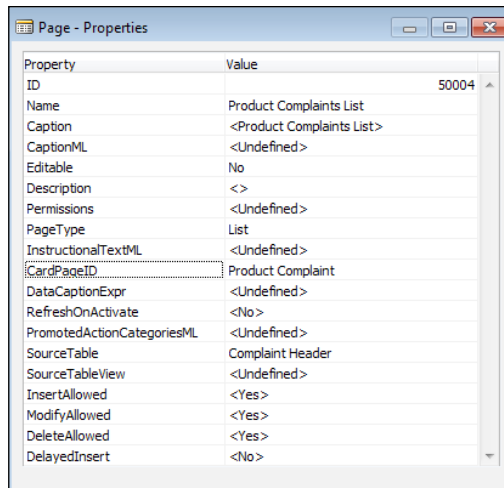


Go through the wizard as described in *Chapter 7, Creating the Application – Pages and Reports*, to create our list page. Add the fields that you want to display as the list. For extra credit, you can add FactBoxes to the list page as described in *Chapter 7, Creating the Application – Pages and Reports*. For this example, we will not add any FactBoxes. When you finish, your page should look something like this:

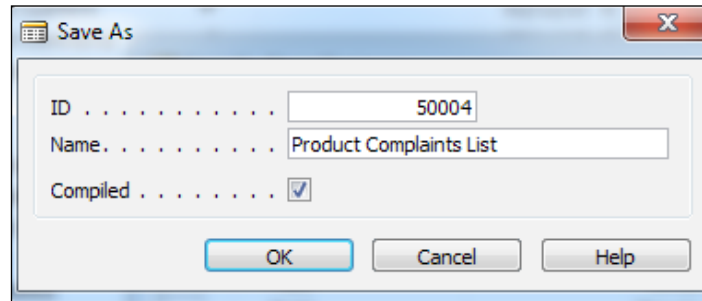


One other attribute you'll notice when working with the list page is that it's generally non-editable. So, we will need to go to the **Page - Properties** window to change the **Editable** property. Press *Page Down* on your keyboard until you hit a new line, and then click on **View | Properties**. Change the **Editable** property to **No**.

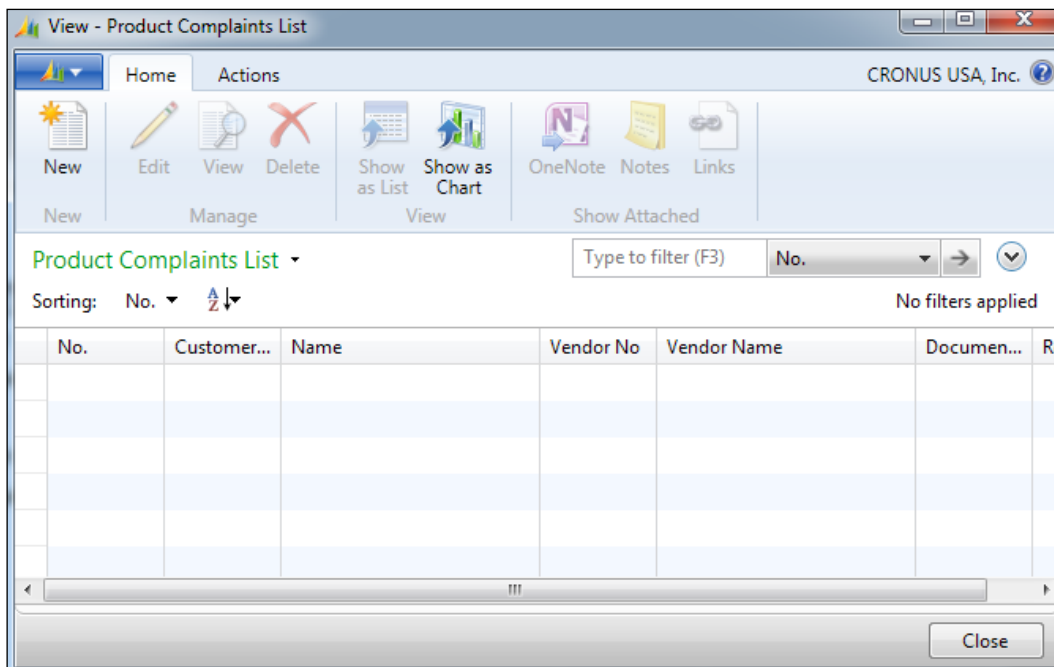
If you looked closely at the **Page - Properties** window, you'll also notice that **CardPageID** is defined. This property will tell the system what page to call when we want to create or edit a new complaint record. Enter page 50000, Product Complaint, in this property.



Close the **Properties** screen and close out of the **Page Designer**. When it prompts you to save, do so, and then give our list page an ID of 50004 and name it **Product Complaint List**.

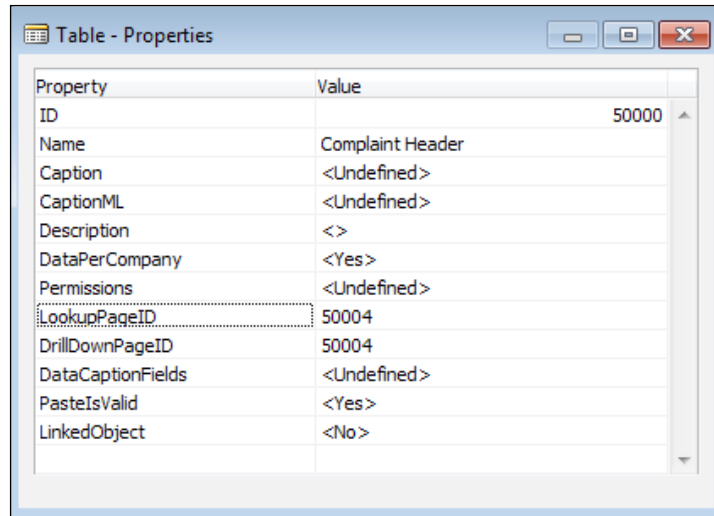


When you run page 50004, you should get the following screen:



The last step in creating a list page is to tie this page to the table. Have you ever wondered how Dynamics NAV knows which page to bring up when you look up a record? It's not done by magic; it's done in the table properties.

Go to **Table 50000 Complaint Header** and bring up the **Table Designer** screen. Go to the table properties and specify the **LookupPageID** and **DrillDownPageID** properties. Instead of typing the full name of the page object, we can reference it using the object ID. In this case, the object ID for **Product Complaint List** is **50004**.



After setting these properties, close the **Table Designer** screen and save the table.

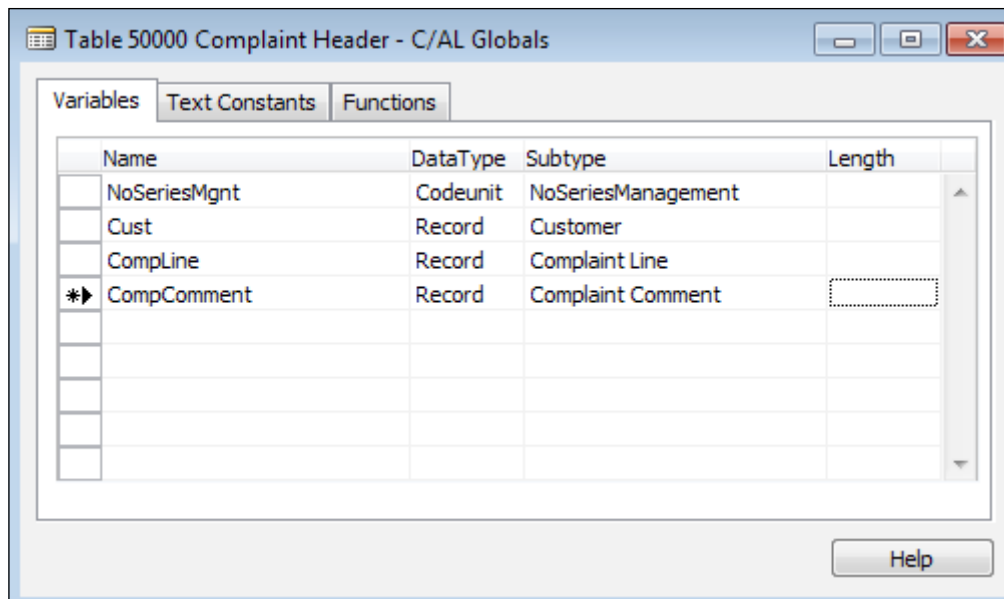
## Data clean up

When a new complaint record is created, it will be inserted into the **Complaint Header** table. When the lines are added, it will be inserted into the **Complaint Line** table. When comments are added, it will add it to the **Complaint Comment** table.

All is fine and dandy when we're adding, but what about when the records are deleted? If a user goes in and deletes a **Complaint Header** record from the **Product Complaints** page, how will Dynamics NAV know to delete the associated **Complaint Line** and **Complaint Comment** records?

The answer unfortunately is not magic (although sometimes I wish it would be); it's done through C/AL programming on the **Complaint Header** table. Like a good student, you would've gone to the **Sales Header** table and looked at the `OnDelete` trigger to draw inspiration.

Let's put the clean-up code into our **Complaint Header** table. Go to **Table 50000** and bring up the **Table Designer** screen. The first thing we need to do is to declare the C/AL Globals for the **Complaint Line** and **Complaint Comment** tables. Go ahead and do that now.



Once you've declared the variables for the record, close the **C/AL Globals** screen and press **F9** to access the C/AL code. Exactly below the `OnDelete` trigger, put in the following code:

```

UNMODIFY()
OnDelete()
    CompLine.SETRANGE("Document No.", "No.");
    CompLine.DELETEALL;

    CompComment.SETRANGE("Document No.", "No.");
    CompComment.DELETEALL;

OnRename()

```

The `SETRANGE` function sets the filter on the tables. In the statement, we are filtering the **Complaint Line** table based on the current **No.** value in the **Complaint Header** table.

The `DELETEALL` function deletes all of the records. Be careful while using the `DELETEALL` function. If you use this without using the `SETRANGE` or `SETFILTER` functions, you'll delete all of the records in the table!

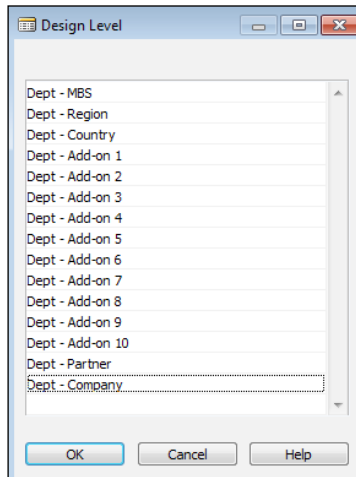
Close the **Table Designer** for table 50000 and save.



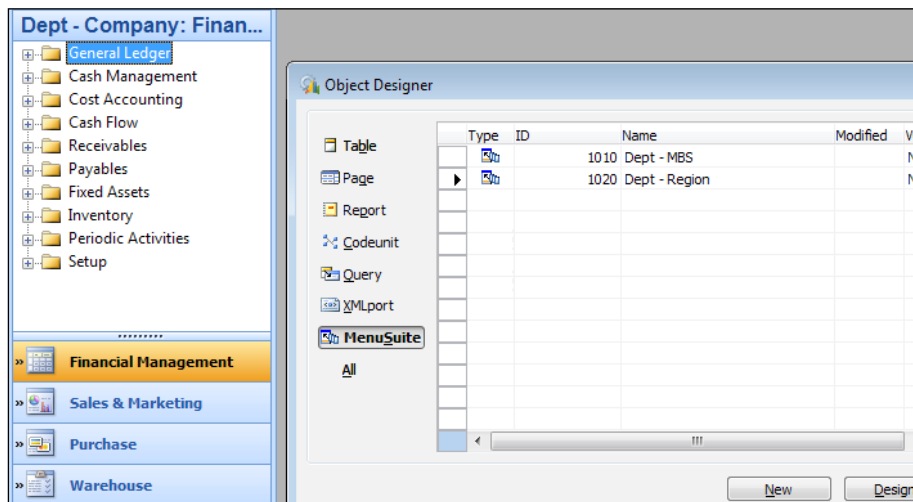
## Adding the application to the RTC menu

Let's add both report 50000, **Item Complaint Log** and page 50004, **Product Complaint List** to the RTC menu so the users can access them.

Go back to the **Object Designer** and click on **MenuSuite**. Click on **New** to add a new MenuSuite for our company. When the **Design Level** screen appears, select **Dept - Company** and click on **OK**.



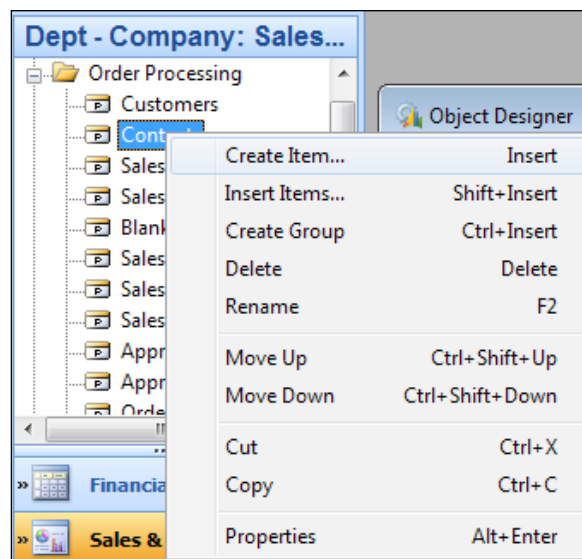
The **MenuSuite** designer is a new panel that will appear in the **Object Designer** screen.



If you click through each of the functional areas such as **Financial Management**, **Sales & Marketing**, and **Purchase**, you'll notice that the layout is identical to the Windows Client when you click on **Departments**. Why? The reason is because this is where the **Department** menu in the Windows Client comes from.

The place where the report and the **Complaint** page would make the most sense is either in **Sales & Marketing**, **Purchase**, **Warehouse**, and/or **Manufacturing**. For our purpose, let's add our module to **Sales & Marketing**.

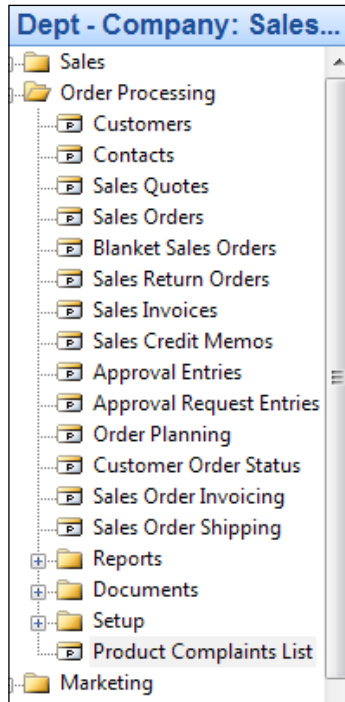
Click on **Sales & Marketing** and expand the **Order Processing** folder. Right-click anywhere below the **Order Processing** folder and click on **Create Item**.



When the **Create Item** menu pops up, fill in our page 50004, **Product Complaints List** details.

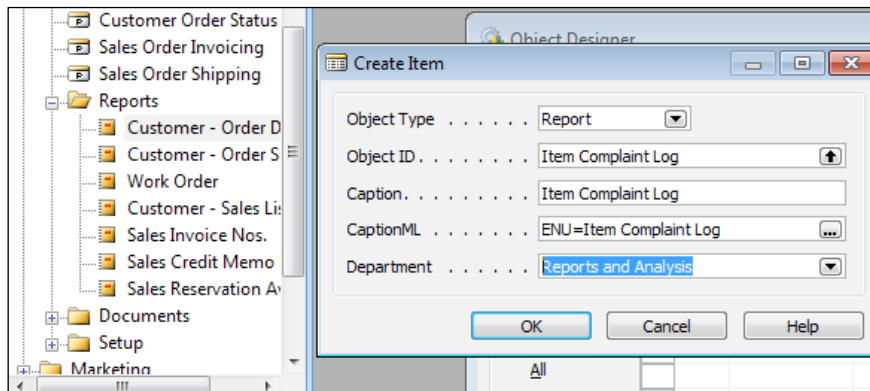
|             |                             |
|-------------|-----------------------------|
| Object Type | Page                        |
| Object ID   | Product Complaints List     |
| Caption     | Product Complaints List     |
| CaptionML   | ENU=Product Complaints List |
| Department  | Lists                       |

Again, our setup is based on the sales order. When you click on **OK**, our new page will be added to the very end of the list.



Right-click on our **Product Complaints List** icon and select **Move Up** to move it exactly below **Sales Credit Memos**.

To add our report 50000, **Item Complaint Log**, expand the **Reports** folder and add our report using the same steps described previously.



After you click on **OK**, move it into alphabetical order and click on **File | Save** to save the MenuSuite.

Using the same methods, add page **50003, Closed Product Complaint** to the **Sales & Marketing History** subfolder.

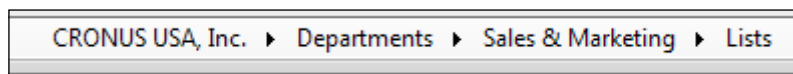
If you want a more detailed explanation on creating and modifying MenuSuites, look no further than Microsoft:

[http://msdn.microsoft.com/en-us/library/dd301366\(v=nav.70\).aspx](http://msdn.microsoft.com/en-us/library/dd301366(v=nav.70).aspx)

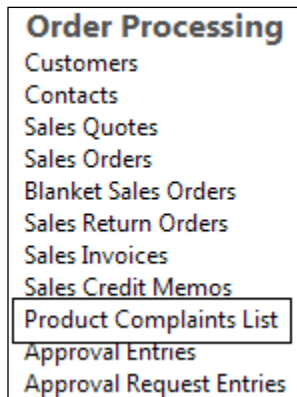
## Testing our application

Are we done? Not quite. Any application of high quality requires a tremendous amount of testing. Our application will be no different.

Open up Dynamics NAV 2013. Go to the following menu:



Under **Order Processing**, we should find our **Product Complaints List**.



Click on the link to bring up the list. Click on **New** to bring up a blank **Product Complaint** card.

The screenshot shows a web application window titled "New - Product Complaint". The interface includes a navigation bar with "Home", "Actions", and "Navigate" tabs. Below this is a ribbon with "View", "New", "Delete", "Manage", "OneNote", "Notes", "Links", and "Show Attached" options. The main content area is divided into three sections:

- General:** Contains input fields for "No.", "Customer No.", "Name", "Address", "Address 2", "City", "Post Code", "County", "Contact Name", "Vendor No.", "Vendor Name", "Document Date", and a "Resolved" checkbox.
- Product Complaint Subform:** A table with columns: "Item No.", "Description", "Quantity Accep...", "Quantity Reject...", "Source Type", and "Source". The "Quantity Accep..." and "Quantity Reject..." columns show a value of "0".
- Customer Credit Information:** Contains fields for "Contact", "Phone No.", "Collection Method", "Blocked", "Credit Limit (LCY)", "Balance (LCY)", "Difference" (0.00), "Aging as of 01/23/14 (showing days overdue)" (0.00), "31-60 Days" (0.00), "Over 60 Days" (0.00), "Payment Terms Code", "Payment Method Code", "Latest Payment Date", and "Latest Payment Amount" (0.00).

Press *Enter* to have the system generate a new number based on our COMP number series. Enter 10000 in the **Customer No.** field and we should see the customer information pop up.

Go down to the lines area and enter 70000 in the **Item No.** field. The item description should pop up. Use it to fill in the **Quantity Accepted** and **Quantity Rejected** fields and notice that the decimal places are now hidden. Add another line for item 70001.

New - Product Complaint

Home Actions Navigate CRONUS USA, Inc.

View Edit New Delete OneNote Notes Links Manage Show Attached

**Product Complaint**

**General**

No.: COMP-10002 Post Code: 31772  
 Customer No.: 10000 County: GA  
 Name: The Cannon Group PLC Contact Name: Mr. Andy Teal  
 Address: 192 Market Square Vendor No.:  
 Address 2: Vendor Name:  
 City: Atlanta Document Date:  
 Resolved:

**Customer Credit Information**

Contact: Mr. Andy Teal  
 Phone No.:  
 Collection Method:  
 Blocked:  
 Credit Limit (LCY): 0.00  
 Balance (LCY): 259,054.90  
 Difference: -259,054.90  
 259,437.76  
 Aging as of 01/23/14 (showing days overdue)  
 31-60 Days: Over 60 Days  
 -382.86  
 0.00  
 0.00  
 Payment Terms Code: 1M(8D)  
 Payment Method Code:  
 Latest Payment Date: 1/12/2014  
 Latest Payment Amount: 104,339.38

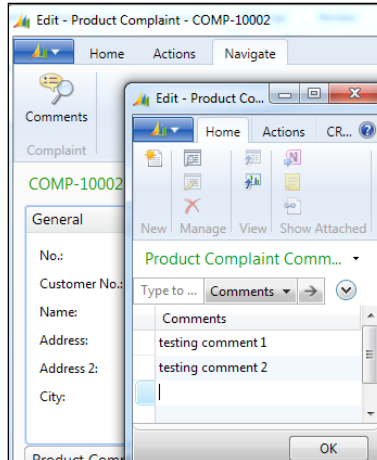
**Product Complaint Subform**

Find Filter Clear Filter

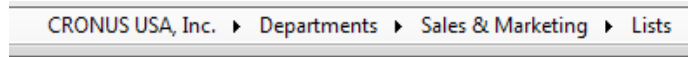
| Item No. | Description | Quantity Accep... | Quantity Reject... | Source Type    | Source |
|----------|-------------|-------------------|--------------------|----------------|--------|
| 70000    | Side Panel  | 1                 | 2                  | Purchase Re... |        |
| 70001    | Base        | 50                | 10                 | Purchase Re... |        |

OK

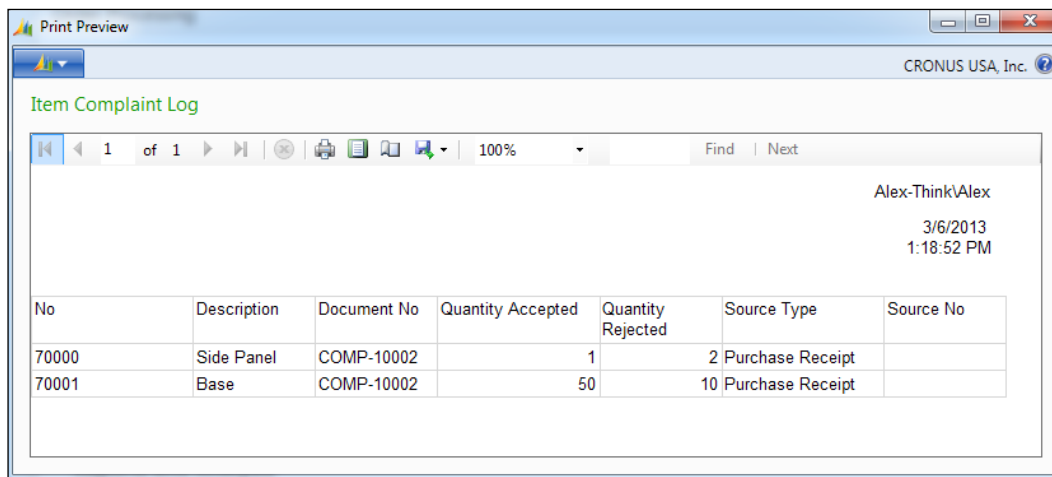
Click on **Navigate** in the ribbons and enter some comments to see if they can be entered properly.



Now it's time to test our report. Close the **Product Complaint** screen and go back to the menu.



Under the **Report and Analysis** heading, click on **Item Complaint Log**. Click on **Preview** (or **Print**) to see what we have.



Awesome!

## Last check of our requirement list

Let's check off what we've done in our requirement list:

|    | A    | B   |
|----|------|---|
| 1  | Done | Requirement   |
| 2  | x    | Every incident needs a unique number so it can be tracked   |
| 3  | x    | The customer needs to be associated with every incident   |
| 4  | x    | The vendor that supplied the good, if applicable, needs to be tracked   |
| 5  | x    | The date of the interaction must be tracked   |
| 6  | x    | The item number in question   |
| 7  | x    | The quantity of the item that was accepted and rejected   |
| 8  | x    | The source of the production order or purchase order  |
| 9  | x    | Detailed comments that is associated with every QMS entry   |
| 10 | x    | There can be 1 or more items per incident   |
| 11 | x    | Any QMS document that has been resolved will need to be marked as so  |
| 12 | x    | The open QMS documents and the closed ones should be displayed as a separate screen   |
| 13 | x    | A report needs to be created on the feedback per item   |
| 14 |      |   |
| 15 | x    | The <b>No.</b> field should automatically be generated based on a number series; similar to the sales order                                   |
| 16 | x    | The name, address, etc. should automatically be populated when the <b>Customer No.</b> is entered   |
| 17 | x    | The <b>Vendor Name</b> should automatically be populated when the <b>Vendor No.</b> is entered  |
| 18 | x    | The <b>Description</b> field on the lines should automatically be populated with the item description if the <b>Item No.</b> field is entered |
| 19 | x    | There's no way on this page to access the report that we created in the previous chapter  |
| 20 | x    | The <b>Quantity Accepted</b> and <b>Quantity Rejected</b> should display no decimal places if no decimals are entered                         |

Do you see anything else that needs to be done? Neither do I.

## Summary

In this chapter, we've extended our functionality from its shell. We've added coding, changed properties, and finally, added the project so the user can access it from the Windows Client menu. We've also ensured that everything on the requirements list is completed.

One thing to do as a developer for any program is testing. After testing, you will no doubt want to add additional features and functionalities so the user can have a rich experience.

Part of extending the user experience is understanding what the standard Dynamics NAV system can do. In the next chapter, we will explore the advanced functionalities in Dynamics NAV and what they're meant for.





# 9

## Dynamics NAV Modules to Address the Specific Needs of Your Business

*"Creativity is not the finding of a thing, but the making something out of it after it is found."*

*– James Russell Lowell*

So far in this book, we've created an application to help us keep track of the complaints from the customers. Although we took a few chapters to create this application, in reality, it will take an experienced Dynamics NAV developer a couple of hours to create this. This is not to say that we shouldn't try to create this application. Even the most experienced Dynamics NAV developer started out as a newbie.

We know that Dynamics NAV is a comprehensive ERP system. There is a lot more functionality out of the box than the basic order processing and accounting functions. As we mentioned throughout this book, the key to unlocking the full potential of Dynamics NAV is to get a basic understanding of the functions and what they're capable of.

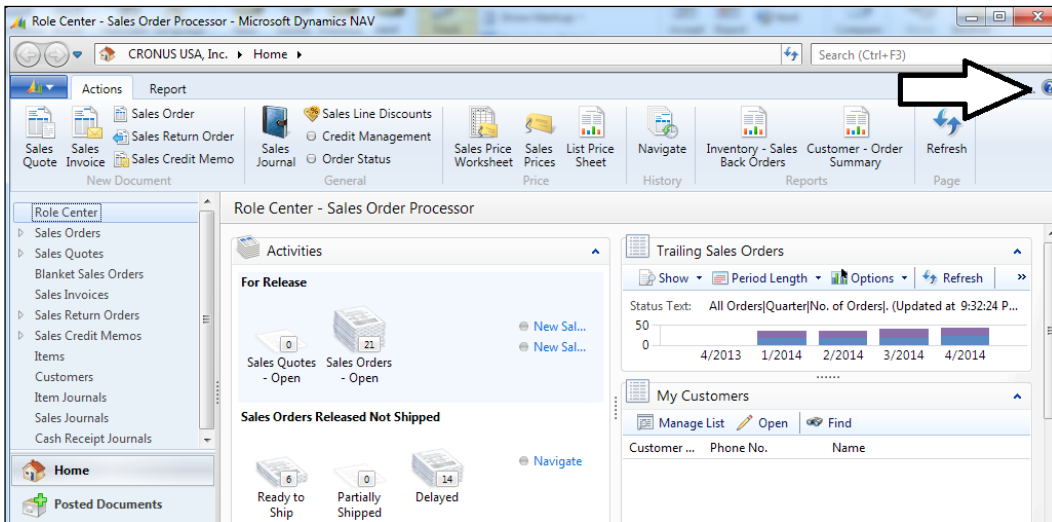
Using this book, we've only scratched the surface on its technical abilities. It's a good starting point for us to further dive in and learn what the software can do for us. Not only is the technical knowledge important, gaining a solid understanding of the out of the box functional abilities is equally, if not more important.

Any average programmer can create new applications within the system. Only a true expert would know how to make minimal changes in the system to gain maximum result and impact for their customers.

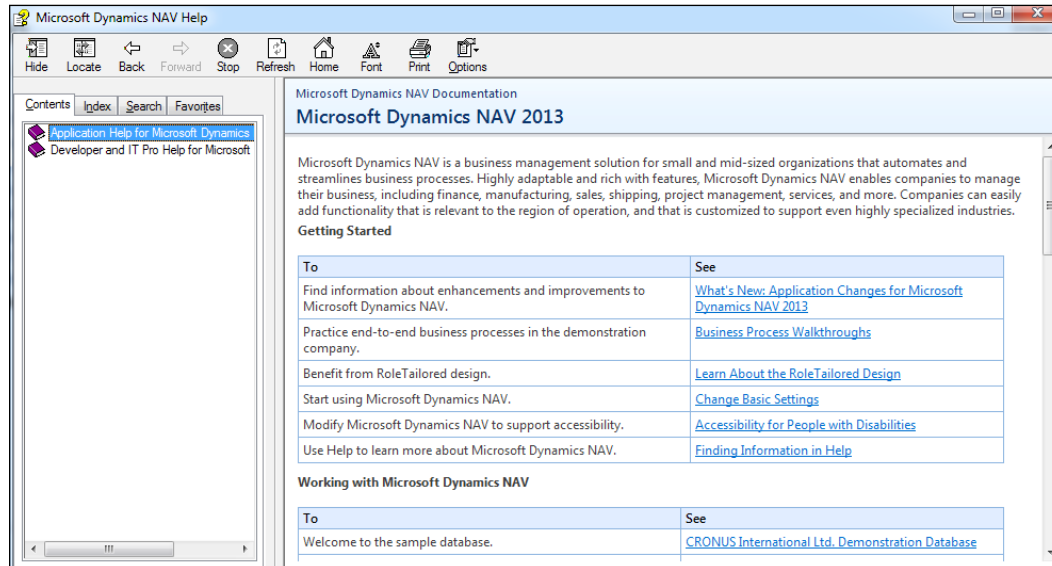
In this chapter, we will briefly talk about these granules and what they're typically used for. From there, it's our job as good developers to learn more about the detailed functions.

## Exploring the Help tool

In addition to the help provided by the Microsoft MSDN site, there's also a Help tool that's installed along with your Dynamics NAV client. To access the Help tool, click on the question mark icon in the upper-right corner of the Dynamics NAV 2013 client.



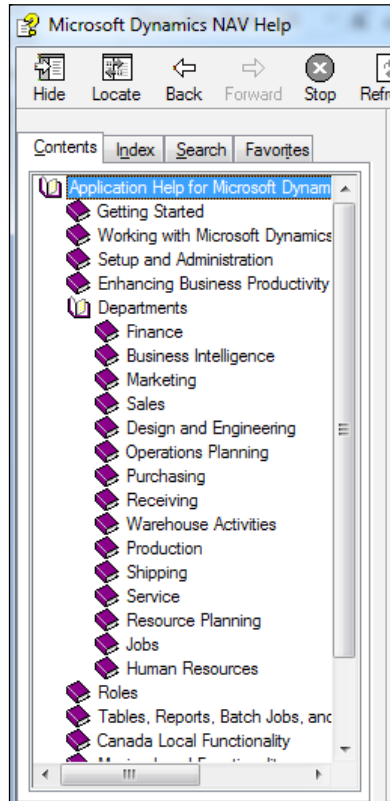
This will display the built-in Help window that comes with your Dynamics NAV 2013 installation, as shown in the following screenshot:



The difference between this Help and the online MSDN Help is that the application Help includes Help for the specific functions within the application. In other words, there is a detailed explanation of what each module (or departments, as Dynamics NAV calls them) does. The built-in Help does a better job of explaining the functionalities than the online site, which is mainly for technical resources.

For technical resources, it's really based on preference. Both do a good job of explaining technical topics. It would make sense to use the online site because contents may be added, updated, or changed.

Click on **Application Help for Microsoft Dynamics NAV 2013** and then click on **Departments**; you'll get a detailed explanation of what each department or module does.

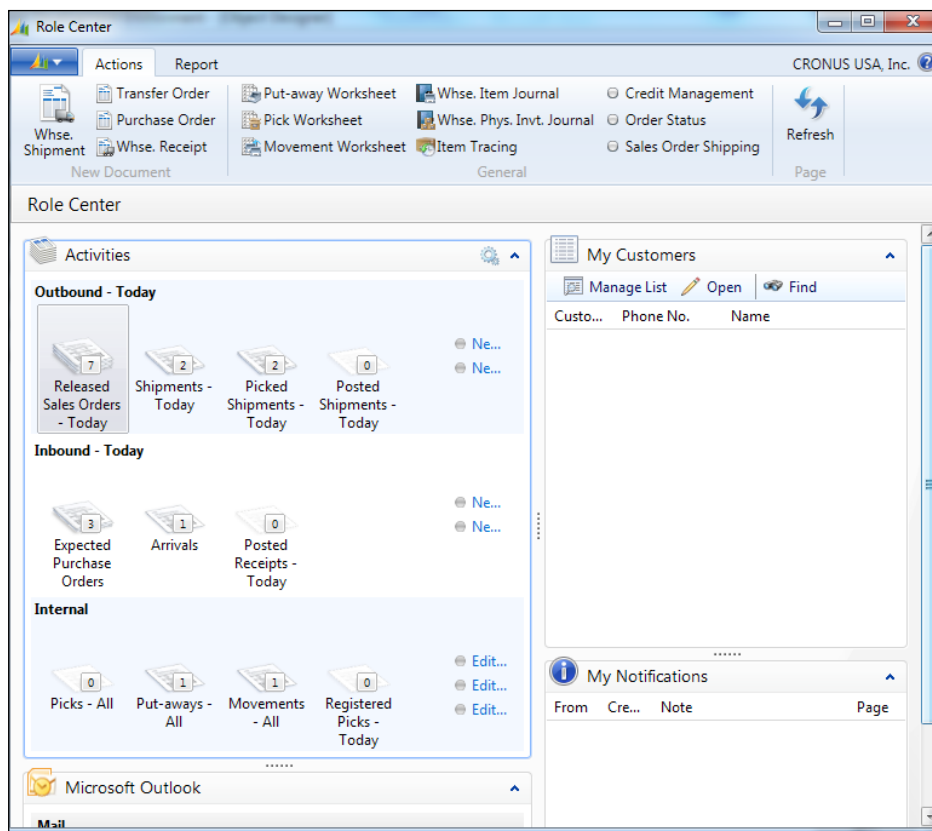


For example, if you click on **Marketing**, you'll get instructions on how to configure and process marketing activities. If you click on **Sales**, you'll get instructions on how to configure and process sales orders.

## Exploring the Warehouse Management functionality

Everything you want to know about the Warehouse Management functionality can be found in the Dynamics NAV Help file under **Shipping, Receiving, and Warehouse Activities**.

If we have a warehouse where it takes too long for people to pick and put-away items, or mistakes frequently occur during the picking and put-away, we should use Warehouse Management to help us manage this process.



In a nutshell, the following are some of the benefits of Warehouse Management:

- Separate out the shipping and receiving process to the warehouse staff to reduce delayed paperwork going back to the office
- Enable bin tracking for the items in the warehouse so people know exactly where the items are in the warehouse
- Allow the warehouse employees to pick and put-away using a scanner so inventory is moved in real time without paper floating around

A couple of key points to keep in mind regarding Warehouse Management are as follows:

- **Warehouse shipment:** This allows us to group one or more sales orders into one shipment. One warehouse shipment can constitute one container or the total number of items loaded into one truck. This tool is very useful if you want to print bill of lading from Dynamics NAV without using external paper documents. A bill of lading is basically a document that tells the truck or ship what items are being shipped.

This function is designed to allow the warehouse staff to process shipments within Dynamics NAV. Without using this function, it's most likely that the accounting department will post the shipment from the sales order after receiving the paperwork for goods that have already left the building.

- **Warehouse receipt:** Similar to the warehouse shipment that groups sales to customers and returns to vendors going out, warehouse receipt works in the exact opposite way. It groups purchases and sales returns into one receipt document.

This function is especially useful when you're receiving products from overseas vendors. When goods are shipped to you from the vendors, for many reasons the vendor will ship you the goods per item instead of per order. So, if you have one item that's on five different purchase orders, the overseas vendor will ship to fulfill the quantity requirements for all the five orders together. This is a nightmare if you're trying to receive per purchase order.

Using the warehouse receipt, you will be able to generate the receipt by item that exactly matches your vendor's bill of lading and commercial invoice.

- **Warehouse activities:** This allows you to control the warehouse picking and put-away process from the bins to the shipping/receiving docks and between bins. The pick and put-away is typically used with warehouse shipment and warehouse receipt because it allows you to create pick and put-away per received container and shipment.

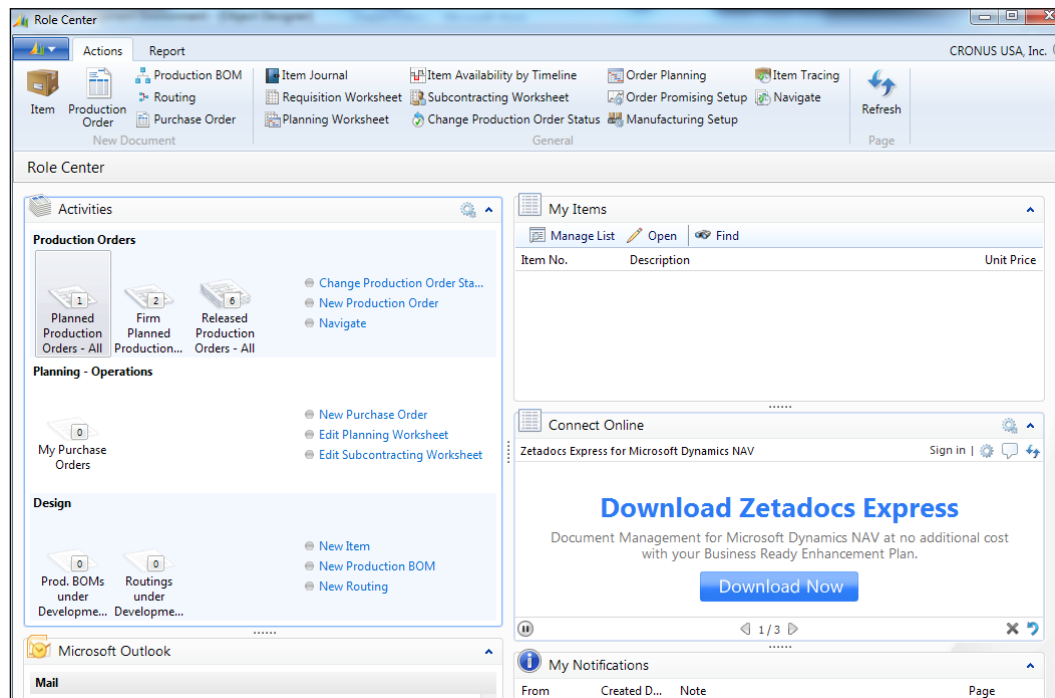
Enabling this also allows you to manage the quantities in your bins so items can be moved from the storage area in the higher levels to your pick bins on the first level or the designated pick area.

- **Handheld barcodes (ADCS):** This is the wireless, real-time bin content barcode functionality. With this, the warehouse will be able to pick, put-away, and move items in real time so the inventory data does not get delayed. In real time, the order entry staff will be able to tell the customer what the status of the order is and how many items have been picked.

## Exploring the Manufacturing functionality

Everything you want to know about the Manufacturing functionality can be found in the Dynamics NAV Help file by navigating to **Design and Engineering | Operation Planning | Production**.

If we buy components and/or raw materials, and then fabricate and assemble them into finished goods, we should use Manufacturing to help us manage this process.





In a nutshell, the following are some of the benefits of using Manufacturing in Dynamics NAV:

- It allows you to create complex bills of materials and routings per item, and it also allows different versions of it.
- It controls the materials and machines that are used per production order. It also calculates the proper costing of the items used and produced. After all, this is one of the main reasons why ERP software was invented.
- It allows you to plan the inventory demands (and surpluses) of the company. The **Material Requirements Planning (MRP)** calculation can be location specific or for the company as a whole.

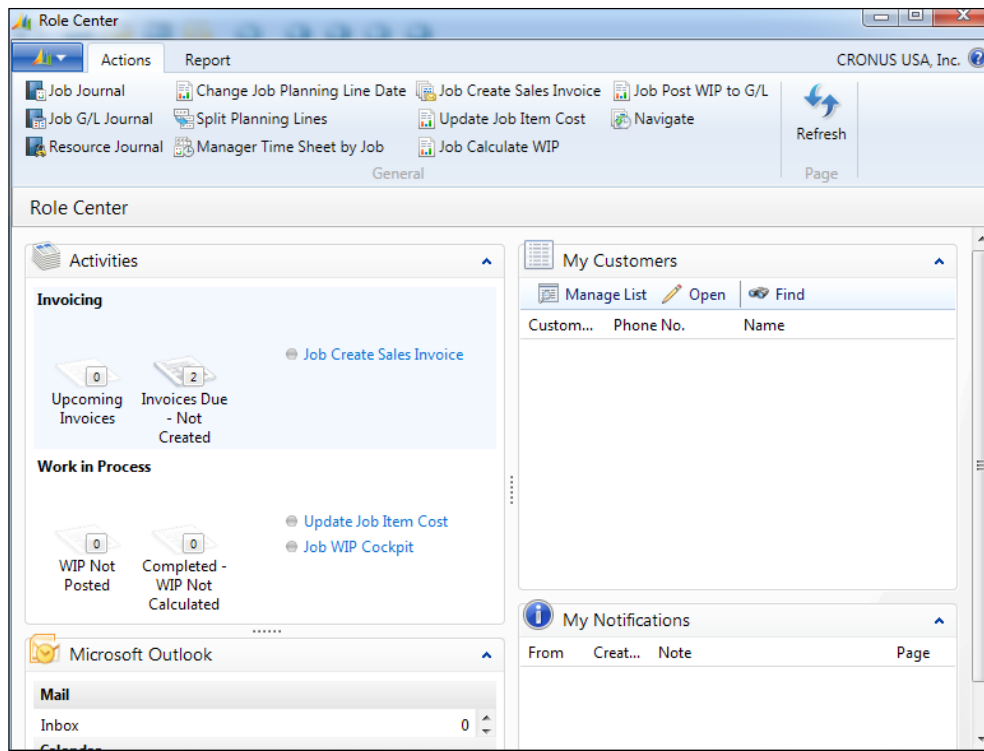
A couple of key points to keep in mind regarding Manufacturing are as follows:

- **Production bills of material and routing:** You can assign a production **bill of materials (BOM)** and routing to an item. Within the production BOM and the routing, you can have different versions and variations. This allows the item designer to create an alternate version of the BOM to do some testing without actually "turning it on".
- **Production orders:** Allows for the production department or the shop floor to manufacture the items based on the production orders created. The production order functionality also has an option for you to run a "what if" analysis based on production orders created.
- **Planning worksheet:** This is the heart of the MRP calculation. Based on the setup on item, location, and items on location, this calculation will give you accurate requirements and what date the item should arrive/produce to satisfy the customer requirement or inventory levels. But as the old saying goes, "Garbage in, garbage out"; we have to make certain we understand all of the moving parts before we turn it on. It may be a good idea to get the help of an experienced Dynamics NAV Manufacturing professional before attempting this.

## Exploring the Jobs functionality

Everything you want to know about the Jobs functionality can be found in the Dynamics NAV Help file by navigating to **Jobs | Resource Planning**.

This module is typically used for companies that deal with long and drawn-out projects. The project can be long government contracts, a plan to deliver a series of products over a period of time, or even a construction company. Whether we're building a house or building the New York subway system, we would set up a job, or a series of jobs, to keep track of it.



In a nutshell, here are some of the benefits of using Jobs in Dynamics NAV:

- Keep track of sales and cost of a long project for all activities
- Allows you to create project plans and look at estimated cost versus the actual cost
- Accrue WIP and recognize the revenue and cost to the General Ledger accordingly

A couple of key points to keep in mind regarding Jobs are as follows:

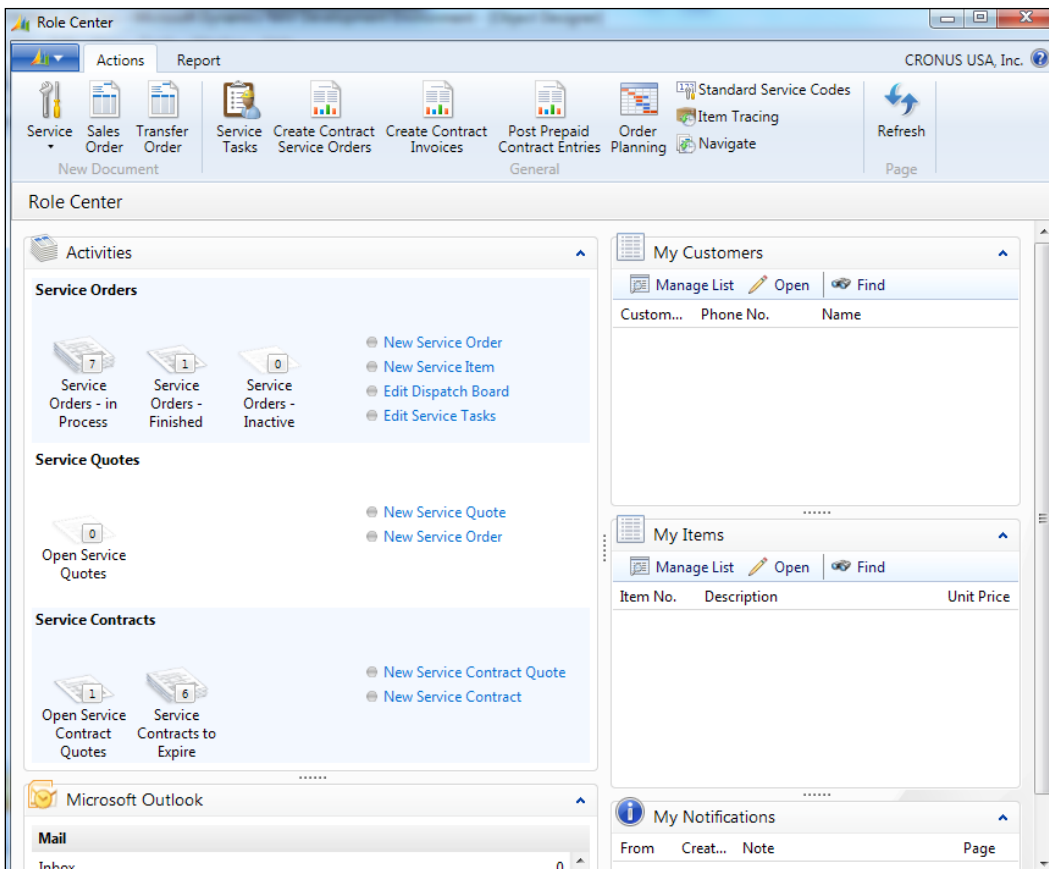
- **Jobs:** The job is the place where all of the activities against a long project get recorded. From the job, you can assign the estimated cost and price of activities against the actual cost and price of the activities.
- **Time sheets:** When the time is spent on a particular job, this screen allows people to easily enter their time spent against a job.

Please note that consulting companies such as your Dynamics NAV partner use the Jobs module to keep track of the cost and sales of their implementation projects.

# Exploring the Service Management functionality

Everything you want to know about the Service Management functionality can be found in the Dynamics NAV Help file under **Service**.

This module is typically used when the company sells items that need to be serviced periodically. For example, if we manufacture and sell air conditioners, we will need to keep track of the warranty and any service calls to the clients' homes where the air conditioner is installed. Service Management should be used to help us manage this.



A common use for this is warranty registration cards. When you buy a stove oven, there's usually a little piece of card that says "Register Your Oven!" When you send the card back, the company receiving your registration card would enter your information in Service Management.

In a nutshell, the following are some of the benefits of using Service Management in Dynamics NAV:

- Keep track of items sold to the customer that need to be serviced periodically.
- Keep track of the warranties for the items. We will be able to identify whether the repair service call is under warranty or not based on what items are installed on the customer's site.
- The service calls can be recorded against the items sold to the customer. Instead of a regular sales order, service orders are created to identify the original item that was sold to the customer and what resource or parts are used to repair or service the item in question.

A couple of key points to keep in mind regarding Service Management are as follows:

- **Service item:** You can specify what specific items are sold or installed on the customer's or end user's site and keep track of warranty for labor and parts. We can also view the service history for this item that's installed at the client site to see if it makes more sense to replace the item than to spend more money repairing it.
- **Service contracts:** If our company sells service contracts, we can enter that information into the service contracts in Dynamics NAV. It allows us to bill the contract on a time interval, and we can specify what items are specifically under the service contract.
- **Service order:** When a customer calls to report a problem with the item they purchased and service is required, we use the service order to track that. This is different from the sales order because the item is already sold to the customer; we just need to either service or repair the item that the customer is calling about.
- **Dispatch board:** Service calls can have a variety of urgency. The dispatch board allows the central office to determine what needs to be resolved first based on the calls coming in.

## Summary

We've gone through all of the key departments (or modules) that will help the specific needs of your business. The brief description described in this chapter does not do justice to Dynamics NAV regarding the additional functionalities that are out of the box.

If any of the functionality described fits what your business does, then explore it! Read up using the Help tool and walkthrough the examples provided. The Help tool gives you a lot of detail (sometimes too much detail) on what the out of the box function does.

Once you understand the basics or the overview of the module, you can take the skills you've learned throughout this book and find out what tables, pages, and reports are associated. You will then have a higher level of discussion with your Dynamics NAV consultant on your business requirements and understand if the recommendations from your Dynamics NAV partner make sense for your company.

In the next chapter, we will find additional resources to help you on your journey to learn about Dynamics NAV and all of its capabilities.

# Additional Resources and Conclusion

*"It's all about quality of life and finding a happy balance between work and friends and family."*

*- Philip Green*

We have come a long way from first downloading our free copy of Dynamics NAV, from defining a task list, to actually creating and testing our application.

We've gained knowledge on how to access the application from the frontend through the Windows Client and tied it to the backend through the Object Designer in the Development Environment.

Dynamics NAV can be a vast and complex system, or it can be easy and simple. It really depends on how you design it and how you implement it.

The contents we've gone through only provided a glimpse of Dynamics NAV. In this section, we will explore some additional resources so you can take control of what you have learned.

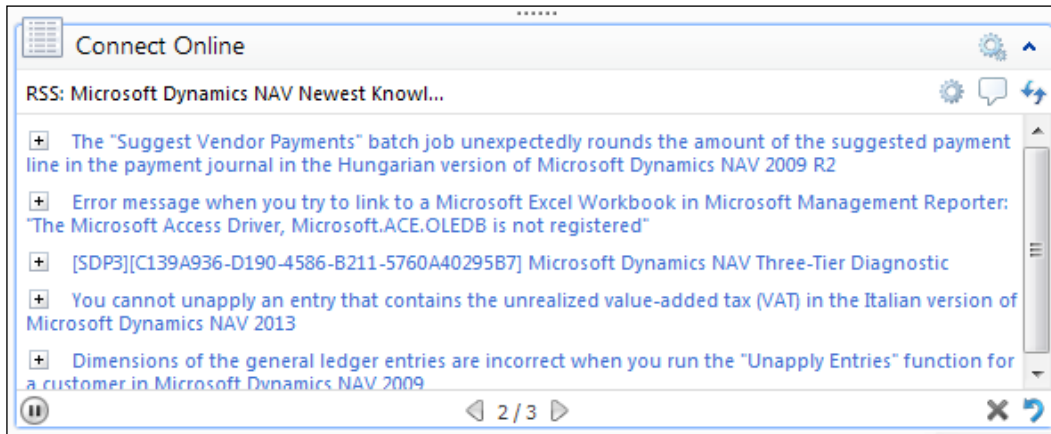
## **Official online resource**

There are a couple of very useful resources available online to further expand our knowledge in Dynamics NAV.

## Connect online

From when you start the **Windows Client (WC)**, you'll be able to select **Connect Online**. You can also add **Connect Online** using **Customize This Page** when you're at the WC home page.

This feature allows you to go into the Dynamics NAV community and see the released knowledge base articles and user forum questions and answers.



When you click on the links provided, it will bring you to the appropriate sites regarding the topic.

## CustomerSource

When you buy your copy of Dynamics NAV, you'll notice that there's a 16 percent Microsoft Annual Enhancement fee that's calculated based on the price of your software. The enhancement plan will give you a lot of benefits, and one of those benefits is access to CustomerSource.

The step-by-step instruction on how to access CustomerSource is present here:

<http://www.microsoft.com/dynamics/en/gulf/customersource.aspx>

This is where you can access and get the latest products released by Microsoft. You also have access to use the official manuals that solution centers use to get their certifications. CustomerSource is also a great place to learn about any discounts and promotions for the product.

## MSDN site

As mentioned throughout this book, almost every technical function in Dynamics NAV has been documented from the Dynamics NAV MSDN site: [http://msdn.microsoft.com/en-us/library/hh173988\(v=nav.70\).aspx](http://msdn.microsoft.com/en-us/library/hh173988(v=nav.70).aspx).

Instead of typing in the URL, as written in this book, the easier way to access this MSDN site is to go to your favorite web browser and type in the keyword `Dynamics NAV 2013 MSDN`.

## Microsoft Dynamics Community

This is the official Dynamics community created by Microsoft. The link for it is as follows:

<https://community.dynamics.com/>

Using this link, you can access the communities for different Dynamics product lines. There are blogs and forums that you can use to ask and answer questions. The staff at Microsoft frequently visit the forums on the community site. If you have questions about specific areas in NAV, you can visit the forums and ask for opinions.

## Unofficial online resource

In addition to the official Microsoft sites for the product, there are also sites created by independent professionals and Microsoft development staff.

## Online forums

Before Microsoft came out with the official community site, there were already dedicated fans of NAV that created their own forum. At the time of writing, the most popular non-official Dynamics NAV forums are:

- [www.mibuso.com](http://www.mibuso.com): This is one of my favorite Dynamics NAV forums. The primary reason I like this forum is because I like phpBB style forums.
- [dynamicsuser.net](http://dynamicsuser.net): This is another forum that services the Dynamics NAV community. This forum has also expanded to service other Dynamics NAV products.

Both of these forums have dedicated experts to help out with user questions. A lot of the users frequent both sites, so you can get your questions answered on both sites.



## Blogs

There are a lot of blogs out there, but most are not that useful to further our education in Dynamics NAV. Some blogs that are the my personal favorites are:

- [blogs.msdn.com/b/nav](http://blogs.msdn.com/b/nav): This blog is maintained by the Microsoft development team for Dynamics NAV. You will be able to find detailed information for every undocumented feature in the core product (real features, not bugs) here. If you're serious about Dynamics NAV, this site should be bookmarked or subscribed to your favorite RSS reader. The information posted here, even though it's from Microsoft employees, is not officially supported by Microsoft, the company.
- [www.waldo.be](http://www.waldo.be): This blog is written by Eric Wauters, a Dynamics NAV MVP. His blog has been one of the community favorites for his updates on the latest and greatest from Dynamics NAV.
- [navigateintosuccess.com](http://navigateintosuccess.com): This blog is maintained and written by a former Microsoft employee, Vjekoslav Babić. He has been a valuable contributor writing about new technologies that deal with NAV.
- [www.dynamicsnavconsultant.com](http://www.dynamicsnavconsultant.com): And last but not least, this blog is written and maintained by Alex Chow, the author of this book. This blog documents Alex's journey in Dynamics NAV, talking about more functional and business-related topics pertaining to Dynamics NAV instead of technical topics.

## Dynamics NAV add-ons

Add-ons are one of the most important aspects of Dynamics NAV. A lot of the requirements for every industry are developed by third parties that service these particular industries.

When you search for "Dynamics NAV add-on" on your browser, you'll come across many sites that look like the official add-on site. *Don't be fooled!* These sites are nothing more than companies masking the domain to trick you into thinking it's a Microsoft site to sell you their own add-ons. Most of the add-ons listed are overpriced and unnecessary. So buyers beware!

When determining the add-ons, it's best to speak with your Dynamics NAV partner, because they should have your interest at heart for the implementation.

If you want to browse the add-ons available for Dynamics NAV, the official site is called Microsoft Pinpoint: <http://dynamics-erp.pinpoint.microsoft.com/>.

## Dynamics NAV solution center finder

It's universally understood that sometimes, despite the hours of searching for the right partner, the relationship just does not work out. One of the main advantages of using Microsoft Dynamics is that you can freely get up and working with any Dynamics solution center you wish.

You are not married (or a prisoner) to the original partner that you bought the product from. There are two ways you can go about finding the right partner. They are as follows:

- Contact your local Microsoft office for a list of local partners
- Visit the Microsoft Pinpoint site and do your own research:  
<http://dynamics.pinpoint.microsoft.com/>

## Summary

So why are we doing this? Why are we spending time and money on a new software? Why are we spending our time to learn this product when we can get someone else to do it? Why should we be involved in this process?

The quote at the beginning of the book is worth bringing up again.

*"It's all about quality of life and finding a happy balance between work and friends and family."*

We're doing this to simplify our life, and in order to do this, we have to also help simplify the life of others! Without understanding what the ultimate goal is, how will we know if the end is all worth it? Part of understanding is to invest in the hard work at the very beginning so it can pay dividends for us later on.

Work in itself is rewarding, however, too much work will do the exact opposite. Because of the competition, we do have to work harder. But along those lines, we also have to work smarter.

It's not healthy to be working all day just to keep up with the competition or satisfy the demands of management. To perform our best, we need to divide our attention to what's important in our life.

Having a balanced life, isn't this why we're doing what we do in the first place?



# Index

## A

**About This Page screen** 85  
**ADCS** 14  
**add-ons** 206  
**Administration Tools** 13  
**analysis report**  
  creating, wizards used 143-151  
**Apple** 27  
**application**  
  adding, to RTC menu 182-185  
  number series, creating for 157-160  
  testing 154-156, 185-188  
**Application Builder** 94  
**Automated Data Capture System.** *See* **ADCS**  
**AutoSplitKey property** 135-137

## B

**BigInteger data type** 93  
**bill of materials (BOM)** 48, 198  
**Binary data type** 93  
**BLOB** 92  
**blogs** 206  
**Boolean data type** 93  
**business software**  
  advantages 73, 74

## C

**C/AL** 17, 94, 103  
**card page**  
  exploring 45, 46  
  personalizing 47  
**Classic Client** 27  
**ClickOnce Installer Tools** 14

**Client/server Application Language.**  
  *See* **C/AL**  
**Client/Server Integrated Development Environment.** *See* **C/SIDE**  
**closed complaints**  
  separate screen, creating for 174-176  
**Cloud license** 22  
**cmdlet** 16  
**code**  
  inserting, in tables 165-167  
  used, for defaulting fields 167-170  
**Code data type** 93  
**Codeunits** 51  
**coding**  
  accessing, in Dynamics NAV 95  
**company**  
  current operation 76  
  drawbacks, identifying 76  
  non-negotiable must-haves, defining 78  
  problems, listing 77  
  solution, designing 79  
**Complaint Comments table**  
  adding 124, 125  
**Complaint Header table** 135, 180  
**Complaint Line table**  
  about 135  
  creating 115, 116  
**components, Dynamics NAV**  
  about 12  
  ADCS 14  
  Administration Tools 13  
  ClickOnce Installer Tools 14  
  documentation 14  
  Microsoft Office Outlook Add-in 14  
  Portal Framework for SharePoint 14  
  Role Tailored Client 13

- server 14
- SQL Server Database Components 14
- Web Server Components 14
- composite primary key**
  - creating 122, 123
- conditional table relationship**
  - creating 118-121
- Connect Online 204**
- Cronus 74-76**
- C/SIDE 17**
- CustomerSource 204**

## D

- data**
  - cleaning up 180, 181
  - searching, filters used 42, 44
- data types, Dynamics NAV**
  - BigIntegers 93
  - Binary 93
  - BLOB 92
  - Boolean 93
  - Code 93
  - Date 93
  - DateFormula 93
  - DateTime 93
  - Decimal 93
  - Duration 93
  - GUID 93
  - Integer 93
  - Option 93
  - RecordID 93
  - TableFilter 93
  - Text 93
  - Time 93
- Date data type 93**
- DateFormula data type 93**
- DateTime data type 93**
- Decimal data type 93**
- decimal values**
  - properties, modifying 173, 174
- DelayedInsert property 137**
- DELETEALL function 181**
- demo license 20**
- departments**
  - exploring 54
- detailed transactions 59, 61**

- development-environment interface 49**
- dispatch board 201**
- documentation 14**
- document page**
  - exploring 47, 48
- Duration data type 93**
- Dynamics NAV**
  - coding, accessing 95
  - components 12
  - data types 92, 93
  - free copy, obtaining 8, 9
  - history 26-28
  - installation files 12
  - official online resource 203-205
  - software, installing 10, 11
  - unofficial online resource 205, 206
- Dynamics NAV 5.0**
  - main menu 27
- Dynamics NAV 2009 26**
- Dynamics NAV add-ons 206**
- Dynamics NAV solution center finder 207**

## E

- Electronic Data Interchange (EDI) 73**
- ERP system 83**

## F

- FactBoxes**
  - about 36, 40, 45
  - linking 132, 133
- FastTabs 46**
- fields**
  - adding, to tables 111, 112
  - defaulting, code used 167-170
  - defaulting, FlowFields used 171, 172
  - table relations, defining in 112, 113
- filters**
  - used, for searching data 42, 44
- FlowFields**
  - used, for defaulting fields 171, 172
- full On-Premise license 21**
- functions**
  - grouping 141

## G

**GET function** 170  
**GUID data type** 93

## H

**handheld barcodes (ADCS)** 197  
**Help tool**  
    accessing 192  
    exploring 192-194  
    versus online MSDN Help 193

## I

**indexes** 94  
**installation files, for Dynamics NAV** 12  
**installation, Visual Studio Web Developer**  
    2010 Express 12  
**Integer data type** 93  
**inventory** 57  
**item description**  
    defaulting, on line table 173

## J

**jobs** 199  
**Jobs functionality**  
    benefits 199  
    exploring 198, 199

## L

**license**  
    Cloud license 22  
    demo license 20  
    full On-Premise license 21  
    MSDN license 20, 21  
    obtaining 19  
**line table**  
    item description, defaulting on 173  
**list page**  
    creating 176-180  
    exploring 35- 37  
    personalizing 38-41

## M

**Manufacturing functionality**  
    benefits 198  
    exploring 197, 198

    production bills of material and routing  
        198  
    production orders 198  
    worksheet, planning 198  
**Material Requirements Planning (MRP)** 198  
**MenuSuite** 52  
**Microsoft Dynamics Community** 205  
**Microsoft Dynamics NAV 2013**  
    **Administration Shell** 16  
**Microsoft Dynamics NAV 2013**  
    **Development Environment** 17  
**Microsoft Dynamics NAV 2013**  
    **Windows Client** 18  
**Microsoft Dynamics NAV**  
    **Administration** 18  
**Microsoft Office Outlook Add-in** 14  
**Microsoft Pinpoint** 206  
**modules** 54  
**MSDN license** 20, 21  
**MSDN site** 205

## N

**Navision Financials Version 2.0**  
    main menu 26  
**number series**  
    creating, for application 157-160  
    tables, programming for 160-164

## O

**Object Designer** 50  
**Object Designer screen** 86  
**objects** 81  
**objects, Dynamics NAV**  
    codeunits 51  
    MenuSuite 52  
    pages 50  
    query 51  
    reports 50, 51  
    tables 50  
    XMLport 51  
**official online resource, Dynamics NAV**  
    about 203  
    Connect online 204  
    CustomerSource 204  
    Microsoft Dynamics Community 205  
    MSDN site 205

**OnDelete trigger** 162  
**OnInsert trigger** 162  
**online forums** 205  
**online MSDN Help**  
    versus Help tool 193  
**OnValidate section** 95  
**Option data type** 93

## **P**

**Page Designer screen**  
    about 87  
    properties, for controls 101  
**PagePartID property** 88  
**pages**  
    about 50  
    linking 138-143  
**Portal Framework for SharePoint** 14  
**Posted Purchase Invoice screen** 61  
**primary key**  
    about 94, 106, 110  
    identifying 106  
**process-only report** 50  
**Product Complaint Comments page**  
    creating 137  
**Product Complaint page**  
    creating 128-131  
**Product Complaint subpage**  
    creating 134, 135  
**properties**  
    modifying, of decimal value 173, 174  
**purchase order** 71

## **Q**

**Quality Management System (QMS)**  
    about 78  
    attributes 79  
    functionalities 79  
**query** 51

## **R**

**RDLC reporting method** 8  
**RecordID field** 93  
**regular report** 50  
**Replenishment tab** 58  
**reports** 50, 51

**Return on Investment (ROI)** 76  
**ribbons** 41  
**role center page**  
    exploring 30, 31  
    search feature 32  
**Role Tailored Client (RTC)** 13, 18  
**RTC environment**  
    exploring 49  
**RTC menu**  
    application, adding to 182-185

## **S**

**sales and marketing** 54-56  
**sales and purchase order screens** 83  
**Sales Header table (36)** 90-92  
**sales order**  
    creating 62-71  
    sales order function 83  
**Sales Order page (42)** 98-101  
**Sales Order Subform page (46)** 102, 103  
**search feature, role center page** 32  
**separate screen**  
    creating, for closed complaints 174-176  
**server** 14  
**service contracts** 201  
**service item** 201  
**Service Management functionality**  
    benefits 201  
    dispatch board 201  
    exploring 200, 201  
    service contracts 201  
    service item 201  
    service order 201  
**service order** 201  
**SETFILTER function** 181  
**SETRANGE function** 181  
**Setup.exe file** 11  
**Sliced Bread** 27  
**SourceTable property** 90  
**SQL Server 2012 folder** 19  
**SQL Server 2012 report builder** 19  
**SQL Server Database Components** 14

## **T**

**Table Designer**  
    about 91

- Data Type column 91
- Description column 92
- Enabled column 91
- Field Class column 92
- Field Name column 91
- Field No. column 91
- Length column 92

**TableFilter 93**

**table relations**

- about 96-98
- defining, in fields 112, 113

**tables**

- about 50, 81
- code, inserting in 165-167
- compiling 109
- creating 106, 107
- fields, adding to 111, 112
- programming, for number series 160-164
- requisites 106
- running 109
- saving 108

**tasks list**

- verifying 189

**Text field 93**

**Time field 93**

**time sheets 199**

**trigger 95**

## U

**unique document numbers**

- generating, automatically 156, 157

**unofficial online resource, Dynamics NAV**

- about 205
- blogs 206
- online forums 205

## V

**vendor 58**

**Vendor List screen 58**

**Visual Studio Web Developer 2010 Express**

- installing 12

## W

**warehouse activities 196**

**Warehouse Management functionality**

- benefits 196

- exploring 195, 196, 197

- handheld barcodes (ADCS) 197

- warehouse activities 196

- warehouse receipt 196

- warehouse shipment 196

**warehouse receipt 196**

**warehouse shipment 196**

**Web Server Components 14**

**Windows Client (WC) interface**

- about 29, 204

- role center page, exploring 30, 31

**wizards**

- used, for creating analysis report 143-151

## X

**XMLport 51**







**Thank you for buying**  
**Getting Started with Dynamics NAV 2013**  
**Application Development**

## **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).

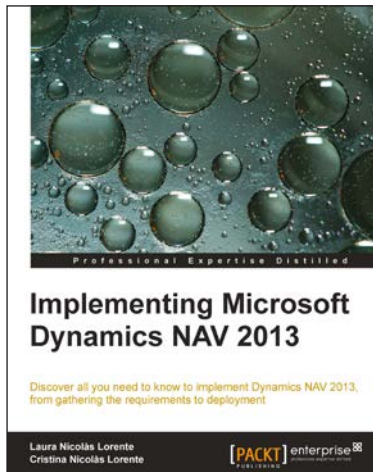
## **About Packt Enterprise**

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

## **Writing for Packt**

We welcome all inquiries from people who are interested in authoring. Book proposals should be sent to [author@packtpub.com](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.

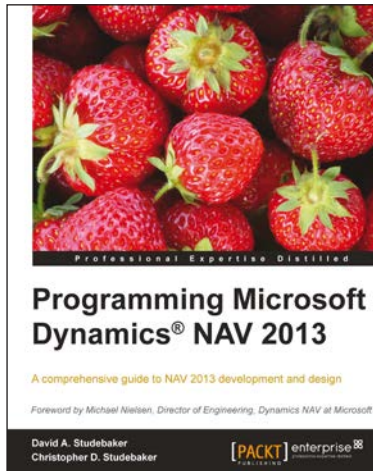


## Implementing Microsoft Dynamics NAV 2013

ISBN: 978-1-849686-02-0      Paperback: 554 pages

Discover all you need to know to implement Dynamics NAV 2013, from gathering the requirements to deployment

1. Successfully handle your first Dynamics NAV 2013 implementation.
2. Explore the new features that will help you provide more value to your customers.
3. Full of illustrations and diagrams with clear step-by-step instructions and real-world tips extracted from years of experience.



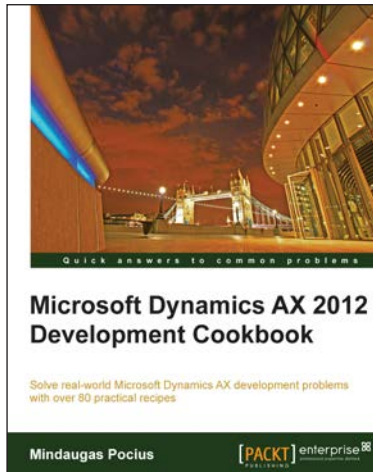
## Programming Microsoft Dynamics NAV 2013

ISBN: 978-1-849686-48-8      Paperback: 630 pages

A comprehensive guide to NAV 2013 development and design

1. A comprehensive reference for development in Microsoft Dynamics NAV 2013, with C/SIDE and C/AL.
2. Brimming with detailed documentation that is additionally supplemented by fantastic examples.
3. The perfect companion for experienced programmers, managers and consultants.

Please check [www.PacktPub.com](http://www.PacktPub.com) for information on our titles



## Microsoft Dynamics AX 2012 Development Cookbook

ISBN: 978-1-849684-64-4      Paperback: 372 pages

Solve the real-world Microsoft Dynamics AX development problems with over 80 practical recipes

1. Develop powerful, successful Dynamics AX projects with efficient X++ code with this book and eBook.
2. Proven recipes that can be reused in numerous successful Dynamics AX projects.
3. Covers general ledger, accounts payable, accounts receivable, project modules and general functionality of Dynamics AX.
4. Step-by-step instructions and useful screenshots for easy learning.



## Microsoft Dynamics AX 2012 Services

ISBN: 978-1-849687-54-6      Paperback: 196 pages

Effectively use services with Dynamics AX and create your own services

1. Learn about the Dynamics AX 2012 service architecture.
2. Explore the new features of JDBC 4.0 Create your own services using wizards or X++ code.
3. Consume existing web services and services you've created yourself.

Please check [www.PacktPub.com](http://www.PacktPub.com) for information on our titles