Professional Expertise Distilled

Microsoft Dynamics GP 2013 Reporting

Second Edition

Create valuable insights for your organization with Microsoft Dynamics GP 2013 reporting tools

David Duncan
Christopher J Liley

[PACKT] PUBLISHING enterprise
professional expertise distilled

Microsoft Dynamics GP 2013 Reporting

Second Edition

Create valuable insights for your organization with Microsoft Dynamics GP 2013 reporting tools

David Duncan

Christopher J Liley

[PACKT] enterprise
professional expertise distilled
PUBLISHING
BIRMINGHAM - MUMBAI

Microsoft Dynamics GP 2013 Reporting

Second Edition

First published: April 2011

Second edition: October 2013

Production Reference: 1161013

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

ISBN 978-1-84968-892-5

www.packtpub.com

Cover Image by Suresh Mogre (suresh.mogre.99@gmail.com)

Credits

Authors
David Duncan
Christopher J Liley

Reviewers
Vaidhyanathan Mohan
Amy Walsh
Kristi Weldon

Acquisition Editor
James Jones

Lead Technical Editor
Susmita Panda

Technical Editors
Gauri Dasgupta
Kapil Hemnani

Project Coordinator
Kranti Berde

Proofreaders
Simran Bhogal
Maria Gould
Ameesha Green

Indexer
Rekha Nair

Graphics
Ronak Dhruv

Production Coordinator
Kyle Albuquerque

Cover Work
Kyle Albuquerque

About the Author

David Duncan has been a consultant and business analyst for over five years. He has driven organizational success through well-defined reporting solutions that provide valuable and insightful information to key stakeholders. He is the co-author of *Microsoft Dynamics GP 2010 Reporting* and has also served as a technical reviewer for *Building Dashboards with Microsoft Dynamics GP 2013 and Excel 2013*, both by Packt Publishing. David, who holds several certifications for Microsoft Dynamics GP and SQL Server, has extensive experience in designing and providing business intelligence and reporting tools for organizations that use Dynamics GP and many other Microsoft SQL Server-based applications.

In his current position as a business analyst, David works with end users at all levels of his organization to provide them with reporting solutions to meet their needs. David enjoys sharing insights gained from these solutions, and he is even happier when he can share these insights using Excel PivotTable and Excel Power Pivot functionality!

David holds a degree from Clemson University. He resides in Rocky Mount, N.C. with his wife, Mary Kathleen, and their newborn daughter, Mary Eliza.

Acknowledgments

I'd like to thank my friend and co-author Chris Liley for all he has done in making this book possible. It has been a great privilege to work with him on both the original book and this update!

To Clinton Weldon and Dwight Specht, both of I.B.I.S., Inc., while we may no longer have the opportunity to work together on a daily basis, the lessons I learned and experiences I gained while working with you continue to guide me on a regular basis.

To our exceptional reviewers, Vaidhyanathan Mohan, Amy Walsh, and Kristi Weldon, thank you for your energy and enthusiasm in reviewing our book. It is most assuredly a better resource for the attention you have given to it.

To the team at Packt Publishing, thank you again for seeing Chris and me through the various stages of publishing this book. As much as I enjoy sharing my passion for reporting with others, Packt Publishing has enabled me to share this passion beyond just my immediate circle of colleagues.

And last, but certainly not least, this book would not be possible without the loving encouragement and support of my wife, Mary Kathleen. Writing a book can be a frustrating and challenging task at times, and you have always been ready with words of reassurance and inspiration!

About the Author

Chris Liley is a Principal Consultant with I.B.I.S., Inc., which is a Microsoft Gold Certified Partner based in Norcross, GA. Chris holds several certifications for both Microsoft Dynamics GP and SQL Server. He is a graduate of Georgia State University with a B.B.A in Accounting, and has worked with the Dynamics GP product since 2001. Chris has also previously participated in the Dynamics Partner Advisory Board, and is co-author of *Microsoft Dynamics GP 2010 Reporting*.

Chris' experience ranges from financial analysis, software implementation, data conversions and integrations, to designing and developing customizations in both the functional and technical area of consulting for Dynamics GP.

In addition, Chris has extensive experience in designing Business Intelligence solutions and has assisted numerous clients in analyzing their business plans with these solutions.

Acknowledgments

First and foremost, my thanks and appreciation goes to my friend and co-author, David Duncan. It was truly a privilege to work with him not only on our 2010 version of this book but also this second edition. I wish him and his wife all the best with their newborn daughter.

I would also like to thank our incredible management team at I.B.I.S., Inc. for allowing me the continued opportunity to learn new tools that ultimately allow me to share my knowledge with the community, as well as their support and encouragement in writing this book.

No book is the sole work of just the author and to our reviewers Vaidhyanathan Mohan, Amy Walsh, and Kristi Weldon, thank you for all your dedication and enthusiasm in reviewing our book. We truely believe that it will be the best possible resource it can be due to the attention to detail you gave it.

Special thanks is reserved for the team at Packt Publishing, thank you again for seeing David and me through the various stages of publishing this book. It was a privilage working with the team and being allowed to share my knowledge with a much wider audience than just my clients.

And finally I want to thank my boss and my friend Clinton Weldon. It is because of his continued mentorship over the past 12 years that I have been able to learn so much about ERP products and bring that knowledge to every customer that I visit.

About the Reviewers

Vaidhyanathan Mohan is a Microsoft certified freelance Microsoft Dynamics GP consultant with expertise in Microsoft Dynamics GP and related technologies. He started his career as a GP developer, and slowly and steadily enhanced his skills on Microsoft Dynamics GP (both on the product and technologies) and became a complete product consultant.

He has worked on various challenging customization developments and Dynamics GP implementations. He is an active participant on all Microsoft Dynamics GP community forums; a Microsoft Dynamics GP technical blogger, namely Dynamics GP—Learn and Discuss (`http://vaidymohan.com`), which is listed on Microsoft's official Dynamics GP blog space.

He is who he is now because of his devoted parents, his brother and family, his wife, and his daughter. He is an avid photography enthusiast (`http://500px.com/seshadri`), music fanatic, coffee addict, and is immensely fond of anything to do with Microsoft Dynamics GP.

Amy Walsh is a Senior Consultant with I.B.I.S., Inc. and is a Microsoft Certified Business Management Solutions Professional. She is a graduate of Georgia Military College and Mercer University with a concentration in Accounting and Finance. Prior to joining the I.B.I.S., Inc. team, she worked in public accounting and in private industry. This experience includes over 15 years working in management, financial accounting, audit, and tax in both domestic and international enterprises, which range from start-up to established global B2B and B2C companies.

Over the last seven years, Amy has been focusing on Microsoft business solutions, ERP implementations, SaaS, Business Intelligence, Reporting, Business Process Improvement, and Accounting. Her experience in various industries has been a cornerstone in helping decision makers and sponsors understand and transition to new technology that keep businesses ahead of the competition. Her goal is to continue helping businesses succeed in their endeavors accomplished by finding the right ERP system and reporting tools.

Kristi Weldon is a veteran writer and researcher. A graduate of Auburn University, she worked for the flagship division of VF Corporation. There she interfaced with internal customers and external vendors to develop syndicated and custom reports, winning the VF Excellence Award for Target Marketing. As a freelancer, she specialized in research, analysis, and reporting for Tommy Hilfiger and Fairchild Strategic Information Services. She served as a software subject matter expert for Apparel Industry Magazine where she eventually became Associate Editor. Her technical writing experience includes interactive user guides, proposals, and presentations. Kristi writes and edits full time in Atlanta, GA.

I would like to thank Chris and David for giving me the opportunity to review this book. It has been an honor and a pleasure.

Table of Contents

Preface

Microsoft Dynamics GP 2013 is a sophisticated **Enterprise Resource Planning** (**ERP**) system with a multitude of features and options. Microsoft Dynamics GP enables you to create and manage a variety of reports that help small and mid-size businesses effectively manage their financial and operational data.

Microsoft Dynamics GP 2013 Reporting, *Second Edition* will show you how to create and manage reports, know what tools to use and when, how to use them, and where to find the data based on how it's being entered into the system with Dynamics GP.

It will empower you with the tools and reports necessary to use Dynamics GP data in making key business decisions. The book addresses the many challenges and frustrations organizations face when preparing to build new reports. Then it moves on to explain how to find your data in the GP system and company databases. The book then dives deep into reporting tools such as SmartLists, SL Builder and Excel Report Builder, Report Writer, Word Templates, SSRS Report Library, and Analysis Cubes Design and Management Reporter amongst others. With this knowledge at hand, you will be capable of selecting the most effective tool for the current reporting environment.

What this book covers

Chapter 1, *Meeting the Reporting Challenge*, provides commentary on the many challenges and frustrations a report developer may face when preparing to build new reports. Developers tasked with report creation must be aware of these challenges and select the most effective reporting tool, or tools, to satisfy the company's reporting needs. As well as using the discussion of the challenges faced with reporting as a springboard for the rest of the book, this chapter will also provide commentary on recent reporting trends in the Dynamics GP space.

Chapter 2, *Where Is My Data and How Do I Get to It?*, helps you get a better understanding of how Dynamics GP stores data. This chapter will provide users with helpful tips for finding and locating their data in the GP system and company databases. Knowing where to begin is a critical first step for any technical resource setting out to develop a new report, and this chapter aims to make the process of beginning a new report an easier one.

Chapter 3, *Working with the Builders – SmartList and Excel Reports*, discusses of our first reporting tools as we introduce the SmartList and the Builders: SmartList Builder and Excel Reports Builder. Users will briefly review how to use basic SmartLists for simple reporting. Readers will learn how to deploy the Excel Reports that duplicate the SmartList favorites in Excel format and offers a live data connection that makes the reports instantly refreshable. The final half of this chapter will focus on using SmartList Builder and Excel Reports Builder tools to create additional reports beyond the standard SmartList/Excel favorites.

Chapter 4, *Report Writer and Word Templates*, covers the built-in report writing function of GP 2013 known as Report Writer. This chapter will introduce the reader to the basic layout and the various functions of Report Writer. By the end of this chapter, readers should be familiar with making basic modifications to standard GP reports. Additionally, readers will be exposed to the capabilities and limitations of the Word Templates and corresponding Word Template Generator that allows GP reports to be rendered in Microsoft Word format.

Chapter 5, *Utilizing the SSRS Report Library*, will introduce the concept of utilizing the well-known **SQL Server Reporting Services** (**SSRS**) tool with Dynamics GP data. This chapter opens with a discussion on deploying the predefined SSRS reports and metrics designed specifically for GP 2013 before covering how to display them in the standalone Business Analyzer application. Finally, we will cover the use of Visual Studio to make modifications to existing SSRS reports, as well as to create new report metrics and KPIs that can be deployed on the GP 2013 Homepage.

Chapter 6, *Designing Your Analysis Cubes for the Excel Environment*, is the first of two chapters which will cover the extensive Analysis Cubes for Excel reporting tool. This first chapter will cover the installation of Analysis Cubes and provide details on the various components that are created by the installation. This chapter will then cover some simple modifications that can be made to the Analysis Cubes data warehouse and Analysis Services database to improve the end user reporting experience. Finally, readers will be given important information to consider when planning an upgrade of an Analysis Cubes for Excel environment.

Chapter 7, Utilizing Analysis Cubes for Excel for Dynamic Reporting, discusses Excel PivotTables that are widely used throughout many organizations including those without GP 2013. The first part of this chapter will explore the use of PivotTables specifically with the Analysis Cubes for Excel product. From here, we will explore the use of the lesser-known Excel CUBE formulas that prove to be a useful skill set to have when building static reports and dashboards based on Analysis Cubes data.

Chapter 8, Designing Financial Reports in Management Reporter, introduce readers to Management Reporter and basic report design. This chapter will provide an overview of components that must be configured prior to using Management Reporter before providing tips for navigating the Management Reporter layout. Finally, this chapter will cover the use of the various building blocks of Management Reporter for report creation.

Chapter 9, Viewing Financial Reports in Management Reporter, continues the discussion on Management Reporter begun in the previous chapter. Here, we will cover information related to the Report Viewer component of Management Reporter. In addition to discussing report generation, this chapter also provides information on managing reports through the use of report packages and version control. Finally, this chapter provides some commentary on navigating reports through the Web Viewer released in Management Reporter 2012.

Chapter 10, Bringing it all Together, brings our book to a close by combining the discussion of reporting challenges and trends broached in the first chapter with the reporting tools discussed in the other chapters. Here, we will consider each challenge in light of the various reporting tools. By the end of this chapter, readers will not only be familiar with each reporting tool, but they will have a better understanding of how and when each reporting tool can be used most effectively in their organization.

Appendix, Comparing the Dynamics GP Reporting Tools Against Different Reporting Challenges, contains helpful tables that can be used as a quick reference guide to see how the reporting tools measure up to the various reporting challenges we have already discussed. By presenting this data in table format, readers can quickly scan across a row to see how each tool meets a particular challenge, or they can scan down a column to see how a single reporting tool measures against each individual challenge.

What you need for this book

The required softwares are as follows:

- Microsoft SQL Server 2008 or later (with Database Engine, Analysis Services, Reporting Services, and Integration Services components installed)
- Microsoft SQL Server Business Intelligence Studio (install this as part of the SQL Server installation) or Visual Studio 2008 or Visual Studio 2010.

  If using Microsoft SQL Server 2012, Microsoft SQL Server Business Intelligence Studio will be called Microsoft SQL Server Data Tools
- Microsoft Dynamics GP 2013 RTM or later
- Microsoft Dynamics GP 2013 Add-in for Microsoft Word
- Microsoft Dynamics ERP Management Reporter 2012
- Analysis Cubes for Microsoft Dynamics GP (Microsoft SQL Server 2008 or SQL 2012 depending on server installation)
- Microsoft Excel 2010 or later

Who this book is for

If you are a Microsoft Dynamics GP developer, consultant, or power user who wants to create and manage reports, then this book is for you. A working knowledge of Microsoft Dynamics GP is required. A basic understanding of business management systems and reporting applications such as Microsoft Excel and SQL Reporting Services is highly recommended.

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 name of the OLAP processing package is `DynamicsGP_<warehouse_db>_OLAP_DB_<server name>_<analysis services database>`, where `<warehouse_db>` is the name of the data warehouse database."

A block of code is set as follows:

```
UPDATE LastUpdated
SET [DateUpdated] = '01/01/1900',
  [LastRow] = 0,
  [TempLastRow] = 0
WHERE TableName IN ('GLAccountMaster',
  'GLTransactionsOpen','GLTransactionsHistory');

TRUNCATE TABLE GLAccountMaster;
TRUNCATE TABLE GLTransactions;
```

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

```
=CUBEMEMBER("GP Financials Cube","[Accounts].[Acct No].[All].
  [000-1100-00]")
=CUBEMEMBER("GP Financials Cube","[Measures].[Amount - GL Trans]")
```

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: "The first three nodes—**Date**, **Date by Month**, and **Quarter by Year**—under the **Master Date** node are known as hierarchies."

[Warnings or important notes appear in a box like this.]

[Tips and tricks appear like this.]

Reader feedback

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

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

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

Customer support

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

Downloading the example code

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

Errata

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

Piracy

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

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

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

Questions

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

1 Meeting the Reporting Challenge

In an increasingly digital world, utilizing proper reporting techniques is essential to make sense of the increasing mountain of data we find stored on our computers.

The task of reporting on and identifying trends in this data is an exciting topic that will only grow in importance as companies continue to take advantage of cheaper disk space and the growing capability of applications and **Enterprise Resource Planning** (**ERP**) systems that can track and record every last detail of conducting a business.

As developers and consultants tasked with report writing, our goal is to provide the decision-makers in our organization with the tools and reports necessary to use this data when making key business decisions. We can help our end users unlock the trends found within the data being analyzed. By providing accurate reports that bridge the gap between disparate information systems, we can provide users across the company with accurate, reliable information. In other words, we can provide them with a single version of the truth so that multiple end users can make decisions from the same set of data.

This book is intended for anyone who might be tasked with developing and creating reports, whether they are for personal use or for use by other report consumers. The term "developer" invokes an image of someone who is technically adept and possesses a unique understanding of the inner-workings of a particular piece of software. But, in the context of report-writing (and this book), the term developer has a much wider meaning. Instead, report developers can include those with more application experience such as senior accountants and day-to-day operations personnel. It can also apply to application consultants who are well-experienced with an organization's ERP application.

As we reach the end of this chapter, we should possess a better understanding of the following:

- Major trends facing report writers and developers in today's hyper-competitive environment
- The end user-driven challenges or requirements that exist when designing a report according to an end user's specifications
- Organizational challenges that may help or hinder us from designing a report that meets the end user's requirements
- Weighing the pros and cons of numerous trends and challenges in light of each other in order to select the most effective report or reporting tool possible

While the focus of this book will primarily rest on reporting tools associated with the Microsoft Dynamics GP ERP application, the concepts required to design and build reports apply to the entire spectrum of ERP applications and various data repositories in general.

Trends in reporting

As companies have adapted to new ways of capturing and recording business data, the techniques for reporting and making sense of this data has also changed. The best report developers are those who understand how these techniques have changed and are willing to update their own skill sets to capitalize on this.

Some of the most recent, important trends in the reporting space include:

- Increased flexibility
- Reporting through all levels of an organization
- Increased access to the report generation process

Understanding how reporting techniques have changed in the last few years provides us an opportunity to improve our ability to produce relevant and useful reports. Developers and consultants who understand how to exploit these trends can position their organization's reporting capabilities to provide a true competitive advantage. Understanding past trends will provide us with insight into the future of reporting so that we may begin taking steps to prepare for new tools and techniques.

Increased flexibility

Designing and developing reports has long been the domain of IT departments. If an end user identified a need for a particular report, he or she would typically have to submit a request to the IT department or some other functional group assigned to report development and creation. The process of designing and creating the report would then be assigned to a developer who, more than likely, had a number of other tasks waiting to be completed. Unsurprisingly, the end user would not receive his or her report for several days, if not weeks or months! By then, it is entirely possible that the end user's needs may have changed and the anticipated report becomes obsolete before it can even be used for the first time.

Is it any wonder then that this inefficient practice has quickly given way to a more flexible process? While many situations still require the involvement of an experienced technical developer; in today's business environment, end users have a reasonable expectation that they will be provided with the tools necessary to create their own reports. As business software becomes more accessible and easier to use, end users increasingly expect to be equipped with the tools they need to create their own reports. This allows end users to design and create reports that match their specifications and to do this in a much quicker time frame than would be possible if this report were being designed by an individual in another department.

On the surface, this trend offers an extraordinary level of efficiency and flexibility that should be welcomed throughout all organizations. Certain challenges with this model do exist, however, and developers and consultants should be aware of these. Of course, end users who wish to create their own reports cannot have unfettered access to an organization's production database, nor will most of them understand the hardware and network requirements for working with reporting tools that handle large amounts of data. Therefore, while traditional IT departments are no longer the key designers and creators of most reports, they still have a key role to play in providing a central oversight for the reporting needs of the entire organization. Additionally, developers and consultants must also expand their focus beyond the IT department to ensure that end users are well-equipped and trained in making the most effective use of their reporting tools.

Reporting through all levels of an organization

In years past, access to reports was typically limited to members of the Executive team. As the long-term decisions makers for the company, it was the Executive team that benefitted the most from reporting tools that provided a snapshot of company performance over a period of time. Reporting at other levels of the organization was generally an afterthought, if it was not ignored entirely. Of course, one reason for this may lie in the fact that ERP systems were generally not as adept at capturing information from all aspects of the business; instead, the focus was on capturing general ledger information, which is generally perceived to be the domain of the Executive and Accounting teams. Additionally, perhaps due to security and trust, Executive team members often did not see a reason to share key metrics and information with lower level team-members.

In recent years, however, as ERP systems have become more adept at capturing a wider variety of information across all functions of business, reporting across all levels of the organization has become more important. Executive teams are no longer the exclusive focus of an organization's reporting efforts. Today, in an effort to gain competitive advantage, all levels of the organization must utilize reporting tools to improve their ability to make decisions on a day-to-day basis. In other words, organizations are now focused on putting their hard-earned business data in the hands of those who can benefit the most from it. They are also discovering that transparency can improve an employee's performance and focus. Employees can quickly and easily see how their individual and team performance is impacting the overall company performance. This leads to greater buy-in and focus from all levels of the organization.

As a developer or a consultant tasked with developing reports, the focus is no longer on just developing balance sheets and income statements for an Executive team. While such reports are still relevant and necessary, additional reports for lower levels of the organization are also important. The Sales manager expects improved insight into customer spending behavior. The Warehouse manager demands access to reports that identify orders behind schedule. Developers must be aware of the various reporting audiences that exist within an organization, the types of reports that are required by these audiences, and the security that is required to ensure that sensitive information is only seen by the right individuals.

Increased access to the report generation process

Successful organizations are the ones that make their data available to all levels of the organization. One reason for this trend lies in the fact that newer, more effective reporting tools have improved the ability of users across all levels of an organization to access company data. In previous years, hardware and resource limitations meant that reports were generated at specific times such as month-end close. In a less-connected business environment, where decisions did not have to be made on a minute-by-minute basis, companies could afford to wait until certain points in time to generate and review their financial statements.

However, in today's hyper-competitive business environment, this is no longer a luxury that organizations can afford. Instead, any report, whether a key financial statement or an aging trial balance, must be capable of being generated at a moment's notice with near-to or real-time company data. Organizations' members cannot afford to wait until the weekly sales meeting or the quarterly Executive team meeting to review key reports; instead, they must be generating and reviewing these reports on a daily basis.

Developers and consultants must be aware of this trend and seek ways to make reports more readily available to an organization. Issues such as latency and hardware must be considered to ensure that reports can be generated at a moment's notice without any negative impact to transactional processing. One of the most effective ways to do this is to make a wide range of reporting tools available to all levels of the organization and then equip team members with the knowledge necessary to utilize these tools effectively.

Challenges to developing and writing reports

Now that we have identified some of the common trends that exist in today's reporting domain, let's take a look at some of the challenges these trends pose to the developer or consultant setting out to create a well-designed report. As we've seen, each trend leads to a number of different challenges. Careful consideration of these challenges can be the deciding factor in whether or not a report gains acceptance from the intended audience.

From our experience with designing reports, we have identified a list of nine common areas or challenges that we think are important to consider before setting out to design or build the perfect report:

- Intended audience
- Data sources
- Latency
- Formatting and presentation
- Ad-hoc reports vs. traditional reports
- Security
- Network access and general IT infrastructure
- Developer resources

While this is certainly not a comprehensive list of all the challenges we will face while designing reports, it does include some of the most important ones to consider when developing and creating reports. The first five challenges relate to how the end user anticipates the report will be designed. Obviously, these challenges should be addressed based on feedback from the end user. The final challenges are less about end user requirements and more about the environment in which we will be designing the report. Often, it can be a challenge to explain to an end user why certain environmental challenges such as lack of developer resources may prevent a particular report from being built to his or her specifications.

Keep in mind that becoming a successful report developer does not always mean delivering a final report to the intended audience. Instead, in many cases, the true "deliverable" that will result from the report writing process will come from equipping the intended audience with the tools necessary for them to create their own reports. As we work our way through the rest of this book, remember that when we discuss creating a successful report, we are really commenting on the overall reporting solution, regardless of what shape or form it may take.

Intended audience

The intended audience is any individual or business entity that will use the report as a means to answer questions about specific business functions. A successful reporting solution will be one that is designed with the ultimate end user in mind. It will answer the question or questions set forth by the intended audience in a clear, concise manner.

Within an organization, an Executive team may request a cash flow statement in order to view the company's cash position and to understand how much cash is available for further investment. An Accounts Payables Coordinator for the same company may need a report that identifies vendor's terms and discounts to determine which vendors should be paid sooner rather than later. Although both audiences may come to the same internal developer for these reports, the developer must be capable of understanding the requirements of the target audience.

A developer may also be tasked with creating a report for an external audience. For example, auditors or external creditors will request reports from an organization as a means to understanding more about an organization's business practices, transaction information, or financial position. It is possible that these reports will need to be submitted to external auditors in a specific, pre-determined format.

In all cases, a report writer must be aware of the "me-me-me" syndrome. Requests for reports are usually the result of a need to answer a question specific to the intended audience's immediate function. To an individual who needs a specific report, no report is more important than the one he or she has requested. Not surprisingly, the requesting user will go to great lengths to prove that his or her report should receive top priority over other, yet-to-be created reports. While more clever developers may use this to their advantage and request bribes in the form of cookies or more vacation time, effective developers will be wary of those who place their reporting needs above all others. Most importantly, if it is your boss whom is the one requesting the report, then by all means, you better pay attention!

This is also a good time to warn report developers to be on the lookout for the **all-knowing** report. Either we have been asked to write a report of this nature or have heard horror stories of it from fellow developers. This is the report that is supposed to give the user every bit of information from the entire system in one report, hence its name, all-knowing. Now, this almost always turns out to be an impossible feat. Sure, it sounds easy enough to the user, but if—and that is a big if—the report can actually be created, does it provide any true value or is it just a discombobulated mess? If tasked with writing this type of report, don't be afraid to question the need and find out if this really should be a single report or multiple reports that present the data in a meaningful way. Often, simply posing the question, "What do you want to accomplish, and how does this report help you accomplish that goal?", will spur a meaningful conversation between the developer and the report consumer. The answer may uncover a need for training users on existing reporting tools—perhaps by sharing this book with the user!—or help the two parties narrow down the focus of the report.

Data sources

While the goal of any ERP system is to provide a single comprehensive location for recording all the functions of a business; in reality, we find that most companies, in addition to the production ERP database, utilize a wide variety of different systems. Each system is incorporated with its own set of business logic for recording data. These silos of information can range from entire database applications down to a static spreadsheet that an end user keeps on his or her desktop. As well as with the increasing move towards cloud-based computing, our data is no longer necessarily found on-premise, it could be found in the form of an RSS feed, Odata feeds from the Windows Azure Marketplace, or some other web-based data source. While a company's reasons for a lack of a single data repository can vary from the lack of money required to combine systems to the inherent difficulty in transitioning from an older system to a newer system, the reality is that this kind of environment provides numerous challenges when it comes to accurate and timely reporting.

Let's think about what some of these challenges of reporting across multiple data sources might be:

- Data may be duplicated across multiple systems. The fact that some of this data may be stored in cloud-based applications, while other data may be stored in on-premise data stores, only adds to the complexity of this challenge.
- The business logic utilized by the systems that capture information may differ from system to system, leading to differences in how the data is stored in each silo.
- Timing differences may exist between silos. Data in one system may be updated on a real-time basis whereas in another system, it might be updated on a weekly basis.
- Levels of granularity may differ among data sources. For example, one data source might record customer information at the sales order line level, while another data source might only record information at the customer level.

Each of these challenges must be managed to provide accurate and effective reporting. First, developers and consultants must be careful to select a reporting tool that can bridge the gap between these disparate systems without adding another independent silo of information. Once this tool is selected, it must be used effectively to present one version of the truth to the end user. By this, we mean that it must provide users with a consistent and reliable look at the data in the report, regardless of how many data sources were used to generate the data in the report in the first place.

Let's be honest, however, accumulating and storing data outside of the ERP database is not always a bad thing. A company cannot and should not expect to conduct all reporting against the production ERP database, which is where the majority of the transactional and statistical information resides. While this method is more likely to allow for real or near-real time reporting, it is also likely that it will cause a decrease in system performance for other users. Two common techniques exist for most companies who want to provide reporting tools to users without impacting on production performance:

- Set up a separate data warehouse. This separate data warehouse contains data from one or more enterprise systems and provides a separate location against which users can generate reports and queries. Data in this warehouse is updated at pre-set intervals via extract, transform, and load tools such as SQL Server Integration Services (SSIS).
- Use SQL Server technologies such as log shipping, database snapshots or, with SQL Server 2012, AlwaysOn Availability Groups, to create copies of the primary user databases. Report solutions can then use these database copies as a data source, thus removing the impact of queries from the primary database(s).

Some companies take the concept of a separate reporting environment one step further by creating a separate reporting server to host reporting related functions including data warehouses or database copies. This completely removes the performance impact of reporting away from the ERP system database.

Regardless of the reporting tool we select, we must be aware of the trade-off with the various sources of data that may be used to generate the report. If we're reporting from a production database, we may encounter additional issues with security and hardware performance. Likewise, if we are extracting data for reporting from a separate data warehouse, we have to be aware that the data may not be in real time. Furthermore, if we plan to utilize multiple sources of data, such as when we want to combine data in an Excel spreadsheet with data stored in our ERP application to generate our report, we must pay attention to challenges such as duplicate data, business logic, and more.

Latency

As we have discussed, trends in reporting have led to more users wanting more data in real time to gain a more competitive business advantage. When we speak of latency, we are referring to the delay between when source data is generated and when it can actually be used in a report. For example, if a journal entry is posted to a revenue account, but the President of our organization must wait for a scheduled report generation process to see the impact of this entry on an income statement, then we have a period of latency between when the source data was generated and when it was retrieved by a particular report.

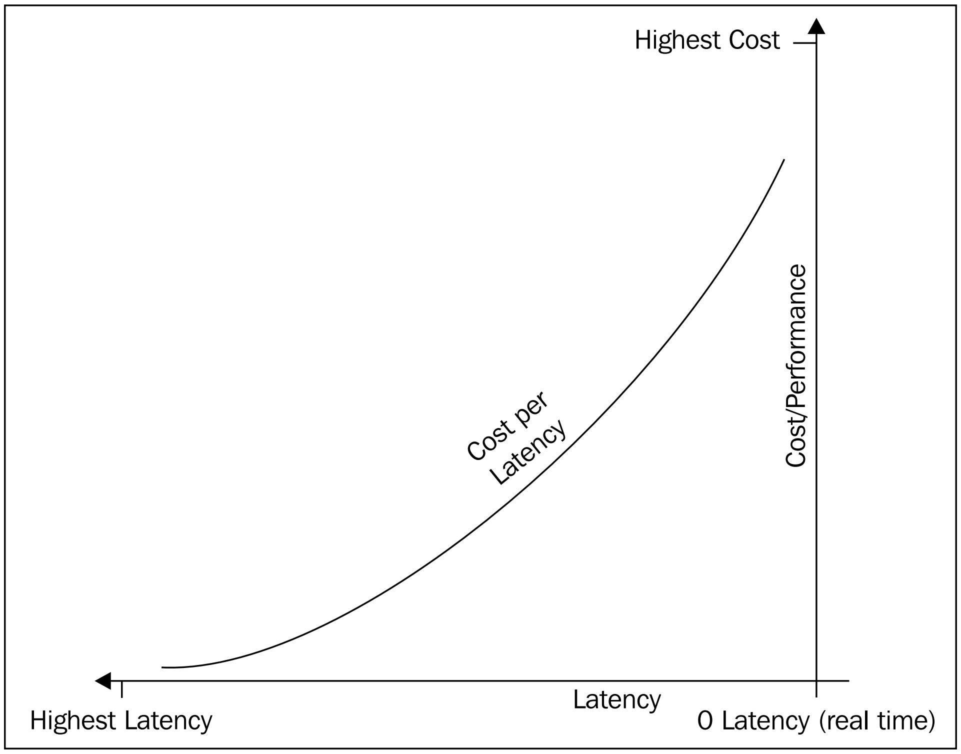
As a report developer, one of the first questions that needs to be answered is, "Does the report need to be real-time or can it be generated after some arbitrary delay?", which can and usually does lead to other challenges. Often, users will want everything in real time.

Mostly, true real-time reporting is either nearly impossible to achieve or can cause a real strain on the infrastructure. In addition to this, the developer must ask, "Is there really a true need for the report to be in real-time?". Take for example, an Accounts Receivable manager who is requesting a real-time aging report. Is the benefit of having this run on real-time data going to make a dramatic difference to what is actually collected on these open receivables? Probably not! Now, if we take an example of a Sales manager requesting a real-time report of open sales orders that have not shipped yet, having this information can make a difference. This will give that Sales manager a report from which he or she can take immediate action by going to shipping and identifying the hold up on the orders.

Another important factor in determining how realistic real-time reporting may be in a particular scenario is how much data is involved. Running real-time reports over significantly large data sets could render our reports out-of-date before we even have a chance to use them. In the worst cases, the underlying data set may actually change during the time it takes to render the report, meaning we are acting on inaccurate information! Sometimes large data sets are unavoidable, and filtering conditions and other techniques must be utilized to trim the delay required to generate the report so that we do not receive obsolete and inaccurate information.

It's important to note that the closer we get to real-time reporting, the cost of resources required to provide such real-time reporting will become greater, while the extra increase in added benefit from access to this real-time data will shrink. At the same time, the cost of accessing real-time data usually outweighs the extra benefit provided by having access to such data. When this happens, we face the challenge of convincing our report-users that the data latency they may experience in their reports must be acceptable.

An example of the concept of increasing costs for a decrease in latency is seen in the following figure. As we move from high latency, for example, data provided the next morning, to low latency, real-time data, the cost of that goes up in terms of both lower performance and higher monetary cost.

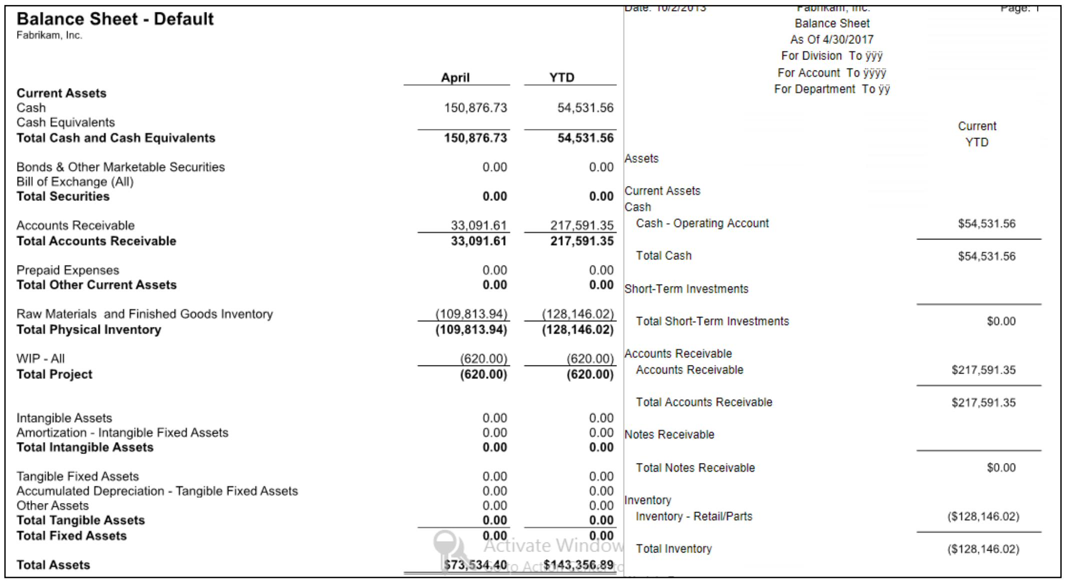


So, how does a report designer weigh the pros and cons of providing end users with access to real-time data? The main thing we want to ask ourselves as report developers when determining whether a report truly needs to be real-time is whether or not the data in the report is critical to spurring immediate action or changes in the day-to-day operations of our organization.

Formatting and presentation

Before selecting an appropriate reporting tool for a given situation, the requirements for formatting and presentation must be given careful consideration. To some degree, the appropriate solution to this challenge will rely on the report's intended audience. External reports, such as those being sent to shareholders, customers, and clients other interested parties will, in most cases, require company logos and other colorful visuals that represent the company in a professional manner. On the other hand, reports being designed for internal use, such as for a Warehouse Manager, may not require extensive formatting and presentation capabilities.

As can be seen in the following figure, you can see how much more presentable the balance sheet on the left is when compared to the standard out of the box balance sheet on the right:



Balance Sheet - Default
Fabrikam, Inc.

	April	YTD
Current Assets		
Cash	150,876.73	54,531.56
Cash Equivalents		
Total Cash and Cash Equivalents	**150,876.73**	**54,531.56**
Bonds & Other Marketable Securities	0.00	0.00
Bill of Exchange (All)		
Total Securities	**0.00**	**0.00**
Accounts Receivable	33,091.61	217,591.35
Total Accounts Receivable	**33,091.61**	**217,591.35**
Prepaid Expenses	0.00	0.00
Total Other Current Assets	**0.00**	**0.00**
Raw Materials and Finished Goods Inventory	(109,813.94)	(128,146.02)
Total Physical Inventory	**(109,813.94)**	**(128,146.02)**
WIP - All	(620.00)	(620.00)
Total Project	**(620.00)**	**(620.00)**
Intangible Assets	0.00	0.00
Amortization - Intangible Fixed Assets	0.00	0.00
Total Intangible Assets	**0.00**	**0.00**
Tangible Fixed Assets	0.00	0.00
Accumulated Depreciation - Tangible Fixed Assets	0.00	0.00
Other Assets	0.00	0.00
Total Tangible Assets	**0.00**	**0.00**
Total Fixed Assets	**0.00**	**0.00**
Total Assets	**$73,534.40**	**$143,356.89**

Date: 10/2/2013 Fabrikam, Inc. Page: 1
Balance Sheet
As Of 4/30/2017
For Division To ÿÿÿ
For Account To ÿÿÿÿ
For Department To ÿÿ

	Current YTD
Assets	
Current Assets	
Cash	
Cash - Operating Account	$54,531.56
Total Cash	$54,531.56
Short-Term Investments	
Total Short-Term Investments	$0.00
Accounts Receivable	
Accounts Receivable	$217,591.35
Total Accounts Receivable	$217,591.35
Notes Receivable	
Total Notes Receivable	$0.00
Inventory	
Inventory - Retail/Parts	($128,146.02)
Total Inventory	($128,146.02)

Understandably, this reporting challenge often takes a back-seat to other, more important challenges such as latency and security. However, providing a clean, colorful report, can play a critical role in a user's acceptance of a report and its contents. Therefore, be mindful, in selecting a reporting tool with easy control of the report formatting and graphical inserts to ensure that the report viewers are more likely to be influenced by the contents of the report.

But, this challenge is not just about using fancy fonts or selecting pretty images and logos from a graphics repository. It is also about providing appropriate tools for improving the readability of your report. For example, if we are developing reports that display key financial metrics, we want our reporting tool to have the ability to display data in straight rows and columns. Viewers of the finished product should be able to follow the logical flow of information presented in the report. Whether this means looking across columns to see how performance has changed over time or down the rows of an income statement to see how revenues compare to expenses over the same period of time, the report should be laid out in logical fashion.

Our report may also be required to meet certain standards for formatting. For example, publically traded companies must submit certain financial statements and these statements must be prepared according to certain principles and standards. A few examples of this include **eXtensible Business Reporting Language** (**xBRL**) reporting and GAAP/FASB/IASB regulations. If this is the kind of report we will be developing, then it should play a prominent role in the selection of the reporting tool.

Ad-hoc reports versus traditional reports

In addition to the challenges covered this far, we must also consider the type of report we need to create. Is this a report that our users will create to address a specific, one-time need? Or, is it a report that users will need to produce repeatedly over a period of time? Additionally, the selection we make here impact how these reports are distributed among the various individuals who need to see the contents of the report.

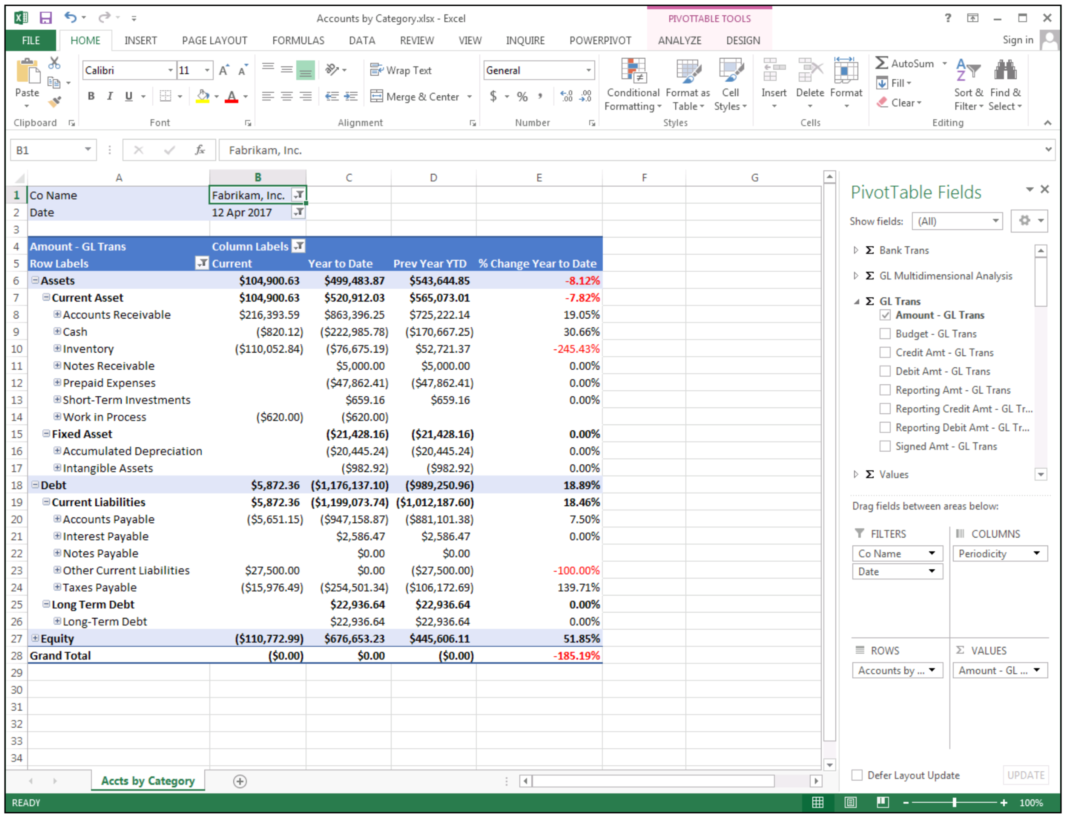
The concept of ad-hoc versus traditional reporting can be broken down into two components:

- Difficulty level of report modification after initial creation
- Report distribution options to end users, decision makers, and other interested parties

Every organization has a need for viewing its financial performance at or over a period of time through the use of financial statements. Financial statements are traditional reports that typically require some level of pre-planning before the initial setup or report creation. These efforts are to determine how they will be structured, the level of detail, generation frequency, and so on. After the report is created, the structure requires very little in the way of modification for future financial periods. Therefore, while it may be helpful, it is not always critical that the reporting tools we use to generate our financial statements offer us exceptional flexibility in changing or modifying reports.

In addition to reporting structures that will be used again and again over multiple financial periods, individual users may have a unique reporting need that arises due to a specific business challenge. Rather than go through the effort of setting up a traditional report that can be used over and over again, our user simply needs a reporting tool that can quickly generate a report on an as-needed basis in order to answer the specific question at hand. In exchange for this ad-hoc capability, ad-hoc report viewers may be willing to allow a trade-off in other areas such as formatting and presentation.

In the following screenshot, we see a sample of the **PivotTable** functionality from Microsoft Excel that provides users an opportunity to query their data in an ad-hoc fashion:



B1: Fabrikam, Inc.

Co Name	Fabrikam, Inc.
Date	12 Apr 2017

Amount - GL Trans	Column Labels			
Row Labels	Current	Year to Date	Prev Year YTD	% Change Year to Date
Assets	**$104,900.63**	**$499,483.87**	**$543,644.85**	**-8.12%**
Current Asset	**$104,900.63**	**$520,912.03**	**$565,073.01**	**-7.82%**
Accounts Receivable	$216,393.59	$863,396.25	$725,222.14	19.05%
Cash	($820.12)	($222,985.78)	($170,667.25)	30.66%
Inventory	($110,052.84)	($76,675.19)	$52,721.37	-245.43%
Notes Receivable		$5,000.00	$5,000.00	0.00%
Prepaid Expenses		($47,862.41)	($47,862.41)	0.00%
Short-Term Investments		$659.16	$659.16	0.00%
Work in Process	($620.00)	($620.00)		
Fixed Asset		**($21,428.16)**	**($21,428.16)**	**0.00%**
Accumulated Depreciation		($20,445.24)	($20,445.24)	0.00%
Intangible Assets		($982.92)	($982.92)	0.00%
Debt	**$5,872.36**	**($1,176,137.10)**	**($989,250.96)**	**18.89%**
Current Liabilities	**$5,872.36**	**($1,199,073.74)**	**($1,012,187.60)**	**18.46%**
Accounts Payable	($5,651.15)	($947,158.87)	($881,101.38)	7.50%
Interest Payable		$2,586.47	$2,586.47	0.00%
Notes Payable		$0.00	$0.00	
Other Current Liabilities	$27,500.00	$0.00	($27,500.00)	-100.00%
Taxes Payable	($15,976.49)	($254,501.34)	($106,172.69)	139.71%
Long Term Debt		**$22,936.64**	**$22,936.64**	**0.00%**
Long-Term Debt		$22,936.64	$22,936.64	0.00%
Equity	**($110,772.99)**	**$676,653.23**	**$445,606.11**	**51.85%**
Grand Total	**($0.00)**	**$0.00**	**($0.00)**	**-185.19%**

PivotTable Fields

Show fields: (All)

- Σ Bank Trans
- Σ GL Multidimensional Analysis
- Σ GL Trans
  - [✓] Amount - GL Trans
  - [ ] Budget - GL Trans
  - [ ] Credit Amt - GL Trans
  - [ ] Debit Amt - GL Trans
  - [ ] Reporting Amt - GL Trans
  - [ ] Reporting Credit Amt - GL Tr...
  - [ ] Reporting Debit Amt - GL Tr...
  - [ ] Signed Amt - GL Trans
- Σ Values

Drag fields between areas below:

FILTERS: Co Name, Date

COLUMNS: Periodicity

ROWS: Accounts by ...

VALUES: Amount - GL ...

[ ] Defer Layout Update   UPDATE

As we answer the challenges associated with traditional reporting tools versus ad-hoc tools, we must also determine how and when the report will be distributed to those end users who do not have the ability to generate the reports. Will these users be reliant on a power user or other application user to generate the report? Will the results need to be emailed, published to a website, or printed for a financial package?

In most, but not all, cases, we will find that the traditional reports are more easily distributed than ad-hoc reports. Also, traditional reports such as financial statements usually contain information that is useful across a diverse user group. This eliminates the need to create multiple stand-alone versions of the same report and reduces the time needed to manage the different types of distribution channels.

Furthermore, these reports contain data presented in a standard format, making the reports useful for users across a wide variety of backgrounds. If a report will be distributed on a regular basis, we must also make additional considerations to manage the security of these reports (which we will discuss momentarily), the process by which these reports will be distributed (for example, via e-mail or via a central location like a company intranet), as well as the manner in which reports will be scheduled for distribution.

On the other hand, many reports generated through ad-hoc reporting tools are not usually distributed to a wide variety of users. By their very nature, ad-hoc reports are developed quickly, and for a single purpose. For example, an external auditor may request a list of specific all journal entries or transactions made by a company for a particular time period. This is a typical request by external auditors as they prepare a company's audit report. However, it isn't really information for which a company will need a traditional report, nor is it a report that will be saved and re-distributed. Rather than expend time and energy developing a report that is used only once, we may be better off utilizing an ad-hoc reporting tool to meet our needs.

A good report developer or consultant will be able to pick up on key requirements from the report requestor and quickly determine whether the information being requested is better pulled in the form of an ad-hoc report or some other type of traditional report.

Security

Few reports exist that can be distributed to anyone and everyone inside and outside of an organization without considering security. Instead, most reports contain sensitive data that must be tightly controlled to ensure it does not fall into the wrong hands. Within an organization, reports developed for one department may not be suitable for employees in another department. For example, consider a report on year-to-date payroll amounts developed for the Director of Human Resources. This type of report is usually considered sensitive information that should not be seen by others in the organization. Consider, as well, reports in the medical field that might contain **Protected Health Information** (**PHI**). This information, by law, must be secured in a certain way before distribution to certain parties in order to protect the privacy of the patients represented by the data.

When selecting a reporting tool, developers and consultants must consider how effective that reporting tool will be when it comes to managing user access to the data contained in the report. Will entire reports be restricted to certain users? Or, will one format be provided to all users with the select data being displayed based on the user viewing the report? It is also worth considering which attributes will be used to determine security. In some cases, it may be the department that a user belongs to, or it may be that user's functional role within the organization that allows him or her access to certain reports. In other cases, still, it may be that reports will be password protected, and only those with the password will be allowed access to the reports.

As developers and consultants tasked with report writing, we often find ourselves with greater access to company data than that of the average end user. With this access comes a high level of responsibility. One of the greatest proprietary advantages a company has over its competitors is the data that it carefully maintains in its ERP and related applications. By selecting reporting tools with security in mind, we take a critical step towards ensuring that hard-earned company data will continue to provide an extra advantage over our competitors.

Network access and general IT infrastructure

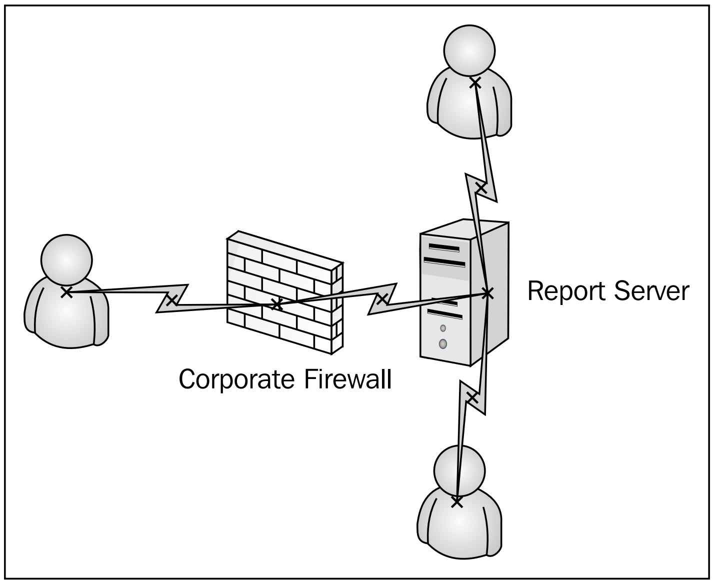
Access and infrastructure play multiple roles in determining our reporting solution. Not only do we need to determine whether our infrastructure will allow the type of reporting we are going to be utilizing from a performance perspective, but we also need to determine if it will provide the necessary access to our intended audience.

Some of the questions we might ask with regard to this reporting challenge include:

- How will our intended audience access the reporting tool?
- How much data is being transmitted across the network?
- Do we have a dedicated server for reporting purposes?
- Do we have sufficient disk space to support a data warehouse?
- Will the reports be memory intensive?

By asking these questions and more, we can make a more accurate selection of a reporting solution.

As the following figure shows, users in our organization may require external access to the reporting server. This requires taking firewalls and network access into consideration as we select a reporting tool. Additionally, we may have users who will be asking for reports internally, and those users may be able to take advantage of a completely different reporting tool.



With the advent of technologies such as Citrix and Terminal Server, the ability to give either internal or external access to users on a large scale has been made easier. Many companies that run some type of ERP system will usually already have one of these solutions in place to grant their users remote access. In addition, this type of infrastructure allows companies to centralize their servers, keep maintenance down to a minimum, and keep the systems standardized. In these types of environments, we can easily add on our reporting solutions.

In deciding what the most appropriate reporting solution is, we must also consider how much data is being transmitted across the network. If, for example, we are trying to pull down large amounts of data in the middle of business hours, what kind of effect is that having on our network? Do we need to add bandwidth to our network or can we work within the current bandwidth we have? The answer to this question is usually imperative to deciding on the most appropriate reporting solution.

We have talked about various areas of infrastructure that need to be considered when writing reports. Another important area is available disk space. Whatever reporting solution we ultimately decide on, will, to some degree, require additional disk space. Whether or not the reporting tool requires enough space for an entire data warehouse or if it can rely on something as simple as enough local disk space that contains a report depository or data store, the report developer or consultant needs to be aware of this. We must also determine how much anticipated growth in the data there will be so that disk space can be planned accordingly. One thing we want to avoid is recommending and building a solution that has a short lifespan because it runs out of disk space and crashes the server! As report developers and consultants, it is our job to figure out how long the lifespan of the solution needs to be based on the customers' needs and plan for that growth, ensuring that end solution indeed fills that need.

The last thing we want report developers and consultants to be aware of from an infrastructure standpoint is how memory intensive the reports can be. We have discussed this briefly in terms of dedicated reporting servers, but we want to specifically point out the advent of **in-memory** reporting tools like PowerPivot for Excel. With this tool, data is stored in-memory on whatever computer is hosting the PowerPivot model. Users must be aware that the more data they add to the model, the more memory they will consume. This can quickly cause issues, especially for other applications on the same machine.

In addition, we want to point out that even with sufficient memory, some reports will just take time to run. A perfect example is a heavy distribution company with thousands of orders. To run an open order report against thousands of orders that might have a large number of line items, we are looking at a potentially long report generation time. It is for these types of reports that we must decide on how frequently the report will be generated. Is it something we can schedule to run overnight to be ready first thing in the morning? Or, is it something we can break down into smaller runs, with filters/parameters to spread out the load? These are just some of the things to think about and discuss with the report requestor.

Developer resources

So far, we have mainly discussed the challenges to do with choosing the proper reporting solution. Another piece of the puzzle is the resources that will be tasked with actually developing the reporting solution.

Many times, equipping our intended audience with the tools necessary for them to create their own reports is the right course of action. In this model, these power users are less technical, but they understand the data they are working with and the output that they need very well. One thing that we as report developers and consultants need to take into account is whether this model works for the organization as a whole, and that the IT department understands that they are going to have less control over the solution as a whole. We also need to make sure that we don't lose corporate governance over sensitive information.

Although empowering users to be able to generate their own reports has grown more and more acceptable, many IT departments don't want to lose control of the management of the data and are concerned that they will lose corporate oversight and create multiple silos of data. Because of this, report development is usually assigned to a dedicated report developer, be that either an internal resource in the organization, or an outside developer or consultant.

Organizations will often determine whether to develop reporting solutions in house or to outsource it based on many factors. The main factors for our purpose are who is available to develop the reports and whether any time and budget constraints exist. Even if the organization has the required skill set in house, the resources might not necessarily have the time to dedicate to developing the report. This is when companies will often look at outside resources to get the job done. On the other hand, factors such as budget constraints may force an organization to keep these resources in-house.

Summary

As we work our way through the rest of this book, always keep in mind that our goal is to provide our end users with accurate and reliable ways to mine the data found in our reporting systems. If we cannot maintain accuracy and reliability, then our end users will quickly seek out another report developer who can meet their needs!

Creating accurate and reliable reports requires the ability to properly identify the end user and corporate requirements. The ability to identify the requirements for a report or reporting tool is a valuable skill to have in today's business environment. Although, this skill doesn't come easily! Instead, developers and consultants must constantly ask challenging questions of end users and the corporate infrastructure to select the most effective reporting tool for the circumstances.

Unsurprisingly, the process of creating a report is not often as easy as creating a data connection to a database and throwing data up onto a computer screen or a PDF file. But, in order to make it easy, we've provided you with information to set you along the way towards identifying the requirements for your report.

As we begin to cover specific reporting tools for Microsoft Dynamics GP, keep the trends and challenges we've discussed in this chapter in the back of your mind. Continue to ask yourself how each reporting tool can help you meet the challenges that we've presented in this chapter.

In the next chapter, we will begin by looking at some tips and tricks to locating our data in the Microsoft Dynamics GP system. Then, we will look at certain tools that can help us in finding the specific data we need to begin writing our reports.

2
Where Is My Data and How Do I Get to It?

Now that we have looked at the latest trends in reporting and have gained an understanding of the many challenges that go along with report development, we are ready to start data gathering and report writing. As any developer or consultant that has been tasked with filling user requests for reports is aware, ultimately the very first question after deciding on the reporting tool is, "Where is my data and how do I get to it?"

Knowing where to begin is a critical first step in the development process. The aim of this chapter is to provide helpful tips for finding and locating data in the Dynamics GP 2013 ERP system and company databases. Although we'll discuss some reporting tools that do not require us to know the SQL database structure for Dynamics GP companies, it is still helpful to understand how GP stores its data.

In this chapter, we will discuss the following:

- Differences between the system database and company databases
- Conventions that are helpful to know and understand when it comes to Microsoft Dynamics GP 2013 data and how it is stored
- Using Resource Descriptions as a tool for finding data from within GP 2013
- Utilizing additional tools, such as the GP 2013 SDK and Support Debugging Tool, to find our data

System databases versus company databases

The first task of identifying where our data is located is making sure we are using the correct database! Microsoft Dynamics GP 2013 utilizes the Microsoft SQL Server platform as its database engine. When Dynamics GP 2013 is installed on our environment and a new company is created, the installation process creates several databases on a server that has been designated during the installation. These databases will store all the information entered through the Dynamics GP 2013 application, and we can use SQL Server Management Studio to access the underlying tables that store this data.

Microsoft Dynamics GP has two types of databases, a system database and company database(s). For first-time report developers and seasoned writers, knowing which of these databases stores a particular piece of information we need is crucial for an accurate report.

System databases

Prior to GP 2013, the system database was named the DYNAMICS database, and this default name could not be changed. Now, with the new installation of GP 2013, the system database can be named as something different than DYNAMICS. This functionality was introduced as a way to allow multiple sets of GP installations to reside on the same SQL server instance. This is especially important for companies providing GP hosting services for multiple companies on a single SQL server instance.

When Microsoft Dynamics GP is first installed and some initial settings are provided, a system database will be created. This database is the system database that can contain up to ten characters. It includes things such as records for each company that you create, the organization's registration information, and the maximum account framework.

While many of us can expect to work in environments where only one system database exists, and where that system database uses the standard 'DYNAMICS' name, we should still be careful not to hard-code references to this database. While we may have been able to take this shortcut in reports for earlier versions of GP, this will surely cause issues when we least expect. Whenever querying company data, use the DBNAME field from the SY00100 table in the company database to ensure the right system database name is being used.

From a reporting standpoint, there is certain information located in the system database that we may need to report on at one time or another. We have provided a quick reference for this information in the following list:

- **Multicurrency System Setups**: This includes the setup of the currencies, the exchange rates, and the currency symbols for those organizations that process transactions in any number of foreign currencies.
- **Intercompany Setup**: This is where the intercompany relationships are stored and the specific dues to/due from accounts are mapped.
- **Organizations Structures**: This will include the organizational levels and entities that have been created for those organizations that use this feature.
- **User Master**: This table stores information about the users in the ERP system, including their user ID and username.
- **User Tasks**: This table stores the tasks that users set up in Dynamics GP. These are the tasks that are displayed on the users' home screens in GP.
- **Company Master**: This stores company setup information, such as whether security is enabled, the company ID is in the form of the company database ID in SQL Management Studio, primary address information, tax schedule defaults, and any number of options for the company.
- **Security Setups**: This includes all of the security tasks, security roles, and user security assignments.
- **User-Company Access**: This includes the companies that each user has access to.

Company databases

Each company that we create in Dynamics GP has its own company database. As information such as transactions, accounts, and customer or vendor data is entered through GP, this information is recorded in individual fields. These fields comprise the smallest unit of data stored. All of this data makes up a record, and a record is grouped with similar records and stored in a table.

For obvious reasons, this data is segmented by a company database so that each company can maintain unique records. In addition to this transactional data, numerous additional company setups exist in the company database. As with the System database, we may need to report on some of these company system setups.

The following is a quick reference to the more common company setup tables:

- **Account Formats**: This stores the chart of accounts format for the company.
- **Posting Definitions**: This stores how the individual modules post to the General Ledger.
- **Company Locations**: This lists additional addresses for each company.
- **Source Document Master and Audit Trail Codes**: Every transaction is assigned both a source document and an audit trail code. This table can be used to report on the full description of these codes.
- **Shipping Methods Master**: This stores the setup details of the shipping methods for the company.
- **Payment Terms Master**: This stores the setup details of the payment terms created for the company.
- **Record Notes Master**: This stores all of the record level notes for the particular company.
- **Comment Master**: This stores predefined comments to be used across multiple series in Dynamics GP.
- **Electronic Funds Transfer**: This stores the EFT setup information for the company for both Payables and Receivables modules. This includes customer and vendor banking information.
- **Period Setup**: This stores the fiscal period setup for the company.
- **Sales/Purchases Tax Tables**: These tables store the tax detail and tax schedule records as well as the tax summary amounts.

Dynamics GP table naming/numbering conventions

Dynamics GP has a rather interesting and sometimes challenging table naming structure. When developers or consultants first see this, they are overwhelmed to say the least. Because Dynamics GP is actually a collection of modules, some of which were developed by outside organizations and later assimilated into the core product, we will find that even the standard table naming and numbering do not always apply depending on which module contains the data we need. Nevertheless, by learning the standard structure and naming convention of the core modules, we will notice that it does make some sense. With this knowledge in hand, as well as some of the resources we will cover later in this chapter, we will even have a head start on understanding where data resides in tables underlying non-core GP modules that might not follow the standard naming convention.

Tables versus Table Groups

When data is entered into windows via the Microsoft Dynamics GP application, that data is stored in tables in the underlying SQL database. In most cases, data entered via a single process can be stored in two or more tables. In such cases, it is common for these tables to be grouped together by a certain naming convention. For example, entering journal entry information may update the Transactions Work table (contains General Ledger transaction header information), the Transaction Amounts Work table (contains the General Ledger transaction distributions), and the Transaction Clearing Amounts Work table (contains the General Ledger clearing transactions distributions). These make up what are called **Table Groups**. These table groups are also referred to as logical tables.

Each Microsoft Dynamics GP table has three names:

- Technical name
- Display name
- Physical name

The technical name is used solely by the software and will usually be seen in some alert messages instead of the display name. The display name is the name that will appear in most of the alert messages generated by the system, and is typically the name used for a given table when referring to it in speech, for example, **Vendor Master**. The physical name is the name that will be found in the SQL database when looking in Microsoft SQL Server Management Studio.

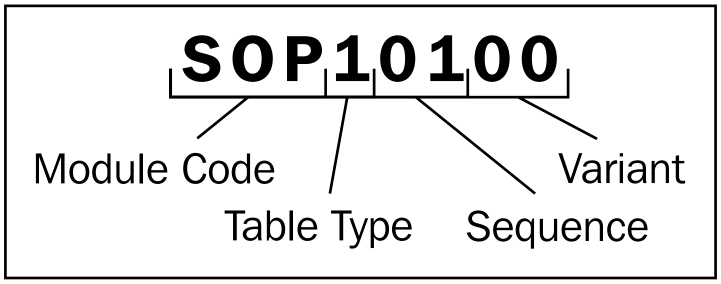
Physical table naming/numbering conventions

When working within the context of a SQL database to generate reports, developers and consultants will make use of the table physical names. A quick scan through the various tables in a standard GP install reveals a bewildering array of table numbers. How can there possibly be any rhyme or reason to these table names? Surprisingly, it does actually follow a certain pattern. For the most part, the table numbering follows a special convention. This schema packs a lot of information into a small number, and it can help developers and consultants know where to begin looking for their data.

As Dynamics GP has grown into a more comprehensive accounting solution, it has expanded, in part, by incorporating third-party applications into the solution. While many of the third-party programmers tried to stick within the relative bounds of the GP physical table naming conventions, as we will soon see, this is not always the case. So, while many of the tables that belong to the core GP modules maintain a fairly standard numbering convention, we will find that this does not hold true for all GP modules.

Generally speaking, GP table physical names contain a two or three digit alpha prefix followed by a five digit number. The three digit prefix represents the module for which the table holds data. The numbers that follow identify what type of data is held in the table. For example, is it posted transaction data? Or is it information related to the module setup?

As we see in the following figure and the following sections of this chapter, the numbering schema for a Dynamics GP physical table can be broken down to reveal information about the kind of data that is found in that table.



Alpha code

Let's begin by taking a look at the alpha-prefix for these tables. We have put together a handy reference of some of the most common prefixes and the modules that they represent:

Prefix	Module	Prefix	Module
AA	Analytical Accounting	MRP	Material Requirements Planning
AF	Advanced Financials	MXLS	Audit Trails/Electronic Signatures
ASI	SmartList	NLB	Navigation List Builder
BM	Bill of Materials (Mfg)	OC	Sales Configurator (Mfg)
CFM	Cash Flow Management	OSRC	Outsourcing (Mfg)
CLM	Certification Manager	PA	Project Accounting
CM	Checkbook Master	PDK	Personal Data Keeper (Proj. Acct.)
CN	Collections Management	PM	Payables Management
CP	Capacity Requirements Planning (Mfg)	POP	Purchase Order Processing
DD	Direct Deposit	PP	Revenue Expense Deferrals
EC	Engineering Change Management (Mfg)	QA	Quality Assurance (Mfg)
ERB	Excel Report Builder	RM	Receivables Management

Prefix	Module	Prefix	Module
EXT	Extender	RT	Routings (Mfg)
FA	Fixed Assets	SC	Sales Forecasting (Mfg)
GL	General Ledger	SLB	SmartList Builder
HR	Human Resources	SOP	Sales Order Processing
ICJC	Job Costing (Mfg)	SVC	Field Service
IV	Inventory	SY	System/Company Setup
IVC	Invoicing (Sales)	UPR	Payroll
MC	Multicurrency	VAT	Intrastat
ME	Electronic Reconcile (EFT)	WC	Work Centers (Mfg)
MOP	Manufacturing Order Processing	WO	Manufacturing Orders

As we can see from the earlier list, in some modules, tables do not share a single common prefix. For example, in the Manufacturing module, tables are broken down even further with Routing tables represented by one set of digits while Material Requirements Planning data is found in tables represented by another set of digits. Other modules use a similar alpha code prefix for all tables in the module. Consider, for example, the Project Accounting module where all project transactions, billing, and revenue recognition tables are represented with the same alpha code. As we said earlier, not all modules will follow the standard naming convention, and this is no different when it comes to alpha prefixes for table names. With experience, we will come to learn which modules have a consistent alpha prefix and which ones are broken down even further into more granular prefixes.

Table type

In addition to knowing the module in which our data is stored, we must also know a bit about the kind of data which we are looking for. Let's take a look at the various types of data that exist in GP tables:

Setup Tables

Almost all modules in GP have setup windows that allow users to define default options or other settings for how that module will be used. The options selected in these windows can be found in the Setup tables.

Master Tables

Some modules, such as Payables Management, allow users to record master records. These master records represent permanent records, such as vendors, for the company. Typically, master records must be entered prior to using a module as transactions will utilize these master records.

Transaction Tables

These tables contain the transaction-level data from Dynamics GP. These range from the most basic of GP transactions, the journal entry, to transactions entered in sub-ledgers such as the distribution records of a posted Receivables invoice. Transactions entered in various modules will, depending on their status, be stored in one of three types of transaction tables. The three transaction tables are as follows:

- **Work**: Unposted transactions can generally be found in work tables. The name is appropriate as these transactions can be considered as "work-in-progress". They have not been committed to the sub or General Ledgers via posting processes, so we should factor this into our thinking when deciding whether or not to include these records in our reports.
- **Open**: Generally speaking, records in these tables have been posted, but are awaiting an additional action before they can be considered "closed" or "history". In the General Ledger module, the open table represents all journal entries for the current open year. In other modules, such as Receivables Management, open tables contain data for receivable transactions that have not yet been fully applied.
- **History**: Transactions that are "completed" generally end up in the history tables. Again, this depends on the module. For example, in Payables Management, fully applied invoices are moved to history. In Purchase Order Processing, however, a routine exists to move completed purchase orders to history. Until this routine is run, records will not be moved to history.

Cross Reference Tables

Some tables represent data that spans multiple modules. For example, GP users can link purchase orders to unfulfilled sales order line items via Sales Order Commitment. The prefix of the table that contains these links indicates that this is a Sales Order Processing table, but in actuality, it contains data from Purchase Order Processing as well.

Other Table types

In addition to these main table types, other table types exist that are less commonly used for reporting purposes. Nonetheless, it is helpful to understand what these table types contain. These less commonly used table types are as follows:

- **Report Options**: Before users can generate a report from the Reports menu, a series of options must be designated for that report. These report options are recorded in a series of report options tables.

- **Temp**: As the name implies, data is only stored in these tables temporarily. Temporary tables can be used in a variety of situations, such as when a user clicks on the **Post** button for a transaction. Although rare, it may be necessary to use a temp table when designing a report that should contain data from the time of posting. For example, a sales invoice can be generated at the time the user posts the invoice and can be based on data stored in a temp table at the time.

Identifying the Table type by the table naming convention

In terms of the physical naming convention for GP tables, the table type is represented by the first digit following the module code. The following table contains the various table types and their associated number in the numbering convention:

Table Type	Number
Master	0
Work	1
Open	2
History	3
Setup	4
Temporary	5
Cross Reference	6
Report Options	7

As we've stressed already throughout this chapter, the numbering scheme used to identify table the type will work fairly well for most modules. Not all modules utilize all table types, so we should not expect to see this consistency among all modules. For example, in the Sales module, sales transactions remain in Work tables (such as SOP10100 and SOP10200) until they are posted, at which point they move to History tables (such as SOP30100 and SOP30200).

Sequence

The next two digits in the numbering convention make up the sequence number. This number indicates the logical table to which the table belongs. As we discussed earlier in the chapter, logical tables are related tables, or table groups, that share similar data. Not only do these table groups share similar data, but they also share the same data type and sequence number when it comes to the physical table numbering convention.

For example, let's consider the following set of logical tables:

- `PM00200` (Vendor Master)
- `PM00201` (Vendor Master Summary)
- `PM00202` (Vendor Master Period Summary)
- `PM00203` (Vendor Accounts)
- `PM00204` (Purchasing 1099 Detail)

These tables comprise the Payables Vendor Master Logical File table group. We can easily see this by the numbers that follow the module code. First, we see that these tables share the same data type—remember, 0 means these are Master tables—and second, we see that these tables share the same sequence number. Although we need other tools to help us determine the name of the table group, we are able to easily scan through a list of table physical names in SQL Management Studio and see that these tables are in the same table group.

Variant

The final two digits of the physical numbering convention represent the logical group variant. Within a logical group, numbers are incremented sequentially. This is evident in our example using the Payables Vendor Master Logical File we just saw, as we see the final two digits of each table increment by one.

In table groups related to transactions, the variant often distinguishes between header tables, detail tables, distribution tables, and other related tables.

Keep in mind that what we have discussed is only a general naming and numbering convention for GP tables in their SQL databases. Unfortunately, as we've already illustrated in earlier sections, these conventions do not hold true in all cases! At the very least, knowing this convention can point us in the right direction. We can rely on other tools and resources to help us pinpoint the right table when these conventions fall short. In just a few moments, we will look at some of the tools and resources that can help us find the right table.

Locating Dynamics GP data using the Resource Descriptions windows

One of the most useful tools for locating data when being tasked with writing reports against Dynamics GP data is the Resource Descriptions tool within the application itself. Resource Descriptions are broken into three distinct windows: **Tables**, **Windows**, and **Fields**. Typically, we find ourselves using a combination of these three windows to locate the specific data we are looking for. There is not really a right or a wrong way to use these windows. Each report developer or consultant could argue that the way he or she uses the windows is the correct way; but ultimately, as long as we are able to identify and find the data we need, we will be that much closer to creating an accurate report that fills our users' needs.

The various **Resource Descriptions** windows are located by navigating to **Microsoft Dynamics GP | Tools | Resource Descriptions**.

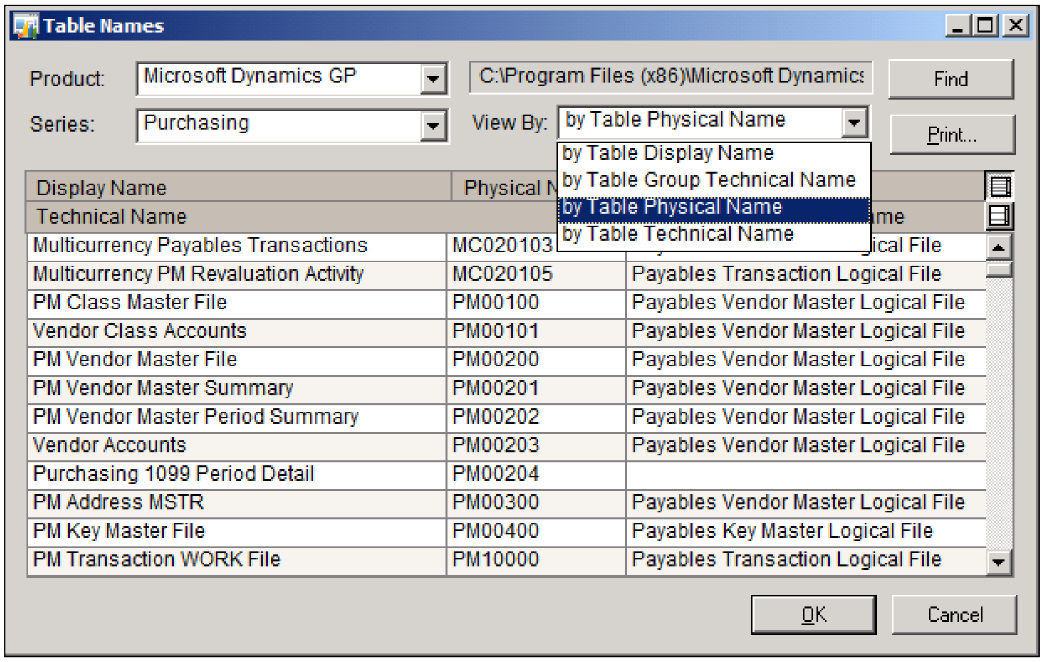
Tables

Table Descriptions let us select a **Product** (Microsoft Dynamics GP, Fixed Assets, and so on) as well as the **Series** (Financial, Sales, and so on). We are then provided with a list of all the tables for the selected product and series. By default, the list of tables is by **Display Name**. We do have the option to change how the tables are sorted by changing the **View By** field. The options are **Table Display Name**, **Table Group Technical Name**, **Table Physical Name**, or **Table Technical Name**. The method we choose will depend on how familiar we are with the table structure. For example, we might select **Table Physical Name** once we are familiar with the naming convention guidelines provided earlier in this chapter. There is also a **Find** button that allows us to search for the table based on any one of its three names.

To access the Table Descriptions, follow these steps:

1. Open Microsoft Dynamics GP.
2. Click on the **Microsoft Dynamics GP** Menu from the toolbar.
3. Select **Tools**.
4. Select **Resource Descriptions**.
5. Select **Tables**.

6. Click on the **Ellipses** button and find your table. In the following screenshot, we are searching for the **PM Vendor Master File** table, which is found under the **Microsoft Dynamics GP Product** and **Purchasing Series**.



Product: Microsoft Dynamics GP
Series: Purchasing
C:\Program Files (x86)\Microsoft Dynamics
View By: by Table Physical Name

Display Name / Technical Name	Physical Name	Group Technical Name
Multicurrency Payables Transactions	MC020103	ical File
Multicurrency PM Revaluation Activity	MC020105	Payables Transaction Logical File
PM Class Master File	PM00100	Payables Vendor Master Logical File
Vendor Class Accounts	PM00101	Payables Vendor Master Logical File
PM Vendor Master File	PM00200	Payables Vendor Master Logical File
PM Vendor Master Summary	PM00201	Payables Vendor Master Logical File
PM Vendor Master Period Summary	PM00202	Payables Vendor Master Logical File
Vendor Accounts	PM00203	Payables Vendor Master Logical File
Purchasing 1099 Period Detail	PM00204	
PM Address MSTR	PM00300	Payables Vendor Master Logical File
PM Key Master File	PM00400	Payables Key Master Logical File
PM Transaction WORK File	PM10000	Payables Transaction Logical File

Once we have found the table we are looking for, we can drill into that table and get additional information for that table. This view includes all of the fields in the selected table, their physical names (as would be seen in the SQL tables), the storage type of the field, and the position.

We can access this additional information by double clicking on the table record in the **Table Names** window. This opens a new window called **Table Descriptions**; as shown in the following screenshot:

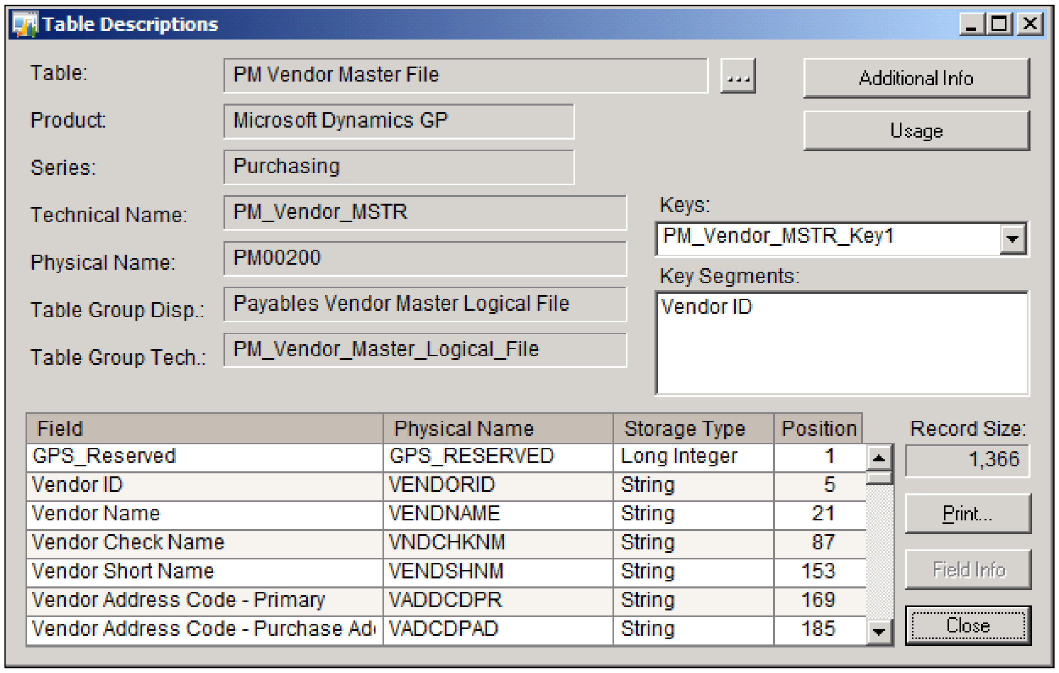
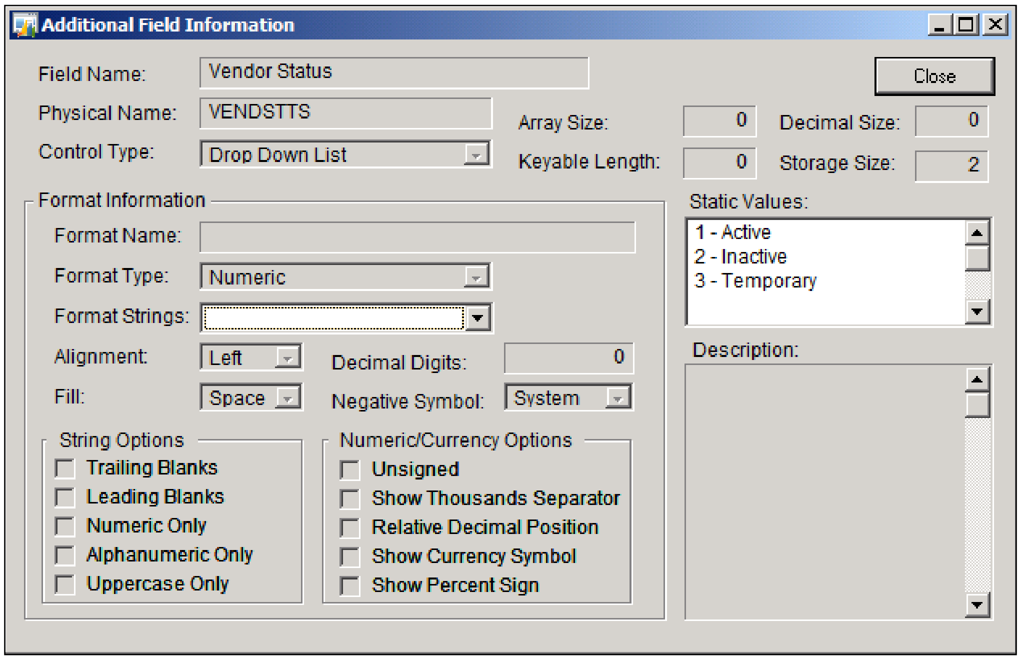


Table:	PM Vendor Master File
Product:	Microsoft Dynamics GP
Series:	Purchasing
Technical Name:	PM_Vendor_MSTR
Physical Name:	PM00200
Table Group Disp.:	Payables Vendor Master Logical File
Table Group Tech.:	PM_Vendor_Master_Logical_File
Keys:	PM_Vendor_MSTR_Key1
Key Segments:	Vendor ID
Record Size:	1,366

Field	Physical Name	Storage Type	Position
GPS_Reserved	GPS_RESERVED	Long Integer	1
Vendor ID	VENDORID	String	5
Vendor Name	VENDNAME	String	21
Vendor Check Name	VNDCHKNM	String	87
Vendor Short Name	VENDSHNM	String	153
Vendor Address Code - Primary	VADDCDPR	String	169
Vendor Address Code - Purchase Ad	VADCDPAD	String	185

From the **Table Descriptions** window, we can further drill into each field and get information such as the **Format Type**, the **Keyable Length** of the field, and if applicable, any **Static Values** if it is a dropdown field.

We can access this detail by double-clicking on a field, or selecting the field and clicking on the **Field Info** button to open the **Additional Field Information** window, as shown in the following screenshot:


Additional Field Information

Field Name:	Vendor Status				
Physical Name:	VENDSTTS	Array Size:	0	Decimal Size:	0
Control Type:	Drop Down List	Keyable Length:	0	Storage Size:	2

Format Information

Format Name:			
Format Type:	Numeric		
Format Strings:			
Alignment:	Left	Decimal Digits:	0
Fill:	Space	Negative Symbol:	System

String Options
- [ ] Trailing Blanks
- [ ] Leading Blanks
- [ ] Numeric Only
- [ ] Alphanumeric Only
- [ ] Uppercase Only

Numeric/Currency Options
- [ ] Unsigned
- [ ] Show Thousands Separator
- [ ] Relative Decimal Position
- [ ] Show Currency Symbol
- [ ] Show Percent Sign

Static Values:
1 - Active
2 - Inactive
3 - Temporary

Description:

Returning to the **Table Descriptions** window, the last pieces of information we can access from this window are additional info such as any secondary tables, the secondary tables' keys, what the related fields are for the tables, and what the relationship type is. We can also see the usage, which gives us all the forms and reports that use this table.

Most of this information can be found by returning to the **Table Descriptions** window and clicking on the **Additional Info** button. This opens the **Additional Table Information** window as seen in the following screenshot:

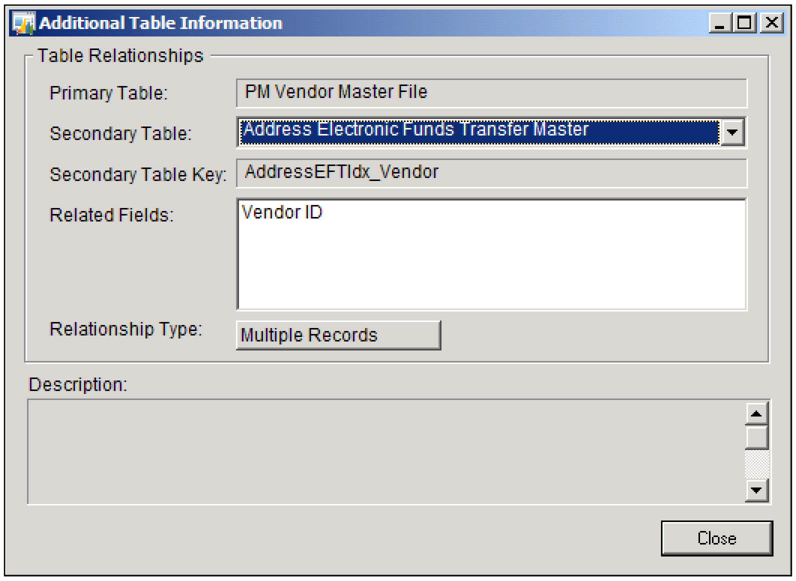


Table Relationships

Primary Table:	PM Vendor Master File
Secondary Table:	Address Electronic Funds Transfer Master
Secondary Table Key:	AddressEFTIdx_Vendor
Related Fields:	Vendor ID
Relationship Type:	Multiple Records

Description:

Also, by clicking on the Usage button from the **Table Descriptions** window seen in an earlier screenshot, we can access the **Table Usage** window as seen the following screenshot by clicking on the **Usage** button:

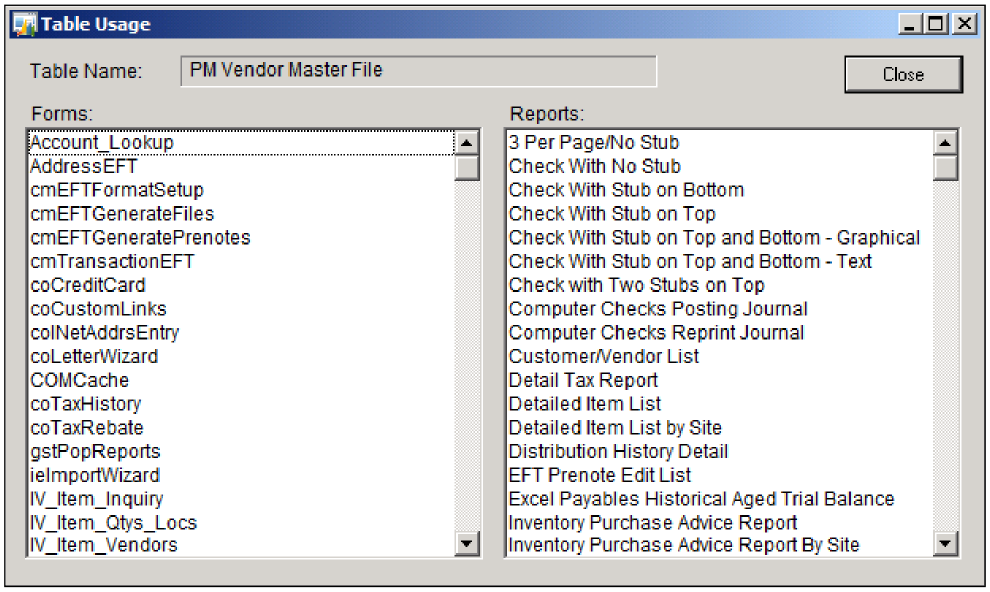


Table Name: PM Vendor Master File

Forms:	Reports:
Account_Lookup	3 Per Page/No Stub
AddressEFT	Check With No Stub
cmEFTFormatSetup	Check With Stub on Bottom
cmEFTGenerateFiles	Check With Stub on Top
cmEFTGeneratePrenotes	Check With Stub on Top and Bottom - Graphical
cmTransactionEFT	Check With Stub on Top and Bottom - Text
coCreditCard	Check with Two Stubs on Top
coCustomLinks	Computer Checks Posting Journal
coINetAddrsEntry	Computer Checks Reprint Journal
coLetterWizard	Customer/Vendor List
COMCache	Detail Tax Report
coTaxHistory	Detailed Item List
coTaxRebate	Detailed Item List by Site
gstPopReports	Distribution History Detail
ieImportWizard	EFT Prenote Edit List
IV_Item_Inquiry	Excel Payables Historical Aged Trial Balance
IV_Item_Qtys_Locs	Inventory Purchase Advice Report
IV_Item_Vendors	Inventory Purchase Advice Report By Site

Typically, we use **Table Descriptions** when we know the name of the table needed and we need to find additional information about that table.

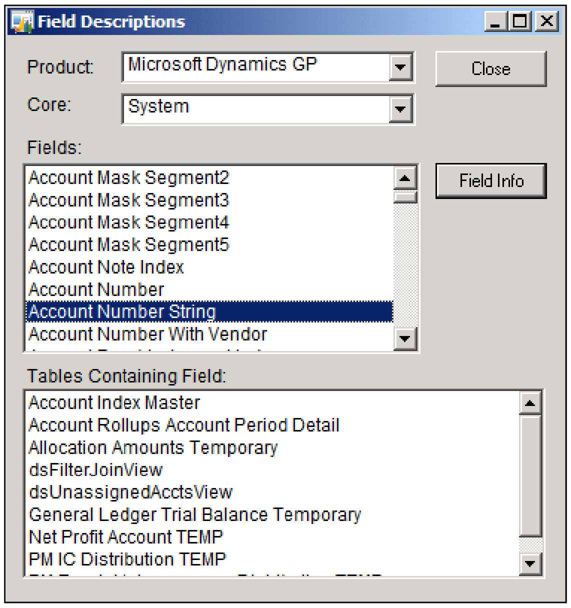
Fields

The **Field Descriptions** window gives us the ability to again select both the **Product** and the **Core** (Series), which contains the field we are looking for. Once both of these fields are selected, we are provided with the field list. Similar to how we could look at field information in **Table Descriptions**, we can look at field information here as well.

To access **Field Descriptions**, follow these steps:

1. Open Microsoft Dynamics GP.
2. Click on the **Microsoft Dynamics GP** Menu from the toolbar.
3. Select **Tools**.
4. Select **Resource Descriptions.**
5. Select **Fields**.
6. Select a **Product** and a **Core**.

For example, in the following screenshot, **Microsoft Dynamics GP** has been selected as the **Product,** and **System** has been selected as the **Core** value. This shows us a list of fields that meet this criteria and a list of the tables that contain the selected field:



Product: Microsoft Dynamics GP
Core: System
Fields:
Account Mask Segment2
Account Mask Segment3
Account Mask Segment4
Account Mask Segment5
Account Note Index
Account Number
Account Number String
Account Number With Vendor

Tables Containing Field:
Account Index Master
Account Rollups Account Period Detail
Allocation Amounts Temporary
dsFilterJoinView
dsUnassignedAcctsView
General Ledger Trial Balance Temporary
Net Profit Account TEMP
PM IC Distribution TEMP

The one added benefit of using this window over others accessible via **Resource Descriptions** is that once we select the field we need, a list will be provided showing us all the tables that contain that field.

This window is typically used when we are not sure exactly which table(s) a requested field resides in. We can use this window to tell us all the table display names that contain the selected field, then we can go to **Table Descriptions** and find the physical (SQL) table name based on the results returned in the **Field Descriptions** window.

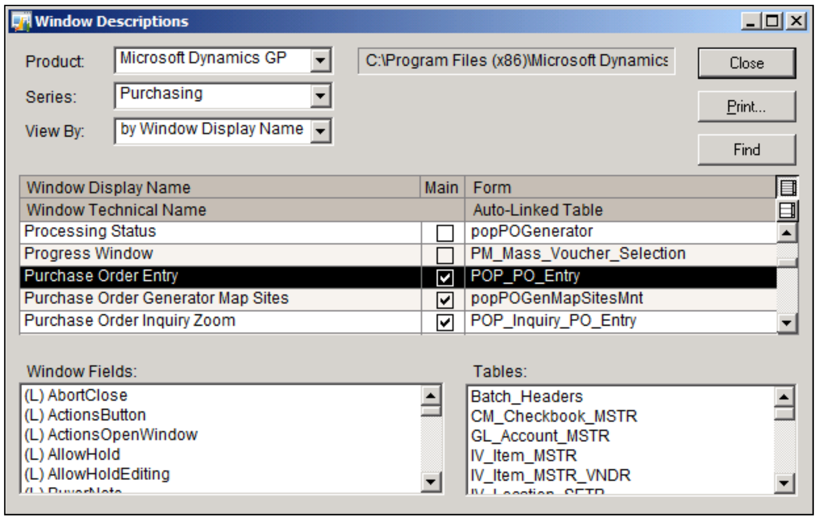
Windows

Windows Descriptions provides us with the ability to view the windows where the data is being recorded or viewed. As with the other **Resource Description** windows, we select the **Product** and **Series**. **Products** relate to the application code dictionaries, that is, Microsoft Dynamics GP, Fixed Assets, and so on. **Series** relate to the series the modules reside in, for instance, General Ledger and Bank Reconciliation reside in the Financial Series. Once we locate the window we are looking for, Purchase Order Entry for example, we will be provided with a list of both the fields in that window and the tables used by the window. We can also use the **Form** name in this window to find the related fields and tables.

To access **Window Descriptions**, follow these steps:

1. Open Microsoft Dynamics GP.
2. Click on the **Microsoft Dynamics GP** Menu from the toolbar.
3. Select **Tools**.
4. Select **Resource Descriptions**.
5. Select **Windows**.
6. Select a **Product**, **Series**, and **View By** from their dropdown lists.

As the following screenshot shows, selecting **Microsoft Dynamics GP** as the **Product**, **Purchasing** as the **Series**, and **by Window Display Name** as the **View By** option presents us with a list of all windows available in this unique combination:



Window Descriptions

Product: Microsoft Dynamics GP

Series: Purchasing

View By: by Window Display Name

C:\Program Files (x86)\Microsoft Dynamics

Window Display Name / Window Technical Name	Main	Form / Auto-Linked Table
Processing Status	☐	popPOGenerator
Progress Window	☐	PM_Mass_Voucher_Selection
Purchase Order Entry	☑	POP_PO_Entry
Purchase Order Generator Map Sites	☑	popPOGenMapSitesMnt
Purchase Order Inquiry Zoom	☑	POP_Inquiry_PO_Entry

Window Fields:
(L) AbortClose
(L) ActionsButton
(L) ActionsOpenWindow
(L) AllowHold
(L) AllowHoldEditing

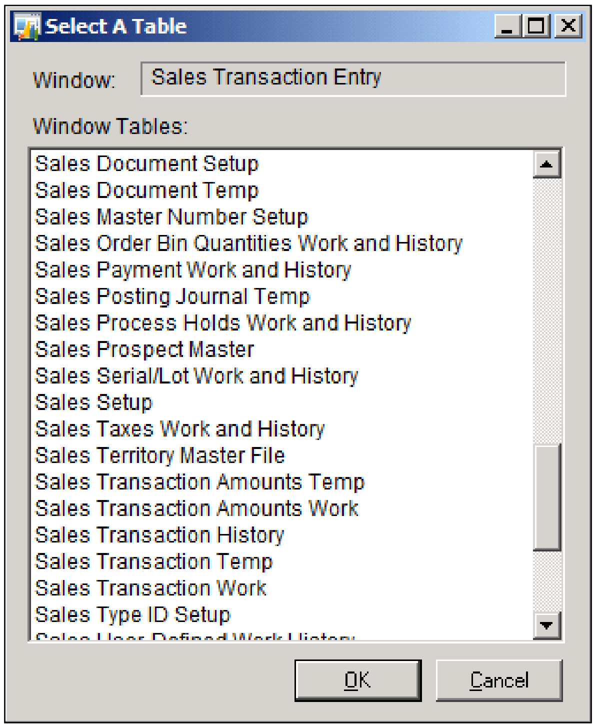
Tables:
Batch_Headers
CM_Checkbook_MSTR
GL_Account_MSTR
IV_Item_MSTR
IV_Item_MSTR_VNDR

We typically use **Window Descriptions** when we know where the requested information is being entered or displayed in Dynamics GP but we are unsure of the field name(s) or the table name(s) used to store the data. As with **Field Descriptions**, once we have a list of possibilities, we can cross-reference to **Table Descriptions**.

The **Table Descriptions** window is a bit constrained in size, and when dealing with a **Product** and **Series** with numerous tables, it can often be time-consuming to scroll through the list to find the right window. Don't neglect the **Find** button in the upper-right hand corner of this window. This works well, assuming we have already made our **Product** and **Series** selection in the appropriate drop-down boxes. After selecting this button to open the **Find** window, we can type in the **Display Name** we are looking for and the window will auto-focus on this row in the scrolling window. Remember, the **Display Name** corresponds to the name of the window as it appears in the menu bar attached at the top of the window in GP.

The Table Import utility

Another very useful resource for finding the data that we need to begin writing our reports is the **Table Import** utility. From any window in Dynamics GP, we can open the table import utility and be provided with all the friendly table display names that are used by this window. For example, opening the **Table Import** window from the **Sales Transaction Entry** window yields the following information:


Select A Table

Window: Sales Transaction Entry

Window Tables:

Sales Document Setup
Sales Document Temp
Sales Master Number Setup
Sales Order Bin Quantities Work and History
Sales Payment Work and History
Sales Posting Journal Temp
Sales Process Holds Work and History
Sales Prospect Master
Sales Serial/Lot Work and History
Sales Setup
Sales Taxes Work and History
Sales Territory Master File
Sales Transaction Amounts Temp
Sales Transaction Amounts Work
Sales Transaction History
Sales Transaction Temp
Sales Transaction Work
Sales Type ID Setup

To access the **Table Import** from any window, follow these steps:

1. Open Microsoft Dynamics GP.
2. Open the Dynamics GP Window you are looking for.
3. Select **Tools**.
4. Select **Integrate**.
5. Select **Table Import**.

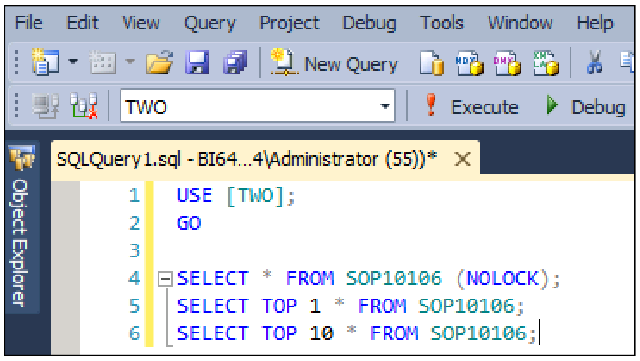
Once we know which table we want to go find our data in, we can again cross-reference the Table Resource descriptions to find the physical SQL table name. This utility is very useful if the user tells you that they need a report that includes information they are entering on a particular screen.

Accessing data at the table level using SQL Management Studio

Once we have identified our physical table name, we can begin to use SQL Management Studio to view the data in the table. To accomplish this, we can write simple select statements to view the data. This can be very useful, as it allows us to see actual data in the underlying table. So, if a user tells us they need information they are entering in Sales User defined fields, we can view that data directly in the table to be sure we are capturing the requested data. A helpful tip for this is to open the **Sales User-Defined Fields Entry** window in Dynamics GP for a specific document number. Then, in SQL Management Studio, we can select the record from the table (in this case, the `SOP10106` table) that equals the same document number we are looking at in the screen. This allows us to ensure we are looking at the correct data.

One thing to keep in mind if we are working on a production database or even a test database with large amounts of data is that we can use commands, such as **NOLOCK**, **TOP 1**, **TOP 10**, to keep us from putting inadvertent locks on the table and to limit the amount of data returned by our queries. We don't necessarily need to select all records from a Sales Transaction table to see the data in that table and begin writing our report. It is much more efficient to use small selections of data at this point in our data gathering.

A sample of these TSQL statements can be seen in the following screenshot:


File Edit View Query Project Debug Tools Window Help
New Query
TWO
Execute
Debug
SQLQuery1.sql - BI64...4\Administrator (55))*
Object Explorer

```
USE [TWO];
GO

SELECT * FROM SOP10106 (NOLOCK);
SELECT TOP 1 * FROM SOP10106;
SELECT TOP 10 * FROM SOP10106;
```

In SQL Management Studio, there is another very useful command that we have at our disposal. This command is called SP help and is very easy to use. We simply open a new query window and type in `sp_help` and our table name (`sp_help SOP10100`). This will return information such as the owner of the table and the created data, along with all of the columns in the table, their names, types, whether they are computed, their length, and if they are nullable amongst other things. We will also be provided with the indexes on the table. All of this information can be helpful to us when we begin writing our reports.

Locating Dynamics GP data with additional tools

Earlier in this chapter, we discussed a logical way for finding data for our report by starting at the database level and working our way down to the field level within the SQL database tables. While this approach utilized the GP naming and numbering convention, as well as the Resource Descriptions and Table Import tools within Dynamics GP, several other tools exist outside of the standard GP application that can help us better understand where our data might be located.

Dynamics GP 2013 Software Development Kit

One useful tool for report writers and developers to use when trying to determine which tables should be used in building a report is the Software Development Kit (SDK), for short. Unlike the Resource Description windows, this tool resides outside of Dynamics GP and requires a separate download. This means that you don't need a workstation installation of Dynamics GP to use this tool, nor does it require you to consume a GP user license when accessing the information it contains.

Downloading the Software Development Kit

New releases of the SDK are timed to coincide with new releases of the Dynamics GP application. For Dynamics GP 2013, the SDK can be installed from the original Dynamics GP 2013 CD software. Look under the **Tools** folder of the `install media` folder for the core GP and other SDK installation executables.

On PartnerSource, however, several additional SDKs can usually be found, many of which contain information related to additional products and tools that can be integrated with the core Dynamics GP application. Although the GP 2013 Product Release page has not been updated with any of these additional SDKs at the time of writing this many of them are still available on the GP 2010 Product Release page. For example, users can find SDKs for Business Portal 5.0 for GP 2010, as well as for eConnect for GP 2010 on the GP 2010 Product Releases page. Our focus will be on the SDK designed for the core application.

Run the installation executable to begin the SDK installation. Accepting defaults will install the program so that it can be opened from the Microsoft Dynamics folder under the **Start** menu.

Using the Software Development Kit

The Software Development Kit is broken up into several sections containing information related to customizing Dynamics GP 2013 and working with the various database objects that exist in each company's SQL database. Among these sections, we will find the following:

- **Table Integration**, **Database Diagrams**: This section provides diagrams of tables related to the selected module. While information about specific fields is not found in this diagram, this documentation can be helpful in displaying a list of tables related to a specific module.
- **Table Integration**, **Design Documents**, and **Transaction Flows**: Documents in this section describe how data flows from one table to the next as transactions are conducted in GP.
- **Additional Products**: Additional modules, such as Manufacturing and Project Accounting, are not included with the information described in the preceding sections. Check this section to find documentation describing the tables and fields related to these modules.

We can use the SDK when we want to get a better understanding of how data flows through the various tables. Data in GP flows through many SQL tables as transactions are conducted in the overlying GP application. Often, we are asked to develop reports that capture data at a certain point in the transactional process, and we can use the SDK to help us determine where to locate this data.

The Support Debugging Tool

The Support Debugging Tool is another valuable tool to have in our toolbox as we seek out our data in GP and its accompanying SQL databases. Over the last few versions of Dynamics GP, the Support Debugging Tool has quickly become the go-to tool for IT departments and Help Desks for debugging GP issues. Although the Support Debugging tool offers a wide range of functionality designed for debugging issues, we can use several components of the Support Debugging Tool during the report writing and development process.

Downloading the Support Debugging Tool

Although the Support Debugging Tool has been around for several iterations, the latest version, Build 17, was timed to coincide with the release of Dynamics GP 2013. This tool, primarily developed by David Musgrave of the Microsoft Dynamics GP Asia Pacific Support team, is a free tool but it can only be downloaded by Microsoft Partners via PartnerSource.

For a list of resources and links related to the Support Debugging Tool, check out the "Support Debugging Tool Portal" at the following link: `http://blogs.msdn.com/b/developingfordynamicsgp/archive/2009/08/07/support-debugging-tool.aspx`

After downloading the tool, the install process requires users to copy and paste a chunk (`.cnk`) file (which is a self-extracting data dictionary file that is used to distribute customizations and third party products) to the Dynamics GP root directory (which is usually located at `C:\Program Files (x86)\Microsoft Dynamics\GP2013`), launching Dynamics GP, and selecting to include the new code when prompted. Upon installation, the 'sa' user will need to log into GP 2013 and grant the appropriate security rights as only members of POWERUSER will have access by default. After this, the tool is accessible from the GP application via the **Tools** menu.

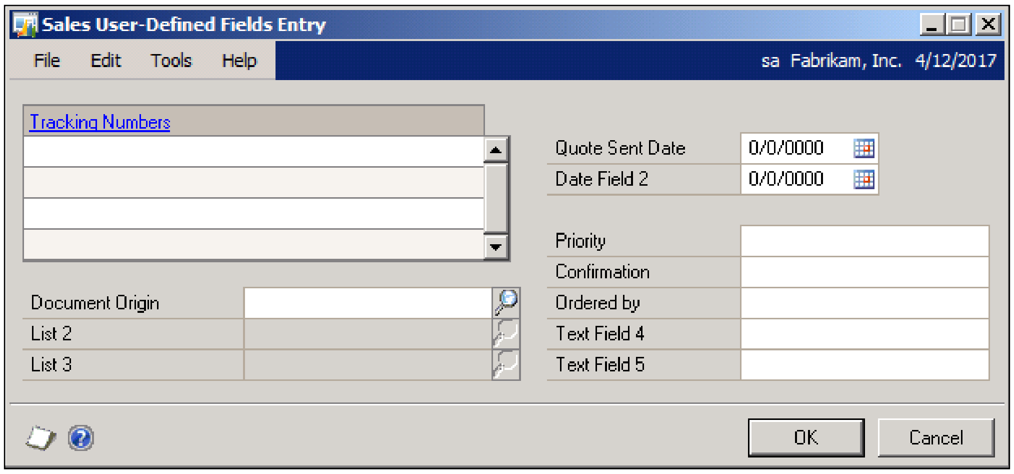
Using the Support Debugging Tool

The primary Support Debugging Tool component that we will explore is the **Resource Information** window. This can be accessed from the Support Debugging Tool by navigating to **Options | Resource Information** from the navigation menus. This functionality is an extended version of the **Resource Descriptions** window that we covered earlier in this chapter.

Like the **Resource Descriptions** window, this tool provides insight into display, technical, and physical names for all windows, tables, and fields in the Dynamics GP application. Unlike the **Resource Descriptions** window, however, this tool combines three windows (**Window Descriptions**, **Table Descriptions**, and **Field Descriptions**) into one! Additionally, we can come to the **Resource Information** window with limited information and the Support Debugging Tool will fill in the rest for us!

Let's use an example to consider one way in which this tool can be used to find our data.

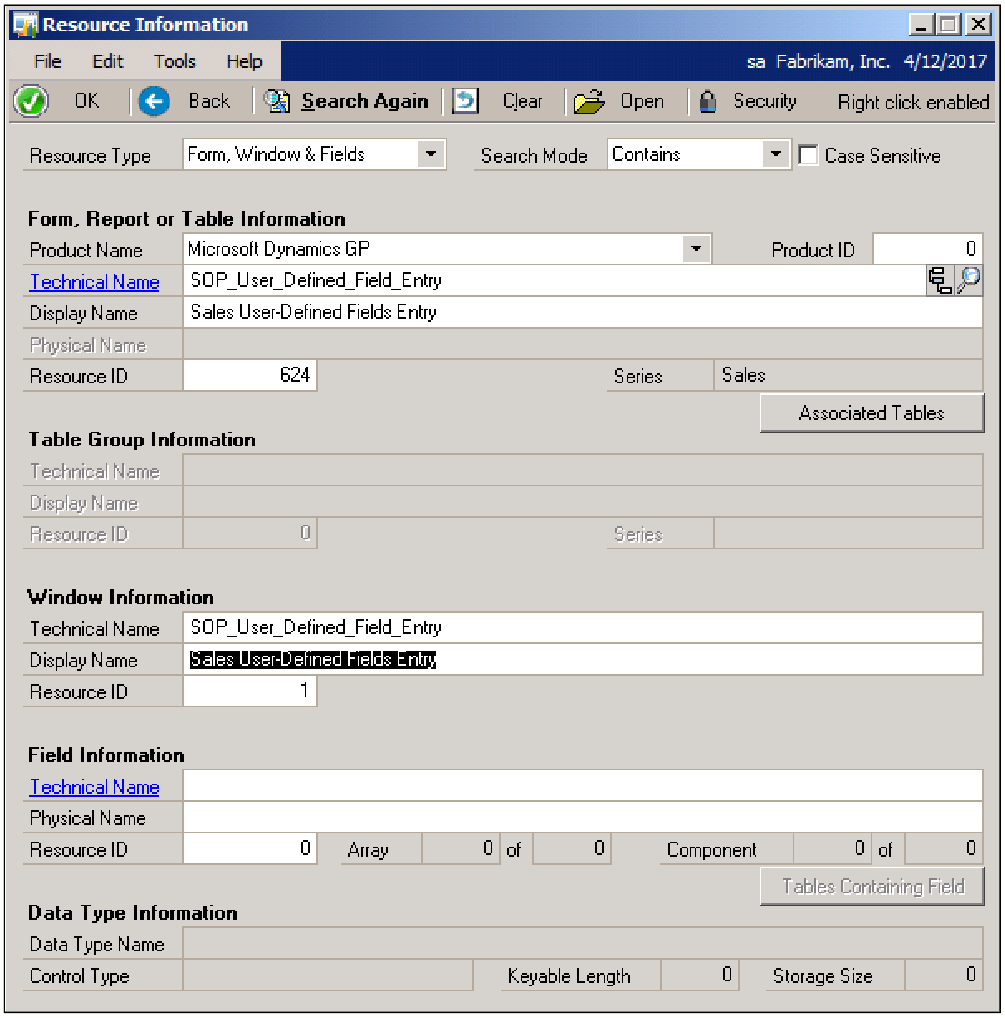
Suppose we want to include the **Text Field 1** field from the **Sales User-Defined Fields Entry** window (**Transactions | Sales | Sales Transaction Entry | User-Defined button**) in our report:



Quote Sent Date	0/0/0000
Date Field 2	0/0/0000
Priority	
Confirmation	
Ordered by	
Text Field 4	
Text Field 5	
Document Origin	
List 2	
List 3	

From the Support Debugging tool, **Debugger** menu, we can open the **Resource Information** window and select **Forms, Windows & Fields** in the **Resource Type** drop down. Near the bottom of the window, we can type the **Display Name** of the window for which we need more information.

Remember, the display name of a window is the name as it appears at the top of the window while it is open in GP. For example, in our earlier screenshot, we see the display name of the window shown in the blue bar at the top of the window. In the following screenshot, we have entered this display name (**Sales User-Defined Fields Entry**) in the **Display Name** field under the **Form, Report or Table Information** heading and are about to tab off the field to reveal additional information about this window:



Resource Type: Form, Window & Fields — Search Mode: Contains — Case Sensitive

Form, Report or Table Information

Product Name	Microsoft Dynamics GP	Product ID	0
Technical Name	SOP_User_Defined_Field_Entry		
Display Name	Sales User-Defined Fields Entry		
Physical Name			
Resource ID	624	Series	Sales

Table Group Information

Technical Name			
Display Name			
Resource ID	0	Series	

Window Information

Technical Name	SOP_User_Defined_Field_Entry
Display Name	Sales User-Defined Fields Entry
Resource ID	1

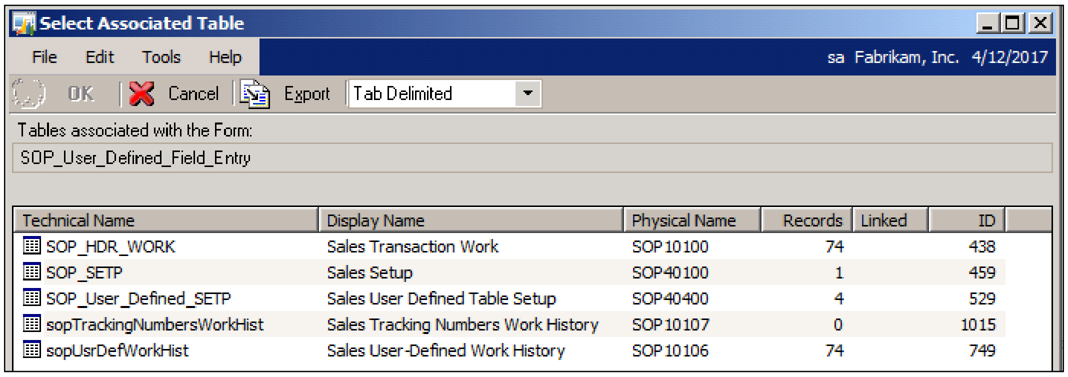
Field Information

Technical Name										
Physical Name										
Resource ID	0	Array	0	of	0	Component	0	of	0	

Data Type Information

Data Type Name					
Control Type		Keyable Length	0	Storage Size	0

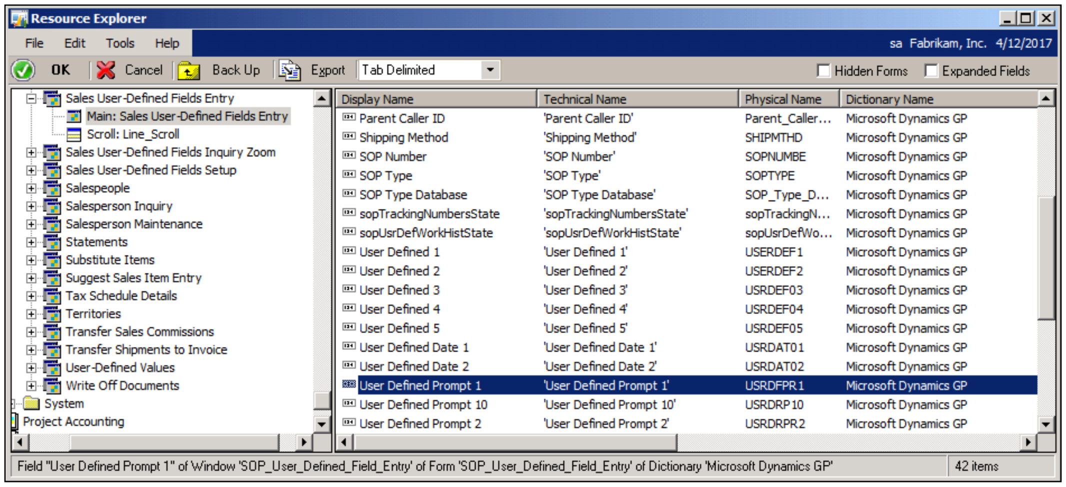
After we tab off of the **Display Name** field, the **Associated Tables** button becomes available and the Support Debugging Tool auto-populates the Technical and Display names for this form in the **Form, Report or Table Information**. Selecting the **Associated Tables** button opens a window displaying information about SQL tables containing data used by this window. This is similar to using the Table Import trick that we discussed earlier in this chapter, but as the following image shows, this method provides more detailed information for the user:



Tables associated with the Form:
SOP_User_Defined_Field_Entry

Technical Name	Display Name	Physical Name	Records	Linked	ID
SOP_HDR_WORK	Sales Transaction Work	SOP10100	74		438
SOP_SETP	Sales Setup	SOP40100	1		459
SOP_User_Defined_SETP	Sales User Defined Table Setup	SOP40400	4		529
sopTrackingNumbersWorkHist	Sales Tracking Numbers Work History	SOP10107	0		1015
sopUsrDefWorkHist	Sales User-Defined Work History	SOP10106	74		749

Back on the **Resource Information** screen, with the technical and display name information displayed, we can select the **Lookup** button beside the **Technical Name** field in the **Form, Report or Table Information** grouping to open the **Resource Explorer** for this form. This allows us to select the **Main** form, and shows us a list of all of the fields in this window. We can scroll through this list until we find the field that corresponds to the one we are looking for. In this case, the name does not exactly match what appears in the window in GP, but **User Defined Prompt 1**, as seen selected in the following image, is close enough to the **User Defined Entry** window and the **Text Field 1** that we know we have found the right field.



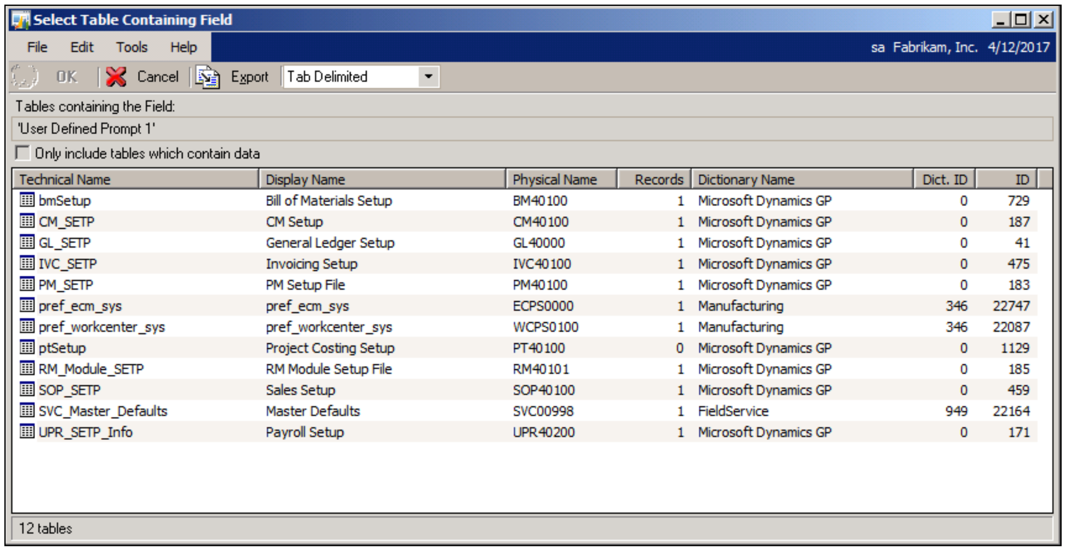
Resource Explorer
File Edit Tools Help
sa Fabrikam, Inc. 4/12/2017
OK Cancel Back Up Export Tab Delimited
Hidden Forms
Expanded Fields

- Sales User-Defined Fields Entry
  - Main: Sales User-Defined Fields Entry
  - Scroll: Line_Scroll
- Sales User-Defined Fields Inquiry Zoom
- Sales User-Defined Fields Setup
- Salespeople
- Salesperson Inquiry
- Salesperson Maintenance
- Statements
- Substitute Items
- Suggest Sales Item Entry
- Tax Schedule Details
- Territories
- Transfer Sales Commissions
- Transfer Shipments to Invoice
- User-Defined Values
- Write Off Documents
- System
- Project Accounting

Display Name	Technical Name	Physical Name	Dictionary Name
Parent Caller ID	'Parent Caller ID'	Parent_Caller…	Microsoft Dynamics GP
Shipping Method	'Shipping Method'	SHIPMTHD	Microsoft Dynamics GP
SOP Number	'SOP Number'	SOPNUMBE	Microsoft Dynamics GP
SOP Type	'SOP Type'	SOPTYPE	Microsoft Dynamics GP
SOP Type Database	'SOP Type Database'	SOP_Type_D…	Microsoft Dynamics GP
sopTrackingNumbersState	'sopTrackingNumbersState'	sopTrackingN…	Microsoft Dynamics GP
sopUsrDefWorkHistState	'sopUsrDefWorkHistState'	sopUsrDefWo…	Microsoft Dynamics GP
User Defined 1	'User Defined 1'	USERDEF1	Microsoft Dynamics GP
User Defined 2	'User Defined 2'	USERDEF2	Microsoft Dynamics GP
User Defined 3	'User Defined 3'	USRDEF03	Microsoft Dynamics GP
User Defined 4	'User Defined 4'	USRDEF04	Microsoft Dynamics GP
User Defined 5	'User Defined 5'	USRDEF05	Microsoft Dynamics GP
User Defined Date 1	'User Defined Date 1'	USRDAT01	Microsoft Dynamics GP
User Defined Date 2	'User Defined Date 2'	USRDAT02	Microsoft Dynamics GP
User Defined Prompt 1	'User Defined Prompt 1'	USRDFPR1	Microsoft Dynamics GP
User Defined Prompt 10	'User Defined Prompt 10'	USRDRP10	Microsoft Dynamics GP
User Defined Prompt 2	'User Defined Prompt 2'	USRDRPR2	Microsoft Dynamics GP

Field "User Defined Prompt 1" of Window 'SOP_User_Defined_Field_Entry' of Form 'SOP_User_Defined_Field_Entry' of Dictionary 'Microsoft Dynamics GP' 42 items

Double-clicking on the line containing our field causes the **Resource Information** window to appear again, this time with the **Field Information** fields populated. Additionally, we notice that the button for **Tables Containing Field** at the bottom right-hand corner of this window can be selected. Selecting this button shows us a list of SQL database tables that contain a field called **USERDEF1**:



Select Table Containing Field
File Edit Tools Help
sa Fabrikam, Inc. 4/12/2017
OK Cancel Export Tab Delimited

Tables containing the Field:
'User Defined Prompt 1'
Only include tables which contain data

Technical Name	Display Name	Physical Name	Records	Dictionary Name	Dict. ID	ID
bmSetup	Bill of Materials Setup	BM40100	1	Microsoft Dynamics GP	0	729
CM_SETP	CM Setup	CM40100	1	Microsoft Dynamics GP	0	187
GL_SETP	General Ledger Setup	GL40000	1	Microsoft Dynamics GP	0	41
IVC_SETP	Invoicing Setup	IVC40100	1	Microsoft Dynamics GP	0	475
PM_SETP	PM Setup File	PM40100	1	Microsoft Dynamics GP	0	183
pref_ecm_sys	pref_ecm_sys	ECPS0000	1	Manufacturing	346	22747
pref_workcenter_sys	pref_workcenter_sys	WCPS0100	1	Manufacturing	346	22087
ptSetup	Project Costing Setup	PT40100	0	Microsoft Dynamics GP	0	1129
RM_Module_SETP	RM Module Setup File	RM40101	1	Microsoft Dynamics GP	0	185
SOP_SETP	Sales Setup	SOP40100	1	Microsoft Dynamics GP	0	459
SVC_Master_Defaults	Master Defaults	SVC00998	1	FieldService	949	22164
UPR_SETP_Info	Payroll Setup	UPR40200	1	Microsoft Dynamics GP	0	171

12 tables

Of course, numerous tables contain a field as generically named as **USERDEF1**, but we can easily scroll through this list and determine that the table that we are probably looking for is the Sales-User Defined Work History table or SOP10106.

By using the Support Debugging Tool we can significantly reduce the amount of time spent searching for our data. The Support Debugging Tool allows us to begin with a limited amount of data as well as it is a great tool to use as we initially set out on our quest for data. If you haven't tried out the Support Debugging Tool yet… what are you waiting for? Check with your Microsoft Partner for more information!

Summary

In this chapter, we discussed both various ways to identify the data we need to create our reports as well as the various tools that assist us with this task. Once the challenges presented in the previous chapter have been identified, the next logical step is finding our data. Keep in mind that although some reporting tools do not require a pre-knowledge of the SQL database or GP company database structure, we can still use our knowledge of GP data structures to build better reports.

To that end, we've used this chapter to explore some of the conventions surrounding how GP data is stored. We began at the database level by exploring the differences between the system and company databases, including the new enhancements to naming system databases in GP 2013. Then we moved on to understanding the naming and numbering convention for tables found within these databases. At first glance, the numbering convention used for GP tables can be confusing, but by this point, we are familiar with some general naming conventions that, can help us in most cases.

Finally, we looked at some tools that can be used for finding this data. Some tools, such as Table Resources, Table Import, and SQL stored procedures can be used with any standard install of GP. Other tools, such as the Software Development Kit for Dynamics GP 2013 and the Support Debugging Tool can also be used with GP data, however they require separate installations.

In the next chapter, we will begin exploring our first reporting tool, SmartList! SmartList is a tool that comes with the standard install of Dynamics GP. We'll also look at additional add-ons, SmartList Builder and Excel Reports Builder. These tools will allow us to extend the concept of SmartLists, and create or customize our own SmartLists and Excel Reports.

3 Working with the Builders – SmartList and Excel Reports

In the last chapter, we explored some methods out of the many methods available to us for locating our data in the ERP system. We began with a look at how the data is stored and then we moved on to some additional tools that can help us understand how data entered in the GP application ends up in the underlying database.

Now, it is finally time to begin writing our new reports. There are many reporting tools available to us in Dynamics GP 2013, and in this chapter, we will be introducing four of the tools that we consider are beginner reporting tools, they are as follows:

- **Default SmartLists**: These SmartLists are included out of the box along with every installation of Dynamics GP
- **Excel Reports**: These reports, which mirror the default SmartLists, must be deployed before they can be used
- **SmartList Builder**: This tool allows users to extend the functionality of the default SmartLists
- **Excel Report Builder**: This tool, like SmartList Builder, allows users to extend the functionality of SmartLists and Excel Reports in an Excel format

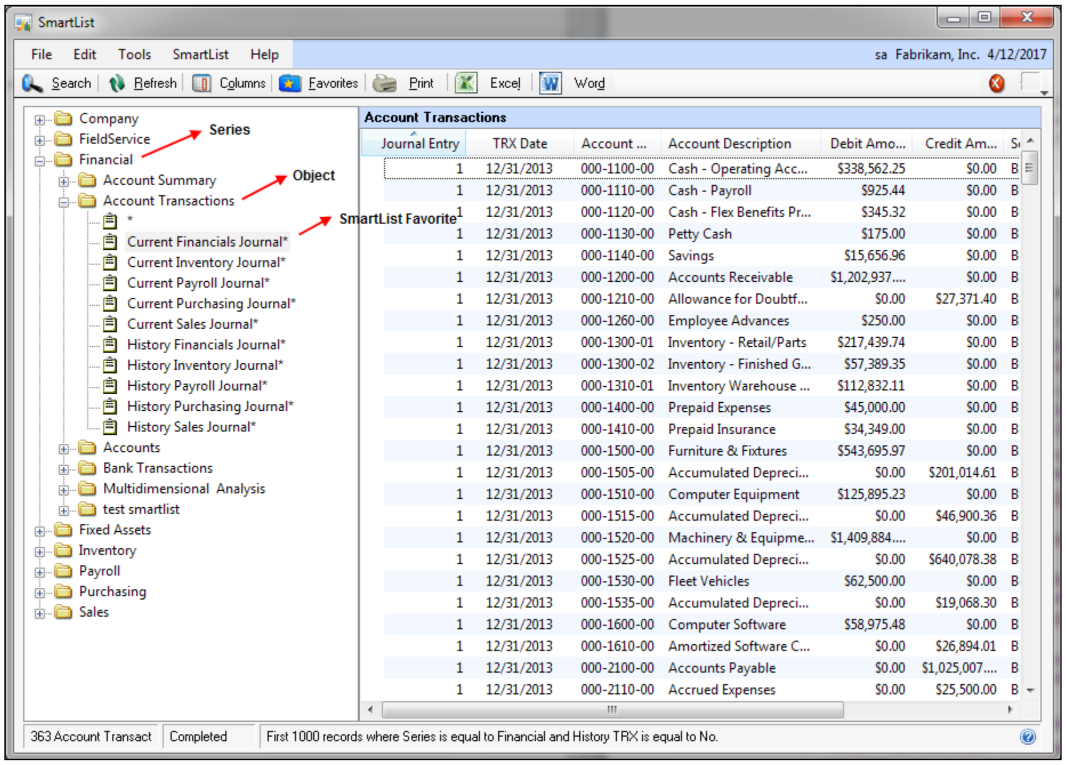
Utilizing default SmartLists to meet basic reporting needs

Before we begin using SmartList to create simple reports, it is important to understand what SmartList is. SmartList is an out of the box module within Dynamics GP that lets developers, consultants, or end users create lists of data within Dynamics GP. As we will see, it is a very easy-to-use tool.

Structure of SmartList

SmartList is laid out in a fairly simple fashion. It is made up of **SmartList Objects** and SmartLists organized by **Series**. Upon opening SmartList and expanding one of the Series folders on the first level, all of the folders we see grouped under this level are referred to as SmartList Objects. We can think of them as a container for similar SmartLists. For instance, the SmartList Object called **Account Transactions** contains a series of SmartLists related to transactions posted to General Ledger. Once we expand one of these objects, we will see any number of **SmartList Favorites** that were either installed with Dynamics GP, identified with an asterisk after the name (*), or were created by users. SmartList Favorites allow us to make minor modifications to a default SmartList and then save our changes so we can simply come back in to SmartList at any time and retrieve a refreshed list of the data without having to set up all our columns and search criteria again.

The easiest way to access a SmartList is to click on the **Microsoft Dynamics GP** menu and select **SmartList**, as seen in the following screenshot:



Journal Entry	TRX Date	Account ...	Account Description	Debit Amo...	Credit Am...	S
1	12/31/2013	000-1100-00	Cash - Operating Acc...	$338,562.25	$0.00	B
1	12/31/2013	000-1110-00	Cash - Payroll	$925.44	$0.00	B
1	12/31/2013	000-1120-00	Cash - Flex Benefits Pr...	$345.32	$0.00	B
1	12/31/2013	000-1130-00	Petty Cash	$175.00	$0.00	B
1	12/31/2013	000-1140-00	Savings	$15,656.96	$0.00	B
1	12/31/2013	000-1200-00	Accounts Receivable	$1,202,937....	$0.00	B
1	12/31/2013	000-1210-00	Allowance for Doubtf...	$0.00	$27,371.40	B
1	12/31/2013	000-1260-00	Employee Advances	$250.00	$0.00	B
1	12/31/2013	000-1300-01	Inventory - Retail/Parts	$217,439.74	$0.00	B
1	12/31/2013	000-1300-02	Inventory - Finished G...	$57,389.35	$0.00	B
1	12/31/2013	000-1310-01	Inventory Warehouse ...	$112,832.11	$0.00	B
1	12/31/2013	000-1400-00	Prepaid Expenses	$45,000.00	$0.00	B
1	12/31/2013	000-1410-00	Prepaid Insurance	$34,349.00	$0.00	B
1	12/31/2013	000-1500-00	Furniture & Fixtures	$543,695.97	$0.00	B
1	12/31/2013	000-1505-00	Accumulated Depreci...	$0.00	$201,014.61	B
1	12/31/2013	000-1510-00	Computer Equipment	$125,895.23	$0.00	B
1	12/31/2013	000-1515-00	Accumulated Depreci...	$0.00	$46,900.36	B
1	12/31/2013	000-1520-00	Machinery & Equipme...	$1,409,884....	$0.00	B
1	12/31/2013	000-1525-00	Accumulated Depreci...	$0.00	$640,078.38	B
1	12/31/2013	000-1530-00	Fleet Vehicles	$62,500.00	$0.00	B
1	12/31/2013	000-1535-00	Accumulated Depreci...	$0.00	$19,068.30	B
1	12/31/2013	000-1600-00	Computer Software	$58,975.48	$0.00	B
1	12/31/2013	000-1610-00	Amortized Software C...	$0.00	$26,894.01	B
1	12/31/2013	000-2100-00	Accounts Payable	$0.00	$1,025,007....	B
1	12/31/2013	000-2110-00	Accrued Expenses	$0.00	$25,500.00	B

363 Account Transact | Completed | First 1000 records where Series is equal to Financial and History TRX is equal to No.

Across the top of the **SmartList** window are the following buttons that allow us to customize and/or create SmartList Favorites:

- **Search**: This button allows us up to four search criteria for our lists.
- **Refresh**: This button forces a refresh of our list. This is useful if we know the data has changed since we saved or opened the SmartList.
- **Columns**: This button allows changing the columns being shown on the list, the order the columns are in, how the lists are sorted, and allows us to add any field that is already built into the object.
- **Favorites**: This button allows us to save our current list as a favorite. There are four types of favorites as follows:
    - **System**: Saves the SmartList so any user has access to the favorite from any company database.
    - **Company**: Saves the SmartList in the current company database only. All users can access it.
    - **User Class**: Saves the SmartList so any user in the same user class as the current user that created the SmartList can access it.
    - **User ID**: Saves the SmartList so only the user that created it can access it.
- **Print**: This button allows printing the list out to the screen or printer, or exporting it in the text, comma delimited, tab delimited, HTML, or XML data file formats.
- **Excel**: This button allows exporting the list directly to Microsoft Excel.
- **Word**: This button allows exporting the list directly to Microsoft Word in a table format.

Basic SmartList concepts

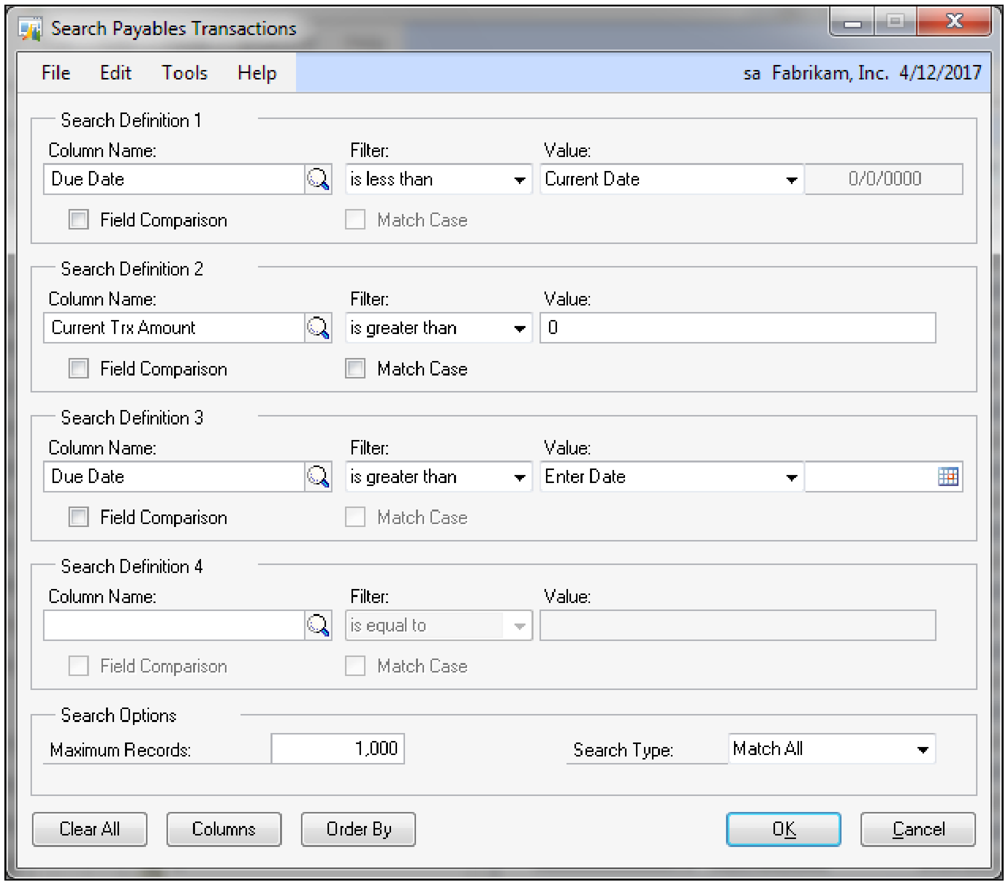
Now that we have a basic understanding of the structure of SmartList, we can begin using some basic concepts to build our SmartList report. The areas we will cover are as follows:

- Narrowing our result list with search criteria
- Adding new columns
- Changing the number of records returned
- Modifying a `dex.ini` switch for faster export to Excel

Narrowing our result list with search criteria

Probably the most important concept of SmartList is the search function. This is what allows us to narrow our result set to the exact data we are looking for. To use the search criteria, we simply click on the **Search** button on the **SmartList** window's toolbar after selecting an existing favorite or the default SmartList represented by an asterisk (*). Let's use the **Past Due Payables** Favorite to see how search criteria are utilized as follows:

1. Open SmartList by clicking on the **Microsoft Dynamics GP** Menu.
2. Select **SmartList**.
3. Expand the `Purchasing` folder.
4. Expand the **Payables Transactions** object.
5. Click on the **Past Due Payables** SmartList.
6. Click on the **Search** button from the toolbar.



Search Definition 1
Column Name: Due Date
Filter: is less than
Value: Current Date 0/0/0000
Field Comparison
Match Case

Search Definition 2
Column Name: Current Trx Amount
Filter: is greater than
Value: 0
Field Comparison
Match Case

Search Definition 3
Column Name: Due Date
Filter: is greater than
Value: Enter Date
Field Comparison
Match Case

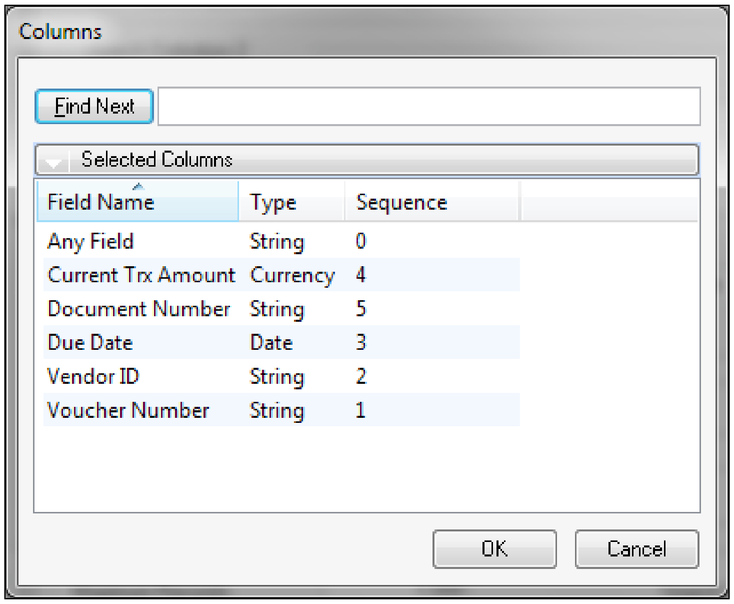
Search Definition 4
Column Name:
Filter: is equal to
Value:
Field Comparison
Match Case

Search Options
Maximum Records: 1,000
Search Type: Match All

In the previous screenshot, we can see that the SmartList is set to return data where the **Due Date** is less than the **Current Date** and the **Current Transaction Amount** greater than **0**. We also notice that the **Search Type** field at the bottom is set to **Match All**. This means that for the data to be returned, all of our search criteria must match. The other option is **Match 1 or More**. This means one or more of the search criteria must match for the data to be returned. Be careful in your selection as choosing one versus the other can give us an entirely different set of data.

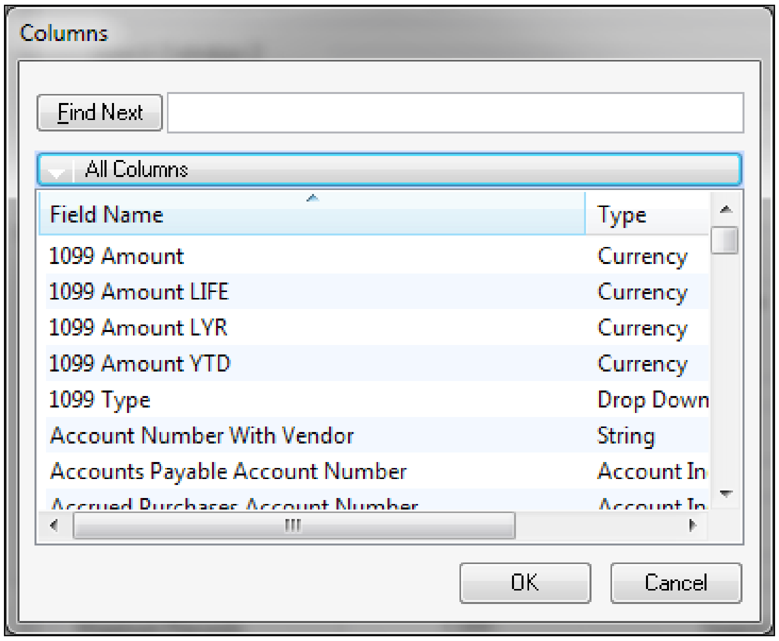
For example, let's say that we want to add one more search criteria to our SmartList. We can easily accomplish this by following these steps:

1. From our search window, click on the lookup magnifying glass on any of the four search definitions.



Field Name	Type	Sequence
Any Field	String	0
Current Trx Amount	Currency	4
Document Number	String	5
Due Date	Date	3
Vendor ID	String	2
Voucher Number	String	1

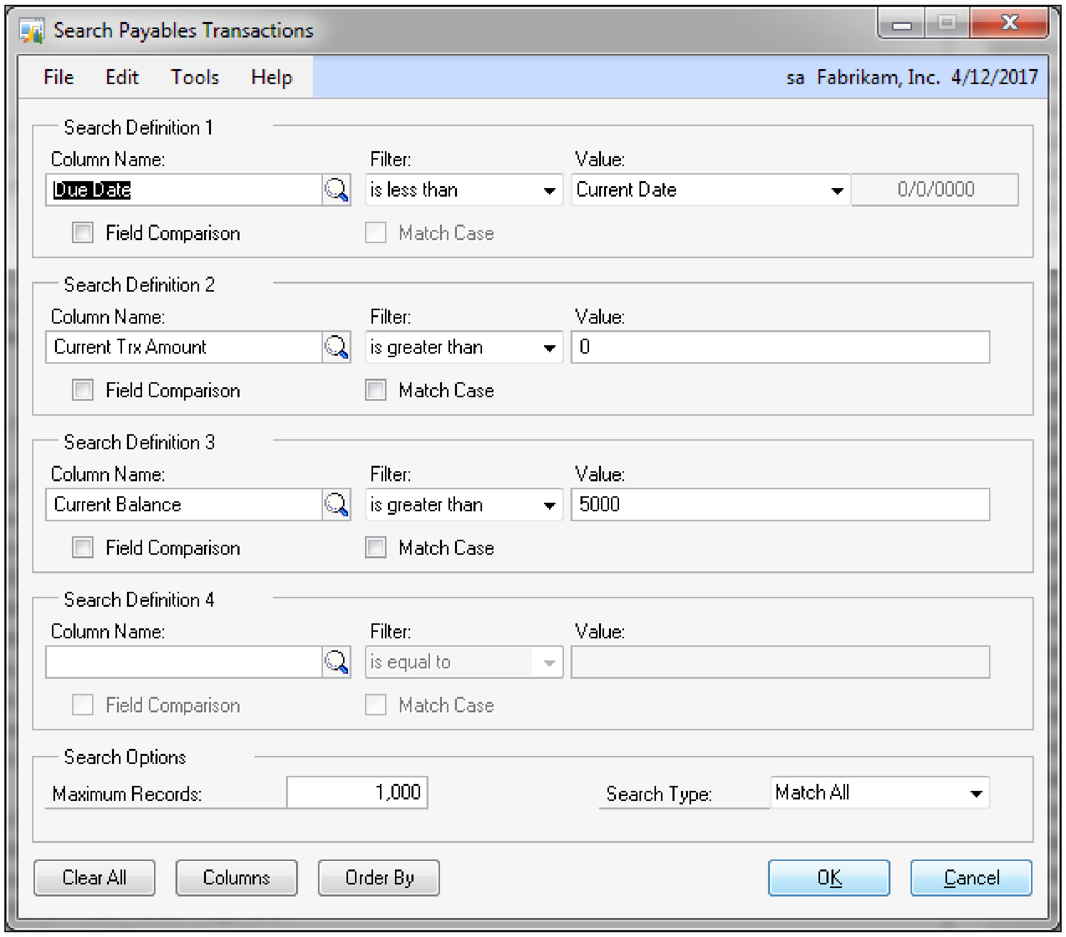
2. We are provided with the list of selected columns on the SmartList. Select a column or click on the **Selected Columns** bar and select **All Columns**.



Field Name	Type
1099 Amount	Currency
1099 Amount LIFE	Currency
1099 Amount LYR	Currency
1099 Amount YTD	Currency
1099 Type	Drop Down
Account Number With Vendor	String
Accounts Payable Account Number	Account In
Accrued Purchases Account Number	Account In

3. We are now provided with a list of the columns available in this object. This gives us the ability to filter our data of the columns that we necessarily don't have on our SmartList. For example, if we included **Document Type** in our SmartList but filtered on the type of **Invoice**, we would see the word Invoice on every line. This redundancy is unnecessary, so we can remove this column from our SmartList.

4. For this SmartList, let's select **Current Balance** as our third search definition with a filter type of **is greater than** and enter a value of `5000`. This will give us all **Past Due Payables** where the vendors' current balance is over $5,000.



Search Definition 1

Column Name:	Filter:	Value:	
Due Date	is less than	Current Date	0/0/0000

☐ Field Comparison ☐ Match Case

Search Definition 2

Column Name:	Filter:	Value:
Current Trx Amount	is greater than	0

☐ Field Comparison ☐ Match Case

Search Definition 3

Column Name:	Filter:	Value:
Current Balance	is greater than	5000

☐ Field Comparison ☐ Match Case

Search Definition 4

Column Name:	Filter:	Value:
	is equal to	

☐ Field Comparison ☐ Match Case

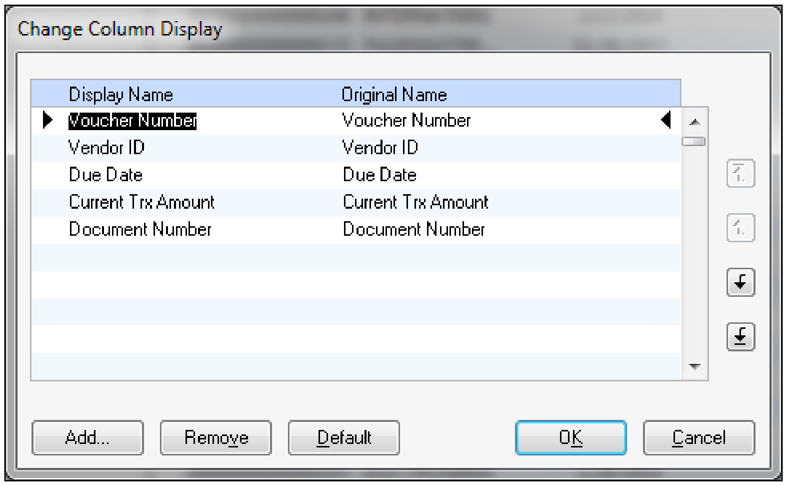
Search Options

Maximum Records: 1,000 Search Type: Match All

Adding new columns

Another important concept of SmartLists is adding columns. We can add any column to our SmartList that is available to us. These columns all come from the source of the object, whether it is an out-of-the-box object that joins the Dynamics GP tables or a custom object that we will discuss later when we explore SmartList Builder and Excel Report Builder. Looking back at our **Past Due Payables** SmartList, we can see the columns being utilized are **Voucher Number**, **Vendor ID**, **Due Date**, **Current Transaction Amount**, and **Document Number**. Let's look at how easy it would be to add a new column to this SmartList for the vendor's **1099 Type** and their current balance:

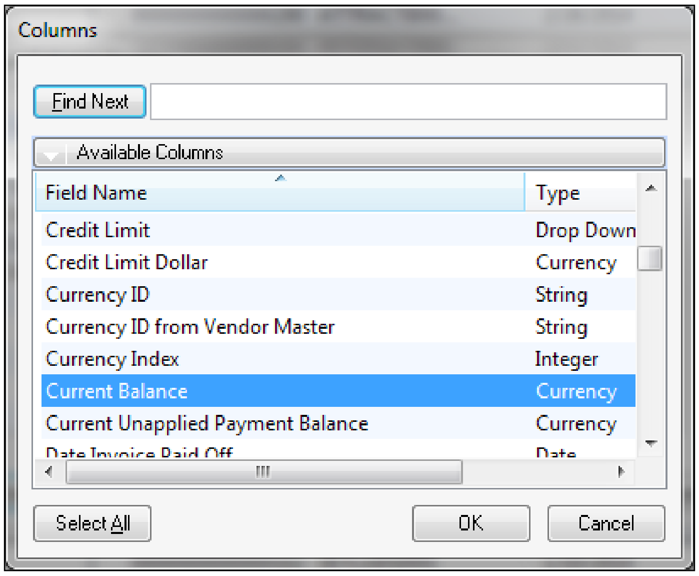
1. Open SmartList from the **Microsoft Dynamics GP** menu.
2. Expand the **Purchasing** folder.
3. Expand the **Payables Transactions** object.
4. Click on the **Past Due Payables** SmartList.
5. Click on the **Columns** button from the toolbar to open the **Change Column Display** window as shown in the following screenshot:



Display Name	Original Name
Voucher Number	Voucher Number
Vendor ID	Vendor ID
Due Date	Due Date
Current Trx Amount	Current Trx Amount
Document Number	Document Number

6. We are provided with a list of the columns selected for our SmartList.

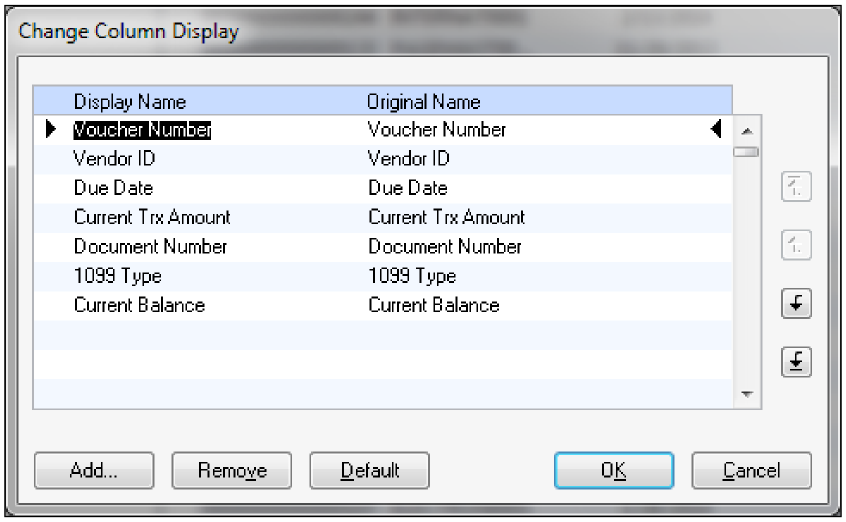
7. Click on the **Add** button.



Columns

Field Name	Type
Credit Limit	Drop Down
Credit Limit Dollar	Currency
Currency ID	String
Currency ID from Vendor Master	String
Currency Index	Integer
Current Balance	Currency
Current Unapplied Payment Balance	Currency
Date Invoice Paid Off	Date

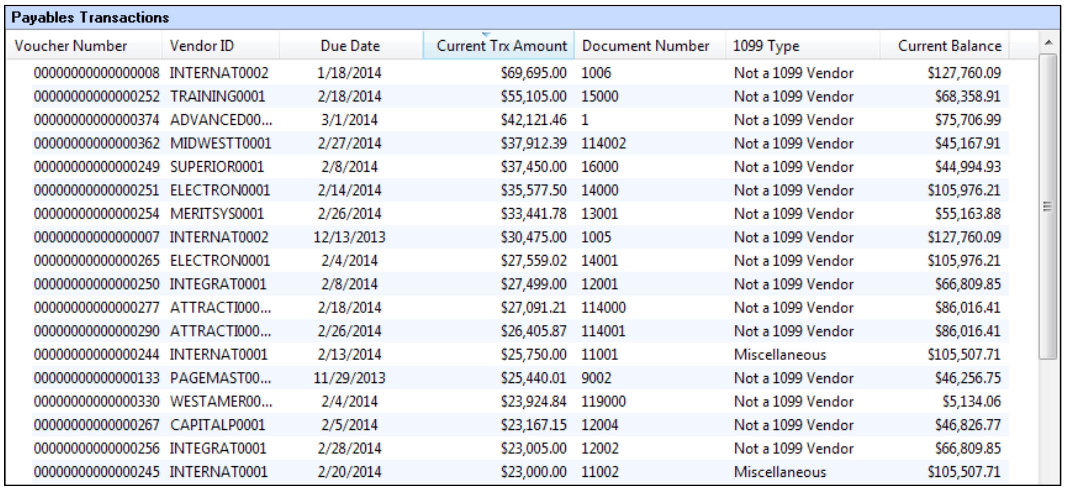
8. As with adding search criteria, we are provided with a list of all the available columns in the object. We can select multiple columns simply by holding the *Ctrl* key as we click on the fields. Let's select **1099 Type** and **Current Balance**.



Change Column Display

Display Name	Original Name
Voucher Number	Voucher Number
Vendor ID	Vendor ID
Due Date	Due Date
Current Trx Amount	Current Trx Amount
Document Number	Document Number
1099 Type	1099 Type
Current Balance	Current Balance

9. We now see our two new fields in the list. Before closing, we can reorder the fields with the **Move Column** buttons on the right or we can remove unnecessary columns.
10. Click on **OK** to return to the SmartList.



Payables Transactions

Voucher Number	Vendor ID	Due Date	Current Trx Amount	Document Number	1099 Type	Current Balance
00000000000000008	INTERNAT0002	1/18/2014	$69,695.00	1006	Not a 1099 Vendor	$127,760.09
00000000000000252	TRAINING0001	2/18/2014	$55,105.00	15000	Not a 1099 Vendor	$68,358.91
00000000000000374	ADVANCED00...	3/1/2014	$42,121.46	1	Not a 1099 Vendor	$75,706.99
00000000000000362	MIDWESTT0001	2/27/2014	$37,912.39	114002	Not a 1099 Vendor	$45,167.91
00000000000000249	SUPERIOR0001	2/8/2014	$37,450.00	16000	Not a 1099 Vendor	$44,994.93
00000000000000251	ELECTRON0001	2/14/2014	$35,577.50	14000	Not a 1099 Vendor	$105,976.21
00000000000000254	MERITSYS0001	2/26/2014	$33,441.78	13001	Not a 1099 Vendor	$55,163.88
00000000000000007	INTERNAT0002	12/13/2013	$30,475.00	1005	Not a 1099 Vendor	$127,760.09
00000000000000265	ELECTRON0001	2/4/2014	$27,559.02	14001	Not a 1099 Vendor	$105,976.21
00000000000000250	INTEGRAT0001	2/8/2014	$27,499.00	12001	Not a 1099 Vendor	$66,809.85
00000000000000277	ATTRACTI000...	2/18/2014	$27,091.21	114000	Not a 1099 Vendor	$86,016.41
00000000000000290	ATTRACTI000...	2/26/2014	$26,405.87	114001	Not a 1099 Vendor	$86,016.41
00000000000000244	INTERNAT0001	2/13/2014	$25,750.00	11001	Miscellaneous	$105,507.71
00000000000000133	PAGEMAST00...	11/29/2013	$25,440.01	9002	Not a 1099 Vendor	$46,256.75
00000000000000330	WESTAMER00...	2/4/2014	$23,924.84	119000	Not a 1099 Vendor	$5,134.06
00000000000000267	CAPITALP0001	2/5/2014	$23,167.15	12004	Not a 1099 Vendor	$46,826.77
00000000000000256	INTEGRAT0001	2/28/2014	$23,005.00	12002	Not a 1099 Vendor	$66,809.85
00000000000000245	INTERNAT0001	2/20/2014	$23,000.00	11002	Miscellaneous	$105,507.71

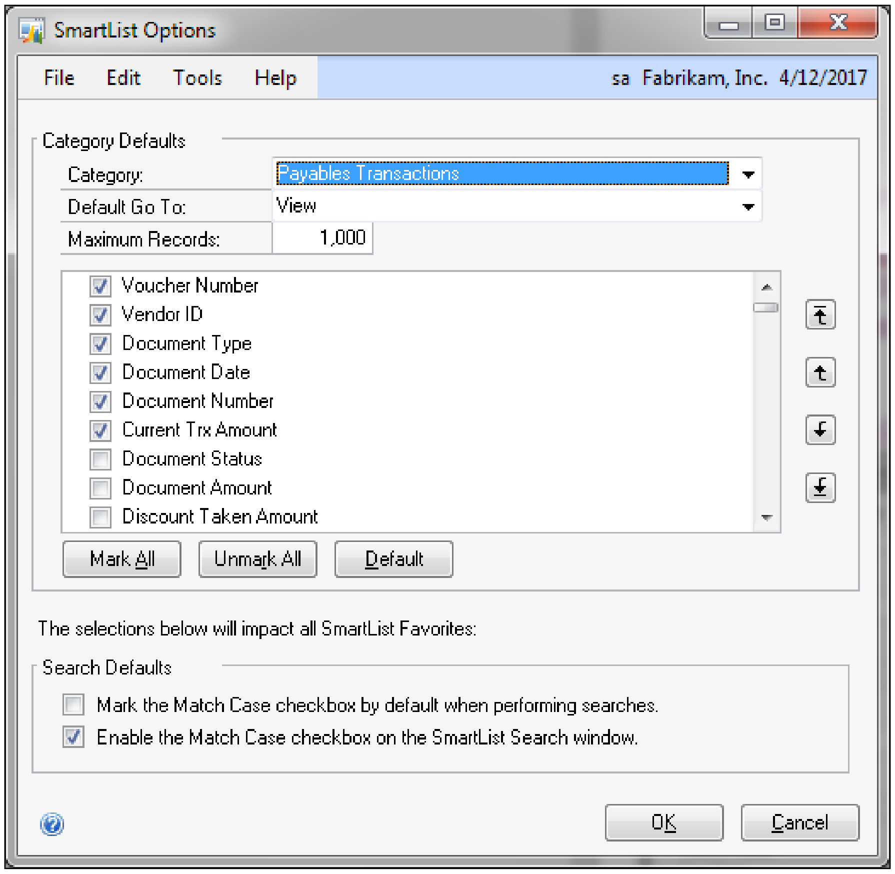
11. We now have our **Past Due Payables** SmartList showing us our new columns. From here we can **Print**, export to **Excel** or **Word**, and save this as a **Favorite**.

Changing the number of records returned

By default, the maximum records returned for a SmartList is set to 1000. As mentioned, this can be changed on our search criteria window. Typically, it is a good idea to keep this setting small while building our SmartLists until we are sure we are returning the data we are actually looking for. Once we are sure of the data being returned, we can then increase the **Maximum Records** field on our SmartList Favorite or we can adjust the **Maximum Records** field for the object. A word of caution on this, it will make the value we set here the default for any SmartList we create from that particular object. Let's look at how we can change this default value for multiple SmartLists at the same time as follows:

1. Open the **Microsoft Dynamics GP** menu.
2. Select **Tools**.

3. Select **System**.
4. Select **SmartList Options**.



Category Defaults

Category: Payables Transactions

Default Go To: View

Maximum Records: 1,000

- [✓] Voucher Number
- [✓] Vendor ID
- [✓] Document Type
- [✓] Document Date
- [✓] Document Number
- [✓] Current Trx Amount
- [ ] Document Status
- [ ] Document Amount
- [ ] Discount Taken Amount

The selections below will impact all SmartList Favorites:

Search Defaults

- [ ] Mark the Match Case checkbox by default when performing searches.
- [✓] Enable the Match Case checkbox on the SmartList Search window.

From this window, we will select **Category**, which corresponds to the SmartList object. We can then change the value in the **Maximum Records** field to increase (or decrease) the number of records allowed when viewing a default SmartList.

Modifying a dex.ini switch for faster export to Excel

One of the most well-known facts that any user, consultant, or report developer who has used SmartList to mine large sets of data and export them to Excel can tell us, is that the export process is extremely time consuming. The main reason for this is how Dexterity communicates with Excel. In short, because SmartList exports records row by row and sets each cell individually, and each cell being written is done by a single Dexterity call for that particular cell, the export routine is very slow.

There is a solution for this. In the `program files` folder for Dynamics GP 2013, there is a configuration file named `dex.ini`. This file can control many settings in GP, but for the purposes of this book, we are focusing just on the switch that will greatly increase our export to Excel performance.

To make this change, you can simply follow these steps:

1. Open Windows Explorer.
2. Navigate to **Program Files(x86)** | **Microsoft Dynamics** | **GP2013** | **Data**.
3. Double-click on the `dex.ini` file to open it.
4. Add a line in the file with the following text:

   ```
   SmartlistEnhancedExcelExport = TRUE
   ```

5. Save and close the `dex.ini` file.

Essentially, what this switch does is instead of exporting row by row, the system will now batch the rows to export together. We will notice a slight delay when we click on the **export** button as the system batches the data, but then it flies as it exports. However, while usually rare, using this switch may cause a loss of formatting with some of the column values. For many, this is a small price to pay for increased SmartList exporting speed!

If we are using SmartList to design a report that reads thousands of records and then will be exported to Excel, this switch is almost a necessity.

Extending SmartList data to Excel by deploying Excel Reports

As we have just seen, one of the common complaints about the default SmartList tool is that they take too long to export to Excel. This is especially true when viewing a transaction related SmartList. For example, let's look at the default SmartList for **Sales Transaction Line Items** without filters. Depending on our environment, this particular SmartList could have thousands, if not hundreds of thousands of records. When selecting to export this to Excel without using the `dex.ini` switch trick for faster exporting, this could take some time. Also, what happens once we get our data into Excel? What do we do the next day after new sales transactions have been added into GP? We will have to run the SmartList and export to Excel all over again just to grab a few extra lines of data. Alternatively, we could apply a filter to find only transactions with a certain date, but this procedure just opens the possibility of data falling through the cracks.

Fortunately, beginning with Microsoft Dynamics GP 10.0, Microsoft has provided the default SmartLists in predefined Excel formats. These Excel Reports are duplicates of the default SmartLists with the one exception that they are accessed directly from Microsoft Excel. Within the Excel document, an **Office Data Connection** (**ODC**) allows the user an instant connection to the data source. Refreshing data within the Excel Report then becomes a simple matter of navigating to **Data | Refresh** from within Excel. No more waiting for thousands of lines to export to SmartList! These reports can be stored in a central shared file location, allowing all users with access to that location to browse these reports.

In this section, we will also take a look at how these reports are deployed, how security to these reports can be controlled, and how to view and modify the deployed reports. Additional information about this process can be found in *Chapter 32*, *Excel Report Deployment* of the *GP 2013 System Setup* guide.

Deploying Excel Reports

Taking advantage of the Excel Reports functionality first requires that they be deployed to a shared location. This shared location can be one of the following:

- A shared folder in a secure server location
- A document library on SharePoint Server
- A document library on SharePoint Online, provided through Office 365

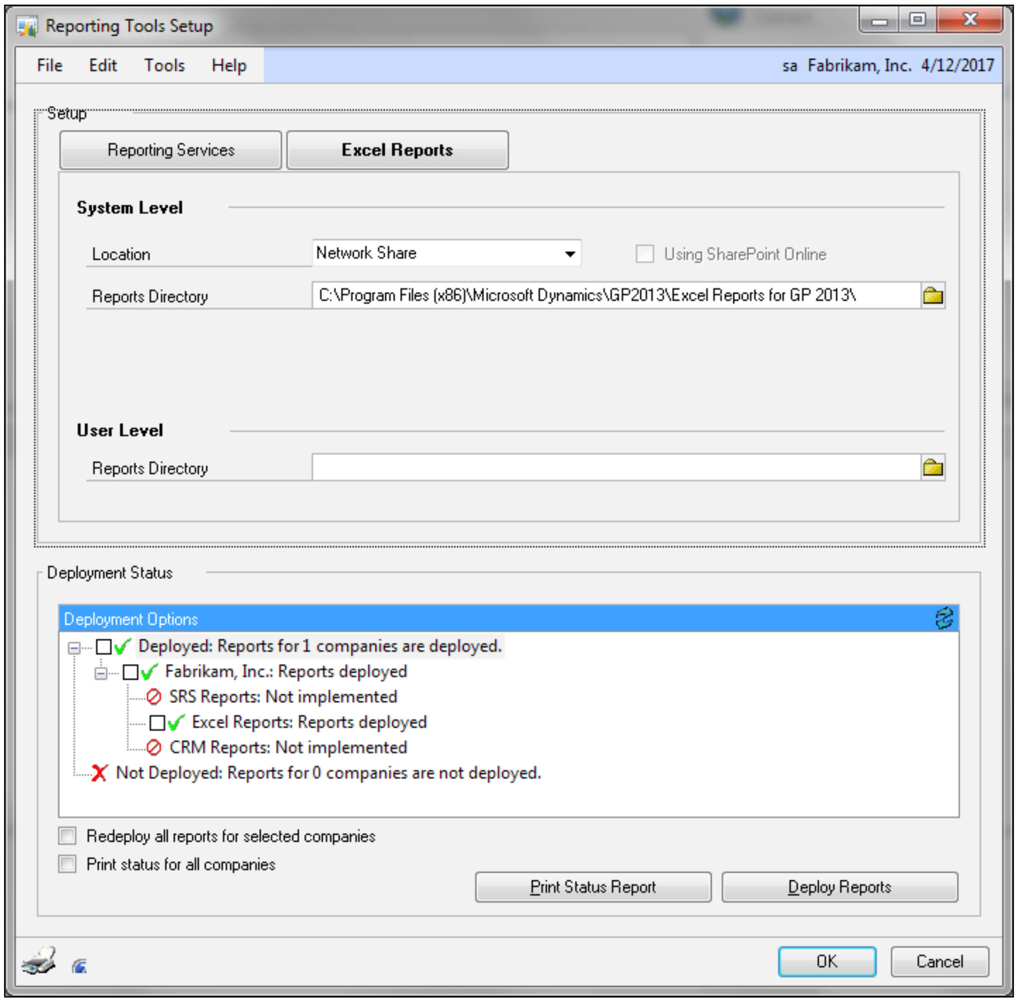
Depending on the modules already installed in GP, the deployment process creates each Excel Report and a corresponding data connection file for each report in the shared location.

In the sample deployment to follow, we will address deploying the Excel Reports to a shared network location. Prior to beginning the deployment process, create the folder, and check the security permissions on that folder. At a minimum, the Windows user performing the installation should have the ability to read and write against the content in this folder. Allowing **Full Control** is preferred, at least for the administrator's Windows user account. Because we are talking about an installation into the computer's file system, it is important that these security permissions are granted to the Windows user, not to the GP user!

The deployment process is initiated from within GP 2013 as follows:

1. The **Reporting Tools Setup** window is accessed via navigating to **Microsoft Dynamics GP | Tools | Setup | System | Reporting Tools Setup**.

2. Click on the **Excel Reports** button.



sa Fabrikam, Inc. 4/12/2017

System Level

Location: Network Share

Using SharePoint Online

Reports Directory: C:\Program Files (x86)\Microsoft Dynamics\GP2013\Excel Reports for GP 2013\

User Level

Reports Directory:

Deployment Status

Deployment Options

Deployed: Reports for 1 companies are deployed.
Fabrikam, Inc.: Reports deployed
SRS Reports: Not implemented
Excel Reports: Reports deployed
CRM Reports: Not implemented
Not Deployed: Reports for 0 companies are not deployed.

Redeploy all reports for selected companies

Print status for all companies

3. The process for deploying the Excel Reports is fairly straightforward. In the **System Level** section, use the **Location** dropdown to select the deployment method. The choices are **Network Share** or **SharePoint**. If using **SharePoint**, there is an additional choice to use **SharePoint Online**. Next, in the **Reports Dictionary** field, enter the path name of the shared folder location where the Excel Reports and data connections will be stored. Although a mapped drive location can be listed here, it is recommended to use the **Universal Naming Convention** (**UNC**) format to identify the location (for example, `\\computer\Excel Reports for GP 2013\Reports`). Before tabbing off this field, the system will check to ensure that a valid pathname has been entered.
4. In the **User Level** section, in **Reports Directory**, enter the folder path that can be used should users decide to modify the system-wide reports for personal use. In this field, a mapped drive location or the UNC path can be listed here. The percentage symbol (`%`) can be used as a wild card in this path (for example, `\\servername\Documents and Settings\%\Reports`).
5. Under **Deployment Status**, click on the plus signs (**+**) to expand the list of companies. Expand each company to display the deployment status of **SRS Reports**, **Excel Reports**, and **CRM Reports**. Check the box for **Excel Reports**. To complete the deployment, click on the **Deployment Reports** button.
6. When deployment is complete, browse out to the folder location that was specified in **Reports Directory** and verify that the reports and data connections were deployed.

Now that we have successfully installed the Excel Reports and their data connections, let's take a look at how security is managed for these reports.

Maintaining security for Excel Reports

Unlike maintaining the security of GP modules and windows via GP user logins, security for Excel Reports is maintained strictly through the **Windows Active Directory User** accounts. Security must be controlled in two places as follows:

- Security permissions for the shared network folder
- Security permissions for the company databases

Shared network folder permissions

As we discussed earlier, the shared network folders must be set up so that reports and data connections can be deployed to those folders. To grant users access to these folders, first right-click on the folder in Windows Explorer, and select **Properties**. This opens the folder properties window from which security privileges to the contents of the folder can be granted or revoked. At a minimum, users should be granted the ability to read and write the contents of the folder.

Once users are granted permission to a particular folder, they can browse to the folder containing the deployed reports and see each of the reports that have been deployed. Double-clicking on the report will open the report in Excel. However, if the Windows Active Directory account for the user opening the report has not also been granted security permissions at the database level, then a warning message will appear, letting the user know that he or she does not have the database level permissions.

Although a user with the folder access can open the Excel Reports, if he or she does not have the appropriate database permissions required to view the content of that report, then the user will not be able to refresh the report.

Database-level permissions

As part of the Excel Reports deployment, new database roles are created in each company database for which reports are deployed. The database roles, all of which begin with an `rpt_prefix`, govern user access to database objects such as views, tables, and stored procedures that are necessary for viewing the data in the Excel Reports.

The roles are named in such a way that it becomes fairly intuitive for someone setting up security to understand which roles allow access to which reports. For example, in order for a user to access data for the payroll-related Excel Reports, the Windows user account for that user must be assigned to the `rpt_payroll` role.

For a more detailed list of which role allows access to a particular report, check the article ***#949524***, which is publicly available on the Microsoft Support site found at `http://support.microsoft.com`.

Also, note that a security role called `rpt_power` user is created by the install. This role has been made a role member of every other role, meaning any users assigned to the `rpt_power` user database role will have access to all the database content required to view the Excel Reports data.

Granting user access to these roles requires creating a Windows logon for the user or group in SQL Management Studio and then adding that user or group to the appropriate `rpt_ roles`. In the case that multiple users will be using the Excel Reports, we recommend setting up a **Windows Active Directory Group** to which individual users can be added. The following security privileges described can then be granted at the group level, making for a much less time-consuming process!

To grant the database role access to a user or group, follow these steps:

1. Open SQL Management Studio.
2. Connect to the GP server.
3. Expand the **Security** node.
4. Right-click on the `Logins` folder, and select **Properties** to open the **Login – New** window.
5. Create a new Windows login for the user or group by selecting the **Windows Authentication** option, and searching for that user or group in the directory.
6. Once all the fields in the **General** tab have been completed, select the **User Mapping** tab to map this user to the various GP-related databases.
7. Click on the **Map** checkbox next to the database which the user should have access to.
8. With a company database selected, in the **Database role membership** pane at the bottom of the window, select the database role which the user or group should be assigned to (remember, the reporting roles are all preceded by `rpt_`).
9. Map the user to additional company databases and reporting roles as needed. Several Excel Reports use SQL views that refer to the tables in the GP System database, so don't forget to map users and groups to reporting roles in this database too!

Assuming these users have the appropriate permissions on the folders containing these Excel Reports, they should now be ready to access and use the Excel Reports!

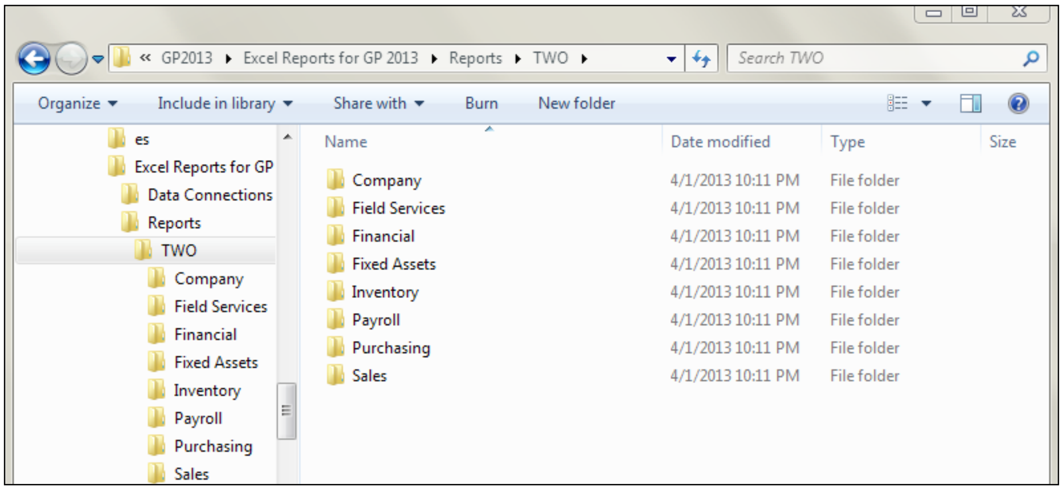
Accessing and using Excel Reports

Deployed reports can be accessed in one of the following two ways: through Windows Explorer or through GP 2013. Remember, security must be set appropriately before reports can be opened or refreshed.

Connecting to Excel Reports via Windows Explorer

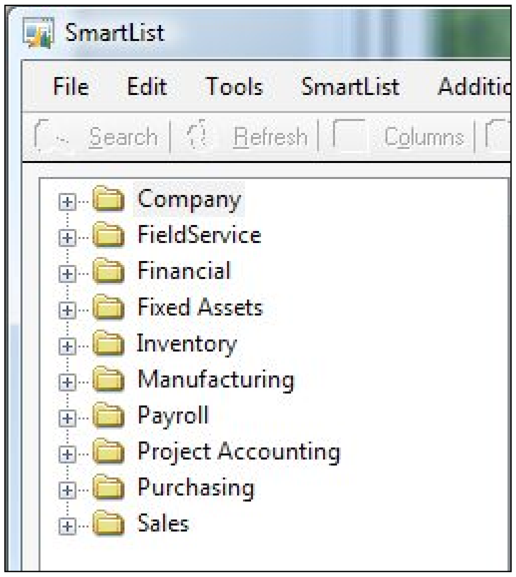
To connect via Windows Explorer, users can browse to the appropriate share folder identified during the deployment in the **System Level Reports** field. Under this folder, another set of folders have been created by the install; these folders correspond to the company databases for which reports and data connections were deployed. If more than one company database was included in the deployment, then a separate folder will exist for each company database.

Underneath each company database folder, another layer of folders has also been created. These folders correspond to different modules in GP, for example, `Financials`, `Purchasing`, `Project`, and so on. They are created depending on which modules are installed in the selected GP company database. Users will also notice that the list of folders at this level corresponds to the same folder grouping list that exists when opening the SmartList from GP. This folder structure is further illustrated in the following screenshot:



Name	Date modified	Type	Size
Company	4/1/2013 10:11 PM	File folder	
Field Services	4/1/2013 10:11 PM	File folder	
Financial	4/1/2013 10:11 PM	File folder	
Fixed Assets	4/1/2013 10:11 PM	File folder	
Inventory	4/1/2013 10:11 PM	File folder	
Payroll	4/1/2013 10:11 PM	File folder	
Purchasing	4/1/2013 10:11 PM	File folder	
Sales	4/1/2013 10:11 PM	File folder	

As we can see in the following screenshot, the list of SmartList Objects corresponds to the folders in the previous screenshot:



Clicking on each folder will reveal a series of Excel documents, each corresponding to one of the default SmartLists that can be found in GP. Even the content of the Excel documents, such as columns and filters, mirrors the content of the similarly named SmartList from GP. These Excel Reports provide users with advantages over the default SmartList in that the data is already stored in Excel, and it can be refreshed instantaneously.

If a user accesses a report on the shared system level directory, makes changes to that report, and then saves it in the same location, those saved changes will be accessible to all other users who access that same report. If a user wants to save that changed report so that only he or she can see the changes, then the report should be saved to the path defined in the **User Level Reports** field of the deployment.

Note that reports are only deployed to the **System Level Reports** path during the initial deployment. Nothing is actually deployed to the **User Level Reports** path. Instead, the **User Level Reports** path is designed to be a repository for users who wish to save the default reports without impacting the original report.

These reports are deployed in Microsoft Excel 2007 format (`*.xlsx`), so users who are still on Microsoft Excel 2003 will need to install the Microsoft Office Compatibility Pack for Word, Excel, and PowerPoint 2007 file formats.

Connecting to Excel Reports via GP 2013

Users can also view these reports from directly within GP. After Excel Reports have been deployed, users can select **Excel Reports** from the list of available Report Lists. This will open up the Excel Reports list detailing each report and data connection to which the Windows user account has access. Selecting to open a report will open the **System Level Reports** report in the Excel format, just as would happen if the user browsed to that report via Windows Explorer.

Creating and publishing new SmartLists using SmartList Builder

Earlier in this chapter, we discussed modifying the default SmartLists that are included with any base install of GP. As wonderful and helpful as these default SmartLists can be, there comes a time in every GP user's life that no SmartList exists to satisfy a particular reporting need. Perhaps an existing SmartList provides some of the data, but it's missing a join to the particular table that provides critical information required by an end user. At times like these, where the ease-of-access and easy-to-use functionality of SmartList is still desired, developers and consultants can use SmartList Builder to extend the basic reporting functionality offered by the default SmartLists.

The beauty of SmartList Builder is that it allows us to replicate the same functionality of regular SmartLists while connecting to data sources beyond just the standard Microsoft Dynamics GP tables. In fact, developers and consultants have the ability to connect to one of four data source objects when developing a new SmartList in the Builder as follows:

- **Microsoft Dynamics GP tables**: This includes tables defined in the Microsoft Dynamics GP and third-party dictionaries
- **Microsoft SQL Server tables**: This includes tables found outside the Dynamics GP dictionaries as well as views written against tables inside and outside the GP dictionaries
- **Data Connections**: This includes SQL views for common fields and tables found in Dynamics GP
- **Extender Resources**: This includes Extender objects that have already been defined via the Extender functionality

When a standard, out of the box SmartList Object doesn't do the trick, we can create our own SmartList Object using Dynamics GP tables or by writing a SQL view that serves as the underlying data source! The possibilities become limitless once SmartList Builder is included in our reporting toolbox.

Understanding the SmartList Builder window

SmartList Builder is accessed directly from GP as follows:

1. Open the **Microsoft Dynamics GP** menu.
2. Select **Tools**.
3. Select **SmartList Builder**.
4. Select **SmartList Builder** for a second time.

This opens the **SmartList Builder** window. Across the top of the Builder, we see the following buttons:

- **Save**: It allows the user to save the current SmartList Object.
- **Clear**: It clears the contents of the SmartList Builder, allowing the user to begin creating a new SmartList Object.
- **Delete**: It deletes the selected SmartList Object.
- **GoTo**: It allows the user to set up a **GoTo** action that can be used to execute a particular action when selecting a record in the SmartList. For example, a GoTo action can be set up to open the **Customer Maintenance** window for a customer identified in the selected SmartList record.
- **Restrictions**: It allows users to set up default restrictions on a SmartList Object so these restrictions do not have to be entered as filters in the resulting SmartList. For example, a restriction can be placed on a new SmartList Object so that only transactions after a certain date will be displayed.
- **Calculated Fields**: It allows users to add calculated columns to the new SmartList Object that perform mathematical operations using one or more fields found in the underlying source data for the SmartList Object.
- **Options**: In this buttons window several different options exist, including one option to create a **Summary SmartList** that allows users to group records by certain criteria and perform other mathematical operations that can summarize large volumes of data. Additionally, from this window, users can select the **Multicompany SmartList** checkbox to combine data from multiple company databases.

Most of the buttons at the top of the **SmartList Builder** window are used to access unique SmartList Builder functions; however, the actual work of creating SmartList Objects occurs in the fields and panes below these buttons. We will explore the use of these fields and panes in the next section where we will create a sample SmartList Object.

Creating a new SmartList Object via SmartList Builder

With the advent of Dynamics GP 2010 and later, GP users have even more control over e-mailing statements to customers. In fact, when setting up customer-related information, e-mail addresses can be assigned to each address setup for that customer via the **Internet Addresses** window (the **Cards** | **Sales** | **Customer** | **Internet Addresses** button). In this window, users can identify the default e-mail address that will appear when accessing the **Receivables E-Mail Detail Entry** windows for transactions using this customer and address.

Unfortunately, however, the e-mail address information entered in these fields cannot be accessed via any of the default **Customer** SmartLists. Instead, if the **Accounts Receivable** coordinator wants an easy way to look up which e-mail address will be the default for a given customer-address combination, he or she must look through each individual customer card. A new SmartList Object must be created to allow this information to be viewable in a SmartList.

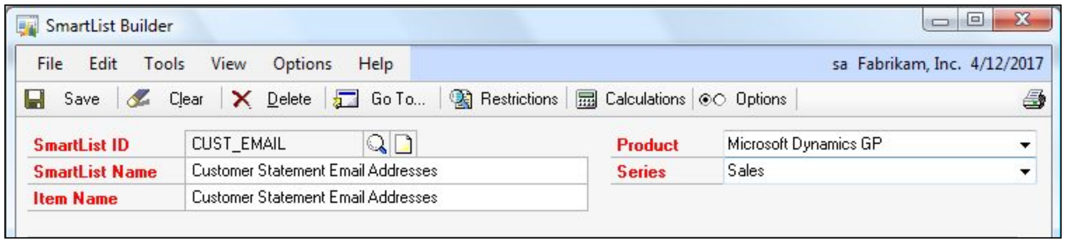
First, we must use the data-seeking skills that we learned in the last chapter to find out which Microsoft Dynamics GP tables contain the basic customer information and the e-mail addresses we need. In GP terms, the data we are seeking can be found in the **Customer Maintenance**, **Customer Address Maintenance**, and **Internet Addresses** windows. Before continuing on through the rest of this exercise, try and see if you can find the display name for the Microsoft Dynamics GP tables that contain the information found in these windows.

If you attempted this exercise, hopefully by now you've discovered that the data we need is found in three tables (the table display name is listed with the physical name included in parentheses) as follows:

- **RM Customer MSTR** (RM00101): This table is located in the **Sales** series and contains one record for each individual customer record in GP 2013.
- **Customer Master Address File** (RM00102): This table is located in the **Sales** series and contains one record for each individual customer-address combination that exists in GP 2013.
- **Internet Addresses** (SY01200): This table is located in the **Company** series and contains the Internet address records for a variety of different record types such as **Items**, **Customers**, **Vendors**, and more. E-mail addresses entered in the **Internet Addresses** window for customers are stored in this table with a **Master Type** of **CUS**.

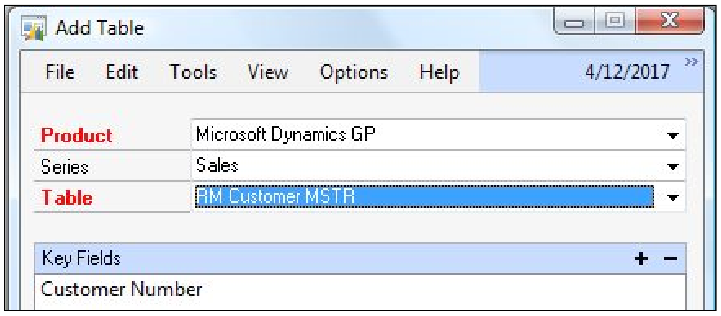
Now that we've identified the tables we are going to use, let's use them to create a new SmartList Object in the SmartList Builder as follows:

1. Open the **Microsoft Dynamics GP** menu.
2. Select **Tools**.
3. Select **SmartList Builder**.
4. Select **SmartList Builder** for a second time.
5. In the **SmartList ID** field, enter `CUST_EMAIL` as the identifier for this SmartList Object.
6. In the **SmartList Name** field, enter `Customer Statement Email Addresses` as the name for this SmartList Object. This name will appear in the left-hand pane of the **SmartList** window and will appear in the menu bar above the SmartList as it is being browsed.
7. In the **Item Name** field, enter `Customer Statement Email Addresses`. **Item Name** can be the same as **SmartList Name** and will appear in the lower left-hand corner of the **SmartList** window where the number of records found in the SmartList is displayed.
8. Enter `Microsoft Dynamics GP` as **Product** and `Sales` as **Series** for this SmartList Object. These options will determine what folder grouping the SmartList Object will fall under. In this case, our SmartList Object will be grouped under the `Sales` folder in the **SmartList** pane. At this point, your SmartList Object header should look like the following screenshot:


SmartList Builder
File Edit Tools View Options Help
sa Fabrikam, Inc. 4/12/2017
Save Clear Delete Go To... Restrictions Calculations Options

SmartList ID	CUST_EMAIL	Product	Microsoft Dynamics GP
SmartList Name	Customer Statement Email Addresses	Series	Sales
Item Name	Customer Statement Email Addresses		

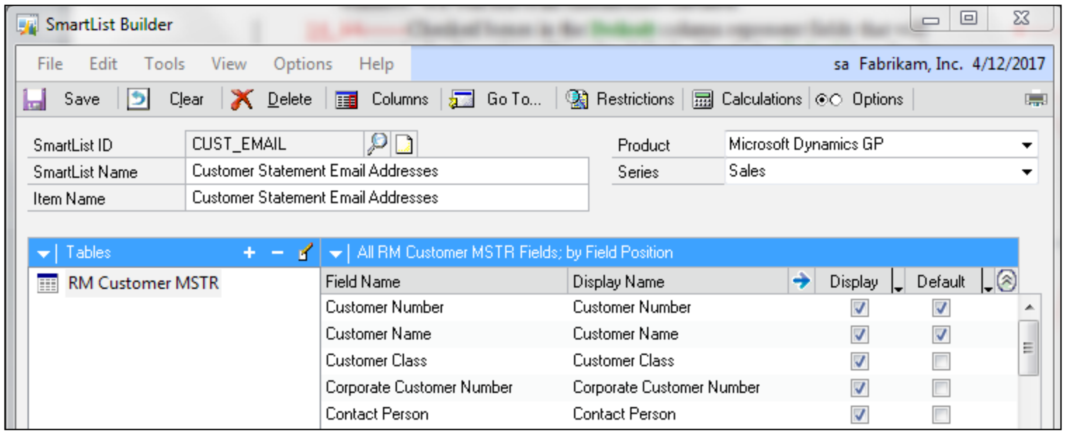
9. On the left-hand side of the **SmartList Builder** window, click on the **Add Table** button. This button looks like a plus (**+**) sign and appears at the top of the left-most pane in this window. Select **Microsoft Dynamics GP Table** to open the **Add Table** window.
10. In the **Add Table** window, select **Microsoft Dynamics GP** as **Product**, **Sales** as **Series**, and **RM Customer Master** as **Table**.



Product	Microsoft Dynamics GP
Series	Sales
Table	RM Customer MSTR

Key Fields
Customer Number

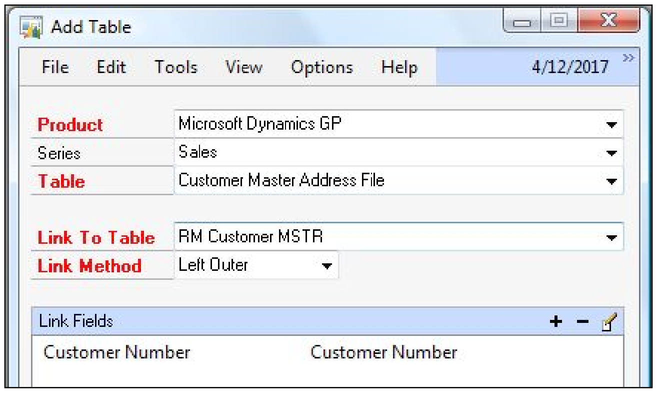
11. We are then given the option to select **Key Fields** for this table. **Key Fields** represent a unique combination of values for each record. In this table, each record contains a unique **Customer Number**, so we will use this as the key field. Notice that SmartList Builder has already listed the key field in the **Key Fields** pane, so we can click on the **Save** button to return to the main **SmartList Builder** window.
12. In this window, we see our newly added table, along with a list of columns from the table that can be included in our SmartList Object.
13. Leaving a checkbox checked in the **Display** column means that after the SmartList Object is created, these columns will be available for user selection by navigating to **Columns** | **Add** from the **SmartList** window. We will leave all checkboxes checked.
14. Checked boxes in the **Default** column represent fields that will appear in the SmartList Object by default. Check the **Default** box for the **Customer Number** and **Customer Name** columns.


SmartList Builder
File Edit Tools View Options Help sa Fabrikam, Inc. 4/12/2017
Save Clear Delete Columns Go To... Restrictions Calculations Options
Tables
RM Customer MSTR
All RM Customer MSTR Fields; by Field Position

SmartList ID	CUST_EMAIL
SmartList Name	Customer Statement Email Addresses
Item Name	Customer Statement Email Addresses
Product	Microsoft Dynamics GP
Series	Sales

Field Name	Display Name	Display	Default
Customer Number	Customer Number	✓	✓
Customer Name	Customer Name	✓	✓
Customer Class	Customer Class	✓	
Corporate Customer Number	Corporate Customer Number	✓	
Contact Person	Contact Person	✓	

15. Now, we need to add an additional table to the first table. Click on the **Add Table** button again and select **Microsoft Dynamics GP Table**.
16. This time, we will add the **Customer Master Address File** table. Select **Microsoft Dynamics GP** as **Product**, **Sales** as **Series**, and **Customer Master Address File** as **Table**.
17. Because we are adding a new table to the first one, we have to tell SmartList Builder how to link the two tables. Because we want all customers from the first table to be displayed in the SmartList Object, regardless of whether or not they have been assigned an address, we will select **Left Outer** as **Link Method**.



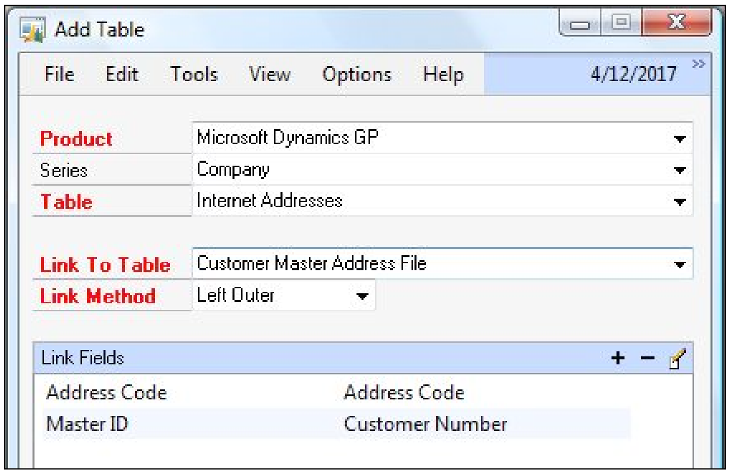
Product	Microsoft Dynamics GP
Series	Sales
Table	Customer Master Address File
Link To Table	RM Customer MSTR
Link Method	Left Outer

Link Fields

Customer Number	Customer Number

18. SmartList Builder identifies common fields between the two tables, and, in this case, **Customer Number** is listed in the **Link Fields** pane. Click on **Save** to return to the **SmartList Builder** window.
19. With **Customer Master Address File** selected in the left-hand pane, select to include the **Address Code** field as a **Default** value in this SmartList Object.
20. Now, let's add our final table to this SmartList Object. Again, with **Customer Master Address File** selected in the left-hand pane, click on the **Add Table** button.
21. This time, we are going to add the **Internet Addresses** table, and link it to the **Customer Master Address File** table. To do this, we can select **Microsoft Dynamics GP** as **Product**, **Company** as **Series**, and **Internet Addresses** as **Table**.

22. Because e-mail addresses are stored for each customer-address relationship, we need to link the **Internet Addresses** table to a table with similar fields. In the **Link to Table** dropdown, select the **Customer Master Address File** table. Again, because we want all customer-address combinations to appear in our SmartList Object, we select **Left Outer** as our **Link Method**. Note that if we only wanted customer-address combinations that contain a value in the e-mail address fields to be displayed in our SmartList Object, we would use the **Equals** option for **Link Method**. This decision often requires consideration of the end user's needs and can be changed at a later time, if necessary.
23. Now we need to link our two tables together. The connection between **Internet Addresses** and **Customer Master Address File** is a bit trickier than the previous connection between two tables. SmartList Builder is intelligent enough to notice that **Address Code** is a commonly linked field between the two tables, but it fails to make a similar connection for the customer number. If we were to leave **Address Code** as the only link, our report would not be effective as numerous customers may share similarly named address codes. To link **Customer Number**, we have to join the **Customer Number** field from the **Customer Master Address File** table to the **Master ID** column in the **Internet Addresses** table. To add this link, click on the **Add Link** button (the plus (**+**) sign next to **Link Fields**) to open the **Add Link** window and make the selections that we've just identified.



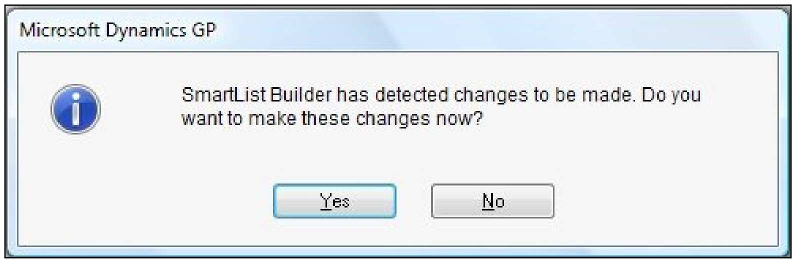
Product	Microsoft Dynamics GP
Series	Company
Table	Internet Addresses
Link To Table	Customer Master Address File
Link Method	Left Outer

Link Fields

Address Code	Address Code
Master ID	Customer Number

24. Click on **Save** twice to return to the **SmartList Builder** window.

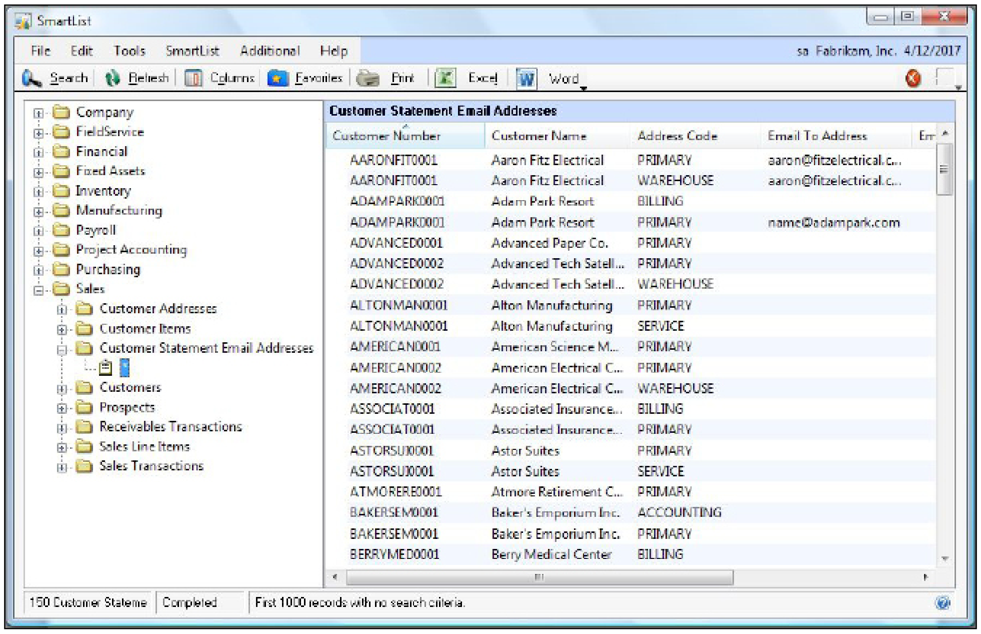
25. With **Internet Addresses** selected in the left-hand pane, select to include the **Email To Address** field as the **Default** value in this SmartList Object.
26. We are now ready to use our new SmartList Object! Save the SmartList Object, and close the **SmartList Builder** window.
27. Open the regular **SmartList** window, by navigating to **Microsoft Dynamics GP | SmartList**.
28. Any time the changes are made via SmartList Builder a message will appear the next time the standard **SmartList** window is opened as shown in the following screenshot:


Microsoft Dynamics GP

SmartList Builder has detected changes to be made. Do you want to make these changes now?

29. Click on **Yes** to allow SmartList Builder to apply the new changes to the existing SmartList.

30. Once SmartList is open, expand the `Sales` folder, and then expand the `Customer Statement Email Addresses` folder to see the newly created SmartList Object. Congratulations, you've created your first SmartList Object using SmartList Builder!



Customer Statement Email Addresses

Customer Number	Customer Name	Address Code	Email To Address	Em
AARONFIT0001	Aaron Fitz Electrical	PRIMARY	aaron@fitzelectrical.c...	
AARONFIT0001	Aaron Fitz Electrical	WAREHOUSE	aaron@fitzelectrical.c...	
ADAMPARK0001	Adam Park Resort	BILLING		
ADAMPARK0001	Adam Park Resort	PRIMARY	name@adampark.com	
ADVANCED0001	Advanced Paper Co.	PRIMARY		
ADVANCED0002	Advanced Tech Satell...	PRIMARY		
ADVANCED0002	Advanced Tech Satell...	WAREHOUSE		
ALTONMAN0001	Alton Manufacturing	PRIMARY		
ALTONMAN0001	Alton Manufacturing	SERVICE		
AMERICAN0001	American Science M...	PRIMARY		
AMERICAN0002	American Electrical C...	PRIMARY		
AMERICAN0002	American Electrical C...	WAREHOUSE		
ASSOCIAT0001	Associated Insurance...	BILLING		
ASSOCIAT0001	Associated Insurance...	PRIMARY		
ASTORSUI0001	Astor Suites	PRIMARY		
ASTORSUI0001	Astor Suites	SERVICE		
ATMORERE0001	Atmore Retirement C...	PRIMARY		
BAKERSEM0001	Baker's Emporium Inc.	ACCOUNTING		
BAKERSEM0001	Baker's Emporium Inc.	PRIMARY		
BERRYMED0001	Berry Medical Center	BILLING		

150 Customer Stateme | Completed | First 1000 records with no search criteria.

Deploying new Excel Reports with Excel Report Builder

As with SmartList Builder, Excel Report Builder is another powerful tool in our reporting toolbox. Though many of the same concepts that exist for SmartList Builder hold true for Excel Report Builder, we are not building a new SmartList Object that will be accessed through SmartList. Instead, as we discussed earlier in the chapter, with the deployment of Excel Reports, we are creating our own Excel Reports, Office Data Connections, or both.

The advantage this tool gives us is that we can designate that our resulting report will either be a simple list of data or a PivotTable in Excel. It basically takes out the step of our users needing to first extract the list of data, and then create the PivotTable. Essentially, the user will open the resulting Excel file and be provided with the fields they can select in their PivotTable. They will not be provided with the list of raw data but instead will select their fields in the PivotTable. The data will refresh and be saved so that next time the spreadsheet is opened, it will be refreshed with the latest data without the Excel Report needing to be republished every time. And if we are using SharePoint, we can also publish the Excel Reports and/or **Office Data Connections** (**ODC**) directly to our SharePoint site.

Also in common with SmartList Builder, Excel Report Builder allows us to replicate the same functionality of regular SmartLists while connecting to data sources beyond just the standard Microsoft Dynamics GP tables. As with SmartList Builder, Excel Reports Builder can also connect to several data source objects; among these are Microsoft Dynamics GP tables, SQL Server tables, Data Connections, and Extender Resources. Refer to our discussion in the *Creating and publishing new SmartLists by using SmartList Builder* section for a description of each of these data sources.

When a standard, out of the box Excel Report doesn't do the trick or doesn't even exist, we can create our own refreshable Excel Report using Dynamics GP tables or writing a SQL view that serves as the underlying data source! The possibilities become limitless with the Builder!

Understanding the Excel Reports Builder window

Excel Report Builder is accessed directly from GP as follows:

1. Open the **Microsoft Dynamics GP** menu.
2. Select **Tools**.
3. Select **SmartList Builder**.
4. Select **Excel Report Builder**.
5. Select **Excel Report Builder** for a second time.

This opens the **Excel Report Builder** window. Across the top of the Builder, we see several buttons. Several of these buttons are similar to the ones found in the **SmartList Builder** window that we discussed earlier in this chapter. Several of these buttons, however, offer functionality specific to Excel Reports Builder as follows:

- **Options**: In this buttons window several different options exist, including one option to **Create a Summary Page** that allows users to group records by certain criteria and perform other mathematical operations that can summarize large volumes of data. Additionally, from this window, users can select the **Multicompany report** checkbox to combine data from multiple company databases, select **to Display totals at the end of each list**, or **to consolidate all reports into a single worksheet**.
- **Drill Down**: It allows the user to set up a **Drill Down** to GP when selecting a record in the Excel Report. For example, a Drill Down can be set up to open the **Customer Maintenance** window for a customer identified in the selected Excel Report record.
- **Publish**: It allows the user to publish the List or PivotTable Excel Report to an Excel or Office Data Connection file or to publish them to a SharePoint site.

Most of the buttons at the top of the **Excel Report Builder** window are used to access unique Excel Report Builder functions; yet, the actual work of creating Excel Reports occurs in the fields and panes below these buttons. We will explore the use of these fields and panes in the next section as we will create a sample Excel Report.

Creating a new Excel Report via Excel Report Builder

We have put together a real life scenario that we have come across at multiple companies to demonstrate how easy it is to create these Excel Reports.

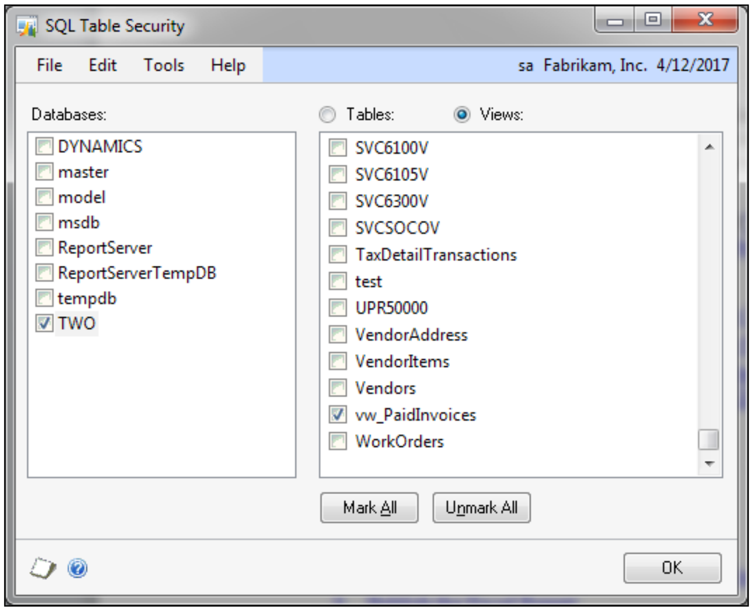
Let's take for example an Accounts Payable department that often needs to answer questions from vendors on what check number actually paid a particular invoice. This can be accomplished in GP using the inquiry windows, by finding the check number and by drilling in to find what document it was applied against. This works well if we just need to find a small number of these transactions, but the question from the end user is almost inevitably asked: Is it possible for me to get a list of all my invoices and the check numbers that paid those invoices for the month, and while we are at it, can we have it refresh every month with new data and be exported to Excel? Our initial response might be to just use SmartList, but as we have learned, SmartList doesn't always allow us to get the exact information we need. In this scenario, because both the Invoice transactions and the Payment transactions are all included in the same SmartList Object, we have no way to join them together. Our next response might be to consider using SmartList Builder to join the tables we need and build this report. This might work, but remember our company wants this data refreshed every month and exported out to Excel. If we fulfill this request using SmartList Builder, our end users would need to open SmartList each month and export an updated list. So, what is the solution? We can use Excel Reports builder to build a custom Excel Report, so all our end users need to do is to open the appropriate spreadsheet every month, click on **Refresh**, and have a completed report, all without even touching Dynamics GP.

So, what do we need to do to create this Excel Report? In this example, we will be using a SQL view, `vw_PaidInvoices`, as our data source, which we have included.

The step-by-step process is as follows:

1. Open the `Create_vw_PaidInvoices` text file (provided) in SQL Management Studio and run it against the `TWO` database to create the view.

2. Before the Builder can see the view, we must grant access to the view via Excel Report Builder security as follows:
    - From the **Microsoft Dynamics GP** menu, navigate to **Tools** | **SmartList Builder** | **Security** | **SQL Table Security**



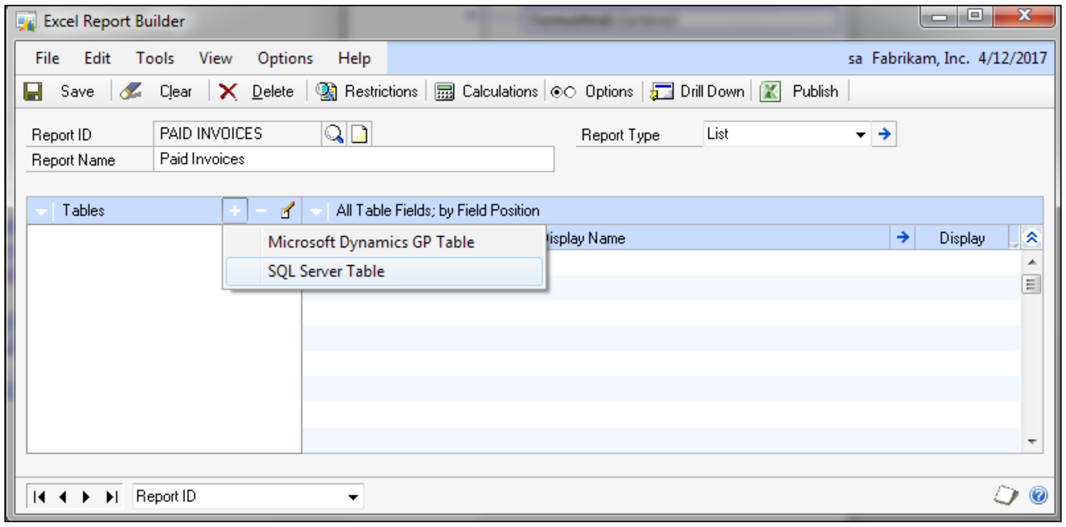
Databases: DYNAMICS, master, model, msdb, ReportServer, ReportServerTempDB, tempdb, ✓ TWO

Views: SVC6100V, SVC6105V, SVC6300V, SVCSOCOV, TaxDetailTransactions, test, UPR50000, VendorAddress, VendorItems, Vendors, ✓ vw_PaidInvoices, WorkOrders

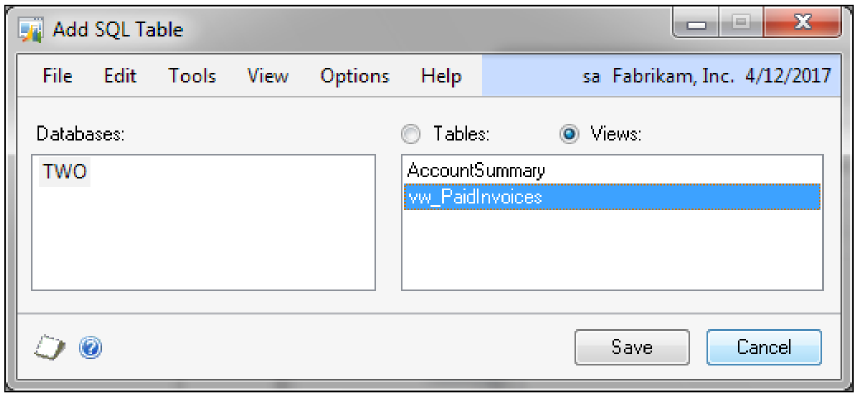
   - Select the database `TWO` from the **Databases:** list
   - Select the **Views** radio button
   - Select **vw_PaidInvoices**
   - Click on **OK**

3. Open Excel Report Builder from the **Microsoft Dynamics GP** menu by navigating to **Tools** | **SmartList Builder** | **Excel Report Builder** | **Excel Report Builder**.
4. Assign **PAID INVOICES** as **Report ID**.
5. Name the Excel Report `Paid Invoices`.
6. Select **List** as **Report Type**.

7. Add the SQL view as follows:
   - Click on the **Add Table** button (plus (**+**) sign next to **Tables**) in the lower-left pane.

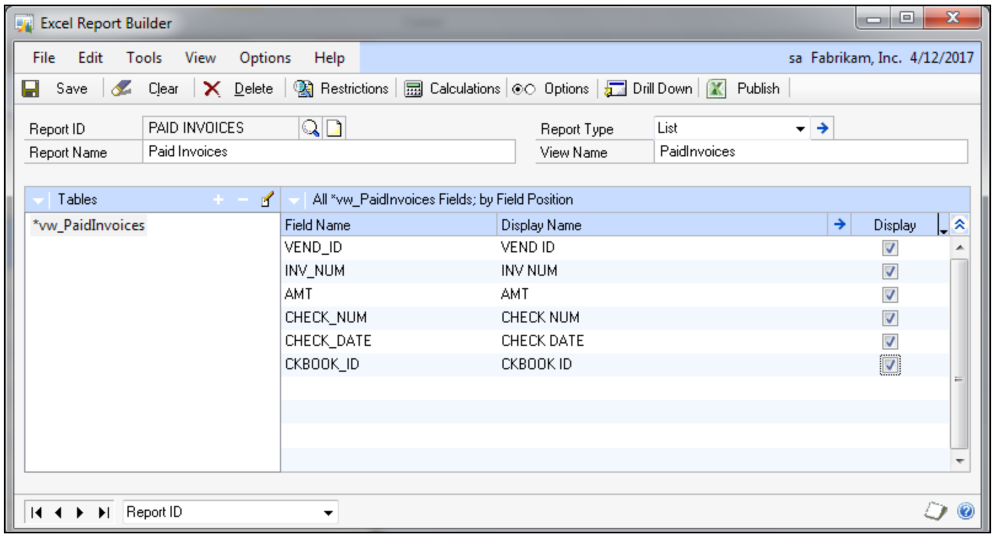

Excel Report Builder
File Edit Tools View Options Help
sa Fabrikam, Inc. 4/12/2017
Save Clear Delete Restrictions Calculations Options Drill Down Publish
Report ID PAID INVOICES
Report Name Paid Invoices
Report Type List
Tables
All Table Fields; by Field Position
Microsoft Dynamics GP Table
SQL Server Table
Display Name
Display
Report ID

   - Select **SQL Server Table** to open the **Add SQL Table** window. Highlight the `TWO` database, select the **Views** radio button, and highlight the `vw_PaidInvoices` view.



   - Click on **Save**.

8. Put a checkmark in the **Display** column for all six fields.

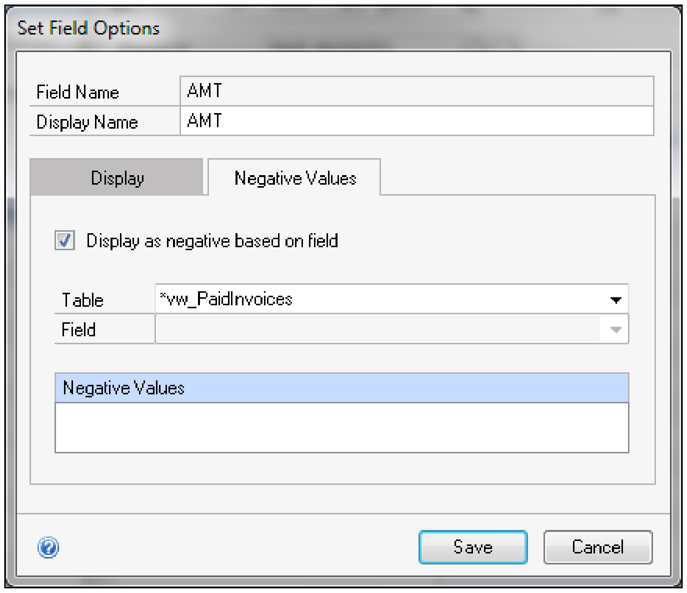

Excel Report Builder
File Edit Tools View Options Help
sa Fabrikam, Inc. 4/12/2017
Save Clear Delete Restrictions Calculations Options Drill Down Publish
Tables
All *vw_PaidInvoices Fields; by Field Position
Report ID

Report ID	PAID INVOICES	Report Type	List
Report Name	Paid Invoices	View Name	PaidInvoices

Tables: *vw_PaidInvoices

Field Name	Display Name	Display
VEND_ID	VEND ID	✓
INV_NUM	INV NUM	✓
AMT	AMT	✓
CHECK_NUM	CHECK NUM	✓
CHECK_DATE	CHECK DATE	✓
CKBOOK_ID	CKBOOK ID	✓

9. Make the necessary changes to the column **Display** formats as follows:
   - Highlight the **AMT** field, and click on the blue arrow on the **Display** column to open the **Set Field Options** window



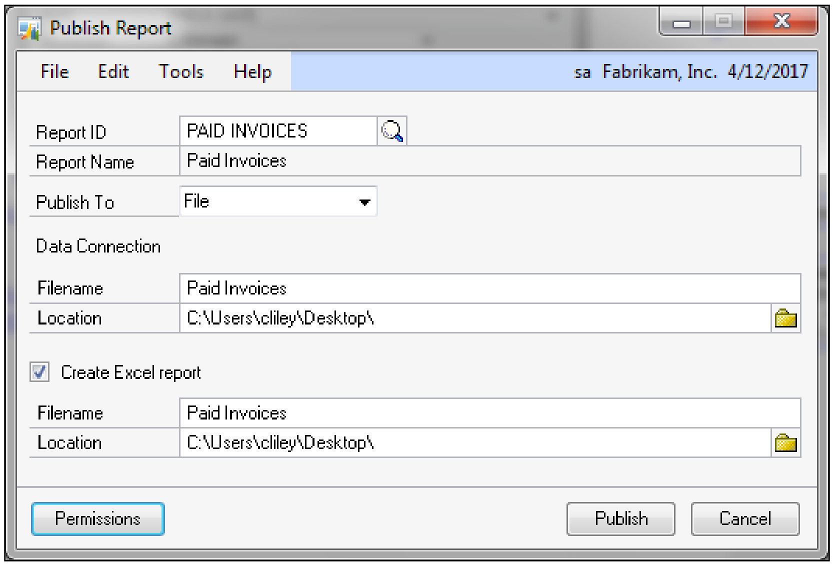
Field Name: AMT
Display Name: AMT

✓ Display as negative based on field

Table: *vw_PaidInvoices
Field:

- On the **Negative Values** tab, mark the checkbox for **Display** as negative based on field
- Click on **Save**

10. Save the Excel Report.
11. Publish the Excel Report as follows:
    - Click on the **Publish** button to open the **Publish Report** window

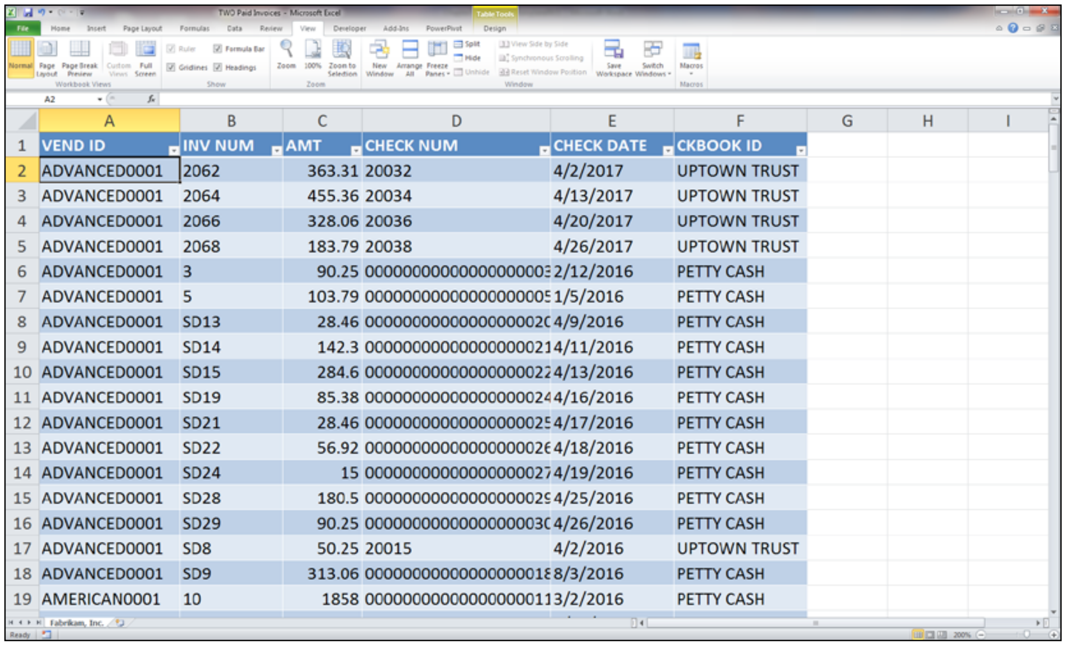


Report ID: PAID INVOICES
Report Name: Paid Invoices
Publish To: File
Data Connection
Filename: Paid Invoices
Location: C:\Users\cliley\Desktop\
☑ Create Excel report
Filename: Paid Invoices
Location: C:\Users\cliley\Desktop\

- Select **File** as **Publish To**, and set both the **Data Connection** and Excel Report locations to your desktop
- Click on **Publish**

12. Browse to your desktop and open the `TWO Paid Invoices.xlsx` file.

> If data connections have been disabled, click on **Enable Content**.



	A	B	C	D	E	F
1	VEND ID	INV NUM	AMT	CHECK NUM	CHECK DATE	CKBOOK ID
2	ADVANCED0001	2062	363.31	20032	4/2/2017	UPTOWN TRUST
3	ADVANCED0001	2064	455.36	20034	4/13/2017	UPTOWN TRUST
4	ADVANCED0001	2066	328.06	20036	4/20/2017	UPTOWN TRUST
5	ADVANCED0001	2068	183.79	20038	4/26/2017	UPTOWN TRUST
6	ADVANCED0001	3	90.25	0000000000000000000003	2/12/2016	PETTY CASH
7	ADVANCED0001	5	103.79	0000000000000000000005	1/5/2016	PETTY CASH
8	ADVANCED0001	SD13	28.46	0000000000000000000020	4/9/2016	PETTY CASH
9	ADVANCED0001	SD14	142.3	0000000000000000000021	4/11/2016	PETTY CASH
10	ADVANCED0001	SD15	284.6	0000000000000000000022	4/13/2016	PETTY CASH
11	ADVANCED0001	SD19	85.38	0000000000000000000024	4/16/2016	PETTY CASH
12	ADVANCED0001	SD21	28.46	0000000000000000000025	4/17/2016	PETTY CASH
13	ADVANCED0001	SD22	56.92	0000000000000000000026	4/18/2016	PETTY CASH
14	ADVANCED0001	SD24	15	0000000000000000000027	4/19/2016	PETTY CASH
15	ADVANCED0001	SD28	180.5	0000000000000000000029	4/25/2016	PETTY CASH
16	ADVANCED0001	SD29	90.25	0000000000000000000030	4/26/2016	PETTY CASH
17	ADVANCED0001	SD8	50.25	20015	4/2/2016	UPTOWN TRUST
18	ADVANCED0001	SD9	313.06	0000000000000000000018	8/3/2016	PETTY CASH
19	AMERICAN0001	10	1858	0000000000000000000011	3/2/2016	PETTY CASH

Now we have our refreshable Excel Report, as can be seen in the preceding screenshot that we can deliver to our user and they can simply open the spreadsheet every month and refresh the data without ever touching the Dynamics GP application.

Hopefully, this real life example has demonstrated how easy it is to utilize the power and flexibility of the SQL server views and Excel Report Builder.

Additional tips and tricks for using both Builders

Although we've covered the basics of creating new SmartList Objects and Excel Reports via SmartList Builder and Excel Reports Builder respectively, numerous tips and things to watch out for exist that can improve the functionality and usefulness of these tools even further. Although we would love to talk about advanced tips and techniques, such as GoTos, Drill throughs, and importing and exporting customized reports, we simply don't have enough space to cover them all! We strongly encourage readers to take advantage of numerous resources, including the SmartList Builder User Guide and the Microsoft Support site. The SmartList Builder User guide can be found online (check CustomerSource or PartnerSource) while the Support site is found at `http://www.support.microsoft.com`. These are great resources to use once you've learned some of the SmartList and Excel Report Builder basics from this book!

Summary

In this chapter, we introduced the first four tools in our reporting toolbox: SmartList for easy-on-demand ad-hoc reporting, SmartList Builder for times when the default SmartLists just don't cut it and we need to build our own SmartList Objects, Excel Reports for viewing SmartList like data in Excel, and Excel Report Builder for when we want to extend the concept of SmartList Builder into Excel and build our own refreshable Excel Reports and/or Office Data Connections.

We began with the basic structure of each tool and what it does and ended with some real life, step-by-step examples of how to use the builder tools. Finally, we looked at some tips and tricks when using the SmartList Builder and Excel Report Builder tools.

In the next chapter, we will dive into Report Writer, which has been the backbone of reporting in Dynamics GP for many years. We'll begin by understanding what Report Writer is, before looking at using the tool to customize reports. We will then explore the Word Template and Word Template Generator features that were new in Dynamics GP 2010 and later versions.

4
Report Writer and Word Templates

In the previous chapter, we explored our first four reporting tools. These tools were mainly used for mining Dynamics GP data and providing us with lists of data. After starting with the structure of each tool and describing how to use them, we gave some real-life examples to hone in on how each tool could be used to complete the example.

In this chapter, we move on to the Report Writer tool. Report Writer has been the underlying basis for reporting in Dynamics GP for many years. Every module in Dynamics GP has standard reports that originate from Report Writer. Many of these are posting journals that print when transactions are posted in the system. There are also reports such as activity reports, trial balances, modules setups, and so on. Any of these standard reports can be modified with the Report Writer tool.

Some examples of situations where we might find the need to modify these standard Dynamics GP reports could be adding aging periods to the Receivables or Payables trial balances, customizing label fields on a report, modifying a check format, adding a logo to a report or removing fields that are unnecessary on a particular report. Many other reasons exist for modifying a standard report.

This chapter will delve into the following areas within Dynamics GP Report Writer:

- Understanding the reports dictionaries and how Dynamics GP treats original reports versus modified reports
- Opening and navigating to the Report Writer windows
- Using Global Modifications to modify all reports in the application
- Modifying an existing Dynamics GP report
- Importing and exporting customized reports
- Utilizing the new Dynamics GP 2013 Word template feature to render reports in Microsoft Word

Storing and accessing Report Writer reports

Before we begin using Report Writer, it is important that we have a basic understanding of the terminology used in the tool as well as how the reports are stored in the system. Report Writer has three types of reports:

- **Original reports**: These are the default reports that are provided with the accounting system.
- **Modified reports**: These are copies of an original report that have had changes made in Report Writer. Modified reports can be used instead of original reports in the accounting system.
- **Custom reports**: These are reports that are created using Report Writer. These can be started with a blank report or by making a copy of an existing report. Stored procedures must be used to print custom reports in the accounting system.

Storing Report Writer reports

Report Writer for Dynamics GP 2013 stores reports in different files depending on which type of report it is. All of the original reports are stored in the main application dictionaries. For instance, Dynamics GP reports are stored in `Dynamics.dic`. In the following table, we can see some of the other more common products and which dictionary their reports are stored in:

Product	**Dictionary**
Project Accounting	PA258.dic
Fixed Assets	FAM.dic
Manufacturing	ICONMFG.dic
Human Resources	HR.dic
Field Service	SRVSADV.dic

All changes and/or additions that are made in Report Writer are stored in the reports dictionary for the corresponding application. Not every application will have a reports dictionary; in fact, they are only created when changes are made. For example, modifications to standard Dynamics GP reports are stored in a dictionary called `Reports.dic`. Let's take a look in the following table for the corresponding report dictionaries for each product presented in the previous table:

Product	**Dictionary**
Project Accounting	PAREPT.dic
Fixed Assets	R309.dic
Manufacturing	ICONRPTS.dic
Human Resources	HRRPTS.dic
Field Service	RPTS949.dic

This method of storage maintains the integrity of the main application dictionary, meaning we never lose our original reports and we can always revert back to the original, if necessary. Additionally, GP allows us to choose whether or not to print the original or modified version of a report. Essentially, when we create a report or select a report to modify, the report is copied into the reports dictionary.

There are two common configurations for storing the report dictionaries when using Report Writer. One configuration is to store the reports dictionaries locally on each workstation. The other configuration is to store the dictionaries on a network share that is accessible by all client workstations. The advantages and disadvantages of each can be seen in the following tables.

The advantages and disadvantages of storing reports dictionaries locally are as follows:

Advantages	**Disadvantages**
Each workstation can utilize a different set of reports. For example, one user/workstation may require sales invoices to be formatted one way while another user/workstation may require it done in a different way.	Report modifications that will be used by more than one user/workstation must be distributed to all other workstations that will require the same report modification.
In the event of a corrupt reports dictionary on one workstation, it is more likely that a reports dictionary from another workstation will be available as a backup.	Numerous, highly specialized reports dictionaries require a greater effort, and attempts to maintain backup copies of each report dictionary.
Multiple users can access Report Writer at any given point in time.	

The advantages and disadvantages of storing reports dictionaries on a network share are as follows:

Advantages	**Disadvantages**
As new reports are created, or modifications are made to existing reports, all users with the appropriate security permissions will have instant access to these reports. We no longer have to go through the process of rolling out changes to each workstation	Access to the Report Writer application is limited to a single user at a time. This is not usually an issue in smaller organizations, but can become quite troublesome in larger organizations
All users can access the same bank of new and modified reports	In the event that the reports dictionary becomes corrupt, all users are affected until a copy of the reports dictionary is restored
	Users are limited to the number of modified and/or new reports that can be used at any given time

When deciding which configuration to choose based on these advantages and disadvantages, take into consideration whether our users will be creating or modifying their own reports or not. Also consider how we plan to share the customized reports among users. These factors play a critical role in our decision-making.

Accessing Report Writer reports

Once we've made our decision on how the reports dictionaries will be stored and begin modifying our first reports, we have two additional steps that must be taken when working with modified reports. These steps will allow us to access our modified reports from GP:

- Setting the Dynamics GP Launch file
- Setting security to custom/modified reports

Now, let's see what each step entails so that we can start using our modified reports!

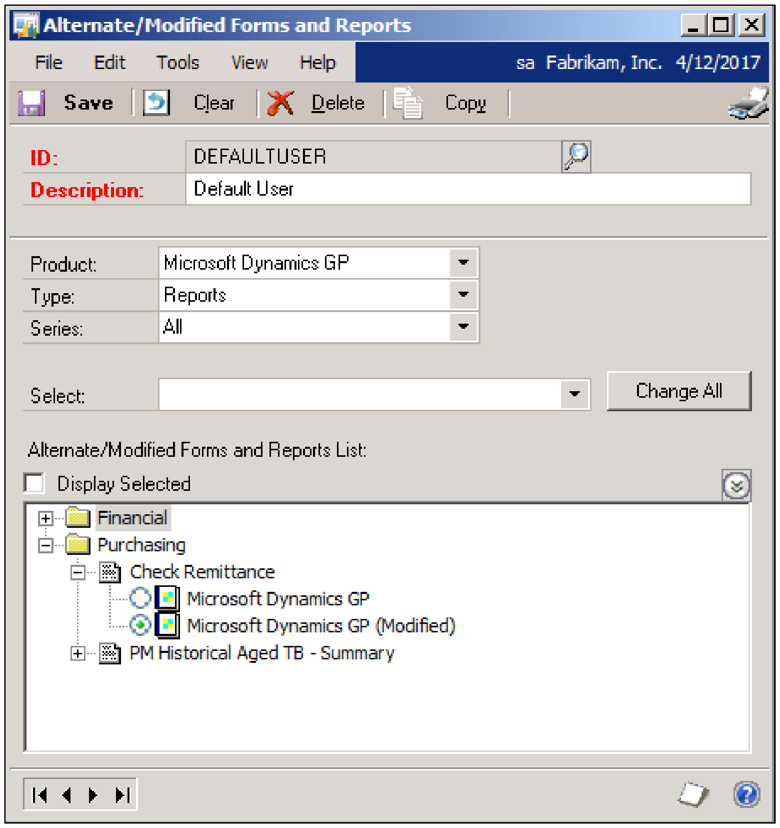
Setting the Dynamics GP launch file

When Dynamics GP is started, the launch file is what tells the runtime engine which dictionaries to use. In addition to providing a list of installed products for this instance, the launch file also lists the file location for the various application, form, and report dictionaries required by GP 2013. By default, these will be stored in the `Data` folder of the Dynamics GP installation. This file can be edited in system setup by administrators or power users, or by opening the `Dynamics.set` file in the `application` folder. If we have chosen a configuration that stores these dictionaries on a network share, we will need to edit the default `Reports.dic` path in the launch file to point to the new location on the network share.

We must take care that we have correctly specified the names and locations; otherwise Dynamics GP may not start up properly or users might receive dictionary-related error messages when attempting to generate reports that have been modified.

Setting security for custom/modified reports

Once we have set our launch file to the correct path for our modified reports, in order for the application to actually use these reports, we must use the **Alternate/Modified Forms and Reports** window inside Dynamics GP. To open this window, as seen in the following screenshot, navigate to **Microsoft Dynamics GP** | **Tools** | **Setup** | **System** | **Alternate/Modified Forms and Reports**.


Alternate/Modified Forms and Reports
File Edit Tools View Help
sa Fabrikam, Inc. 4/12/2017
Save
Clear
Delete
Copy
Change All
Alternate/Modified Forms and Reports List:
Display Selected

ID: DEFAULTUSER
Description: Default User

Product: Microsoft Dynamics GP
Type: Reports
Series: All
Select:

- Financial
- Purchasing
  - Check Remittance
    - Microsoft Dynamics GP
    - Microsoft Dynamics GP (Modified)
  - PM Historical Aged TB - Summary

To set access, we use the following steps:

1. Specify an **ID**. This **ID** will be attached to specific users in the user security setup. This can allow us to give different users access to different modified reports.
2. Select the **Product** containing the modified report. Selecting **Microsoft Dynamics GP**, for example, will show us a list of objects stored in the dictionary for the core Dynamics GP product.

3. Choose to display **Reports** in the **Type** drop-down.
4. Although we don't need to select anything here, notice the new **Select** drop-down. This drop-down provides new functionality to quickly modify settings for multiple reports at once, either by setting them back to the original version or changing them to another version.
5. Locate the modified report by browsing through the tree view. These reports are grouped by the module in which they are used.
6. Once the modified report has been located, change the selection from the default version to the modified version (in most cases, we will see the word **Modified** to designate the customized report version).
7. Click on **Save**.

After we have set the appropriate security, any users with the Alternate/Modified Forms and Report ID that we have modified will have the ability to print the modified versions of the reports in place of the original versions.

If we ever need to remove a modified version of a report, we need to be sure to set the security back to use the original version of the report; otherwise Dynamics GP will not be able to properly access the reports.

Opening and navigating the Report Writer windows

As with SmartList Builder and Excel Reports Builder, the Report Writer application can be accessed from within Dynamics GP. Using the Report Writer application requires switching from the main GP 2013 windows to the Report Writer interface with its own set of buttons and menu layouts. Let's take a look at how we can open Report Writer and then explore some of the various buttons and menus that appear in this application.

Setting security permissions to use Report Writer

Before we can use the Report Writer application, we must check to ensure that we have the appropriate security permissions to do so. This, of course, is not necessary for GP logins that are assigned to the POWERUSER role, as these users, by default, have access to the Report Writer application.

Standard GP 2013 security can be used to grant access to the Report Writer application. Users must be granted access to the `ADMIN_SYSTEM_008` security task in order to open Report Writer from GP. This security task should be assigned to a security role which, in turn, should be assigned to the particular user who needs access to Report Writer.

If a wide variety of users on the same system will be accessing Report Writer, we recommend setting up a new security role for Report Writer. Assign the ADMIN_SYSTEM_008 task to this role. This new role can then be assigned to all of the users who require access to Report Writer. This is much easier to remember rather than assigning the task to a wide variety of roles!

Opening Report Writer

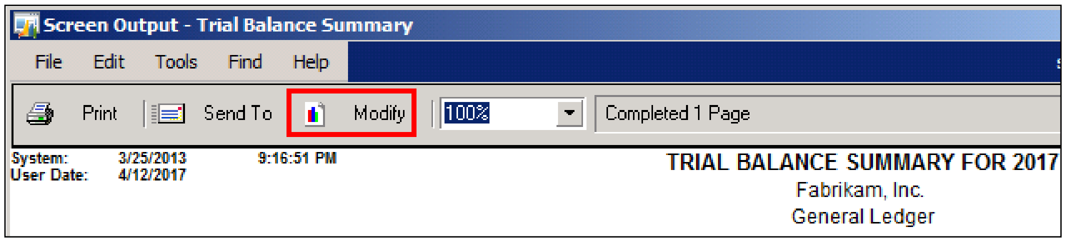
The Report Writer application can be opened in one of several ways. Users can open Report Writer and select a report to modify, or users can run a report in GP and select to modify the report from the screen output. This will open Report Writer with the selected report already on display in the application.

To open Report Writer without first selecting a report, do the following:

1. Open the **Microsoft Dynamics GP** menu.
2. Navigate to **Tools** | **Customize** | **Report Writer**. Alternatively, users can press *Alt* + *F9* to perform this action.
3. A pop-up window will appear asking users to select the **Product**. Select **Microsoft Dynamics GP** to open Report Writer with a connection to the main product dictionary. This will allow you to modify reports stored in this dictionary. To modify reports in other dictionaries, such as Manufacturing, we can select that **Product** at this time.
4. After selecting a **Product, click on OK** to allow Microsoft Dynamics GP to switch to the Report Writer application.
5. Users must then click on the **Reports** button to choose from a list of reports in this product dictionary that can be modified.

As another option to open the Report Writer application and select a report from a list, users can also open Report Writer with a report already printed to the screen. This can be done using the **Summary Trial Balance** report from the **Financials** module:

1. Navigate to **Reports** | **Financial** | **Trial Balance** to open the **Trial Balance Report** window.
2. Select **Summary** in the **Reports:** drop-down box.
3. Select an existing report option and select **Insert** to add this report to the **Print List**. The selected report should print to the screen.
4. Click on **Print** to print the report to screen.
5. When the **Screen Output** window for the selected report appears, note the name of the report as it appears in the blue bar at the top of the output window. This is the name of the report as it will appear in Report Writer.
6. Click on the **Modify** button at the top of the window (highlighted in the screenshot below) to open Report Writer for this report. Alternatively, this same action can be completed by pressing *Ctrl* + *F9* on the keyboard:



System: 3/25/2013 9:16:51 PM
User Date: 4/12/2017

TRIAL BALANCE SUMMARY FOR 2017
Fabrikam, Inc.
General Ledger

7. Report Writer will open with the **Trial Balance Summary** report available for editing.

Report Writer Resource windows

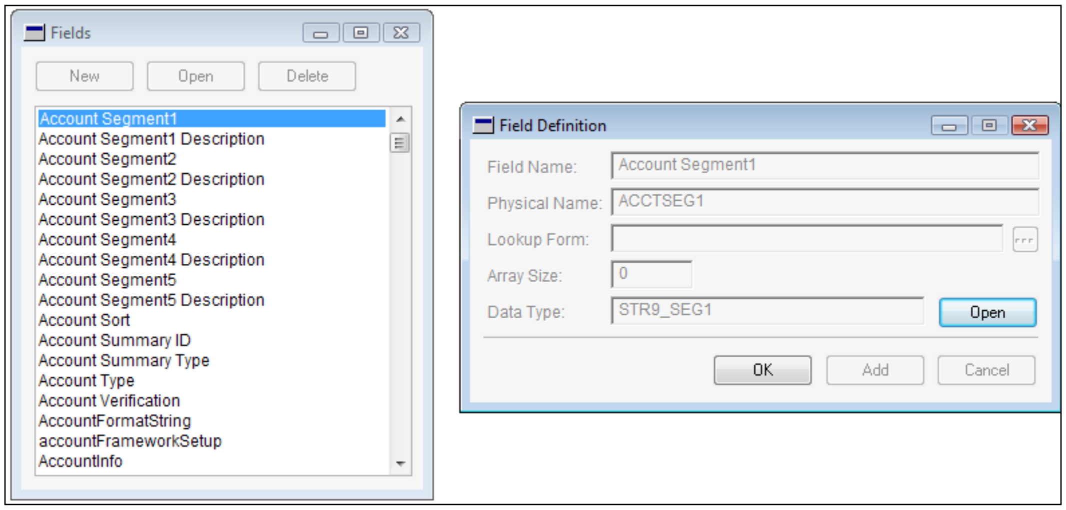
Numerous windows can be accessed from within Report Writer that can provide users with more information about the resources that can be used in the application. Many of these windows are accessible from the **Resources** menu found in the Report Writer menu bar. Since one of the most basic resources in Report Writer is the field, we'll begin with that window.

Fields

Navigating to **Resources | Fields** opens the **Fields** window. Alternatively, users can click on the **Fields** button from the toolbar. The window that appears displays a list of global fields that can be used in the Report Writer application.

Double-click on a field name to open the **Field Definition** window for that global field. This window provides users with the physical name of the field as well as the data type referenced by the selected field.

These windows are seen in the following screenshot:



Account Segment1
Account Segment1 Description
Account Segment2
Account Segment2 Description
Account Segment3
Account Segment3 Description
Account Segment4
Account Segment4 Description
Account Segment5
Account Segment5 Description
Account Sort
Account Summary ID
Account Summary Type
Account Type
Account Verification
AccountFormatString
accountFrameworkSetup
AccountInfo

Field Name: Account Segment1
Physical Name: ACCTSEG1
Lookup Form:
Array Size: 0
Data Type: STR9_SEG1

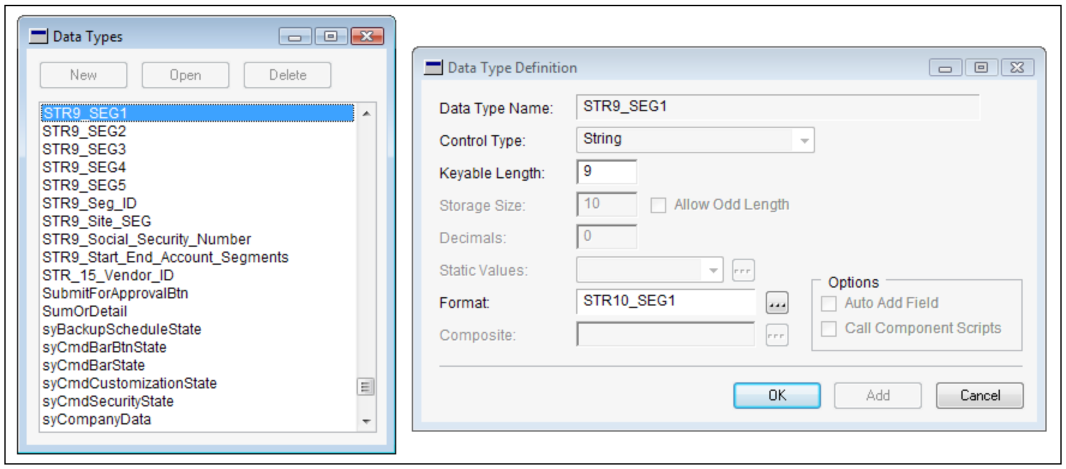
Clicking on the **Open** button from the **Field Definition** window takes us to the **Data Types Definition** window.

Data types

To open the **Data Types** window, navigate to **Resources | Data Types** from the menu bar. Additionally, users can click on the **Data Types** button from the toolbar.

The **Data Types** window displays a list of the data types used by the various fields found in Report Writer. As we saw previously, each global field in Report Writer is assigned a corresponding data type.

With a data type selected in the lookup list, click on the **Open** button to open the **Data Type Definition** window and see more information about the selected data type.



STR9_SEG1
STR9_SEG2
STR9_SEG3
STR9_SEG4
STR9_SEG5
STR9_Seg_ID
STR9_Site_SEG
STR9_Social_Security_Number
STR9_Start_End_Account_Segments
STR_15_Vendor_ID
SubmitForApprovalBtn
SumOrDetail
syBackupScheduleState
syCmdBarBtnState
syCmdBarState
syCmdCustomizationState
syCmdSecurityState
syCompanyData

Data Type Name:	STR9_SEG1
Control Type:	String
Keyable Length:	9
Storage Size:	10
Allow Odd Length	☐
Decimals:	0
Static Values:	
Format:	STR10_SEG1
Composite:	
Auto Add Field	☐
Call Component Scripts	☐

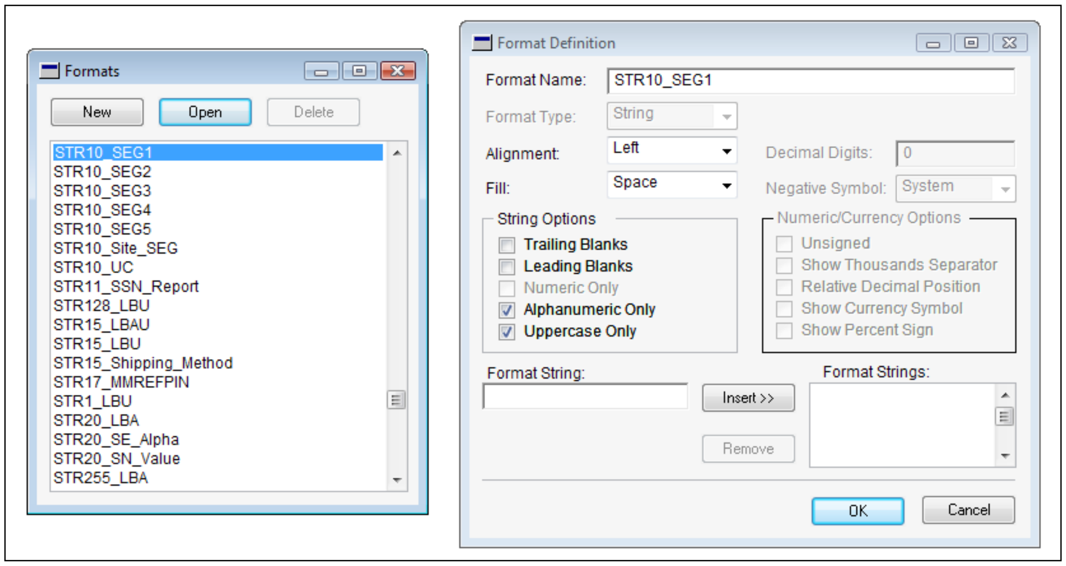
We can use data types to control the **Keyable Length** of a field as well as the formatting for a field. Don't forget that it can be tempting to change the **Keyable Length** or **Format** values for a particular data type with a specific field in mind; however, a single data type can be assigned to multiple fields! Changing attributes such as the selected format for a single data type will change those attributes for all fields assigned to that data type!

To change the format type assigned to a particular data type, click on the ellipses button next to the **Format** field to open the **Format Lookup** window with the specified format already selected. Double-click on a format value in the **Format Lookup** window to apply that formatting type to the selected data type.

Formats

The **Formats** window returns a list of formats that are available for this report dictionary. As we might expect, the formats listed in this window govern how data types and, in turn, fields, will appear on Report Writer-based reports.
To see the list of formats that already exist in Report Writer, users can navigate to **Resources** | **Formats** from the menu bar.

To learn more about an existing format, users can click on the **Open** button to open the **Format Definition** window. If this window has not been opened from the **Data Types Definition** window (as we described earlier in this section), double-clicking on a format name will also open the **Format Definition** window. From here, we can view a list of attributes controlling how this format will be visible to the user. Depending on the value selected in the **Formats** window, certain areas of this window may be greyed out and be unavailable for selection.



STR10_SEG1
STR10_SEG2
STR10_SEG3
STR10_SEG4
STR10_SEG5
STR10_Site_SEG
STR10_UC
STR11_SSN_Report
STR128_LBU
STR15_LBAU
STR15_LBU
STR15_Shipping_Method
STR17_MMREFPIN
STR1_LBU
STR20_LBA
STR20_SE_Alpha
STR20_SN_Value
STR255_LBA

Format Name: STR10_SEG1
Format Type: String
Alignment: Left
Fill: Space
Decimal Digits: 0
Negative Symbol: System

String Options
- [ ] Trailing Blanks
- [ ] Leading Blanks
- [ ] Numeric Only
- [✓] Alphanumeric Only
- [✓] Uppercase Only

Numeric/Currency Options
- [ ] Unsigned
- [ ] Show Thousands Separator
- [ ] Relative Decimal Position
- [ ] Show Currency Symbol
- [ ] Show Percent Sign

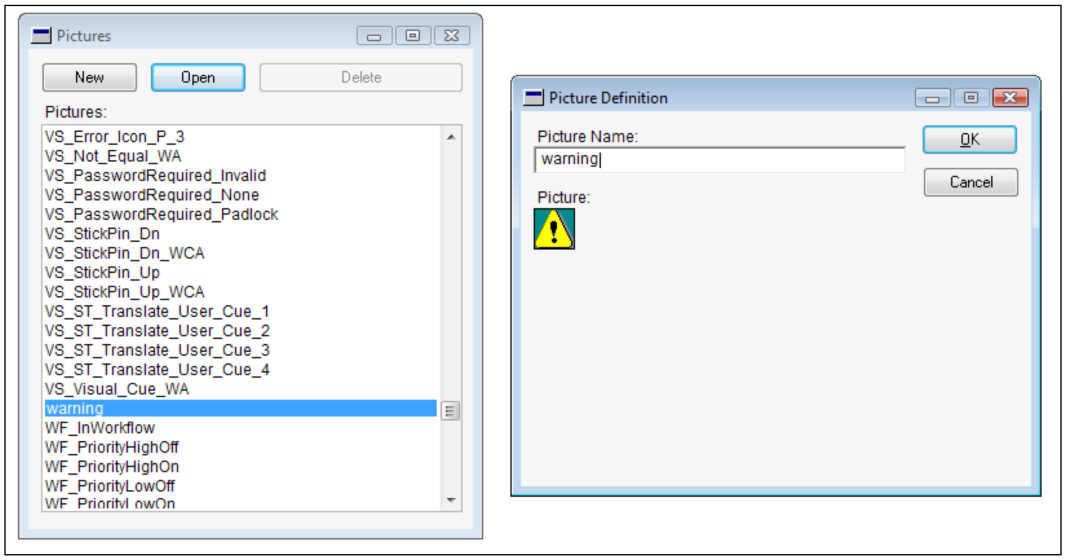
Format String:
Format Strings:

The **Formats** window is unique to the windows we have covered thus far as it allows users to create new resources for use in Report Writer. Clicking on the **New** button opens the **Format Definition** window with blank fields. Users can enter a new **Format Name** and choose from several settings to create a new format definition for use in Report Writer. This format definition can then be assigned to different data types.

Pictures

Navigating to **Resources | Pictures** opens the Pictures Library list. In this list, users can see all of the pictures that are available for selection in the Report Writer application. After a picture is added to the Picture Library, it can be used on multiple graphical reports.

Users can select a picture from the list and click on the **Open** button or double-click on the picture to open the **Picture Definition** window. This window displays the selected picture, such as the warning icon seen in the following screenshot:



Pictures:
VS_Error_Icon_P_3
VS_Not_Equal_WA
VS_PasswordRequired_Invalid
VS_PasswordRequired_None
VS_PasswordRequired_Padlock
VS_StickPin_Dn
VS_StickPin_Dn_WCA
VS_StickPin_Up
VS_StickPin_Up_WCA
VS_ST_Translate_User_Cue_1
VS_ST_Translate_User_Cue_2
VS_ST_Translate_User_Cue_3
VS_ST_Translate_User_Cue_4
VS_Visual_Cue_WA
warning
WF_InWorkflow
WF_PriorityHighOff
WF_PriorityHighOn
WF_PriorityLowOff
WF_PriorityLowOn

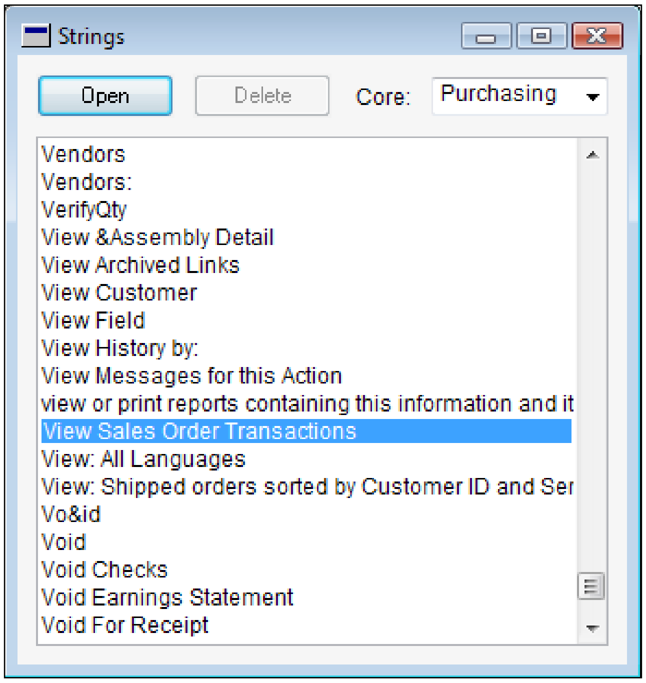
Picture Name: warning

Picture:

Later on in this chapter, we will cover some tips and tricks involving this window as we add a logo to one of our reports.

Strings

The **Strings** window can be opened by navigating to **Resources** | **Strings** from the menu bar. String values are used throughout Report Writer and Dynamics GP as text fields. The resulting list shows us the various string values that exist in the current dictionary:

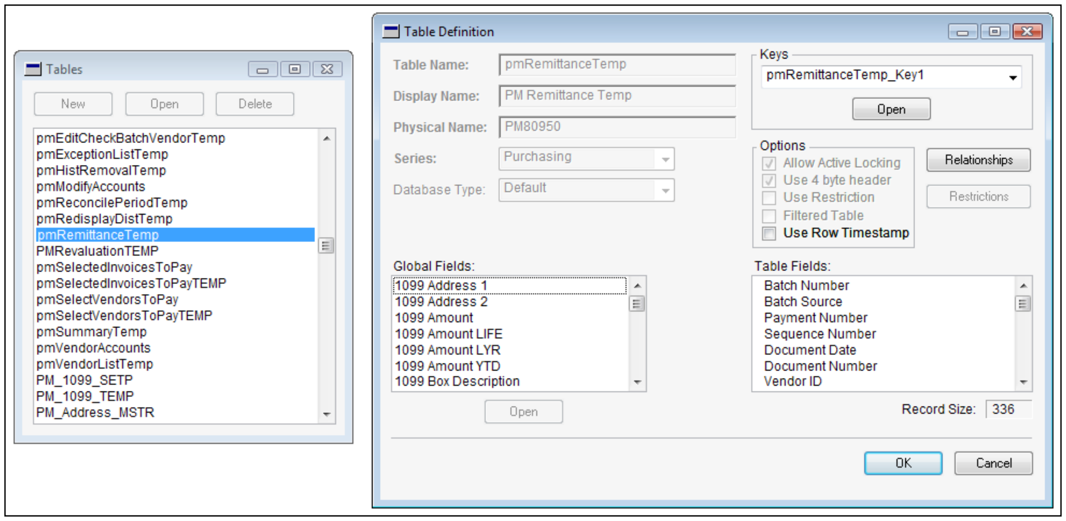


Core: Purchasing

Vendors
Vendors:
VerifyQty
View &Assembly Detail
View Archived Links
View Customer
View Field
View History by:
View Messages for this Action
view or print reports containing this information and it
View Sales Order Transactions
View: All Languages
View: Shipped orders sorted by Customer ID and Ser
Vo&id
Void
Void Checks
Void Earnings Statement
Void For Receipt

To see the strings broken down by series, change the value in the drop-down box at the upper right-hand corner of this window.

Clicking on the **Tables** button and the **Tables** option from the menu bar opens the **Tables** lookup. From here, we can see a list of tables that exist for the selected dictionary. Selecting a table and clicking on **Open** or double-clicking on the table name opens the **Table Definition** window:



pmEditCheckBatchVendorTemp
pmExceptionListTemp
pmHistRemovalTemp
pmModifyAccounts
pmReconcilePeriodTemp
pmRedisplayDistTemp
pmRemittanceTemp
PMRevaluationTEMP
pmSelectedInvoicesToPay
pmSelectedInvoicesToPayTEMP
pmSelectVendorsToPay
pmSelectVendorsToPayTEMP
pmSummaryTemp
pmVendorAccounts
pmVendorListTemp
PM_1099_SETP
PM_1099_TEMP
PM_Address_MSTR

Table Name: pmRemittanceTemp
Display Name: PM Remittance Temp
Physical Name: PM80950
Series: Purchasing
Database Type: Default

Keys: pmRemittanceTemp_Key1

Options
- [✓] Allow Active Locking
- [✓] Use 4 byte header
- [ ] Use Restriction
- [ ] Filtered Table
- [ ] Use Row Timestamp

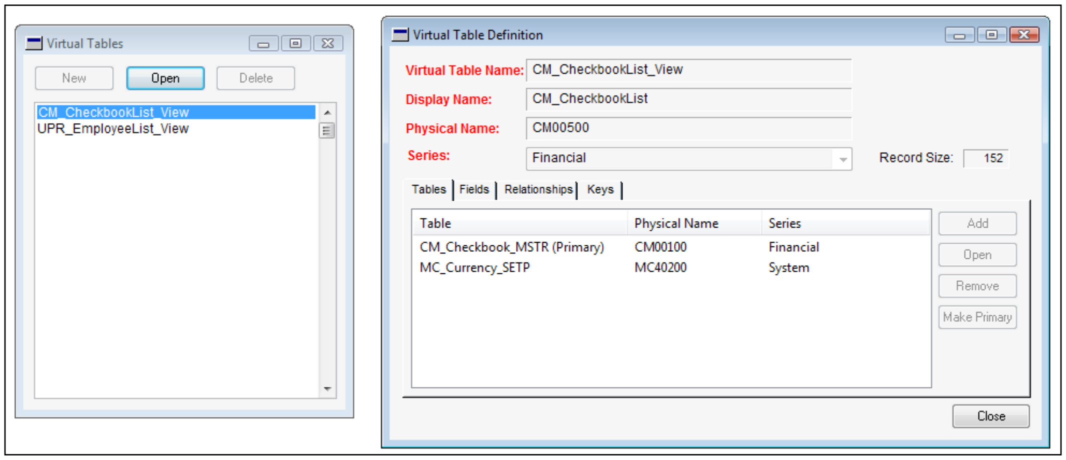
Global Fields:
1099 Address 1
1099 Address 2
1099 Amount
1099 Amount LIFE
1099 Amount LYR
1099 Amount YTD
1099 Box Description

Table Fields:
Batch Number
Batch Source
Payment Number
Sequence Number
Document Date
Document Number
Vendor ID

Record Size: 336

From the **Table Definition** window, we can see a list of fields that exist in this table. We can also click on the **Relationships** button to create relationships to other tables with similar fields. These relationships can be used in the report modification process to add additional fields to existing reports.

Also, notice that navigating to **Tables** | **Virtual Tables** from the menu bar opens a list of virtual tables in the current dictionary. Selecting a virtual table and clicking on **Open** or double-clicking on the table name opens the **Virtual Table Definition** window, as seen in the following screenshot with the **CM_Checkbook_List_View** virtual table selected:



Virtual Table Name: CM_CheckbookList_View
Display Name: CM_CheckbookList
Physical Name: CM00500
Series: Financial
Record Size: 152

Table	Physical Name	Series
CM_Checkbook_MSTR (Primary)	CM00100	Financial
MC_Currency_SETP	MC40200	System

Virtual tables are used as a way to combine data from multiple tables, similar to how a view might be used in SQL. Unfortunately, virtual tables can only be created by an application developer, so most of us will stick to creating table relationships via the **Table Definitions** window!

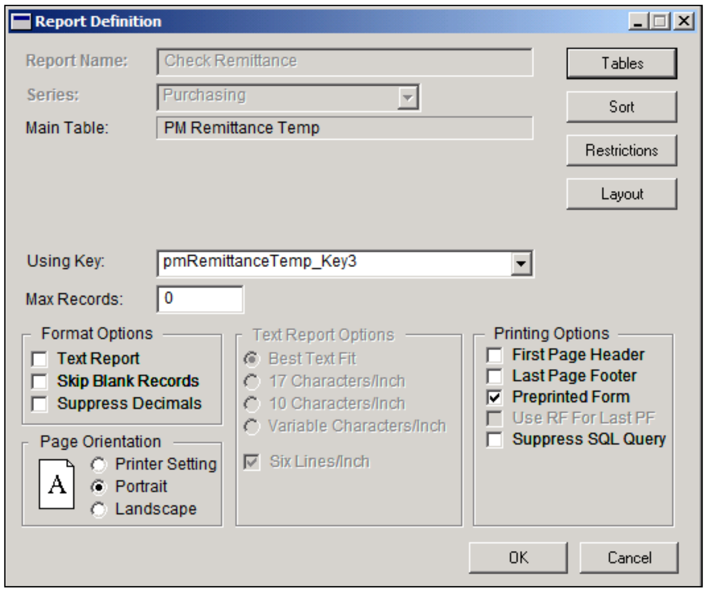
Report Modification windows

Earlier in this section, we discussed some of the windows accessible from the Resources menu as well as the Report Writer toolbar. Now, let's take a look at what the Report Writer layout looks like when we're actually modifying a report.

Report Definition

The **Report Definition** window is the first window that appears after a report is selected from the Reports list. From this table, we can gain access to other windows, such as the **Report Layout** window, that we will need in order to make modifications to a report.

The following screenshot shows the **Report Definition** window for the **Check Remittance** report that we will use later on in this chapter in the section on Word Template modifications:


Report Definition

Report Name: Check Remittance

Series: Purchasing

Main Table: PM Remittance Temp

Using Key: pmRemittanceTemp_Key3

Max Records: 0

Format Options
- [ ] Text Report
- [ ] Skip Blank Records
- [ ] Suppress Decimals

Page Orientation
- ○ Printer Setting
- ● Portrait
- ○ Landscape

Text Report Options
- ● Best Text Fit
- ○ 17 Characters/Inch
- ○ 10 Characters/Inch
- ○ Variable Characters/Inch
- [✓] Six Lines/Inch

Printing Options
- [ ] First Page Header
- [ ] Last Page Footer
- [✓] Preprinted Form
- [ ] Use RF For Last PF
- [ ] Suppress SQL Query

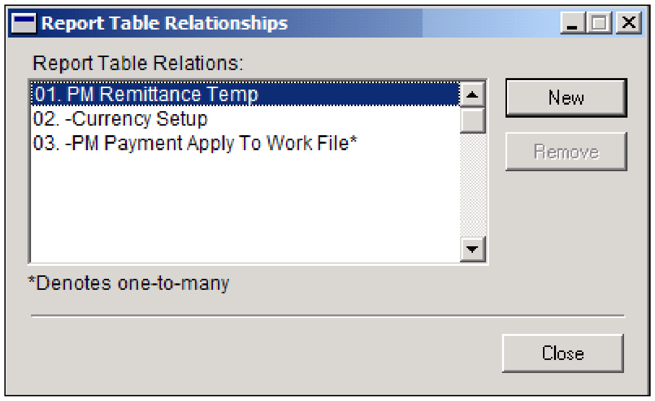
From this window, we can control several important settings for the selected report. For example, we can choose the page orientation, whether or not a page header will print on the first page or if a page footer should print on the last page, as well as select from a list of formatting options.

Elsewhere on the same window, the very important **Using Key** drop-down value allows us to control how data will be sorted in our report. If an appropriate key for sorting does not exist, we will have to create a sorting definition in the **Sorting Definition** window. We cover this window a little while later in this chapter.

Report Table Relationships

Clicking on **Tables** on the **Report Definition** window opens the **Report Table Relationships** window. We can use this window to add new source tables that should be used in the selected report.

The following screenshot shows a list of the table relationships for the Check Remittance report:



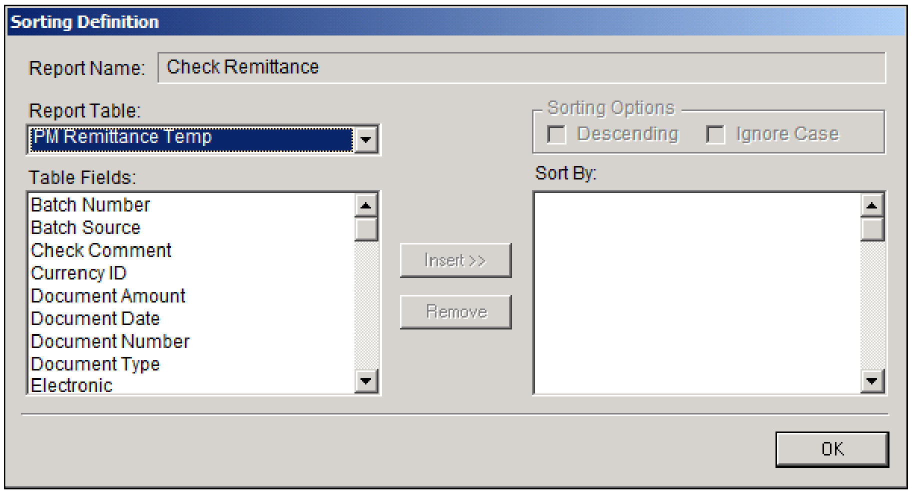
01. PM Remittance Temp
02. -Currency Setup
03. -PM Payment Apply To Work File*

*Denotes one-to-many

Before a table can be added via the **Report Table Relationships** window, a valid relationship must exist between the new table and one of the existing tables in the window. Use the **Table Definition** window, which is described in the **Strings** section of this chapter, to create these relationships.

Sorting Definition

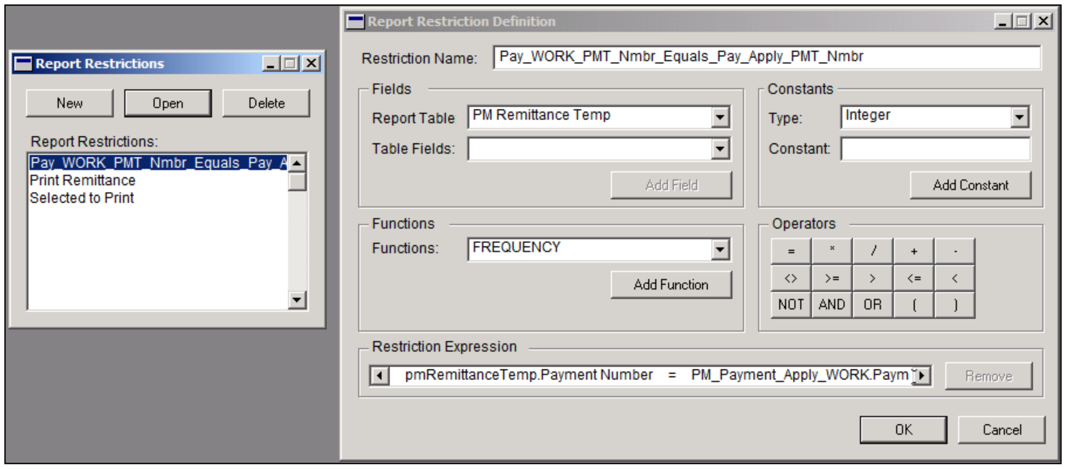
As we might expect, clicking on the **Sort** button on the **Report Definition** window opens the **Sorting Definition** window. In the next screenshot, we see a list of fields that we can select for sorting within the **Check Remittance** report:



The **Sorting Definition** window allows us to control the order in which data will be printed on our report. For most reports, an appropriate key for sorting will be defined on the **Report Definition** window, so this window will be unnecessary. In cases where a valid key cannot be used for sorting, we can use the **Sorting Definition** window to identify our own sorting requirements. This method is, however, less efficient than using the key value on the **Report Definition** window for sorting.

Report Restrictions

Clicking on **Restrictions** from the **Report Definition** window opens a list of restrictions that have been created for this report. This window can be used to add new restrictions to the report, or we can select an existing restriction to view the **Report Restriction Definition** window.



Report Restrictions list:
- Pay_WORK_PMT_Nmbr_Equals_Pay_A
- Print Remittance
- Selected to Print

Restriction Name: Pay_WORK_PMT_Nmbr_Equals_Pay_Apply_PMT_Nmbr
Report Table: PM Remittance Temp
Table Fields:
Type: Integer
Constant:
Functions: FREQUENCY
Restriction Expression: pmRemittanceTemp.Payment Number = PM_Payment_Apply_WORK.Paym

Report restrictions can be added to a report to control the output that appears when the report is actually printed.

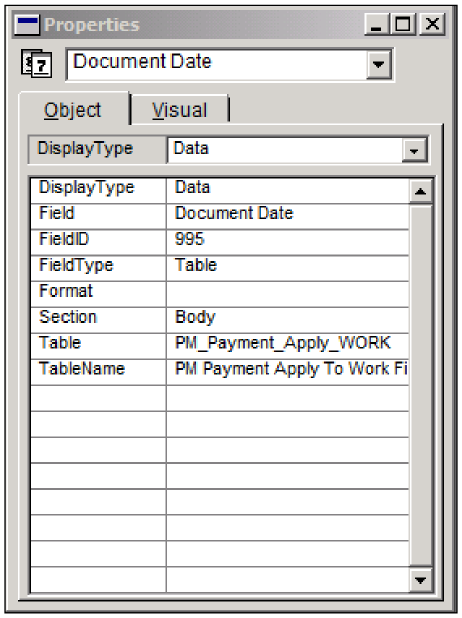
Layout

Clicking on the **Layout** button on the **Report Definition** window opens the window for the selected report. This is where the bulk of the work in Report Writer occurs, as it displays the various sections of the report and the various fields that are contained in each section.

Accompanying the **Report Layout** window are two smaller windows that change depending on which fields are selected in the **Report Layout**.

Properties

When a field is selected in the report layout, the **Properties** window displays information about that field. The **Object** tab identifies what kind of data the selected field represents. Switching to the **Visual** tab allows users to see how the field is formatted for this particular report. Here, in the following screenshot, we see the **Properties** window for the **Document Date** field in the Remittance Header section of the Check Remittance report:

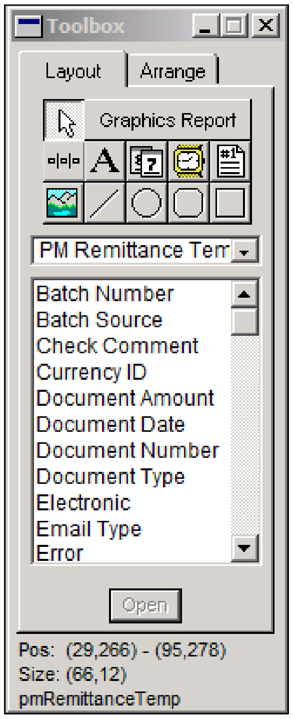


DisplayType	Data
Field	Document Date
FieldID	995
FieldType	Table
Format	
Section	Body
Table	PM_Payment_Apply_WORK
TableName	PM Payment Apply To Work Fi

Toolbox

The **Toolbox** window is where users can select fields to be added to the report layout. A drop-down box allows users to switch between tables that have been defined for this report. Selecting a different table or value in this drop-down will change the fields available for selection.

In addition to using the **Toolbox** to select fields, other objects can be selected for addition on the report layout. These objects range from shapes to text boxes to pictures. We can also click on the **Arrange** tab to display a list of options for arranging data in a selected cell.



Batch Number
Batch Source
Check Comment
Currency ID
Document Amount
Document Date
Document Number
Document Type
Electronic
Email Type
Error

Pos: (29,266) - (95,278)
Size: (66,12)
pmRemittanceTemp

Now that we've explored some of the windows that we will be using in Report Writer, let's move on to actually working with this tool!

Modifying all reports in the application by using global modifications

Although the most common use for Report Writer is to make modifications to individual reports, Dynamics GP Report Writer does allow for changes that affect all of the reports in a particular application. In this section, we will cover each of these global resources that can be modified in detail.

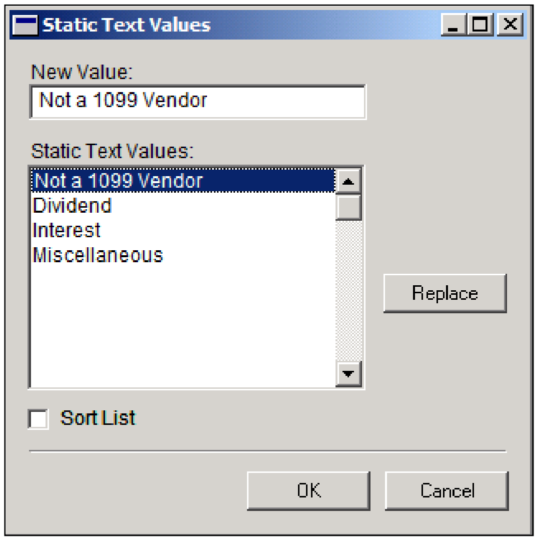
Data types

In addition to being able to adjust the **Keyable Length** as we discussed earlier, certain control types allow us to specify static values for the data type. An example of this would be the text on a push button or items included in a list box. Static text can be used in the following control types:

- **Drop-down list**: This indicates the selections in the drop-down
- **List box**: This indicates the selections in the list
- **Visual switch**: This displays two or more text values that will be displayed based on user clicks

To view/modify static text, do the following from within Dynamics Report Writer and follow these steps:

1. Navigate to **Resources** | **Data Types**.
2. Select a **Data Type**, **1099_Type** for instance.
3. Click on **Open**.
4. Click on the ellipsis button next to the **Static Text** field.
5. Highlight **Not a 1099 Vendor**, as shown in the following screenshot.



New Value: Not a 1099 Vendor

Static Text Values:
Not a 1099 Vendor
Dividend
Interest
Miscellaneous

☐ Sort List

6. Type `Non 1099` in the **New Value** field.
7. Click on **Replace**.
8. Click on **OK**.

Formats

Formats are the extra characters, spacing, and attributes that can be applied to a data type to format data when it is displayed or printed on reports. The easiest way to understand what formats are is to think of them as masks that just change the look of the information for a field without changing the data.

There are several formatting options available to change how data will appear:

- Data alignment
- String and composite formats
- Currency format
- Numeric format

These options are described in detailed tables in *Chapter 18*, *Formats* of the *Report Writer User Guide*.

To view/modify formats, do the following from within Dynamics Report Writer follow these steps:

1. Navigate to **Resources** | **Formats**.
2. Select a **Format**, **Phone_Number** for example.
3. Click on **Open**.
4. Select any of the options on the window and modify the **Format String** as necessary.
5. Click on OK.

One thing to keep in mind is that not all composites use a format string. Some are defined through the program code.

Pictures

Although Report Writer is not well known for exceptional graphics capabilities, we can still add logos and pictures to our modified reports. Before we can use these pictures, however, we must add them to the Report Writer picture library. Report Writer limits the file size of pictures to 32 KB. For larger logos and pictures, we can use a photo editor to resize the image, or we can crop the image into several smaller parts. These smaller images can then be combined in the Report Writer layout.

To add a new picture, do the following from within Dynamics Report Writer:

1. Navigate to **Resources** | **Pictures**.
2. Click on **New**.
3. Type a name in the **Picture Name** field.
4. With the picture in our clipboard, we choose **Paste** from the **Edit** menu or press *CTRL* + *V* to paste the picture.
5. Click on **OK** to add the picture to the library.

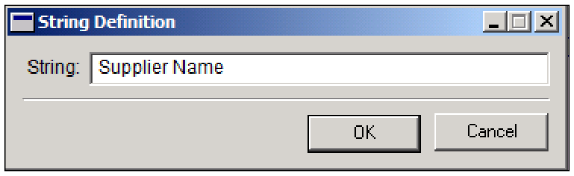
This functionality is extremely useful for adding company logos to our customer-facing reports!

Strings

We can use the strings resource to update all occurrences of a string in one step instead of changing the string everywhere it occurs. An example of this would be changing the words `Vendor Name` to `Supplier Name`. By changing the string, we don't have to change every individual text value for that field.

To modify a string, do the following from within Dynamics Report Writer and follow these steps:

1. Navigate to **Resources** | **Strings**.
2. Select the dictionary **Core** or series, **Purchasing** for example.
3. Highlight **Vendor Name**.
4. Click on **Open**.



String: Supplier Name

5. Type in the new value for the **String** name, `Supplier Name`, in our example.
6. Click on **OK**.

Modifying an existing Dynamics GP report

Although Dynamics GP Report Writer allows you to not only modify original reports but also create new reports, the focus of this book is on modifying existing reports. With the ever-expanding amount of reporting tools available, there are usually better options for creating reports from scratch than using Report Writer. Let's take a look at some of the common modifications made to reports before diving into a step-by-step example.

Common modifications

Modifying existing reports is the most common task performed with Dynamics Report Writer. *The Report Writer User's Guide* that is included with the GP 2013 installation lists some of the more common modifications:

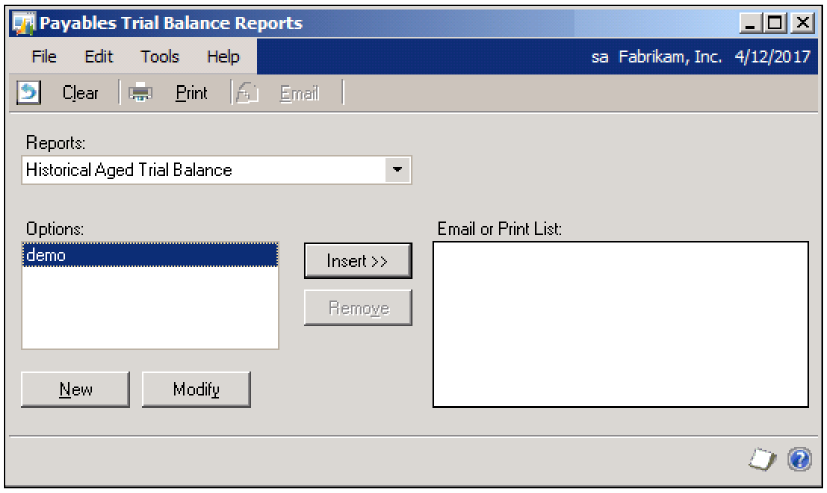
- **Modifying the layout**: This category includes any of the basic layout changes that may be required due to a particular user's preference, such as a desire to standardize customer-facing reports to an organization's marketing standards, or modify a check format to fit on pre-printed check stock, and so on. Fortunately, most of these changes do not require use of the advanced Report Writer tools, and users are capable of making these changes within the **Report Layout** window.
- **Changing the page orientation**: Although many reports are designed to print in landscape format, some users may prefer to view the report in portrait orientation. Although this allows more records to fit on a single page, it may require sacrificing space between fields or removing unnecessary fields entirely. Setting a report's page orientation in Report Writer also allows us to override a user's print settings and always print the report in the specified orientation.
- **Adding or removing fields**: Although many of the default Report Writer reports come preloaded with the relevant fields required for a particular report, we will often find a need to add a new field to a report. For example, if the **multi-bin** functionality in **Inventory Control** module is enabled, we may want to add bin location fields to our inventory-related reports. On the other hand, we may find that the default reports include too much information; therefore, we can use Report Writer to remove these fields.

- **Using VBA with reports**: If we have Modifier with VBA installed in our environment, we can use VBA code to customize our reports beyond the capabilities offered through the standard Report Writer. Please note, however, this functionality is not supported in a web client-based GP environment. This can be an important consideration for organizations considering the GP web client who have an extensive need for using reports customized with VBA.

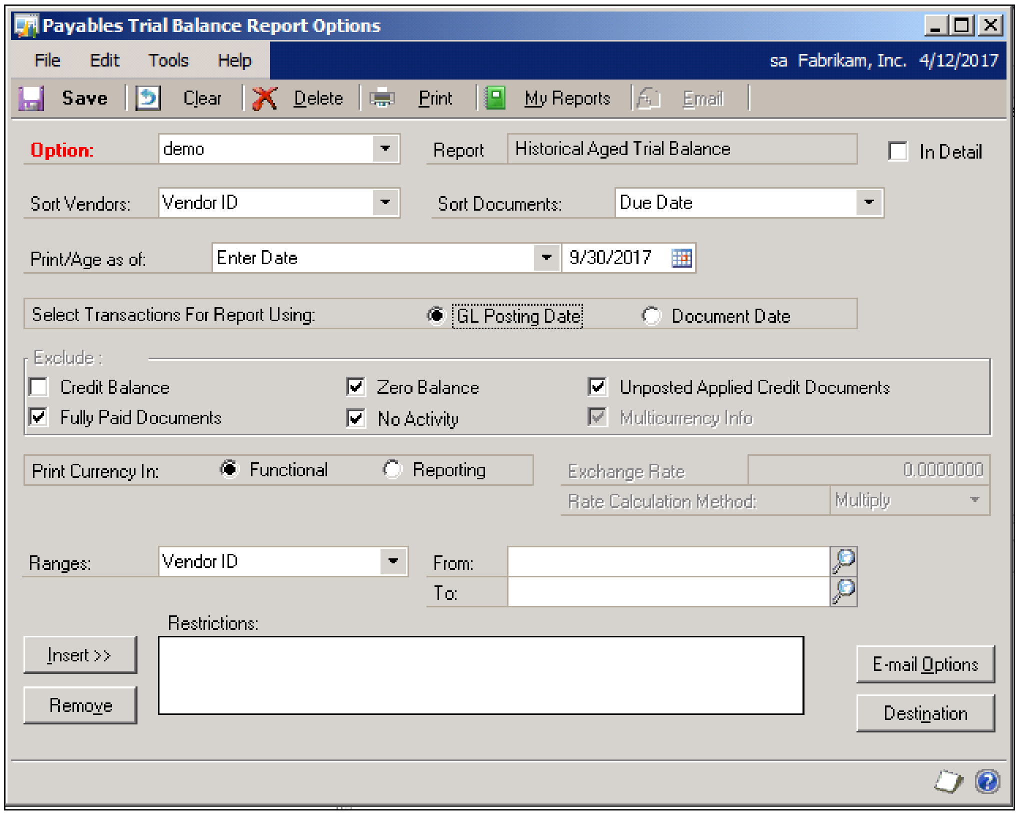
Modifying a report – adding aging periods to Payables Trial Balance Report

Dynamics GP allows for up to seven aging periods in Accounts Payable. By default, the **Historical Aged Trial balance** summary report only prints the first four. In this step-by-step example, we will modify this aging report to include two additional aging periods:

1. In Dynamics GP, navigate to **Reports** | **Purchasing** | **Trial Balance**.
2. This will open the **Payables Trial Balance Reports** window. From here, choose **Historical Aged Trial Balance** from the **Reports:** drop-down, highlight the **demo** option and click on **Modify**.



3. This will open the **Payables Trial Balance Report Options** window. From here, set your parameters by changing the necessary fields. For our example, set the following fields:
    - Uncheck the **In Detail** checkbox.
    - Set **Print/Age as of** to enter date and enter a period end date (You can use the drop-down menu to select **Current Date**, **End of Month**, **End of Period**, and so on. We are using 2017 for our year because of our sample data).
    - Accept the rest of the defaults or add any range filters as necessary.


Payables Trial Balance Report Options
File Edit Tools Help
sa Fabrikam, Inc. 4/12/2017
Save Clear Delete Print My Reports Email

Option: demo   Report Historical Aged Trial Balance   ☐ In Detail

Sort Vendors: Vendor ID   Sort Documents: Due Date

Print/Age as of: Enter Date 9/30/2017

Select Transactions For Report Using: ● GL Posting Date ○ Document Date

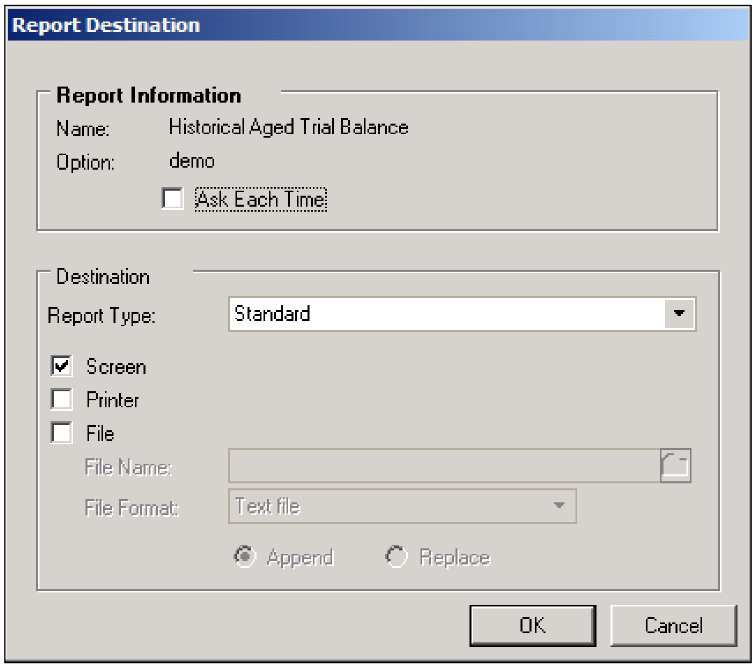
Exclude:
☐ Credit Balance ☑ Zero Balance ☑ Unposted Applied Credit Documents
☑ Fully Paid Documents ☑ No Activity ☑ Multicurrency Info

Print Currency In: ● Functional ○ Reporting   Exchange Rate 0.0000000   Rate Calculation Method: Multiply

Ranges: Vendor ID   From: To:

Restrictions:

4. Click on **Destination** to open the **Report Destination** window. Accept the default **Report Type** as **Standard** (**Template** will be covered later in this chapter with Word Templates) and select the **Screen** checkbox. Click on **OK**.



Report Destination

Report Information

Name: Historical Aged Trial Balance

Option: demo

☐ Ask Each Time

Destination

Report Type: Standard

☑ Screen

☐ Printer

☐ File

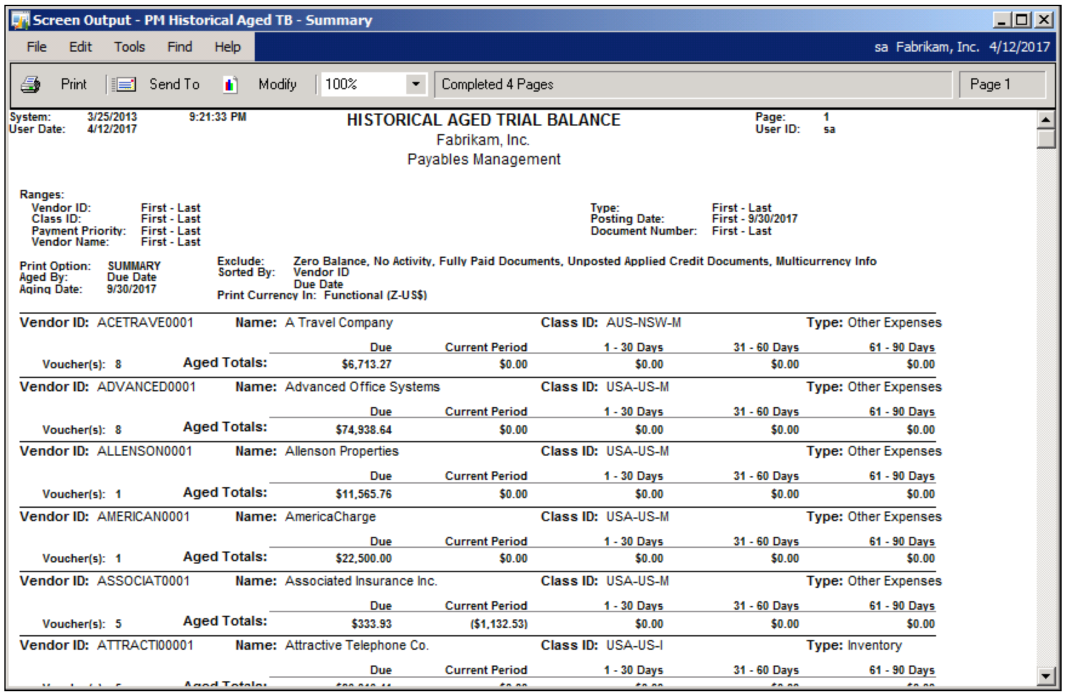
File Name:

File Format: Text file

(●) Append ( ) Replace

5. On our **Payables Trial Balance Report options** window, click on the **Print** button.

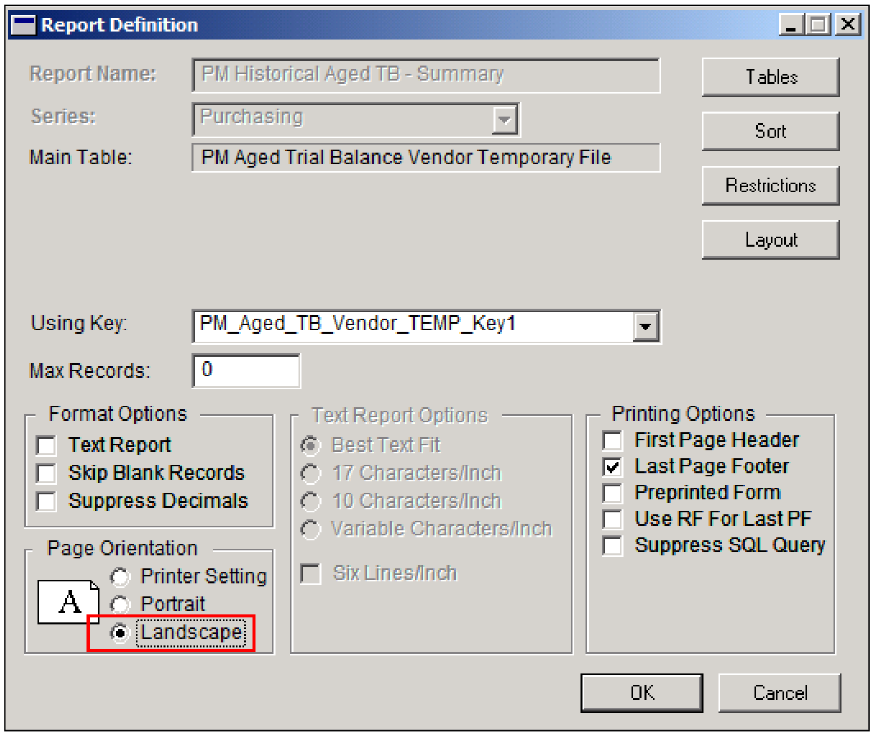
6. This will print our summary HISTORICAL AGED TRIAL BALANCE report to the screen. We can verify from here that we only have the first four aging periods.


Screen Output - PM Historical Aged TB - Summary
File Edit Tools Find Help
sa Fabrikam, Inc. 4/12/2017
Print Send To Modify 100%
Completed 4 Pages
Page 1

```
System:     3/25/2013        9:21:33 PM           HISTORICAL AGED TRIAL BALANCE                    Page:      1
User Date:  4/12/2017                                     Fabrikam, Inc.                           User ID:   sa
                                                      Payables Management

Ranges:
  Vendor ID:          First - Last                                       Type:              First - Last
  Class ID:           First - Last                                       Posting Date:      First - 9/30/2017
  Payment Priority:   First - Last                                       Document Number:   First - Last
  Vendor Name:        First - Last

Print Option:  SUMMARY        Exclude:     Zero Balance, No Activity, Fully Paid Documents, Unposted Applied Credit Documents, Multicurrency Info
Aged By:       Due Date       Sorted By:   Vendor ID
Aging Date:    9/30/2017                   Due Date
                              Print Currency In:  Functional (Z-US$)

Vendor ID:  ACETRAVE0001     Name:  A Travel Company              Class ID:  AUS-NSW-M          Type: Other Expenses
                                          Due     Current Period      1 - 30 Days     31 - 60 Days     61 - 90 Days
   Voucher(s):  8      Aged Totals:   $6,713.27          $0.00            $0.00            $0.00            $0.00
Vendor ID:  ADVANCED0001     Name:  Advanced Office Systems       Class ID:  USA-US-M           Type: Other Expenses
                                          Due     Current Period      1 - 30 Days     31 - 60 Days     61 - 90 Days
   Voucher(s):  8      Aged Totals:  $74,938.64          $0.00            $0.00            $0.00            $0.00
Vendor ID:  ALLENSON0001     Name:  Allenson Properties           Class ID:  USA-US-M           Type: Other Expenses
                                          Due     Current Period      1 - 30 Days     31 - 60 Days     61 - 90 Days
   Voucher(s):  1      Aged Totals:  $11,565.76          $0.00            $0.00            $0.00            $0.00
Vendor ID:  AMERICAN0001     Name:  AmericaCharge                 Class ID:  USA-US-M           Type: Other Expenses
                                          Due     Current Period      1 - 30 Days     31 - 60 Days     61 - 90 Days
   Voucher(s):  1      Aged Totals:  $22,500.00          $0.00            $0.00            $0.00            $0.00
Vendor ID:  ASSOCIAT0001     Name:  Associated Insurance Inc.     Class ID:  USA-US-M           Type: Other Expenses
                                          Due     Current Period      1 - 30 Days     31 - 60 Days     61 - 90 Days
   Voucher(s):  5      Aged Totals:     $333.93     ($1,132.53)           $0.00            $0.00            $0.00
Vendor ID:  ATTRACTI00001    Name:  Attractive Telephone Co.      Class ID:  USA-US-I           Type: Inventory
                                          Due     Current Period      1 - 30 Days     31 - 60 Days     61 - 90 Days
```

7. Click on **Modify** (note, we will only have access to modify if our user has the security role discussed in the *Accessing Report Writer reports* section of this chapter).

8. When asked to save changes to the report option demo, answer **Save**.
9. After selecting **Save**, we will automatically be taken to the **Report Layout** for our report in Report Writer. (Depending on your screen resolution, you will probably also see the **Toolbox**, **Report Definition**, and **Properties** windows).
10. Close the **Report Layout** and on the **Report Definition** window, change the **Page Orientation** to **Landscape**.



Report Name: PM Historical Aged TB - Summary
Series: Purchasing
Main Table: PM Aged Trial Balance Vendor Temporary File
Using Key: PM_Aged_TB_Vendor_TEMP_Key1
Max Records: 0

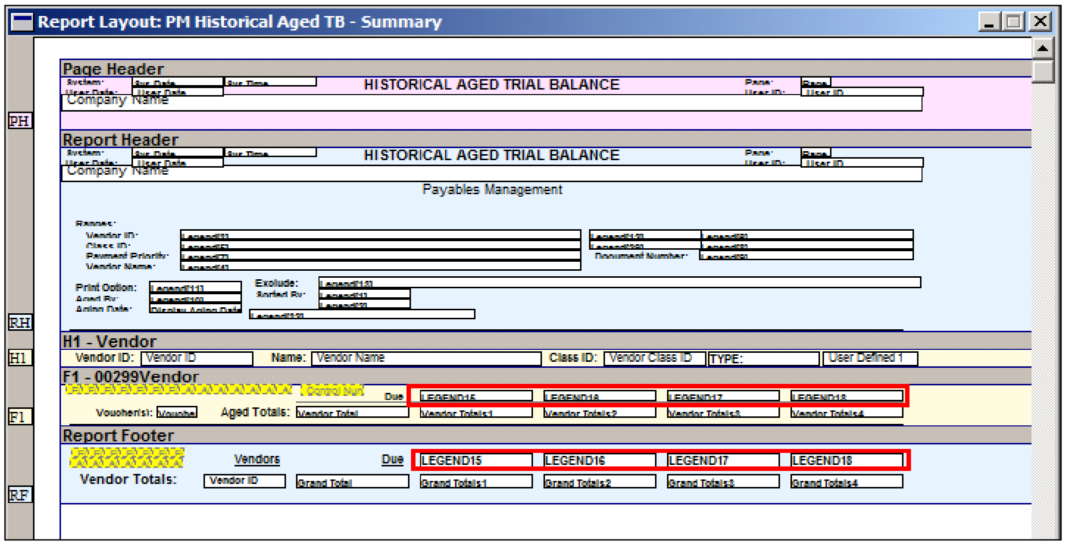
Format Options
- [ ] Text Report
- [ ] Skip Blank Records
- [ ] Suppress Decimals

Text Report Options
- (●) Best Text Fit
- ( ) 17 Characters/Inch
- ( ) 10 Characters/Inch
- ( ) Variable Characters/Inch
- [ ] Six Lines/Inch

Printing Options
- [ ] First Page Header
- [✓] Last Page Footer
- [ ] Preprinted Form
- [ ] Use RF For Last PF
- [ ] Suppress SQL Query

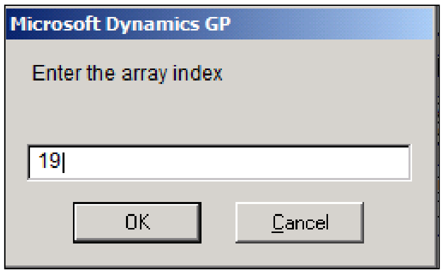
Page Orientation
- ( ) Printer Setting
- ( ) Portrait
- (●) Landscape

11. Click on the **Layout** button to return to the **Report Layout** window. Notice the calculated fields for aging periods in the vendor detail and **Report Footer** sections (highlighted in screenshot):



12. From the **Toolbox**, we select **Calculated Fields** from the drop-down list. Scroll down until we find the **LEGEND** fields. We will find that Report Writer only has predefined calculated fields for the four default aging periods.
13. On the **Toolbox** window, click on **New** to create a new **LEGEND** field for the fifth and sixth aging periods we are adding to this report.
14. On the **Calculated Field Definition** window, set the following fields:
    - **Name**: Set name to `LEGEND19`
    - **Result Type**: Set to **String**
    - **Expression Type**: Accept default value of **Calculated**
    - On the **Fields** tab, **Resources** drop-down menu, choose **RW Legends** from the drop-down
    - On the **Fields** tab, **Field** drop-down menu, choose **Legend** from the drop-down menu

15. Click on **Add** on the **Fields** tab and enter `19` as the **array index**.

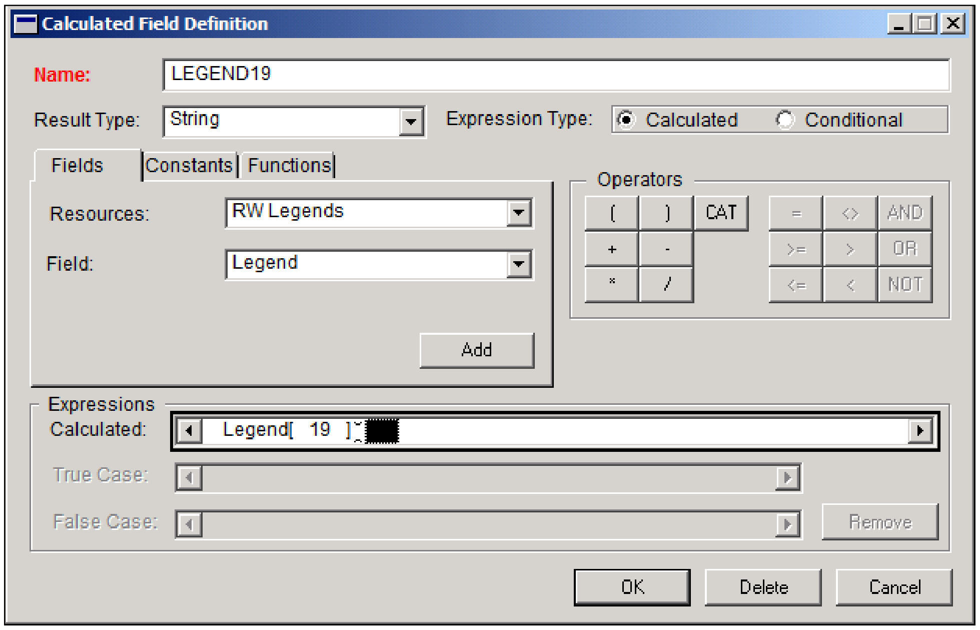


Microsoft Dynamics GP

Enter the array index

19

16. Our completed calculated field should look like the following screenshot:



Calculated Field Definition

Name: LEGEND19

Result Type: String

Expression Type: (● Calculated) ○ Conditional

Resources: RW Legends

Field: Legend

Expressions

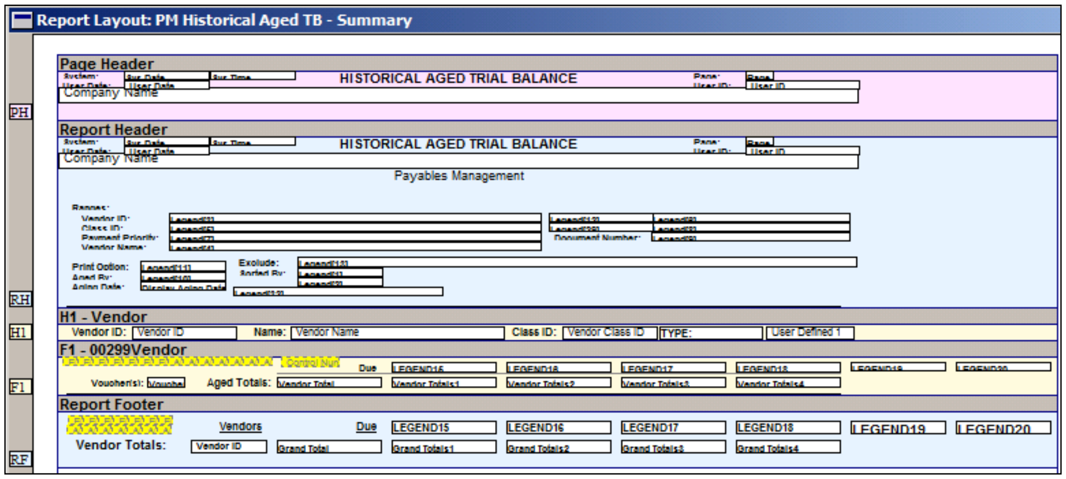
Calculated: Legend[19]

True Case:

False Case:

17. Repeat the steps to create a calculated field for **LEGEND20**, using `20` as the array index value.

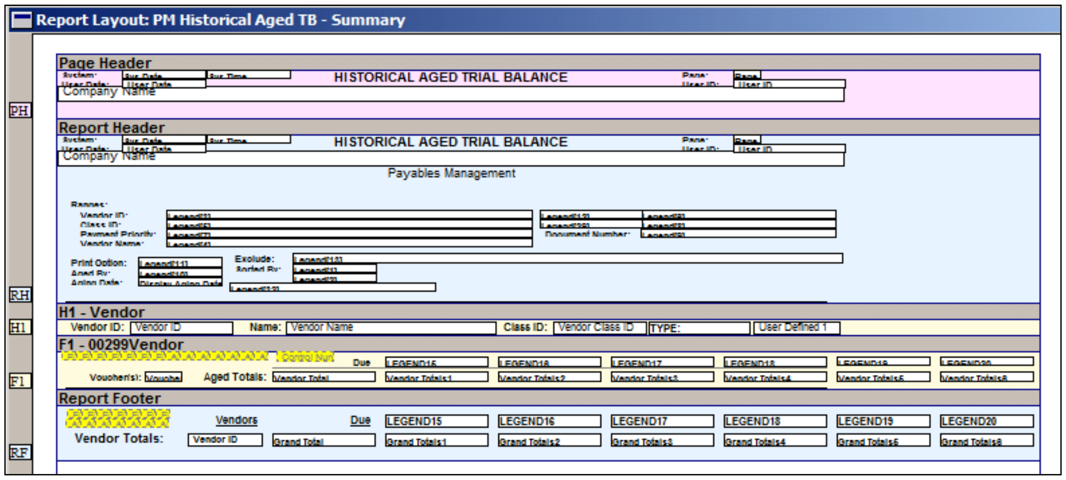
18. Once the calculated fields are added, we can drag them on to the report layout from the **Toolbox**. Once on the layout, we can set the same properties on our fields as the other legend fields in that section. Using the **Object** and **Visual** tabs on the **Properties** window, toggle back and forth between one of the existing column headings and the newly inserted headings to assist in matching the properties of both headings.



19. Now that our column headings have been added, we need to add our **Vendor Totals** and **Grand Totals** for our fifth and sixth aging periods.

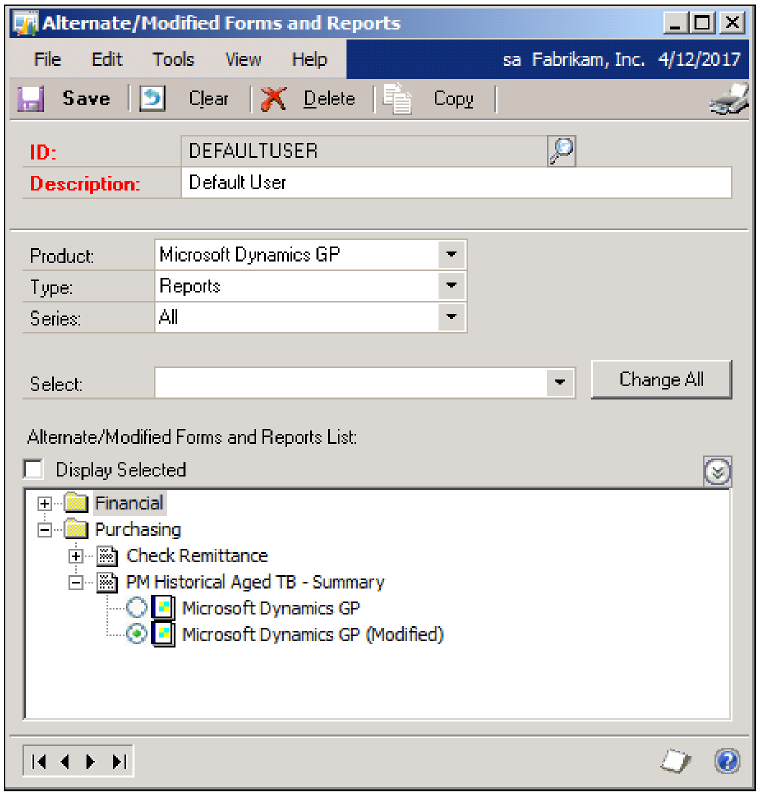
20. Dynamics Report Writer has the necessary calculated fields for the **Vendor Totals** and **Grand Totals** already created, we just need to add them to the report.

21. From the **Toolbox**, choose **Calculated Fields** from the drop-down menu and scroll through the list until you locate **Vendor Totals5**, **Vendor Totals6**, **Grand Totals5**, and **Grand Totals6**. Drag-and-drop them onto the report layout in the same manner as we did with our **LEGEND** fields. Set the field properties as necessary by comparing other fields that are currently on the report.



22. Now we can see that we have the column headings, the vendor totals, and the grand totals for our fifth and sixth aging periods.
23. Close the **Report Layout** and click on **Save** when prompted to save the layout.
24. On the menu bar, click on **File** | **Microsoft Dynamics GP** to return to **Dynamics**.
25. Click on **Save** to save changes to the report when prompted.

26. Remember, before we can print our modified version of the Aging report, we need to set the security to the modified version as described earlier in the chapter in the *Setting security to custom/modified reports* section. The following screenshot is what our Alternate/Modified Forms and Reports window should look like:

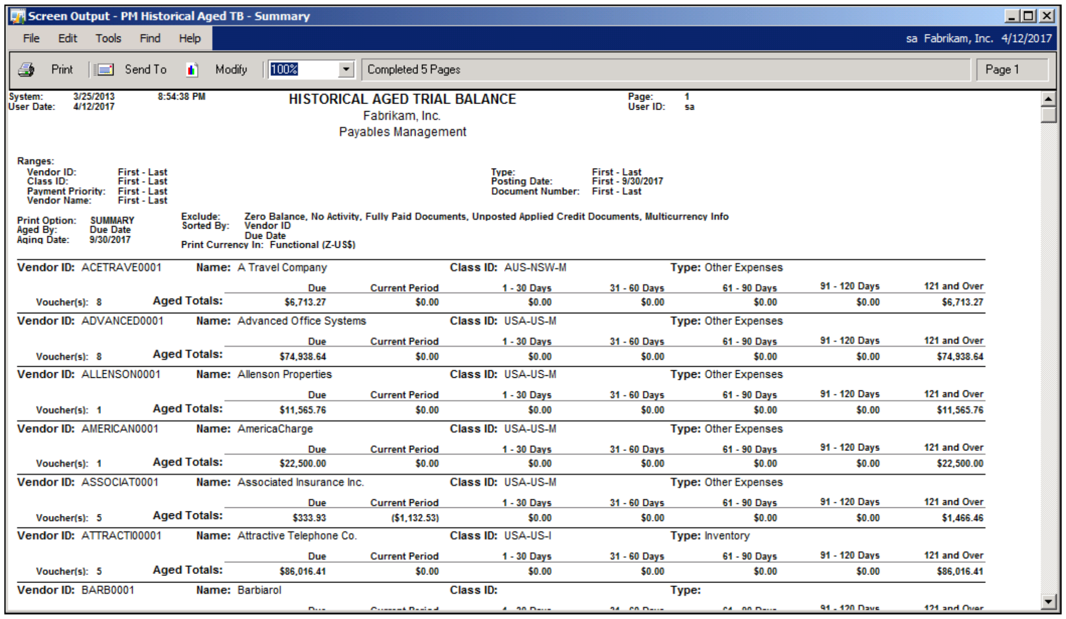

Alternate/Modified Forms and Reports
File Edit Tools View Help
sa Fabrikam, Inc. 4/12/2017
Save Clear Delete Copy
Change All

ID: DEFAULTUSER
Description: Default User
Product: Microsoft Dynamics GP
Type: Reports
Series: All
Select:
Alternate/Modified Forms and Reports List:
Display Selected

- Financial
- Purchasing
  - Check Remittance
  - PM Historical Aged TB - Summary
    - Microsoft Dynamics GP
    - Microsoft Dynamics GP (Modified) [✓]

27. Repeat steps 1 to 6 of this exercise to print our modified version of the report.

28. In the following screenshot, we can now see that our fifth and sixth aging periods are now printing on our summary **HISTORICAL AGED TRIAL BALANCE** report. It may take several attempts at switching back and forth between GP and the Report Writer application to fine-tune other elements of the report, such as line breaks between sections and missed formatting.


Screen Output - PM Historical Aged TB - Summary
File
Edit
Tools
Find
Help
sa Fabrikam, Inc. 4/12/2017
Print
Send To
Modify
100%
Completed 5 Pages
Page 1

```
System:      3/25/2013     8:54:38 PM        HISTORICAL AGED TRIAL BALANCE              Page:     1
User Date:   4/12/2017                               Fabrikam, Inc.                     User ID:  sa
                                                  Payables Management

Ranges:
  Vendor ID:         First - Last                                        Type:              First - Last
  Class ID:          First - Last                                        Posting Date:      First - 9/30/2017
  Payment Priority:  First - Last                                        Document Number:   First - Last
  Vendor Name:       First - Last

Print Option:  SUMMARY        Exclude:     Zero Balance, No Activity, Fully Paid Documents, Unposted Applied Credit Documents, Multicurrency Info
Aged By:       Due Date       Sorted By:   Vendor ID
Aging Date:    9/30/2017                   Due Date
                              Print Currency In:  Functional (Z-US$)

Vendor ID:  ACETRAVE0001     Name:  A Travel Company           Class ID:  AUS-NSW-M      Type: Other Expenses
                                    Due   Current Period   1 - 30 Days   31 - 60 Days   61 - 90 Days   91 - 120 Days   121 and Over
   Voucher(s):  8   Aged Totals:   $6,713.27     $0.00        $0.00          $0.00          $0.00          $0.00        $6,713.27
Vendor ID:  ADVANCED0001     Name:  Advanced Office Systems    Class ID:  USA-US-M       Type: Other Expenses
                                    Due   Current Period   1 - 30 Days   31 - 60 Days   61 - 90 Days   91 - 120 Days   121 and Over
   Voucher(s):  8   Aged Totals:  $74,938.64     $0.00        $0.00          $0.00          $0.00          $0.00       $74,938.64
Vendor ID:  ALLENSON0001     Name:  Allenson Properties        Class ID:  USA-US-M       Type: Other Expenses
                                    Due   Current Period   1 - 30 Days   31 - 60 Days   61 - 90 Days   91 - 120 Days   121 and Over
   Voucher(s):  1   Aged Totals:  $11,565.76     $0.00        $0.00          $0.00          $0.00          $0.00       $11,565.76
Vendor ID:  AMERICAN0001     Name:  AmericaCharge              Class ID:  USA-US-M       Type: Other Expenses
                                    Due   Current Period   1 - 30 Days   31 - 60 Days   61 - 90 Days   91 - 120 Days   121 and Over
   Voucher(s):  1   Aged Totals:  $22,500.00     $0.00        $0.00          $0.00          $0.00          $0.00       $22,500.00
Vendor ID:  ASSOCIAT0001     Name:  Associated Insurance Inc.  Class ID:  USA-US-M       Type: Other Expenses
                                    Due   Current Period   1 - 30 Days   31 - 60 Days   61 - 90 Days   91 - 120 Days   121 and Over
   Voucher(s):  5   Aged Totals:     $333.93  ($1,132.53)     $0.00          $0.00          $0.00          $0.00        $1,466.46
Vendor ID:  ATTRACTI0001     Name:  Attractive Telephone Co.   Class ID:  USA-US-I       Type: Inventory
                                    Due   Current Period   1 - 30 Days   31 - 60 Days   61 - 90 Days   91 - 120 Days   121 and Over
   Voucher(s):  5   Aged Totals:  $86,016.41     $0.00        $0.00          $0.00          $0.00          $0.00       $86,016.41
Vendor ID:  BARB0001         Name:  Barbiarol                  Class ID:                 Type:
```

With this step-by-step guide, we should now have more confidence in using the Dynamics Report Writer to make modifications to the existing Dynamics GP Reports!

Importing and exporting customized reports

Reports that are modified via Report Writer are portable, meaning we can make a modification to a report and transfer it to other workstations or GP instances. Now that we've explored how to customize our own reports, we may want to utilize the Import and Export functionality to transfer the modified report to another local reports dictionary. This can be used in cases such as the one we discussed earlier in this chapter where individual workstations on the same system are accessing separate report dictionaries. It can also be used to transfer customized reports from one company database to another or even from one system to another.

The ability to import and export customized reports means the developers and consultants can create custom reports on a test database, test them, and solicit user feedback and approval—without ever impacting the production environment! Once a report customized on a test database has been approved by the user, the customized report can be wrapped in a package file and imported into the production database.

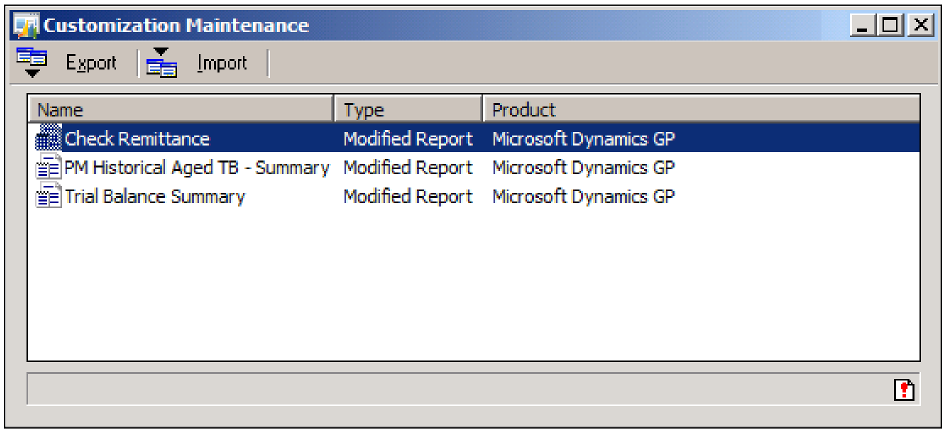
The Import and Export functionality for customizations is also an invaluable tool in saving prior versions of reports. For any given report on a GP workstation, only the default report and one modified report can be used. Before making additional changes to the modified version of the report, it may be worth exporting a package file for the customized report that is being modified. In this way, prior versions of a single report can be maintained, and if any issues arise from the modified report, a previous version can be imported to prevent interruptions to the reporting process.

Exporting customized reports

Before a customized report can be transferred to another environment, the report must be converted to a package file. This can be done from within GP 2013:

1. Open the **Microsoft Dynamics GP** menu.
2. Navigate to **Tools | Customize | Customization Maintenance** to open the **Customization Maintenance** window.

The **Customization Maintenance** window (seen in the following screenshot) contains a list of the various customizations that exist in the current GP installation. This includes forms of customizations beyond just reports modified by Report Writer. Other customizations, such as those originating from Modifier or VBA will appear in this window as well. To help us understand the customizations, GP lists each customization, followed by the **Type** of customization and the **Product** in which the customization resides. Customized reports related to **Fixed Assets** will most likely appear in the **Fixed Assets Product** whereas basic reports, such as the **Trial Balance Summary**, will appear as part of the standard **Microsoft Dynamics GP Product**.



Name	Type	Product
Check Remittance	Modified Report	Microsoft Dynamics GP
PM Historical Aged TB - Summary	Modified Report	Microsoft Dynamics GP
Trial Balance Summary	Modified Report	Microsoft Dynamics GP

3. Select the customized reports that should be exported to the package file. By pressing the *Ctrl* key, users can select multiple components. Holding down the *Shift* key while selecting two non-adjacent components will allow users to select the entire range of components between the two selected components.
4. Click on the **Export** button to open the **Windows Explorer** window to save the package file.
5. Select a location on the network or local computer in which the package file can be saved. Package files will be denoted with a `*.package` suffix. If desired, rename the first portion of the file name to a more descriptive term for easier identification. In the **Save as Type** drop-down box, **Customization Package (*.package)** should be displayed.
6. Click on **Save** to save the package file to the selected location. This file can now be used to import these customizations to another GP install accessing a separate reports dictionary.

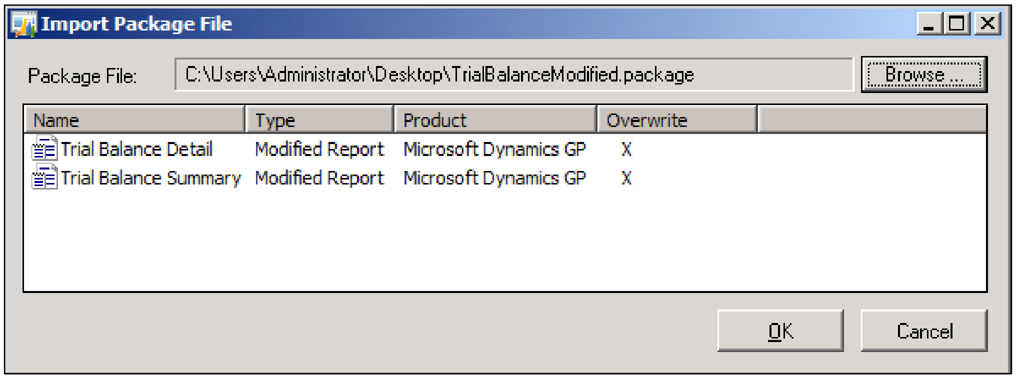
Importing customized reports

Once we've exported a package file from another GP install, we can use that package file as the source file for a customization import. This will apply the customizations to the new installation, and users can then choose whether or not the modified or default reports should be visible to the users. Be careful! When importing a package file containing a customized report to another GP workstation on which the same report has also been customized, the existing report will be overwritten! Before going through with the import, be absolutely certain that the report(s) should be overwritten!

Prior to importing a customization, we should have our package file ready to go. For tips on creating such a package file, refer to the previous section on exporting customized reports from GP. If necessary, it may be worth completing a test import using a test database to ensure that our import will bring in the correct customizations. Better safe than sorry!

Once we are ready to go through with the import, the process is fairly easy. In the following example, we'll use the `TrialBalanceModified.package` file to import two modified reports, the `Trial Balance Summary` and the `Trial Balance Detail`.

1. Open the **Microsoft Dynamics GP** menu.
2. Navigate to **Tools | Customize | Customization Maintenance** to open the **Customization Maintenance** window. This is the same window we used in the previous section to export our customized reports.
3. Click on the **Import** button to open the **Import Package File** window.
4. Click on the **Browse** button to open the **Windows Explorer** window to select an existing `*.package` file. In this case, we will select the `TrialBalanceModified.package` file. A list of the reports found in the selected package file is displayed in the **Import Package File** window. Reports in this package file for which a customized version already exists on the target database, will display an **X** in the **Overwrite** column.



Package File: C:\Users\Administrator\Desktop\TrialBalanceModified.package

Name	Type	Product	Overwrite
Trial Balance Detail	Modified Report	Microsoft Dynamics GP	X
Trial Balance Summary	Modified Report	Microsoft Dynamics GP	X

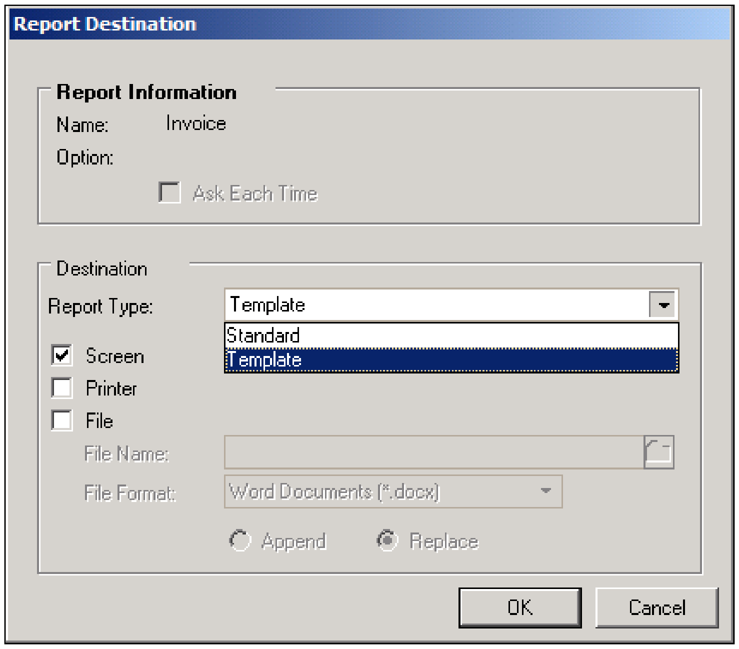
5. Assuming we wish to import this package file, we can click on the **OK** button. Microsoft Dynamics GP will offer a helpful warning message that existing items will be overwritten by the import. Assuming that this is desired behavior, we can click on the **OK** button to complete the import.
6. The **Import Package File** window will close, and we are returned to the **Customization Maintenance** window. If the import was successful, a message to this effect appears at the bottom left-hand corner of the window. Otherwise, we can click on the **Errors** button at the bottom right-hand corner of the window to see and print a list of errors from the import process.

Completing the import process adds the customized reports to the customized reports dictionary referenced by the current GP client. If this is the first time the customized report is being used in this environment, use the **Alternate/Modified Forms and Reports** window to select the modified report(s) for use in this new environment. Details on how to do this can be found earlier in this chapter.

We have now successfully exported and imported our own package file of customized reports. This functionality can be quite the time-saver for developers and consultants who work with multiple company databases and systems!

Rendering reports with the Microsoft Word template feature

Users who are upgrading to GP 2013 from a version prior to GP 2010 may be surprised to see a new feature when printing certain reports. As seen in the **Report Destination** window screenshot below, users are now given a choice of selecting a **Report Type** of **Standard** or **Template**.


Report Destination

Report Information
Name: Invoice
Option:
[ ] Ask Each Time

Destination
Report Type: Template
Standard
Template
[✓] Screen
[ ] Printer
[ ] File
File Name:
File Format: Word Documents (*.docx)
Append [✓ Replace]

The Template option was a new feature introduced for GP 2010 that allows users to print certain standard GP reports in Word template format. Once a report is printed to Word template, the contents of the report can be reformatted, the report can be saved to a file directory location for later reference, or it can also be included as an email attachment. In short, this option brings a fresh, new look to the report generation process in Dynamics GP!

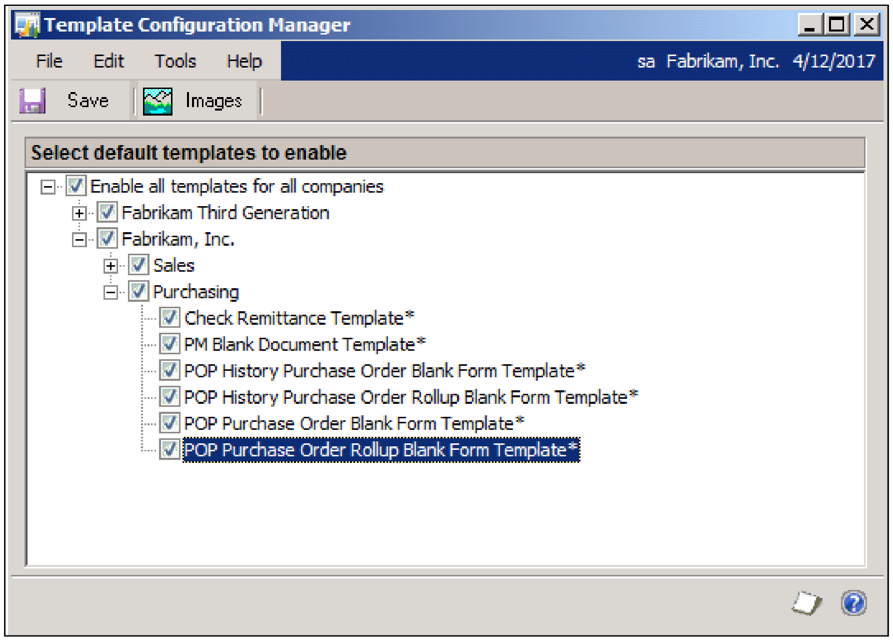
Before a report can be printed in Word template form, however; it must be template-enabled, a process which requires the use of Microsoft Dexterity when created from scratch. However, this can be a difficult process that requires a high degree of technical expertise, time, and patience to create Word templates for reports beyond ones that already have Word templates created for them. Then with the release of GP 2010 R2 came the Word Template Generator. This utility offers users an easy way to go beyond the default 32 reports that already have Word templates defined!

First, let's take a look at the steps required to verify that the default templates are enabled for use in GP 2013, then we'll explore the process of making some basic modifications to existing Word templates, and finally we will look at how to use the Word Template Generator to create new Word templates.

Enabling Word templates

The **Template Configuration Manager** window should be the first stop for anyone seeking to work with the default Word templates in GP 2013. This window controls the ability, among other things, to turn the Word template feature on and off in a company database, to assign a company logo to all reports with an associated Word template, and to determine which individual reports should be made available in Word template format. As we've mentioned earlier, 32 reports in GP 2013 have associated Word templates, so we can use this window to perform these functions during our initial setups of GP 2013.

The **Template Configuration Manager** window can be accessed from GP 2013 by navigating to **Reports** | **Template Configuration**. We can expand the various nodes to see a screenshot similar to the following:


Template Configuration Manager
File
Edit
Tools
Help
sa Fabrikam, Inc. 4/12/2017
Save
Images
Select default templates to enable

- ✓ Enable all templates for all companies
  - ✓ Fabrikam Third Generation
  - ✓ Fabrikam, Inc.
    - ✓ Sales
    - ✓ Purchasing
      - ✓ Check Remittance Template*
      - ✓ PM Blank Document Template*
      - ✓ POP History Purchase Order Blank Form Template*
      - ✓ POP History Purchase Order Rollup Blank Form Template*
      - ✓ POP Purchase Order Blank Form Template*
      - ✓ POP Purchase Order Rollup Blank Form Template*

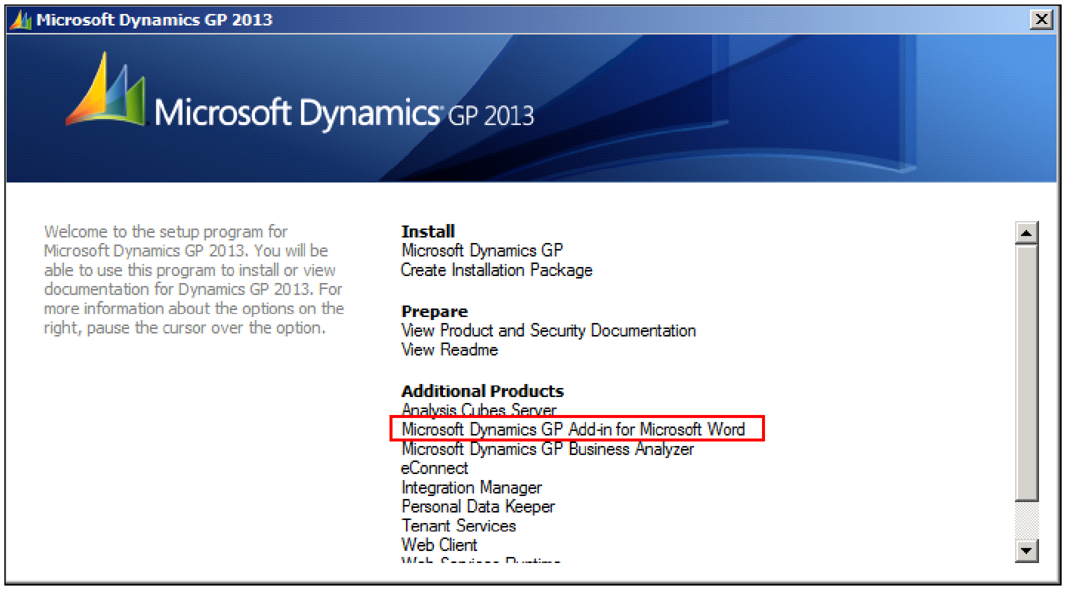
In this window, we can select the default templates that should be enabled on a company-by-company basis. Earlier, we mentioned that GP comes pre-loaded with 32 default Word templates. The first time we open this window, we see those pre-loaded templates. Modified templates or new templates created via the new Word Template Generator will not be displayed in this view. Expanding the **Enable all templates for all companies** node displays a list of company databases. Underneath each company database node, we see a list of series that contain Word templates that can be assigned for that company. By default, only the **Sales** and **Purchasing** series contain predefined Word templates that can be assigned or unassigned from the company databases.

If we don't want a particular report available for printing in its Word template format, we can deselect that report from this list. **Template** will no longer display as an option in the Report Options window when printing this report.

From this window, we can also decide if the standard report formats should be made available for printing along with the templates for Word template-enabled reports. If we want to force users to use the Word templates wherever possible, we can deselect the option to **Allow printing of standard report when template is available** at the bottom of this window.

Installing the Dynamics GP add-in for Microsoft Word

Word templates can also be modified. Before modifying or creating new templates from scratch with the new Word template functionality, a special add-in must be installed for Microsoft Word. This add-in is available as an additional product that comes with the installation media of GP 2013. Run the `setup.exe` file from the installation media and select **Microsoft Dynamics GP Add-in for Microsoft Dynamics** to begin the installation. The following screenshot shows an example of some of the other additional products that can be installed from this window:



Welcome to the setup program for Microsoft Dynamics GP 2013. You will be able to use this program to install or view documentation for Dynamics GP 2013. For more information about the options on the right, pause the cursor over the option.

Install
Microsoft Dynamics GP
Create Installation Package

Prepare
View Product and Security Documentation
View Readme

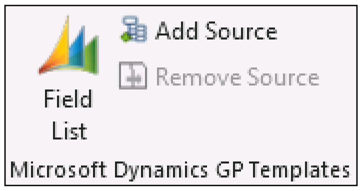
Additional Products
Analysis Cubes Server
Microsoft Dynamics GP Add-in for Microsoft Word
Microsoft Dynamics GP Business Analyzer
eConnect
Integration Manager
Personal Data Keeper
Tenant Services
Web Client

This tool only needs to be installed on workstations where modifications will be made to the Word templates. It is not necessary to install this tool in order to generate Word template reports from GP 2013!

Once this add-in has been installed, a new section is added to the **Developer** tab in the ribbon in Microsoft Word. If the **Developer** tab is not enabled for the ribbon, enable it with the following steps:

1. Open Microsoft Word.
2. Click on **File** in the upper left-hand corner of the application.
3. Click on **Options** to open the **Word Options** window.
4. In Word 2013, click on the **Customize Ribbon** tab and confirm that the **Developer** tab is checked in the right-hand pane.
5. Click on **OK** to close the **Word Options** window.

The **Developer** tab should now appear at the top of the window. A new section seen in the following screenshot by the name of **Microsoft Dynamics GP Templates** should now appear in the **Developer** tab:



We will use this section to add new fields once we begin modifying the Word templates.

Understanding the Word template modification process

When a template-enabled report is printed, a combination of the standard report definition, a custom XML file containing the report definition and data, and the Word template file are all merged in a Template Processing Engine to create the final report. Fortunately, we don't have to be experts in the underlying technical details to be able to make modifications to the Word templates.

One of the most common misconceptions when beginning work with the new Word template feature is that only the Word template itself must be modified. While this is true in certain cases such as changing the font and color of a particular field or adding a watermark to a document, other Word template changes require modifications to the underlying report definition in Report Writer. As a general rule of thumb, we should remember that if we are going to add new data to a Word template, we must first add the tables and fields to support that new data to the accompanying report definition in Report Writer.

If one of the template-enabled reports is modified in Report Writer and is then selected as the default report for use in the GP application, a new Word template must be created to use with the modified report. Until this new Word template is created, users will not be able to select the **Template** option when printing the report, even if the Word template for the original report is enabled.

Modifying the presentation of a default Word template

Now that we've discussed how the Word template modification process works, let's start making some changes to our existing Word templates! Since the process for making minor formatting changes to an existing template differs from the process for adding and removing fields, we have split this into two separate sections.

Applying simple formatting changes and password protection to an existing template

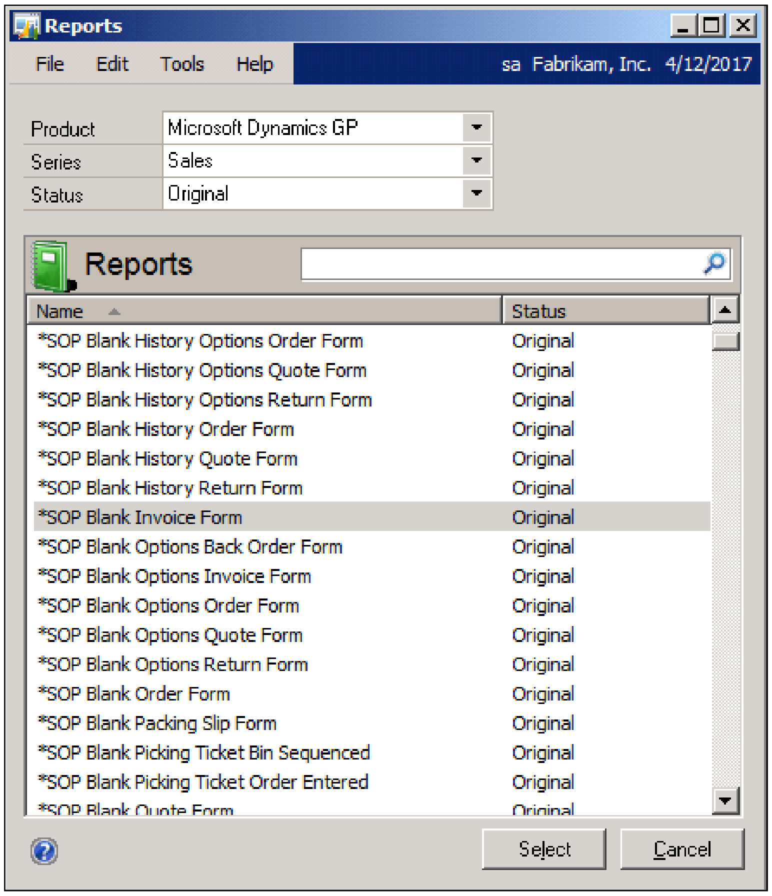
For our first modification, we'll modify the default **SOP Blank Invoice Form** template by changing the **Total** field to a red font color so that customers will easily see how much they owe. We'll also make the **Comment** fields larger and italicized to draw the customer's focus to important comments that may exist on the invoice. Finally, we will enable a process that forces this document to be generated in read-only mode, preventing users from making changes after printing an invoice document from GP.

In the end, our modifications should improve the visual appeal of this report for customers who will receive these invoices.

Let's begin:

1. Open the **Report Template Maintenance** window by navigating to **Reports | Template Maintenance**.
2. Click on the **Report Name** field in the area that says **Click here to select a report**.
3. Click on **More Reports** to open the **Reports** window.
4. In the **Product** field, select **Microsoft Dynamics GP**.
5. In the **Series** field, select **Sales**.
6. In the **Status** field, select **Original**.

7. Scroll down to find and select the **SOP Blank Invoice Form**.



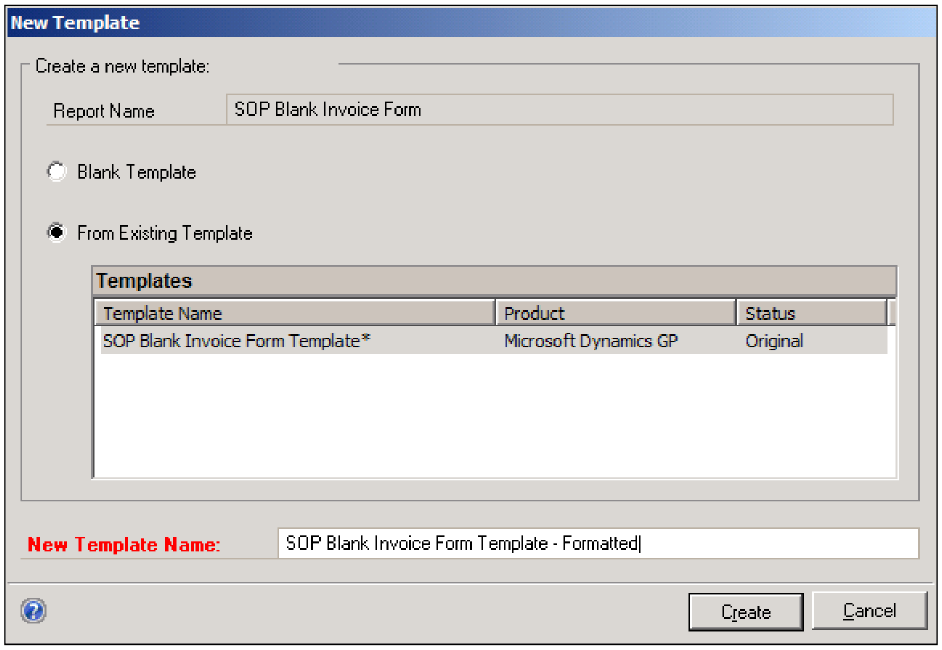
Product: Microsoft Dynamics GP
Series: Sales
Status: Original

Reports

Name	Status
*SOP Blank History Options Order Form	Original
*SOP Blank History Options Quote Form	Original
*SOP Blank History Options Return Form	Original
*SOP Blank History Order Form	Original
*SOP Blank History Quote Form	Original
*SOP Blank History Return Form	Original
*SOP Blank Invoice Form	Original
*SOP Blank Options Back Order Form	Original
*SOP Blank Options Invoice Form	Original
*SOP Blank Options Order Form	Original
*SOP Blank Options Quote Form	Original
*SOP Blank Options Return Form	Original
*SOP Blank Order Form	Original
*SOP Blank Packing Slip Form	Original
*SOP Blank Picking Ticket Bin Sequenced	Original
*SOP Blank Picking Ticket Order Entered	Original
*SOP Blank Quote Form	Original

8. Click on the **Select** button to return to the **Report Template Maintenance** window.
9. Since we cannot modify the default template, we must create a copy of the template. To do this, select the default report in the **Template Name** pane. Click on the **New** button at the top of the window to open the **New Template** window.
10. In the **New Template** window, select **From Existing Template**.

11. Enter the value `SOP Blank Invoice Form Template – Formatted` in the **Template Name** field.


New Template

Create a new template:

Report Name: SOP Blank Invoice Form

Blank Template [✓ From Existing Template]

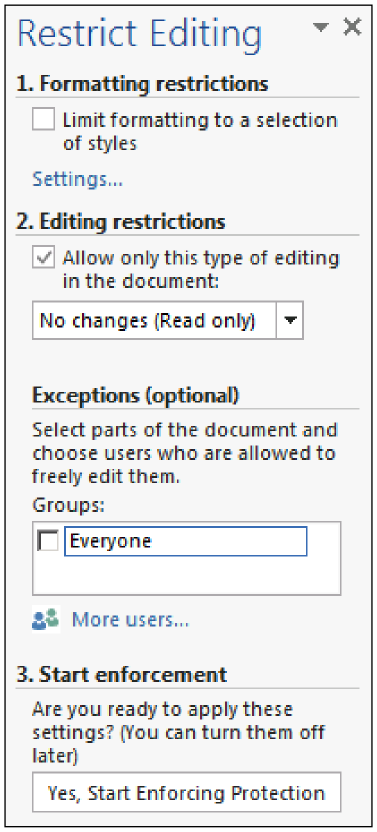
Templates

Template Name	Product	Status
SOP Blank Invoice Form Template*	Microsoft Dynamics GP	Original

New Template Name: SOP Blank Invoice Form Template - Formatted

12. Click on the **Create** button to create this new template and return to the **Report Template Maintenance** window.
13. Select the newly added report template from the **Template Name** pane and click on the **Modify** button to open the report for modification in Word.
14. Towards the bottom right-hand corner of the report, find the **Total** field. Select the string of **X** values next to this field. A small screen tip will appear above the **X** values identifying this as the **F/O Document Amount** field.
15. On the **Home** tab in the ribbon, click on the **Font Color** button and change the font color of this field to red.
16. Now, find and select the first row of comment fields on the bottom left-hand corner of this template. This screen tip is identified as **sopUsrDefWorkHist**.

17. On the **Home** tab in the ribbon, click on the **Italics** button to italicize the comment text. Additionally, adjust the size of the text in this field to `10`. Because the comment text fields share a table with the **total values** on the right-hand side of the report, we want to make sure we don't make the comment text fields too large, as this may cause the total fields to spread out too far from each other.
18. Repeat the previous step for the additional comment lines.
19. Now, let's apply read-only permissions to our document so that it cannot be modified after printing in GP. In the **Review** tab in the ribbon, select **Restrict Editing** in the **Protect** group.
20. In the **Editing Restrictions** section of the **Restrict Editing** pane, check the option **Allow only this type of editing in the document** and ensure **No Changes (Read Only)** is selected in the drop-down. The **Restrict Editing** pane should now look like the following screenshot:



Restrict Editing

1. Formatting restrictions

☐ Limit formatting to a selection of styles

Settings...

2. Editing restrictions

☑ Allow only this type of editing in the document:

No changes (Read only)

Exceptions (optional)

Select parts of the document and choose users who are allowed to freely edit them.

Groups:

☐ Everyone

More users...

3. Start enforcement

Are you ready to apply these settings? (You can turn them off later)

Yes, Start Enforcing Protection

21. Click on the **Yes, Start Enforcing Protection** button under the **Start Enforcement** section of the pane. In the next window, enter a password to protect this document.
22. With all modifications made to the report, navigate to **File | Save As**.
23. Save the template to a file directory location that can be accessed in a later step. If an option exists in the file directory to **Maintain compatibility with previous versions**, select this option. Once the template is saved, close the Word document.
24. Return to the **Report Template Maintenance** window. In the **Template Name** pane, select the report template that was just renamed and modified.
25. Click on the green plus sign at the top right-hand corner of the **Template Name** pane.
26. In the file directory, browse to the location in which the modified template was saved. Select that template and click on **Open**. When the override warning message appears, click on **Yes** to replace the template.

Setting the default Word template for a company database

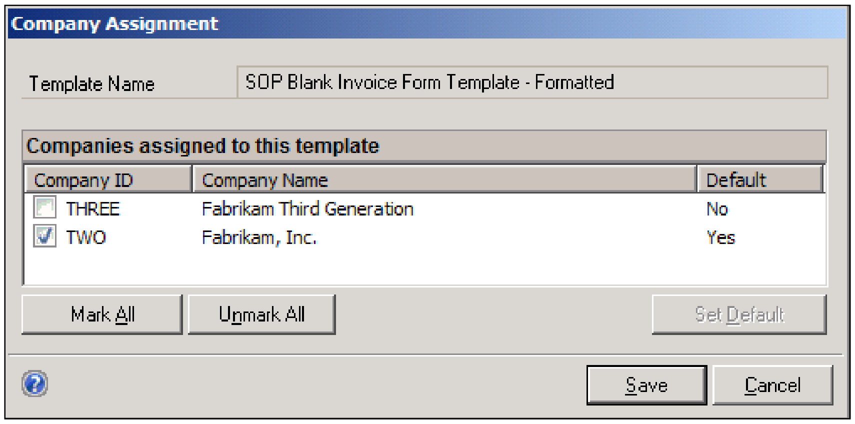
Now that we've modified our first Word template, we have a few steps remaining to assign the Word template as the default template for use in GP. Within a single GP company database, only one template version for each report can be selected as the default template. This default template is the one that will print every time the report is printed to the template **Report Type**.

At least one report template must be set as the default template for the company database. Additionally, depending on the type of Word template we are working with, we have the ability to assign a modified template to a range of customers or customer classes (if we are working with sales reports) or vendors or vendor classes (if we are working with purchasing reports).

In this example, we'll set this report as the default report for the current company database:

1. Return to the **Report Template Maintenance** window and select the modified template in the **Template Name** pane.
2. Click on the **Assign** button and select **Company** to open the **Company Assignment** window.

3. Select the checkbox next to the current company database as shown in the following screenshot:

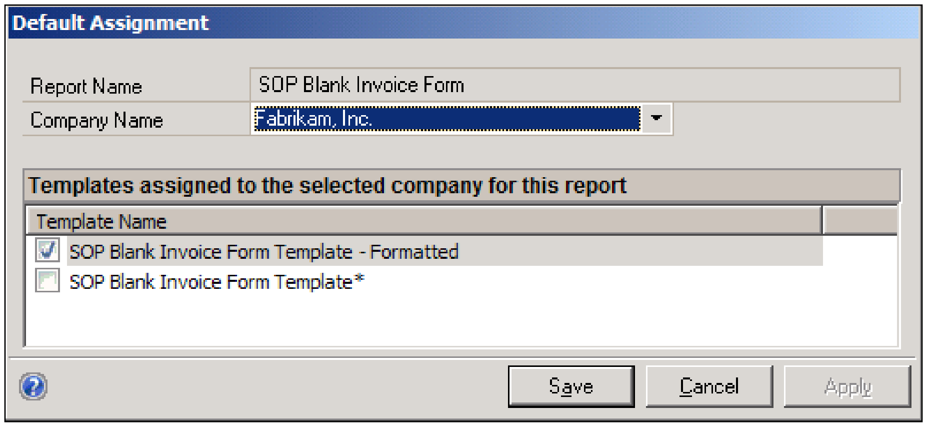


Template Name: SOP Blank Invoice Form Template - Formatted

Companies assigned to this template

Company ID	Company Name	Default
☐ THREE	Fabrikam Third Generation	No
☑ TWO	Fabrikam, Inc.	Yes

4. Click on the **Set Default** button to open the **Default Assignment** window. Note the **Company Name** drop-down field in this window. If we had other company databases installed, we could select those databases to assign different default templates to each database.
5. Select the report that should be assigned as the default template. In this case, we will select the **SOP Blank Invoice Form Template – Formatted** report.



Report Name: SOP Blank Invoice Form

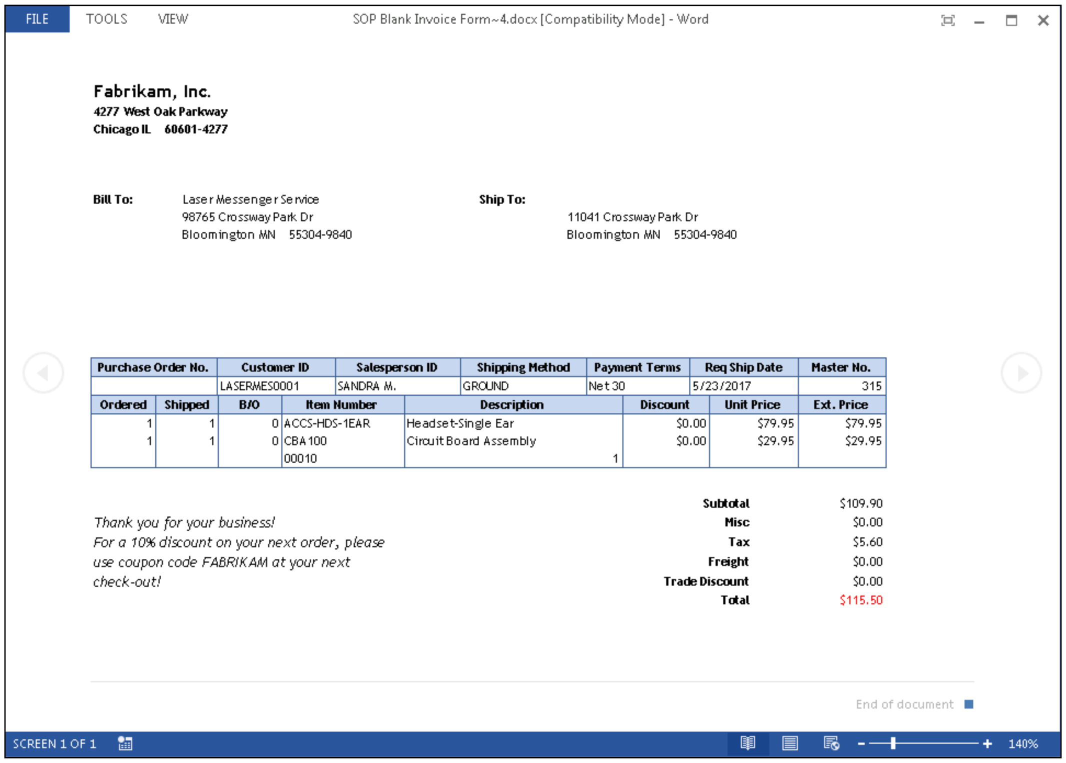
Company Name: Fabrikam, Inc.

Templates assigned to the selected company for this report

Template Name
☑ SOP Blank Invoice Form Template - Formatted
☐ SOP Blank Invoice Form Template*

6. Click on **Save** to close this window and save the selected report as the default report for this company database. Click on **Save** on the **Company Assignment** window, as well.

To see the effects of our Word template modifications, let's print a sales invoice to the screen using the **Blank Paper** format and **Template** as the **Report Type**.



SOP Blank Invoice Form~4.docx [Compatibility Mode] - Word

Fabrikam, Inc.
4277 West Oak Parkway
Chicago IL 60601-4277

Bill To: Laser Messenger Service
98765 Crossway Park Dr
Bloomington MN 55304-9840

Ship To:
11041 Crossway Park Dr
Bloomington MN 55304-9840

Purchase Order No.	Customer ID	Salesperson ID	Shipping Method	Payment Terms	Req Ship Date	Master No.
	LASERMES0001	SANDRA M.	GROUND	Net 30	5/23/2017	315

Ordered	Shipped	B/O	Item Number	Description	Discount	Unit Price	Ext. Price
1	1	0	ACCS-HDS-1EAR	Headset-Single Ear	$0.00	$79.95	$79.95
1	1	0	CBA 100	Circuit Board Assembly	$0.00	$29.95	$29.95
			00010	1			

Thank you for your business!
For a 10% discount on your next order, please use coupon code FABRIKAM at your next check-out!

Subtotal	$109.90
Misc	$0.00
Tax	$5.60
Freight	$0.00
Trade Discount	$0.00
Total	$115.50

End of document

Congratulations, you've now modified your first Word template! Notice how the changes we made to the Word template have been applied to this generated report and if we attempt to make any changes, we are notified that this document is protected from unintentional editing.

Adding fields to an existing Word template

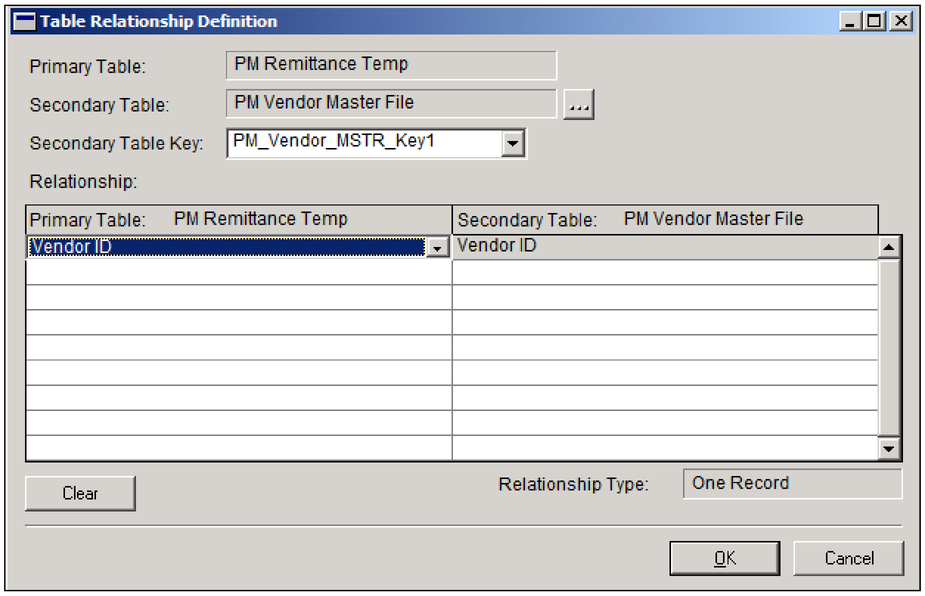
Now that we've explored making formatting changes to Word templates, let's dig a little deeper and take a look at what is required to add new information to an existing Word template. In this example, we'll modify the `Check Remittance` report by adding the **Vendor Contact** name from the **Vendor Master** table to the **Remittance Header** section of the report. We will first add it to the underlying Report Writer report, and then we will add it to the Word template.

Adding new fields to the Check Remittance template

Remember, as we said earlier, adding new fields to a Word template requires that we first make this change in the underlying report definition. To do this, we'll need to use Report Writer. Since we've already covered some basic modifications using Report Writer, we won't need to go into as much detail in this section as we will with making the change to the actual Word template:

1. Open Report Writer by selecting **Microsoft Dynamics GP** | **Tools** | **Customize** | **Report Writer**.

   Before we go any further, we should note that the **Vendor Contact** field is found in the **PM Vendor Master File** table. This table is not included in the default report definition for the Check Remittance, nor does a pre-existing relationship exist between this table and any of the existing tables in the report definition. We must first create a table relationship between the **PM Remittance Temp** table and the **PM Vendor Master File**. Click on the **Tables** button to find the **pmRemittance Temp** table and create a relationship between that table and the **PM Vendor Master File** table. The **Table Relationship Definition** window should look like the following screenshot:



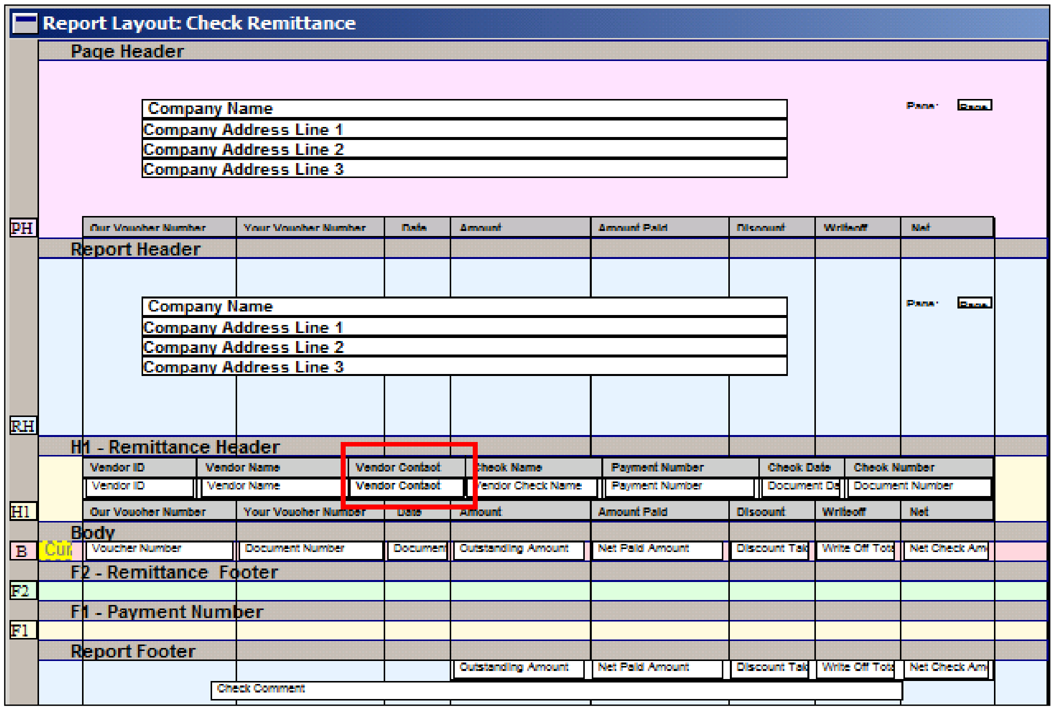
Primary Table: PM Remittance Temp
Secondary Table: PM Vendor Master File
Secondary Table Key: PM_Vendor_MSTR_Key1
Relationship:

Primary Table: PM Remittance Temp	Secondary Table: PM Vendor Master File
Vendor ID	Vendor ID

Relationship Type: One Record

Once the relationship is made, we can open our report to make the necessary modifications.

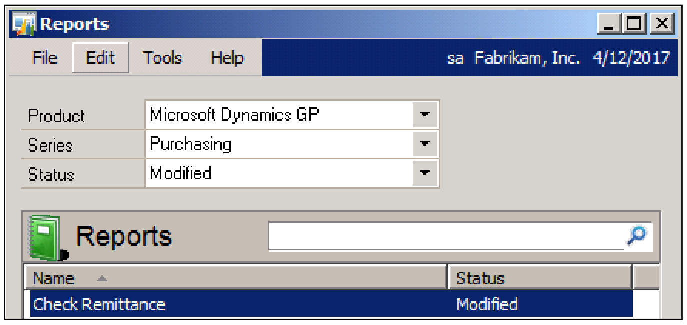
2. Click on the **Reports** button to open the list of reports. Find and select the **Check Remittance** report and click on the **Insert** button to add it to the list of **Modified Reports**.
3. With **Check Remittance** selected in the **Modified Reports** pane, click on **Open** to open the **Report Definition** for the Check Remittance report.
4. From the **Report Definition** window, select **Tables** to open the **Report Table Relationships** window. Click on the **New** button to open the **Related Tables** window. Select the **PM Vendor Master File** table and click on **OK** and then **Close** to create this report table relationship.
5. Back on the **Report Definition** window, click on the **Layout** button to open the **Report Layout** window.
6. In the **Toolbox** window that opens alongside the report layout, select **PM Vendor Master File** from the drop-down.
7. Find and select the **Vendor Contact** field. Click-and-drag this field into the **H1 – Remittance Header** section of the report layout. Move the existing fields around to make room for the newly added field. Use the **Properties** window to modify the formatting of the new field to match the formatting of the existing fields in the same section. Additionally, add a text string to display the words **Vendor Contact** over the newly added field. When all modifications are made, our report should look similar to the following screenshot:



8. With all changes made, save and close the report layout and report definition.
9. Navigate to **File** | **Microsoft Dynamics GP** to return to the GP 2013 application.

With the changes made to the underlying report definition, our focus now turns to modifying the Word template to reflect these changes. We return to the **Report Template Maintenance** window to make the following changes:

1. Navigate to **Reports** | **Template Maintenance** to open the **Report Template Maintenance** window.
2. Click on the **Report Name** field in the area that says: **Click here to select a report**.
3. Select **More Reports** to open the **Reports** window.
4. In the **Product** field, select **Microsoft Dynamics GP**.
5. In the **Series** field, select **Purchasing**.
6. In the **Status** field, select **Modified**.
7. Scroll down to find and select the **Check Remittance** report.



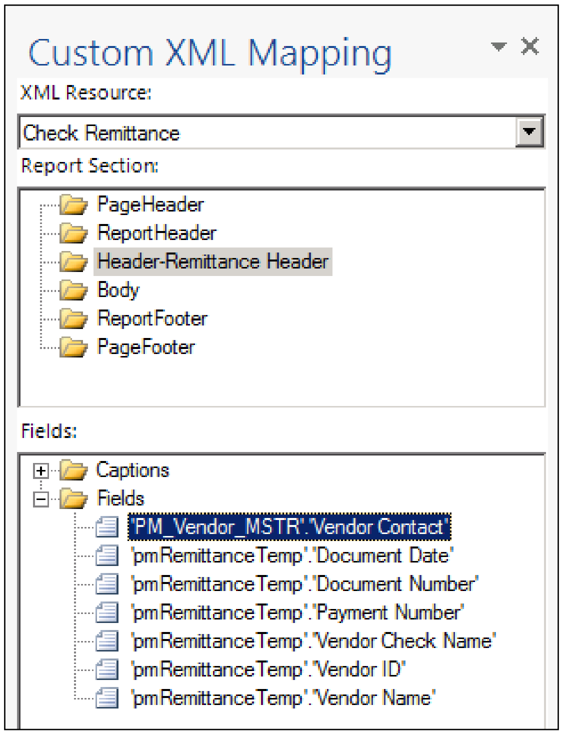
Product: Microsoft Dynamics GP
Series: Purchasing
Status: Modified

Reports

Name	Status
Check Remittance	Modified

8. Click on the **Select** button to return to the **Report Template Maintenance** window.
9. Click on the **New** button to open the **New Template** window.
10. Select **From Existing Template**.
11. Enter the value `Check Remittance Template – Vendor Contact` in the **Template Name** field and click on **Create** to return to the **Report Template Maintenance** window.

12. Select the template in the **Template Name** pane and click on **Modify** to open the newly created template in Word for modification.
13. With Word open, select the **Developer** tab on the ribbon. Select the **Field List** icon in the **Microsoft Dynamics GP Templates** section of this tab. If you do not see the **Developer** tab or the **Microsoft Dynamics GP Templates** section, refer to the earlier section on *Installing the Microsoft Dynamics GP add-in for Microsoft Word*.
14. In the newly opened field's list, select the drop-down under **XML Resource** and select **Check Remittance**. We are now shown a list of the sections in this report. Selecting a section displays a list of the fields that are available for use in that section. This is why it is so important that we modify the underlying report definition in Report Writer first; if we had not added the **Vendor Contact** field to the report in Report Writer, we would not be able to select that field in the Word template.

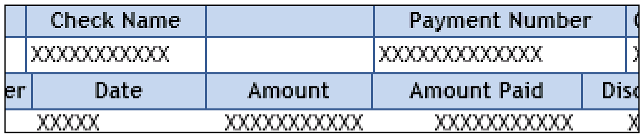


Check Remittance

PageHeader
ReportHeader
Header-Remittance Header
Body
ReportFooter
PageFooter

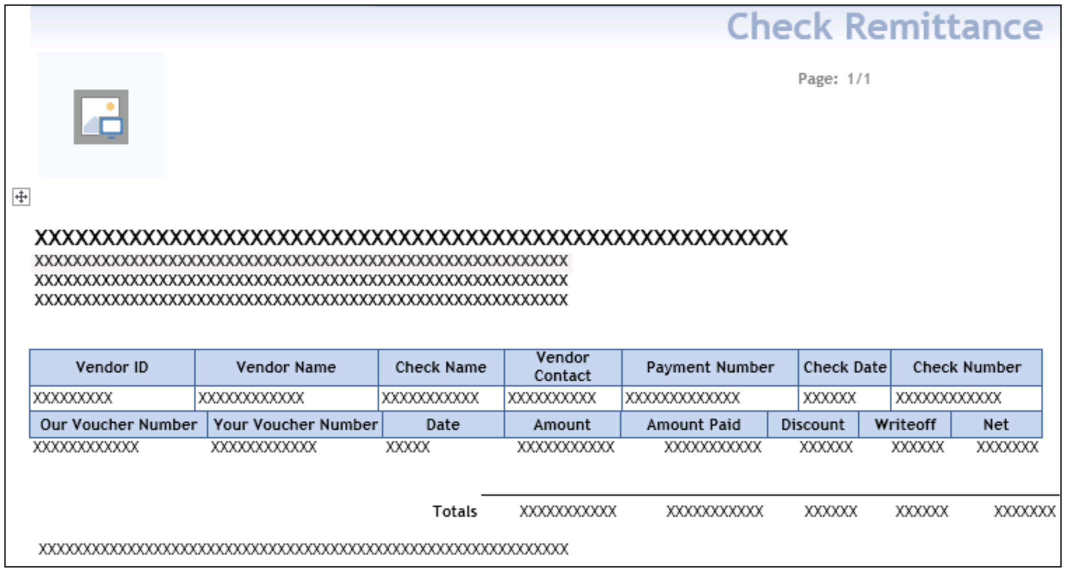
Captions
Fields
'PM_Vendor_MSTR'.'Vendor Contact'
'pmRemittanceTemp'.'Document Date'
'pmRemittanceTemp'.'Document Number'
'pmRemittanceTemp'.'Payment Number'
'pmRemittanceTemp'.'Vendor Check Name'
'pmRemittanceTemp'.'Vendor ID'
'pmRemittanceTemp'.'Vendor Name'

15. In the Word document, find the cell containing the words **Check Name**. Right-click on this cell and select **Split Cells**. Split the cell into two columns. Repeat this step for the cell below the newly split cells. This creates a new set of cells where we can add our **Vendor Contact** caption and **Vendor Contact** field, but prevents the cells to the right from going off the edge of the page.



	Check Name		Payment Number	(
	XXXXXXXXXXX		XXXXXXXXXXXXX	X
er	Date	Amount	Amount Paid	Disc
	XXXXX	XXXXXXXXXXX	XXXXXXXXXXX	X

16. In the **Report Section** of the **Field List**, select **Header – Remittance Header**. In the **Fields** section, under the **Captions** folder, select **Vendor Contact**.
17. Click-and-drag the **Vendor Contact** caption to the upper-most cell that was just added to the template. At this point, it may be tempting to just type the words in this cell, but this is not recommended! In fact, although these words will appear in the template, when printing this template, the words will no longer appear. We must use the captions and fields from the **Field List** to generate and display data we want to see in the template!
18. Return to the **Field List** and select **PM_Vendor_MSTR.Vendor Contact** under the **Fields** folder.
19. Click-and-drag the **PM_Vendor_MSTR.Vendor Contact** field to the upper-most cell that was just added to the template.
20. Compare the newly added caption and field to surrounding cells to ensure that they are formatted and aligned in similar fashion. Use the **View Gridlines** mode under **Table Tools | Layout** to more easily see the table lines so that aligning columns is easier to manage. Use the **Home** and **Table Tools | Layout** tabs on the ribbon to make these changes. At the end, your report template should look similar to the following screenshot:


Check Remittance

Page: 1/1

XXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXX
XXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXX
XXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXX
XXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXX

Vendor ID	Vendor Name	Check Name	Vendor Contact	Payment Number	Check Date	Check Number
XXXXXXXXX	XXXXXXXXXXXX	XXXXXXXXXXX	XXXXXXXXXX	XXXXXXXXXXXXX	XXXXXX	XXXXXXXXXXXX

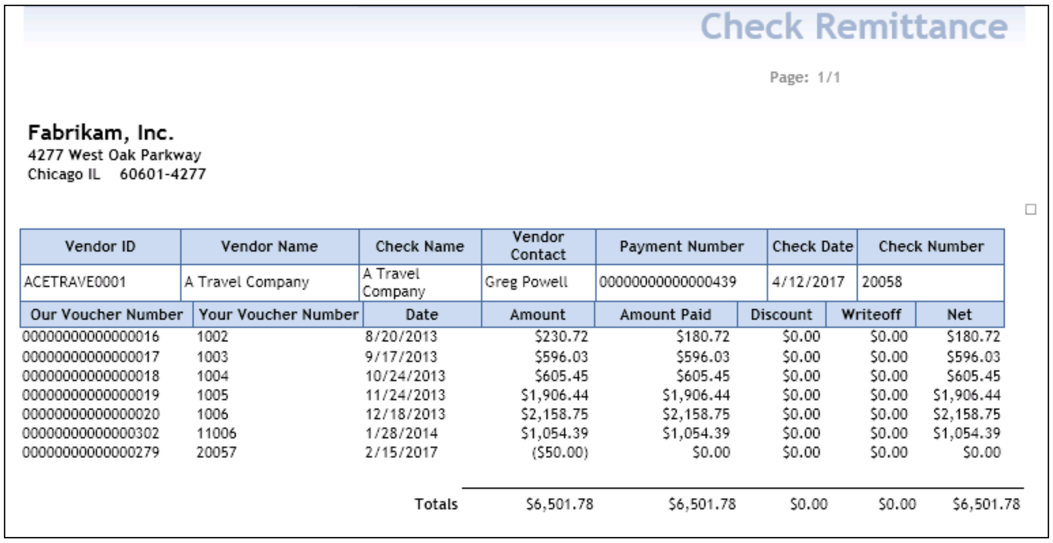
Our Voucher Number	Your Voucher Number	Date	Amount	Amount Paid	Discount	Writeoff	Net
XXXXXXXXXXXX	XXXXXXXXXXXX	XXXXX	XXXXXXXXXXX	XXXXXXXXXXX	XXXXXX	XXXXXX	XXXXXXX
		Totals	XXXXXXXXXXX	XXXXXXXXXXX	XXXXXX	XXXXXX	XXXXXXX

XXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXX

21. With all modifications made to the report, navigate to **File** | **Save As**.
22. Save the template to a file directory location that can be accessed in a later step. If an option exists in the file directory to **Maintain compatibility with previous versions**, select this option. Once the template is saved, close the Word document.
23. Return to the **Report Template Maintenance** window. In the **Template Name** pane, select the report template that was just renamed and modified.
24. Click on the green plus sign at the top right-hand corner of the **Template Name** pane.
25. In the file directory, browse out to the location in which the modified template was saved. Select that template and click on **Open**. When the override warning message appears, select **Yes** to replace the template.

At this point, we must now grant access to the modified report via the **Alternate/Modified Forms and Reports** window and assign this report as the default for this company. Steps on how to do this were included at the end of the earlier section, Modifying the presentation of a default Word template. Refer to these instructions to set the newly modified template as the default template when the **Template** option is selected for the **Check Remittance** report.

Once we've set this template as the default, let's print a **Check Remittance** report to see what our efforts have created!



Check Remittance

Page: 1/1

Fabrikam, Inc.
4277 West Oak Parkway
Chicago IL 60601-4277

Vendor ID	Vendor Name	Check Name	Vendor Contact	Payment Number	Check Date	Check Number
ACETRAVE0001	A Travel Company	A Travel Company	Greg Powell	0000000000000439	4/12/2017	20058

Our Voucher Number	Your Voucher Number	Date	Amount	Amount Paid	Discount	Writeoff	Net
0000000000000016	1002	8/20/2013	$230.72	$180.72	$0.00	$0.00	$180.72
0000000000000017	1003	9/17/2013	$596.03	$596.03	$0.00	$0.00	$596.03
0000000000000018	1004	10/24/2013	$605.45	$605.45	$0.00	$0.00	$605.45
0000000000000019	1005	11/24/2013	$1,906.44	$1,906.44	$0.00	$0.00	$1,906.44
0000000000000020	1006	12/18/2013	$2,158.75	$2,158.75	$0.00	$0.00	$2,158.75
0000000000000302	11006	1/28/2014	$1,054.39	$1,054.39	$0.00	$0.00	$1,054.39
0000000000000279	20057	2/15/2017	($50.00)	$0.00	$0.00	$0.00	$0.00
		Totals	$6,501.78	$6,501.78	$0.00	$0.00	$6,501.78

Now that we know how to modify existing report templates, we can use these techniques to make simple or complex changes to our existing templates. Now, let's take a look at how we can create entirely new report templates when an existing template does not already exist for us to modify.

Using Word Template Generator to create additional Word templates

In Dynamics GP 2013, users can utilize the Word Template Generator to create Word templates for any Dynamics GP report. As mentioned earlier, GP comes pre-loaded with 32 reports already configured with Word templates. If a report is not included in this list and users want to see it rendered in Word, then one of two options exist:

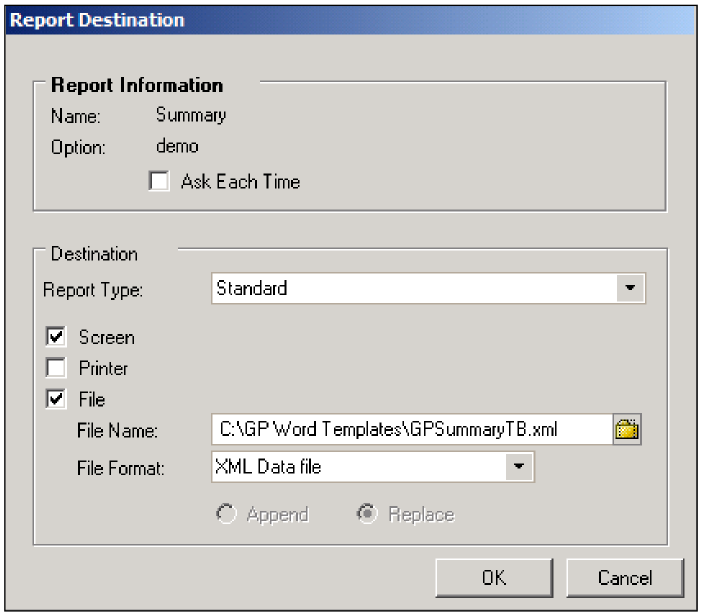
- Manually create a new Word report template from scratch.
- Use the Word Template Generator utility to create the Word Template based on the original version.

It can be quite time-consuming to create Word report templates from scratch, so it is highly likely most users will use the Word Template Generator. Let's create a Word Template for our Summary GL Trial Balance report.

Create a Word template for Summary GL Trial Balance

The Report Writer user guide has a nice section on the Word Template functionality in Dynamics GP 2013. One section that may be of particular value to users is one that details specific reports that have known issues with the Word Template Generator. Additionally, users should be aware that certain reports containing more complex Report Writer layouts will have issues converting cleanly to Word Templates. Be sure to refer to the section on Generated template expectations for more information before beginning to use the Word Template Generator.

1. Our first step is to generate the report in GP. Navigate to the GL Summary Trial Balance by navigating to **Reports** | **Financial** | **Trial Balance**. Change the **Reports** drop-down to **Summary**.
2. Create a new report option, or modify an existing option so that we can set the report destination properly. In the **Trial Balance Report Options** window, click on the **Destination** button to open the **Report Destination** window.
3. Choose to print the report to **Screen** and to **File**. In the **File Format** drop-down, select the **XML Data File** format. Click on the folder icon in the **File Name** field and browse to a location where the XML file should be saved. It may be necessary to change the **File Name** and/or the **Save As Type** drop-down menu to **XML Data File**.

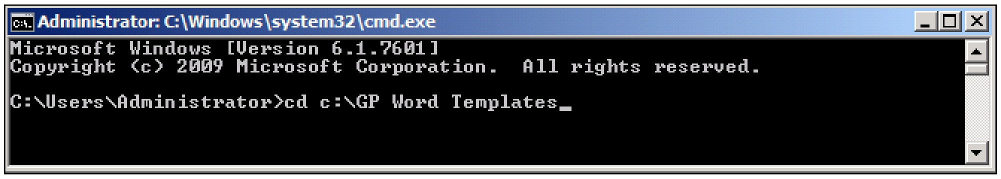


Report Destination

Report Information
Name: Summary
Option: demo
☐ Ask Each Time

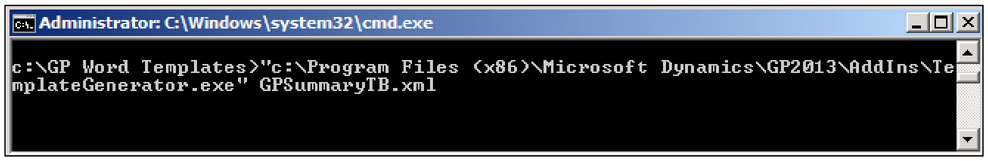
Destination
Report Type: Standard
☑ Screen
☐ Printer
☑ File
File Name: C:\GP Word Templates\GPSummaryTB.xml
File Format: XML Data file
Append [✓ Replace]

4. Click on **OK**, then **Print**, to generate the report. The report should generate to the screen and an XML file should appear in the location specified in the **Report Destination** window. Before closing the screen output, make a note of the report name in the menu bar. In this case, we see the report name is **Trial Balance Summary**.
5. Outside GP, open a command prompt.
6. Use the following command to change the working folder to the one in which the XML file was saved. In this case, the XML file was saved in a folder at `C:\GP Word Templates`.



```
Microsoft Windows [Version 6.1.7601]
Copyright (c) 2009 Microsoft Corporation.  All rights reserved.

C:\Users\Administrator>cd c:\GP Word Templates_
```

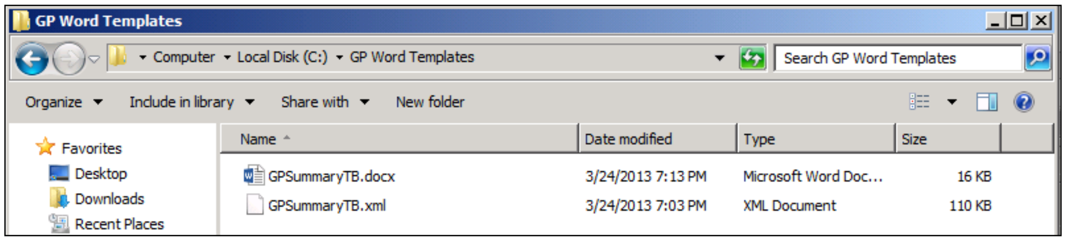
7. Now, use the following command (this may require some modification depending on the environment) to generate the Word template document for the selected report. In this case, the report was saved using the file name of `GPSummaryTB.xml`. Additionally, note that the `TemplateGenerator.exe` file used for this process is located in the `AddIns` folder of the default Dynamics folder.



```
c:\GP Word Templates>"c:\Program Files (x86)\Microsoft Dynamics\GP2013\AddIns\Te
mplateGenerator.exe" GPSummaryTB.xml
```

In addition to using the steps described here that require using the command prompt, users can also use the Windows File Explorer to drag-and-drop the `.xml` file onto the `.exe` file for the Word Template Generator. This will generate the template file in the target folder. To some, this may sound like the easier method to use; however, in many complex environments, it is easier to simply use the command prompt to specify the `.xml` file and `.exe` file locations. Either way, you have a couple of options depending on your needs!

8. In the folder where we saved our XML file, a new file with extension `.docx` will be created. Verify that this file was created.

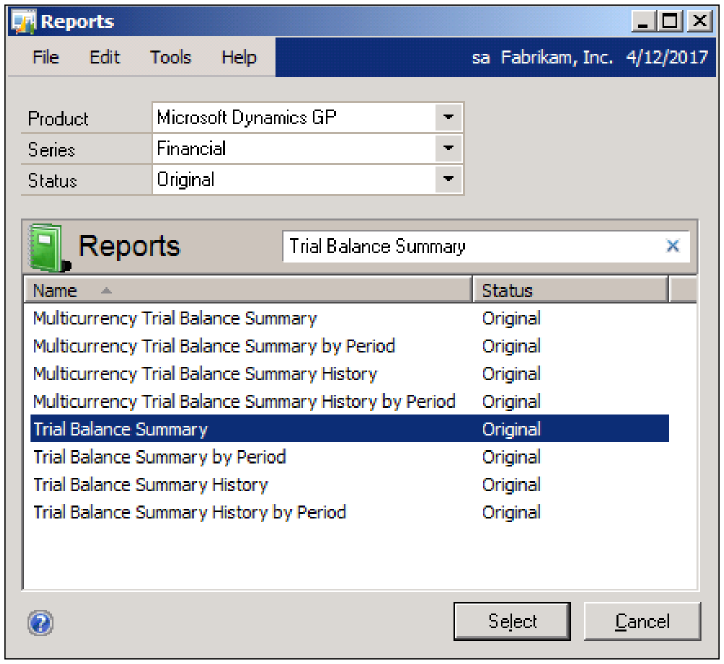

GP Word Templates

Computer ▸ Local Disk (C:) ▸ GP Word Templates

Search GP Word Templates

Name	Date modified	Type	Size
GPSummaryTB.docx	3/24/2013 7:13 PM	Microsoft Word Doc...	16 KB
GPSummaryTB.xml	3/24/2013 7:03 PM	XML Document	110 KB

9. Return to GP 2013 and open the **Report Template Maintenance** window by navigating to **Reports** | **Template Maintenance**.
10. In the **Report Name** drop-down, select **More Reports** to open the **Reports** window. In the **Product** drop-down, select **Microsoft Dynamics GP**; in the **Series** drop-down, select **Financials**; and in the **Status** drop-down, select **Original**. Find and select the **Trial Balance Summary** report (remember, we verified the report name in an earlier step).


Reports

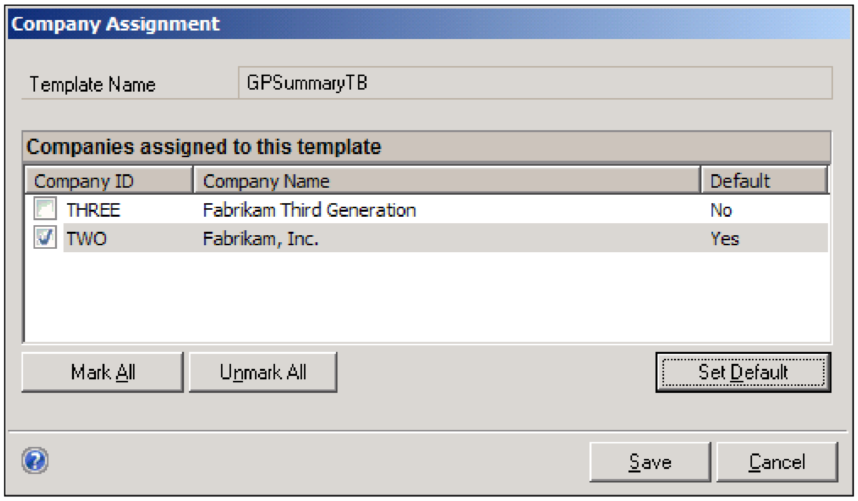
sa Fabrikam, Inc. 4/12/2017

Product: Microsoft Dynamics GP
Series: Financial
Status: Original

Reports: Trial Balance Summary

Name	Status
Multicurrency Trial Balance Summary	Original
Multicurrency Trial Balance Summary by Period	Original
Multicurrency Trial Balance Summary History	Original
Multicurrency Trial Balance Summary History by Period	Original
Trial Balance Summary	Original
Trial Balance Summary by Period	Original
Trial Balance Summary History	Original
Trial Balance Summary History by Period	Original

11. Click on the **Select** button to return to the **Report Template Maintenance** window. Click on the **Add Template** button that looks like a green plus sign. In the **Add Template...** window, browse to the folder containing the newly created `.docx` file. Select that file and select the **Open** button. This adds the newly created Word template to GP.
12. Before closing the **Report Template Maintenance** window, click on the **Assign** button and click on **Company** to open the **Company Assignment** window for this report template. Choose any companies to which this template should be assigned. Optionally, click on the **Set Default** button to mark this template as the default template. Once this is complete, click on the **Save** button.

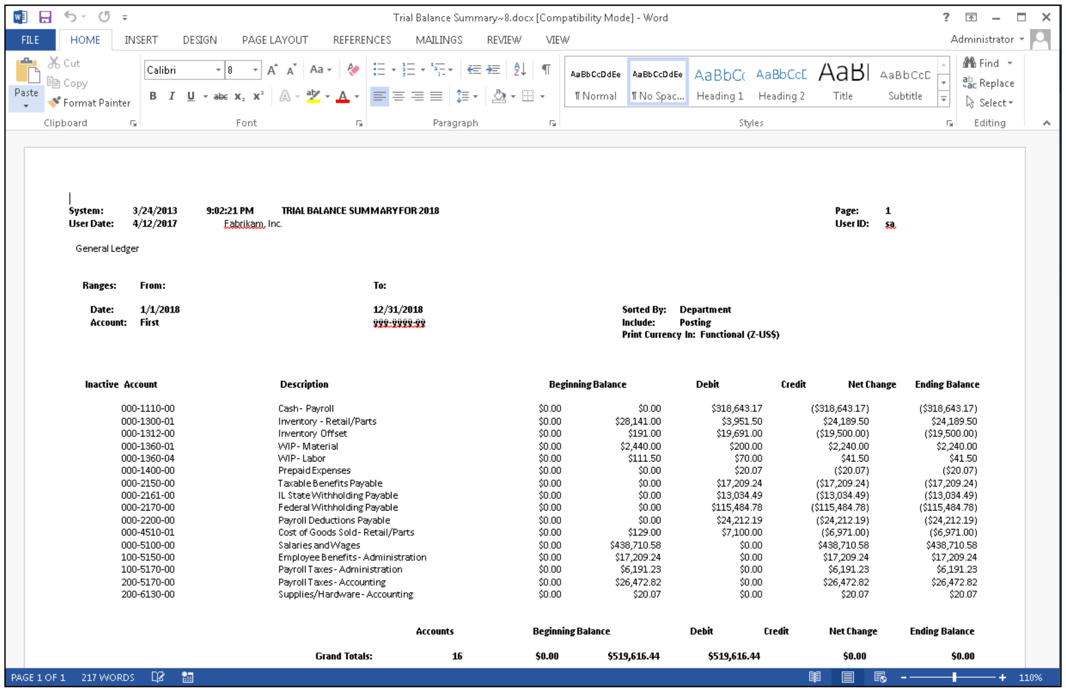


Template Name: GPSummaryTB

Companies assigned to this template

Company ID	Company Name	Default
☐ THREE	Fabrikam Third Generation	No
✓ TWO	Fabrikam, Inc.	Yes

13. Close the **Report Template Maintenance** window and return to the **Report Options** window for the Summary Trial Balance report option we used earlier. Click on the **Destination** button and set the **Report Type** destination to **Template**. Print the report to the screen and verify that our report now renders properly in Word format.



```
System:     3/24/2013    9:02:21 PM    TRIAL BALANCE SUMMARY FOR 2018                                   Page:     1
User Date:  4/12/2017        Fabrikam, Inc.                                                             User ID:  sa

 General Ledger

   Ranges:     From:                               To:

     Date:     1/1/2018                            12/31/2018                  Sorted By:    Department
     Account:  First                               ÿÿÿ-ÿÿÿÿ-ÿÿ                 Include:      Posting
                                                                               Print Currency In:  Functional (Z-US$)

  Inactive  Account             Description                         Beginning Balance      Debit        Credit      Net Change     Ending Balance

            000-1110-00         Cash - Payroll                      $0.00        $0.00     $318,643.17   ($318,643.17)   ($318,643.17)
            000-1300-01         Inventory - Retail/Parts            $0.00   $28,141.00       $3,951.50     $24,189.50      $24,189.50
            000-1312-00         Inventory Offset                    $0.00      $191.00      $19,691.00    ($19,500.00)    ($19,500.00)
            000-1360-01         WIP - Material                      $0.00    $2,440.00         $200.00      $2,240.00       $2,240.00
            000-1360-04         WIP - Labor                         $0.00      $111.50          $70.00         $41.50          $41.50
            000-1400-00         Prepaid Expenses                    $0.00        $0.00          $20.07        ($20.07)        ($20.07)
            000-2150-00         Taxable Benefits Payable            $0.00        $0.00      $17,209.24    ($17,209.24)    ($17,209.24)
            000-2161-00         IL State Withholding Payable        $0.00        $0.00      $13,034.49    ($13,034.49)    ($13,034.49)
            000-2170-00         Federal Withholding Payable         $0.00        $0.00     $115,484.78   ($115,484.78)   ($115,484.78)
            000-2200-00         Payroll Deductions Payable          $0.00        $0.00      $24,212.19    ($24,212.19)    ($24,212.19)
            000-4510-01         Cost of Goods Sold - Retail/Parts   $0.00      $129.00       $7,100.00     ($6,971.00)     ($6,971.00)
            000-5100-00         Salaries and Wages                  $0.00  $438,710.58           $0.00    $438,710.58     $438,710.58
            100-5150-00         Employee Benefits - Administration  $0.00   $17,209.24           $0.00     $17,209.24      $17,209.24
            100-5170-00         Payroll Taxes - Administration      $0.00    $6,191.23           $0.00      $6,191.23       $6,191.23
            200-5170-00         Payroll Taxes - Accounting          $0.00   $26,472.82           $0.00     $26,472.82      $26,472.82
            200-6130-00         Supplies/Hardware - Accounting      $0.00       $20.07           $0.00         $20.07          $20.07

                                            Accounts     Beginning Balance       Debit       Credit     Net Change     Ending Balance

                        Grand Totals:          16      $0.00    $519,616.44    $519,616.44            $0.00            $0.00
```

Congratulations, you now have the ability to create Word templates for nearly any report in GP!

Summary

In this chapter, we covered the Report Writer tool in GP 2013. As one of the oldest report writing tools associated with Dynamics GP, numerous reports in the current versions of GP originated and are still maintained with Report Writer. Although newer, easier to use reporting tools continue to be developed, developers and consultants will still find themselves needing a bit of Report Writer know-how in order to make modifications to the standard GP 2013 reports!

Just as we covered one of the oldest reporting tools in the Dynamics GP space, we also took a look at one of the newest tools in the same arena. In an effort to provide us with more user-friendly, portable reports, Microsoft added the Word template functionality to Dynamics GP 2010 and expanded on it in later versions by offering several new Word templates and a very useful Word Template Generator. Rather than printing clunky Report Writer reports to screen, numerous reports can now be printed to a Microsoft Word document. This Word document can then be formatted, saved, and even attached to an email document. Many people have been hesitant to touch the Word template functionality because of its close ties to Report Writer, but we believe that what we've covered in this chapter should give you the knowledge you need to approach this tool with confidence!

In the next chapter, we'll explore the SSRS Reports Library and the reports that comprise this toolset. The SSRS Reports Library offers us a predefined set of reports that can be deployed to an SSRS environment, but we'll also explore the basics of creating a simple SSRS report from scratch.

5
Utilizing the SSRS Report Library

Of the reporting tools covered in this book, Microsoft **SQL Server Reporting Services** (**SSRS**) is perhaps the most well-known and widely used tool of all. As one of the components of SQL Server, SSRS can be used for reporting across a wide range of data sources. SSRS provides a mechanism for designing and creating reports, publishing reports to a report server, managing access to these reports through role-based security, and viewing these reports through a web-based connection. In short, SSRS is capable of managing the entire reporting process from beginning to end.

Because of its versatility and sophistication, however, SSRS can be a challenging tool to learn in the context of being used with a single application such as Dynamics GP 2013. In years past, report development via SSRS has often been considered the domain of the technical experts on staff. But this is changing. With recent versions of GP, even the less technically adept developers and consultants are now able to add SSRS as an important instrument in their reporting toolset. Dynamics GP 10.0 introduced a series of predefined SSRS-based reports that cover a wide range of modules and can easily be deployed in an environment in which SSRS is installed and configured properly. These SSRS reports are accessible just as any other SSRS report with one exception: we can also modify GP to allow us to access these SSRS reports directly from GP.

Dynamics GP 2010 took the predefined SSRS report concept a step further and introduced us to SSRS-based **Charts** and **Key Performance Indicators** (**KPI**). Although charts and KPIs have been available in earlier versions of GP, they were not SSRS-based and could not be easily modified. With GP 2010, developers and consultants were able to deploy a series of predefined charts and KPIs and they can also create their own metrics that can be displayed on the GP Homepage. Now with GP 2013, some of these very same reports can be launched directly from master records in GP (that is, Customer Maintenance can launch a sales analysis report).

Obviously, a lot can be achieved with SSRS. Our goal with this chapter is not to provide a step-by-step guide to using SSRS in order to create reports from scratch. That would require far more than just a single chapter. To be quite honest, numerous resources already exist that describe SSRS in far greater depth than we can here. For readers who are interested in learning how to use SSRS to its fullest capabilities, we strongly suggest that you take the time to discover and read through those resources.

What we will cover in this chapter, however, is how the predefined SSRS reports and charts can be used to jumpstart our use of SSRS as a valuable tool for our organization. Using these reports will allow us to form a base from which far more effective and useful reports can be developed. In this chapter, we'll cover the following topics:

- Getting started with, and the prerequisites for, Reporting Services
- Deploying the predefined SSRS reports and metrics to the report server
- Exposing the predefined reports and metrics in Dynamics GP 2013 with Business Analyzer
- Managing security for these reports
- Using Visual Studio to modify a predefined SSRS report
- Using Report Builder to create a new metric

Getting started with SSRS

Before we can deploy and utilize the SSRS reports available for Dynamics GP, we must have an SSRS Report Manager site available to us. Installing and configuring this is actually a very easy task. Dynamics GP supports Windows Server 2008, 2008 R2, and 2012, and SQL Server Versions 2008 w/SP1 or later, 2008 R2 and 2012. For purposes of this book, we assume that SSRS is already installed and properly configured.

Prerequisites for SSRS

In order to use Reporting Services, there are quite a few components that must be installed before we can install SSRS for use with Dynamics GP. The following chart from Microsoft lists the necessary operating system, database, web server, and web browser requirements:

Item	**Requirements**
Operating system 32-bit and 64-bit supported	**Server** Windows Server 2008 with latest service pack Windows Server 2008 R2 with latest service pack Windows Server 2012 (native mode only) **Client** Windows 7 with latest service pack
Database	Microsoft SQL Server 2008 with service pack 1 or later Microsoft SQL Server 2008 R2 Microsoft SQL Server 2012
SharePoint	Microsoft SharePoint Foundation 2010 with service pack 1 Microsoft SharePoint Server 2010 with service pack 1
Web browser	Internet Explorer 9 Internet Explorer 8
Microsoft Dynamics GP	Microsoft Dynamics GP 2013

Deploying SSRS reports and metrics

A new feature was added beginning with GP 10.0 that allows users to deploy predefined SSRS reports for use with GP data. Numerous reports span many GP modules, including:

- Financials and bank reconciliation
- Receivables Management and Payables Management
- Sales Order Processing and Purchasing Order Processing
- Inventory
- Project Accounting
- Payroll
- Human Resources
- Contract Administration
- Manufacturing
- Returns Management

If set up properly, these reports can be accessed from within GP, meaning users will spend less time browsing out to the SSRS landing page and finding the correct report in the correct folder. Instead, they will select from the list of custom reports to see the reports that were deployed using the **Reporting Tools Setup** for Dynamics GP.

Additionally, with the introduction of GP 2013, users are now able to deploy charts and metrics that are streamed through the Business Analyzer right on the users' home page when logging into the application. This feature is only available for users who are running on SQL Server 2008 or later and GP 2013.

Let's take a look at the steps required to deploy and utilize our predefined SSRS reports.

Deploying predefined Reporting Services reports and metrics for Dynamics GP

Once our Reporting Services instance is configured, we can begin deploying our reports for a single company database as follows:

1. Launch Dynamics GP 2013 by navigating to **Start** | **Programs** | **Microsoft Dynamics** | **GP 2013** | **GP**, and log in to a company.
2. Click on **Microsoft Dynamics GP** | **Tools** | **Setup** | **System** | **Reporting Tools Setup**. The **Reporting Services** tab should be selected.
3. Select **Native** for the **Report Server Mode** field.
4. In the **Report Server URL** field, enter the location of the report server. Depending on whether it is a **Native** or **SharePoint Integrated** mode installation, the syntax will vary as follows:

   ```
   http://localhost/ReportServer
   ```

   or

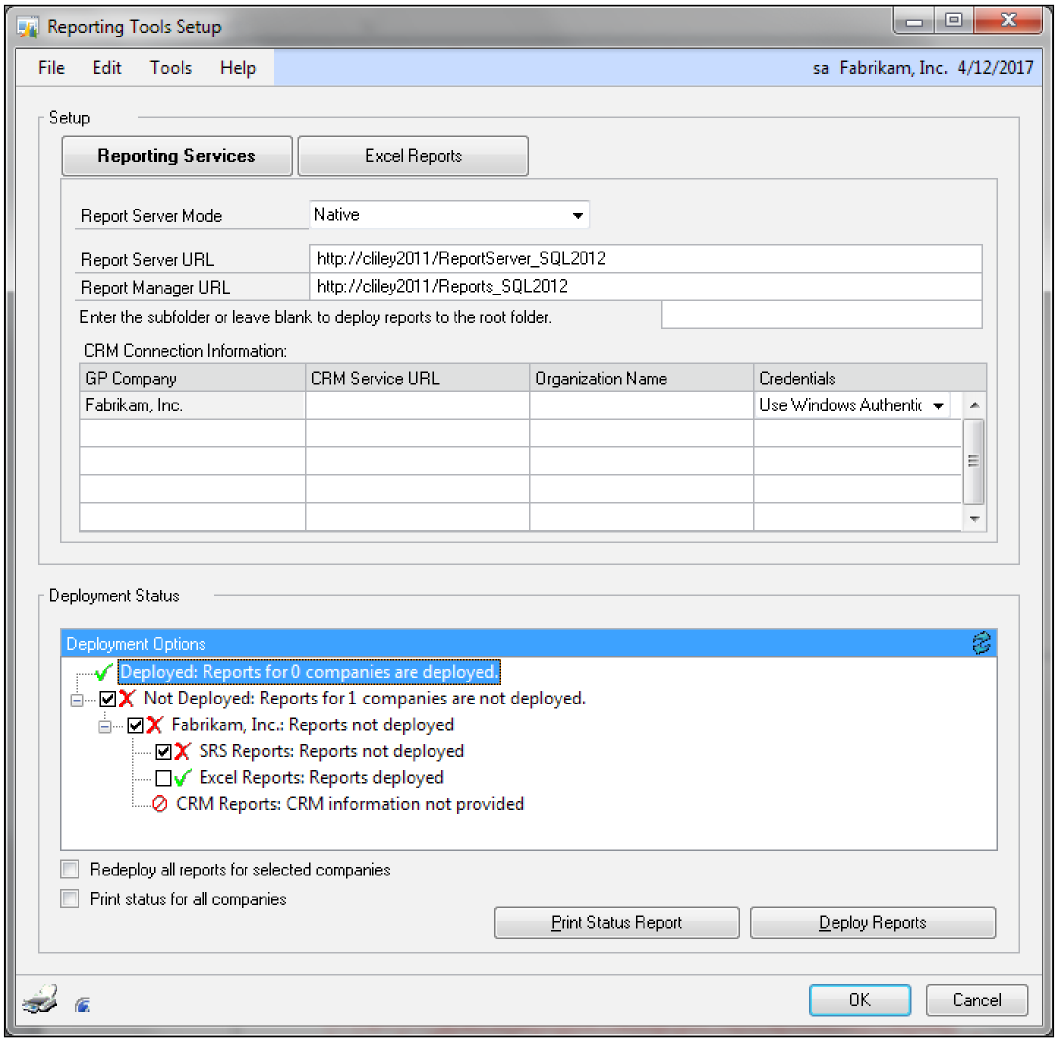
   ```
   http://IISServername/reports/Pages/Folder.aspx
   ```

5. In the **Report Manager URL** field, enter the location of the Report Manager. Depending on whether it's a **Native** or **SharePoint Integrated** mode installation, the syntax will vary:

   ```
   http://localhost/Reports
   ```

   or

   ```
   http://IISServername/reports/Pages/Folder.aspx
   ```



Report Server Mode: Native
Report Server URL: http://cliley2011/ReportServer_SQL2012
Report Manager URL: http://cliley2011/Reports_SQL2012
Enter the subfolder or leave blank to deploy reports to the root folder.

CRM Connection Information:

GP Company	CRM Service URL	Organization Name	Credentials
Fabrikam, Inc.			Use Windows Authentic

Deployment Status

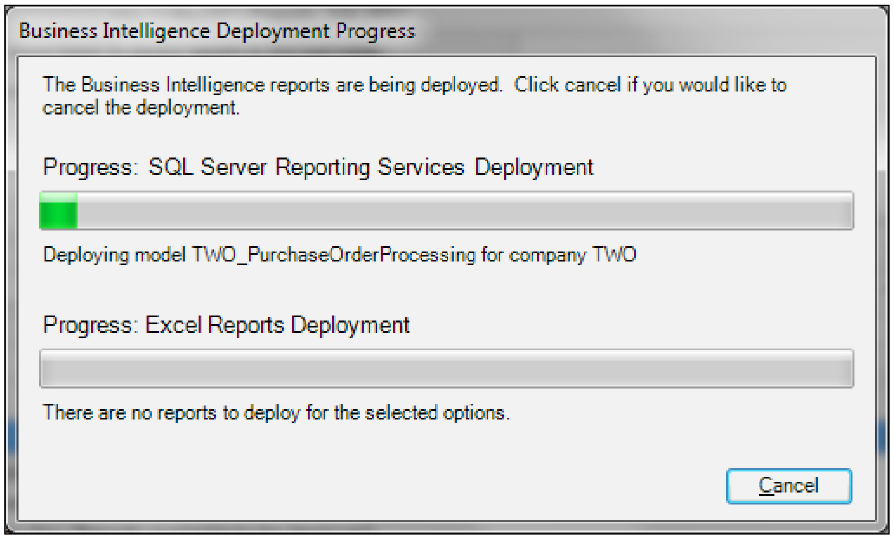
Deployment Options
- ✓ Deployed: Reports for 0 companies are deployed.
- ☑ ✗ Not Deployed: Reports for 1 companies are not deployed.
  - ☑ ✗ Fabrikam, Inc.: Reports not deployed
    - ☑ ✗ SRS Reports: Reports not deployed
    - ☐ ✓ Excel Reports: Reports deployed
    - ⊘ CRM Reports: CRM information not provided

☐ Redeploy all reports for selected companies
☐ Print status for all companies

If SSRS is using a port other than the default (80), the URLs listed in the previous steps may need to be modified to include the new port number. The URL would then look similar to the following (where 81 is the new port number): `http://localhost:81/ReportServer/ReportService.asmx`.

Also, if we deploy the reports in a multi-tenant environment with multiple instances of Dynamics GP, we can enter a folder name to deploy the reports to a single Reporting Services instance.

6. In the **Deployment Options** tree view, all items not currently deployed will be marked. We can exclude any item by unmarking the check box.
7. Click on **Deploy Reports**, and the **Business Intelligence Deployment Progress** window will appear. If you don't have appropriate permissions to deploy reports, a window will open that will prompt you for the domain/user.


Business Intelligence Deployment Progress

The Business Intelligence reports are being deployed. Click cancel if you would like to cancel the deployment.

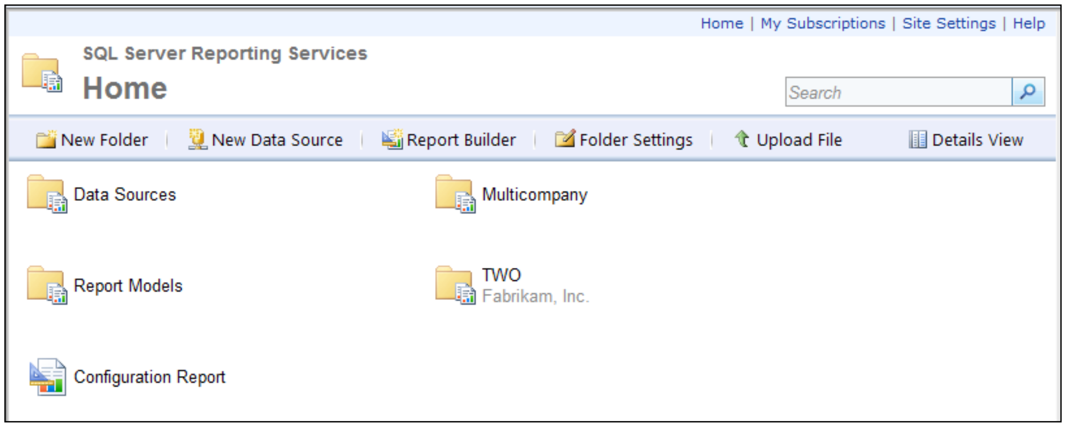
Progress: SQL Server Reporting Services Deployment

Deploying model TWO_PurchaseOrderProcessing for company TWO

Progress: Excel Reports Deployment

There are no reports to deploy for the selected options.

Once the reports have been deployed, they can be viewed from the Report Manager. The default URL for the Report Manager is `http://<ComputerName>/reports`. If we deploy these reports to the **Native** mode, the home page for SSRS will look similar to the following screenshot:



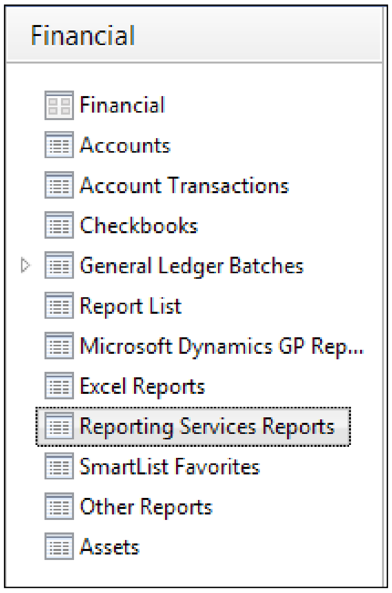
Notice that during the installation, several new folders were created. First, a `Data Sources` folder is created. This folder contains data sources for the `DYNAMICS` database as well as the GP company database selected during the install. A second folder is named after the GP company database. This folder contains the deployed reports as well as the charts and KPIs, grouped together in folders by specific modules. Additionally, there is a `Multicompany` folder that includes KPIs that are designed for multiple companies. Finally, we will see a `Report Models` folder containing all of the report models generated by this deployment.

Using the predefined SSRS reports

Now that we have deployed the predefined SSRS reports, there are multiple ways to access and use these reports with Dynamics GP 2013. We are able to view these reports via the functional area pages in their report list, by going directly to the Report Manager site or by using report assignment to assign reports to the available maintenance windows.

Viewing SSRS reports in the Dynamics GP report list

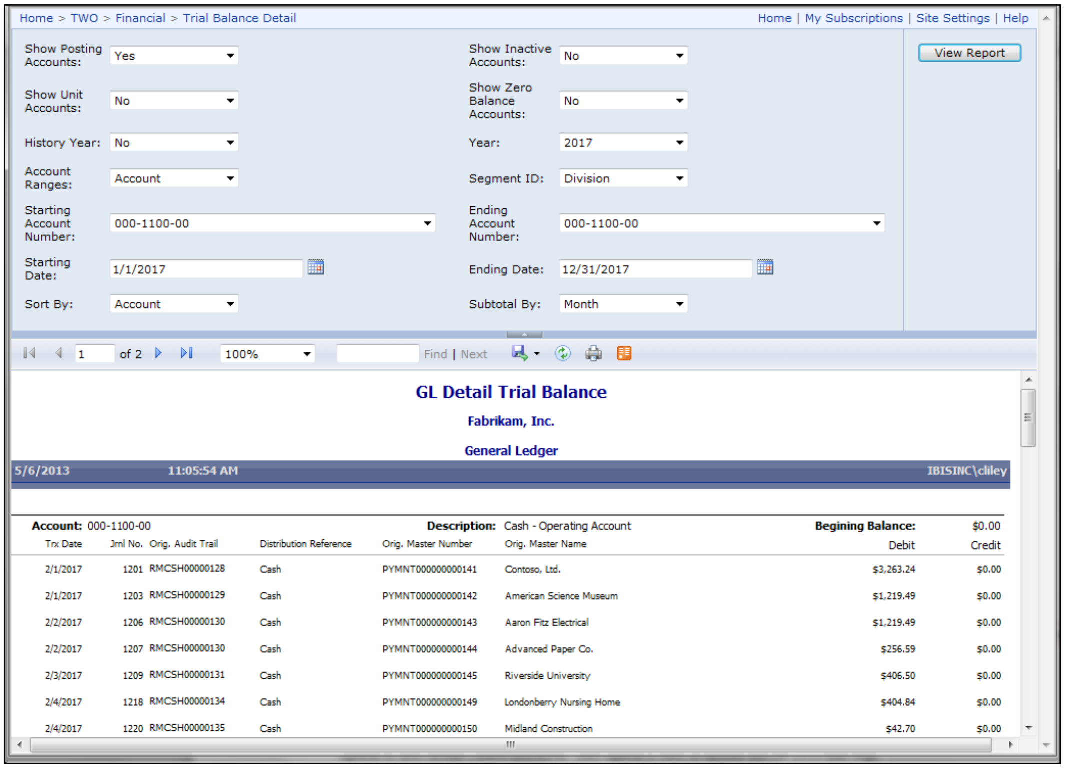
One of the more efficient ways for a GP user to access the predefined SSRS reports is directly through the application. As long as the reports are deployed in the correct folder structure through **Reporting Tools Setup**, they will be available in the area pages. Once a functional area page is open, selecting **Reporting Services Reports** will open a list of the deployed SSRS reports for the functional area.



Launching predefined SSRS reports directly from Report Manager

For those users that don't have or need direct access to the Dynamics GP application, the predefined SSRS reports can be accessed directly from the Report Manager website. Most of the predefined SSRS reports contain parameters that can be modified by the user in order to change the report content. Some reports have default values for parameters, enabling the report content to be viewed immediately upon opening the report. Others, however, require the user to specify values before the report content can be generated.

For example, one of the predefined reports found in the **Financial** module is **Trial Balance Detail**. Assuming we have followed the instructions for enabling the selection of the predefined reports directly from GP, we can scroll through our list of reports in the **Custom Reports** window and double-click on **Trial Balance Detail**. This opens the **Trial Balance Detail** report in SSRS, as seen in the following screenshot:


Home > TWO > Financial > Trial Balance Detail
Home | My Subscriptions | Site Settings | Help
View Report
1 of 2
100%
Find | Next

Show Posting Accounts:	Yes	Show Inactive Accounts:	No
Show Unit Accounts:	No	Show Zero Balance Accounts:	No
History Year:	No	Year:	2017
Account Ranges:	Account	Segment ID:	Division
Starting Account Number:	000-1100-00	Ending Account Number:	000-1100-00
Starting Date:	1/1/2017	Ending Date:	12/31/2017
Sort By:	Account	Subtotal By:	Month

GL Detail Trial Balance

Fabrikam, Inc.

General Ledger

5/6/2013 11:05:54 AM IBISINC\cliley

Account: 000-1100-00 **Description:** Cash - Operating Account **Begining Balance:** $0.00

Trx Date	Jrnl No.	Orig. Audit Trail	Distribution Reference	Orig. Master Number	Orig. Master Name	Debit	Credit
2/1/2017	1201	RMCSH00000128	Cash	PYMNT000000000141	Contoso, Ltd.	$3,263.24	$0.00
2/1/2017	1203	RMCSH00000129	Cash	PYMNT000000000142	American Science Museum	$1,219.49	$0.00
2/2/2017	1206	RMCSH00000130	Cash	PYMNT000000000143	Aaron Fitz Electrical	$1,219.49	$0.00
2/2/2017	1207	RMCSH00000130	Cash	PYMNT000000000144	Advanced Paper Co.	$256.59	$0.00
2/3/2017	1209	RMCSH00000131	Cash	PYMNT000000000145	Riverside University	$406.50	$0.00
2/4/2017	1218	RMCSH00000134	Cash	PYMNT000000000149	Londonberry Nursing Home	$404.84	$0.00
2/4/2017	1220	RMCSH00000135	Cash	PYMNT000000000150	Midland Construction	$42.70	$0.00

At the top of this particular report, we see 14 different parameters that can be used to control the content of the generated report. Although there are some default parameters, it is highly likely we will want to change some of the other parameters to control what appears in our report. Because we are using Fabrikam sample data in the displayed example, we can set our data parameters to pull transactions between January 1, 2017 and December 31, 2017 to view detailed transactions for the current year. Once we've selected or changed our parameters, we can select **View Report** to regenerate our report.

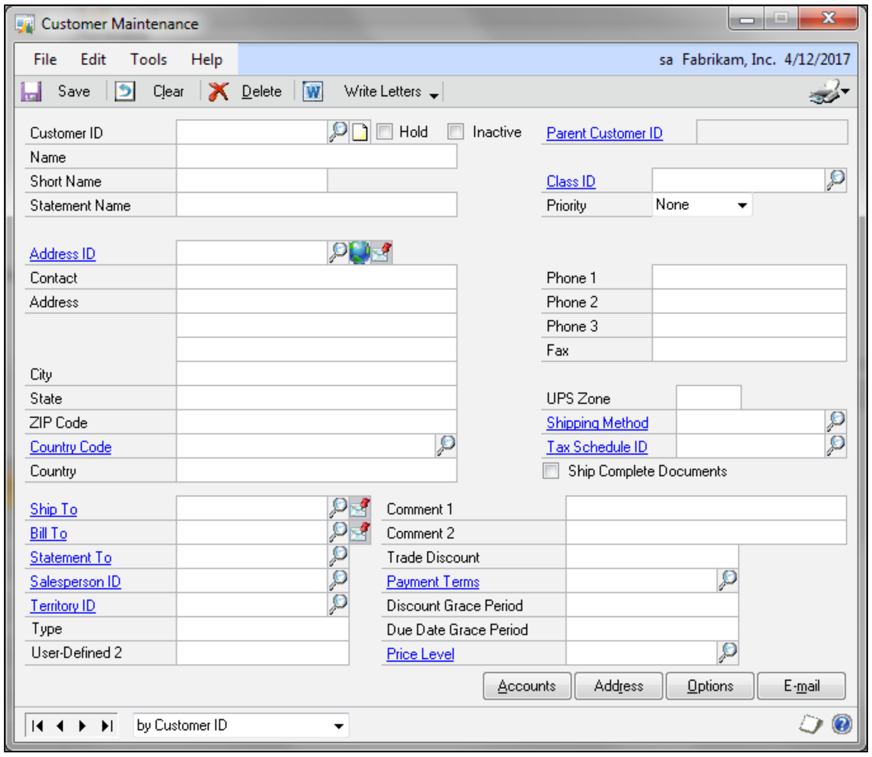
Assigning and using predefined SSRS reports on GP forms

With the release of GP 2013, if you have deployed the predefined SSRS reports, you can print these directly from GP 2013 windows. The following GP windows allow this functionality:

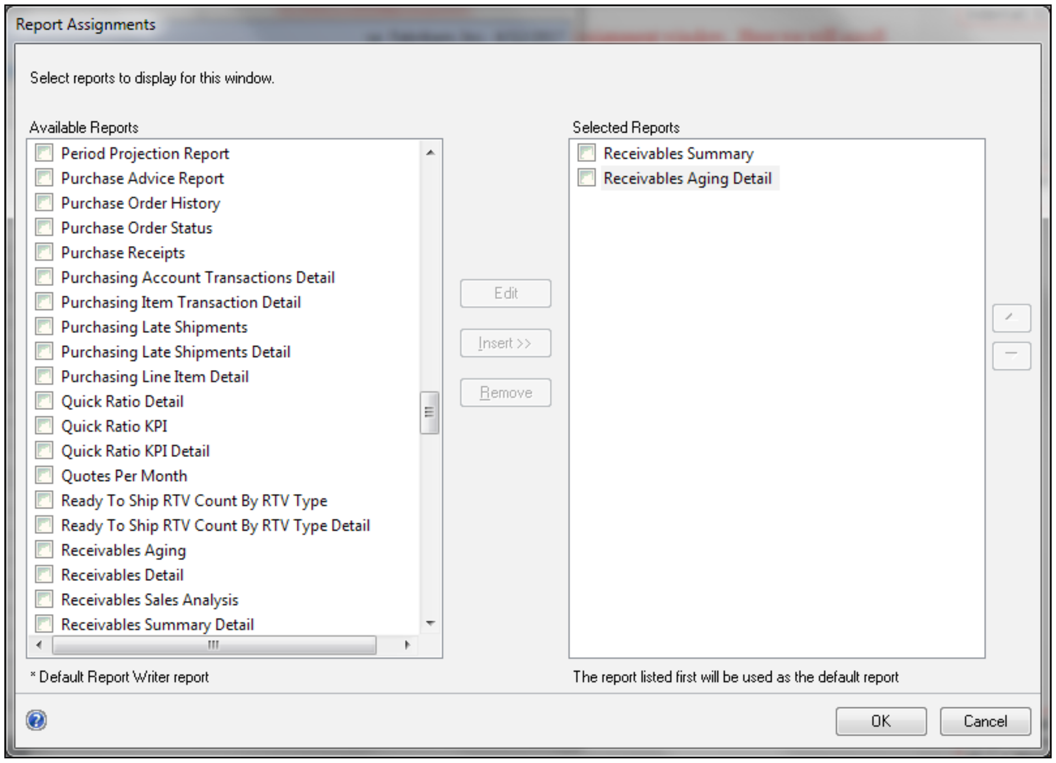
Account Maintenance	Checkbook Maintenance
Customer Maintenance	Salesperson Maintenance
Sales Prospect Maintenance	Vendor Maintenance
Employee Maintenance	Applicant Maintenance
Item Maintenance	

Before this functionality will work, we must first assign the reports to the specific window. Let's take a look at how to assign these reports, using the **Customer Maintenance** window as an example as follows:

1. Open **Customer Maintenance** by navigating to **Cards | Sales | Customer**.



2. In the upper-right corner of the **Customer Maintenance** window, we will see a printer icon with a drop-down menu to the right. Clicking on this drop-down menu will allow us to select **Assign Reports**.
3. We will be provided with the **Report Assignments** window. Here, we will scroll through the **Available Reports** list and select the report(s) that we want to assign to this window. Note that for this functionality to work, the report must accept the key parameter from the window, in this case `CUSTNMBR` (customer number). After selecting the report(s), click on insert to add them into the **Selected Reports** list.


Report Assignments

Select reports to display for this window.

Available Reports
- Period Projection Report
- Purchase Advice Report
- Purchase Order History
- Purchase Order Status
- Purchase Receipts
- Purchasing Account Transactions Detail
- Purchasing Item Transaction Detail
- Purchasing Late Shipments
- Purchasing Late Shipments Detail
- Purchasing Line Item Detail
- Quick Ratio Detail
- Quick Ratio KPI
- Quick Ratio KPI Detail
- Quotes Per Month
- Ready To Ship RTV Count By RTV Type
- Ready To Ship RTV Count By RTV Type Detail
- Receivables Aging
- Receivables Detail
- Receivables Sales Analysis
- Receivables Summary Detail

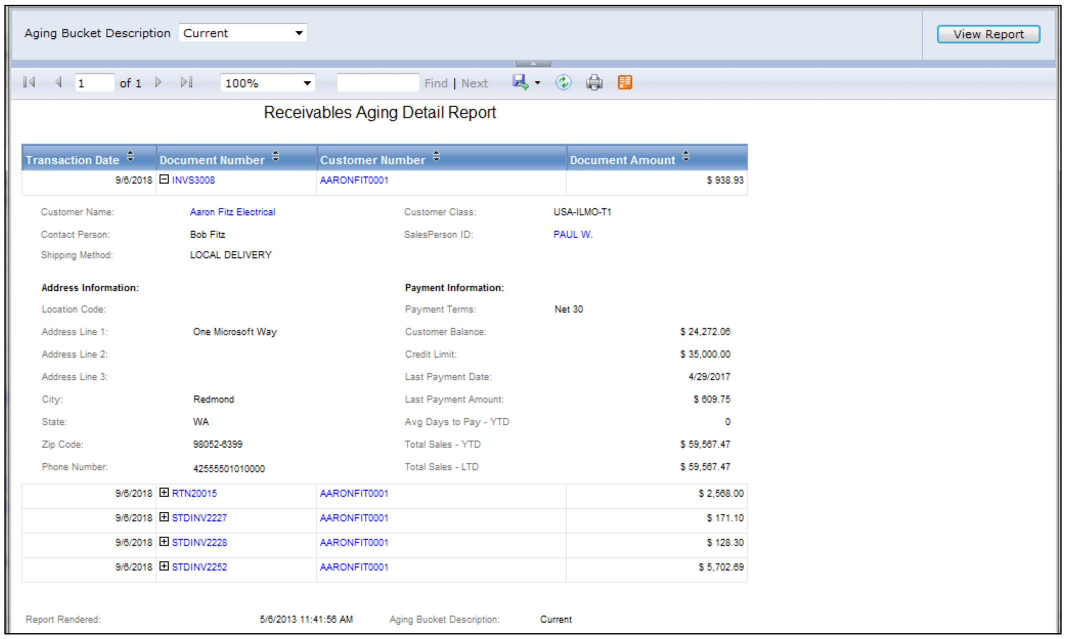
Selected Reports
- Receivables Summary
- Receivables Aging Detail

* Default Report Writer report

The report listed first will be used as the default report

4. Click on **OK** to close the window.
5. Back on the **Customer Maintenance** window, we will look up for a customer. Now when we click on the print icon, we will be provided with the list of reports that we just assigned.

6. Select the report you want to view and the report will be displayed in the Report Manager for the customer record we have opened in **Customer Maintenance**.



Aging Bucket Description: Current

View Report

1 of 1 | 100% | Find | Next

Receivables Aging Detail Report

Transaction Date	Document Number	Customer Number	Document Amount
9/6/2018	⊟ INVS3008	AARONFIT0001	$ 938.93

Customer Name:	Aaron Fitz Electrical	Customer Class:	USA-ILMO-T1
Contact Person:	Bob Fitz	SalesPerson ID:	PAUL W.
Shipping Method:	LOCAL DELIVERY		
Address Information:		**Payment Information:**	
Location Code:		Payment Terms:	Net 30
Address Line 1:	One Microsoft Way	Customer Balance:	$ 24,272.06
Address Line 2:		Credit Limit:	$ 35,000.00
Address Line 3:		Last Payment Date:	4/29/2017
City:	Redmond	Last Payment Amount:	$ 609.75
State:	WA	Avg Days to Pay - YTD	0
Zip Code:	98052-6399	Total Sales - YTD	$ 59,567.47
Phone Number:	42555501010000	Total Sales - LTD	$ 59,567.47

Transaction Date	Document Number	Customer Number	Document Amount
9/6/2018	⊞ RTN20015	AARONFIT0001	$ 2,568.00
9/6/2018	⊞ STDINV2227	AARONFIT0001	$ 171.10
9/6/2018	⊞ STDINV2228	AARONFIT0001	$ 128.30
9/6/2018	⊞ STDINV2252	AARONFIT0001	$ 5,702.69

Report Rendered: 5/6/2013 11:41:56 AM Aging Bucket Description: Current

Perhaps the most useful part about this new functionality is that the master record is passed to the report as a parameter. That means if we assign a sales-related report that contains a parameter for **Customer Number** to the **Customer Maintenance** window, the value of the selected customer will be passed to this parameter when opening the report from the drop-down menu.

Viewing charts and KPIs using Business Analyzer

With the separate Business Analyzer add-on for Microsoft Dynamics GP, users can view the charts and KPIs that are installed with the predefined SSRS reports that we have described earlier in this chapter. Business Analyzer offers users the ability to view these highly-visual presentations of information either through a separate Business Analyzer client installed on the client workstation or directly from the GP 2013 Homepage. While Business Analyzer is not a reporting tool in the sense that we can use it on its own to create a report from scratch, it is an important enough vehicle for displaying and distributing valuable reports developed in SSRS that we felt it worth being included in this chapter.

Before we can use this functionality available through Business Analyzer, however, we must first install and configure it.

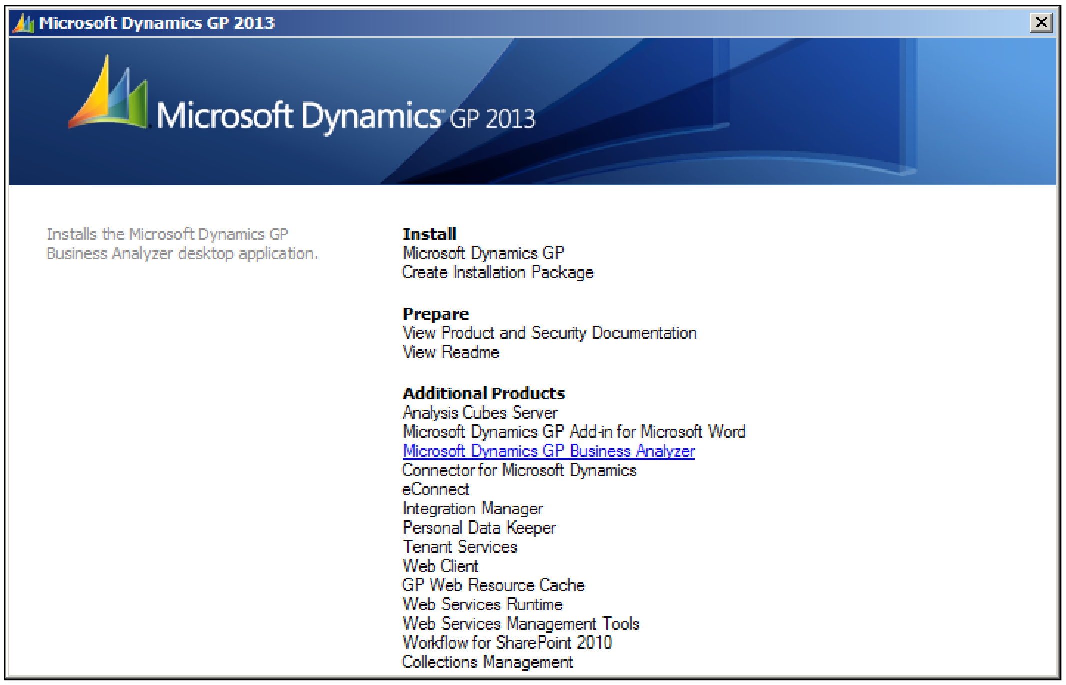
Installing and configuring Business Analyzer

Business Analyzer must be installed from the GP 2013 installation media. It must be installed on any client workstation on which this tool will be utilized. Before we begin, remember that the predefined Reporting Services reports and metrics for GP 2013 must already be deployed. Refer to earlier sections in this chapter if this has not already been completed on your environment. Additionally, security access to these reports must be configured (this topic we will be covered later in this chapter), but for the purpose of this exercise, we will assume we are using an account with administrative permissions on the SSRS site and at the GP database level.

Now, let's see what the Business Analyzer installation involves:

1. From the client workstation, open the GP 2013 installation media folder and double-click on the `setup.exe` file.

2. From the **Microsoft Dynamics GP 2013** installation splash page, click on **Microsoft Dynamics GP Business Analyzer**, and click on **Install**.



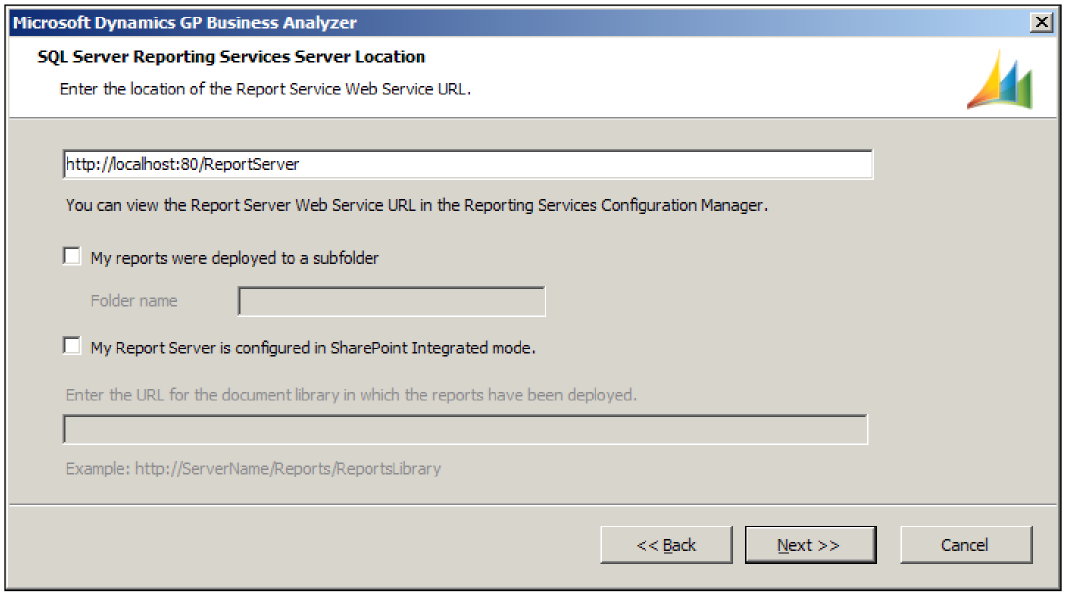
Installs the Microsoft Dynamics GP Business Analyzer desktop application.

Install
Microsoft Dynamics GP
Create Installation Package

Prepare
View Product and Security Documentation
View Readme

Additional Products
Analysis Cubes Server
Microsoft Dynamics GP Add-in for Microsoft Word
Microsoft Dynamics GP Business Analyzer
Connector for Microsoft Dynamics
eConnect
Integration Manager
Personal Data Keeper
Tenant Services
Web Client
GP Web Resource Cache
Web Services Runtime
Web Services Management Tools
Workflow for SharePoint 2010
Collections Management

3. Read and accept the terms of the License Agreement. Then, click on **Next**.
4. Enter the path to the **Report Server Web Service URL** in the first field. This is the same URL that was entered previously when installing the predefined Reporting Services reports. If you have forgotten the URL, check the SSRS Configuration Manager from the SSRS server to find this link.
5. If your SSRS instance contains predefined reports and metrics for multiple GP instances, click on the box **My reports were deployed to a subfolder** to specify which folder contains reports for the instance on which Business Analyzer is being installed. Additionally, if the report installation was completed in the **SharePoint Integrated** mode, click on the **My Report Server is configured in SharePoint Integrated mode** checkbox, and specify the URL path to the document library that contains the reports. Prior to clicking on **Next**, our screen looks as shown in the following screenshot:


Microsoft Dynamics GP Business Analyzer

SQL Server Reporting Services Server Location

Enter the location of the Report Service Web Service URL.

http://localhost:80/ReportServer

You can view the Report Server Web Service URL in the Reporting Services Configuration Manager.

☐ My reports were deployed to a subfolder

Folder name

☐ My Report Server is configured in SharePoint Integrated mode.

Enter the URL for the document library in which the reports have been deployed.

Example: http://ServerName/Reports/ReportsLibrary

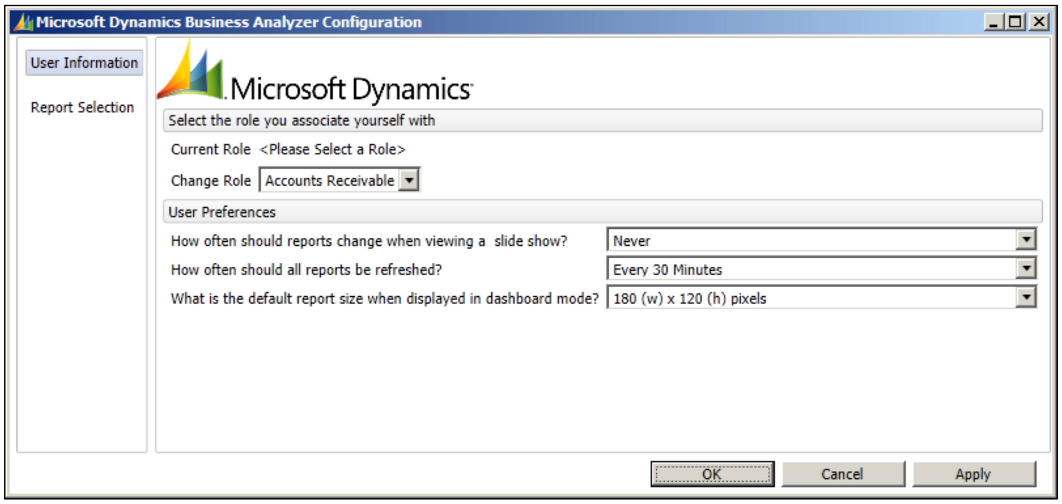
6. On the next screen, click on **Install** to begin the installation of Business Analyzer. Click on **Exit** once it has completed successfully.

Remember, this installation process must be completed on all client workstations that will utilize this tool. If your environment contains numerous workstations, consider setting up an installation package. Refer to the Business Analyzer documentation found on the installation media under **AdProd** | **Business Analyzer** for this document.

Now that we've installed Business Analyzer on this client workstation, let's configure it for use as follows:

1. From the **Start** menu, browse to the `Microsoft Dynamics` folder, expand **Business Analyzer**, and select **Microsoft Dynamics GP Business Analyzer**. Since we are opening it for the first time, we will immediately see the **Microsoft Dynamics Business Analyzer Configuration** screen with the default settings selected.
2. The first order of business should be to select the role that is most closely aligned with the user that will be using this client in the **Change Role** drop-down box. Once a default role such as **Accounts Receivable** is selected, notice that a prompt appears, offering the option of loading default reports for this role. Selecting **Yes** will automatically select the available reports related to this role and can make an easier task of the job of shifting through the myriad of charts and KPIs to find the most useful ones.

3. Several other options can be customized from this window, all of which we will leave at their default setting as follows:
    - **How often should reports change when viewing a slideshow?**: Use this drop-down box to define how quickly reports will cycle from one to the next when viewing reports in a slideshow format
    - **How often should all reports be refreshed?**: Change this setting to control how often underlying data is queried for new information to update reports
    - **Default Report Size when displayed in dashboard mode**: This setting will likely depend on the monitor resolution and size as well as user preference, so for now, we will leave this at the default setting



Select the role you associate yourself with

Current Role <Please Select a Role>

Change Role Accounts Receivable

User Preferences

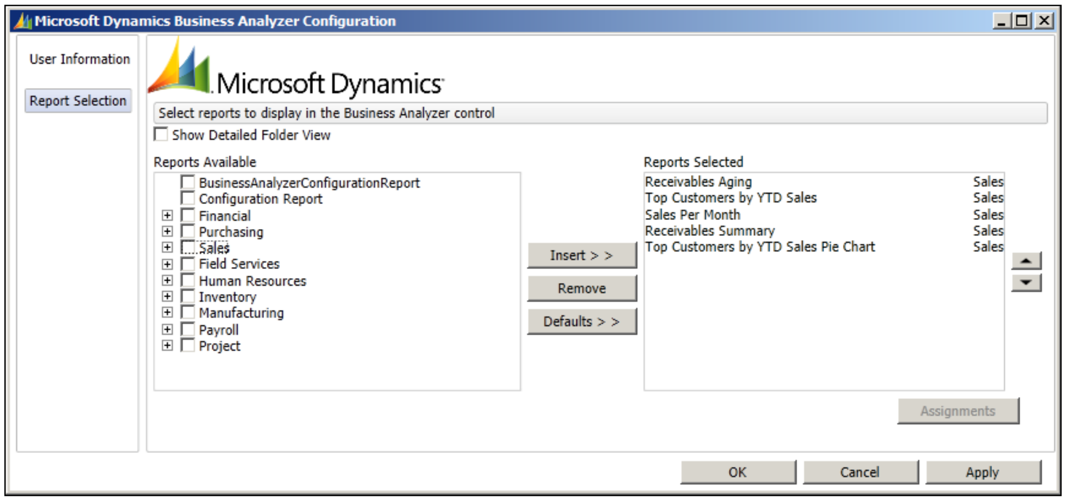
How often should reports change when viewing a slide show? Never

How often should all reports be refreshed? Every 30 Minutes

What is the default report size when displayed in dashboard mode? 180 (w) x 120 (h) pixels

4. Now, let's click on the **Report Selection** tab, so we can review the list of predefined charts and KPIs that were automatically selected for us when we changed our role. These reports appear in the right-hand pane titled **Reports Selected**. In the left-hand **Reports Available** pane, we can scroll through the list of available reports to select additional reports for viewing. For easier viewing, uncheck the **Show Detailed Folder View** box at the top of the screen.

Remember, while we can select any of the predefined reports to include in our viewer, we probably only want to select charts and KPIs, as these are tailor-made for the kind of dashboard type viewing for which we are utilizing Business Analyzer. Additionally, resist the temptation to select too many reports and assign them to the Business Analyzer. Doing so will make the Business Analyzer cumbersome and difficult to use, and that completely defeats the purpose of this tool.



Select reports to display in the Business Analyzer control

☐ Show Detailed Folder View

Reports Available
- ☐ BusinessAnalyzerConfigurationReport
- ☐ Configuration Report
- ☐ Financial
- ☐ Purchasing
- ☐ Sales
- ☐ Field Services
- ☐ Human Resources
- ☐ Inventory
- ☐ Manufacturing
- ☐ Payroll
- ☐ Project

Reports Selected	
Receivables Aging	Sales
Top Customers by YTD Sales	Sales
Sales Per Month	Sales
Receivables Summary	Sales
Top Customers by YTD Sales Pie Chart	Sales

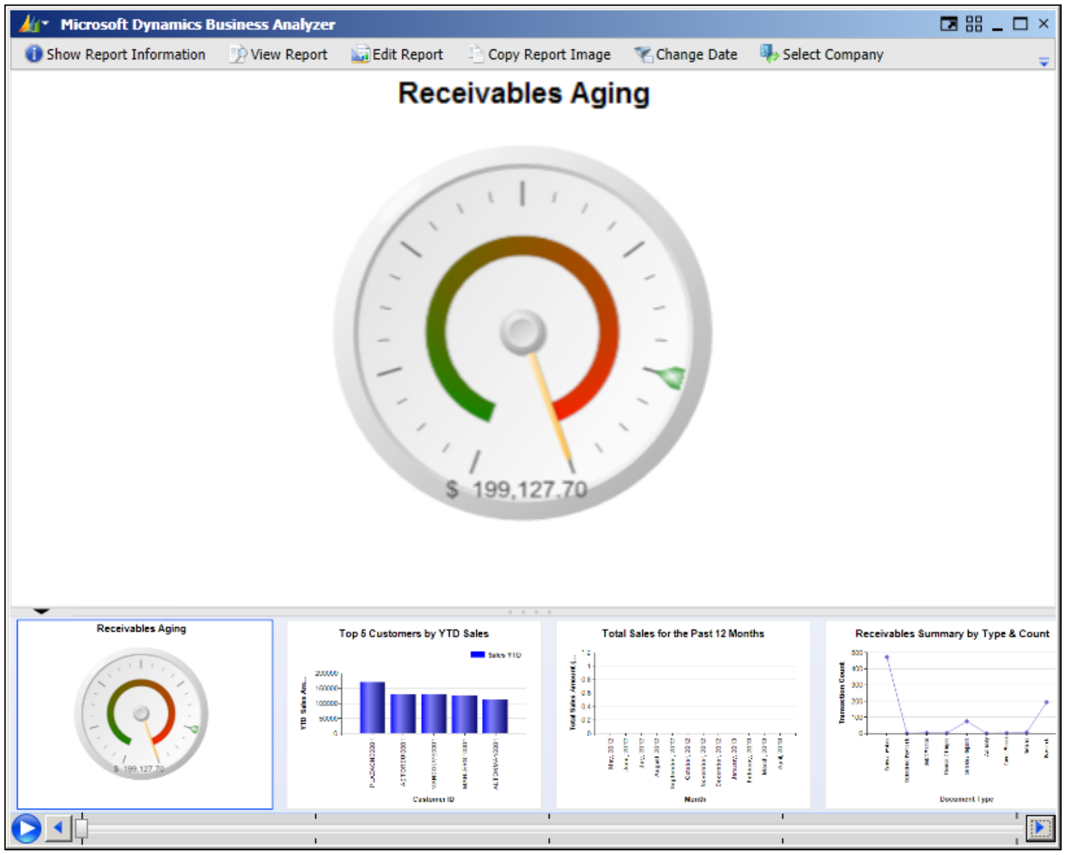
5. Once all reports have been selected, click on **Apply**, then on **OK**, to finish the configuration process.

Now that we've completed installation and configuration, the Business Analyzer application will open, and we will see our first selected report. Now, let's take a look at some ways in which we can utilize Business Analyzer to improve our ability to access and view summary level information from our GP environment.

Using the standalone Business Analyzer client

Once we've installed and configured Business Analyzer, we can utilize the Business Analyzer client found under the `Microsoft Dynamics` folder under the **Start** menu. The beauty of accessing the tool through this manner is that we do not have to log in to GP to see our GP data. This is especially convenient for users in the organization who do not need access to GP for transactional reasons, but do need the ability to access GP information in an inquiry fashion.

The Business Analyzer client opens in the dashboard/split mode, meaning the selected metric is displayed in large format at the top of the client, while a ribbon bar at the bottom of the client allows us to cycle through and select other reports we wish to view in the upper pane.


Microsoft Dynamics Business Analyzer
Show Report Information
View Report
Edit Report
Copy Report Image
Change Date
Select Company
Receivables Aging
$ 199,127.70
Receivables Aging
Top 5 Customers by YTD Sales
Total Sales for the Past 12 Months
Receivables Summary by Type & Count

Note that the lower pane can be collapsed or resized via the slim bar separating the two panes.

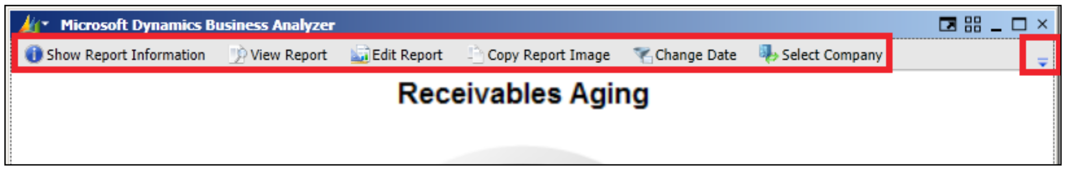
To enable the slideshow mode, click on the play icon at the bottom left-hand corner of the lower pane. This cycles through the selected reports, displaying each one in the upper pane, for the predetermined length of time previously set. For those of us with multiple monitors at our desk, we can set up the Business Analyzer on the second monitor, and let it run on its own, allowing us to keep tabs on what is happening in our organization throughout the day. Better yet, in offices grouped by functional areas (for example, an office where all A/R clerks are grouped together in a cluster of nearby desks) we can mount a computer monitor on the wall, set up the Business Analyzer on the attached computer, and set it to cycle through A/R metrics throughout the day. In addition to the dashboard/split mode we are currently viewing, we can set this to run in the Full Screen mode by right-clicking on the background of a metric and selecting the Window mode. From here, we can select the Full Screen mode to really expand the size of the selected chart/metric.

Let's take a look at some of the other functions available to us from within the Business Analyzer client.

At the top left-hand corner, notice the drop-down menu next to the GP icon. We have several handy options to choose from here as follows:

- **Filter Reports**: If reports from multiple companies or series were selected, use this option to quickly filter the list of visible reports to the selected company or series
- **Reload Reports**: Use this if you need to quickly refresh all reports based on the latest information
- **Options**: Selecting this option will return users to the **Business Analyzer Configuration** window to change or modify settings made right after the installation

Just below the blue menu bar at the top of the client, users will notice a series of action labels. By default, the labels on this bar will consist of icons only. To change this, click on the drop-down box at the far right of the panel (see the following screenshot) and select **Show Action Labels**. This will give us text along with the images and will assist us as we are finding our way around the client for the first time.



Now that we can see what each button on the action labels pane represents, let's go through them:

- **Show Report Information**: Click on this button to see when the selected metric was last refreshed and to see what company information is represented by the metric.
- **View Report**: Use this option to open the corresponding metric in the SSRS site via a web browser.
- **Edit Report**: This option opens the Report Builder application and allows us to make changes to the selected metric. This is generally not recommended except for those who have report design experience, as making changes to shared reports will affect all users who view this report. We will cover more on using Report Builder to modify reports in a later section of this chapter.
- **Copy Report Image**: This cool feature takes a snapshot of the selected chart/metric and copies it to the computer's clipboard. From here, the image can be pasted just like any other image, making it easy to send a snapshot of a report to a colleague via e-mail, paste into a PowerPoint presentation, and much more.
- **Change Date**: If the selected metric has a date parameter, this button can be used to change the date parameter and the data rendered by the report.
- **Select Company**: In multi-company environments, each predefined metric will show data for one company at a time. Use this option to change the company displayed by the selected metric.

Clearly, the standalone Business Analyzer client offers more than what meets the eye. We can use it as a rich visual tool for cycling through charts and metrics that provide us with an overview of our organization's performance. We can also use it as a launching pad to additional reports on the SSRS site that provides even more detailed information.

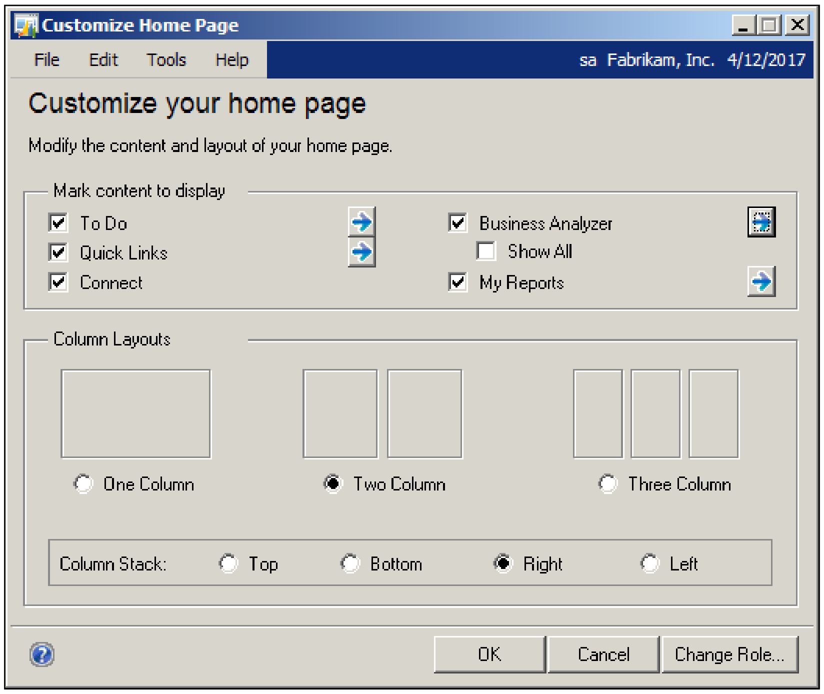
Now that we've seen what the standalone client can do, let's look at using the Business Analyzer from within the Dynamics GP application.

Using Business Analyzer from within the Dynamics GP client

In Version GP 2010, the Dynamics GP Homepage included a section named **Metrics** that could be used to expose the predefined charts and KPIs from within the GP application. While this functionality still exists in GP 2013, it has a new look and a new name courtesy of the Business Analyzer tool.

Having installed the predefined charts and KPIs earlier in this chapter, and just having completed the Business Analyzer installation and configuration, let's look at the steps involved in setting this up within GP:

1. Log in to Microsoft Dynamics GP 2013.
2. From the GP 2013 Homepage, select **Customize this page** in the upper right-hand corner to open the **Customize Home Page** window.
3. Check the box for **Business Analyzer**.

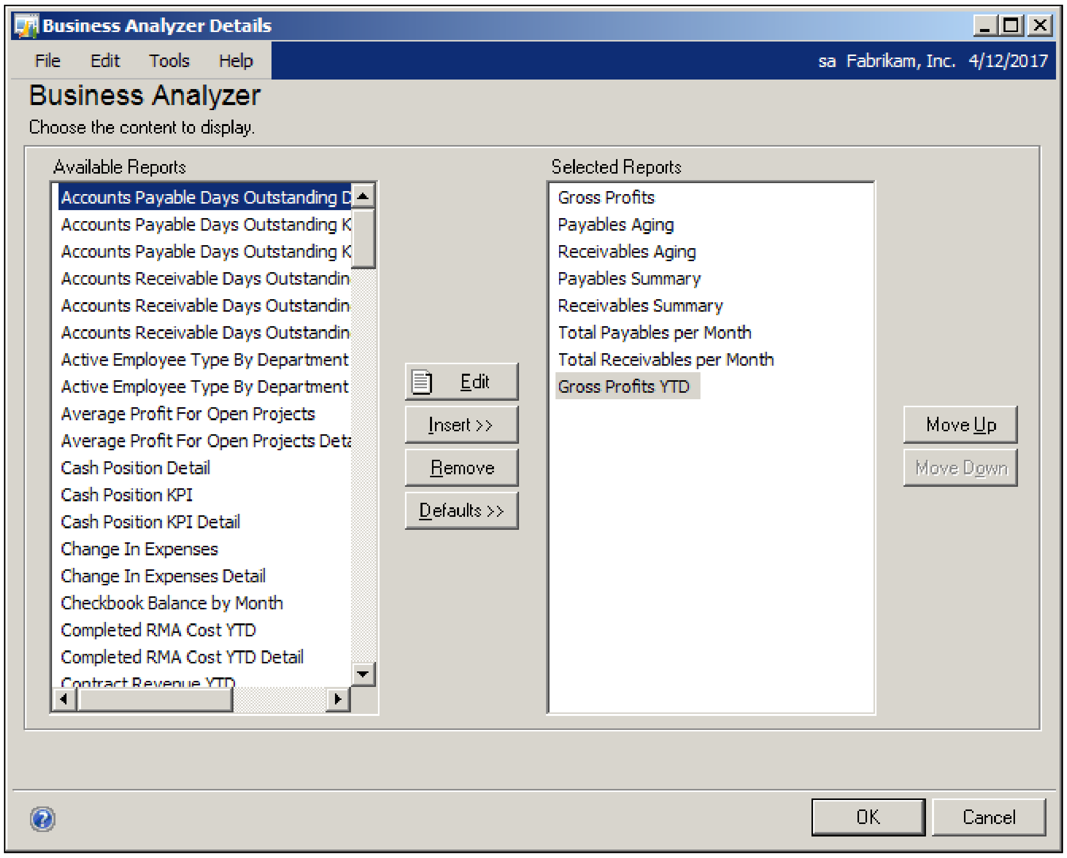


Mark content to display:
[✓] To Do
[✓] Quick Links
[✓] Connect
[✓] Business Analyzer
[ ] Show All
[✓] My Reports

Column Layouts: One Column [✓ Two Column] Three Column

Column Stack: Top Bottom [✓ Right] Left

4. Click on the arrow next to **Business Analyzer** to open the **Business Analyzer Details** window. From here, we can select charts and KPIs in the list on the left-hand side, and click on the **Insert >>** button to move them into the right-hand pane, so they will be visible from the Business Analyzer section of our home page. We can have a completely separate set of reports visible from within GP than from within the standalone Business Analyzer client.
5. Additionally, note the presence of the **Defaults >>** button on this window. Clicking on this button will override our existing charts and KPIs with the predefined list of charts and KPIs for the role that we selected when setting up our user for the first time in GP. It might be a good idea to accept the default charts and KPIs and then pick and choose additional reports to add to the defaults.



Business Analyzer

Choose the content to display.

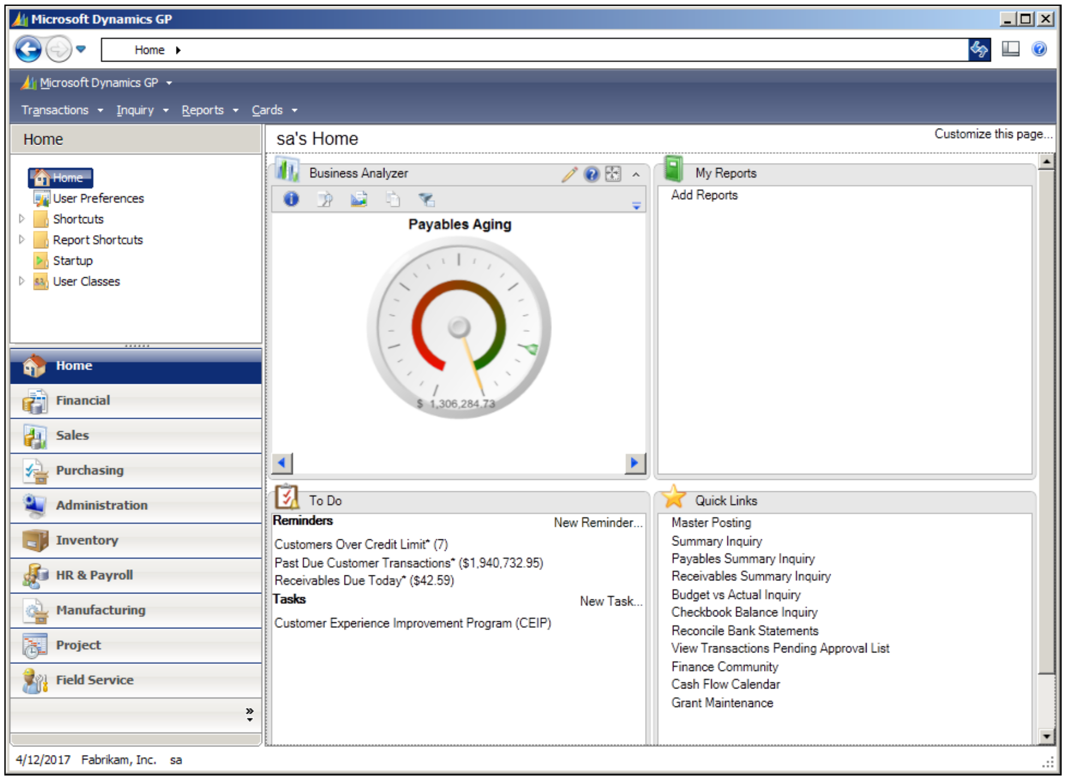
Available Reports

Accounts Payable Days Outstanding D
Accounts Payable Days Outstanding K
Accounts Payable Days Outstanding K
Accounts Receivable Days Outstandin
Accounts Receivable Days Outstandin
Accounts Receivable Days Outstandin
Active Employee Type By Department
Active Employee Type By Department
Average Profit For Open Projects
Average Profit For Open Projects Deta
Cash Position Detail
Cash Position KPI
Cash Position KPI Detail
Change In Expenses
Change In Expenses Detail
Checkbook Balance by Month
Completed RMA Cost YTD
Completed RMA Cost YTD Detail
Contract Revenue YTD

Selected Reports

Gross Profits
Payables Aging
Receivables Aging
Payables Summary
Receivables Summary
Total Payables per Month
Total Receivables per Month
Gross Profits YTD

6. Click on **OK** twice to return to the GP Homepage.
7. We will now notice a new section on our GP Homepage entitled **Business Analyzer**. In order to see the actual charts and KPIs we selected, we need to refresh the Homepage. This can be accomplished by clicking on the Refresh icon at the top of the screen between the address bar and the Layout icon.
8. Now that we can see charts and KPIs in our **Business Analyzer** section, we have options for moving the section, setting it to a larger size, or hovering over the section to reveal the auto-collapsing action labels pane. The action labels pane offers many of the same functions as we saw in the standalone client.



Customers Over Credit Limit* (7)
Past Due Customer Transactions* ($1,940,732.95)
Receivables Due Today* ($42.59)

Customer Experience Improvement Program (CEIP)

Master Posting
Summary Inquiry
Payables Summary Inquiry
Receivables Summary Inquiry
Budget vs Actual Inquiry
Checkbook Balance Inquiry
Reconcile Bank Statements
View Transactions Pending Approval List
Finance Community
Cash Flow Calendar
Grant Maintenance

Congratulations, Business Analyzer is now set up and running in your GP environment! Enjoy making the most of having data in a rich dashboard-like presentation at your fingertips!

Configuring security for Reporting Services

Once we've installed the predefined reports metrics, we need to define which users will have access to these objects. Security for SSRS reports is two-fold: first, we must grant access to the actual reports in the Report Manager, and then, we must grant the database access to the report so that it can actually retrieve data from the database.

Note that after the initial deployment, only users who are members of the web server's local administrator group and the database server's local administrator group will be able to view and generate reports from the Report Manager. All other users must wait for the security policy to be modified so that they can also view and generate data for the reports.

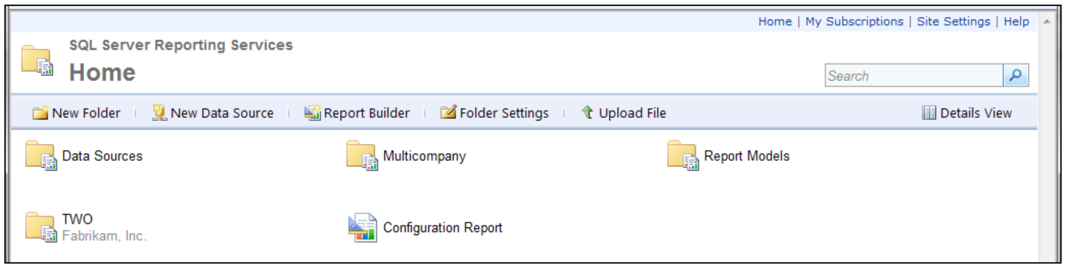
Assigning access to the Reporting Services website

The first step to setting up security for the predefined SSRS reports is to set security for the Reporting Services website. This is done by granting access to Windows **Group or User** to **Role(s)** that have been set up in the Report Manager. Although individual Windows users can be granted access to these SSRS roles, we recommend setting up Windows groups, if possible.

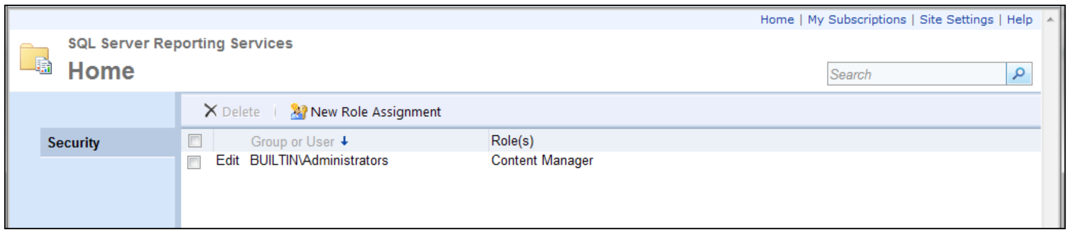
The process of granting access to Reporting Services can be done within the Report Manager or SQL Server Management Studio while using an account with administrative privileges on the server. In this example, we'll use the Report Manager to grant access to the Reporting Services website. In this example, we have set up a Windows group called `GP Users`. To this group, we have added users who will need access to the Reporting Services website in order to view the predefined reports that we just deployed.

1. In this example, in which we assume a **Native** mode installation, connect to the Report Manager home page. By default, this URL is `http://<ComputerName>/reports`.

2. Click on the **Folder Settings** button in the following screenshot:



3. The **Security** tab will be selected and your view should look similar to the following screenshot:

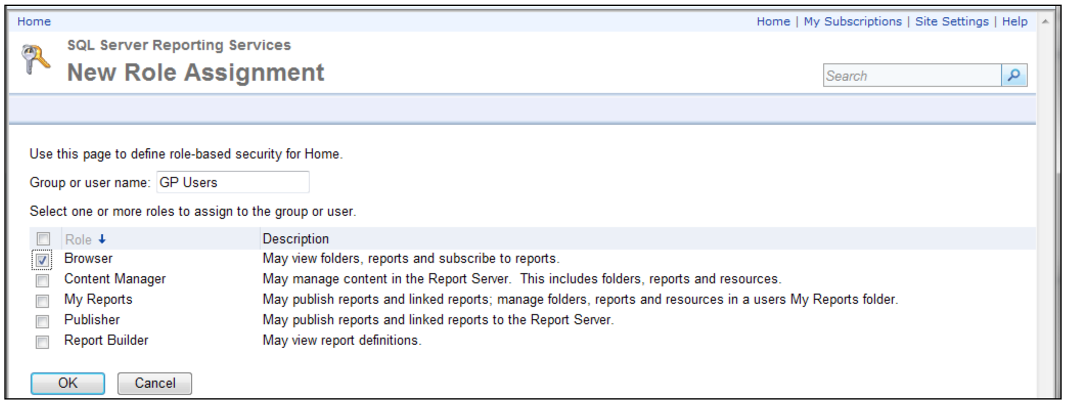

Home | My Subscriptions | Site Settings | Help
SQL Server Reporting Services
Home
Search
Security
Delete
New Role Assignment

	Group or User	Role(s)
Edit	BUILTIN\Administrators	Content Manager

The **BUILTIN\Administrators** group should already be assigned the role of **Content Manager** for the Home level. SSRS security operates via inherited permissions, meaning that if we assign a group or user to an item-level role at the Home level, the permissions granted by that role will be extended to all subfolders of the **Home** page, including the `TWO` and `Data Sources` folders.

4. Click on **New Role Assignment** to open a list of the item-level roles that have been created in SSRS.
5. In the **Group or user name** field, we'll enter `GP Users` as the name of the group being granted permissions.

6. Select the predefined **Browser** role to allow users in the **GP Users** group to view folders, reports, and subscribe to reports in the **Home** page. To see a list of all tasks assigned to the **Browser** role (or any other role, for that matter), click on the role name. This will open the **Edit Role** window and allow us to see the tasks assigned to this role. Click on the browser's **Back** button to return to the **New Role Assignment** window seen in the following screenshot:


Home
Home | My Subscriptions | Site Settings | Help
SQL Server Reporting Services
New Role Assignment
Search

Use this page to define role-based security for Home.

Group or user name: GP Users

Select one or more roles to assign to the group or user.

	Role	Description
✓	Browser	May view folders, reports and subscribe to reports.
	Content Manager	May manage content in the Report Server. This includes folders, reports and resources.
	My Reports	May publish reports and linked reports; manage folders, reports and resources in a users My Reports folder.
	Publisher	May publish reports and linked reports to the Report Server.
	Report Builder	May view report definitions.

7. After confirming that the **Browser** role is selected for the **GP Users** group, click on **OK** to commit this change. If an error message appears, often it means that the group or user entered in the previous window was either misspelled or has not been set up in Windows.
8. Otherwise, we will be returned to the **Folder Settings** page of the Home level to see that our group has been assigned to the **Browser** role. The permissions granted by this role have now been extended to the folders below the Home level. We can confirm this by clicking into any of these folders, and navigating to **Folder Settings | Security**.

In this example, we have used one of the predefined item-level roles to grant read-only report permissions to a Windows group. Additional roles can be set up if our company's security requirements dictate stricter security measurements. Also, in multi-company databases, a company may require that users in one group have access to reports for a single company database. This would require manipulating the inherited permissions policy that we discussed earlier in our example. To handle these specific security needs, we strongly encourage readers to refer to *SQL Server Books Online* or other SSRS material for information on how to properly set up the Reporting Services security.

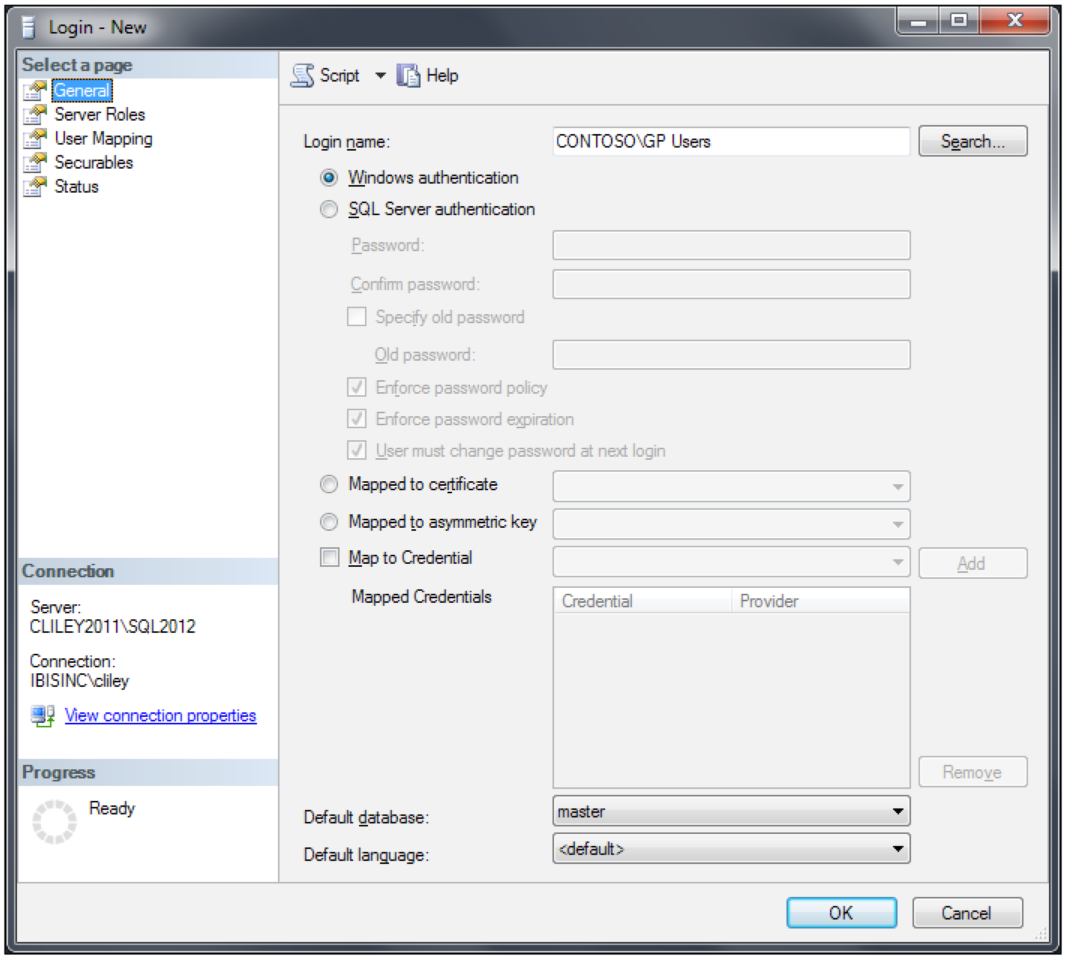
Setting up database security for predefined SSRS reports

By default, the data sources that were deployed with the predefined SSRS reports are set up to allow connections to the database via Windows integrated security. In other words, when generating an SSRS report referencing data in a GP database, the current Windows user account will be authenticated against the database. If that individual user, or a group to which that user has been assigned, has been granted permissions to the appropriate roles in the `GP` and `DYNAMICS` databases, then the report will return data from that database.

In the following steps, we will create a SQL login for our newly created **GP Users** group. Then, we will map that SQL login to several database roles that have been set up to control the database access for SSRS users as follows:

1. Open SQL Management Studio.
2. In the **Connect to Server** window, select **Database Engine** as the server type.
3. In the **Server Name** field, enter the name of the server that contains the GP databases.
4. In the **Authentication** field, select either **Windows Authentication** or **SQL Server Authentication**. The selected user should have administrative rights on the server.
5. Click on **Connect**.
6. Expand the **Security** node.
7. Right-click on the **Logins** node, and click on **New Login** to open the **Login – New** window.
8. In the **Login name** field, enter the name of the Windows **Group or User** being granted the database access in the form of `<domain>\<username>` or `<domain>\<groupname>`. Click on the **Search** button to look up the group or user from a list.

9. On the **Login – New** window, confirm that **Windows Authentication** is selected.



Login name: CONTOSO\GP Users

[✓ Windows authentication] SQL Server authentication

Server: CLILEY2011\SQL2012

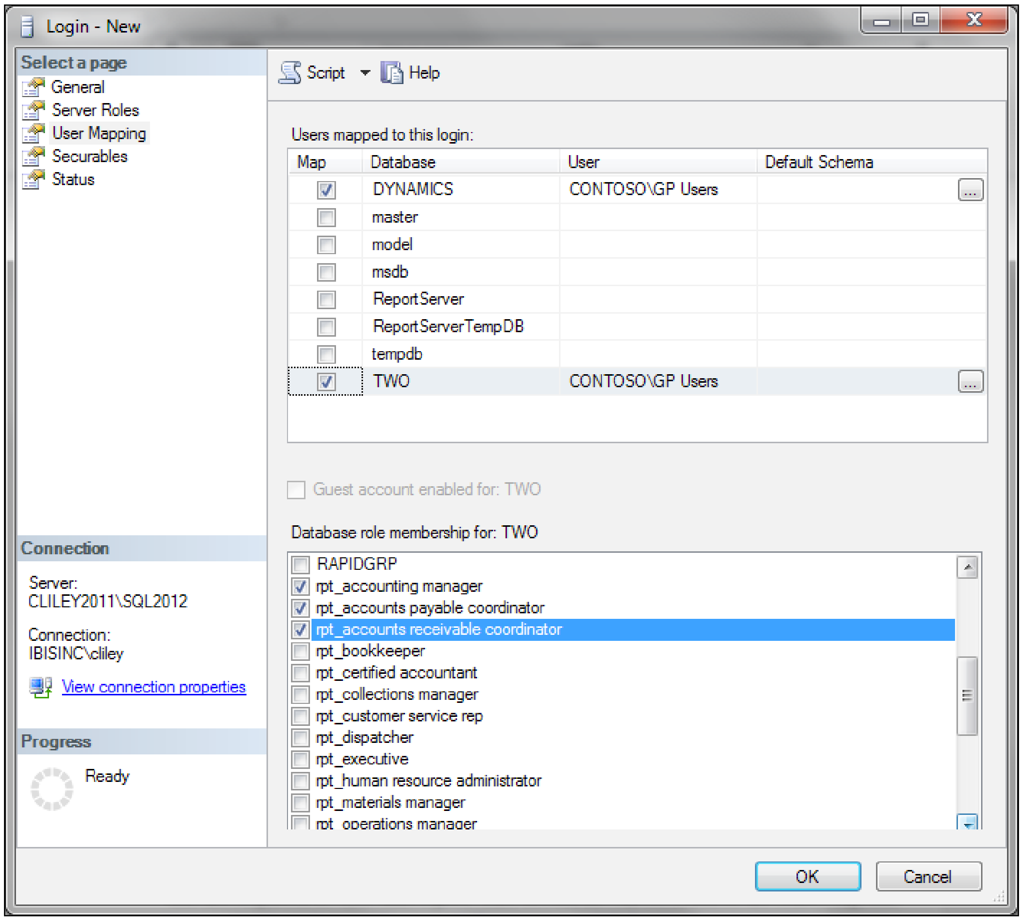
Connection: IBISINC\cliley

Default database: master

Default language: <default>

10. Click on the **User Mapping** page from the left-hand pane.
11. Click on the checkbox next to the **DYNAMICS** database.
12. In the **Database Role Membership** pane at the bottom of the window, click on the checkbox for the **rpt_all user** role.
13. Back in the database pane, click on the checkbox next to the GP database for which SSRS reports were deployed.

14. In the **Database Role Membership** pane at the bottom of the window, click on the checkbox for the appropriate roles preceded by `rpt_`. Refer to the link at the end of this section for more information on how to determine which roles are necessary for the group or user.



Login - New

Select a page
- General
- Server Roles
- User Mapping
- Securables
- Status

Script ▾ Help

Users mapped to this login:

Map	Database	User	Default Schema
✓	DYNAMICS	CONTOSO\GP Users	
	master		
	model		
	msdb		
	ReportServer		
	ReportServerTempDB		
	tempdb		
✓	TWO	CONTOSO\GP Users	

Guest account enabled for: TWO

Database role membership for: TWO

- [ ] RAPIDGRP
- [✓] rpt_accounting manager
- [✓] rpt_accounts payable coordinator
- [✓] rpt_accounts receivable coordinator
- [ ] rpt_bookkeeper
- [ ] rpt_certified accountant
- [ ] rpt_collections manager
- [ ] rpt_customer service rep
- [ ] rpt_dispatcher
- [ ] rpt_executive
- [ ] rpt_human resource administrator
- [ ] rpt_materials manager
- [ ] rpt_operations manager

Connection

Server:
CLILEY2011\SQL2012

Connection:
IBISINC\cliley

View connection properties

Progress

Ready

OK Cancel

15. Repeat the mapping process for any additional GP company databases.
16. Click on **OK** to add the new login to the SQL server database with the user mappings that we have just defined.

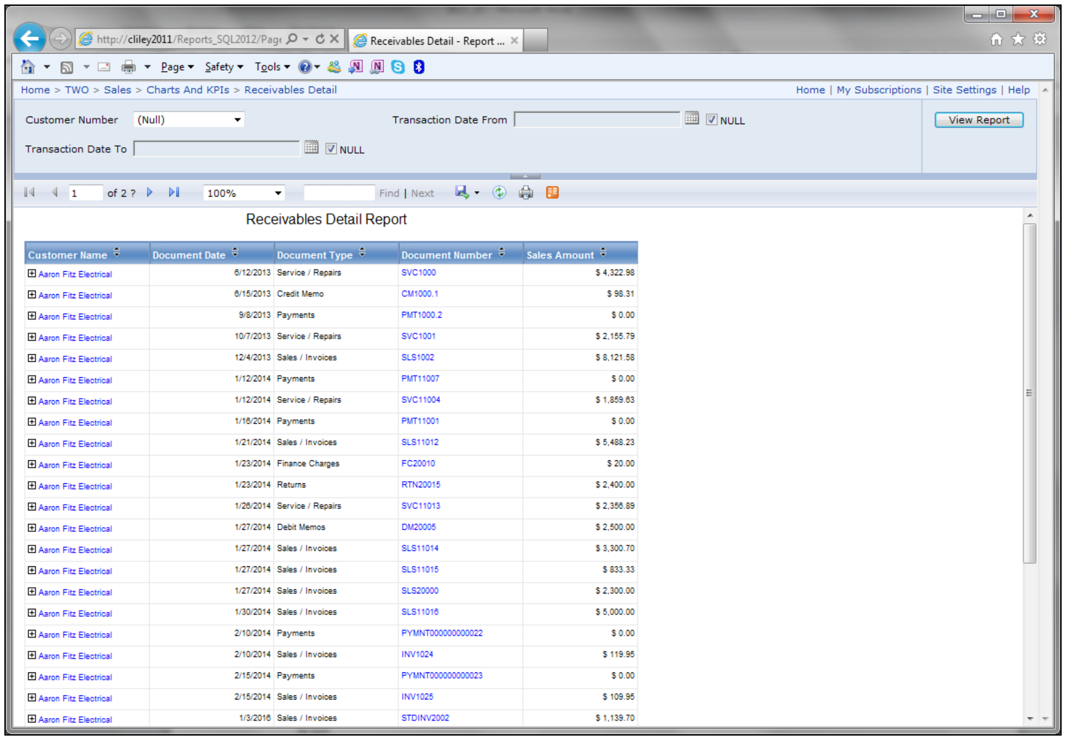
If the predefined Excel Reports described in *Chapter 3, Working with the Builders – SmartList and Excel Reports*, have also been deployed to this server, the `rpt_ database` roles will have already been created on the server. These database roles will be used for both the predefined Excel Reports and the predefined SSRS reports.

For those of us with a CustomerSource or PartnerSource login, a handy document can be downloaded that details each of the SSRS reports and the corresponding database roles that are required to view that report. This document can be found at the following link:

```
https://mbs.microsoft.com/fileexchange/?fileID=80628e60-729d-4e47-
bbd5-37af74ea39c7
```

Modifying default reports with Visual Studio

Now that we have learned how to deploy the predefined reports, how to access them in Dynamics GP 2013, and how to assign security to them, we will now take a look at modifying these reports with Visual Studio to address our particular organization's needs. We will look at modifying the **Receivables Detail Report** (see the following screenshot) to allow us to drill back to the document in GP 2013.


http://cliley2011/Reports_SQL2012/Page
Receivables Detail - Report ...
Page
Safety
Tools
Home > TWO > Sales > Charts And KPIs > Receivables Detail
Home | My Subscriptions | Site Settings | Help
View Report
1 of 2 ?
100%
Find | Next

Customer Number (Null)
Transaction Date From [✓ NULL]
Transaction Date To [✓ NULL]

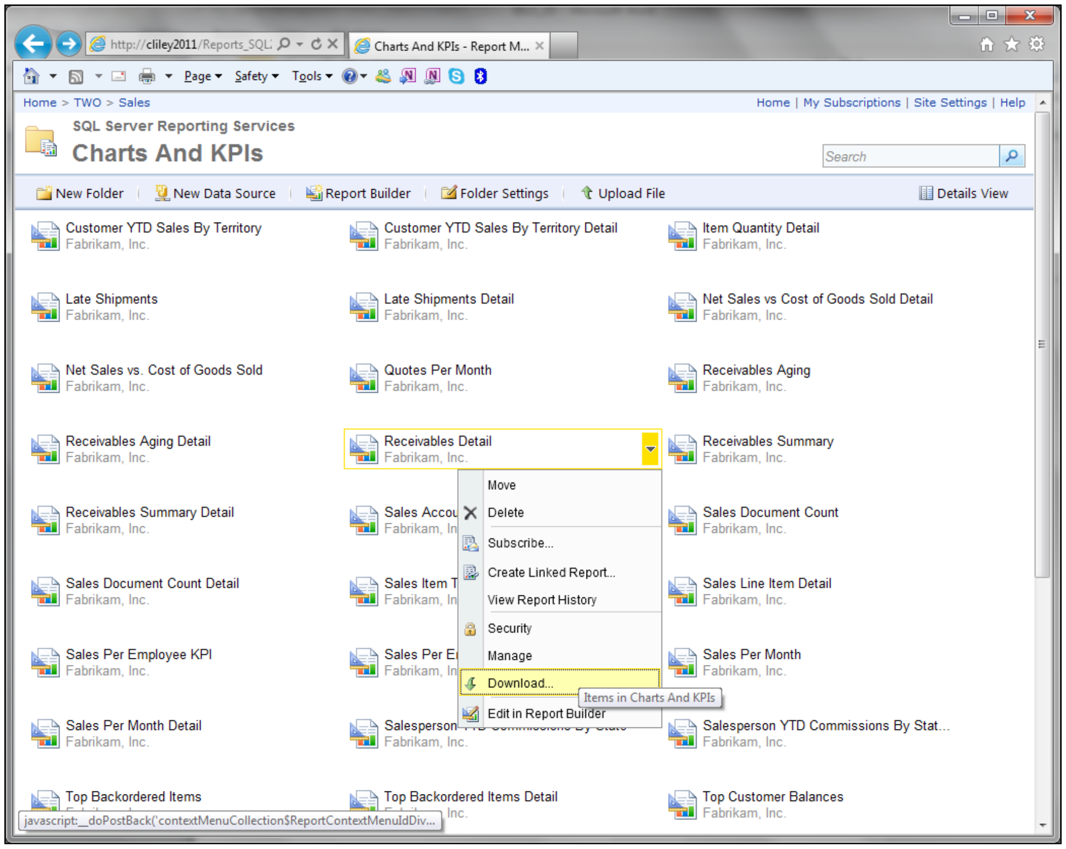
Receivables Detail Report

Customer Name	Document Date	Document Type	Document Number	Sales Amount
Aaron Fitz Electrical	6/12/2013	Service / Repairs	SVC1000	$ 4,322.98
Aaron Fitz Electrical	6/15/2013	Credit Memo	CM1000.1	$ 98.31
Aaron Fitz Electrical	9/8/2013	Payments	PMT1000.2	$ 0.00
Aaron Fitz Electrical	10/7/2013	Service / Repairs	SVC1001	$ 2,155.79
Aaron Fitz Electrical	12/4/2013	Sales / Invoices	SLS1002	$ 8,121.58
Aaron Fitz Electrical	1/12/2014	Payments	PMT11007	$ 0.00
Aaron Fitz Electrical	1/12/2014	Service / Repairs	SVC11004	$ 1,859.63
Aaron Fitz Electrical	1/16/2014	Payments	PMT11001	$ 0.00
Aaron Fitz Electrical	1/21/2014	Sales / Invoices	SLS11012	$ 5,488.23
Aaron Fitz Electrical	1/23/2014	Finance Charges	FC20010	$ 20.00
Aaron Fitz Electrical	1/23/2014	Returns	RTN20015	$ 2,400.00
Aaron Fitz Electrical	1/26/2014	Service / Repairs	SVC11013	$ 2,356.89
Aaron Fitz Electrical	1/27/2014	Debit Memos	DM20005	$ 2,500.00
Aaron Fitz Electrical	1/27/2014	Sales / Invoices	SLS11014	$ 3,300.70
Aaron Fitz Electrical	1/27/2014	Sales / Invoices	SLS11015	$ 833.33
Aaron Fitz Electrical	1/27/2014	Sales / Invoices	SLS20000	$ 2,300.00
Aaron Fitz Electrical	1/30/2014	Sales / Invoices	SLS11016	$ 5,000.00
Aaron Fitz Electrical	2/10/2014	Payments	PYMNT000000000022	$ 0.00
Aaron Fitz Electrical	2/10/2014	Sales / Invoices	INV1024	$ 119.95
Aaron Fitz Electrical	2/15/2014	Payments	PYMNT000000000023	$ 0.00
Aaron Fitz Electrical	2/15/2014	Sales / Invoices	INV1025	$ 109.95
Aaron Fitz Electrical	1/3/2016	Sales / Invoices	STDINV2002	$ 1,139.70

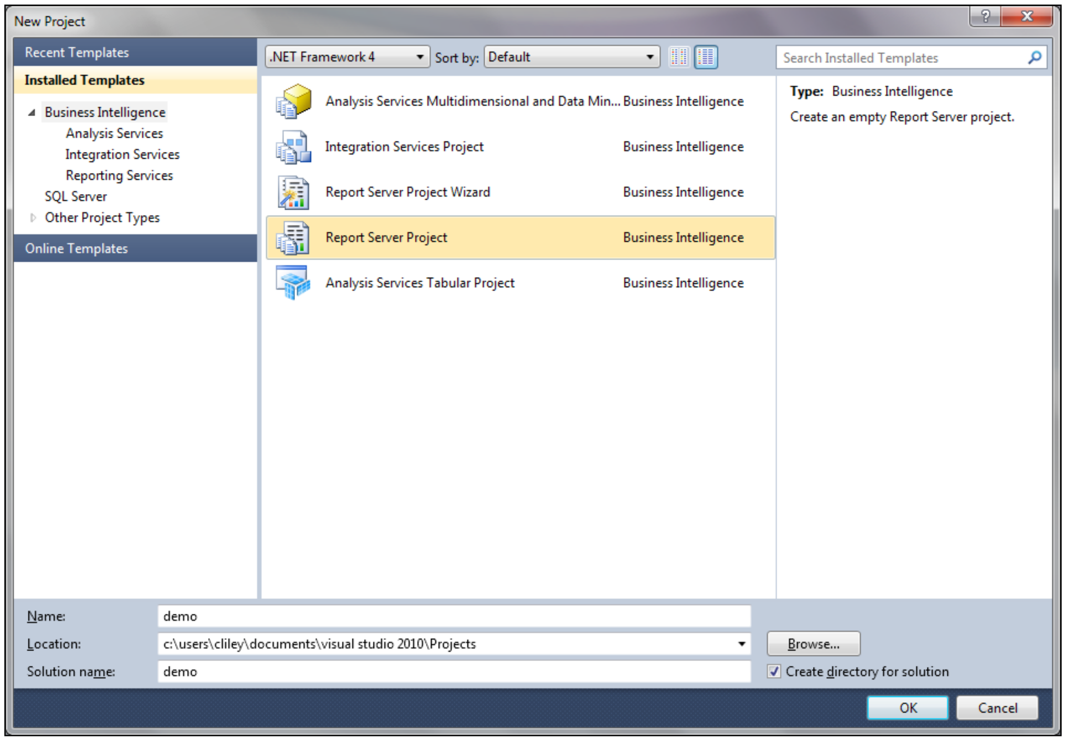
In order to begin, we need to get the report into Visual Studio so that we can work with it. There is a Visual Studio solution file provided with Dynamics GP 2013 that can be opened in Visual Studio and changes can be made there and redeployed to the report server. Another way, which we will discuss here, is to copy down the report file from the report server and open that file in Visual Studio. One advantage of doing it this way is that we can be sure we are using the most recent copy that is in use.

Let's go ahead and take a look at the steps required to copy and make changes to our existing report file:

1. Launch the Report Manager by opening a web browser, and type the **Report Manager URL** into the address bar.
2. Browse the folders on the Report Manager to the location of the **Receivables Detail**, and click on the dropdown on the report.
3. Click on the **Download** option.



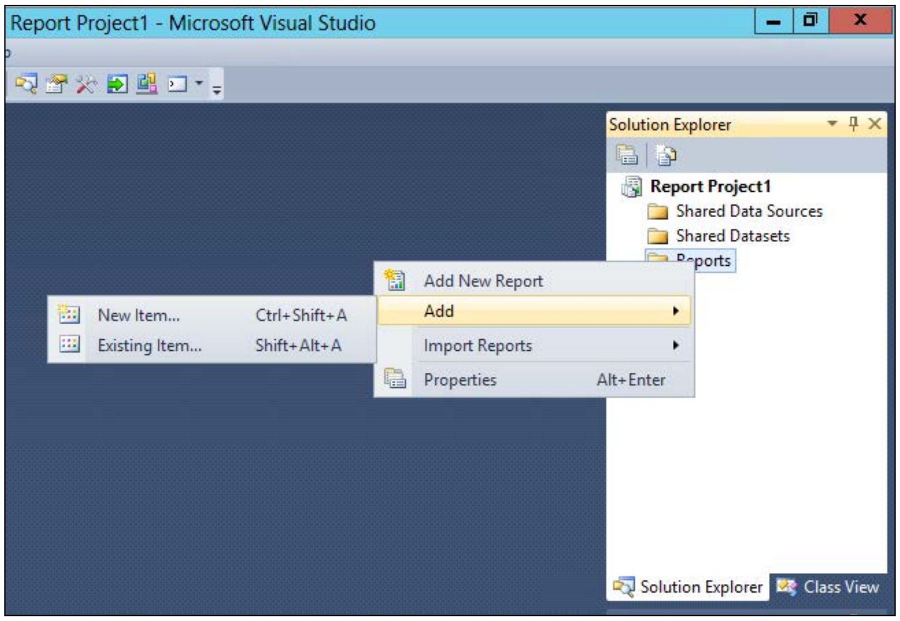
4. Save the `Receivables Detail.rdl` file to a local folder or a file share.
5. Launch Visual Studio by navigating to **All Programs** | **Microsoft Visual Studio 2010** | **Microsoft Visual Studio 2010**.
6. Create a new project by navigating to **File** | **New** | **Project from the menu bar**.


New Project
Recent Templates
Installed Templates
Business Intelligence
Analysis Services
Integration Services
Reporting Services
SQL Server
Other Project Types
Online Templates
.NET Framework 4
Sort by: Default
Search Installed Templates
Analysis Services Multidimensional and Data Min... Business Intelligence
Integration Services Project Business Intelligence
Report Server Project Wizard Business Intelligence
Report Server Project Business Intelligence
Analysis Services Tabular Project Business Intelligence
Browse...
OK
Cancel

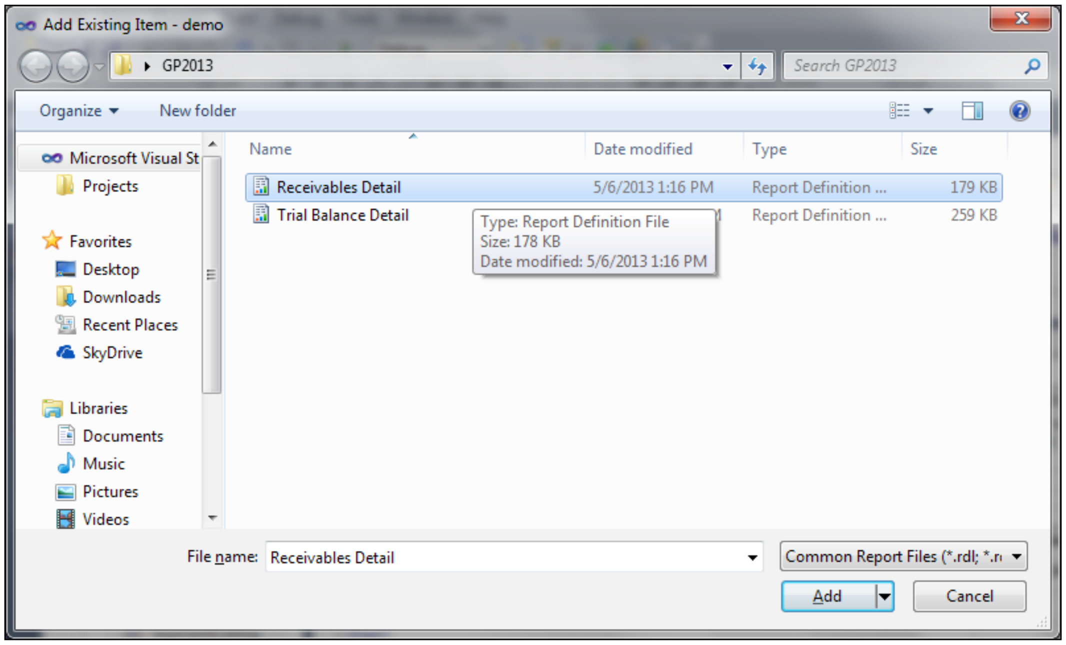
Type: Business Intelligence
Create an empty Report Server project.

Name: demo
Location: c:\users\cliley\documents\visual studio 2010\Projects
Solution name: demo
[✓] Create directory for solution

7. As seen in the previous screenshot of the **New Project** Window, select **Report Server Project**, enter `Demo` for the name of the project, browse to a location to store the project, and enter `Demo` as the solution name.
8. Click on **OK**.
9. Once the project is created, in the right-hand pane will be **Solution Explorer**. We want to add our existing `Receivables Detail.rdl` file to our project.
10. In **Solution Explorer**, right-click on **Reports**, then navigate to **Add** | **Existing Item**.



11. Browse to the location where we stored the file, and click on **Add** to open the **Add Existing Items** window as seen in the following screenshot:



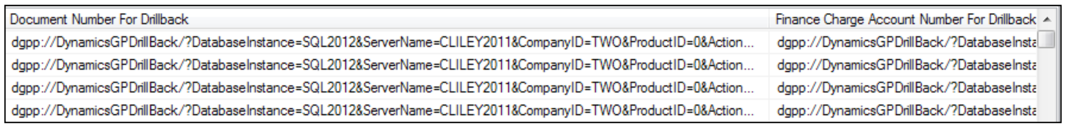
Name	Date modified	Type	Size
Receivables Detail	5/6/2013 1:16 PM	Report Definition ...	179 KB
Trial Balance Detail		Report Definition ...	259 KB

Type: Report Definition File
Size: 178 KB
Date modified: 5/6/2013 1:16 PM

File name: Receivables Detail
Common Report Files (*.rdl; *.r

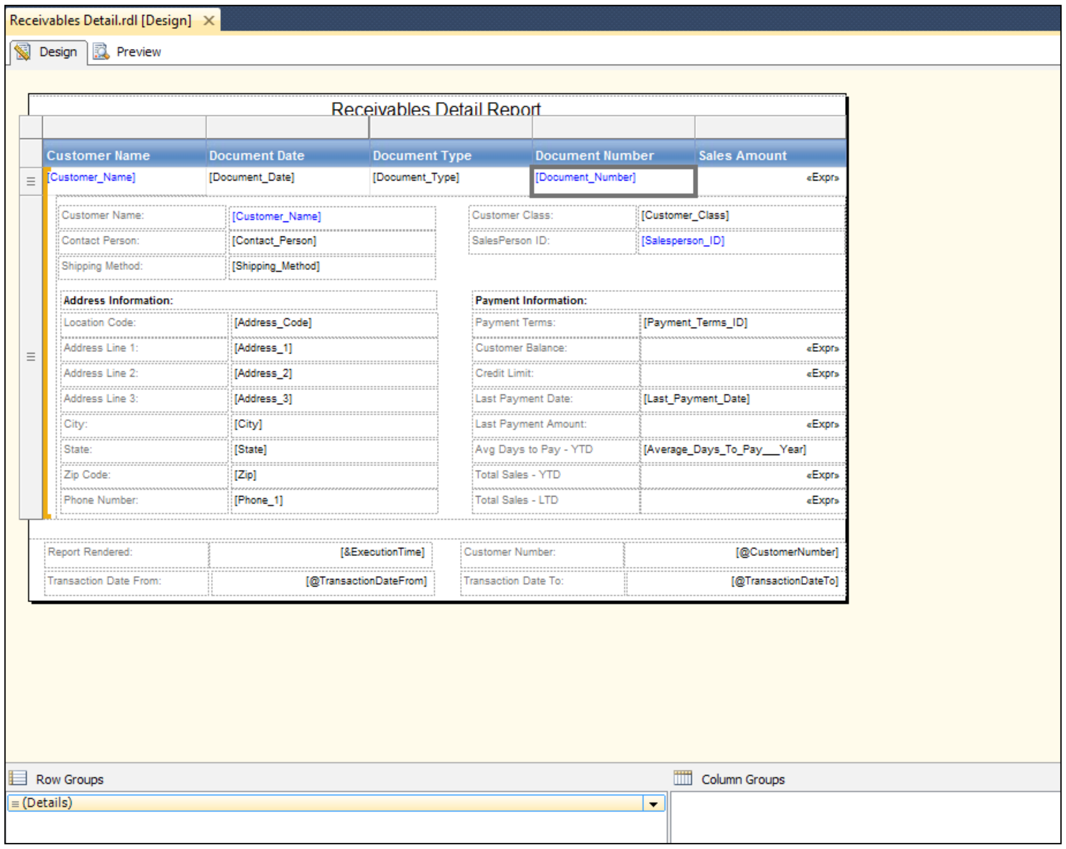
12. In **Solution Explorer**, `Receivables Detail.rdl` will now show up as a report. Double-click on it to open the report in the **Design** mode.

13. Before we are able to add-in this drillback functionality to our report, we need to have the drillback URL included in our dataset. These drillback URLs are located at the end of the field list in the views that SmartList is based on. In our example, we are using the Receivables Transactions view. The drillback fields should look like the following screenshot:



Document Number For Drillback	Finance Charge Account Number For Drillback
dgpp://DynamicsGPDrillBack/?DatabaseInstance=SQL2012&ServerName=CLILEY2011&CompanyID=TWO&ProductID=0&Action...	dgpp://DynamicsGPDrillBack/?DatabaseInsta
dgpp://DynamicsGPDrillBack/?DatabaseInstance=SQL2012&ServerName=CLILEY2011&CompanyID=TWO&ProductID=0&Action...	dgpp://DynamicsGPDrillBack/?DatabaseInsta
dgpp://DynamicsGPDrillBack/?DatabaseInstance=SQL2012&ServerName=CLILEY2011&CompanyID=TWO&ProductID=0&Action...	dgpp://DynamicsGPDrillBack/?DatabaseInsta
dgpp://DynamicsGPDrillBack/?DatabaseInstance=SQL2012&ServerName=CLILEY2011&CompanyID=TWO&ProductID=0&Action...	dgpp://DynamicsGPDrillBack/?DatabaseInsta

14. Once we have added the drillback column to our dataset, we can proceed. In the **Design** layout, we will select our **Document Number** field (see the following screenshot).



Receivables Detail Report

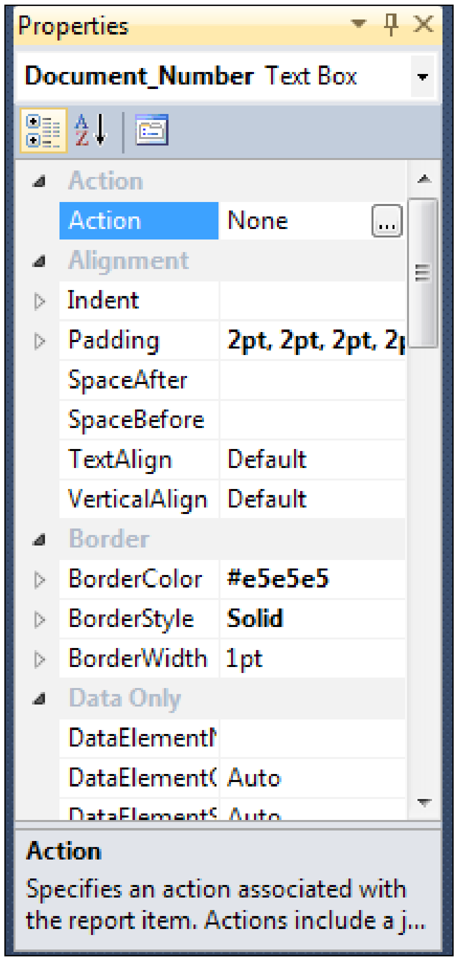
Customer Name	Document Date	Document Type	Document Number	Sales Amount
[Customer_Name]	[Document_Date]	[Document_Type]	[Document_Number]	«Expr»

Customer Name:	[Customer_Name]	Customer Class:	[Customer_Class]
Contact Person:	[Contact_Person]	SalesPerson ID:	[Salesperson_ID]
Shipping Method:	[Shipping_Method]		

Address Information:		**Payment Information:**	
Location Code:	[Address_Code]	Payment Terms:	[Payment_Terms_ID]
Address Line 1:	[Address_1]	Customer Balance:	«Expr»
Address Line 2:	[Address_2]	Credit Limit:	«Expr»
Address Line 3:	[Address_3]	Last Payment Date:	[Last_Payment_Date]
City:	[City]	Last Payment Amount:	«Expr»
State:	[State]	Avg Days to Pay - YTD	[Average_Days_To_Pay___Year]
Zip Code:	[Zip]	Total Sales - YTD	«Expr»
Phone Number:	[Phone_1]	Total Sales - LTD	«Expr»

Report Rendered:	[&ExecutionTime]	Customer Number:	[@CustomerNumber]
Transaction Date From:	[@TransactionDateFrom]	Transaction Date To:	[@TransactionDateTo]

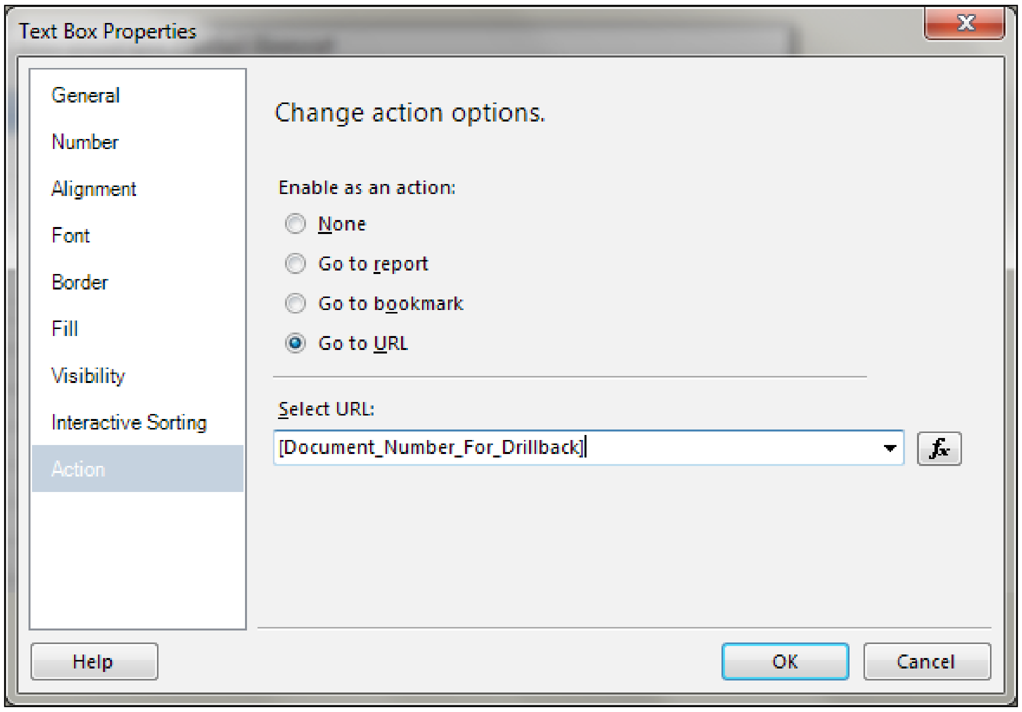
15. In the **Properties** pane, we want to find the **Action** property.



Action	
Action	None
Alignment	
Indent	
Padding	2pt, 2pt, 2pt, 2
SpaceAfter	
SpaceBefore	
TextAlign	Default
VerticalAlign	Default
Border	
BorderColor	#e5e5e5
BorderStyle	Solid
BorderWidth	1pt
Data Only	
DataElementN	
DataElement(	Auto
DataElementS	Auto

Action
Specifies an action associated with the report item. Actions include a j...

16. In the **Action** property, we want to click on the ellipsis to open the **Text Box Properties** window. Here we will select **Go to URL**, and select our drillback column from our source dataset.



Change action options.

Enable as an action:

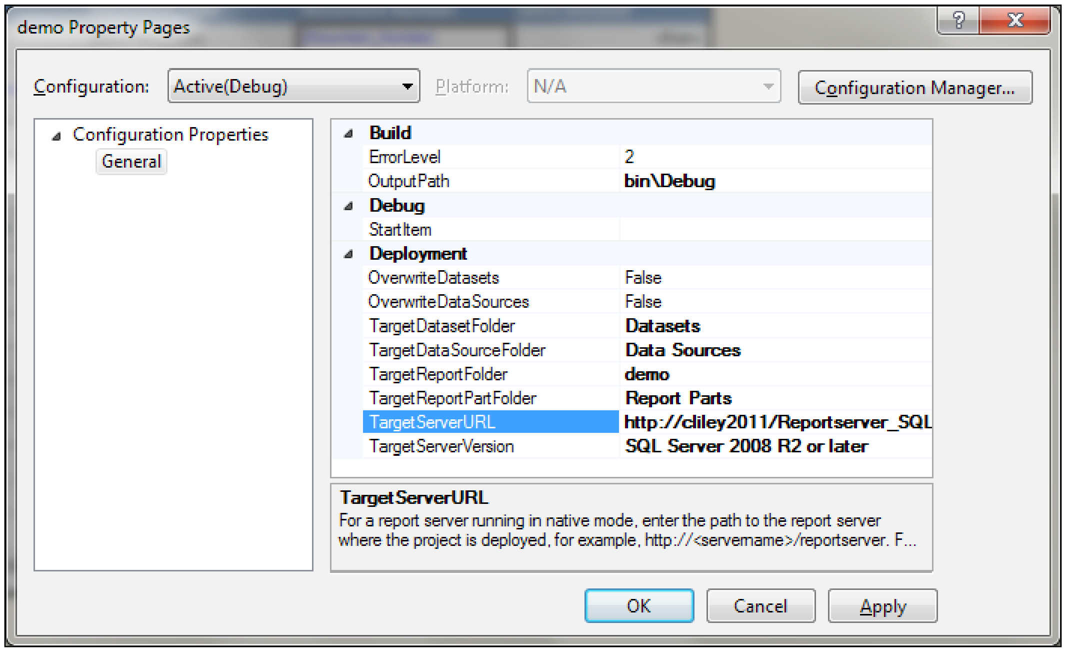
- None
- Go to report
- Go to bookmark
- [✓ Go to URL]

Select URL:

[Document_Number_For_Drillback]

17. Click on **OK** and then on **Save**.

18. Now that we have made our change to our report, we need to redeploy it to the report server. To do this, we need to set our project deploy options in **Solution Explorer**.
19. Right-click on the **Demo** project, and click on **Properties** to open the **Property Pages** window as seen in the following screenshot:

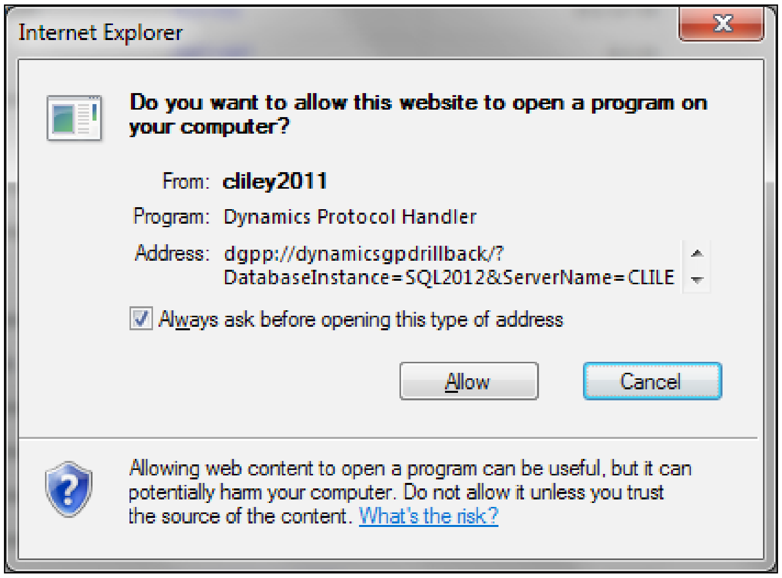


Build	
ErrorLevel	2
OutputPath	bin\Debug
Debug	
StartItem	
Deployment	
OverwriteDatasets	False
OverwriteDataSources	False
TargetDatasetFolder	Datasets
TargetDataSourceFolder	Data Sources
TargetReportFolder	demo
TargetReportPartFolder	Report Parts
TargetServerURL	http://cliley2011/Reportserver_SQL
TargetServerVersion	SQL Server 2008 R2 or later

TargetServerURL
For a report server running in native mode, enter the path to the report server where the project is deployed, for example, http://<servername>/reportserver. F...

20. Set **TargetServerURL** to be the report server site. The default location is typically `http://<computer name>/ReportServer`.
21. Make any necessary changes to the target folders. If replacing the old report, make sure you change the folder name here to the folder where the original report is located.
22. Click on **OK**.
23. Right-click on the `Receivables Detail.rdl` report, and select **Deploy**.
24. After the deployment finishes, you will be provided with messages in the **Output** pane located at the bottom of the window.

25. Browse back to the Report Manager and regenerate the report. We can now click on any of the document numbers to drill back to GP. Note that, the first time we do this, we will be asked to allow access, similar to the following screenshot:



Internet Explorer

Do you want to allow this website to open a program on your computer?

From: **cliley2011**

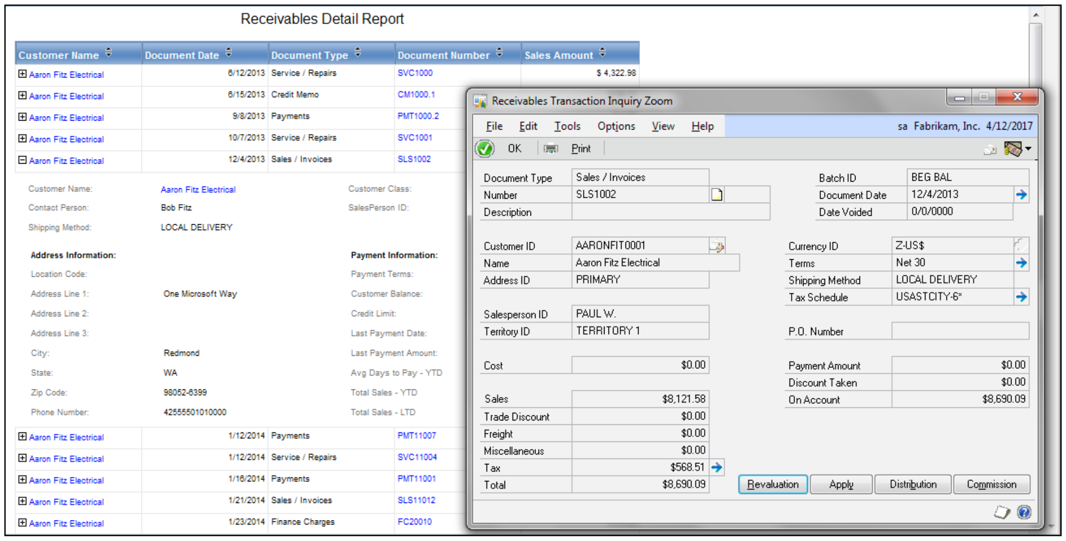
Program: Dynamics Protocol Handler

Address: dgpp://dynamicsgpdrillback/?DatabaseInstance=SQL2012&ServerName=CLILE

[✓] Always ask before opening this type of address

Allowing web content to open a program can be useful, but it can potentially harm your computer. Do not allow it unless you trust the source of the content. What's the risk?

26. After clicking on **Allow**, GP will open the inquiry window for the document we clicked.



Receivables Detail Report

Customer Name	Document Date	Document Type	Document Number	Sales Amount
⊞ Aaron Fitz Electrical	6/12/2013	Service / Repairs	SVC1000	$ 4,322.98
⊞ Aaron Fitz Electrical	6/15/2013	Credit Memo	CM1000.1	
⊞ Aaron Fitz Electrical	9/8/2013	Payments	PMT1000.2	
⊞ Aaron Fitz Electrical	10/7/2013	Service / Repairs	SVC1001	
⊟ Aaron Fitz Electrical	12/4/2013	Sales / Invoices	SLS1002	

Customer Name:	Aaron Fitz Electrical	Customer Class:
Contact Person:	Bob Fitz	SalesPerson ID:
Shipping Method:	LOCAL DELIVERY	

Address Information:		**Payment Information:**
Location Code:		Payment Terms:
Address Line 1:	One Microsoft Way	Customer Balance:
Address Line 2:		Credit Limit:
Address Line 3:		Last Payment Date:
City:	Redmond	Last Payment Amount:
State:	WA	Avg Days to Pay - YTD
Zip Code:	98052-6399	Total Sales - YTD
Phone Number:	42555501010000	Total Sales - LTD

⊞ Aaron Fitz Electrical	1/12/2014	Payments	PMT11007
⊞ Aaron Fitz Electrical	1/12/2014	Service / Repairs	SVC11004
⊞ Aaron Fitz Electrical	1/16/2014	Payments	PMT11001
⊞ Aaron Fitz Electrical	1/21/2014	Sales / Invoices	SLS11012
⊞ Aaron Fitz Electrical	1/23/2014	Finance Charges	FC20010

Receivables Transaction Inquiry Zoom

sa Fabrikam, Inc. 4/12/2017

Document Type	Sales / Invoices	Batch ID	BEG BAL
Number	SLS1002	Document Date	12/4/2013
Description		Date Voided	0/0/0000
Customer ID	AARONFIT0001	Currency ID	Z-US$
Name	Aaron Fitz Electrical	Terms	Net 30
Address ID	PRIMARY	Shipping Method	LOCAL DELIVERY
		Tax Schedule	USASTCITY-6*
Salesperson ID	PAUL W.		
Territory ID	TERRITORY 1	P.O. Number	
Cost	$0.00	Payment Amount	$0.00
		Discount Taken	$0.00
Sales	$8,121.58	On Account	$8,690.09
Trade Discount	$0.00		
Freight	$0.00		
Miscellaneous	$0.00		
Tax	$568.51		
Total	$8,690.09		

This is a fairly simple example of how you can modify the predefined reports and customize them to an organization's particular need. Although this example was simply adding a drillback to GP, we could just as easily add fields, change layouts, report groupings, add/remove parameters, and so on. Also, this example used Visual Studio to make these modifications, but as you will learn in the next section, we can use SSRS Report Builder to make modifications as well.

Creating a new reporting metric via Report Builder

Earlier in this chapter, we discussed the deployment of predefined charts and KPIs that can then be made visible on the GP 2013 Homepage in Business Analyzer. In this section, we'll take a look at creating a metric that shows us the top ten items on unposted sales orders by extended price. A metric of this sort can give members of our organization insight into which items are generating the most sales at the moment. As the data source for this metric, we'll use one of the SQL views that is generated in each GP company database by the predefined Excel Reports deployment that was covered in *Chapter 3*, *Working with the Builders – SmartList and Excel Reports*, of this book: `dbo.SalesLineItems`. Remember, SSRS metrics are only available for GP 2010 and later, and SQL Server 2008 and later environments!

In the previous example, we used Visual Studio to modify a default report. In this example, however, we are going to use Report Builder to create a new metric from scratch. Report Builder is a component of SSRS that can be used to modify and create new reports and charts. We will use Report Builder in this example because it has a simple, easy-to-use user interface that will seem familiar to anyone who has used the Microsoft Office suite of programs.

To begin, we will open our web browser to our Reports Manager or Reports Library depending on whether or not SSRS is configured in the **Native** or **SharePoint Integrated** mode. From this location, we should be able to see our GP company database folder that contains the predefined reports and charts.

If SSRS is configured in the **SharePoint Integrated** mode, navigate to **New | Report Builder Report** to open the Report Builder application. If SSRS is configured in the **Native** mode, click on the button that says **Report Builder**. Either way, this will launch the Report Builder application from which we can begin creating a new report as follows:

1. In the blank report that appears, right-click, and navigate to **Insert | Chart** to open the **Select Chart Type** window.

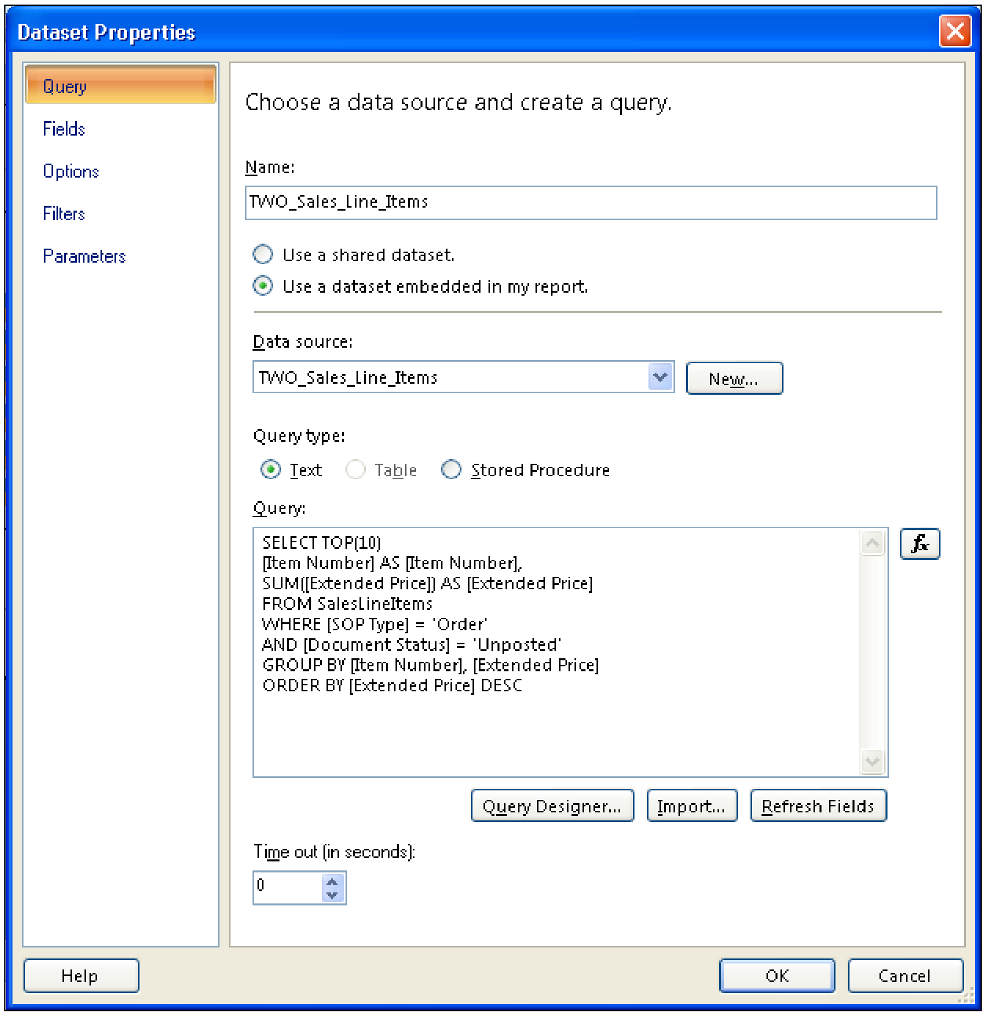
2. In the **Bar** chart section, select the **Bar** chart type, and click on **OK**.
3. In the **Dataset Properties** window, enter `TWO_Sales_Line_Items` as the name for the new dataset that we will create.
4. Select **Use a dataset embedded in my report**.
5. Click on the **New** button in the **Data Source** field to open the **Data Source Properties** window.
6. Name the data source as `TWO_Sales_Line_Items`.
7. Select **Use a connection embedded in my report**.
8. In the **Select** connection type drop-down menu, select **Microsoft SQL Server**. Notice that we can also connect to a variety of other sources, including **Microsoft SQL Server Analysis Services**.
9. Click on **Build** to open the **Connection Properties** window.
10. Enter the name of the GP company database server in the **Server Name** drop-down menu.
11. Enter the GP company database name in the next drop-down menu, and click on **OK** twice. For this example, the selected GP database should be one for which predefined Excel Reports have been deployed.
12. In the **Dataset Properties** window, select **Text** as **Query Type** and enter the following SQL script in the **Query** box:

    ```
    SELECT TOP(10)
    [Item Number] AS [Item Number],
    SUM([Extended Price]) AS [Extended Price]
    FROM SalesLineItems
    WHERE [SOP Type] = 'Order'
    AND [Document Status] = 'Unposted'
    GROUP BY [Item Number], [Extended Price]
    ORDER BY [Extended Price] DESC
    ```

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

13. Click on **OK**.

14. The window should now look like the following screenshot:



Choose a data source and create a query.

Name: TWO_Sales_Line_Items

Use a shared dataset. [✓ Use a dataset embedded in my report.]

Data source: TWO_Sales_Line_Items

Query type: [✓ Text] Table Stored Procedure

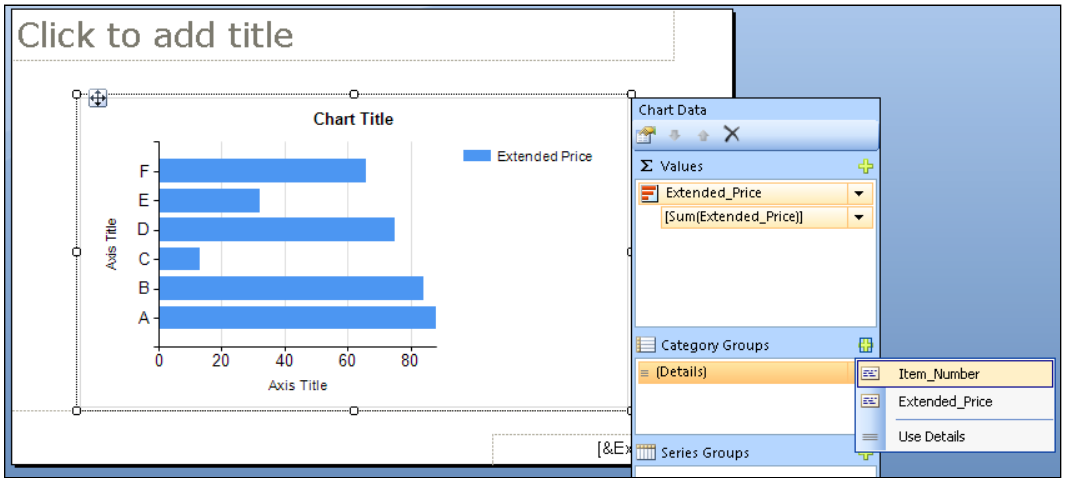
Query:

```
SELECT TOP(10)
[Item Number] AS [Item Number],
SUM([Extended Price]) AS [Extended Price]
FROM SalesLineItems
WHERE [SOP Type] = 'Order'
AND [Document Status] = 'Unposted'
GROUP BY [Item Number], [Extended Price]
ORDER BY [Extended Price] DESC
```

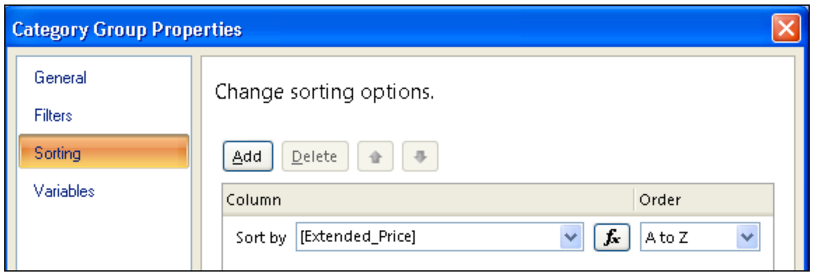
Time out (in seconds): 0

A sample view of the chart will be added to the blank template in the **Report Builder** pane. The chart displaying actual data will not appear until we have published the report and actually generated the chart. Before we do this, we should make several formatting changes to the chart to make it easier to read. Report Builder's format is fairly intuitive, especially for those of us who are familiar with the Microsoft Office suite of applications. Formatting this chart will be fairly easy for anyone who has experience working with charting tools in Microsoft Excel. We will take a look at some of the formatting changes we can make to this chart, but first let's make sure we define the data labels and series values for our chart as follows:

1. Double-click anywhere in the chart area. A pop up window entitled **Chart Data** should appear. Click on the green plus (**+**) sign next to **Values**, and select **Extended_Price** to define the series values of the chart.
2. In the **Category Groups** section, click on the green plus (**+**) sign, and select **Item_Number** as seen in the following screenshot. This will set the item number as the data label on the x axis of this chart.



3. Once **Item_Number** is added to the **Category Groups** section, click on the **Item Number** drop-down box, and select **Category Group Properties**.
4. In the **Category Group Properties** window, select **Sorting**. Change the **Sort by** drop-down value from **[Item_Number]** to **[Extended_Price]**. This will change the sorting functionality of the chart from alphabetical by item number to descending order by extended price. Click on **OK** to close the window.



Change sorting options.

Column	Order
Sort by [Extended_Price]	A to Z

Now that we've defined our data for the chart, let's make some formatting changes. Feel free to experiment with additional formatting changes to find the appropriate visual effect as follows:

1. Use grab handles to resize the chart in the center of the template.
2. Double-click on **Chart Title**, and rename the chart to `Top Ten Items on Unposted Sales Orders (Extended Price)`.
3. Delete both the x axis and y axis title by right-clicking on them, and deselecting **Show Axis Title**.
4. To ensure that all of the data labels appear on the x axis, right-click on the vertical axis values, and select **Vertical Axis Properties**. On the **Axis Options** tab, in the **Axis Range and Interval** section, select the **Side Margins** drop-down box, and select **Disabled**. Click on **OK** to return to the chart.
5. To format the y axis data labels, right-click on the horizontal axis values, and select **Horizontal Axis Properties**. Click on the **Number** tab. Change the number format category to **Currency**. Select the option **Use 1000 Separator**. Click on **OK**.

Once we are done formatting the chart, we should save it to the Reports Manager or Reports Library. To ensure that the chart will be available in GP, we need to add the chart to the `Charts and KPIs` folder under one of the company-module folders that was created by the deployment of the predefined SSRS reports and metrics as follows:

1. Click on the **Report Builder** menu button, and select **Save As**.
2. Browse to the Report Manager (or Reports Manager) and find the GP company database folder. Expand this folder and then expand the folder for the **Sales** module. Expand the `Charts and KPIs` folder. At this point, the file path should be similar to the following URL (if using the **SharePoint Integrated** mode):

   ```
   http://<computername>/reports/Reports Library/TWO/Sales
   ```

3. Give the file a name and click on the **Save** button.

Our saved chart will now show up in the list of charts and KPIs available for selection in GP when we are selecting the metrics in Business Analyzer for our GP Homepage. By default, the metric and the data within it will appear for Windows users who have administrative privileges on the report server as well as the GP database. If this is not the case, refer to the instructions on the Reporting Services Security Setup section for more information on how to do this.

Summary

In this chapter, we explored some of the ways in which SSRS can be integrated with Dynamics GP 2013. Although experienced developers and consultants have been using SSRS to develop reports for Dynamics GP for quite some time, it is only through recent versions of GP that Microsoft has begun providing predefined SSRS reports and metrics that can be deployed to a report server. By first installing SSRS and then deploying these predefined reports and metrics, even developers and consultants who do not have a tremendous amount of experience with SSRS can begin taking advantage of its versatile and useful functionality. We demonstrated how this is possible by discussing how to modify one of the existing SSRS reports. We also demonstrated how users can take advantage of the SSRS-based metrics and Business Analyzer in GP 2013 and SQL Server 2012 by creating our own metrics. Although this chapter has only scratched the surface of what can be accomplished with SSRS, it will set your organization on the right path towards seeing the benefit of this toolset.

In the next chapter, we will take a look at another component of SQL Server: **SQL Server Analysis Services** (**SSAS**). This is a powerful **online analytical processing** (**OLAP**) tool that can be used to mine an organization's data and provide critical business intelligence capabilities. It is capable of aggregating large sets of data quickly and efficiently so that this data can be queried through languages, such as the **Multidimensional Expression** (**MDX**) language.

Microsoft has developed a product named Analysis Cubes for Excel that utilizes SSAS and data generated by Dynamics GP. Although the default product can be used straight out of the box, this is a highly customizable reporting tool. Before we cover the process of querying the data generated by Analysis Cubes, we're going to take a different approach and first demonstrate how the product can be customized to meet your organization's business intelligence requirements. We'll cover some of the basic design topics of Analysis Cubes for Excel including its various SQL Server components and how they can be customized for an improved end user experience.

6
Designing Your Analysis Cubes for the Excel Environment

Business intelligence—every organization wants it. But can every organization actually afford it? If our organization is thinking of a business intelligence environment that has been built specifically for a single company with highly visual and fancy teams and individual-based dashboards and KPIs, the answer is probably *No, we can't afford it!* The time and effort required to build such a custom solution is not cheap, and without clear focus and vision from the get-go, these kinds of projects can quickly get bogged down. For the majority of us, we need something cheaper and less expensive to maintain.

Let's not lose sight of what we're really after when we say our organization needs BI capabilities. Simply put, we need improved tools for finding, analyzing, and making sense of the mountain of data that we are storing on our servers and databases. As we improve our ability to store this data in **Enterprise Resource Planning (ERP)** systems, it's only going to get harder to find and make sense of this data. This data is unique to our organization; no other organization has access to the same data that we do. If we want to be successful in an increasingly competitive business environment, our company has to be able to find trends in its data and act on those trends before anyone else can. We need BI tools to help us do this.

Fortunately, for those of us who work in the Microsoft Dynamics GP space, we have a readily available BI tool available at our disposal: Analysis Cubes for Excel. With this product, we can quickly and easily implement an out of the box BI environment that, with just a few extra tweaks and modifications, can provide users in our organization with the ability to interact with data in ways that until now, they could only dream of. The real beauty of this tool doesn't lie in the fact that it's relatively easy and inexpensive to implement as BI solutions go (although this is a major plus); instead, the beauty is found in providing our organization with a straightforward BI platform that will enable our users to think outside the box even more when it comes to how they analyze their data. Our users will actually have the control and ability they always craved for while creating their own reports and metrics.

The components of the Analysis Cubes for Excel product are also highly customizable. So, even though many companies are perfectly happy to implement the cubes, make a few minor tweaks to the database, and then turn their users loose; many other companies are choosing to extend their Analysis Cubes environment even further and design it to be even more relevant to their organization's needs. There's no right way to do this. It really boils down to what works best for each organization.

In the next two chapters, we're going to take a look at how we can implement and design Analysis Cubes for Excel in our own organization. This chapter will explore the backend topics of the Analysis Cubes product as follows:

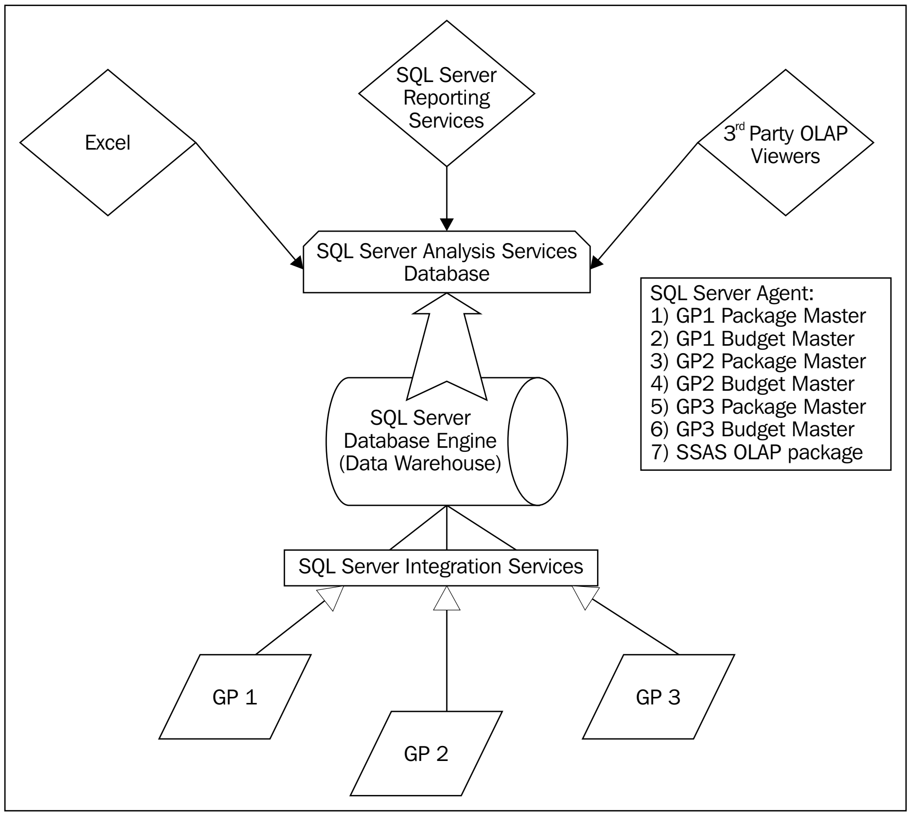
- Understanding how the components of the Analysis Cubes environment interact
- Providing a step-by-step guide for installing the product
- Reviewing simple modifications that can customize our new business intelligence solution for our organization

The next chapter will focus on the actual building and designing of reports against the databases created and modified in this chapter.

Understanding the components of the Analysis Cubes environment

The Analysis Cubes product offering is composed of several different components, each of which can be modified and customized to fit the needs of our organization. Although each of these components plays a critical role in the overall end product, understanding how each component works in relation to the others can be challenging. Fortunately for us, the components that make up the cubes are fairly well-known as they are spread across the Microsoft SQL Server stack.

Let's take a look at the various components that comprise the Analysis Cubes environment. The following figure shows an overview of an Analysis Cubes solution deployed in an environment containing three separate GP company databases:



From the preceding figure, we can see the various components and their connections to each other. Notice how the users are connecting to the **SQL Server Analysis Services** (**SSAS**) database via viewers such as Microsoft Excel, SQL Server Reporting Services, or any one of a number of viewers designed to read data from an Analysis Services database. As we'll learn later, this is where our data eventually ends up, and all of our reporting will be done against this database.

Aside from the GP company databases and the OLAP viewers, which we'll discuss in the next chapter, we can see that Analysis Cubes is comprised of four different elements, all of which coincide with a different component of SQL Server as follows:

- A data warehouse maintained within the **SQL Server Database Engine**
- A SSAS database
- Several integration packages found within **SQL Server Integration Services** (**SSIS**)
- A cube load and data processing job scheduled via SQL Server Agent

SQL Server Database Engine (data warehouse)

The centerpiece of the Analysis Cubes environment is the data warehouse. This collection of approximately 50 tables serves as a repository for data that will eventually be processed by the Analysis Services database. Regardless of the number of company databases that are included in the installation of the cubes, only a single data warehouse must be maintained, making it far easier to manage in cases where we might have multiple GP company databases.

Connecting to the data warehouse

To add or remove tables, view data stored in these tables, or make other modifications to the data warehouse, open SQL Server Management Studio and connect to the Database Engine component of SQL Server. In the list of databases that appear under the **Databases** node, we will see **DynamicsGPWarehouse**, which is the default name for the data warehouse created by the installation.

Understanding the tables in the data warehouse

With over 50 tables to choose from, it can often be a challenge to identify the right table to work within the data warehouse. Fortunately, our search can be made easier by thinking of the data warehouse as an extraordinarily simplified version of the GP company databases. Back in *Chapter 2*, *Where Is My Data and How Do I Get to It?*, we reviewed the often confusing naming and numbering schema associated with tables in GP company databases. With Analysis Cubes, however, the data warehouse tables and their associated fields are much easier to understand. For example, instead of having to recall that the `Customer Master` table in GP 2013 is the `RM00101` table, we see a table in **DynamicsGPWarehouse** actually named `Customers`. As in GP, we can logically group the tables in the data warehouse by their function. In the data warehouse, we have the following kinds of tables:

- **Setup and Maintenance tables**: The contents of tables, such as `SystemVersion` and `LastUpdated`, should never be deleted. Tables such as `SystemVersion` identify settings selected during the installation. Other tables such as `LastUpdated` may need to be modified as part of regular maintenance and updates required for the data warehouse.
- **Dimension tables**: Just like the master tables in the GP company databases (presented in *Chapter 2*, *Where Is My Data and How Do I Get to It?*), these tables contain data specific to individual records maintained in GP. For example, the `Vendors` and `Customers` tables contain a single record for each vendor and customer respectively. As we will see later, data found in these tables are used as attributes in the dimensions of the Analysis Services database.
- **Fact tables**: Several tables in the data warehouse, such as `GLTransactions` and `SalesDetail` contain transaction-related data. Depending on the module, the table may include distribution information, so we can see amounts that have been posted to various GL accounts. Again, as with the dimension tables we just mentioned, the fact tables provide a fairly simplified version of the tables and data we might find in the GP company databases. Although some of the data found in the fact tables is used by the dimensions in the Analysis Services database, the primary purpose of the records in these tables is to provide the numbers for the cubes. As we'll soon see, these numbers display as measures in the Analysis Services database.

Remember, only a single data warehouse is created during the install regardless of the number of GP company databases included during the installation. If we take a look at the master and transaction tables, we can see that each record has an associated **CompanyID** field. This allows us to combine data from multiple company databases into a single database without losing the ability to identify the originating source of each master or transactional record.

An SSAS database

Unlike the data warehouse, which is a relational database, the Analysis Services database is actually a collection of **Online Analytical Processing** (**OLAP**) data sources, as it is more commonly called, and data mining objects. These objects work together to aggregate data into predefined hierarchies and dimensions, giving us the ability to mine large sets of data quickly and easily. When browsing cubes in an Analysis Services database, users will find themselves beginning with a very high-level, aggregated look at the data. Then, when our user spots an anomaly or some other piece of data that looks interesting, he or she can quickly drill down to the underlying source data to learn more about it. This is made possible by the incredible OLAP and data mining functionality found in SSAS databases.

One common misconception among many who are beginning work with Analysis Cubes is that in order to view data, we must connect to the data warehouse. In reality, when we use an OLAP viewer such as Microsoft Excel, we are actually creating a data connection to an Analysis Services database. If the default database name is kept during the installation, we will be creating a connection to a database named `Dynamics GP Analysis Cubes`.

In the Analysis Cubes environment figure that we included earlier in this chapter, we can see that the Analysis Services database sits on top of the data warehouse. Through a process known as "processing the Analysis Services database", data stored in the data warehouse is pulled into the Analysis Services database, where it is aggregated and stored in the various dimensions and measure groups that comprise the cubes. Without getting too much into the specifics of the various OLAP methodologies, this is accomplished through the use of the **Multidimensional OLAP** (**MOLAP**) technology. Rather than compute report values at the time an end user requests that data via a report (as is done with **Relational OLAP** (**ROLAP**) environments), MOLAP environments compute these aggregated values ahead of time, making the cube browsing experience much faster for the end user.

By default, the Analysis Services database that we will create during our installation utilizes MOLAP. Although MOLAP environments provide us with quicker access to the data at runtime, some drawbacks of this type of environment do exist. In order to process and aggregate the data from the data warehouse into the Analysis Services database, we must utilize a substantial amount of server resources. Although most server environments are capable of handling the requirements of processing the SSAS database without adding additional resources, most companies will choose to process the database during off-peak hours to prevent performance issues for the rest of the server. This leads us to another potential drawback of MOLAP databases: data lag. If our server hardware only processes the data from the data warehouse once a day, then it stands to reason that the data in our cubes will always lag behind real-time data and be slightly out of date. Whether or not this lag is acceptable is a decision we will have to weigh against the perceived benefits of the faster and easier data access that the cubes stand to offer.

SSIS packages

In the last section, we discussed how processing the Analysis Services database causes the data from the data warehouse to be aggregated and stored in the SSAS database objects. Now, it is time for us to discuss how the data made it to the data warehouse in the first place. This is the job of SSIS. The installation of Analysis Cubes will create a series of SSIS packages that are responsible for collecting data from the various GP company databases included in our installation and adding that data to the data warehouse.

The data load for Analysis Cubes is, by and large, an incremental data load. Rather than wiping the slate clean each time data is loaded, most of the SSIS packages attempt to identify the records that have been inserted or modified in the GP company database since the last time a data load package was executed. Only this additional data is transferred over to the data warehouse and the existing data in the data warehouse remains unchanged. Because the Analysis Services database is processed against the data warehouse, we should make sure that the data load packages are executed just prior to the database processing. This will ensure that the SSAS database will be processed with the most up-to-date information available from the GP company database(s).

Several different kinds of SSIS packages are created by the installation as follows:

- **Individual Load Packages**: A series of approximately 25 generic load packages is created. These packages are responsible for loading the master and transactional tables in the data warehouse. These packages, however, should not be run individually.
- **Company Master Packages**: Instead of executing the individual load packages, users should run the master packages that are created to correspond with each company database. Each master package is designed to pass down connection information for a specific GP company database to the individual load packages.
- **Budget Packages**: A series of packages are created to allow us to load budget data from each GP company database. Run these only in the event that GP budget data should be included with the cubes. Later on in this chapter, we will address some additional steps required to load budget data.
- **Cube Processing Package**: A single OLAP package is created that when executed, processes the entire Analysis Services database.

That's a lot of packages to keep track of, but as we'll see in the next section, we can use another SQL Server-related tool to keep track of them for us.

In versions of Analysis Cubes for GP 10.0 and earlier, SSIS packages were encrypted. This meant that users who wanted to add additional fields or see more data on where a particular field originated were out of luck. Fortunately, several months after GP 2010 was released, Microsoft announced that the packages would no longer be encrypted. This is great news and greatly enhances our ability to customize the data that we are bringing into the data warehouse.

SQL Server Agent job

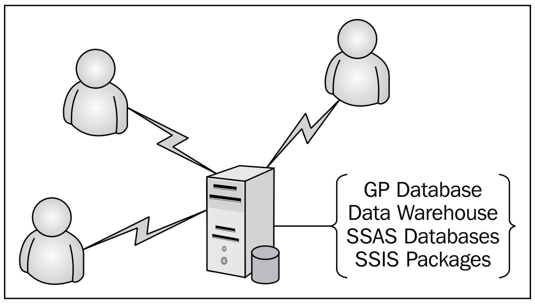
SQL Server Agent is a Windows service accessible from within SQL Server Management Studio that allows us to create and schedule jobs. In the case of Analysis Cubes, we can use SQL Server Agent to create a job containing the master and budget SSIS packages as well as the OLAP processing package that should be run after data has been loaded into the data warehouse. Once we've set up our job to execute SSIS packages in the desired order, we can schedule the job to run at a certain time. For example, we could set SQL Server Agent to execute a job that loads the data warehouse and then processes the SSAS database, beginning at 3 a.m. in the morning. This way, we can ensure that the cubes will be ready for access first thing in the morning.

Multiple tier installations

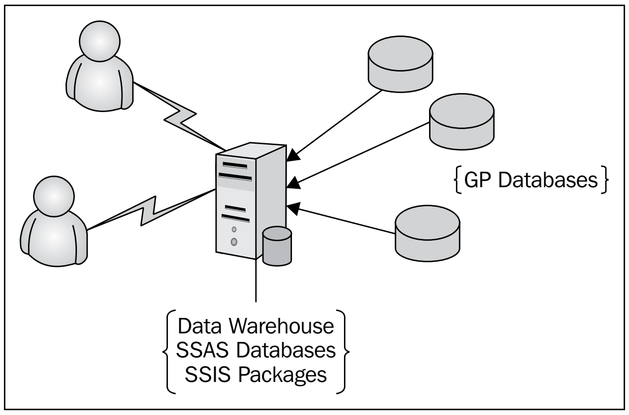
Now that we've covered the various components of Analysis Cubes, let's talk about the server environment to which these components will be installed. When we go through the installation in the next section, we'll see that we are given a choice which server each component should be installed to. Each choice offers several pros and cons.

Essentially, we have three options as follows:

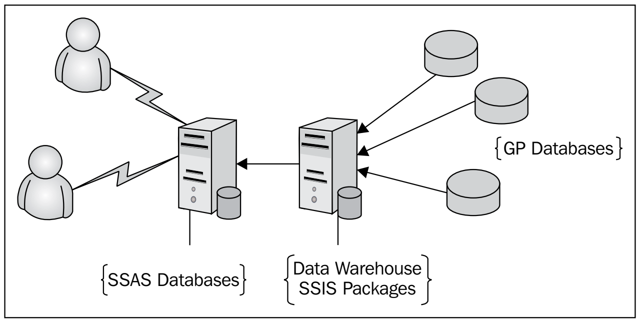
- **Single-server installation**: In this option, the data warehouse, the SSAS database, SSIS packages, and GP company databases will all reside on a single server. This configuration minimizes the amount of time required to load the data from GP company databases into the data warehouse. However, in order to process the Analysis Services database, we will take up valuable server resources that may have an adverse impact on users who are in the GP environment.



- **Two-server installation**: In this option, the data warehouse, the SSAS database, and SSIS packages are installed on a server separate than the server containing GP company databases. Now, when we process and query the SSAS database, we can ensure we are minimizing the impact on GP performance. However, this option does, introduce the possibility of longer data load times as data must now be transferred to another instance, and depending on which portion of a package is executing at the time, across a linked server.



- **Three-server installation**: In this option, the data warehouse and SSIS packages are installed on one server, the SSAS database is installed on a second server, and GP databases are hosted on a third server. With this configuration, the SSAS database is given the maximum amount of resources for data processing and querying, but we are still faced with the issue of potentially long data load times as data is pulled into the data warehouse via a linked server.



Before we begin installing the cubes in our environment, we should take some time to think of the pros and cons of the different tier-installation options available. For some of us, our decision may be easy: a lack of dedicated server resources may force us to utilize the single server install and schedule the SSAS database processing for off-peak hours to reduce performance issues. Other organizations may have a dedicated report server on which the SSAS database and the data warehouse can be kept. We'll still want to limit the frequency with which we process the SSAS database, but at least we can feel more comfortable in doing this during business hours without impacting the performance of our ERP database.

Installing Analysis Cubes

Now that we have a basic understanding of what Analysis Cubes for Excel is as well as the components that make up the Analysis Cubes environment, we will now walk through the installation process for Analysis Cubes for Excel. The installation of Analysis Cubes for Excel is actually quite simple, but it is also very dependent on having the proper permissions to perform the installation, as well as completing steps in a particular order. The installation process is made up of the following three steps:

- Reviewing the pre-installation checklist
- Installing the Server Configuration Wizard
- Using the Server Configuration Wizard to deploy the cubes

Reviewing the preinstallation checklist

Before proceeding with our installation of Analysis Cubes, we need to review our pre-installation checklist as follows:

- **Review system requirements for server components**: This includes determining which of the deployment configurations discussed earlier in this chapter we will be using in our environment
- **Review security requirements for installing and configuring server components**:
    - To install and configure server components, the Windows user account that will be used to perform the install must be in the Windows groups in the following chart, as described in the latest Analysis Cubes for Excel manual:

Server computer	**Windows groups**
Microsoft Dynamics GP Server Computer	Administrators
SQL Server Computer for Microsoft Dynamics GP Company Databases	All versions of SQL Server: Administrators SQL Server 2008: SQLServerMSSQLUser SQL Server 2012: SQLServerMSSQLUser
Data Warehouse Database Server Computer (with Integration Services installed)	All versions of SQL Server: Administrators SQL Server 2008: SQLServerMSSQLUser; SQLServerDTSUser SQL Server 2012: SQLServerMSSQLUser
SQL Server Analysis Services Computer	All versions of SQL Server: Administrators SQL Server 2008: SQLServerMSOLAPUser SQL Server 2012: SQLServerMSSQLUser

    - The Dynamics GP server must be given access to read and write data on the data warehouse database server if it is a different physical machine

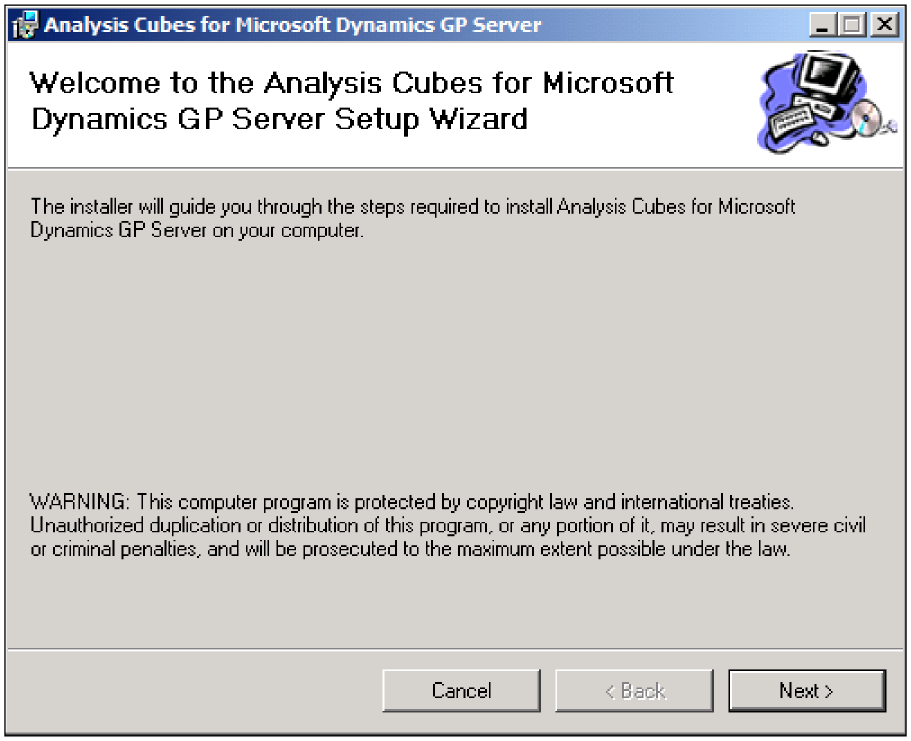
- Gather information required for configuring the server components as follows:
    - Dynamics GP company databases that will be used to populate the data warehouse database. Generally, only Production databases are included in this list.
    - The name that will be used for the data warehouse database, if it will differ from the default choice of `DynamicsGPWarehouse`.
    - The SQL Server name or instance name on which the data warehouse database will reside.
    - The locations of the data and log-files on the SQL Server instance for the newly created data warehouse.
    - SSIS packages to install.
    - Determine whether or not to:
        - Populate the warehouse database with detailed General Ledger transaction information for various modules or to use summary information
        - Populate each cube with all transactions or enter an earliest date to import them from company databases
        - To include multicurrency information and if so, which reporting currency and exchange rate table to use for each company
        - Define a password for the DynamicsUser SQL Server login account that will be used by company databases to access the data warehouse database using SQL Server Authentication

Installing the Server Configuration Wizard

Now that we have ensured we have met all system requirements and obtained the necessary security permissions to perform the installation, we can proceed with installing the Server Configuration Wizard . The installation can be started from the Dynamics GP 2013 Installation Media or by downloading the Analysis Cubes Installer package from either the PartnerSource or CustomerSource portals.

To install the Server Configuration Wizard, follow these steps:

1. Locate the `Microsoft_DynamicsGP11_AnalysisCubesServer_x86.msi` file you downloaded and double-click on the file to open **Dynamics GP Server Setup Wizard**.



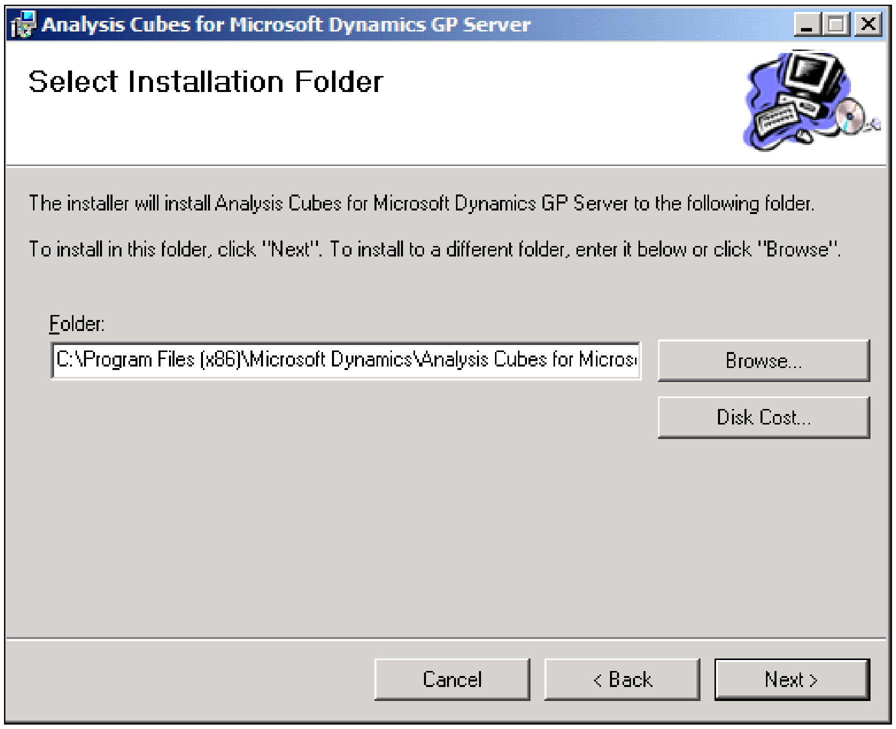
Welcome to the Analysis Cubes for Microsoft Dynamics GP Server Setup Wizard

The installer will guide you through the steps required to install Analysis Cubes for Microsoft Dynamics GP Server on your computer.

WARNING: This computer program is protected by copyright law and international treaties. Unauthorized duplication or distribution of this program, or any portion of it, may result in severe civil or criminal penalties, and will be prosecuted to the maximum extent possible under the law.

2. Click on **Next** to proceed.

3. On the **License Agreement** window, select the **I Agree** radio button, and click on **Next**.


Analysis Cubes for Microsoft Dynamics GP Server
Select Installation Folder
Folder:
Browse...
Disk Cost...
Cancel
< Back
Next >

The installer will install Analysis Cubes for Microsoft Dynamics GP Server to the following folder.

To install in this folder, click "Next". To install to a different folder, enter it below or click "Browse".

C:\Program Files (x86)\Microsoft Dynamics\Analysis Cubes for Micros

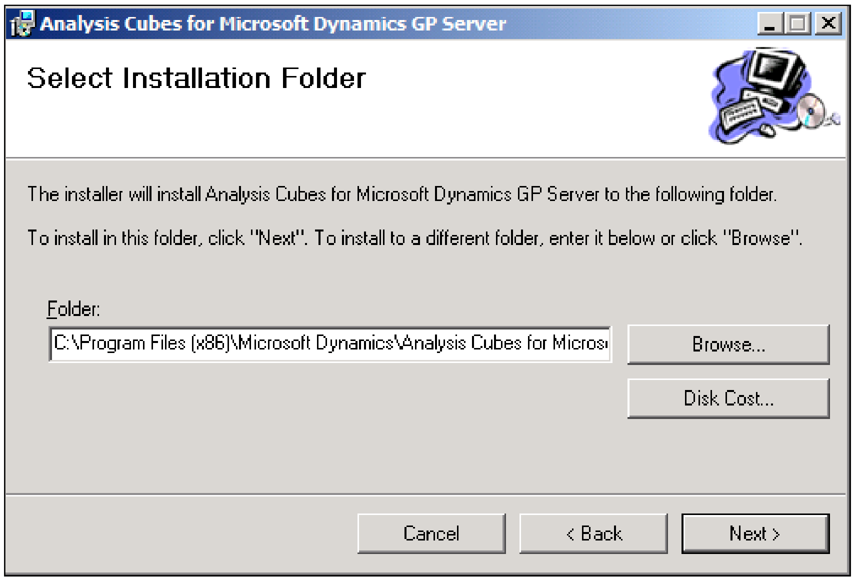
4. Enter the path for the destination folder to install the server configuration wizard program in, or accept the default. The default path is `C:\Program Files\Microsoft Dynamics\Analysis Cubes for Microsoft Dynamics GP Server\`. Click on **Next**.
5. Click on **Next** to complete the installation.
6. Click on **Close** to close the wizard.

Using the Server Configuration Wizard to deploy cubes

Once we have installed the Server Configuration Wizard, we need to run the wizard to actually deploy the cubes. During the configuration wizard installation process, two files, named `Microsoft.Dynamics.GP.AnalysisCubes.ConfigurationWizard2008.exe` and `Microsoft.Dynamics.GP.AnalysisCubes.ConfigurationWizard2012.exe` are extracted to the destination folder. Shortcuts to these files, named **Analysis Cubes Configuration Wizard for SQL Server 2008** and **Analysis Cubes Configuration Wizard for SQL Server 2012** are also created on the desktop. Make sure you run the file for the version of SQL you are using. In the examples throughout this chapter and *Chapter 7*, *Utilizing Analysis Cubes for Excel for Dynamic Reporting*, we will be using Analysis Cubes for SQL 2012.

To use the Server Configuration Wizard to deploy the cubes, follow these steps:

1. Browse to the folder location where the Server Configuration Wizard was installed, or run **Analysis Cubes Configuration Wizard for SQL Server 2012** from the desktop.
2. On the **Data Warehouse Setup** window, enter the values for the fields based on the information you gathered in the pre-installation checklist section.


Analysis Cubes for Microsoft Dynamics GP Server
Select Installation Folder

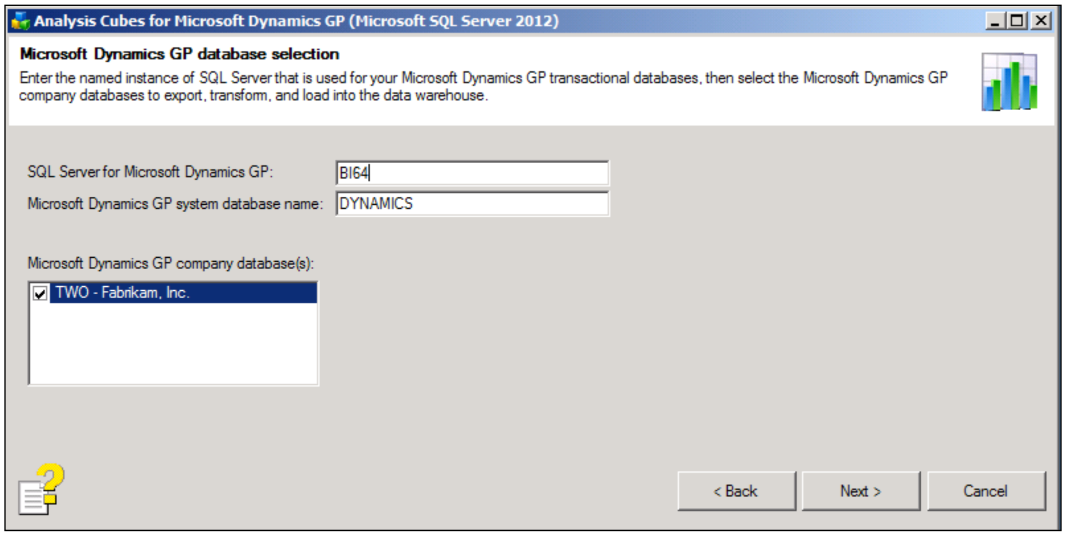
The installer will install Analysis Cubes for Microsoft Dynamics GP Server to the following folder.

To install in this folder, click "Next". To install to a different folder, enter it below or click "Browse".

Folder: C:\Program Files (x86)\Microsoft Dynamics\Analysis Cubes for Micros

Browse...
Disk Cost...
Cancel
< Back
Next >

3. On the **Microsoft Dynamics GP database selection** window, enter your Dynamics GP server name. Because GP 2013 introduced the concept of named system databases, a new field now appears in the GP 2013 installer for **Microsoft Dynamics GP system database name**. Enter the name of your system database in this field, select each company database(s) that will be included in the data warehouse, and click on **Next**.


Analysis Cubes for Microsoft Dynamics GP (Microsoft SQL Server 2012)

Microsoft Dynamics GP database selection

Enter the named instance of SQL Server that is used for your Microsoft Dynamics GP transactional databases, then select the Microsoft Dynamics GP company databases to export, transform, and load into the data warehouse.

SQL Server for Microsoft Dynamics GP: BI64

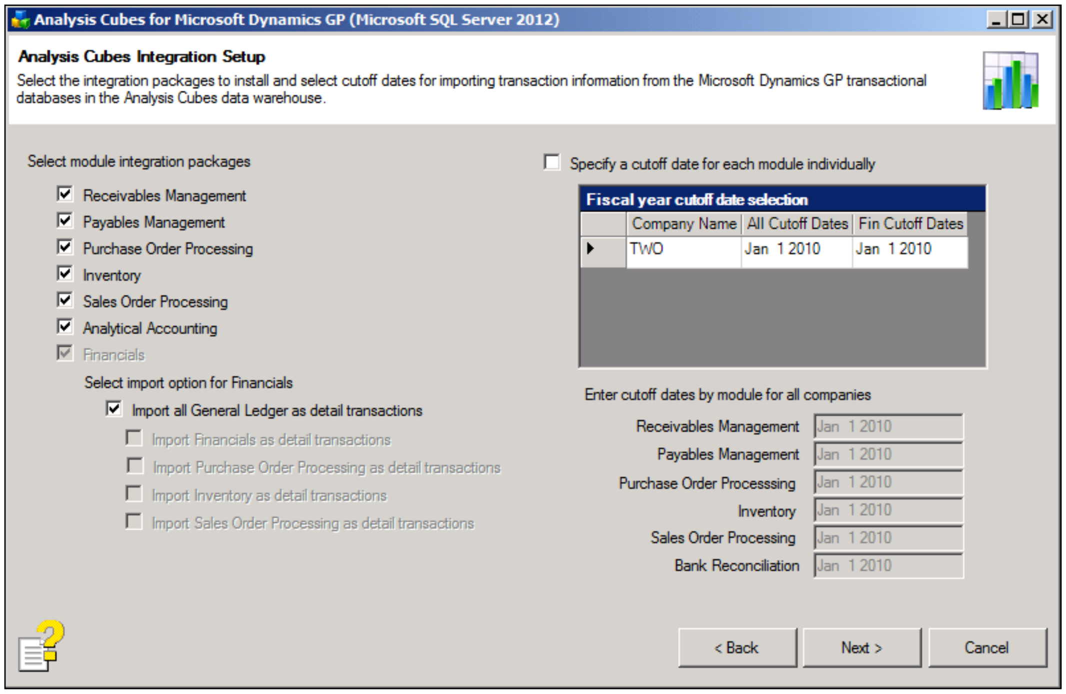
Microsoft Dynamics GP system database name: DYNAMICS

Microsoft Dynamics GP company database(s):

- [✓] TWO - Fabrikam, Inc.

Generally, only production databases will be selected in this window. Because the data warehouse combines data for all companies into a single environment, it can be confusing to users to have a collection of test and production data within one location.

4. On the **Analysis Cubes Integration Setup** window, select the modules to include in the data warehouse and whether to import transactions in detail or in summary. A cutoff date can also be selected to limit transactions that are loaded.
5. Click on **Next**.



Analysis Cubes for Microsoft Dynamics GP (Microsoft SQL Server 2012)

Analysis Cubes Integration Setup

Select the integration packages to install and select cutoff dates for importing transaction information from the Microsoft Dynamics GP transactional databases in the Analysis Cubes data warehouse.

Select module integration packages

- [✓] Receivables Management
- [✓] Payables Management
- [✓] Purchase Order Processing
- [✓] Inventory
- [✓] Sales Order Processing
- [✓] Analytical Accounting
- [✓] Financials

Select import option for Financials

- [✓] Import all General Ledger as detail transactions
- [ ] Import Financials as detail transactions
- [ ] Import Purchase Order Processing as detail transactions
- [ ] Import Inventory as detail transactions
- [ ] Import Sales Order Processing as detail transactions

[ ] Specify a cutoff date for each module individually

Fiscal year cutoff date selection

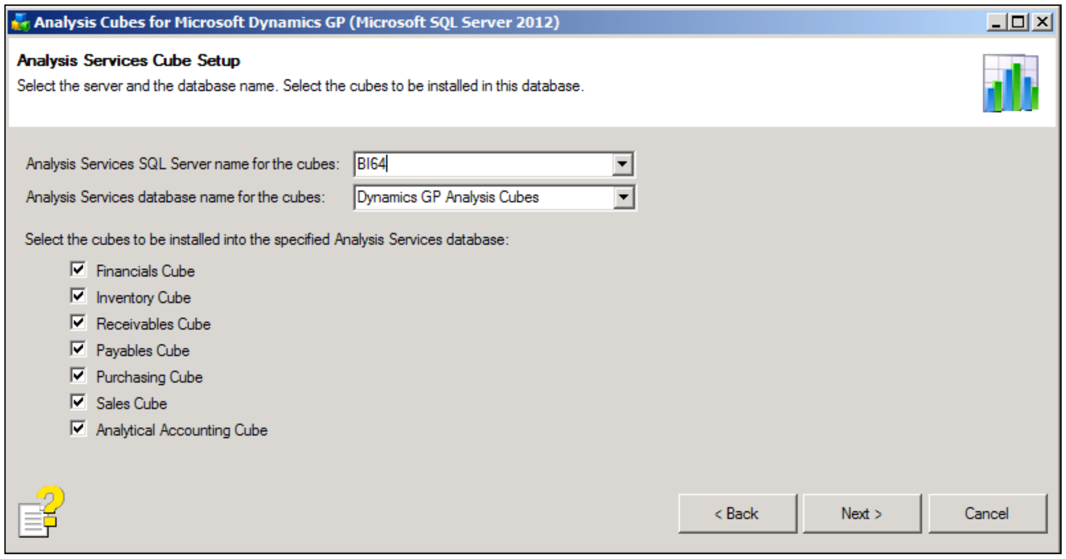
Company Name	All Cutoff Dates	Fin Cutoff Dates
TWO	Jan 1 2010	Jan 1 2010

Enter cutoff dates by module for all companies

Receivables Management	Jan 1 2010
Payables Management	Jan 1 2010
Purchase Order Processsing	Jan 1 2010
Inventory	Jan 1 2010
Sales Order Processing	Jan 1 2010
Bank Reconciliation	Jan 1 2010

Analysis Cubes requires all GL transaction history to be included in the data warehouse for accurate calculation of beginning balance forward records for Balance Sheet accounts. Unless you have an extremely compelling reason for doing otherwise, accept the default recommendation for cutoff dates so that all data will be included.

6. On the **Analysis Services Cube Setup** window, select **Analysis Services SQL Server name for the cubes** defined in the preinstallation checklist and choose which cubes to install.
7. Click on **Next**.



Analysis Cubes for Microsoft Dynamics GP (Microsoft SQL Server 2012)

Analysis Services Cube Setup

Select the server and the database name. Select the cubes to be installed in this database.

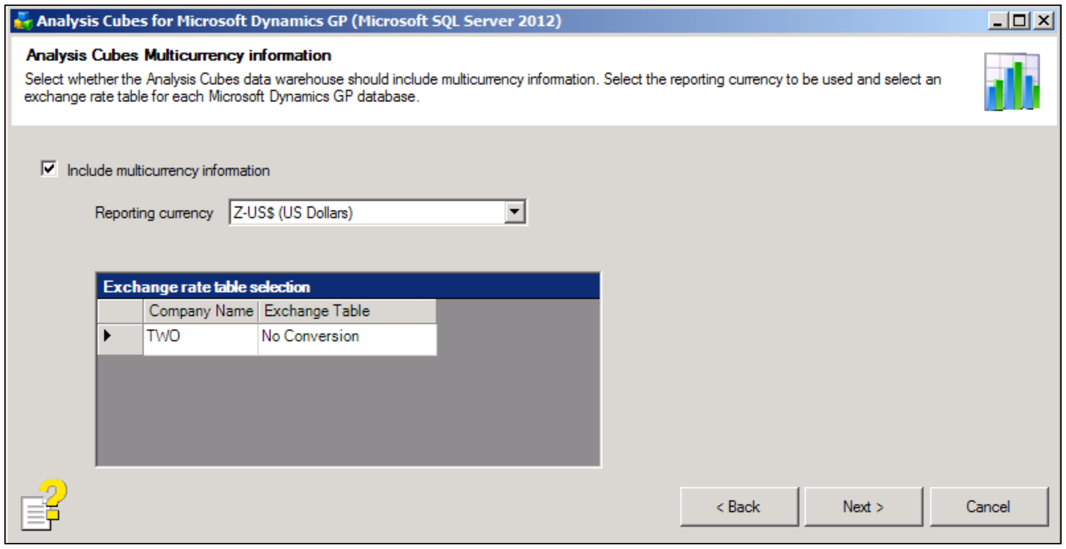
Analysis Services SQL Server name for the cubes: BI64

Analysis Services database name for the cubes: Dynamics GP Analysis Cubes

Select the cubes to be installed into the specified Analysis Services database:

- ✓ Financials Cube
- ✓ Inventory Cube
- ✓ Receivables Cube
- ✓ Payables Cube
- ✓ Purchasing Cube
- ✓ Sales Cube
- ✓ Analytical Accounting Cube

8. On the **Analysis Cubes Multicurrency Information** window, if there is a need to use a reporting currency for multicurrency, check the **to Include multicurrency information** box. Choose **Reporting currency**, and then which **Exchange table** to use.
9. Click on **Next**.


Analysis Cubes for Microsoft Dynamics GP (Microsoft SQL Server 2012)

Analysis Cubes Multicurrency information

Select whether the Analysis Cubes data warehouse should include multicurrency information. Select the reporting currency to be used and select an exchange rate table for each Microsoft Dynamics GP database.

☑ Include multicurrency information

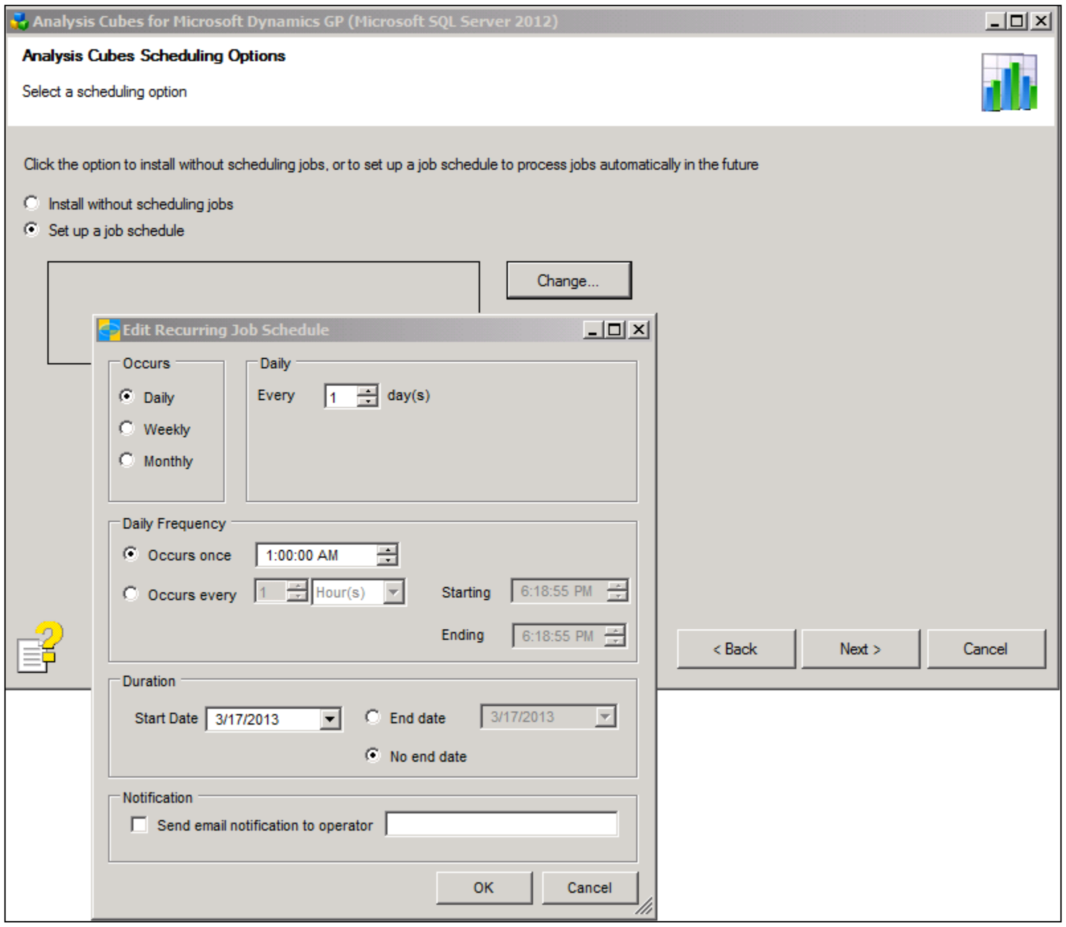
Reporting currency: Z-US$ (US Dollars)

Exchange rate table selection

Company Name	Exchange Table
TWO	No Conversion

This window can be a bit misleading. Choose to include multicurrency information only if the following applies in your environment: one or more GP databases utilize a functional currency different than the currency you wish to use for reporting purposes, or you wish to see options for reporting originating measure amounts along with functional currency amounts. If your GP databases are configured for multicurrency but they all share a functional currency that matches the one in which you wish to report, this box probably does not need to be checked.

10. On the **Analysis Cubes Scheduling Options** window, choose whether to have the wizard create a job schedule or to install without a schedule. The job schedule can be set up at a later time.



Analysis Cubes for Microsoft Dynamics GP (Microsoft SQL Server 2012)

Analysis Cubes Scheduling Options

Select a scheduling option

Click the option to install without scheduling jobs, or to set up a job schedule to process jobs automatically in the future

Install without scheduling jobs [✓ Set up a job schedule]

Edit Recurring Job Schedule

Occurs: [✓ Daily] Weekly Monthly

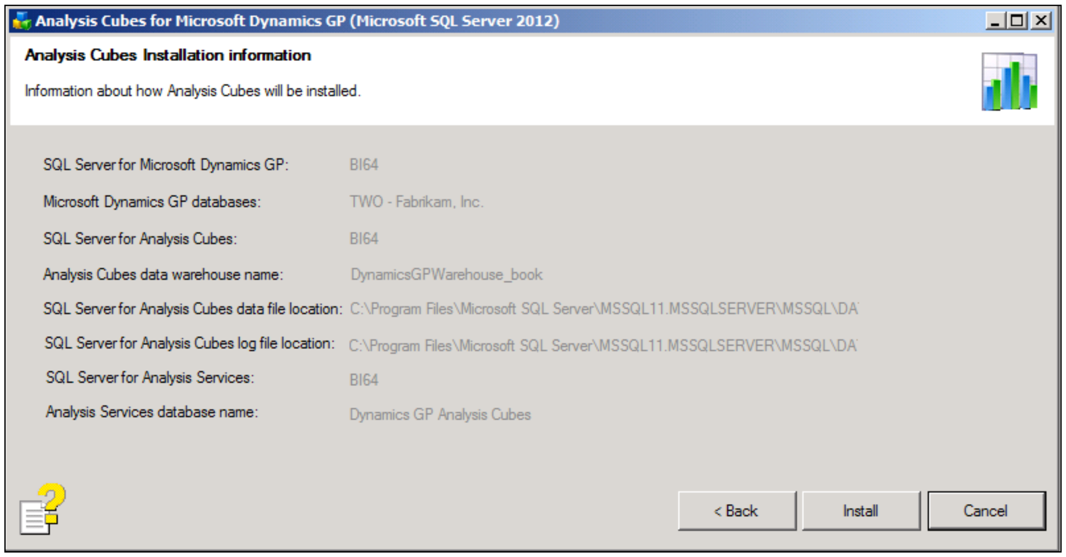
Daily: Every 1 day(s)

Daily Frequency: [✓ Occurs once] 1:00:00 AM; Occurs every 1 Hour(s) Starting 6:18:55 PM Ending 6:18:55 PM

Duration: Start Date 3/17/2013; End date 3/17/2013 [✓ No end date]

Notification: Send email notification to operator

11. On the **Analysis Cubes Installation Information** window, review the information and click on **Install**.


Analysis Cubes for Microsoft Dynamics GP (Microsoft SQL Server 2012)
< Back
Install
Cancel

Analysis Cubes Installation information

Information about how Analysis Cubes will be installed.

SQL Server for Microsoft Dynamics GP:	BI64
Microsoft Dynamics GP databases:	TWO - Fabrikam, Inc.
SQL Server for Analysis Cubes:	BI64
Analysis Cubes data warehouse name:	DynamicsGPWarehouse_book
SQL Server for Analysis Cubes data file location:	C:\Program Files\Microsoft SQL Server\MSSQL11.MSSQLSERVER\MSSQL\DA
SQL Server for Analysis Cubes log file location:	C:\Program Files\Microsoft SQL Server\MSSQL11.MSSQLSERVER\MSSQL\DA
SQL Server for Analysis Services:	BI64
Analysis Services database name:	Dynamics GP Analysis Cubes

12. The **Analysis Cubes Installation Complete** window will provide information about the SQL Server configuration changes that were made.
13. Click on **Exit**.

Though these are fairly detailed installation instructions, it is always recommended to download the latest Analysis Cubes User Guide from the PartnerSource or CustomerSource portals, and check for any updates or other important information for the installation process.

Populating the data warehouse and processing cubes

Once the server configuration is completed and the data warehouse is installed, three master SQL Server Integration Services packages will be installed for each company. These packages are what actually populate the data warehouse database with data from Microsoft Dynamics GP company databases. To run these packages, connect to Integration Services in SQL Server Management Studio, navigate to **Stored Packages | MSDB | DynamicsGPWarehouse**, and right-click on them. The names of the master packages are as follows:

- `DynamicsGP_<source_db>_to_<warehouse_db>_PackageMaster`
- `DynamicsGP_<source_db>_to_<warehouse_db>_Run_GL_Budgets_Master`
- `DynamicsGP_<source_db>_to_<warehouse_db>_Run_AA_Budgets_Master`

Here `<source_db>` is the name of the Dynamics GP company (there will be these three packages for each company you selected to install) and `<warehouse_db>` is the name of the data warehouse database.

> The master package for Analytical Accounting should be run only if Analytical Accounting budget data should be included in the data warehouse.

If during the installation a scheduled job was not created to run the packages, this should be created now.

Once the master packages have been run and the schedule created, the OLAP processing package needs to be run in SQL Server Management Studio to process the cubes. The name of the OLAP processing package is `DynamicsGP_<warehouse_db>_OLAP_DB_<server name>_<analysis services database>`, where `<warehouse_db>` is the name of the data warehouse database. To run this package, connect to Integration Services in SQL Server Management Studio, navigate to **Stored Packages | MSDB | DynamicsGPWarehouse**, right-click on the OLAP package, and click on run package to execute.

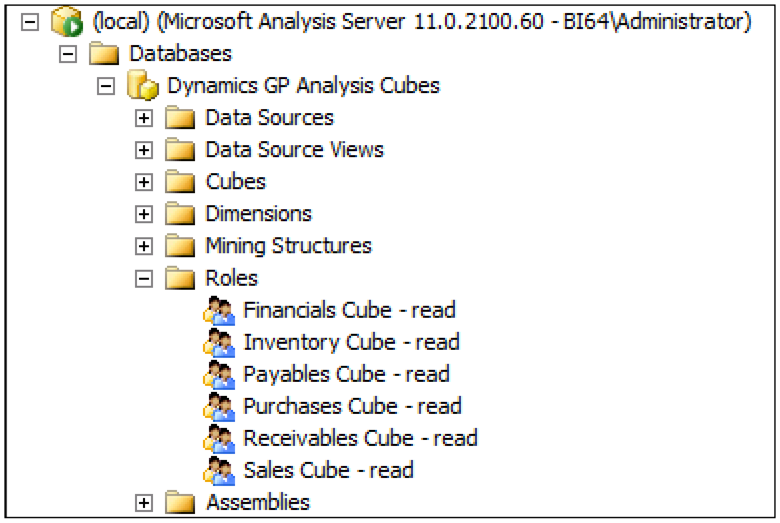
Granting security access to cubes

Security to the cubes is controlled through the use of SSAS security roles. Domain users or groups are assigned to one or more security roles, each of which allows a different level access to a single cube, multiple cubes, or specific objects within a cube. Although custom security roles can be set up in an Analysis Services database, the Analysis Cubes installation actually creates several default security roles for us. For us, then, allowing users to access the cubes is simply a matter of deciding which roles the users should be assigned to.

Accessing and modifying the security roles is fairly straightforward. In this example, we'll use a computer that has SQL Server Management Studio installed on it to accomplish this and perform the following steps:

1. Open SQL Server Management Studio by navigating to **Start | Programs | Microsoft SQL Server | SQL Server Management Studio**.
2. In the **Connect to Server** window, select **Analysis Services** as **Server Type**.
3. In the **Server Name** field, enter the name of the server containing the Analysis Services database.
4. The **Authentication** method should be set to **Windows Authentication**. The Windows account we are logged in as should be a member of the Analysis Services server role to ensure that we can connect properly. For best results, use the account that was used to perform the Analysis Cubes installation.
5. Once the connection is made, expand the **Databases** node.
6. Find and expand the node for the Analysis Services database created by the installation. If the default name was kept, then expand the **Dynamics GP Analysis Cubes** node.
7. Expand the **Roles** node. This will show us a list of the available security roles in this SSAS database.

We can see that a single role exists for each cube that was installed. Essentially, assigning a Windows user or group to a specific security role will allow that user or members of that group read-only access to the cube identified by the role name.



For example, to grant a user access to the **Financials** cube, we can double-click on the role named **Financials Cube – read**. Selecting the **Membership** tab brings us to a window where we can select to add users and groups who should have read-only access to the **Financials** cube. One recommendation that we can make for organizations that will have numerous users accessing the cubes is to utilize Windows groups. If we have ten users who will need access to six cubes, we will be required us to add each user to each separate security role. Six assignments per user multiplied by ten users can quickly add up to a time-consuming chore that is difficult to maintain going forward.

Instead, we recommend setting up a Windows group named **AnalysisCubesReadOnly** or similar. By adding the ten users to this group via Active Directory, we can just add this single group to each of the cube security roles. This greatly reduces the time required to maintain our cube security.

Exploring the Analysis Services database

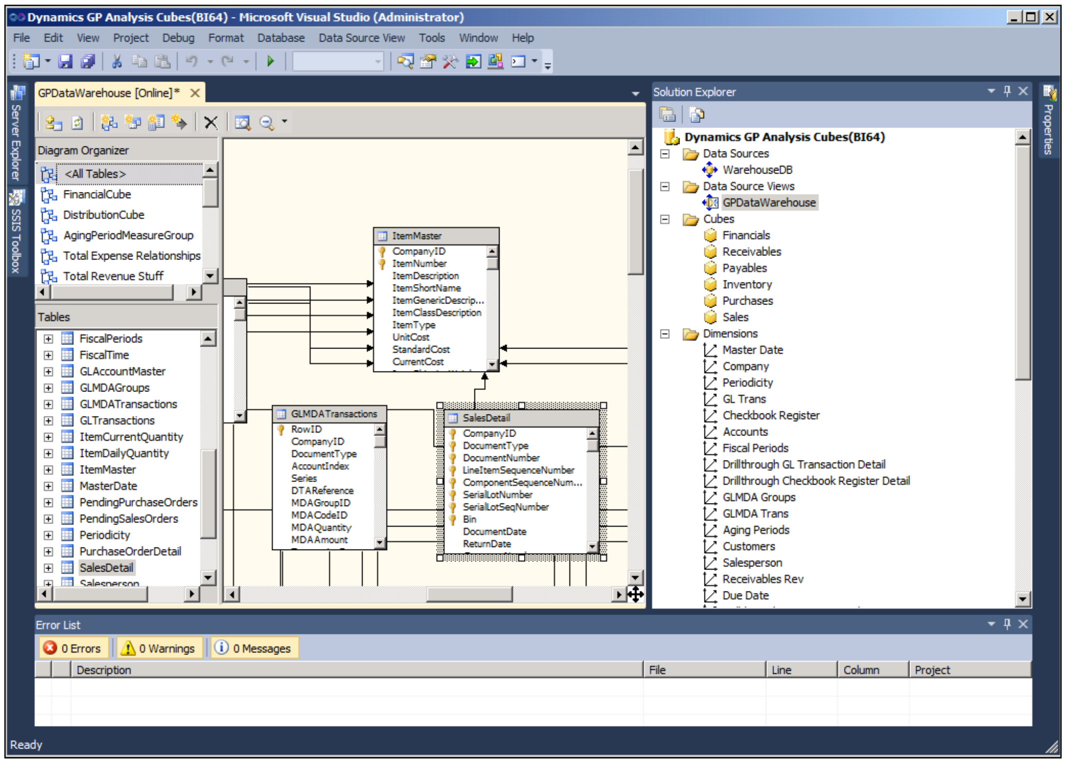
Earlier, we discussed connecting to the Analysis Services database via SQL Server Management Studio. As we saw then, connecting to the SSAS database via SQL Server Management Studio is useful in some situations, such as when we need to add or remove users to security roles or when we need to process the entire database at once. By and large, however, most of our access to the Analysis Services database will occur through SQL Server Data Tools. SQL Server Data Tools replaces the more commonly known **Business Intelligence Development Studio** (**BIDS**). If you are using SQL 2008 or earlier, then use BIDS in place of SQL Server Data Tools for the remainder of this chapter. Each product is installed as a component of the SQL Server installation, and is found under the same `Program Files` folder as SQL Server. Those of us who are familiar with Visual Studio will have no problem using SQL Server Data Tools as it's nothing more than a spinoff of Visual Studio dedicated to designing and maintaining business intelligence projects and environments.

When working with Analysis Services databases in SQL Server Data Tools, we must remember that we are working directly against the server. Any changes made to the database will immediately be saved back to the server. Mistaken changes, such as inadvertently deleting an attribute or deleting a table from the data source view, will have to be restored manually or through an SSAS database backup.

To connect to the default Analysis Services database created by our installation, we can follow these steps:

1. From our local computer, navigate to **Start** | **Programs** | **Microsoft SQL Server 2012** | **SQL Server Data Tools**.
2. Once BIDS loads, navigate to **File** | **Open** | **Analysis Services Database**.
3. In the **Connect to Database** window, enter the server that contains the Analysis Cubes database in the **Server Name** field.
4. Once a connection is made to the server, use the drop-down box for the **Database** field to select the Analysis Services database.
5. Click on **OK** to connect to the database.

Now that we've made a connection to our Analysis Services database, we'll take a look at the various objects that exist in this database.



Understanding objects in the Analysis Services database

By understanding what the various objects in our SSAS database represent, we can begin understanding how we might customize them.

The data source view

The data source view represents the data model upon which the rest of the Analysis Services database is built. Objects such as dimensions and measure groups will be based on tables and structures found in the data source view. In the data source view, we can combine and add multiple data sources from multiple environments into one location.

Dimensions

The key building block of any report that accesses an OLAP database is the attribute. A dimension is a collection of similar attributes that can be grouped together. For example, our installation created a dimension named **Customers** that contains numerous attributes (for example, **Customer Class**, **Corporate Customer Number**, **Salesperson**, and so on) that are associated with a customer in GP. Later, when we discuss reporting against the cubes, we'll use attributes to slice and divide our measures into more meaningful values. Additionally, we will see that attributes that share a common grouping can be grouped into a predefined hierarchy.

Measure groups

Just as dimensions contain attributes, measure groups are a collection of measures. Measures are the different numbers that we will expect to include in our cube reports. For example, in the **Financials** cube, we will see **Debit Amount** and **Credit Amount** as measures that we can include in our report. A measure is almost always useless unless it has been reported against attributes from a related dimension.

Cubes

Cubes are objects responsible for bringing similar measures and dimensions together in the same location. It would not make sense for us to try to report measures from a payables invoice against a list of our customers, so it's highly unlikely that we'd ever see a **Payables** measure group in the same cube as a customer-related dimension.

Security roles

Security roles govern our access to the objects in the Analysis Services database. Without being included in a security role, we will be limited in what we can see when we try to create connections to the Analysis Services database.

Advanced objects (KPIs, Translations, Perspectives, Partitions, and so on)

Numerous other objects (KPIs, Translations, Perspectives, Partitions, and so on) exist in the Analysis Services database, but they are not heavily used by the Analysis Cubes for Excel product offering and will not be covered in much detail here. Nonetheless, these objects can enhance the value found in the products, so we encourage our interested readers to pursue information about these objects in other resources such as *SQL Server Books Online*.

Modifying our Analysis Cubes environment

One of the extraordinary benefits of Analysis Cubes for Excel is its endless possibilities for customization. The Analysis Cubes product gives us a solid base from which we can then extend to include other customizations. Customizations to the cube environment run the gamut from adding new attributes and measures to an existing cube, adding a new cube for another GP module to the existing SSAS database, to even incorporating data from a completely separate application in the same data warehouse as our GP data.

Although some customizations such as incorporating data from applications outside of GP require fairly extensive knowledge of the various SQL Server components involved, we have identified a few simple customizations that we can make to customize our environment. In this section, we'll explore the following customizations:

1. Renaming existing attributes in a dimension.
2. Adding attributes to a dimension.
3. Modifying the **Account Category** hierarchy.
4. Adding GP Budget and Forecast Data to the cubes.

For each of these examples that require modification to the Analysis Services database, we'll use SQL Server Data Tools to make the changes. Refer to the *Exploring the Analysis Services database* section in this chapter for information on connecting to the SSAS database via SQL Server Data Tools or BIDS.

As we modify and customize our environment to suit our needs, it is advisable to keep a list of the changes we make. As we will see later on in the *Considerations for upgrading Analysis Cubes for Excel* section, having a list of changes can greatly speed up the upgrade process.

Additionally, keep in mind that our next chapter will focus on connecting to Analysis Cubes and designing reports. With that in mind, we won't be able to see the real impact of our changes from this chapter until we start designing our cube reports in the next chapter.

Renaming existing attributes in a dimension

Perhaps one of the easiest changes to make that can customize our Analysis Cubes environment for our organization is to rename certain attributes. Probably the best example of this can be found in the **Accounts** dimension of the **Financials** cube. By default, this dimension happens to contain two attributes named similar to the following:

1. **Account Segment 1**
2. **Account Segment 1 Desc**

Now imagine that we have several GP company databases in our data warehouse, each of which shares a similar chart of accounts with three segments in a 3-4-2 format. The first segment represents divisions, the second segment represents natural accounts, and the third segment represents geographical territories. As we will see in the next chapter, Analysis Cubes provides a great way to create reports based on individual segments of our chart of accounts.

For example, if we were to include the **Account Segment 1** attribute in a cube report, we would expect to see individual numbers for example, 100, 200, 300, and so on with each representing a distinct numeric value that can be found in Segment 1 of the chart of accounts in our GP database. Similarly, if we were to include **Account Segment 1 Desc** in our report, we might expect to see values for example, **Administration**, **Accounting**, **Sales**, and so on to reflect the names of various departments that correspond to the numerical values we just listed.

Now, let's go back to the attributes in our **Accounts** dimension. Wouldn't it be better if we could rename our attributes to reflect the fact that segment 1 actually represents divisions in our chart of accounts? Let's rename these to `1 Department Acct` and `1 Department Acct Desc` so that the attribute will be more easily understood by the average cube user.

With SQL Server Data Tools open and connected to our SSAS database, let's perform the following steps to rename our attributes:

1. In **Solution Explorer**, find and double-click on the **Accounts** dimension to open it in the main viewer.
2. In the **Attributes** section of the main viewer, find and right-click on the attribute named **Acct Segment 1** and select **Rename**.
3. Rename the attribute to `1 Department Acct`.
4. Repeat this process for the attribute named **Acct Segment Description 1**, but instead rename it as `1 Department Acct Desc`.
5. Navigate to **File | Save Selected Items** to save these changes on the server.

If desired, this process can be repeated for additional account segments used by our organization. Now, when users browse the **Financials** cube—or any other cube that uses the **Accounts** dimension for that matter—they will see a more intuitive description of the attributes that reflect the first segment of our organization's chart of accounts.

Adding new attributes to a dimension

By default, dimensions in our Analysis Cubes installation contain selected attributes that can be included in our reports. These attributes correspond to a field in the underlying source table of the data warehouse for the selected dimension. For example, the **Customer Name** attribute in the **Customers** dimension corresponds to the **CustomerName** field in the **Customers** table in the data warehouse. Not surprisingly, the **Customers** table is the underlying source table for the **Customers** dimension. If we want to add additional attributes to the **Customers** dimension, we should look at the **Customers** table in the data warehouse to see what fields are available for inclusion. We can also consider tables joined to the **Customers** table in the data source view, but for the sake of simplicity, we'll stick with the **Customers** table in our example.

Now, let's assume that our organization is using the **User Defined Field 1** field on the **Customer** card in GP to store the old customer number from the ERP system that our organization used before making the decision to migrate to GP. It would be helpful if we could include this information in the **Customers** dimension so that it can be used in our cube reports.

Fortunately for us, values from **User Defined Field 1** are already stored in the **Customers** table of the data warehouse as the **UserDef1** field, so it will be easy for us to add this as an attribute to the **Customers** dimension via SQL Server Data Tools as follows:

1. In **Solution Explorer**, find and double-click on the **Customers** dimension to open it in the main viewer.
2. The main viewer is now split into several sections, two of which are named **Attributes** and **Data Source View**. Find and select the **Customers** table in the **Data Source View** pane.
3. Scroll through the **Customers** table until we find the **UserDef1** field.
4. Click-and-drag the **UserDef1** field from the **Data Source View** pane and drop it into the **Attributes** pane.
5. Navigate to **File | Save Selected Items**.

6. When prompted with a message that the following objects will also be saved, click on **OK**. Note that doing this will render those SSAS objects inaccessible to cube users until they have been reprocessed.
7. We should then be asked if we want to reprocess the objects that have been affected by this change. Assuming we are able to devote server resources to this process, select **Yes**.
8. Select **Run** when the **Process Objects** window appears. This will reprocess the objects with recent changes and add our new field to the **Customers** dimension.

Once our database has finished processing, we should be able to browse the **Customers** dimension in a cube report to see our newly added attribute. Adding a new attribute to a dimension is a good way to customize the cubes to fit the unique reporting needs of our organization.

Modifying the Account Category hierarchy by editing the GLAccountCategory table

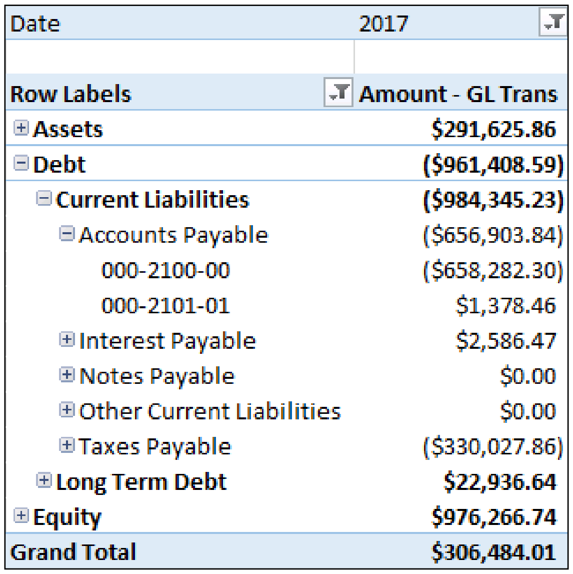
Most Dynamics GP users are familiar with the **Account Category** field. It is that nice, little, required field on the **Account Maintenance** window that many users even wonder why it is even there. For Dynamics GP purposes, it really does not provide much functionality except for placement on out of the box Dynamics GP financial reports, which most users don't use in the first place. However, in the data warehouse, **Account Category** takes on much more meaning and can let us really drill down and organize our data.

In cubes such as the **Financials** cube, we will see an **Accounts** dimension that contains a hierarchy named **Accounts by Category**. This hierarchy combines four related attributes that, together, provide an incredibly useful way of summarizing individual account balances with other similar accounts. These attributes are as follows:

- **Account Main Category**
- **Account Broad Category**
- **Account Category**
- **Account No**

Only the data from the **Account Category** and **Account No** attributes exists in GP. The other two attributes, **Account Main Category** and **Account Broad Category**, are set at the data warehouse level. We can assign different Broad and Main categories to the Account Categories hierarchy from GP to create these natural hierarchies.

For example, suppose we have several payables accounts such as **Accounts Payable**, **Taxes Payable**, and **Interest Payable**. We can assign these Account Categories to a broad category named **Current Liabilities**. In turn, we can assign the **Current Liabilities** broad category to an **Account Main Category** named **Debt**. As we will see when we begin browsing the cube, this creates a hierarchy that looks like the following screenshot when browsing the cube from Excel:



Date	2017
Row Labels	Amount - GL Trans
⊞ Assets	$291,625.86
⊟ Debt	($961,408.59)
⊟ Current Liabilities	($984,345.23)
⊟ Accounts Payable	($656,903.84)
000-2100-00	($658,282.30)
000-2101-01	$1,378.46
⊞ Interest Payable	$2,586.47
⊞ Notes Payable	$0.00
⊞ Other Current Liabilities	$0.00
⊞ Taxes Payable	($330,027.86)
⊞ Long Term Debt	$22,936.64
⊞ Equity	$976,266.74
Grand Total	$306,484.01

Rather than view the accounts under **Accounts Payable** individually or the accounts assigned to Account Categories in the **Current Liabilities** Broad Category, we can summarize these related accounts into higher level attributes that make it easier for us to make sense of our data. Additionally, if we have a multicompany environment with different charts of accounts in each company, we can use this hierarchy to assist with its reporting.

One suggestion for those using Analysis Cubes in multicompany environments is prior to installing and loading Analysis Cubes data for the first time, consider standardizing Account Categories across all companies. Each company database should use the same Account Categories, and these Account Categories should be listed in the same order in the **Account Category Setup** window inside GP (for example, if **Accounts Receivable** is number three in this window for one company database, every other company database should list **Accounts Receivable** as third in the list of account categories). Doing this will also standardize account categories in our data warehouse, allowing us to effectively use this hierarchy for multicompany reporting.

So, how to do we modify this **Accounts by Category** hierarchy? This can be done after the first initial data warehouse data load has occurred. If we use SQL Management Studio to query the **GLAccountCategory** table in the data warehouse, we will see a list of account categories from GP. Additionally, we will see columns such as **AccountBroadCategory** and **AccountMainCategory**. By using the T-SQL UPDATE statements to modify these columns for each account category listed in the table, we can control how the hierarchy will appear in our cube.

In the most recent screenshot, we demonstrated how the **Accounts Payable** account category from GP can be made to roll up into the **Current Liabilities** and **Debt** broad and main categories. Having an Account Category named **Accounts Payable** in your own environment is made possible by updating the **Accounts Payable** record in the **GLAccountCategory** table, and setting the **AccountBroadCategory** field to **Current Liabilities** and the **AccountMainCategory** field to **Debt**.

Also in this table, be sure to take note of the **NativeSign** field. This handy little feature allows us to control how account balances will appear in our **Financials** cube. By default, the **Financials** cube comes with several measures that show us account balances, two of which are utilized more often than most, as follows:

- **Amount**
- **Signed Amount**

The **Amount** measure is calculated as debit minus credit. For accounts that typically carry a debit balance, these amounts will show as positive, and for accounts that carry a credit balance, they will usually display as a negative. But, for financial reporting purposes (and to keep our executive team from any unwanted panic attacks), we may prefer that some of our normal credit balance accounts, such as **Revenue**, be displayed as a positive value in cube reports. This is where the **NativeSign** field and **Signed Amount** measure comes into play.

By setting the **NativeSign** field for the **Revenue** account category record in the **GLAccountCategory** table to **Credit**, the cube will display account balances for our **Revenue** accounts (assuming they are actually carrying a credit balance) as a positive amount when the **Signed Amount** measure is used. In certain cases, this may be preferred over the **Amount** measure, which would be showing a negative balance for the same accounts.

Any time that changes are made to the **GLAccountCategory** table, it is recommended to run the following SQL script against the data warehouse to clear out the **GLAccountMaster** and **GLAccountCategory** tables in the data warehouse (make a backup of the data warehouse first.):

```
UPDATE LastUpdated
SET [DateUpdated] = '01/01/1900',
  [LastRow] = 0,
  [TempLastRow] = 0
WHERE TableName IN ('GLAccountMaster',
  'GLTransactionsOpen','GLTransactionsHistory');

TRUNCATE TABLE GLAccountMaster;
TRUNCATE TABLE GLTransactions;
```

After executing this script, re-run the data load and cube processing job under SQL Server Agent to reload these tables and reprocess the cubes with the most recent changes to the **GLAccountCategory** table.

Adding GP budgets and forecasts to cubes

Another common modification is to include budget information from the budget IDs that have been loaded into our GP company databases. We can choose to have values from one budget ID populate a **Budget Amount** measure and to have values from another budget ID populate a **Forecast Amount** measure. This way, we can compare budget and forecast information with our actual transactions via the **Financials** cube. Of course, in order to do this, we must actually have budget information stored in our GP company databases in the form of budget IDs.

Earlier in this chapter, we discussed the various packages that must be run to populate our data warehouse. For each company database included in our installation, we can expect to see a corresponding SSIS package ending with `Run_GL_Budgets_Master`. This package must be included in our SQL Server Agent job created to load and process the cubes if we want to see budget and forecast values populated in our data warehouse.

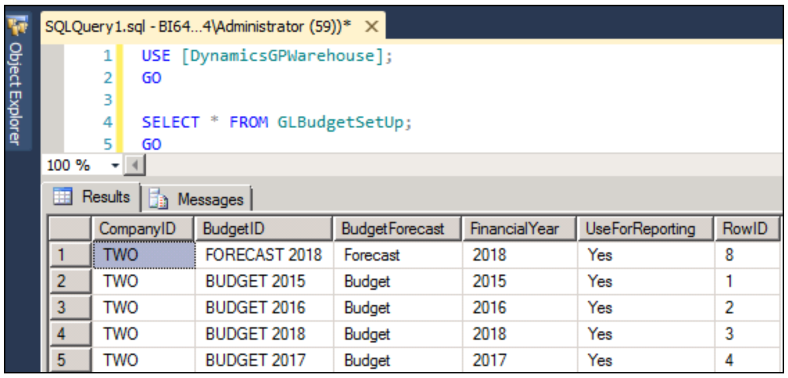
Modifying the GLBudgetSetup table

Once we've confirmed that our budget packages are in place, our next step is to identify the budget IDs that will be loaded in our data warehouse. This is done by editing the `GLBudgetSetup` table data warehouse database. After running the SQL Server Agent job at least once, this table will include a list of budget ID's for each GP company that is included in our installation. Before we modify our table, we should note that only one budget ID can be used for budget numbers per company ID and fiscal year. Likewise, only one budget ID can be used for forecast numbers per company ID and fiscal year.

In order to make this change, first we need to update the **BudgetForecast** column in the `GLBudgetSetup` table with the string value of **Budget** for each **BudgetID** that should contribute to the budget numbers for a particular company ID and fiscal year. Likewise, we need to update the **Budget Forecast** column in the same table with the string value of **Forecast** for each **BudgetID** that should contribute to the budget numbers for a particular company ID and fiscal year.

Additionally, we need to update the **UseForReporting** column in the same table with the string value of **Yes**. This will tell the budget packages that data from this budget should be included in the data warehouse during the next data load.

Our next screenshot shows an example of what the **GLBudgetSetup** table might look like in a sample Fabrikam environment. We can see that four budgets have been identified and set to use for reporting. Additionally, we see that a budget ID named `FORECAST 2018` has also been added to GP, and it has been set to populate the **Forecast** measure in the data warehouse.



```
USE [DynamicsGPWarehouse];
GO

SELECT * FROM GLBudgetSetUp;
GO
```

	CompanyID	BudgetID	BudgetForecast	FinancialYear	UseForReporting	RowID
1	TWO	FORECAST 2018	Forecast	2018	Yes	8
2	TWO	BUDGET 2015	Budget	2015	Yes	1
3	TWO	BUDGET 2016	Budget	2016	Yes	2
4	TWO	BUDGET 2018	Budget	2018	Yes	3
5	TWO	BUDGET 2017	Budget	2017	Yes	4

After making these changes, we have one final step to complete before re-running the cube load and processing the job to populate our budget data in the data warehouse and the cubes.

Adding the Budget measure to the Financials cube

Once we've set a budget and forecast to be used for each company-fiscal year combination, data from these budget IDs will be inserted into the `GLTransactions` table of the data warehouse as **Budget/Forecast** document types. The actual budget and forecast numbers are populated in the **Budget** and **Forecast** fields of this table, respectively. Surprisingly, these fields are not exposed as measures in the **Financials** cube. Before we can view this data in the **Financials** cube, we must add these two fields as measures in the **Financials** cube.

Adding a measure to a cube is similar to adding an attribute to a dimension. We can accomplish this through SQL Server Data Tools as follows:

1. In **Solution Explorer**, find and double-click on the **Financials** cube to open it in the main viewer.
2. Select the **Cube Structure** tab to open the **Financials** cube in a view that contains, among other panes, the **Measures** and **Data Source View** panes.
3. In the **Data Source View** pane, find the `GLTransactions` table. Scroll through this table and find the **Budget** field.
4. Click-and-drag the **Budget** field into the **Measures** pane and drop it. The **Budget** measure should now be added to the GL Trans measure group.
5. Right-click on the newly added **Budget** measure and select **Properties**. In the **Properties** window that appears, find the **FormatString** value. In the drop-down box, select **Currency** as the format for this measure.
6. Repeat this process for the **Forecast** measure by selecting it from the `GLTransactions` table and dragging it to the **Measures** pane. Additionally, follow the step of assigning this newly added measure to the **Currency** format.
7. Navigate to **File | Save Selected Items** to save these changes.

In order to see the effects of our changes, we must run the SQL Server Agent job that loads the data warehouse and reprocesses the cubes. Our budget packages should load budget and forecast data from the selected budget IDs into their respective budget and forecast fields in the `GLTransactions` table. This should be made visible to users in the **Financials** cube through the GL Trans measure group. We'll see how this looks in the next chapter when we discuss cube reporting.

Considerations for upgrading Analysis Cubes for Excel

Unlike many other components of Dynamics GP 2013, Microsoft does not offer an efficient upgrade path for users who must migrate their cubes from a GP 2010 to a GP 2013 environment. As we saw earlier in this chapter, Analysis Cubes for Excel touches many different areas of SQL Server and as such, Microsoft appears to have found it easier to suggest that users simply install the latest version of Analysis Cubes for Excel for the new GP 2013 environment. While this should be relatively easy to do, especially after having read this chapter, it does mean that if you had previously applied customizations to your GP 2010 cube environment, you now have to manually reapply all of those changes to the new cube environment. As we suggested earlier in this chapter, hopefully you kept a list of these changes. If you haven't been maintaining a list of changes, then now is a good time to start documenting those changes so you'll be prepared for the next upgrade.

SSAS does offer some options for deploying Analysis Services databases to other environments, and some of those can come in handy here if you're willing to try the unofficial upgrade policy. Assuming you haven't made changes at the SSIS or data warehouse level, these options can help you transfer all of the changes made at the Analysis Services database level. Before using one of these options, you will need to run through the standard installation for Dynamics GP 2013 in order to create the SSIS packages and the data warehouse pointing to the new GP 2013 database(s).

The most common methods of deploying an SSAS database(s) are as follows:

- Backup and restore the Analysis Services database to your new GP 2013 Analysis Services instance. Use SQL Management Studio or SQL Server Data Tools to access the database and change the data connection to point to the newly installed GP 2013 data warehouse.
- Use SQL Server Data Tools to create a solution file of your GP 2010 SSAS database (create a new `Import from Server (Multidimensional and Data Mining)` project). From within SQL Server Data Tools, change the **Deployment** properties, then use the **Build Solution** and **Deploy Solution** menu options to deploy the database against the GP 2013 SSAS instance.
- Use SQL Server Data Tools to create a solution file of your GP 2010 SSAS database. Build the solution and then use **Analysis Services Deployment Wizard** to deploy the resulting `.asdatabase` file to the SSAS instance of your choice.

While each of these options has its own way of getting the job done, remember, they will only deploy changes made at the SSAS database level. If you've added new columns or tables to the data warehouse and customized existing or new SSIS packages to support loading data to these tables, you'll have to manually move those as well.

One easy way to tell if your Analysis Services database is in sync with the data warehouse is to try refreshing the **Data Source View** (**DSV**) from within SQL Server Data Tools while connected to the SSAS database. This will match the table definitions that are part of the SSAS DSV with the table definitions in the data warehouse. If any tables or columns are missing, you'll quickly be notified of these changes.

The fact that Analysis Cubes for Excel touches so many components of SQL Server makes the process of upgrading cubes from prior versions a bit more cumbersome than other components of GP. Remember to install the latest version of the cubes for GP 2013 and create backups of all components (that is, the data warehouse, SSIS packages and the SSAS database) to protect yourself in the event you need a restore point while attempting to port the changes from one environment to another.

Summary

Because Analysis Cubes for Excel is such a highly customizable product with several different components that must be maintained on the backend, we have divided our discussion on Analysis Cubes into two chapters. This chapter covers the numerous aspects related to installing and setting up Analysis Cubes for first time usage. Then we made some minor modifications to the Analysis Services database making the cubes more useful to our organization. In doing so, our readers have gained a measure of just how far we can expand the usefulness of this reporting tool. It does take some time to learn the various inner workings of SQL to the point that we can make customizations. But the benefits of being able to do this are tremendous. As we move forward into the future of increased data consumption and storage, designing and maintaining OLAP data mining tools will be critical in the reporting toolset.

In the next chapter, we'll reap the benefits of this groundwork. We will see how our customizations look through the use of one of the best OLAP viewers out there, Microsoft Excel 2013. We will explore how to use Microsoft Excel 2013's robust PivotTable functionality to create a connection and browse the cubes. Then, we take a look at what it takes to build an Excel-based dashboard using Excel cube formulas. These dashboards offer a portal through which we can view data that we can refresh, through multiple cubes in a single place. If we are looking to impress our organization's executive team, building colorful and engaging dashboards will be a real asset.

7
Utilizing Analysis Cubes for Excel for Dynamic Reporting

We are heading into the final chapters of this book, with only two reporting tools left in our toolset and we have saved the best ones for last. Until now, we have covered reporting tools that allow us to extract data sets from Dynamics GP or write reports by accessing the Dynamics GP data directly. In all of these tools, any type of summation or grouping that needed to be done, we had to create ourselves and know what tables to join together and where to find this data. In the previous chapter, we learned that Analysis Cubes for Excel is an out of the box data warehouse that does all of this summation and grouping for us.

Now that we have learned the backbone of the Analysis Cubes for Excel product, we will explore how to connect to this data and create reports with every accountant's favorite tool, Microsoft Excel.

In this chapter, we'll discuss the following topics:

- Creating a connection to individual cubes via Microsoft Excel
- Exploring the Excel PivotTable and Analysis Cubes interface
- How to create ad-hoc reports using Analysis Cubes
- Using Excel's CUBE formulas to build static reports and dashboards
- PivotTable functionality found in Excel 2010 and Excel 2013

Let's get started with this incredibly exciting reporting tool.

Using an OLAP viewer to connect to the SSAS database

Once we have deployed our Analysis Cubes, in order to actually start writing our reports, we need to make a connection to our Analysis Cubes database. There are many OLAP viewers on the market as well as on the Internet, available for download. Anyone familiar with Microsoft SQL Server Management Studio can also connect directly to the Analysis Services database by connecting to the SSAS engine. From here, SQL Management Studio can be used as the OLAP browser.

However, in our opinion Microsoft's Analysis Cubes product was clearly designed with Excel in mind, so let's get started with exploring the cubes through Microsoft Excel.

Creating a connection to cubes

When creating a connection to the cubes, an individual connection must be created for each cube. For example, if we wanted to get information out of our **Financials** cube and our **Sales** cube, we would need to create two connections in order to accomplish this.

Older versions of GP offered a **Create PivotTable** window that could be used from within the Dynamics GP application, to create Excel worksheets with PivotTable reports using data from our cubes. However, there were downsides to using the **Create PivotTable** window as follows:

- It required a Dynamics GP license, as a user must be logged into Dynamics GP in order to use the window
- It did not let us preview the resulting PivotTable

The preferred method, as it was then, is to simply use Microsoft Excel to make the necessary connection to the cubes, as this allows us to preview the results as we are building our PivotTable.

Creating a new connection

In our examples, we will be using Microsoft Excel 2013. To create a connection to the cubes, we open Excel and browse to our **Data** tab. From the ribbon bar, click on **From Other Sources**, and select **From Analysis Services**. This opens **Data Connection Wizard**. In the server name field, enter the name of the database server where the data warehouse database resides, and enter the login information.

Once logged in, we will need to choose the database that contains our data; if we accepted our default data warehouse name during installation, this database will be **Dynamics GP Analysis Cubes**. We are also provided with a list of the cubes that we can select from. Click on **Financials** and then click on **Next**.

> Two main reasons exist for why the **Financials** cube—or any cube, for that matter—does not appear in our drop down: our login information has not been granted security permissions to the cube or the cube has not been fully processed. Both of these topics are covered in more detail in *Chapter 6, Designing Your Analysis Cubes for the Excel Environment*.

The next step in the wizard is to name our connection. By default, the wizard will use `<ServerName><Data Warehouse Name><Cube Name>.odc` as the name of the connection. The key point to note in this step is the **Friendly Name** field. This will also default to the preceding nomenclature. It is a good idea to change this to something much simpler (such as `GP Financials Cube`), as it will make it much easier to use in cube formulas. We can also set a location for where the **ODC** (**Office Data Connection**) file will be stored. We will discuss the differences in storing locally or on a shared network location later in this section.

Once the wizard is completed, we will be prompted with the **Import Data** window where we can decide whether to go ahead and create a **PivotTable Report**, **PivotChart**, and **PivotTable Report**, or **Only Create the Connection**.

Storing connection files locally or on a network share

As with many other reporting tools we have discussed, we always have the decision as to whether to save our reports locally or on some network share where multiple users can access the report. Our PivotTable reports and dashboards with Analysis Cubes for Excel are no different. We can choose to save our Excel spreadsheets using either method, and we have provided some insight into each.

Storing on a network share through the use of mapped drive letters or the **UNC** (**Universal Naming Convention**), allows for any user given the proper permissions to use the report. This makes sharing the report easy. This method also makes backup much easier, as the connections and the reports themselves can easily be centrally managed.

Storing locally on a user's machine can also be an acceptable method of storage; however, in order to share a report or connection, users will need to *repoint* their data connection to the data connection on their local workstation. This makes managing the connections and backing them up much more difficult. In the next section, we'll cover how to repoint PivotTables and workbooks to a different data source.

Repointing to a different data source

Depending on our situation, there are two methods for repointing to a new data source or a new data source location, as follows:

- **Change the Data Source for a single PivotTable**
- **Change the Data Source for an entire Workbook**

Our choice of method will usually depend on the number and nature of the PivotTables that we have stored in our Excel workbook.

Changing the data source for a single PivotTable

If we find the need to change our data source for a single PivotTable, we can perform the following steps:

1. Select the PivotTable that needs to be changed. Selecting the PivotTable adds the **PivotTable Tools** options to the ribbon bar.
2. Select **Analyze**, then **Change Data Source** under the **Data** tab.
3. The **Change PivotTable Data Source** window will open. We will see our current connection name, which will be the friendly name defined earlier.
4. Click on **Choose Connection**. This will bring up the **Existing Connections** window where we can choose any existing connection in the list, or we can browse for more.
5. Once we have selected our new connection, simply click on **Open**, and then click on **OK**.

Changing data sources for an entire workbook

If the situation arises where we need to change our data source for an entire workbook, use the **Data** tab found on the Microsoft Excel ribbon. From there, select **Connections**. This will bring up the **Workbook Connections** window which will allow us to either add a new connection or change the properties of our existing connection, by allowing us to browse out to an ODC file located either on a local machine or a network share.

Using an existing connection to connect to a cube

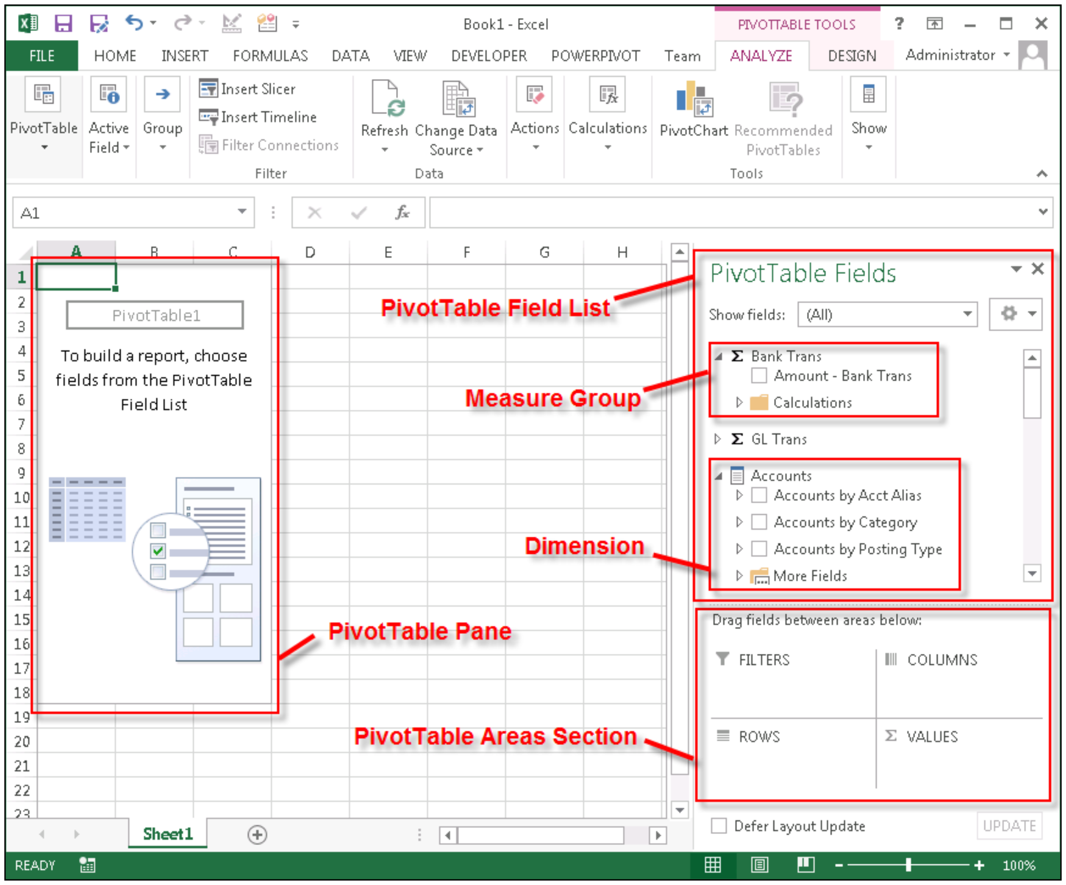
Up until now, we have discussed using Microsoft Excel 2013 to create a data connection to our data warehouse at the time, we are ready to start writing our PivotTable report. One of the many benefits of these data connections is that once they are made, they can be used over and over again. We typically won't see the need to create multiple connections to the **Financials** cube, for instance. To connect to an existing connection, open Microsoft Excel, and select the **Data** tab. Select **Existing Connections** and choose an existing data connection from the available list or browse for more. This will open the **Import Data** dialog where we can choose to create our report.

Another way to accomplish this is to click on the **Insert** tab in Excel, choose PivotTable, and select either of the PivotTables. This will bring up the **Create PivotTable** window where we can select to **Use an External Data Source**, and click on **Choose Connection** to browse the list of available connections, or browse for more.

Excel PivotTable – Analysis Cubes interface

For those of us familiar with Excel PivotTables, this next section will also be familiar. Be sure to pay close attention, however, because we'll point out some of the differences between PivotTables that use a standard dataset—such as a table in the same workbook— versus PivotTables that use Analysis Services as a data source.

The following screenshot shows us some of the features of Excel PivotTables that we will cover in this section:



The PivotTable pane

When we first create a connection to our cube, our Excel worksheet is transformed into an Excel PivotTable. The most immediate indication of this is the PivotTable pane that will appear in the upper left-hand corner of the spreadsheet. This pane is where our report will be built. Right now, it is blank because we have not yet added any measures or attributes to our report.

As we begin to add measures and attributes to our report, the PivotTable pane will expand. Applying filters or refreshing the data over a period of time will cause this data to change, and the PivotTable pane will either expand or contract, depending on how much data is being presented. In short, PivotTables are dynamic, and we can't expect to control how much spreadsheet "real estate" our report will take up.

One thing to keep in mind before we go any further, is that clicking outside of the pane will cause the PivotTable options on the toolbar and the **Field List** to disappear. Don't panic! Just click back on the pane to get it to reappear.

The PivotTable Field List

Elsewhere in our Excel spreadsheet, we see a sidebar titled **PivotTable Fields** or, as it used to be called, the PivotTable Field List. Although our report will be built in the PivotTable pane, most of our interaction with the PivotTable will occur within the context of the Field List. This sidebar is split up into two main components: the **Fields Section** and the **Areas Section**.

A button with a gear-shaped symbol at the upper-right hand corner of the Field List allows us to control the layout of the Field List. The default view, **Fields Section and Areas Section Stacked**, will be used throughout the rest of this chapter and is probably the best view for preserving extra room in our spreadsheet. We definitely encourage those with a larger monitor to try the next view on the list—this view splits the Field List down the middle and places the Fields and Areas Sections side-by-side.

The Fields Section

In the Fields Section, we see a list of groupings. Unlike a PivotTable connected to a standard dataset, the Field List for a PivotTable connected to an Analysis Services database is separated into measure groups and dimensions. In the previous chapter, we learned about these components and how they form part of the design of the cube. Now, let's consider them in light of how they are used in the reporting context.

Measure groups

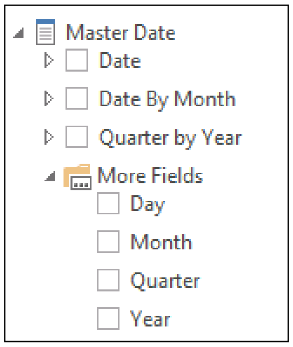
Measure groups are always found at the top of the PivotTable Field List. They are easily identified by the sigma (**Σ**) symbol. Within each measure group, we can expect to see several related measures. We will use these measures to generate the numerical values in our report.

Nearly every cube will contain a special measure group titled **Values**. Measures in this grouping are generally calculated measures. In other words, these measures do not exist in a data warehouse table as a specific field, but they are instead created through the Analysis Services database. For the end user, however, these measures act and function as any other measure in one of the major measure groups.

Dimensions

Measures are pretty important, considering they will comprise the actual values in our reports, but a single number by itself is pretty useless. Instead, we will rely on attributes to slice and dice these numbers into more meaningful measures. Attributes are found on the Field List grouped together by corresponding dimension. Remember from our last chapter that a dimension is a collection of similar attributes. In the Field List, dimensions are identified by a rectangular symbol that looks like a list of attributes.

Depending on the dimension, we may see a plain list of attributes, or we may see a list of hierarchies, followed by a node named **More Fields**. The following screenshot offers an example in the form of the **Master Date** dimension that is present in every cube:

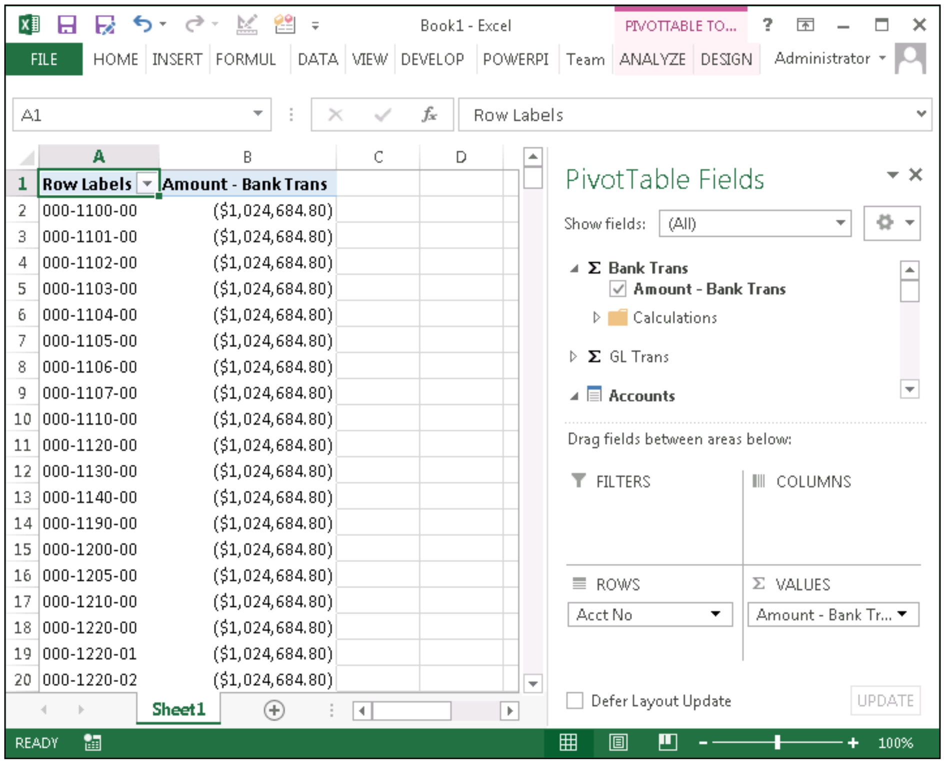


The first three nodes—**Date**, **Date by Month**, and **Quarter by Year**—under the **Master Date** node are known as hierarchies. We can confirm this by the presence of a plus (**+**) sign in front of each node. Hierarchies are groups of attributes that can be ordered in a natural manner. For example, in the **Master Date** dimension, if we expand the **Date** hierarchy node, we can see that it consists of four attributes: **Year**, **Quarter**, **Month**, and **Day**. Hierarchies are useful because they give us a way to easily expand our PivotTable nodes in order to drill down to more detailed data. If we were to add the **Date** hierarchy to our report, we'd have a high level look at all of the years in our data warehouse. If we wanted to drill down to a more detailed level for one of those years, we could select the node for one of those years to drill down to the quarters of that year.

If we only want to see a single attribute, such as **Month** in our report, we can expand the **More Fields** node to find a list of individual attributes. We can select from this list to ensure that only the selected attribute—and not additional attributes, as with a hierarchy—will be used in our report.

Incompatible dimensions and measures

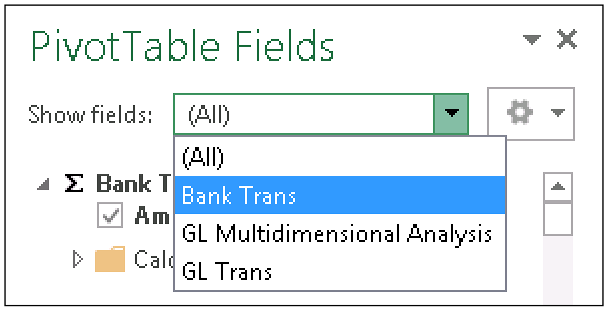
Depending on how the underlying cube is designed, some measures may not be compatible with attributes from certain dimensions. When a measure is combined with an incompatible attribute in a PivotTable, the result is usually fairly obvious, as shown in the following screenshot:



Row Labels	Amount - Bank Trans
000-1100-00	($1,024,684.80)
000-1101-00	($1,024,684.80)
000-1102-00	($1,024,684.80)
000-1103-00	($1,024,684.80)
000-1104-00	($1,024,684.80)
000-1105-00	($1,024,684.80)
000-1106-00	($1,024,684.80)
000-1107-00	($1,024,684.80)
000-1110-00	($1,024,684.80)
000-1120-00	($1,024,684.80)
000-1130-00	($1,024,684.80)
000-1140-00	($1,024,684.80)
000-1190-00	($1,024,684.80)
000-1200-00	($1,024,684.80)
000-1205-00	($1,024,684.80)
000-1210-00	($1,024,684.80)
000-1220-00	($1,024,684.80)
000-1220-01	($1,024,684.80)
000-1220-02	($1,024,684.80)

In the preceding example, the **Acct No** attribute from the **Accounts** dimension is paired with a measure from the **Bank Trans** measure group. Because this report is attempting to combine attributes from the General Ledger module with measures from the Bank Reconciliation module, the two measures are incompatible and the result is the same measure repeating itself over and over.

One way to ensure that only compatible dimensions and measures are used in a report is to take advantage of the drop-down box at the top of the PivotTable Field List. This drop-down box is found underneath the **Show Fields:** text, as seen in the following screenshot from the **Financials** cube:



As we can see from the previous screenshot, the selections in the drop-down box correspond to the two major measure groups available in the cube. Selecting **Bank Trans** will update the Field List, so that only measures and dimensions compatible with this measure group will be displayed. In fact, with **Bank Trans** selected, the **GL Trans** measure group as well as the **Accounts** and **GL Trans** dimensions will be hidden from the Field List. Only the **Bank Trans** measure group will be available. We can be assured that no matter what attribute we select for our slicing and dicing, it will be compatible with the **Bank Trans** measure group.

The Areas Section

The Areas Section of the Field List is divided into four areas, each controlling a different aspect of our PivotTable report. In order to build our PivotTable, attributes and measures must be selected or dragged from the Fields Section into one of these four areas. As we add and remove these from the Areas Section, the PivotTable pane will constantly adjust to reflect these new selections.

Before we get down to the business of creating a specific report in our PivotTable, let's take a look at the different units that comprise the Areas Section as follows:

- **Values**: The **Values** section is where measures from the Field List should be placed. It's easy to make this connection considering the symbol for the **Values** section (that is, the sigma (**Σ**) symbol), is the same as the symbols for the measure groups found in the Field List. As we add measures to the **Values** section, numbers will begin to appear in our PivotTable. These numbers won't mean much until we've added attributes to the other three portions of the Areas Section.

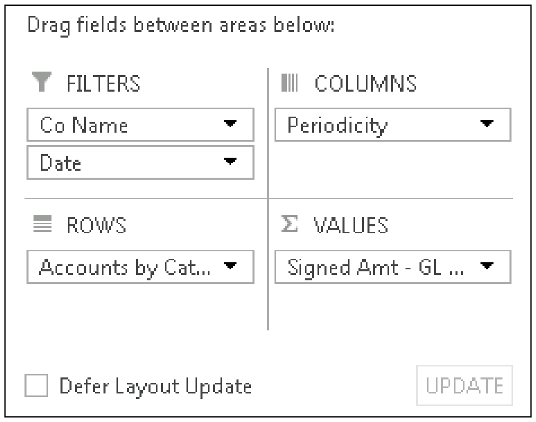
- **Row Labels**: Adding attributes in the **Row Labels** section will cause the measures that we just added to the **Values** section to be sliced in a vertical direction. The distinct records that make up the attribute will become the row labels for our PivotTable report. For example, if we were attempting to create a refreshable Trial Balance report, we would add the **Account Number** attribute to the **Row Labels** section, so that we can see a row of account numbers down the left-hand side of our report.
- **Column Labels**: Just like with the **Row Labels** section, we can add attributes to the **Column Labels** section in order to slice our values. However, adding attributes to the **Column Labels** section will cause the distinct records of this attribute to be arrayed across the top of the report. Add attributes to the **Column Labels** section for effective side-by-side reporting. One popular attribute for the **Column Labels** section is **CompanyID,** from the **Company** dimension. In a multicompany environment, this allows us to quickly and easily compare the same values across multiple company databases.
- **Report Filter**: The last area in which we can add an attribute from the Field List is the **Report Filter** section. Adding an attribute here will cause that attribute to appear in the upper left-hand corner of our PivotTable report. We must then click on this filter in the actual report to select the values on which we want to filter the data in our report. This is incredibly useful for restricting the amount of data that actually appears in our report.

Now that we understand the basics of the PivotTable in Excel, let's try creating some reports of our own.

Creating ad-hoc reports

With our basic knowledge of PivotTable, creating reports is actually quite simple and can be very fun once we get started. It's an experience similar to opening presents on Christmas morning; you don't really know how cool your report will be until you start playing with the different measures and matching them with different dimensions. An easy report to start with and get an idea of the power of Analysis cubes for Excel, is a simple one that uses the **Accounts by Category** dimension, the **Signed Amount – GL Trans** measure, and the **Periodicity** dimension as the columns.

To accomplish this, we will use what we have learned in earlier sections to create a new data connection or use an existing connection. Once we have made our data connection, we will drag our measures and dimensions into our Areas Section. We can see from the following screenshot how each of these is placed in our Areas Section to create the report:


Drag fields between areas below:

FILTERS: Co Name, Date

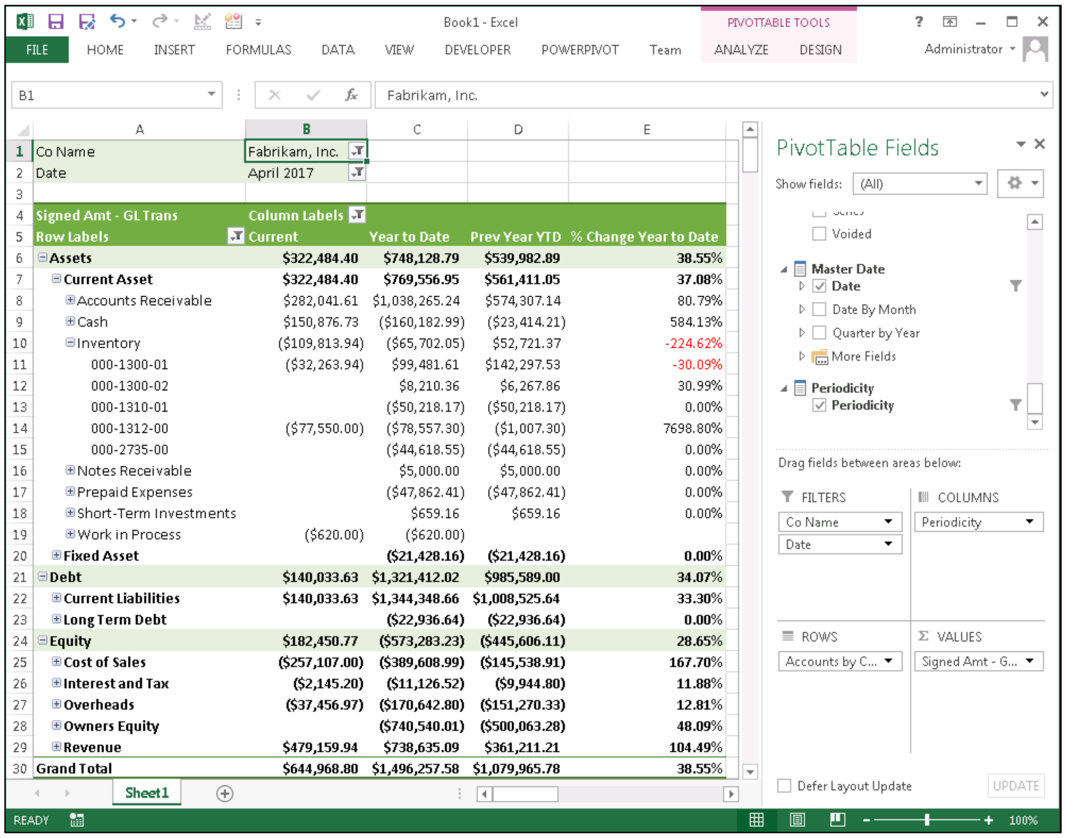
COLUMNS: Periodicity

ROWS: Accounts by Cat...

VALUES: Signed Amt - GL ...

The **Periodicity** dimension is a rather interesting dimension, as it includes columns such as current amount, period to date amount, previous year period to date percent changes, and more, all in one dimension. This is very useful as it saves time having to try to write complicated formulas to calculate all of these amounts. The **Periodicity** dimension is pretty useless unless we define a date from which all calculations should originate. In fact, the underlying calculations require that whenever we use the **Periodicity** dimension, we also use the **Date** hierarchy from the **Master Date** dimension. Using other attributes or hierarchies from the **Date** dimension will not work with **Periodicity**.

In the following screenshot, we can see our final product, and the impressive thing is that it only took us a few minutes to build. It would be difficult to build this report, this fast, in Management Reporter or SSRS.



B1: Fabrikam, Inc.

Co Name	Fabrikam, Inc.			
Date	April 2017			
Signed Amt - GL Trans	**Column Labels**			
Row Labels	**Current**	**Year to Date**	**Prev Year YTD**	**% Change Year to Date**
⊟Assets	**$322,484.40**	**$748,128.79**	**$539,982.89**	**38.55%**
⊟Current Asset	**$322,484.40**	**$769,556.95**	**$561,411.05**	**37.08%**
⊞Accounts Receivable	$282,041.61	$1,038,265.24	$574,307.14	80.79%
⊞Cash	$150,876.73	($160,182.99)	($23,414.21)	584.13%
⊟Inventory	($109,813.94)	($65,702.05)	$52,721.37	-224.62%
000-1300-01	($32,263.94)	$99,481.61	$142,297.53	-30.09%
000-1300-02		$8,210.36	$6,267.86	30.99%
000-1310-01		($50,218.17)	($50,218.17)	0.00%
000-1312-00	($77,550.00)	($78,557.30)	($1,007.30)	7698.80%
000-2735-00		($44,618.55)	($44,618.55)	0.00%
⊞Notes Receivable		$5,000.00	$5,000.00	0.00%
⊞Prepaid Expenses		($47,862.41)	($47,862.41)	0.00%
⊞Short-Term Investments		$659.16	$659.16	0.00%
⊞Work in Process	($620.00)	($620.00)		
⊞Fixed Asset		**($21,428.16)**	**($21,428.16)**	**0.00%**
⊟Debt	**$140,033.63**	**$1,321,412.02**	**$985,589.00**	**34.07%**
⊞Current Liabilities	**$140,033.63**	**$1,344,348.66**	**$1,008,525.64**	**33.30%**
⊞Long Term Debt		**($22,936.64)**	**($22,936.64)**	**0.00%**
⊟Equity	**$182,450.77**	**($573,283.23)**	**($445,606.11)**	**28.65%**
⊞Cost of Sales	**($257,107.00)**	**($389,608.99)**	**($145,538.91)**	**167.70%**
⊞Interest and Tax	**($2,145.20)**	**($11,126.52)**	**($9,944.80)**	**11.88%**
⊞Overheads	**($37,456.97)**	**($170,642.80)**	**($151,270.33)**	**12.81%**
⊞Owners Equity		**($740,540.01)**	**($500,063.28)**	**48.09%**
⊞Revenue	**$479,159.94**	**$738,635.09**	**$361,211.21**	**104.49%**
Grand Total	**$644,968.80**	**$1,496,257.58**	**$1,079,965.78**	**38.55%**

PivotTable Fields

Show fields: (All)

Voided

Master Date: ✓ Date; Date By Month; Quarter by Year; More Fields

Periodicity: ✓ Periodicity

FILTERS: Co Name, Date

COLUMNS: Periodicity

ROWS: Accounts by C...

VALUES: Signed Amt - G...

This is just the beginning. Wait until we see how we can combine multiple PivotTable reports and charts to create the beginning stages of a dashboard.

Using PivotTable design features to change report appearances

It's usually pretty obvious when a co-worker uses a default Microsoft Office template to produce a presentation. Don't be one of those people. Take the extra time to utilize the easy-to-use PivotTable design features, to improve the look and feel of your report. All of these design features can be accessed from the **PivotTable Tools | Design** and **Analyze** tabs. Let's take a look at some of these features.

Changing the Report Layout

On the **PivotTable Tools | Design** tab, we can control different aspects of the **Report Layout**. The following options are available when working with the Layout section of the ribbon bar:

- **Subtotals**: For some reports, we may not want to see subtotals for every grouping. We can use the following options to control when subtotals appear in our PivotTable:
    - **Do Not Show Subtotals**
    - **Show All Subtotals at the Bottom of the Group**
    - **Show All Subtotals at the Top of the Group**
    - **Include Filtered Items in Totals**
- **Grand Totals**: They are the totals for all of the groups found at the very bottom of the PivotTable, or at the far right of the PivotTable. Like subtotals, we can control whether or not grand totals appear in our PivotTable by using the following options:
    - **Off For Rows and Columns**
    - **On For Rows and Columns**
    - **On For Rows Only**
    - **On For Columns Only**

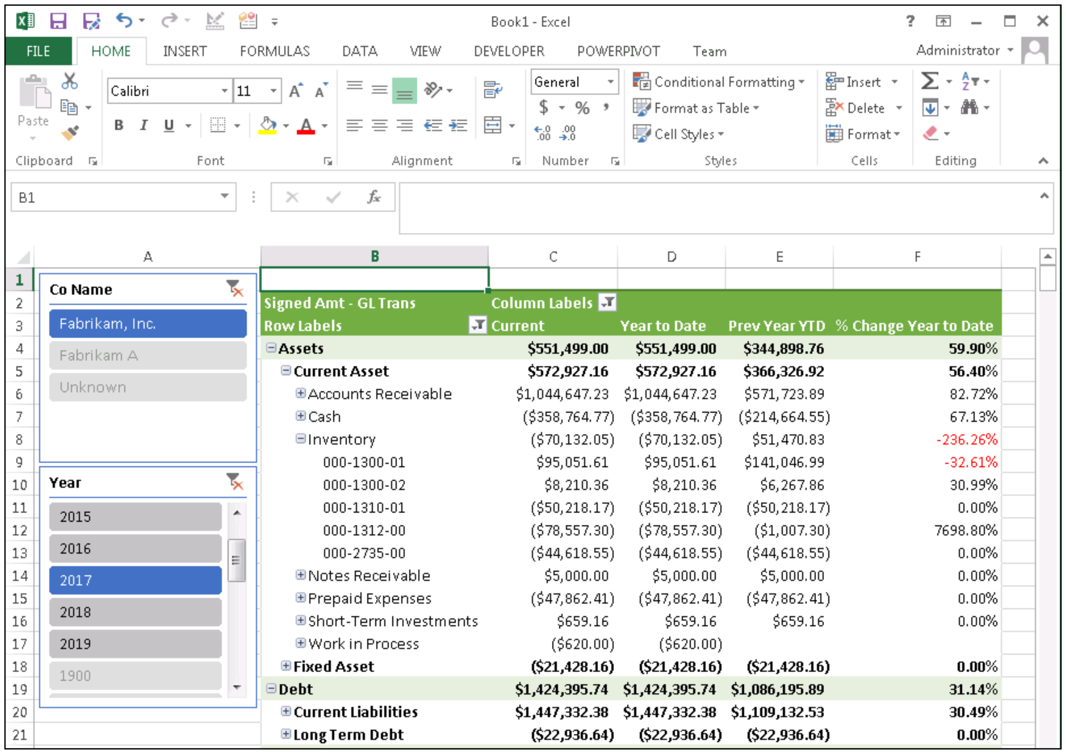
- **Report Layout**: As the name implies, we can use the options found here to control the way our PivotTable appears in our Excel spreadsheet with the following options:
    - **Show in Compact Form**: This is the default view for newly created PivotTables. It's best used for larger reports when lots of data needs to be consolidated into a smaller, more condensed report.
    - **Show in Outline Form**: Depending on the report, this report layout can appear a bit messy. Usually, turning off subtotals can help make this report layout a bit cleaner.
    - **Show in Tabular Form**: This layout is very similar to what many users are familiar with from SmartLists. As with the outline form, it might help to turn off subtotals to make this a cleaner report.
    - **Repeat All Item Labels**: This is new to Excel 2010. By using this option in conjunction with the tabular report layout, we can have item labels repeat down multiple rows, making this even more like a SmartList.
    - **Do Not Repeat Item Labels**: Use this option to turn off the item label functionality if it has been enabled.
- **Blank Rows**: For formatting and readability purposes, it may be helpful to insert a blank row at the end of each grouping. Use these options to control this functionality:
    - **Insert Blank Line after Each Item**
    - **Remove Blank Line after Each Item**

Applying styles to PivotTables

In addition to the Report Layout, we can also apply a wide range of styles to our PivotTable report. Like the Report Layout options, these options are found under **PivotTable Tools | Design** (remember, select the PivotTable pane to see these options on the ribbon bar). We can control such things as the color scheme on the headers, groups and detail lines, whether or not we want banded rows or columns, and whether to include coloring on the row and column headers. By applying such styling to our PivotTables, we can create some really professional looking reports and add some flavor so that every report doesn't look like the same old default template.

Using slicers to filter PivotTable data

Slicers were an awesome addition to Microsoft Excel 2010. In earlier versions of Excel, we were forced to use **Report Filters** in order to filter our data in our PivotTables. Though this worked very well, it was not always easy to know what the current filtering state was, especially if it included multiple items. With the addition of slicers, we have been given a method of providing buttons that we can click on to filter the data. So what is so awesome about these slicers? The buttons light up when we click on them and stay lit, so that at any time we can see the current filtering state even if we are slicing on multiple fields. We can see from the following screenshot, how these slicers work:



Signed Amt - GL Trans	Column Labels			
Row Labels	Current	Year to Date	Prev Year YTD	% Change Year to Date
⊟Assets	$551,499.00	$551,499.00	$344,898.76	59.90%
⊟Current Asset	$572,927.16	$572,927.16	$366,326.92	56.40%
⊞Accounts Receivable	$1,044,647.23	$1,044,647.23	$571,723.89	82.72%
⊞Cash	($358,764.77)	($358,764.77)	($214,664.55)	67.13%
⊟Inventory	($70,132.05)	($70,132.05)	$51,470.83	-236.26%
000-1300-01	$95,051.61	$95,051.61	$141,046.99	-32.61%
000-1300-02	$8,210.36	$8,210.36	$6,267.86	30.99%
000-1310-01	($50,218.17)	($50,218.17)	($50,218.17)	0.00%
000-1312-00	($78,557.30)	($78,557.30)	($1,007.30)	7698.80%
000-2735-00	($44,618.55)	($44,618.55)	($44,618.55)	0.00%
⊞Notes Receivable	$5,000.00	$5,000.00	$5,000.00	0.00%
⊞Prepaid Expenses	($47,862.41)	($47,862.41)	($47,862.41)	0.00%
⊞Short-Term Investments	$659.16	$659.16	$659.16	0.00%
⊞Work in Process	($620.00)	($620.00)		
⊞Fixed Asset	($21,428.16)	($21,428.16)	($21,428.16)	0.00%
⊟Debt	$1,424,395.74	$1,424,395.74	$1,086,195.89	31.14%
⊞Current Liabilities	$1,447,332.38	$1,447,332.38	$1,109,132.53	30.49%
⊞Long Term Debt	($22,936.64)	($22,936.64)	($22,936.64)	0.00%

To add one or more slicers, from our **PivotTable Tools | Analyze** menu, we choose **Insert Slicer** from the ribbon bar. Scroll through the list of available dimensions and select the one to use for our slicer. From there, we simply resize and position the slicer window to where we want it, click on the button for the value we want to filter on, and we are done. We can then add as many more slicers as we want.

Slicers are a great addition and are really great to use as replacements for filters on PivotTables. Though we have described creating a slicer for a single PivotTable, slicers can also be used to connect charts and other PivotTables when creating dashboards, and we can use these slicers to control the filtering of the entire dashboard.

To connect multiple PivotTables from the same cube to a slicer, add the slicer to your report. Then, right-click on the slicer and select **Report Connections**. Here, a list of available PivotTables will be shown, and clicking on the checkbox next to additional PivotTables will connect them to this slicer. We like to create a single tab at the front of the workbook, populate them with slicers, connect them to other PivotTables in our workbook, and use this tab as a one-stop location for changing our entire workbook or dashboard.

Utilizing Excel CUBE formulas

In the last section, we explored creating Analysis Cube reports by using the PivotTable functionality in Excel. For those of us who have used Excel's PivotTable functionality, it is a powerful tool that allows for effortless ad-hoc data analysis. When combined with Analysis Services, PivotTables can provide users with an unparalleled ability to peruse aggregated data and drill down to the underlying data quickly and accurately.

But, for all of its usefulness, some drawbacks do exist for using Excel PivotTable functionality to access data in an Analysis Services database.

First, it's difficult to combine PivotTables in a worksheet containing other PivotTables or Excel objects, such as charts and graphs. The dynamic nature of PivotTables also makes it extremely difficult to control the size of the generated report. Adding and subtracting measures and attributes, or expanding and collapsing hierarchies in the PivotTable, will cause the PivotTable pane to expand or contract within the spreadsheet. If the PivotTable expands into an area of our spreadsheet that already contains data, Excel will prompt us with a message that the PivotTable is about to replace the contents of the existing cell. Obviously, this is not ideal in most cases.

Aside from the potential of erasing sensitive data in the same spreadsheet, another drawback of using PivotTables, is that it's difficult to combine data from a PivotTable with data outside of the PivotTable. This data can exist in another cube, or even be a self-contained Excel spreadsheet that is used on a daily basis. Suppose that we use an Excel PivotTable connected to the **Financials** cube to generate a report showing us up-to-date cash balances for each of our company databases. How do we use data from this PivotTable in conjunction with data that exists outside of this PivotTable? Although an Excel function (`GETPIVOTDATA`) exists for this purpose, it can be an unwieldy tool at times. We need a better way to combine data from the cubes with data from other sources.

Fortunately, since the release of Excel 2007, new formulas have been created to allow users to create connections to a cube in an Analysis Services database or to a PowerPivot data model. By understanding and utilizing these new cube formulas, we can overcome some of the drawbacks presented by using PivotTables as an exclusive means for generating cube data. This is achieved, in part, by locking cube data into specific cells within a spreadsheet. Now, cube data is no longer free to roam about our spreadsheet. Additionally, because this data is no longer found within a PivotTable, our ability to combine this data with data from other sources is greatly improved.

In order to begin using the variety of cube formulas that exist in Excel, we must first understand the two most basic cube formulas. An understanding of these formulas serves as the building block for using the rest of the cube formulas in Excel. After we've used these basic formulas to calculate the year-to-date balance of our operating cash account, we'll go a step further and use more advanced formulas to generate a refreshable top ten list of customers by current aging amount.

Basic cube formulas

Before we look at our basic formulas, keep in mind that each cube formula requires that we first identify a cube connection. To use a connection, it must be an active connection in the existing workbook. Refer to our earlier section for more information on cube connections within an Excel workbook. In the examples in this section, however, we will use a connection that we have named `GP Financials Cube`. Additionally, our samples will utilize the sample company database that can be installed with Dynamics GP.

CUBEMEMBER

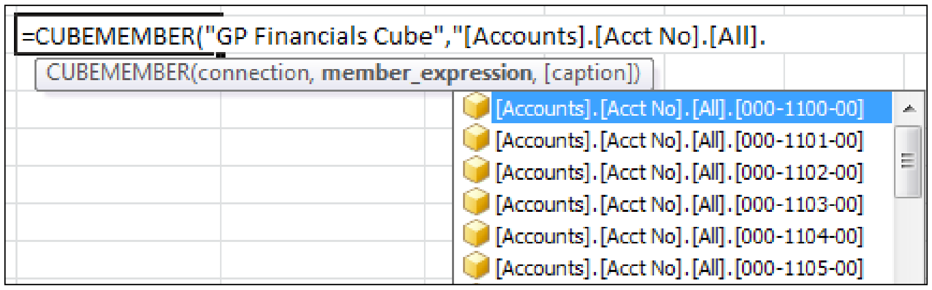
The `CUBEMEMBER` formula is used to introduce attributes into our report. These attributes can then be referenced by `CUBEVALUE` formulas. If you can recall from our previous discussion on PivotTables, we can drag-and-drop an attribute such as **Account Number** from the Fields Section into the **Row Labels** section of our report, and a list of unique account numbers would populate in our PivotTable. What we did not mention at that time, however, is that each individual account number is actually known as a unique member of the **Account Number** attribute in the **Accounts** dimension. With the `CUBEMEMBER` formula, it is not enough that we just reference an attribute; instead, we must actually specify the unique member that we want to see in the active cell.

Say, for example, we want to reference a specific account number in our cell. The `CUBEMEMBER` formula would look something like the following:

```
=CUBEMEMBER("GP Financials Cube","[Accounts].[Acct No].[All].
  [000-1100-00]")
```

If we've entered our formula correctly, we should see **000-1100-00** in the cell containing this formula. Of course, this assumes we are using the sample database; however, if the sample database is not being used, users simply need to substitute a valid account for the final section of the formula. By seeing an account number returned in our field, this indicates that the account number exists in our cube, and any `CUBEVALUE` formula that references this cell will be sliced by this account number.

Our second argument, the member expression, is a text string that requires us to identify the dimension, the attribute, and finally, the specific member that we wish to reference via this formula. Note that each argument is bracketed in quotation marks. These quotation marks are useful because they let Excel know that we're entering a new formula argument, and Excel responds by providing us with a drop-down box from which we can select from a list of available values for this argument. The following screenshot shows an example of the screen tips that are available as we progress through our most recent **CUBEMEMBER** formula:



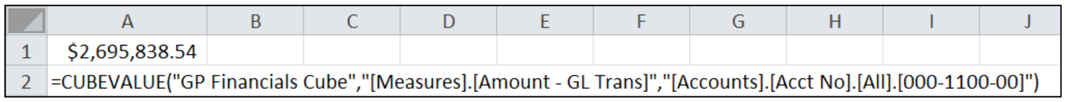
```
=CUBEMEMBER("GP Financials Cube","[Accounts].[Acct No].[All].
CUBEMEMBER(connection, member_expression, [caption])
[Accounts].[Acct No].[All].[000-1100-00]
[Accounts].[Acct No].[All].[000-1101-00]
[Accounts].[Acct No].[All].[000-1102-00]
[Accounts].[Acct No].[All].[000-1103-00]
[Accounts].[Acct No].[All].[000-1104-00]
[Accounts].[Acct No].[All].[000-1105-00]
```

We can use our arrow keys and *TAB* to select one of these values for use in our formula. As we get into more complex cube formulas, this feature will make the process considerably easier for us.

CUBEVALUE

According to the Microsoft Excel **Help** function, the `CUBEVALUE` formula **returns an aggregated value from the cube**. In other words, we will use this formula when we want to return a specific measure in the current cell. For example, we can use the `CUBEVALUE` formula to return the **Amount – GL Trans** value for a specific account using our **Financials** cube as follows:

```
=CUBEVALUE("GP Financials Cube","[Measures].
  [Amount - GL Trans]","[Accounts].[Acct No].[All].[000-1100-00]")
```



	A	B	C	D	E	F	G	H	I	J
1	$2,695,838.54									
2	=CUBEVALUE("GP Financials Cube","[Measures].[Amount - GL Trans]","[Accounts].[Acct No].[All].[000-1100-00]")									

Once it has calculated, the cell containing this formula will display the total sum of debits minus credits for the selected account for all dates and all transactions in our data warehouse. In other words, we will see the value at the intersection of the **Amount – GL Trans** measure and the **000-1100-00** account. Can you picture how this would look in a PivotTable?

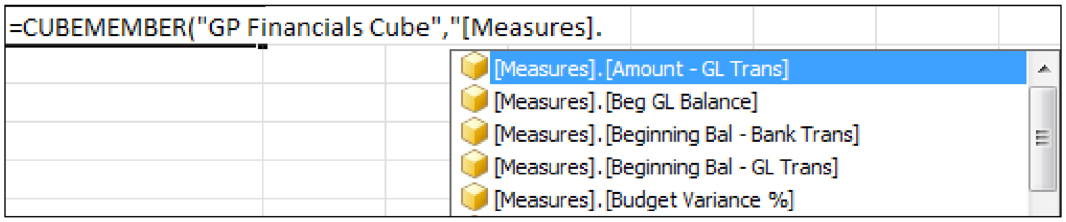
Combining the CUBEVALUE and CUBEMEMBER formulas

Now that we've covered both the `CUBEVALUE` and `CUBEMEMBER` formulas, it's time to combine them. As we've said before, in order to be truly useful, our `CUBEVALUE` formula should reference one or more `CUBEMEMBER` formulas that are used to slice the value into a more meaningful one.

Before we combine the `CUBEVALUE` and `CUBEMEMBER` formulas, let's take a quick step back. In the previous section, we actually inserted our two member references (remember, these were enclosed in square brackets), directly into the `CUBEVALUE` formula. While this is certainly allowed, it's not always best practice. Instead, it's more useful if we separately take these two strings out of our `CUBEVALUE` formula, and insert them into our spreadsheet as `CUBEMEMBER` formulas like the following:

```
=CUBEMEMBER("GP Financials Cube","[Accounts].[Acct No].[All].
  [000-1100-00]")

=CUBEMEMBER("GP Financials Cube","[Measures].[Amount - GL Trans]")
```

Again, we can use the helpful screen tips to make the process of typing formulas easier, as shown in the following screenshot:



```
=CUBEMEMBER("GP Financials Cube","[Measures].
```

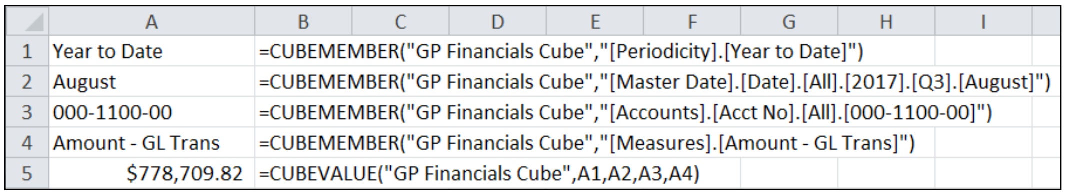
[Measures].[Amount - GL Trans]
[Measures].[Beg GL Balance]
[Measures].[Beginning Bal - Bank Trans]
[Measures].[Beginning Bal - GL Trans]
[Measures].[Budget Variance %]

Now, let's consider an example in which we display the year-to-date balance of our operating cash account in a cell. To begin this example, our two `CUBEMEMBER` formulae from the previous example are found in cells `A3` and `A4`, respectively. We've also taken the liberty to add two additional `CUBEMEMBER` formulas referencing a year-to-date calculation from the **Periodicity** dimension, and a specific date from the **Date** hierarchy of the **Master Date** dimension in cells `A1` and `A2`, respectively.

Our `CUBEVALUE` formula is being entered into cell `A5` as follows:

```
=CUBEVALUE("GP Financials Cube","[Measures].
  [Amount - GL Trans]",A1,A2,A3,A4)
```

The following screenshot shows the final product, with column **A** showing the calculated result of the formula found to its immediate right in column **B** (the formulas in this and other screenshots in this section have been added for emphasis. Your final report will most likely not include these formulas.):



	A	B
1	Year to Date	=CUBEMEMBER("GP Financials Cube","[Periodicity].[Year to Date]")
2	August	=CUBEMEMBER("GP Financials Cube","[Master Date].[Date].[All].[2017].[Q3].[August]")
3	000-1100-00	=CUBEMEMBER("GP Financials Cube","[Accounts].[Acct No].[All].[000-1100-00]")
4	Amount - GL Trans	=CUBEMEMBER("GP Financials Cube","[Measures].[Amount - GL Trans]")
5	$778,709.82	=CUBEVALUE("GP Financials Cube",A1,A2,A3,A4)

It's actually pretty simple. Once we've entered our `CUBEMEMBER` formula(s), all we need to do is add references to the cell containing the unique member in our `CUBEVALUE` formula. If we need to reference multiple members, we can add additional arguments that reference these formulas in their respective cells. This is the same thing as if we were to add additional attributes to the **Row Labels** and **Column Labels** section of our PivotTable, except now, we have greater control in determining where specific values and members should be placed in our spreadsheet.

In the example we've just used, we now have greater flexibility in referencing the year-to-date cash balance as a value in a non-cube report elsewhere in this spreadsheet. Rather than having to check the balance in GP and plug it into our report each time, we want to update the values; we can use the refresh button to refresh our year-to-date balance with the latest numbers from the data warehouse.

This is an extraordinarily simple example, but we're beginning to see some of the value that cube formulas can bring to our everyday reporting needs, especially when those reporting needs exist within an Excel workbook.

Building a top-10 table

In our last section, we explored using the `CUBEMEMBER` and `CUBEVALUE` sections to access data stored in an Analysis Services database without using a PivotTable. This allowed us to lock data into specific cells, and gives us greater flexibility in formatting and controlling the appearance of our spreadsheet.

Now, let's explore two slightly more advanced cube formulas that can help us build a refreshable report showing our top 10 customers based on current aging amount. We'll use the `CUBESET` and `CUBERANKEDMEMBER` formulas to achieve this. Note that we'll use a connection to our Receivables cube, named `GP Receivables Cube` in our example. Again, we will use our sample database for these examples.

CUBESET

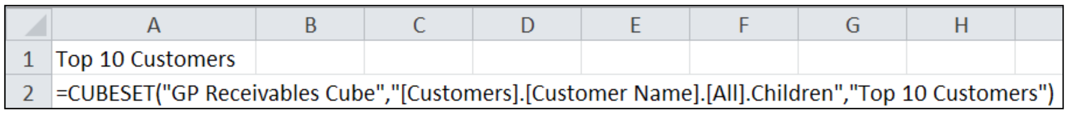
The `CUBESET` formula is used to define a set or range of individual members in a single cell. We can then reference this cell with a `CUBEVALUE` member, just like we would with a `CUBEMEMBER` function. For example, suppose we had three customers that we like to group together for reporting purposes because they are actually child customers of a parent account that does not exist in GP. In our earlier examples, we would have to create three separate `CUBEMEMBER` formulas, each referencing one of the three individual members (or customers). Then, we would have to reference each of these formulas with three separate `CUBEVALUE` formulas and a final summation, in order to find a single value for the parent entity. With `CUBESET`, we only have to define our set once, and then we can use a single formula to represent all three customers at once.

Let's use the `CUBESET` formula to create a set containing all of the customers in our data warehouse as follows:

```
=CUBESET("GP Receivables Cube","[Customers].
  [Customer Name].[All].Children","Top 10 Customers")
```

As usual, our first argument identifies the connection to the cube. Our second argument identifies the members that should make up the set. In this case, we are defining our set as all of the individual members, or children, of the **Customer Name** attribute in the **Customers** dimension. Finally, because the formula in this cell represents so many values, we are using the third argument, the caption, to return a value to our cell that is descriptive of what the cell actually contains.

The following screenshot shows the calculation, plus the final result in an Excel spreadsheet:



	A	B	C	D	E	F	G	H
1	Top 10 Customers							
2	=CUBESET("GP Receivables Cube","[Customers].[Customer Name].[All].Children","Top 10 Customers")							

Now, instead of creating 10 separate CUBEMEMBER formulas for 10 separate customers, we have these 10 customers represented by a single formula in a single cell. Additionally, even as our list of top 10 customers changes over time, this cell will always continue to point to the most current group of top 10 customers.

CUBERANKEDMEMBER

One of the problems with a CUBESET formula is that we can't always tell which individual members comprise the set. For example, suppose we have a CUBESET that represents all of our individual customers from GP. How do we know who those customers are? One of the ways we can extract these customers from the set is through the use of the CUBERANKEDMEMBER formula.

With the CUBERANKEDMEMBER formula, we can return a specific member from the list. Suppose that we wanted to return the individual member with the third highest On Account balance, from a cube set containing a list of customers ranked in order from highest balance to lowest balance? We could use a CUBERANKEDMEMBER to achieve this. It's important to realize that the ranking order is actually specified in the CUBESET formula; we are merely using the CUBERANKEDMEMBER formula to designate which individual member we want returned.

A sample CUBERANKEDMEMBER formula might look as follows:

```
=CUBERANKEDMEMBER("GP Receivables Cube",A1,3)
```

This formula will return the third value in the ranked cube set found in cell `A1`. We have to look in cell `A1` to identify the cube set and the order in which it is ranked.

Let's use our newfound knowledge of `CUBESET` and `CUBERANKEDMEMBER` to create a refreshable table of our top 10 customers by current receivables balance.

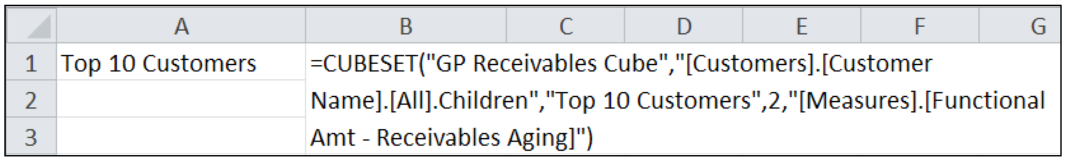
Creating the table

Before we begin, we need to modify the `CUBESET` formula that we created in our earlier example. In that example, we simply created a set containing all of our customers. We did not identify any particular order in which that set should be arranged, and so we will do that now as follows:

```
=CUBESET("GP Receivables Cube","[Customers].
  [Customer Name].[All].Children","Top 10 Customers",
    2,"[Measures].[Functional Amt - Receivables Aging]")
```

This formula adds two additional arguments. The fourth argument tells us the order in which we'd like to sort our set. A value of `2` tells us that we'd like to sort it in descending order (use the screen tips to help you populate this formula.). The fifth and final argument identifies the value that should be used to determine the sort order. In this case, we have selected the `Functional Amt - Receivables Aging` measure, as this reflects the total aging balance from the last time aging was run in GP.

So far, our report should look like the following screenshot (with formulas added for emphasis):

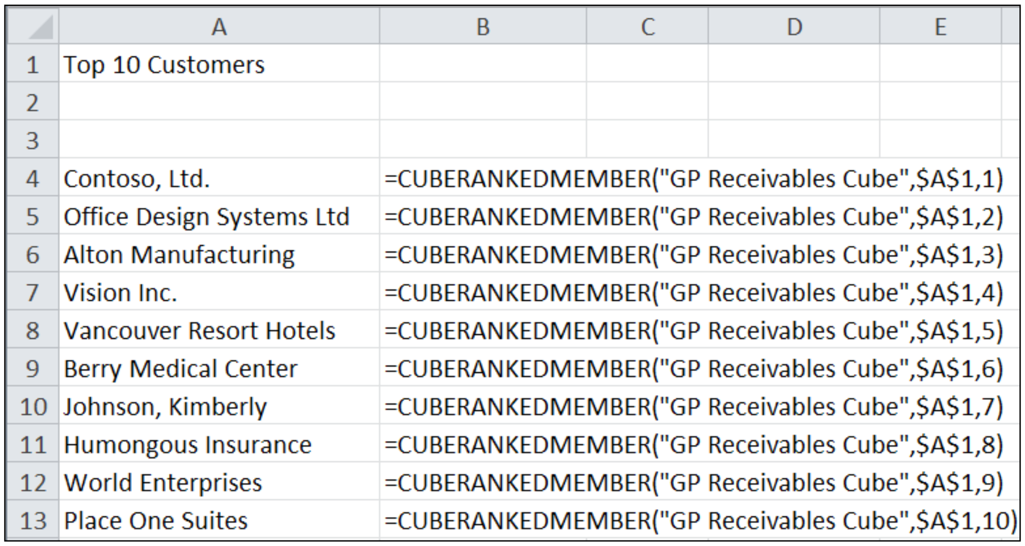


	A	B	C	D	E	F	G
1	Top 10 Customers	=CUBESET("GP Receivables Cube","[Customers].[Customer Name].[All].Children","Top 10 Customers",2,"[Measures].[Functional Amt - Receivables Aging]")					
2							
3							

Next, we need to add some `CUBERANKEDMEMBER` formulas to our report so we can actually identify the top 10 customers in the set. Our first formula will appear as follows:

```
=CUBERANKEDMEMBER("GP Receivables Cube",$A$1,1)
```

This formula references the CUBESET formula that we just entered in cell A1 and tells Excel to return the first customer from that set. We can copy and paste this formula down one cell and change the rank argument to a 2 value in order to return the second customer. Again, we can copy this, change the rank argument to a 3 value to return the third customer, and so on. Eventually, our report should look like the following screenshot:



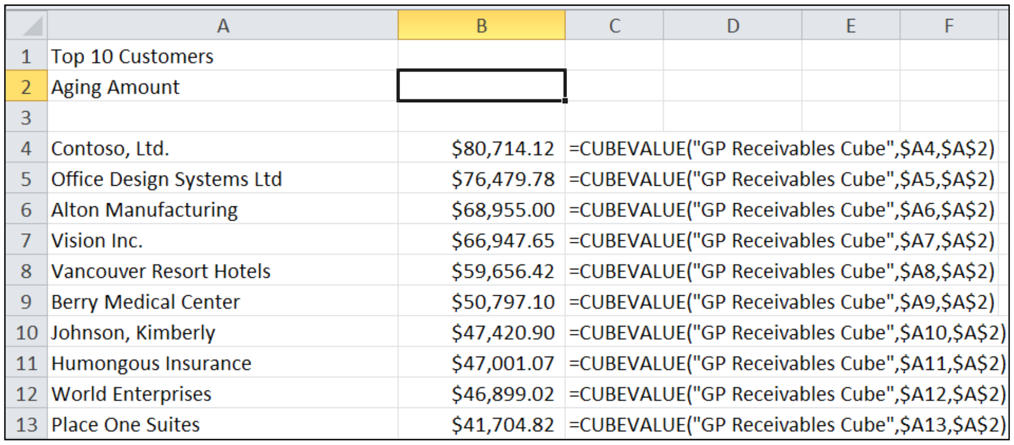
	A	B	C	D	E
1	Top 10 Customers				
2					
3					
4	Contoso, Ltd.	=CUBERANKEDMEMBER("GP Receivables Cube",A1,1)			
5	Office Design Systems Ltd	=CUBERANKEDMEMBER("GP Receivables Cube",A1,2)			
6	Alton Manufacturing	=CUBERANKEDMEMBER("GP Receivables Cube",A1,3)			
7	Vision Inc.	=CUBERANKEDMEMBER("GP Receivables Cube",A1,4)			
8	Vancouver Resort Hotels	=CUBERANKEDMEMBER("GP Receivables Cube",A1,5)			
9	Berry Medical Center	=CUBERANKEDMEMBER("GP Receivables Cube",A1,6)			
10	Johnson, Kimberly	=CUBERANKEDMEMBER("GP Receivables Cube",A1,7)			
11	Humongous Insurance	=CUBERANKEDMEMBER("GP Receivables Cube",A1,8)			
12	World Enterprises	=CUBERANKEDMEMBER("GP Receivables Cube",A1,9)			
13	Place One Suites	=CUBERANKEDMEMBER("GP Receivables Cube",A1,10)			

Notice how the rank number changes in each successive row.

It appears that our report is lacking only one thing: values. While we can already tell that **Contoso, Ltd** is our customer with the highest aging amount, wouldn't it be helpful if we could see the actual dollar value associated with this customer? Absolutely! To do this, we first need to enter a CUBEMEMBER formula in cell B1 that identifies the measure we wish to display, as follows:

```
=CUBEMEMBER("GP Receivables Cube",
  "[Measures].[Functional Amt - Receivables Aging]","Aging Amount")
```

Finally, we need to add CUBEVALUE formulas that reference both CUBERANKEDMEMBER formulas and the CUBEMEMBER formula that identifies the type of measure to be displayed. In the end, our spreadsheet should look as the following screenshot:

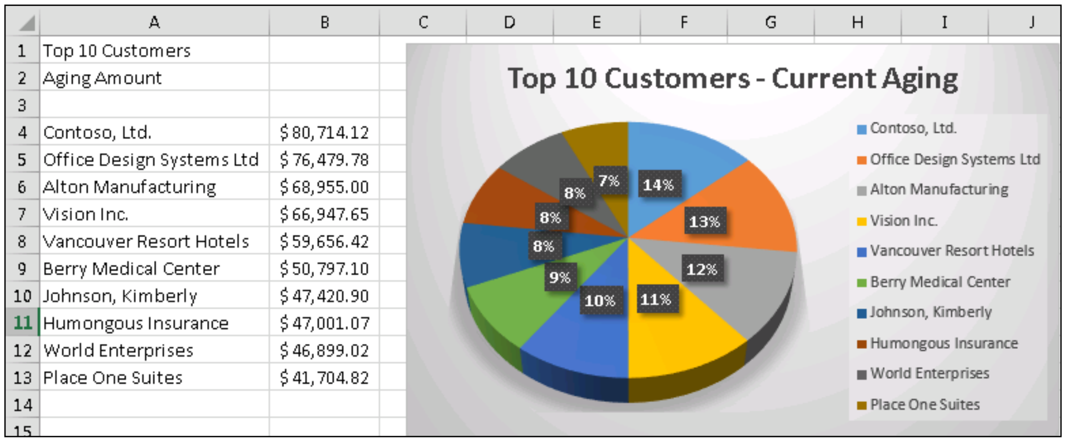


	A	B	C	D	E	F
1	Top 10 Customers					
2	Aging Amount					
3						
4	Contoso, Ltd.	$80,714.12	=CUBEVALUE("GP Receivables Cube",$A4,$A$2)			
5	Office Design Systems Ltd	$76,479.78	=CUBEVALUE("GP Receivables Cube",$A5,$A$2)			
6	Alton Manufacturing	$68,955.00	=CUBEVALUE("GP Receivables Cube",$A6,$A$2)			
7	Vision Inc.	$66,947.65	=CUBEVALUE("GP Receivables Cube",$A7,$A$2)			
8	Vancouver Resort Hotels	$59,656.42	=CUBEVALUE("GP Receivables Cube",$A8,$A$2)			
9	Berry Medical Center	$50,797.10	=CUBEVALUE("GP Receivables Cube",$A9,$A$2)			
10	Johnson, Kimberly	$47,420.90	=CUBEVALUE("GP Receivables Cube",$A10,$A$2)			
11	Humongous Insurance	$47,001.07	=CUBEVALUE("GP Receivables Cube",$A11,$A$2)			
12	World Enterprises	$46,899.02	=CUBEVALUE("GP Receivables Cube",$A12,$A$2)			
13	Place One Suites	$41,704.82	=CUBEVALUE("GP Receivables Cube",$A13,$A$2)			

Congratulations! We now have a refreshable report that will always show us a list of our top 10 customers based on the current aging amount. If we save this spreadsheet, open it at a later date, and navigate to **Data** | **Refresh All**, our report will update to display data from the latest cube load and processing. If a new customer has supplanted **Contoso, Ltd.** as the customer with the highest balance, that customer would then be displayed in cell `A4`.

Adding a chart for visual effect

If we're really feeling creative, we can also add a chart to our report that will refresh and update, based on the contents in the table we've just created. As we refresh the data in our report, the data in the chart will also be refreshed. For example, the following screenshot shows what happens when we add a pie chart to the spreadsheet:



	A	B
1	Top 10 Customers	
2	Aging Amount	
3		
4	Contoso, Ltd.	$ 80,714.12
5	Office Design Systems Ltd	$ 76,479.78
6	Alton Manufacturing	$ 68,955.00
7	Vision Inc.	$ 66,947.65
8	Vancouver Resort Hotels	$ 59,656.42
9	Berry Medical Center	$ 50,797.10
10	Johnson, Kimberly	$ 47,420.90
11	Humongous Insurance	$ 47,001.07
12	World Enterprises	$ 46,899.02
13	Place One Suites	$ 41,704.82
14		
15		

With Microsoft Excel 2013, we can easily insert a graph by selecting our data set, clicking on the **Insert** button in the ribbon bar, and then clicking on the chart of our choice. With just a few simple formatting changes, we can really enhance our spreadsheet with cool graphics that provide valuable, at-a-glance information.

Creating dashboards from Analysis Cubes data – a few thoughts

As we've seen from our previous examples, one of the great benefits of Analysis Cubes is that users can quickly and easily create their own reports from scratch. With the ad-hoc capabilities provided via PivotTable functionality, users can quickly move the building blocks of their report around in a manner that suits them best. If for some reason a report is lost, it's usually not too difficult to recreate that report from scratch.

Most users, however, will identify several key reports and/or metrics of information that they want to refer to on a regular basis. Rather than create these reports from scratch every time they open a new Excel workbook, these users will begin saving these reports in an Excel workbook. Then, anytime they want to refer to that information, they'll open up that workbook, click on **Refresh All**, and be on their way. Unbeknown to some of these users, what they're doing is engaging in a primitive form of "dash boarding".

In short, a dashboard is nothing more than a single, one-stop place for users to quickly review, analyze, and interact with data that is relevant to them. A dashboard for a purchasing agent, for example, may show several metrics and lists related things, such as the value of items on pending purchase orders, orders expected to be received in the next week, items that will require purchase in the next week, and so on. All of this information can be consolidated into bite-sized pieces of information that can then be consolidated into a single spreadsheet. Each morning, when the purchasing agent arrives in the office, he or she can pull up that spreadsheet, click on refresh, and have a quick birds-eye view of the purchasing landscape.

The topic of building effective dashboards is one that can take up several chapters, if not an entire book. We hope that this chapter has given you some thoughts on how you can create a dashboard using nothing more than a single Excel workbook, and a connection to the various cubes provided by Analysis Cubes for Excel. Some dashboards are created by creating a workbook that consists of several useful PivotTables, with one PivotTable per spreadsheet. Other, more advanced dashboards may make use of the cube formulas to create static subsets of data that can be combined into a single spreadsheet. In this manner, we can combine data from multiple cubes and even multiple data sources into a single, easy-to-access place.

Eventually, depending on our organization's environment, these cube reports and dashboards, using Analysis Cubes data as the underlying data source, can be published to an Office SharePoint Server website. This enables an entire team or organization to have access to the same reports and a single version of the truth. Truly, the possibilities become endless with business intelligence tools such as Analysis Cubes and Excel.

The seven default cubes

Now that we've covered the basics of reporting against the cubes, let's make a few comments about each individual cube. When possible, we'll comment on the multiple measure groups that exist in a given cube. Refer to our previous discussion on incompatible measure groups and dimensions to ensure that mismatched measures and attributes do not cause havoc in your own report.

The Financials cube

The **Financials** cube is probably the most useful and, not surprisingly, most popular cube. Data in this cube is pulled from the transaction tables of the General Ledger and Bank Reconciliation modules from GP. Additionally, multidimensional analysis information is included in this cube for those organizations that make use of this tool in GP.

Users should note the presence of an **Amount – GL Trans** measure and a **Signed Amount – GL Trans** measure in the **GL Transactions** measure group. The **Amount – GL Trans** measure is a measure calculated as debits minus credits. The **Signed Amount – GL Trans** measure group takes the **Amount – GL Trans** measure a step further and adjusts the sign of the measure to a plus or minus depending on the native sign of the selected account. The native sign of each account is managed at the account category level via the `GLAccountCategory` table in the data warehouse. For example, under normal circumstances, we might expect an account assigned to the Sales Account Category to have a normal credit balance. If we used the **Amount – GL Trans** measure, this would show up as a negative in our report. In many circumstances, we would rather have this credit balance display as a positive, so we assign a native sign of Credit to the Sales Account Category and use the **Signed Amount – GL Trans** measure to display this value as a positive. In this chapter, we discussed how the native sign functionality can be utilized on the database end to change how these values appear in our reports.

The **Financials** cube is an incredible tool for building instantly refreshable trial balances, balance sheets, and other financial statements. Additionally, by building these reports in PivotTables, we can take advantage of the extra drill through and filtering tools that make these PivotTables so powerful. Although the ability to control formatting and the ordering of accounts is somewhat limited with PivotTables, we have seen some companies make use of the Excel CUBE formulas to overcome these obstacles and create instantly refreshable, professional-looking financial statements.

The Accounts Receivable cube

Data in this cube is pulled from the Receivables Management module from Dynamics GP. As this cube contains this information along with the aging period's aging details, it can be used to create reports such as cross-company aging reports.

Two separate measure groups exist in this cube. One, the **Receivables Aging** measure group, pulls data from the aging related fields in the Customer Master Summary table (RM00103). These fields in GP are updated anytime aging is run against a particular customer, so users should note that the cube does not actually perform the aging routine. Instead, aging-related data in the cube is only up-to-date as of the last time aging was run in GP prior to the cube load and processing job run.

On the other hand, the **Receivables Revenue** measure group contains measures from Open and History transactions and distribution tables in the Receivables module. By including distribution information, we can use this measure group for account-level insights into our Receivables transactions. Apply information, such as Credit Memos applied to an invoice or payments from customers, is not included in this cube.

The Accounts Payable cube

Data in this cube is pulled from the Payables Management module from Dynamics GP. We can create reports such as Total Expenses by Vendor for a period using this cube.

Unlike the **Accounts Receivables** cube, the **Accounts Payables** cube does not include any aging information. Instead, only a single measure group, **Total Expense**, exists to provide us insight into values associated with our Payables transactions. These values are pulled from the transaction and distribution level tables in the Payables Management module, so we are able to gain an account-level perspective of these transactions.

Also, like the **Accounts Receivables** cube, we should not expect to see information related to payments or how they are applied to invoices. Unfortunately, this information is not included in the cube and separate customizations are required to include this sort of information.

The Sales cube

Data in this cube is pulled from the Sales Order Processing module in Dynamics GP. Unlike the **Accounts Receivable** cube, this cube does not include distribution information, so we won't see dimensions such as **Accounts**. This cube does include an **Items** dimension, so we can use it to gain better insights into concepts such as the volume of inventory, items sold to a particular customer, or which items are generating the most revenue in a given period.

Two separate measure groups exist in this cube, and they relate to the stage at which a Sales Transaction exists in GP. The **Pending Sales** dimension pulls data from Open Sales Transaction tables, so generally speaking, we can expect to see unposted sales transactions reflected in this measure group. On the other hand, the **Sales Detail** measure group is comprised of data found in the History transaction tables. We can expect to find posted sales transactions, as well as voided transactions here. Dimensions with similar names to the measure groups (that is, **Pending Sales** and **Sales Detail**) also exist in this cube, so we must be sure not to mix and match incompatible dimensions with the wrong measure group.

The Purchases cube

Data in this cube is pulled from the Purchase Order Processing module in Dynamics GP. Although the **Accounts Payable** cube includes distribution information, this cube does not. Like the **Sales** cube, however, it does include an **Items** dimension, so we can use this cube to find things such as all orders for a particular item, statistics on purchasing volume by item, orders by vendor for a particular period, and more.

Also, like the **Sales** cube, two separate measure groups exist that reflect purchase order data depending on the stage a particular purchase order has reached. The **Pending Purchase Order** measure group reflects all purchase orders that are still open in GP. On the other hand, the **Purchase Order Detail** measure group reflects purchase orders that exist in a received or closed status, as well as those that have been manually moved to history. Recall that in GP, purchase orders must be moved to history via the Completed Purchase Orders routine.

As with the **Sales** cube, dimensions with similar names to the measure groups (that is, **Pending Purchase Order** and **Purchase Order Detail**) also exist in this cube, so we must be sure not to mix and match incompatible dimensions with the wrong measure group.

The Inventory cube

Data in this cube is pulled from the Inventory module in Dynamics GP. Two measures, **Inventory History** and **Inventory On Hand**, exist in this cube. The **Inventory History** measure group can be used along with the **Item Daily Quantity** dimension to provide the cube users insight into the numerous inventory transactions that can exist in GP from modules such as Sales Order Processing, Purchase Order Processing, and Inventory. On the other hand, the **Inventory On Hand** measure group can be used in conjunction with the **Item Current Quantity** dimension to provide users with current inventory information. This sort of information is found in GP windows, such as Item Inquiry and is only current as of the last time the cube job was run.

The **Inventory** cube can be a great way to keep a finger on inventory levels in an organization (or across multiple organizations).

The Analytical Accounting cube

Shortly after the release of GP 2010, Microsoft announced that a new cube, one for Analytical Accounting, had been added to the line-up of cubes available as part of the standard Analysis Cubes for the Excel package. As with other cubes, this cube provides users with the ability to crunch massive amounts of data generated by its respective module, so that users can easily and quickly analyze their data. If your company utilizes analytical accounting, this cube is certainly worth utilizing.

Summary

As we can see, Analysis Cubes for Excel is a very robust reporting tool. It can actually be quite fun to use and gives us the ability to create a wide array of reports that we may have found impossible or extremely difficult to create in some of the other reporting tools available to us. We can take what we have learned in this chapter and go beyond the examples we have provided, and become a rockstar in our organization by providing powerful dashboards for our executive and management teams to understand the health of our company.

If you thought that was interesting, wait until you see the Management Reporter application. In the next chapter, we will be introduced to the Management Reporter by exploring how to install, configure, and set up the product. We will then dive into topics such as understanding the range of building blocks and creating a basic report, row, column, and reporting tree definitions.

8 Designing Financial Reports in Management Reporter

In the next two chapters, we'll explore the Management Reporter application. Management Reporter is a great tool to use when our goal is to enable multiple, non-IT users to create professional-looking financial statements based on source data from our ERP environment. Management Reporter is Microsoft's platform used to create a single performance management solution that can be used across the entire Microsoft Dynamics stack. The release of such a product represents a break from the old way of creating financial reports via Microsoft FRx. Microsoft will more than likely continue to build upon and improve the technology and functionality of Management Reporter.

Because Management Reporter offers direct integration with our source databases, we can expect our reports to display information that is up-to-date as of the moment we generate a particular report. Additionally, as was the case with its predecessor, Microsoft FRx, Management Reporter continues the concept of self-service reporting. Creating reports with Management Reporter is made easier through the use of concepts like the building blocks and an easy-to-use user interface. Such concepts make report creation the domain of the actual users who need the report and not that of a possibly overwhelmed IT department.

Although some initial training may be required to bring our key Report Designers up to speed on the many functions and concepts available in Management Reporter, the end results are well worth it. The ability to create high-quality reports will now be in the hands of the users who interact with and understand the data the most.

Management Reporter offers a tremendous amount of functionality for designing and viewing reports. To this end, we've broken up our coverage of Management Reporter into two chapters:

- In *Chapter 8, Designing Financial Reports in Management Reporter*, we will focus our discussion on the underlying architecture of Management Reporter. Then, we will move to the use of the Report Designer interface and the building blocks to create our reports.
- In *Chapter 9, Viewing Financial Reports in Management Reporter*, we will cover the process of generating and viewing our reports using the Report Viewer, as well as offer some additional commentary on some of the changes between Management Reporter and FRx.

Now that we have an idea of what the next two chapters will cover, let's take a look at what we will specifically cover in this first chapter on Management Reporter:

- Underlying architecture of Management Reporter
- Overview of installation of Management Reporter
- Management Reporter Security
- Navigating the Report Designer interface
- Working with the building blocks
- Improvements in Management Reporter 2012
- Taming the building block sprawl

Before we begin interacting with the tool, let's start by taking a look at the architecture of Management Reporter.

Management Reporter architecture

With the release of Management Reporter, Microsoft has updated its codebase from the one previously used for FRx Financial Reporting. Previous users of FRx will find the logic of Management Reporter very similar to FRx with the new look and feel of other Microsoft applications such as Outlook, Excel, and Word. An understanding of this new architecture is important, as this can drive things such as, hardware and system requirements for the Management Reporter environment.

Management Reporter is built with 64-bit environments in mind and does indeed fully support these environments. The codebase has been updated to the powerful Microsoft .NET Platform and a new C# codebase that provides far superior application stability than its predecessors. In addition to the codebase, Management Reporter uses SQL Server for its data storage, which is a great leap from the days of the Microsoft Access database that FRx used.

With the release of Management Reporter 2012, **Internet Information Services** (**IIS**) is no longer a requirement for installation. In addition, Management Reporter 2012 has added a new way to integrate with Dynamics GP data, which is called the **Data Mart**. The Data Mart creates a summarized copy of our ERP data, which is optimized for financial reports. Due to these improvements, one can expect enhancements in performance, stability, and multiuser capabilities.

Installing and configuring Management Reporter

Before installing Management Reporter, we should first review the various installation options available to us. Before deciding on which option to use, we must first gain an understanding of our customer's environment. Is this a new installation, a server move, or something else? The answers to these questions will aid us in our decision. The following options are available for installing Management Reporter:

- Install the Management Reporter application service and process service and create a new database
- Install the Management Reporter application service and process service and connect to an existing database
- Create the Management Reporter database without installing the application service or the process service
- Install the Management Reporter process service only and connect to an existing database
- Install the Management Reporter application service only and create a new database

The following table summarizes the various installation options for Management Reporter and their impact:

Approach	**Type of Service**	**Database**
Full – New	Application and Process	New
Full – Existing	Application and Process	Existing
Database Only	N/A	New
Process service only	Process	Existing
Application service only	Application	New

After deciding on the install option to use, we will want to follow the step-by-step installation instructions for that option in the installation guide. This installation guide can be downloaded from PartnerSource or CustomerSource or found on the installation media. It is recommended that we create an MR admin user in Active Directory prior to the installation and assign it to the local Administrators group. We should then be sure that we are logged in as this user when we perform the installation.

During the installation, several server components will be installed. These components and their purpose are as follows:

- **Application Service**: This controls access to the data and provides connectivity to the client workstations
- **Process Service**: This is used to generate the reports
- **Management Reporter Database**: This serves as the storage repository for all of the building blocks, available reports, and any historical reports

While we will defer the actual steps of installing the product to the Management Reporter installation guide found on PartnerSource and CustomerSource, we will offer a brief overview of the installation process as well as some detail on post-installation configurations required before using Management Reporter.

Installation overview

Due to information changing as service packs and/or feature packs are released, we feel that using the latest installation guide (check PartnerSource/CustomerSource for updates) is always best practice when performing our installations. The main steps for installation are as follows:

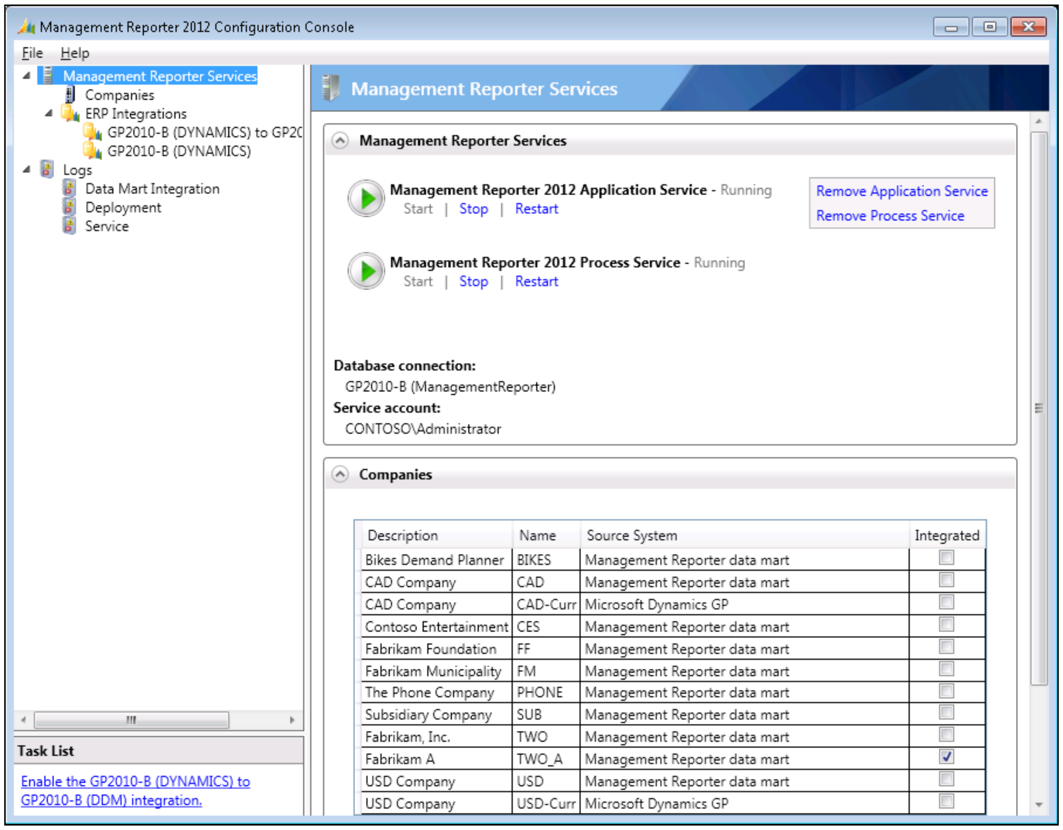
1. Install Management Reporter Server.

2. Use the Configuration Console to configure the Legacy data provider or Data Mart.
3. Install Management Reporter Client.

Let's take a look at some of the additional configurations that need to be completed once the installation is completed.

Configuring Management Reporter Server components

After the installation of the server components is complete, we will use the Configuration Console to configure the Management Reporter services as shown in the following screenshot:



Enable the GP2010-B (DYNAMICS) to GP2010-B (DDM) integration.

Management Reporter Services

Management Reporter 2012 Application Service - Running

Management Reporter 2012 Process Service - Running

Database connection:
GP2010-B (ManagementReporter)
Service account:
CONTOSO\Administrator

Companies

Description	Name	Source System	Integrated
Bikes Demand Planner	BIKES	Management Reporter data mart	☐
CAD Company	CAD	Management Reporter data mart	☐
CAD Company	CAD-Curr	Microsoft Dynamics GP	☐
Contoso Entertainment	CES	Management Reporter data mart	☐
Fabrikam Foundation	FF	Management Reporter data mart	☐
Fabrikam Municipality	FM	Management Reporter data mart	☐
The Phone Company	PHONE	Management Reporter data mart	☐
Subsidiary Company	SUB	Management Reporter data mart	☐
Fabrikam, Inc.	TWO	Management Reporter data mart	☐
Fabrikam A	TWO_A	Management Reporter data mart	☑
USD Company	USD	Management Reporter data mart	☐
USD Company	USD-Curr	Microsoft Dynamics GP	☐

Importing companies

We must first add and configure a company in Management Reporter before it can be used as a data provider. In order to add and configure a company as a data provider, we must complete the following steps:

1. Open **Management Reporter 2012 Configuration Console** by navigating to **Start | Programs | Microsoft DynamicsERP | Management Reporter | Configuration Console**.
2. Expand the **Management Reporter Services** node.
3. Click on **Import** in the display pane.
4. Enter credentials for a user account in the ERP database.
5. Review the list of companies and click on **Import**.

With Management Reporter installed and our data provider configured, we can now turn towards actually using the product.

Registering Management Reporter

Once installation is complete, we need to register Management Reporter. This is done before connecting to our data provider for Dynamics GP. Follow the next set of steps to complete the Management Reporter registration:

1. Open the Report Designer by navigating to **Start | Programs | Microsoft Dynamics ERP | Management Reporter | Report Designer**.
2. Navigate to **Tools | Registration**.
3. Enter the license key.
4. Click on **Validate**.

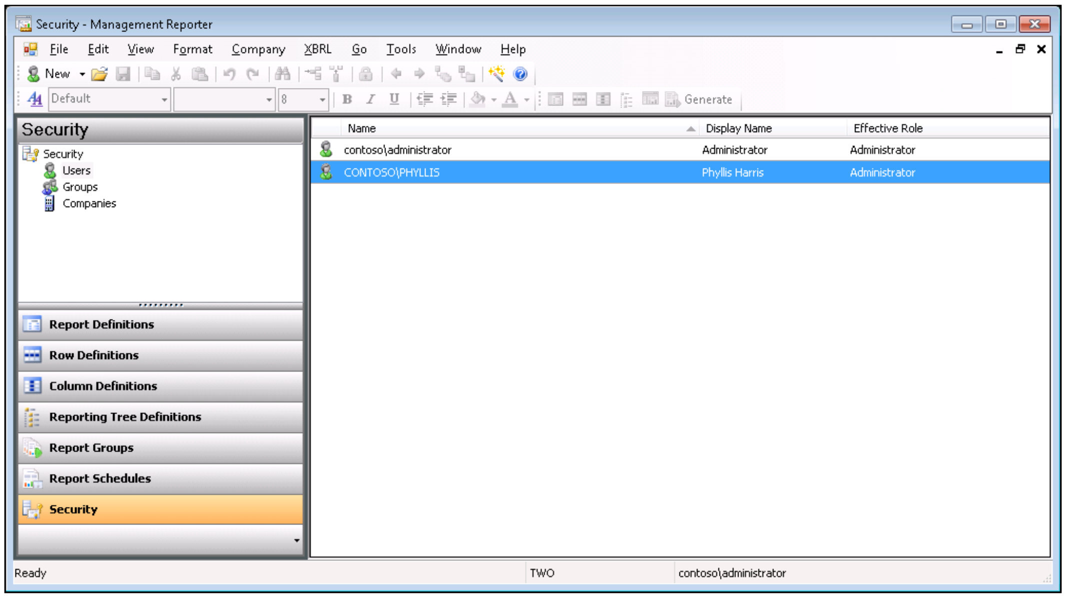
Management Reporter should now be registered for use.

Management Reporter security

Security in Management Reporter consists of **Users**, **Groups**, and **Companies**. Management Reporter works with Active Directory so that the user's domain login will be used to access Management Reporter. Don't forget, Dynamics GP still requires SQL authentication. Consequently, users will still be prompted to enter their SQL or GP login to access the data provider.

Users

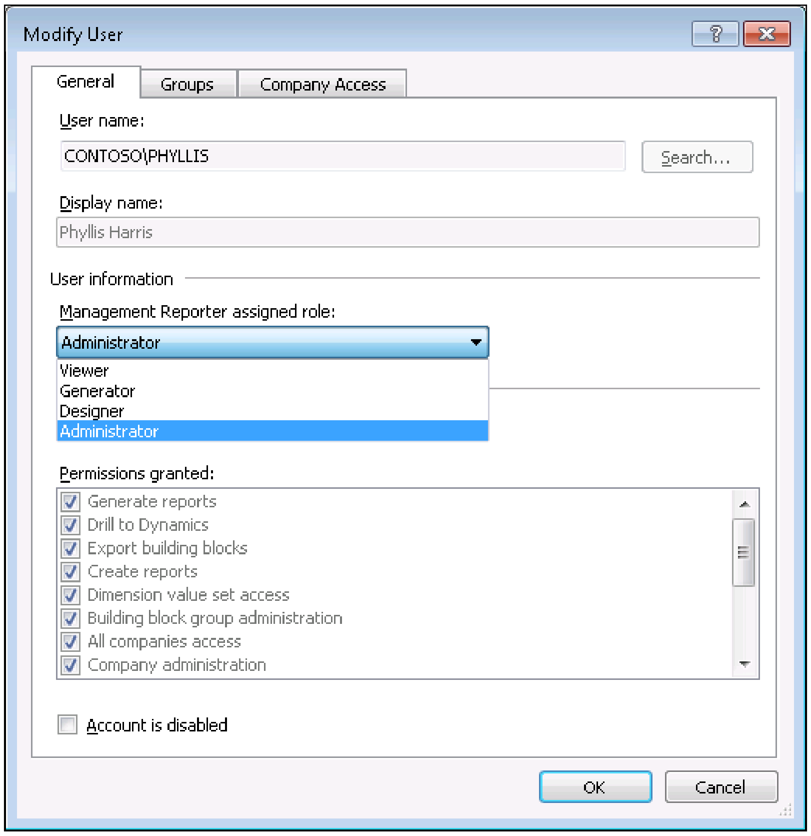
Before users can access Management Reporter, we will need to create their user login in the application. When we are adding users to Management Reporter, we are actually adding the user's existing Active Directory account. To do this, we will select the **Security** button from the left-hand pane in the Report Designer as seen in the following screenshot:



Name	Display Name	Effective Role
contoso\administrator	Administrator	Administrator
CONTOSO\PHYLLIS	Phyllis Harris	Administrator

Ready | TWO | contoso\administrator

Once the **Security** pane is open, we will highlight the **Users** node, and click on **New** from the Menu bar. This will open the **Modify User** window. On the **Modify User** window, we will browse out to the user's account in Active Directory. Once we have located and selected the user's account, then our **Modify User** window should look like the following screenshot:



User name: CONTOSO\PHYLLIS

Display name: Phyllis Harris

Management Reporter assigned role: Administrator

Permissions granted:

- [✓] Generate reports
- [✓] Drill to Dynamics
- [✓] Export building blocks
- [✓] Create reports
- [✓] Dimension value set access
- [✓] Building block group administration
- [✓] All companies access
- [✓] Company administration

[ ] Account is disabled

After selecting our user, we then assign the user to one of the four predefined roles, which are as follows:

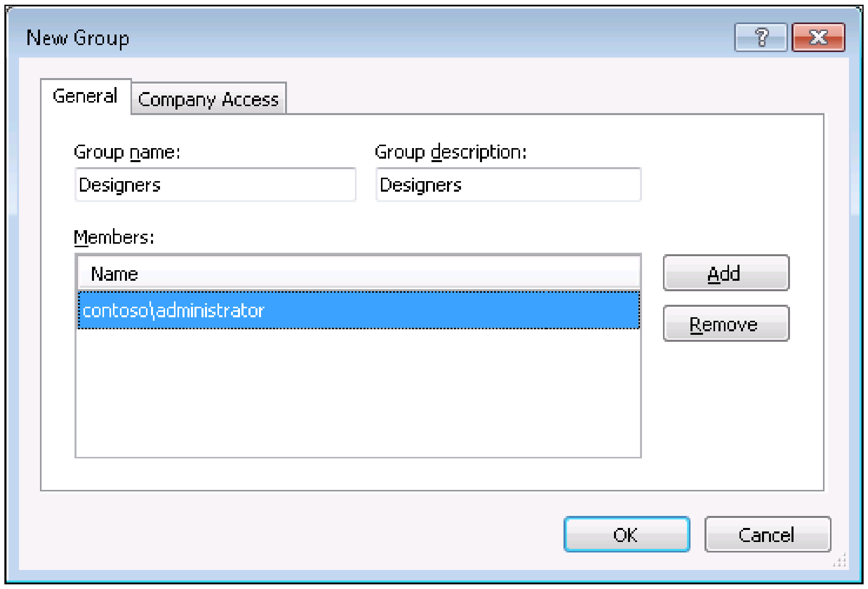
- **Viewer**: This role only allows access to view reports in the Report Viewer, which will be covered in the next chapter.
- **Generator**: This role allows the user to generate and export reports.
- **Designer**: This role allows the same access as **Generator** and adds the ability to create reports, manage dimension sets, and administer building block groups.

- **Administrator**: This role allows full access of Management Reporter including managing security. We want to limit this to maybe one or two users.

> These predefined roles cannot be modified. After assigning the user role, we can also use this window to set a user's account as disabled. In addition, we can assign the user to a group if we wish and also assign company access.

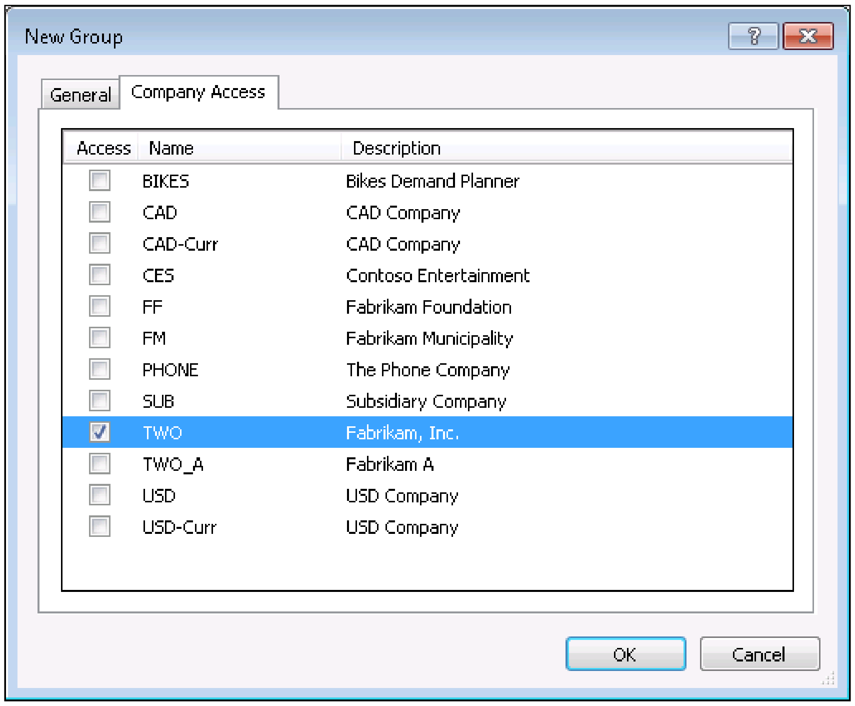
Groups

Groups are optional, but they are a great way to group users together to manage company access and application roles. To add a group, we select **New** from the Menu bar while in the **Security** pane, and select **Group**. We can then give our group a name and description. We can see from the following screenshot that we have created a **Designers** group and added **contoso\administrator** to this group:



Group name: Designers
Group description: Designers
Members:
Name
contoso\administrator

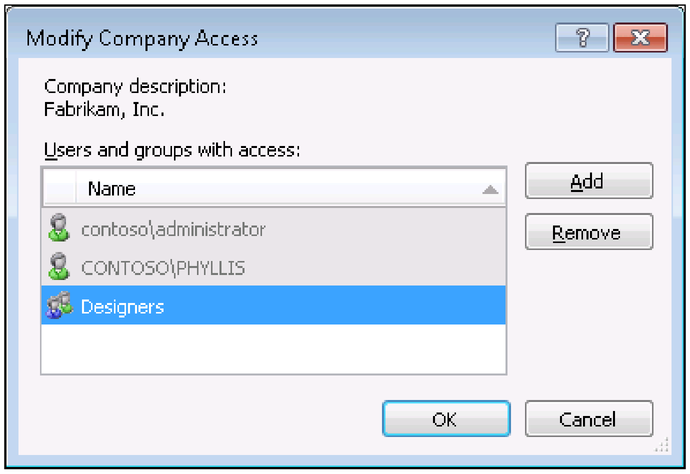
Once we've created our group, we can click over to the **Company Access** tab—as seen in the following screenshot—and select one or more companies that this group will be able to access:



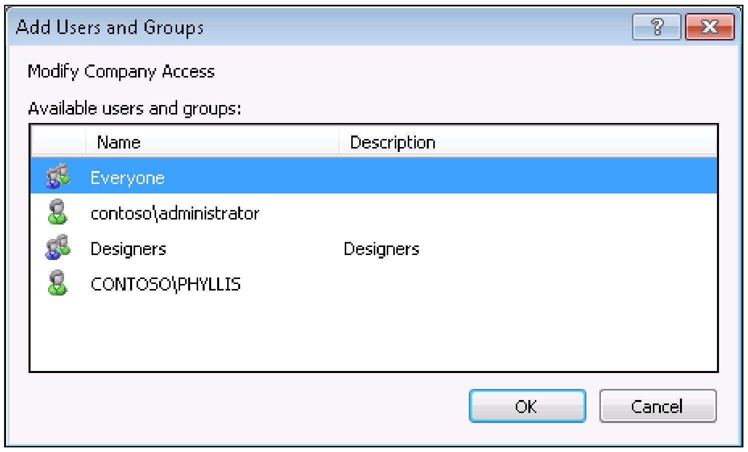
Access	Name	Description
☐	BIKES	Bikes Demand Planner
☐	CAD	CAD Company
☐	CAD-Curr	CAD Company
☐	CES	Contoso Entertainment
☐	FF	Fabrikam Foundation
☐	FM	Fabrikam Municipality
☐	PHONE	The Phone Company
☐	SUB	Subsidiary Company
✓	TWO	Fabrikam, Inc.
☐	TWO_A	Fabrikam A
☐	USD	USD Company
☐	USD-Curr	USD Company

Companies

Management Reporter also allows us to add or modify security at the company level. To do this, we select the **Company** node from the **Security** pane. This will give us a list of the companies set up in Management Reporter. We can double-click on a company to view the users and/or groups that have access to that company. Users or groups can be added or removed from this window as seen in the following screenshot:



To add a new user or group, simply click on **Add**, select a user or group from the list as seen in the following screenshot, and click on **OK**:


Add Users and Groups

Modify Company Access

Available users and groups:

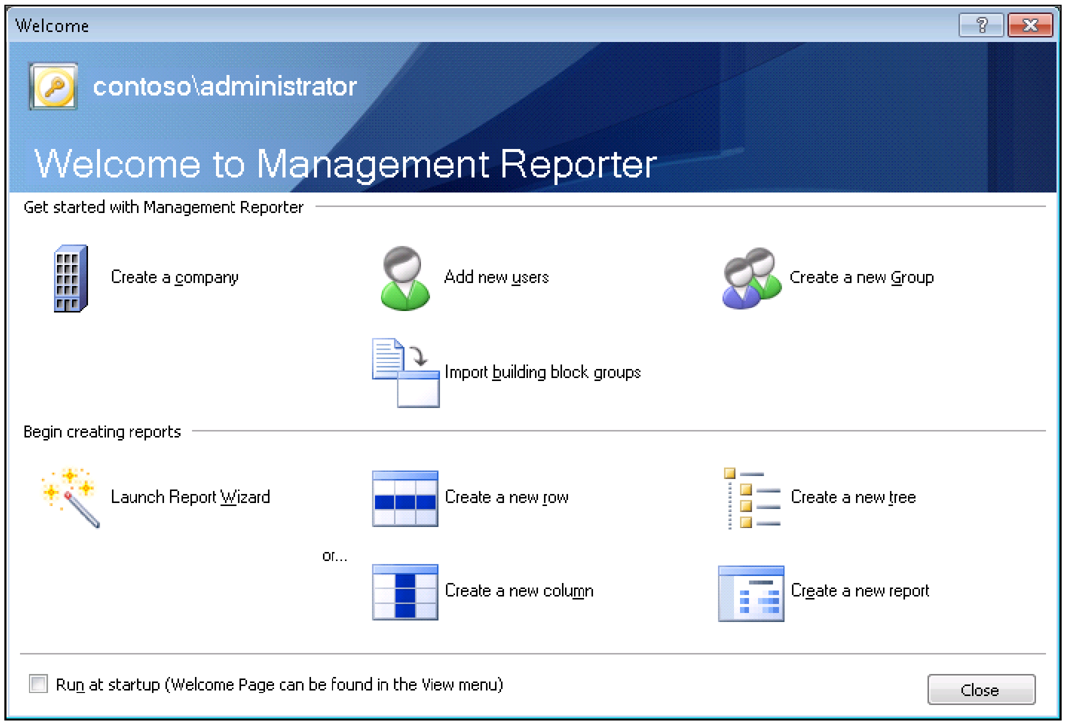
Name	Description
Everyone	
contoso\administrator	
Designers	Designers
CONTOSO\PHYLLIS	

Navigating the Management Reporter Report Designer interface

Since the Report Designer is where we will spend the majority of our time working with the various building block definitions that will form our reports, it is worth our while to become more familiar with the general layout of the Report Designer. Fortunately, the latest version of Management Reporter includes a layout that is similar in both style and feel to other Microsoft products, including Dynamics GP. This makes the process of navigating the Report Designer and other Management Reporter components much more intuitive.

For example, after launching the Report Designer, a Management Reporter 2012 splash screen appears and calls to mind the splash screen that accompanies the launch of GP 2013.

If we are opening Management Reporter for the first time, a **Welcome** screen appears as we can see from the following screenshot; we can quickly launch other areas of the application to immediately begin working on new building blocks, creating new companies, or adding new users, depending on the option we have selected:

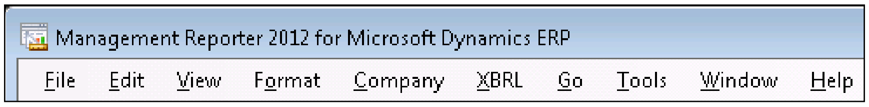


If you're one of those people who prefer to close this window immediately upon opening, save yourself some time by unchecking the option to **Run at Startup** found at the bottom left-hand corner of the **Welcome** screen. This will prevent the window from opening in the future. Users can re-open or re-enable the **Welcome** page by navigating to **View** | **Welcome Page** from the Menu bar after the Report Designer has fully launched.

Once the Report Designer is fully opened, we see a Menu bar across the top of the window with the traditional **File**, **Edit**, and **Help** menus alongside menus specific to Management Reporter. Additionally, we see a Navigation Pane down the left-hand side of the page that calls to mind the Navigation Pane found in GP Versions 10.0 and later.

The Menu bar

Although the traditional Menu bar has been replaced with a ribbon bar in many Microsoft products, Management Reporter still displays the traditional Menu bar at the top of the application. As we can see in the following screenshot, alongside the traditional menus such as **File**, **Edit**, and **Window**, we see other menus such as **XBRL**, **Company**, and **Tools** that offer access to commands specific to Management Reporter. Let's take a look at some of the commands accessible from the Menu bar that are specific to Management Reporter in the following screenshot:



File

From this menu, we can easily start creating a new definition or open an existing definition in one of the building blocks. Near the bottom of the **File** menu, we find a section from which we can open definitions from the building blocks on which we've recently worked. In environments where we have numerous definitions within a single building block (more on this later in the Grouping Building blocks section of this chapter), this can help ensure that we are selecting the most recent definition. In this menu, we are also provided with a **Close All** option that comes in handy when we have too many reports, rows, columns, or trees open.

Edit

From the **Edit** menu, we access our familiar Undo and Redo commands. Undo is considered a favorite among Management Reporter users as this functionality was not available in FRx. We also have access to other traditional editing functionality such as **Cut**, **Copy**, **Paste**, and **Replace**.

Also, depending on what we have opened in the Management Reporter workspace, we may see additional commands on the **Edit** menu. For example, if we access the **Edit** menu while we have a Row definition building block open, we will see additional commands such as **Insert Rows from Source System** or **Renumber Rows**. Options such as these can be extraordinary time savers, especially when we are working with entities that contain a large number of accounts.

View

The **View** menu allows us to control what is visible in our Management Reporter workspace. For example, if we'd rather increase the amount of real estate in which we have to create new and modify existing building blocks, we can navigate to **View | Navigation Pane** to remove the Navigation Pane from our view. Objects that are already visible in the Management Reporter workspace are designated with a checkmark in the **View** menu flyout.

Format

One of the great benefits of Management Reporter over other tools such as FRx is the ability to format cells and modify how they will appear in a generated report. Some of the commands that we might use in order to modify a cell's formatting can be found under the **Format** menu.

We can create predefined styles and templates for use throughout the Management Reporter application by navigating to **Format | Styles and Formatting**. This opens the **Styles and Formatting** window from which we can create font templates, each with predefined settings such as font size, font color, and other font formatting options. Later, instead of selecting each individual cell and each individual setting to format that cell, we can select the cell and select the predefined template to apply to that cell. All of the individual settings assigned to that template are then applied to the selected cell.

Company

The **Company** menu may not be visible to all users, as access to this menu is restricted only to users that have been granted Report Designer or administrator access within Management Reporter.

From this menu, we can view or modify some of our company (or entities, as they are more commonly called in Management Reporter) information.

Finally, we can use the Building Block Groups command to create new groupings of the building blocks for a specific entity. This is a useful tool that can help us manage what we have termed "building block sprawl" and we'll cover this functionality in more detail later in this chapter.

XBRL

Another option available from the Menu bar is the **XBRL** menu. The commands in this menu are designed to support the **XBRL** (**eXtensible Business Reporting Language**) functionality found in Management Reporter. XBRL is becoming increasingly more common as the financial industry seeks to find ways to standardize and make consistent the process of reporting across multiple organizations and multiple industries. The process of utilizing XBRL involves "tagging" a financial report based on a taxonomy created by a regulating authority. In this way, all reports under the same taxonomy share common tags that make it easier for financial analysts to review and compare financial statements from multiple companies.

From the **XBRL** command menu on the **Menu** bar, we can import and modify the taxonomy schema that we need for our reporting purposes. This requires administrator or Report Designer security access. This is accessed by navigating to **XBRL | Taxonomies**. This opens the **XBRL Taxonomies** window, from which we can enter the name of our taxonomy in the **Name** field and the path to the schema—either through a web URL or by browsing to a path on our local workstation if we've already downloaded the schema—so that it can be loaded into Management Reporter.

Additional commands on the **XBRL** menu allow us the necessary control we need in order to work with our imported taxonomy schema. Before we can create the XBRL instance document that contains the actual data with our tags, we properly set the entities and units by accessing them from the **XBRL** menu. Finally, once we've set these up, we must also make sure that we define an XBRL column in our column definition building block and links in our row definition to a specific XBRL taxonomy.

Go

If we have disabled the Navigation Pane from the **View** menu, we may find the commands found on this menu to be limited in their usefulness. By selecting one of the building blocks under the **Go** menu, we can choose which building block has primary focus in the Navigation Pane. This has the same effect as if we were to click on a specific building block at the bottom of the Navigation Pane. Additionally, notice the keyboard commands that correspond to each building block. While knowing these keyboard shortcuts is not critical to using Management Reporter, it can save us the extra time required to shift from the keyboard to the mouse.

If we don't have the Navigation Pane open, the only real effect this menu has is to change the focus of the **New** button on the Standard toolbar. For example, if we navigate to **Go | Row Definition**, clicking on the **New** button on the Standard toolbar will cause Management Reporter to open a new Row definition.

Tools

The **Tools** menu contains a wealth of commands that we can use to access important areas of Management Reporter. Many of these commands are only available to users with administrator or Report Designer access. Some of the useful commands in the **Tools** menu include the following:

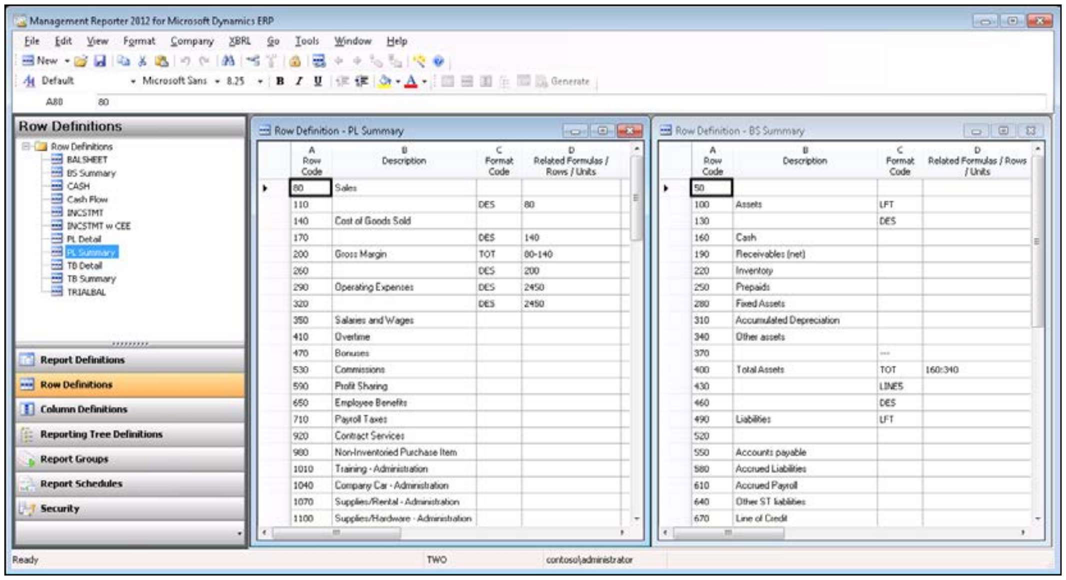
- **Protect**: This option is only available when an administrator or Report Designer has a reporting definition open. Essentially, this option allows users to password protect certain building blocks in order to restrict access to only those users who know the password.
- **Report Queue Status**: From this command, we can access the report queue to see the status of recently generated reports.
- **Report Wizard**: This helpful tool can get us started down the road towards creating a new financial statement. From the Report Wizard, we can select from a list of report templates, then, depending on the report template we've selected, the Report Wizard asks us a series of questions that help us determine the final format of our report. This is a great tool for beginners to use in order to see how Management Reporter creates the necessary definitions to support our selections.
- **Missing Account Analysis**: This tool searches for financial accounts and dimensions that might be missing across all row definitions, reporting tree definitions, and report definitions in a building block group.

- **Checked Out**: In order to see which users have building blocks open (and are thus "checked out"), we can access this command. It provides us with a list of the building blocks that are currently checked out and the user that has checked it out.

Other commands that are more administrative in nature, such as commands to open the **Connection** and **Registration** windows, are also available from the **Tools** menu for users with administrative access.

Window

Since new building blocks are opened in full screen mode, it can often be a challenge managing the various windows we have opened in Management Reporter. Several of the commands under the **Window** menu can help us manage our workspace more easily. For example, if we have two separate Row definitions open, we can select **Tile Vertically** or **Tile Horizontally** to get Management Reporter to resize and rearrange our two definitions for vertical or horizontal comparisons, respectively. The following screenshot shows what Management Reporter might look like with two Row definitions rearranged side by side with the **Tile Vertically** command:



Row Definition - PL Summary

A Row Code	B Description	C Format Code	D Related Formulas / Rows / Units
80	Sales		
110		DES	80
140	Cost of Goods Sold		
170		DES	140
200	Gross Margin	TOT	80-140
260		DES	200
290	Operating Expenses	DES	2450
320		DES	2450
350	Salaries and Wages		
410	Overtime		
470	Bonuses		
530	Commissions		
590	Profit Sharing		
650	Employee Benefits		
710	Payroll Taxes		
920	Contract Services		
980	Non-Inventoried Purchase Item		
1010	Training - Administration		
1040	Company Car - Administration		
1070	Supplies/Rental - Administration		
1100	Supplies/Hardware - Administration		

Row Definition - BS Summary

A Row Code	B Description	C Format Code	D Related Formulas / Rows / Units
50			
100	Assets	LFT	
130		DES	
160	Cash		
190	Receivables (net)		
220	Inventory		
250	Prepaids		
280	Fixed Assets		
310	Accumulated Depreciation		
340	Other assets		
370		---	
400	Total Assets	TOT	160:340
430		LINES	
460		DES	
490	Liabilities	LFT	
520			
550	Accounts payable		
580	Accrued Liabilities		
610	Accrued Payroll		
640	Other ST liabilities		
670	Line of Credit		

Additionally, from the **Window** menu, we can pinpoint specific building blocks to return to focus in our workspace. For example, if we have our Profit and Loss Row definition open in the forefront, and we want to quickly switch to our Balance Sheet Row definition, we can navigate to **Window | Balance Sheet** to bring this Row definition to the forefront of the window. This only works if we've already opened the Row definition during this session with Management Reporter.

Help

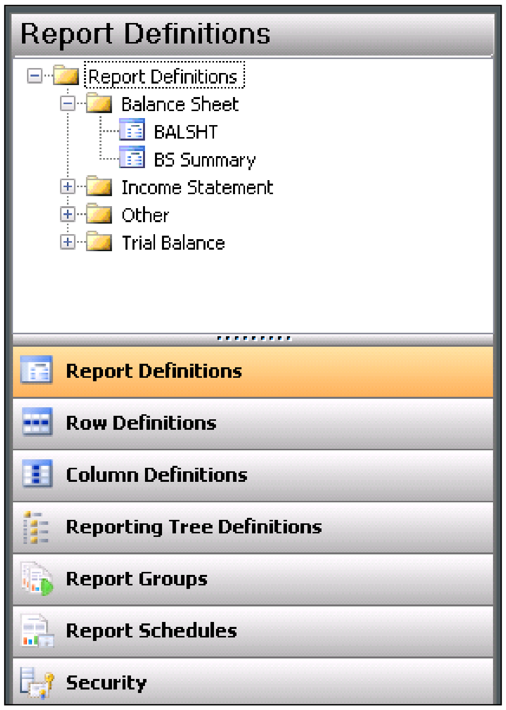
For all the assistance it provides, the help functionality in products such as Management Reporter and GP 2013 are often overlooked by users. Often, users question the relevance or use of a certain tool without ever thinking to look to the help files associated with the product. We strongly encourage users, especially those who are just getting started with Management Reporter, to utilize the **Help** command whenever possible. Navigating to **Help | Management Reporter Help** opens a searchable index of help topics related to Management Reporter. This index can be an invaluable tool in understanding the inner workings of Management Reporter. For those familiar with accessing the Dynamics GP Help by using the *F1* key, that functionality also works in Management Reporter.

The Navigation Pane

At the far left hand of the Management Reporter workspace, users will note a sidebar that looks very similar to the sidebar that appears in Dynamics GP 2013. At the bottom of this sidebar are several buttons, and clicking on these buttons dictates what appears in the uppermost portion of the sidebar.

If the Navigation Pane is not visible in the Management Reporter workspace, it may have been disabled. In order to re-enable the Navigation Pane, navigate to **View | Navigation Pane** from the Menu bar.

From the following screenshot, we see that the buttons correspond to the four building blocks we can utilize in building our reports. A fifth tab, **Security**, allows us to view more detail surrounding our Management Reporter security setup.



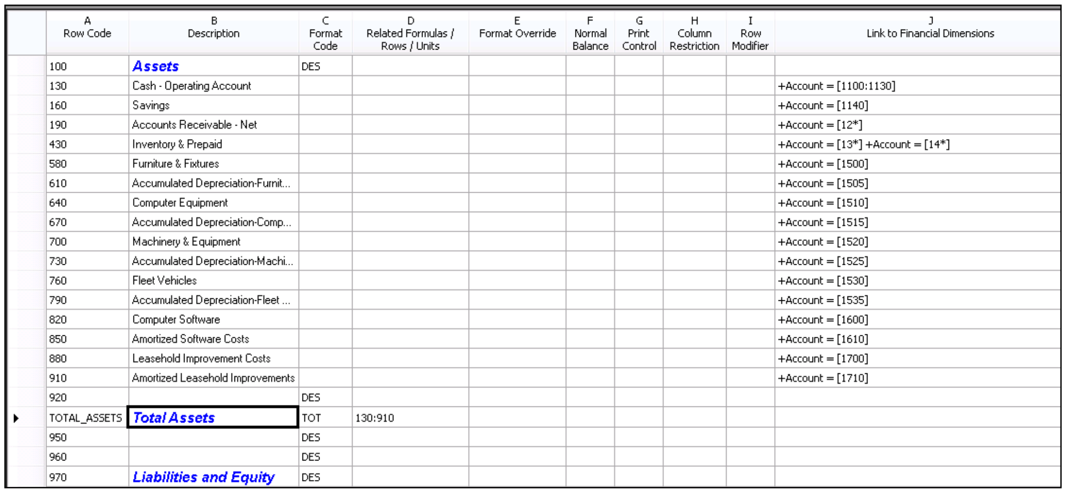
Selecting one of the building block buttons displays a list of existing definitions for that building block. From here, we can select one of the existing definitions to begin working with it. Additionally, the contents of each building block can be organized in folders, and this will be covered in more detail later in this chapter.

Working with the Management Reporter building blocks

Now that we have an understanding of the layout of Management Reporter, we can begin designing our reports starting with creating all of our building blocks. These building blocks consist of **Row**, **Column**, **Reporting Tree**, and **Report** definitions. Due to the sheer amount of detail and options available in each of the definitions, we can't possibly cover them all in a single chapter. We will, however, be discussing the layouts and the more commonly used options in each definition type. For the complete listing of all the available options, we recommend reviewing the user documentation and the Management Reporter help.

Row definitions

Row definitions are a required component for any report. We can think of them as exactly as they sound; they are the rows of the report. Typically, these are account numbers, but we can also use departments, cost centers, and so on as our rows. Let's look at the layout of the Report definition as shown in the following screenshot:



A Row Code	B Description	C Format Code	D Related Formulas / Rows / Units	E Format Override	F Normal Balance	G Print Control	H Column Restriction	I Row Modifier	J Link to Financial Dimensions
100	***Assets***	DES							
130	Cash - Operating Account								+Account = [1100:1130]
160	Savings								+Account = [1140]
190	Accounts Receivable - Net								+Account = [12*]
430	Inventory & Prepaid								+Account = [13*] +Account = [14*]
580	Furniture & Fixtures								+Account = [1500]
610	Accumulated Depreciation-Furnit...								+Account = [1505]
640	Computer Equipment								+Account = [1510]
670	Accumulated Depreciation-Comp...								+Account = [1515]
700	Machinery & Equipment								+Account = [1520]
730	Accumulated Depreciation-Machi...								+Account = [1525]
760	Fleet Vehicles								+Account = [1530]
790	Accumulated Depreciation-Fleet ...								+Account = [1535]
820	Computer Software								+Account = [1600]
850	Amortized Software Costs								+Account = [1610]
880	Leasehold Improvement Costs								+Account = [1700]
910	Amortized Leasehold Improvements								+Account = [1710]
920		DES							
TOTAL_ASSETS	***Total Assets***	TOT	130:910						
950		DES							
960		DES							
970	***Liabilities and Equity***	DES							

We can see from the preceding screenshot that there are quite a few columns that can be populated in a Row definition as follows:

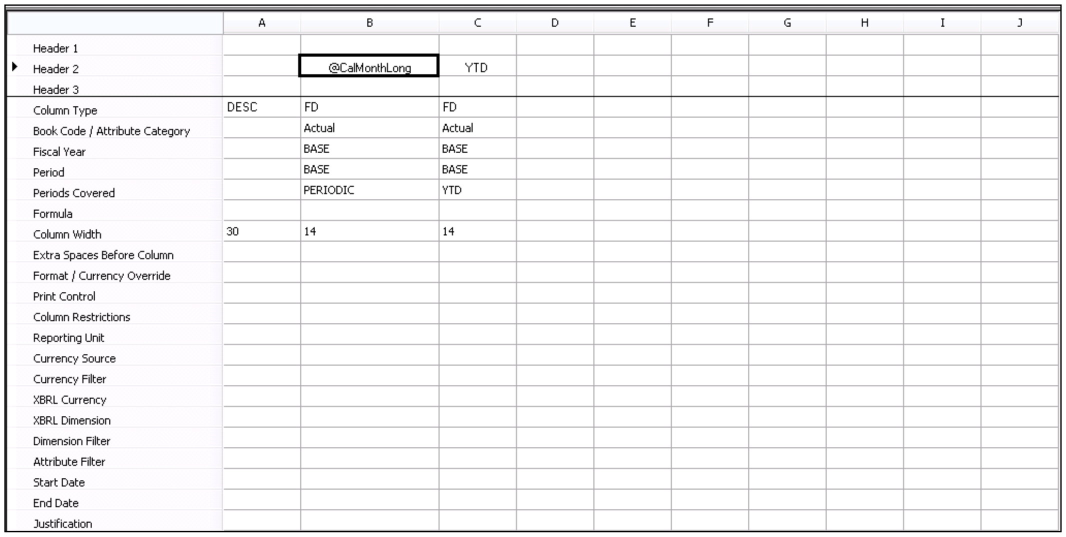
- **Row Code**: We can think of this as a line number similar to Microsoft Excel. We will use this in our **Related Formulas / Rows / Units** column to perform subtotals, totals, calculations, and so on. Typically, this column is incremented in a way that allows for us to enter rows in between. This code can be a number or alphanumeric as we can see in the preceding screenshot for **TOTAL_ASSETS**. Row codes can be renumbered at any time and will automatically update all occurrences in our **Related Formulas / Rows / Units** column.
- **Description**: This is where we put our account description section headers such as **Assets**, section totals such as **Total Assets**, and so on. We can also control how these will look by applying font styles such as size, bold, and shading.
- **Format Code**: This column is a code that basically tells the report what the line is doing. The most commonly used format codes are as follows:
    - **DES**: This means description line.
    - **TOT**: This means that it is a total row. This can be a range of row codes, 130:910 for instance, or it can be an addition or subtraction of row codes.
    - **CAL**: This is a calculation. This allows for complex calculations. We can multiply and divide using calculations, as well as use `if/then/else` statements in a calculation.
    - **CBR**: Change base row is used when creating reports that do some type of percentage allocation such as having each row be displayed as a percent of sales.
- **Related Formulas / Rows / Units**: This is the column where we will actually put our calculations for **TOT** and **CAL** format codes. We can also use this column to restrict a line to a particular reporting tree unit if we are using a Reporting Tree. Lastly, we can relate a row to another row. An example of this would be relating a header row or an underscore to a subtotal so that when the subtotal is zero and suppressed, the header row and/or underscore will also be suppressed.
- **Format Override**: This column allows us to change the formatting of currencies, decimals, and percentages.

- **Normal Balance**: This column lets us control whether an account is typically a credit in Dynamics GP and will reverse the sign. An example of this would be Revenue accounts. By setting this to **C**, revenue numbers would show as positive numbers on our report.
- **Print Control**: This allows us to set options such as Non printing, use currency symbol, suppress if all zeros, and so on for our rows. This is on a row-by-row basis. We can also specify whether the row prints debit or credit balances only if we wish.
- **Column Restrictions**: This column has multiple uses depending on the type of row. One of the common uses is to limit the printing of the row's amounts to specific columns.
- **Row Modifier**: This allows for overriding the period data or overriding the book code. We can override a row to pull the beginning balance for that row if we wish. As for overriding the book code, this would allow us to use, say, an amount from a budget for a particular row instead of the actuals.
- **Link to Financial Dimensions**: This is probably the most important column in the Row definition. This is where we actually link to our GL data. We can either type in the segment we are building the row off of and the segment value(s), or we can double-click on to get a lookup window that will format the syntax for us. Keep in mind, this can be accounts, departments, cost centers, and so on. The important thing to note is that whatever value we are using for our link, they *must* be of consistent lengths.

As we mentioned, more detail on additional options exists in the user documentation.

Column definitions

As with Row definitions, Column definitions are also required to generate a report. Column definitions are used to create the columns of the report and typically include the actual, budget, or forecast data for particular periods or months. In more advanced reports such as consolidated reports, columns can also contain companies, departments, cost centers, and so on. Let's take a look at the Column definition layout as shown in the following screenshot:



	A	B	C	D	E	F	G	H	I	J
Header 1										
Header 2		@CalMonthLong	YTD							
Header 3										
Column Type	DESC	FD	FD							
Book Code / Attribute Category		Actual	Actual							
Fiscal Year		BASE	BASE							
Period		BASE	BASE							
Periods Covered		PERIODIC	YTD							
Formula										
Column Width	30	14	14							
Extra Spaces Before Column										
Format / Currency Override										
Print Control										
Column Restrictions										
Reporting Unit										
Currency Source										
Currency Filter										
XBRL Currency										
XBRL Dimension										
Dimension Filter										
Attribute Filter										
Start Date										
End Date										
Justification										

As we can see in the preceding screenshot, the Column definitions are broken into two sections, a header section and the detail section. In the header, we can either type in headings or we can use auto text that will let the column headers be dynamically set based on the report. An example of this would be using the `@CalMonthLong` text to display the abbreviated month name for the month for which we generate the report.

> If we want to remove the day of the week from the long date format, we can change this in the **Regional Settings** in the **Control Panel** on the server or workstation.

The detail section is where we actually specify what type of column we need as well as where to get the data and for which period. The more common selections are the following:

- **Column Type**: This specifies what type of data will be in the column. The most common are as follows:
    - **DESC**: Descriptions from the **Description** column in the Row definition.
    - **FD**: Amounts from the **Financial** dimension (that is Dynamics GP data).
    - **CALC**: A calculated column. An example of this would be if we needed to add three columns together to calculate a quarter to date column.

- **Book Code**: This lets the system know what book to get the data from. This can be actuals, budget, or forecasts.
- **Fiscal Year**: This is the year for the column. This can be hardcoded, set to BASE (the year the report is generated for), or any combination of BASE+# or BASE-#.
- **Period**: This is similar to fiscal year, it is the period the column will have. This can also be set to BASE, BASE+#, BASE-#, or hardcoded.
- **Period Covered**: This tells the system whether the column should have current activity, YTD activity, or force in beginning balances.

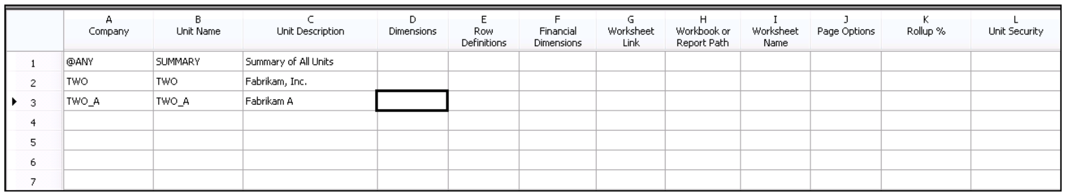
These are the required lines, but there are quite a few other options that can be set. Some of the more commonly used are as follows:

- **Formula**: This is used in conjunction with a **CALC** column type. This is where the actual formula resides, for instance `B+C+D` or `B:D`.
- **Print Control**: This can be used to set conditional column printing. This is typically used in twelve month rolling columns. As each month becomes current, that month will show up on our report so we don't have empty columns for future periods.
- **Reporting Unit**: Typically used in side by side reporting where we need to restrict a column to a particular unit of a Reporting Tree, company, or department for example.
- **Dimension Filter**: This is used when we want to filter a column of data to a particular dimension.

As with our Row definitions, we did not cover every column option in detail. For information on some of the other less used options, see the user guide or help.

Reporting Tree definitions

Reporting Trees are optional components of reports, however, they do provide a very powerful method of performing company consolidated reporting based on any type of tree hierarchy you can design. Typically, this is company, department, or cost center based as shown in the following screenshot:



	A Company	B Unit Name	C Unit Description	D Dimensions	E Row Definitions	F Financial Dimensions	G Worksheet Link	H Workbook or Report Path	I Worksheet Name	J Page Options	K Rollup %	L Unit Security
1	@ANY	SUMMARY	Summary of All Units									
2	TWO	TWO	Fabrikam, Inc.									
▸ 3	TWO_A	TWO_A	Fabrikam A									
4												
5												
6												
7												

As we can see from the preceding screenshot, there are a few columns that seem pretty straightforward, and it is no coincidence that these are the most used when creating a Reporting Tree definition, as follows:

- **Company**: This tells the system which Dynamics GP company to read the data for that unit from. A value of `@ANY` means to use any company and is typically reserved for summary units.
- **Unit Name**: This is a short name for the unit.
- **Unit Description**: This is a longer description for the unit and can be used in the Report definition as part of the page headers.
- **Dimensions**: This is where we enter the dimensions that our tree is going to be built from. Again, typically, these are the department numbers or cost centers that we are creating a reporting hierarchy for.

Other columns that are sometimes used are as follows:

- **Row Definitions**: This can be used in a circumstance where we are consolidating two companies that have two different charts of accounts. We can use this to link each unit to a different Row definition to account for this.
- **Row Link and External File**: These are used when linking to an Excel spreadsheet.
- **Unit Security**: This can be used to set access to individual units of the reporting hierarchy.

Although Reporting Trees can give us highly segmented reports, they are also usually the building block that causes the most problems when generating reports. Reporting Trees can be as small or large as we want to make them. It all depends on what our end product needs to be. Our advice is to always start small. Build a small roll up to try out the design. If it works, we can continue building the entire hierarchy.

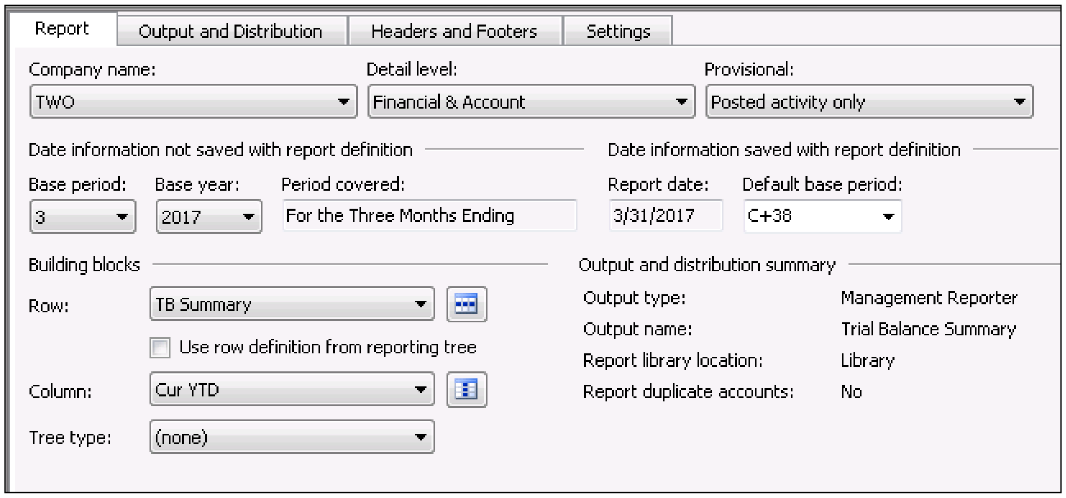
If we run into a problem generating a report with a tree, we can simply turn off the tree and see if the report generates without it. If this is the case, we know our tree is the culprit and we can begin to search the tree for errors.

Report Definitions

The final building block is the Report definition. This is where we actually combine our Row, Column, and if necessary, Reporting Tree definitions to form our report. When creating a Report definition, we can also set up the output and distribution, headers and footers information, and any additional settings.

Report

As we define our Report definition, we notice several tabs across the top of the window. The first of these, as seen in the following screenshot, is the **Report** tab:



Company name: TWO
Detail level: Financial & Account
Provisional: Posted activity only

Date information not saved with report definition
Base period: 3
Base year: 2017
Period covered: For the Three Months Ending

Date information saved with report definition
Report date: 3/31/2017
Default base period: C+38

Building blocks
Row: TB Summary
☐ Use row definition from reporting tree
Column: Cur YTD
Tree type: (none)

Output and distribution summary

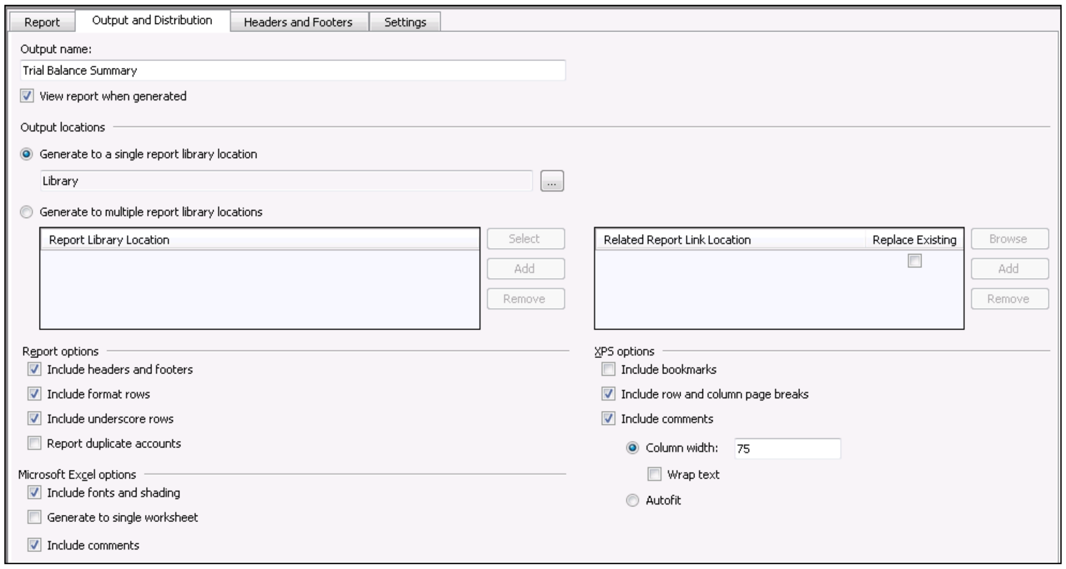
Output type:	Management Reporter
Output name:	Trial Balance Summary
Report library location:	Library
Report duplicate accounts:	No

On the **Report** tab, we set up the following fields:

- **Company name:** Select the Dynamics GP company from the drop-down menu or select `@ANY`.
- **Detail level:** Selections here range from **Financial** to all the way down to the **Financial, Account, and Transaction** detail.
- **Provisional:** Allows for running on posted activity, unposted activity, or both.
- **Base period:** The period that will be set as our base period. Typically, the current month.
- **Base year:** The year that will be set as our base year. Typically, the current year.
- **Building blocks**:
    - **Row:** Select the Row definition from the drop-down menu. If using a Reporting Tree with Rows assigned to individual tree units, check the box to **Use row definition** from Reporting Tree.
    - **Column:** Select the Column definition from the drop-down menu.
    - **Tree type:** If using a tree, select **Reporting Tree** as the tree type, and select the tree to use in the report.

Output and Distribution

Beyond the **Report** tab, we see the **Output and Distribution** tab, which contains options for modifying the appearance of our final report as shown in the following screenshot:



Output name: Trial Balance Summary

☑ View report when generated

Output locations

● Generate to a single report library location

Library

○ Generate to multiple report library locations

Report Library Location

Related Report Link Location	Replace Existing
	☐

Report options

☑ Include headers and footers

☑ Include format rows

☑ Include underscore rows

☐ Report duplicate accounts

Microsoft Excel options

☑ Include fonts and shading

☐ Generate to single worksheet

☑ Include comments

XPS options

☐ Include bookmarks

☑ Include row and column page breaks

☑ Include comments

● Column width: 75

☐ Wrap text

○ Autofit

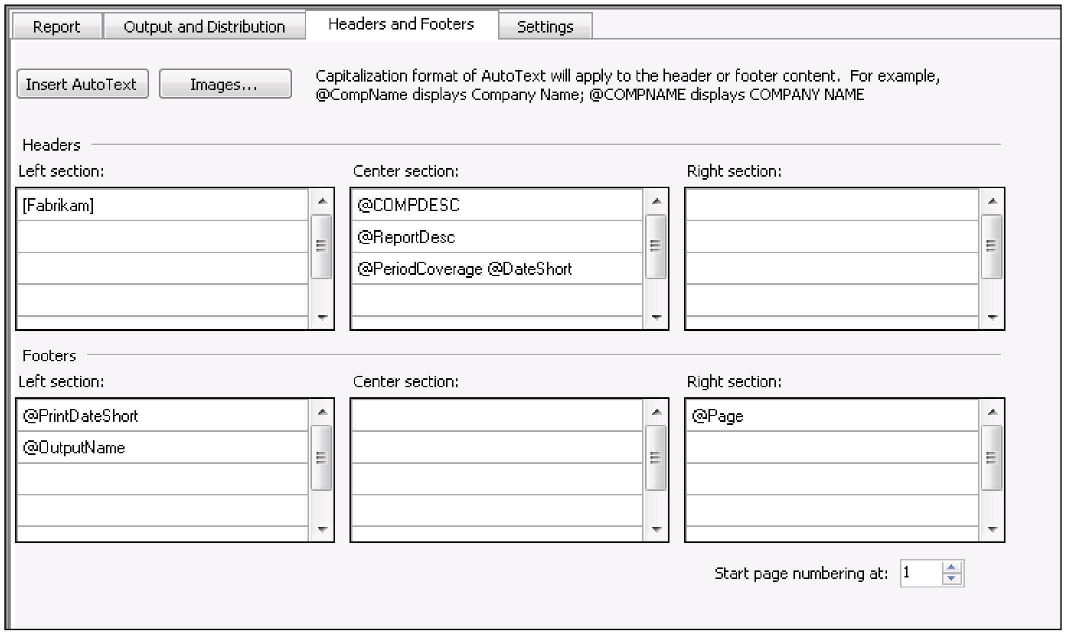
On the **Output and Distribution** tab, we assign the following fields:

- **Output Type**: Select Management Reporter, Microsoft Excel, or the XBRL Instance document.
- **Output Name**: This is the name of the report.
- **Report library location**: Only used for Management Reporter type. This is the location in the Report Library where the report will be stored. The Report Library will be covered in more detail in the next chapter.
- **Output path**: This is used for both Microsoft Excel and XBRL Instance document types. This is the location on the local machine or network file share where the file is to be stored.
- Management Reporter Options:
    - Include headers and footers
    - Include format rows
    - Include underscore rows

- Other Options:
    - Replace output file without warning
    - View report when generated
- Exception reporting

Headers and Footers

Our third tab is the **Headers and Footers** tab, which can be seen in the following screenshot:



Capitalization format of AutoText will apply to the header or footer content. For example, @CompName displays Company Name; @COMPNAME displays COMPANY NAME

Headers

Left section: [Fabrikam]

Center section: @COMPDESC, @ReportDesc, @PeriodCoverage @DateShort

Right section:

Footers

Left section: @PrintDateShort, @OutputName

Center section:

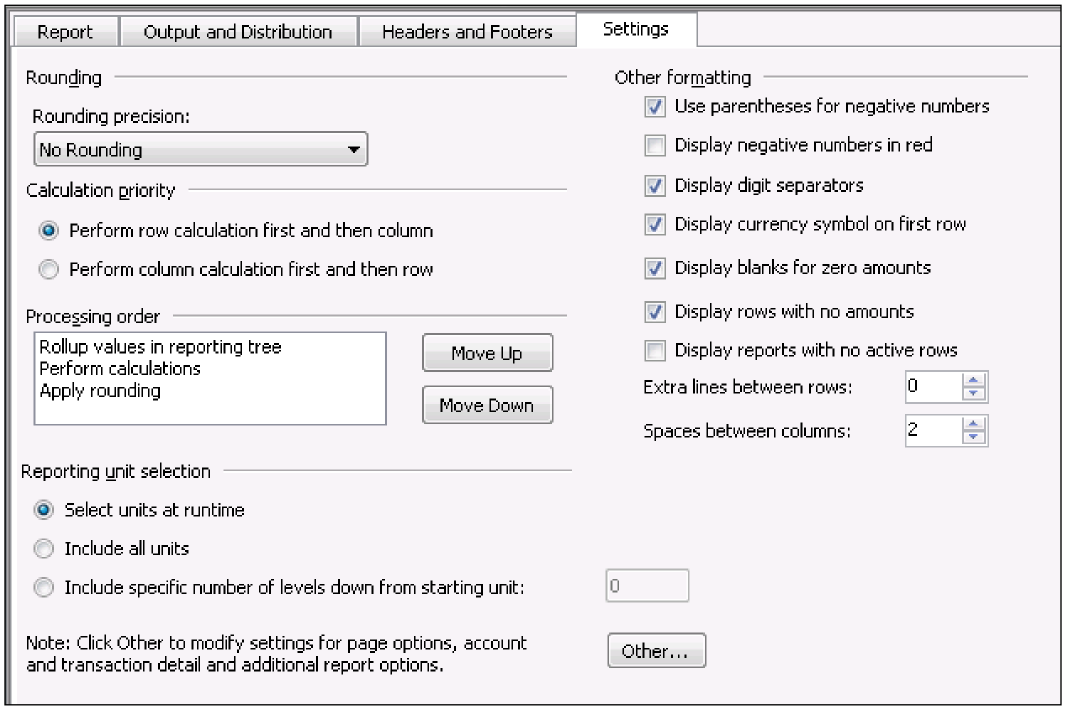
Right section: @Page

Start page numbering at: 1

In the **Headers and Footers** tab, we have many different AutoText selections to choose from for report names, dates and times, page numbering, and so on. We can also quickly and easily add images to any of the sections by clicking on the **Images** button and adding the image.

Settings

The last tab is the **Settings** tab.



Rounding

Rounding precision: No Rounding

Calculation priority

[✓ Perform row calculation first and then column]

Perform column calculation first and then row

Processing order

Rollup values in reporting tree
Perform calculations
Apply rounding

Reporting unit selection

[✓ Select units at runtime]

Include all units

Include specific number of levels down from starting unit: 0

Note: Click Other to modify settings for page options, account and transaction detail and additional report options.

Other formatting

- [✓] Use parentheses for negative numbers
- [ ] Display negative numbers in red
- [✓] Display digit separators
- [✓] Display currency symbol on first row
- [✓] Display blanks for zero amounts
- [✓] Display rows with no amounts
- [ ] Display reports with no active rows

Extra lines between rows: 0

Spaces between columns: 2

This is where we will find options such as rounding preferences, calculation priorities, and processing order as well as how to handle blank rows, negative numbers, rows with no amounts, and so on.

Improvements in Management Reporter 2012

Since the previous release of Management Reporter, many of the concerns users had have been addressed. In addition to these updates, there were also a number of improvements made as follows:

- **Data Mart storage**: This improvement continuously moves data from Dynamics GP to a separate database which allows it to be optimized for much faster report generation.

- **Multicurrency support**: This functionality allows users to assign multiple currencies including Functional, Local, or Reporting in Column Definitions.
- **Web based Report Viewer**: With this new feature, report viewers can view financial reports without having to have access to the Management Reporter desktop client report viewer. This will also be the default viewer when viewing reports published to SharePoint.
- **Missing Account Analysis Tool**: With this handy new feature, report designers can quickly identify building blocks that may be missing one or more new accounts or segments.

Tools for reducing building block sprawl

One of the challenges of using an enterprise-wide reporting tool such as Management Reporter is that multiple users can access the application and each user can create his or her own building blocks. As we've seen, the building blocks are extremely portable and can be used in multiple **Report Definitions**. Through good training and careful report design, our users can be taught to utilize existing building blocks to assist in the creation of new reports before creating a brand new set of building blocks.

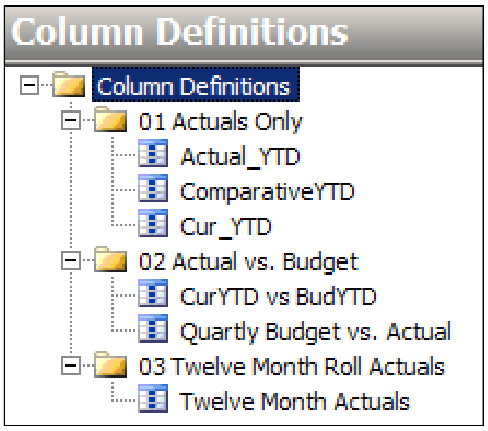
Inevitably, however, we encounter the building block sprawl. This is especially common in multicompany and multiuser environments. The symptoms of the building block sprawl usually make this a pretty easy disease to diagnose:

- Selecting **Row Definition** or **Column Definition** in the Navigation Pane reveals a seemingly unorganized collection of building blocks that, in some cases, may extend past the bottom of the screen.
- Multiple building blocks exist that share similar names such as **P and L**, **Profit and Loss**, **PL**, or **Income Statement** in the **Row Definitions** pane or **Actual_YTD** or **Cur_YTD** in the **Column Definitions** pane.
- When queried on the purpose of specific building blocks, users can only point to the ones they've used most recently with little to no knowledge of how other building blocks are used or why they even exist.
- The building block sprawl can cause users to build incorrect or incomplete financial statements. We should do our best to proactively manage the creation of too many building blocks in our Management Reporter environment. Fortunately, Management Reporter provides us with several tools that can be used to get a handle on an exploding list of building blocks.

Grouping building blocks in the Navigation Pane with the use of folders

One of the great new features of Management Reporter is the ability to group building blocks in folders in the Navigation Pane. Folders can be added to each of the four building blocks and the contents of each building block can be grouped within these folders. By judiciously creating folders in each building block area, we can encourage our users to insert newly created building blocks in the appropriate folder.

For example, we might have several **Column definitions** that can be used depending on whether or not we only want to see posted amounts or if we want to see budget amounts compared to actual amounts for a given time period. We can use folder groupings to group all Column definitions that contain budget amount columns together and all Column definitions that don't contain budget amount columns in another folder. Such a scenario might look as shown in the following screenshot:



Column Definitions

- Column Definitions
  - 01 Actuals Only
    - Actual_YTD
    - ComparativeYTD
    - Cur_YTD
  - 02 Actual vs. Budget
    - CurYTD vs BudYTD
    - Quartly Budget vs. Actual
  - 03 Twelve Month Roll Actuals
    - Twelve Month Actuals

Several other concepts are worth noting about using folders to group building blocks, as follows:

- First, we can see in the previous screenshot that numbers can be used to sort folders in a particular order in the Navigation Pane. Due to the nature of the sorting engine, if we plan to have more than 10 folders in a certain building block, it is helpful to begin numbering with 01, 02, 03, and so on. Otherwise, if we used only a single digit for the first nine folders, we would see the tenth folder appear as second in the list (that is, the sort order would appear as 1, 10, 2, 3, 4, 5, and so on).

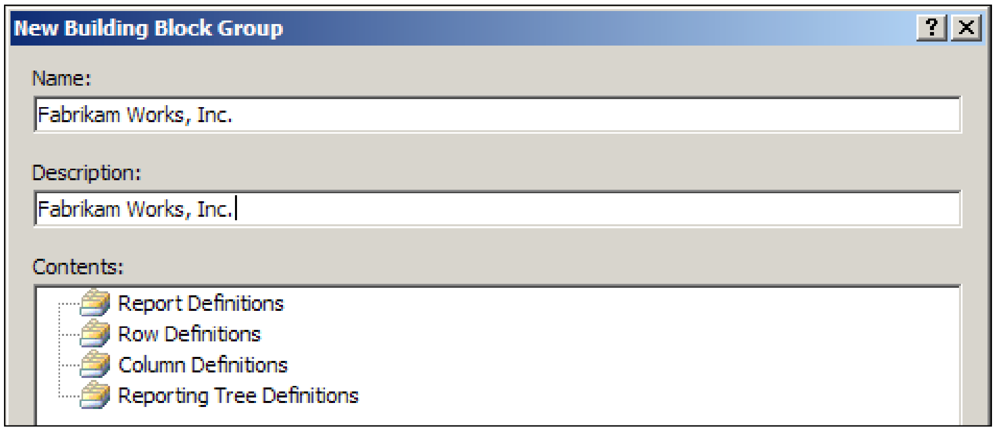
- Second, only single level folder groupings can be created. While it would be nice if folder within folder groupings were allowed, this is currently not possible in Management Reporter. Hopefully, we'll see this in a future version.
- By setting up folders ahead of time, we can be proactive in taming the building block sprawl before it starts.

Creating building block groups in environments with multiple entities

Another handy feature offered in Management Reporter to tame our building block sprawl is the building block groups feature. This feature is especially useful in environments where we have multiple entities (or companies) set up for reporting purposes. By assigning a building block group to an entity, we can control what building blocks appear in Management Reporter when that entity is selected for reporting.

Creating a new building block group and assigning it an entity requires a fairly easy set of steps as follows:

1. From the **Menu** bar, navigate to **Company | Building Block Groups**.
2. Click on the **New** button to open the **New Building Block Groups** window.
3. Enter a **Name:** and **Description:** for the new building block group and click on **OK**.

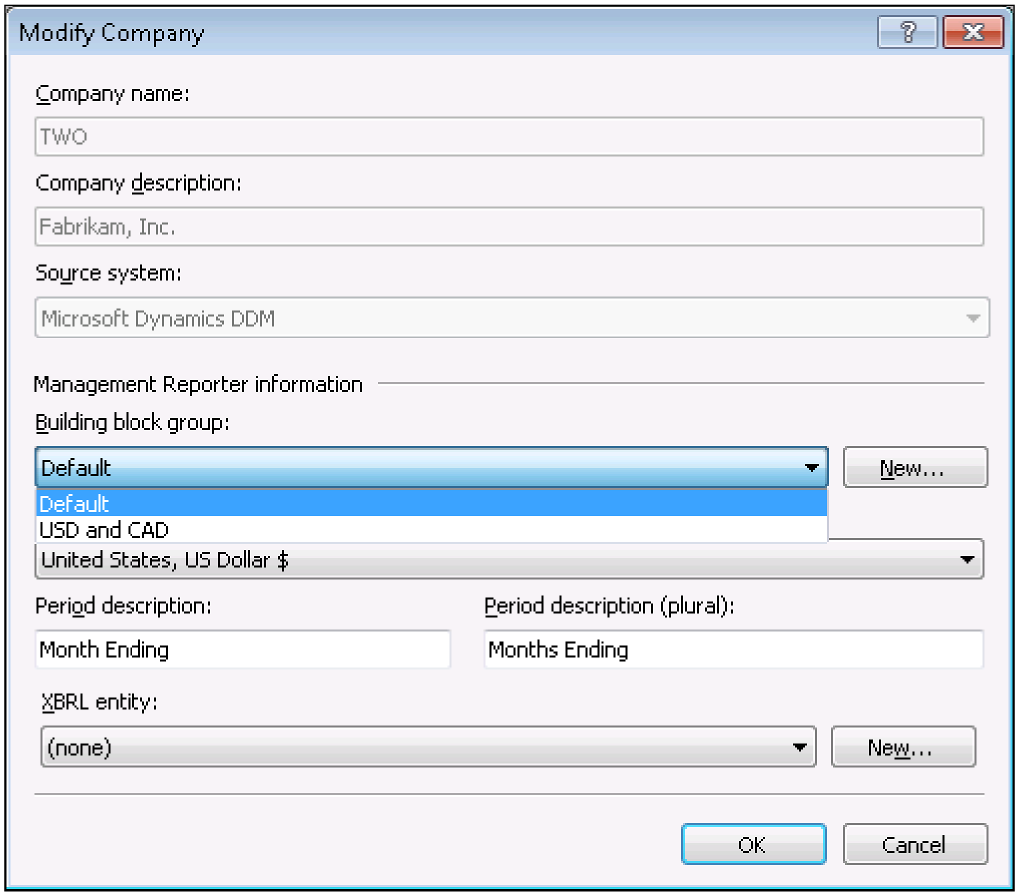


Name: Fabrikam Works, Inc.

Description: Fabrikam Works, Inc.

4. After the new building block group has been created, close the **New Building Block Groups** window.

5. To assign the building block group to the current entity, navigate to **Company** | **Companies** from the Menu bar.
6. Select the company to assign to the new building block group and click on **Modify**.
7. In the **Modify Company** window, change the value in the **Building block group:** drop-down menu to the newly created building block group.


Modify Company

Company name: TWO

Company description: Fabrikam, Inc.

Source system: Microsoft Dynamics DDM

Management Reporter information

Building block group: Default

Default
USD and CAD

United States, US Dollar $

Period description: Month Ending

Period description (plural): Months Ending

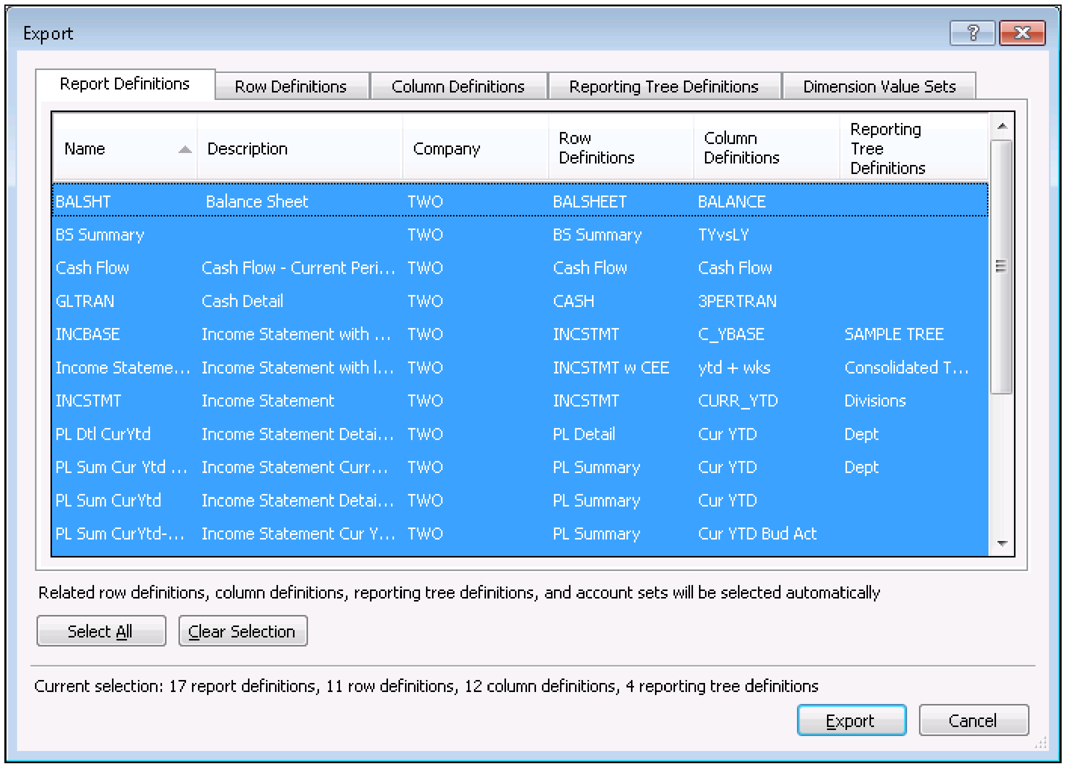
XBRL entity: (none)

8. Click on **OK** to close the window.

Management Reporter will refresh and the Navigation Pane will no longer show the building blocks associated with the building block group prior to making this change. Now, any new building blocks that we create in this entity will be assigned to the building block group we just created.

In order to see the same building block in other entities, we can use the building block import and export functionality to duplicate the building blocks. The following set of instructions demonstrate how we can export two Report definitions and their associated Row and Column definitions from our **Default** building block group into our newly created **Fabrikam Works, Inc.** building block group:

1. From the Menu bar, navigate to **Company** | **Building Block Groups**.
2. Select the building block group from which building blocks are to be exported.
3. Click on the **Export** button to open the **Export** window.
4. Select the Report definitions to be exported. Use the *Ctrl* key to select multiple Report definitions at once. As new building blocks are included in the selection, notice how the **Current selection:** field updates to let us know how many building blocks of each type will be included in our export as shown in the following screenshot:

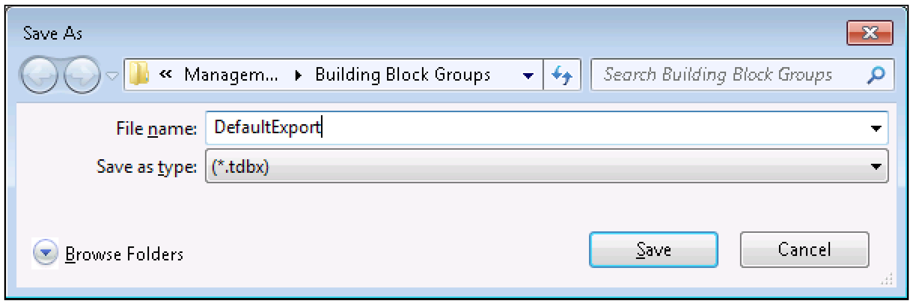


Name	Description	Company	Row Definitions	Column Definitions	Reporting Tree Definitions
BALSHT	Balance Sheet	TWO	BALSHEET	BALANCE	
BS Summary		TWO	BS Summary	TYvsLY	
Cash Flow	Cash Flow - Current Peri…	TWO	Cash Flow	Cash Flow	
GLTRAN	Cash Detail	TWO	CASH	3PERTRAN	
INCBASE	Income Statement with …	TWO	INCSTMT	C_YBASE	SAMPLE TREE
Income Stateme…	Income Statement with I…	TWO	INCSTMT w CEE	ytd + wks	Consolidated T…
INCSTMT	Income Statement	TWO	INCSTMT	CURR_YTD	Divisions
PL Dtl CurYtd	Income Statement Detai…	TWO	PL Detail	Cur YTD	Dept
PL Sum Cur Ytd …	Income Statement Curr…	TWO	PL Summary	Cur YTD	Dept
PL Sum CurYtd	Income Statement Detai…	TWO	PL Summary	Cur YTD	
PL Sum CurYtd-…	Income Statement Cur Y…	TWO	PL Summary	Cur YTD Bud Act	

Related row definitions, column definitions, reporting tree definitions, and account sets will be selected automatically

Current selection: 17 report definitions, 11 row definitions, 12 column definitions, 4 reporting tree definitions

5. Additionally, we can select the other tabs at the top of the **Export** window to identify and select individual building blocks for export that may not be included by association with any of the selected definitions.
6. Once we've found and selected all of our definitions for export, we can click on the **Export** button.
7. We are prompted to save an export file of the `.tdbx` file format. Save this file to an accessible location as shown in the following screenshot:


Save As

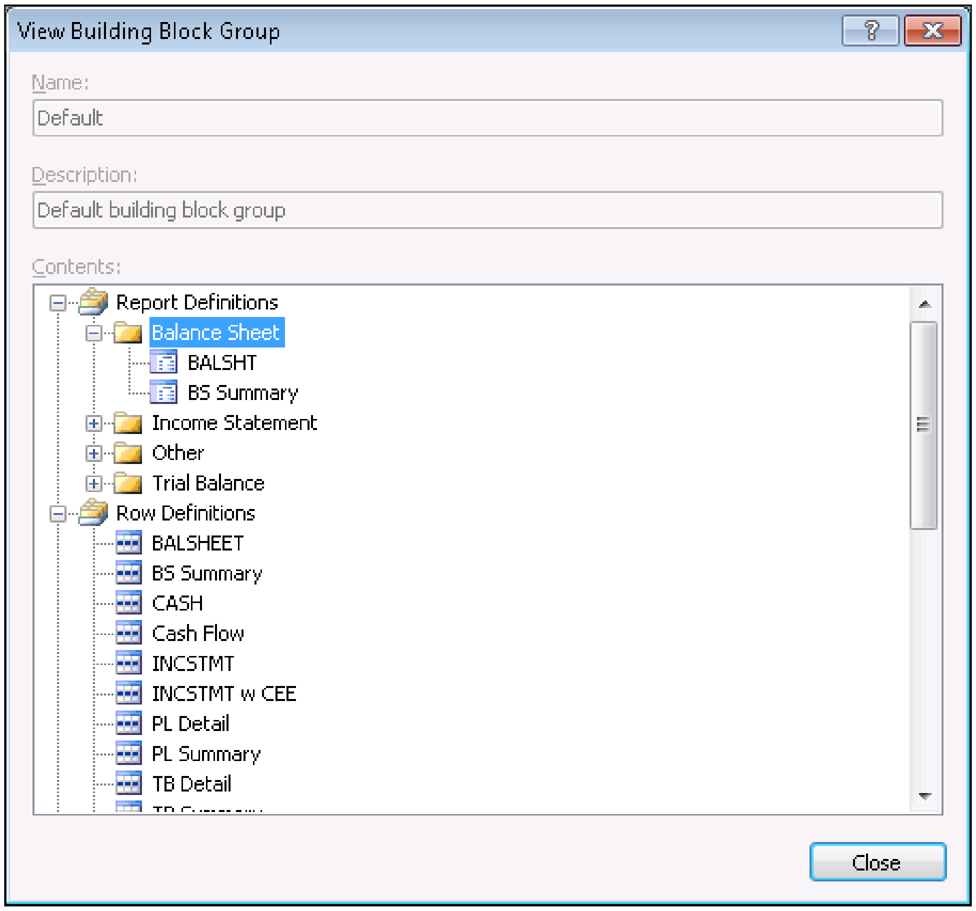
« Managem... ▸ Building Block Groups

File name: DefaultExport

Save as type: (*.tdbx)

8. Once the file has been saved and we are returned to the **Building Block Groups** window (**Company** | **Building Block Groups**), we should then select the building block group into which we wish to import our newly created file.
9. Once we've selected the target building block group, we can select the **Import** button to open the file explorer window.
10. Find and select the `.tdbx` file saved in our earlier step. Click on **Open**.
11. In the **Import** window, select the definitions that should be imported into the new building block group. Be sure to check the various tabs at the top of the screen to make sure all relevant definitions are included in the import.

12. Once our import is complete, we can select the target building block group in the **Building Block Groups** window, and click on the **View** button to open the **View Building Block Group** window. We can then see the various definitions that have been imported into this building block group as shown in the following screenshot:



Name: Default

Description: Default building block group

Contents:

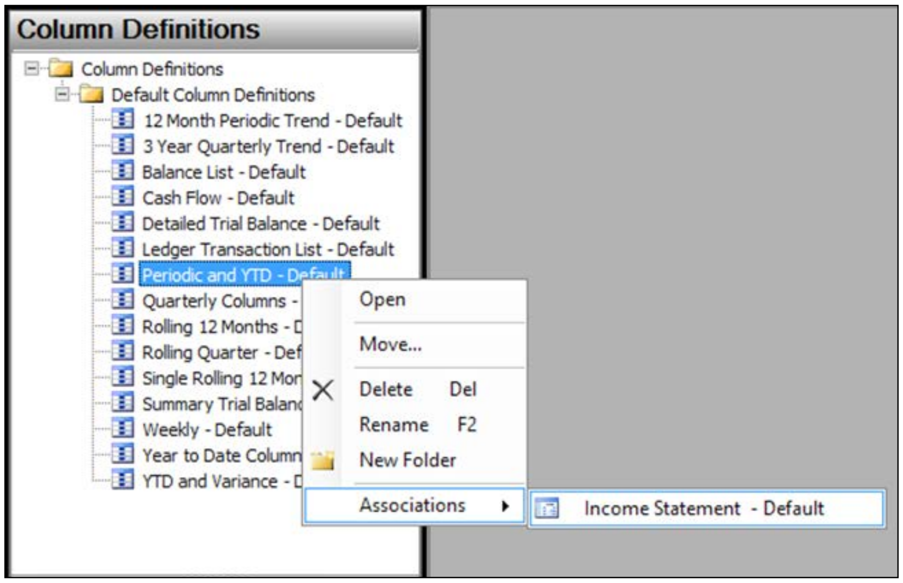
- Report Definitions
  - Balance Sheet
    - BALSHT
    - BS Summary
  - Income Statement
  - Other
  - Trial Balance
- Row Definitions
  - BALSHEET
  - BS Summary
  - CASH
  - Cash Flow
  - INCSTMT
  - INCSTMT w CEE
  - PL Detail
  - PL Summary
  - TB Detail

Now that we've learned how to use building block groups, we may want to consider creating a building block group for each separate entity as we set up Management Reporter. This will allow us to maintain a separate set of building blocks for each company from the very beginning. After creating these building block groups, we may also want to consider creating a generic set of building blocks and using the import/export functionality to copy this generic set from one building block group to the next. In this manner, each company will have a clean set of basic building blocks from which to begin.

Finding and eliminating unused building blocks by using building block associations

As the number of building blocks in our system increases, it can be a challenge to tell which building block is used with a particular Report definition. Over a period of time, certain building blocks may no longer even be used in Management Reporter. In these cases, the best course of action might be to find and identify these building blocks so that a decision can be made on whether or not they should remain in the system.

Fortunately, Management Reporter provides us with a handy little tool in the form of the **Associations** feature that can help us make this determination. If we have a particular building block open in the Navigation Pane, we can right-click on a specific building, and select **Associations** to see the Report definitions associated with the selected building block. The following screenshot shows us that the **CURR YTD** Column definition is associated with the Report definition's named **INCSTMT** and **TRIALBAL** as shown in the following screenshot:



By using the **Associations** feature in Management Reporter, we can quickly identify building blocks that are not associated with any Report definitions. While this does not necessarily mean the building block should be removed, it does give us some insight into whether or not this is a duplicate building block or one that can possibly be removed to make room for more utilized building blocks.

Summary

Before we can actually generate professional looking financial statements based on the latest data in our source system, we need to have the tools to build those reports. In this chapter, we covered the basic tools and building blocks that we need in order to design our financial statements. We discussed the underlying architecture of Management Reporter, how to navigate through the Report Designer interface, and how to use the building blocks to create our Row, Column, Reporting Tree, and Report definitions. We also covered some tips and techniques for taming the inevitable building block sprawl in a multicompany, multiuser environment.

In our next chapter, we will see the end result of our work with the reporting tools we used in this chapter. Specifically, we will look at the various options related to generating a report from the Report Designer. Then, we will cover the separate Report Viewer interface from which we can view and manage the reports we've generated. Finally, we will discuss some of the differences between Management Reporter and FRx.

9
Viewing Financial Reports in Management Reporter

Continuing from where we left off in the previous chapter on Report Design, this chapter will take us further into Management Report and utilizing the Report Viewer to organize and view our generated reports. The Report Viewer for Management Reporter is the tool we will primarily discuss to view our generated reports. However, we will also talk about the advantages of the Web Viewer and how we can distribute reports using the Web Viewer. We will also discuss how to use the viewer's library functionality to organize our reports in a number of ways including by company, report type, time frame, and so on. Additionally, we can add external supporting documents such as Microsoft Excel or Word documents to our library. The topics we will discuss with regards to Report Viewer for Management Reporter are as follows:

- Overview of Report Viewer for Management Reporter
- Report Library permissions
- Generating reports via Report Designer
- Navigating the Report Viewer interface
- Navigating reports via the Web Viewer

After reading this chapter, we should be able to not only generate board quality financial reports, but also distribute those across our organization in a meaningful and easy-to-understand structure.

Overview of Report Viewer for Management Reporter

Report Viewer for Management Reporter is the component of Management Reporter that is used to display our generated reports and organize them into a meaningful structure, thus making it easier for our users to find the reports they are looking for. It is important to note that we will discuss the Report Library often in this chapter, and this is not to be confused with the Report Viewer. The Report Library is simply the repository that stores and organizes our objects. The Report Viewer is the tool itself.

Before we proceed with generating our reports, it is important to discuss how security permissions work in Report Viewer for Management Reporter. This will ensure that we don't create an issue where either one of our users or ourselves don't have proper permissions to view the reports that have been generated.

Report Library permissions

Security in the Report Library works a little differently than it does in the Report Designer, but it also builds upon what we've already seen. As we learned in the previous chapter, security is broken out by roles. The roles function differently in the Report Viewer than they do in the Report Designer. Let's take a quick look at what permissions the roles provide in the Report Viewer:

- **Administrator**: Administrative users have full access to all viewing tasks
- **Designer**: Users assigned this role may delete, edit, create folders, rename reports and folders, export reports, and view authorized reports
- **Generator**: Users assigned this role may delete, edit, create folders, rename reports and folders, export reports, and view authorized reports within the Report Viewer
- **Viewer**: Members of this role may view and export reports as authorized

In addition to the access provided by the Report Designer roles, the Report Viewer permissions can be set to view, create, edit, and delete objects in the Report Library. This includes objects such as folders, reports, report versions, and any external supporting documents. These can be granted to Management Reporter users or groups by a user with administrative privileges. We also have the ability to restrict access to certain versions of a report, such as the final version that is to be published to our executive board. Let's look at the four permissions for the Report Library as follows:

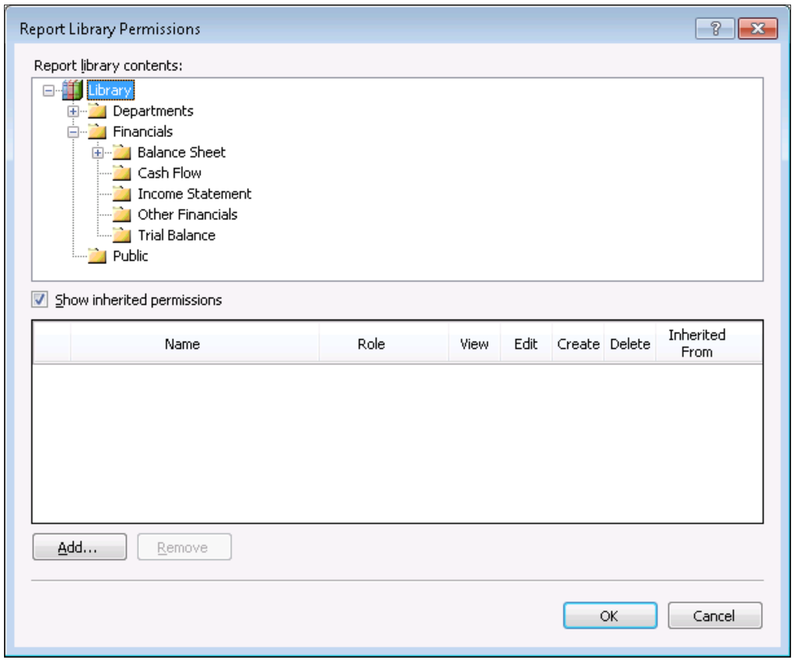
- **View**: This allows users to view the contents of any folder, report, report version, and external supporting documents

- **Create**: This allows users to create new subfolders and add supporting documents to the Report Library or folder
- **Edit**: This allows users to edit folder names, report names, and supporting documentation
- **Delete**: This allows users to delete folders, reports, report versions, and supporting documents

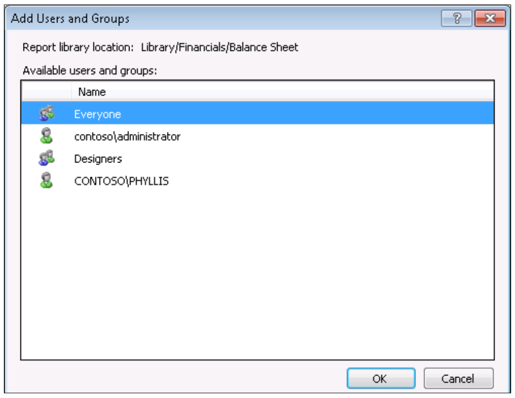
By default, all new users are added to the `Everyone` group. This is a system group and cannot be edited or deleted. There is also a `Public` folder that the `Everyone` group has access to. Users can generate reports to this location if they are having trouble generating to a location with restricted access. This `Public` folder can be renamed or deleted if it is not going to be used, but it is highly recommended to keep it and just create a new structure to use.

To assign or change the library permissions, we browse to the **Report Library Permissions** window by performing the following steps:

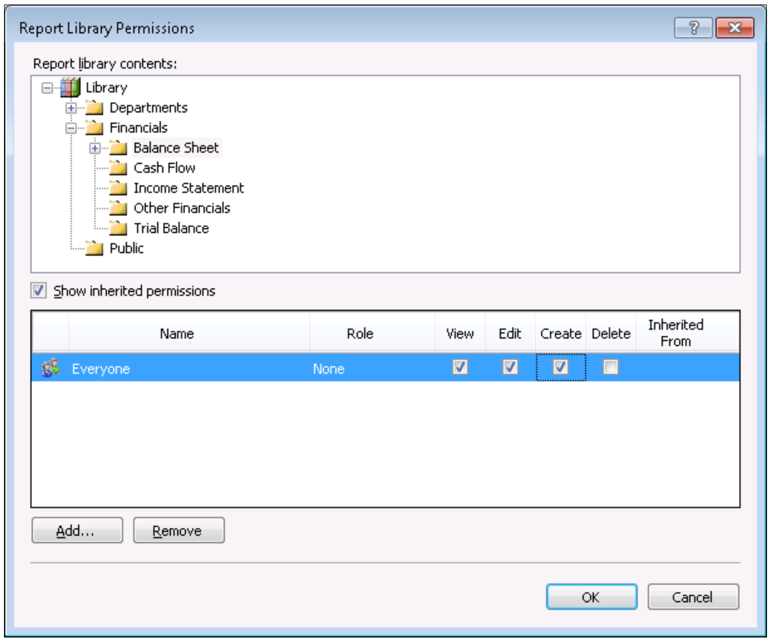
1. Launch Report Viewer for Management Reporter by navigating to **Start** | **Programs** | **Microsoft Dynamics** | **Management Reporter 2012** | **Report Viewer**.
2. Click on **Tools** to open the **Report Library Permissions** window, as shown in the following screenshot:



3. On the **Report Library Permissions** window, we can browse through the objects in the **Library** node including folders, reports, and external documents.
4. Locate the object we wish to grant permissions to, highlight it, and click on **Add** to open the **Add Users and Groups** window. In this case, we have the `Balance Sheet` folder selected, as shown in the following screenshot:



5. On the **Add Users and Groups** window, select the user or group that we wish to add to the object, and click on **OK**. Note that the object location is listed at the top of the window. In the following screenshot, we've selected the `Balance Sheet` folder:



Name	Role	View	Edit	Create	Delete	Inherited From
Everyone	None	✓	✓	✓		

6. Back on the **Report Library Permissions** window, we will now see the user or group that we selected. If that user or group is assigned a Management Reporter role, that will be displayed in the **Role** column. We can also select or deselect the **Show inherited permissions** option to show us which permissions have been inherited from parent objects.
7. Select any combination of **View**, **Edit**, **Create**, and **Delete** that we wish this user or group to have permission to perform.
8. Click on **OK**.

Report Library permissions can be as simple or as complex as we need for our organization. We want to take this into account when designing our final solution and library structure.

Generating reports via Report Designer

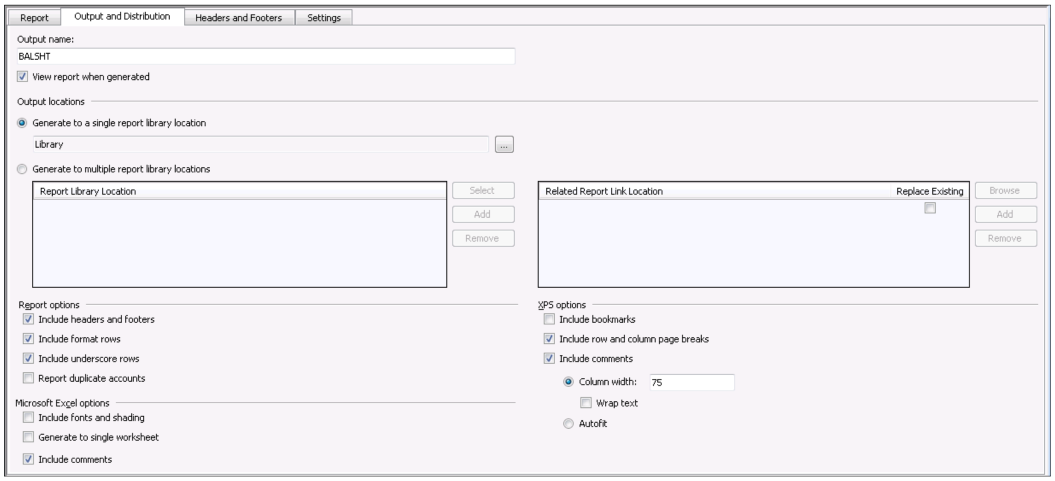
In the previous chapter, we covered the process of creating row definitions, column definitions, and reporting trees. These building blocks are then combined together to create our final report through the use of the Report Definition, which we also covered in the previous chapter. We will now go through the final step of actually generating our report.

To generate a report, we need to have a Report Definition where the row and column definitions have been assigned. Depending on our reporting needs, we may also assign a reporting tree definition to our report definition. Once we've designated the specific building blocks to use with our report definition, we can work our way through the various tabs on the report definition to control the final output of our report. In this chapter we will focus on specific fields that are on the **Output & Distribution** tab, but for more information on the various other tabs refer to the *Report Definitions* section in *Chapter 8*, *Designing Financial Reports in Management Reporter*.

On the **Output and Distribution** tab of a Report definition building block, we can designate the output options for our report. Management Reporter 2012 offers several options for distributing our generated reports.

We can generate our Management Reporter reports to a single Report Library location or to multiple locations with multiple forms of delivery. We can even publish links to our reports in network locations or in a Microsoft SharePoint document library.

The Report Viewer and reports generated to a single report library location will be the focus of the remainder of this chapter. With this output option selected, our view will look like the following screenshot:



Output name: BALSHT

[✓] View report when generated

Output locations

(●) Generate to a single report library location

Library ...

( ) Generate to multiple report library locations

Report Library Location

Related Report Link Location	Replace Existing
	☐

Report options

[✓] Include headers and footers

[✓] Include format rows

[✓] Include underscore rows

[ ] Report duplicate accounts

Microsoft Excel options

[ ] Include fonts and shading

[ ] Generate to single worksheet

[✓] Include comments

XPS options

[ ] Include bookmarks

[✓] Include row and column page breaks

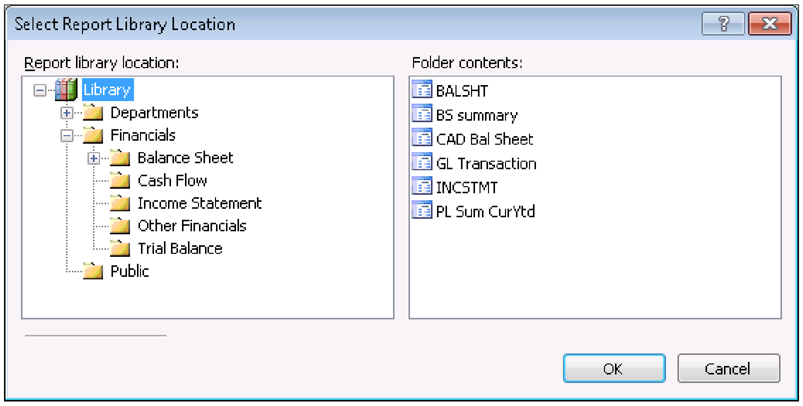
[✓] Include comments

(●) Column width: 75

[ ] Wrap text

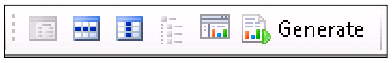
( ) Autofit

Notice the ellipsis (**...**) button next to the Report Library location field. Prior to generating our Trial Balance report, we can click on the ellipsis (**...**) button to open the **Select Report Library Location** window, as shown in the following screenshot:

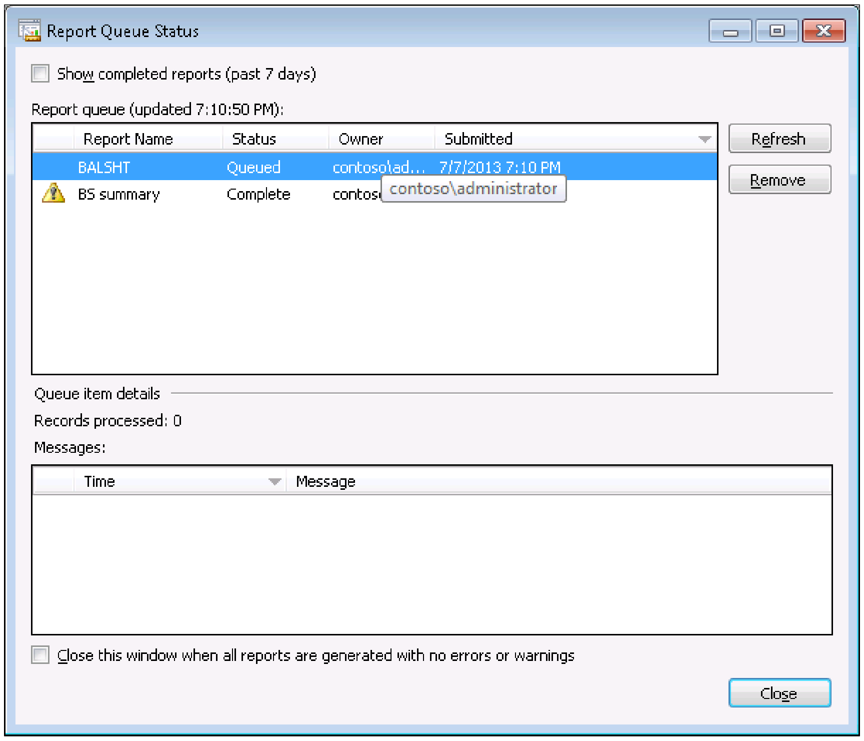


This window can be used to control where our newly generated report will appear within the Report Library. If we aren't ready to assign the report to a folder just yet, we can select to generate our report to the `Public` folder under the **Library** node. Later, if we want to change the location of our report, we can move it to another folder.

Assuming that we've reviewed and made changes to the settings on the Report Definition tab, we are now ready to generate our report to the Report Viewer. To do this, click on the **Generate** button on the Management Reporter toolbar. This button is highlighted in the following screenshot:



The **Report Queue Status** window will appear, and users will see their report listed in the queue. Depending on its status in the report generation process, we will see various statuses such as **Queued**, **Processing**, and **Complete**. An example of this can be be seen in the following screenshot:


Report Queue Status
Show completed reports (past 7 days)
Refresh
Remove
Close this window when all reports are generated with no errors or warnings
Close

Report queue (updated 7:10:50 PM):

Report Name	Status	Owner	Submitted
BALSHT	Queued	contoso\ad...	7/7/2013 7:10 PM
BS summary	Complete	contos	

contoso\administrator

Queue item details

Records processed: 0

Messages:

Time	Message

Once the report has finished processing, the **Report Queue Status** window can be closed manually or will automatically close if we have checked the option in the bottom-left corner of the window. Our newly generated report will open in the Report Viewer, assuming that we also have permission to view this report.

Navigating the Report Viewer interface

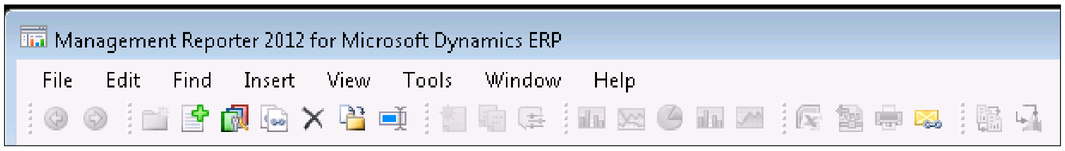
Although the Report Viewer is completely separate from the Report Designer, the look and feel of the two is very similar. The Report Viewer is the simpler of the two to understand in terms of navigation, commands, and options. In this section, we'll explore some of the key menus and concepts that are only available to us in the Report Viewer.

Overview of the Report Viewer interface

Just like we saw in the Report Designer component, the Report Viewer component has the look and feel of Microsoft Dynamics GP and other Microsoft applications. Across the top of the interface, we find our familiar menu bar, with menus and commands that extend our Report Viewer experience. Down the left-hand side of the interface, we find the **Navigation Pane**. If not, it can be enabled for viewing.

Menu bar

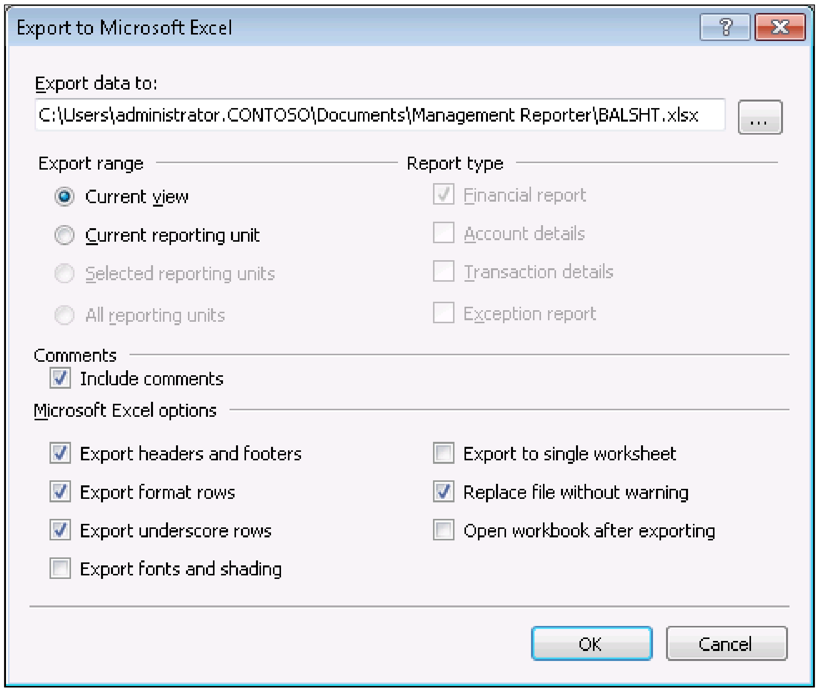
Although menu names such as **File**, **Edit**, and **Tools** may be familiar to many of us, some of the commands within these windows are specific to the Report Viewer. These commands are found at the top of the interface, in a view similar to the following screenshot:



File

As we can open multiple reports for viewing in the Report Viewer, we can use the **File** menu to quickly close the current report. Or, if we want to start from scratch, we can navigate to **File** | **Close All** to close all reports that we have opened in the Report Library.

From the **File** menu we have access to the **Print**, **Export**, **Send an Instant Message**, and **Send Link using Email** commands. Depending on the setup of printers on our environment, we can print the selected report to a printer or even to a PDF file if we have a PDF printer setup. If we decide to use the plugin for Microsoft Lync, we will have the option to send the report to one of our Lync contacts; how cool is that. The **Export** command allows us to convert our existing report into another output type. If we choose to create an Excel report, the **Export to Microsoft Excel** window appears, as shown in the following screenshot:


Export to Microsoft Excel

Export data to:
C:\Users\administrator.CONTOSO\Documents\Management Reporter\BALSHT.xlsx

Export range
- [●] Current view
- [ ] Current reporting unit
- [ ] Selected reporting units
- [ ] All reporting units

Report type
- [✓] Financial report
- [ ] Account details
- [ ] Transaction details
- [ ] Exception report

Comments
- [✓] Include comments

Microsoft Excel options
- [✓] Export headers and footers
- [✓] Export format rows
- [✓] Export underscore rows
- [ ] Export fonts and shading
- [ ] Export to single worksheet
- [✓] Replace file without warning
- [ ] Open workbook after exporting

Once we've identified our settings and selected to Export to Excel, our new Excel file will be deposited in the location found in the **Export data to:** field.

Note that the export to Excel functionality only works for Excel 2007 and higher versions.

Edit

The **Edit** menu provides us with a limited set of commands that may behave differently depending on the kind of object we have selected in the Report Viewer interface. The three main commands in the **Edit** menu (**Delete**, **Move**, and **Rename)** are typically available on the command menu that appears upon right-clicking on the selected object in the interface.

The **Move** command is particularly useful when we want to rearrange the location of our generated reports in the various folders we've created in the Navigation Pane. The **Rename** command is also useful when we need to modify the name of our selected object.

Find

The **Find** menu is where we will locate our common window's find functionality as well as our Go To command, which allows us to quickly switch back and forth between the Report Library and the Report Data sections of the interface. Alternatively, we can click on the buttons in the Navigation Pane.

Insert

From the **Insert** menu, we have access to the **Insert External File** command which is extraordinarily useful in the event that we need to add supporting documentation in the form of non-Management Reporter reports to the Report Viewer. These external files, which can include PDF or Excel format, can be combined with Management Reporter reports to create report packages that our management team can use to access all financial statements and supporting documentation in one location. We can also create a quick chart of a row or column of data or add a comment to our report.

View

From the **View** menu, we can control which toolbars and windows will appear in our report. For example, if we want to remove the Navigation Pane to provide more space for the selected report, we can navigate to **View | Navigation Pane** to clear the checkbox and the Navigation Pane from the interface.

Additionally, we can select **Show Versions** to see additional versions associated with the selected report.

Tools

From the **Tools** menu, we can modify our connection to the Management Reporter database, open the **Report Library Permissions** window, or click on **Options** to open a window from which we can modify a few settings related to Management Reporter. In the **Options** window, we can control the default file location to which exported files will be sent when using the **Export** commands.

Window

If we have multiple reports opened in the Report Viewer interface, the **Window** menu can help us easily switch back and forth between these reports. Commands on this window are only available if more than one report is open in the Report Viewer.

Help

As one might expect, the **Help** menu offers us the ability to open the Management Reporter help functionality. From here, we have a searchable index of topics related to Management Reporter, and this should be one of the first stops for anyone seeking to learn more about Management Reporter related topics.

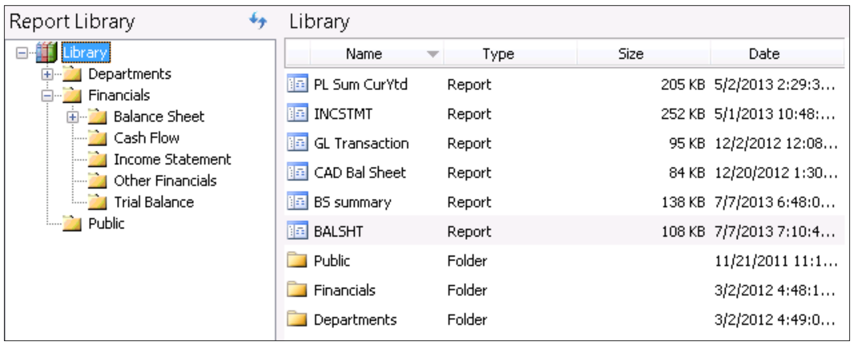
Also from the **Help** menu, we can access the **About Management Reporter** command to see more information about the version of Management Reporter we are currently using.

The Navigation Pane

When compared to the Navigation Pane in the Report Designer, the one in the Report Viewer is much simpler. Here, we only see two buttons: **Report Library** and **Report Data**. While viewing the contents of a report, we will have the **Report Data** button selected. At all other times, we can expect to see the **Report Library** button enabled.

With Report Library selected, we will see a list of the folders that we have created underneath the **Library** node. By clicking on a folder or node, we will see a list of reports available in the **Library** section of the Report Viewer interface. Reports not assigned to any of our folders will appear when we have the **Library** node selected.

When compared to that in the Report Designer, one of the most unique aspects of the Navigation Pane in the Report Viewer is that we can create subfolder groupings. This allows us to create multilayer folder structures and grants us a greater degree of organizational control over our reports. The following screenshot shows an example of what a Navigation Pane would look like if we arranged our reports in a report type structure:



Name	Type	Size	Date
PL Sum CurYtd	Report	205 KB	5/2/2013 2:29:3...
INCSTMT	Report	252 KB	5/1/2013 10:48:...
GL Transaction	Report	95 KB	12/2/2012 12:08...
CAD Bal Sheet	Report	84 KB	12/20/2012 1:30...
BS summary	Report	138 KB	7/7/2013 6:48:0...
BALSHT	Report	108 KB	7/7/2013 7:10:4...
Public	Folder		11/21/2011 11:1...
Financials	Folder		3/2/2012 4:48:1...
Departments	Folder		3/2/2012 4:49:0...

As we've previously said, the ability to organize reports in folders and subfolders introduces us to new possibilities for storing and organizing our reports. Some of the common ways in which we can group our reports are as follows:

- **By Company**: In multicompany environments, this can be extraordinarily useful to us in the event that we want to store our report versions on a company by company basis.
- **By Report Type**: Others may choose to store generated reports based on the type of report that has been generated. For example, variations of a balance sheet financial statement can be stored in a `Balance Sheet` folder.
- **By Time Frame**: In an effort to organize reports over a length of time, others still may choose to organize reports by Time Frame. For example, at the end of each year, all report versions can be moved into a folder with a name corresponding to the recent year.

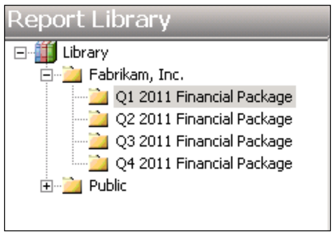
In addition to creating single level folder structures as previously described, the subfolder functionality allows us to mix and match these groupings for even more control over our reports. In the preceding screenshot, we saw an example of an organization that has grouped reports in a folder structure arranged By Company and By Report Type. Next year, this organization could archive prior year versions of these reports by adding an additional Time Frame layer so that we have a Time Frame-Company Name-Report Type hierarchy.

Folders and their contents can be moved by right-clicking on the object and selecting **Move**, or by dragging-and-dropping the object to its new location in the Navigation Pane.

Inserting external files to create report packages

Now, let's take a look at one of the interesting ways we can leverage Management Reporter to impress the management team members in our organization. By using a combination of folder groupings and the ability to insert external files into the Report Viewer, accounting teams can create report packages to present a full package of financial statements and supporting documentation in one easy-to-access location. Our management team will love the ease with which they can locate, view, and browse the contents of this report package that has been prepared by the accounting team. Depending on how our reports are designed, they may even be able to use additional functionality, such as report drill through to answer their own questions about the source of a particular roll up balance.

In order to accomplish this we need to set up a folder (or folders) that will contain the contents of our report package. We need to put some thought into the best way to store our reports depending on our audience, as well as any security concerns. As we mentioned earlier, we can mix and match a combination of groupings including By Company, By Report Type, or By Time Frame. In this example, we've created a folder structure with a Company-Time Frame format with the time format broken down by the Quarter. The following screenshot shows what this might look like in the Report Library:


Report Library
Library
Fabrikam, Inc.
Q1 2011 Financial Package
Q2 2011 Financial Package
Q3 2011 Financial Package
Q4 2011 Financial Package
Public

Next, we need to add our supporting documentation to the Report Viewer. Let's say we've created the following supporting documents to be included in our report package:

- A Table of Contents for the report package
- A Summary of Performance for the quarter
- An Excel spreadsheet containing supporting information for our Income Statement

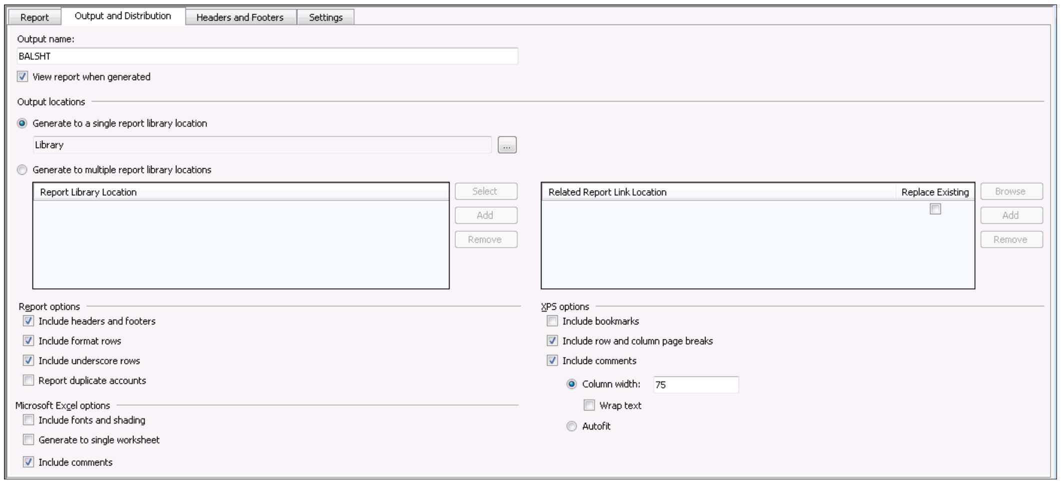
These supporting documents must be added to the Report Viewer with the **Insert External Files** command as follows:

1. In the Navigation Pane, select the folder to which these external files should be added.
2. From the Report Viewer, navigate to **Insert** | **Insert External Files**.
3. In the **Insert External File** window, find the first internal file to be added to the Report Viewer.
4. Click on **OK** to add it to the selected folder in the Report Viewer.
5. Repeat this process for additional supporting documentation.

Once these files have been added to the Report Viewer, they can be treated in a similar fashion to the reports that also exist in the Report Viewer. For example, if we need to move these files to another folder, we can drag-and-drop them in the Navigation Pane or we can right-click on them, and select **Move**. Additionally, we may wish to rename files, and this can be done by right-clicking on the file, and selecting **Rename**.

When documentation, such as Excel or Word files are added to the Report Viewer, these documents are stored as a snapshot of the original document stored outside of the Report Viewer. In other words, if we make changes to the original document (outside of the Report Viewer), then we need to reinsert the file again to ensure that we have the most up-to-date file in the Report Viewer.

Now that we've added our supporting documentation, we need to add our reports to the same folder. If we've already generated the reports from the Report Designer with a Management Reporter output type, we simply need to find these reports in the Report Viewer and make sure that they are added to the appropriate folder(s) containing our report package. If not, we may need to generate up-to-date versions of these reports from the Report Designer. While in the Report Designer, don't forget that we can control the destination folder for our newly generated report by selecting the ellipsis (**...**) button next to the Report Library location field in the **Output and Distribution** tab of our Report Definition, as shown in the following screenshot:



Output name: BALSHT

[✓] View report when generated

Output locations

● Generate to a single report library location: Library

○ Generate to multiple report library locations

Report Library Location

Related Report Link Location — Replace Existing [ ]

Report options
- [✓] Include headers and footers
- [✓] Include format rows
- [✓] Include underscore rows
- [ ] Report duplicate accounts

Microsoft Excel options
- [ ] Include fonts and shading
- [ ] Generate to single worksheet
- [✓] Include comments

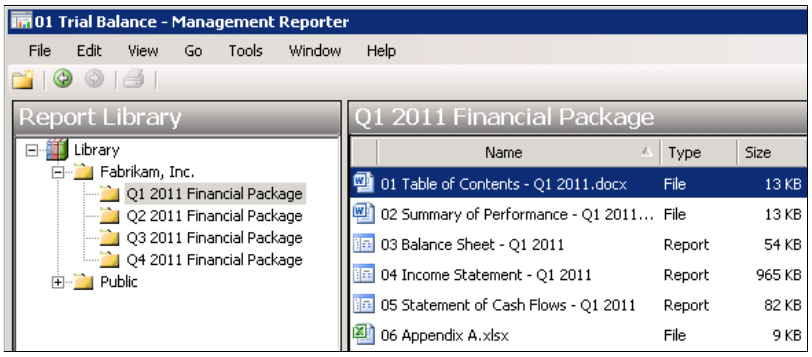
XPS options
- [ ] Include bookmarks
- [✓] Include row and column page breaks
- [✓] Include comments
  - ● Column width: 75
    - [ ] Wrap text
  - ○ Autofit

Now that we've added both our supporting documentation and the appropriate reports to our folder(s), we only need to complete a few more steps before we can share this report package with our management team.

Since we've likely pulled our documentation from a number of different sources, and because the **Library** pane of the Report Viewer interface typically sorts documents in a given folder by name, it's likely we have an unordered list of documents in our folder. To fix this and present a more ordered appearance to our report package, we can add a numerical prefix to each report that corresponds to the order in which we want it to appear in our folder. For example, if we want our Table of Contents to appear first in the order, we can use the rename functionality to insert the prefix **1** at the beginning of the object name.

If we have fewer than 10 documents in our package, then it's acceptable to use single digit prefixes to order our documents. If we plan to include 10 or more documents within the same folder, then it's recommended to use a two digit prefix, even for the first nine documents. If we don't do this, then the 10th document will appear in the sort order between the first and second documents. By adding a zero to the beginning of the first nine documents, we ensure that they will remain sorted properly when the 10th and additional documents are added.

The following screenshot shows our folder with all of our 2011 Q1 documents sorted in their appropriate order and ready for viewing by the management team:


01 Trial Balance - Management Reporter
File
Edit
View
Go
Tools
Window
Help
Report Library
Library
Fabrikam, Inc.
Q1 2011 Financial Package
Q2 2011 Financial Package
Q3 2011 Financial Package
Q4 2011 Financial Package
Public
Q1 2011 Financial Package

Name	Type	Size
01 Table of Contents - Q1 2011.docx	File	13 KB
02 Summary of Performance - Q1 2011...	File	13 KB
03 Balance Sheet - Q1 2011	Report	54 KB
04 Income Statement - Q1 2011	Report	965 KB
05 Statement of Cash Flows - Q1 2011	Report	82 KB
06 Appendix A.xlsx	File	9 KB

Our last step is to grant the appropriate permissions to this folder to our management team. At a minimum, we need to grant the Report Viewer access to our management group for this folder.

Once these steps have been completed, we will have a fully functioning, professional-looking report package that our management team can access at their next meeting. They can, at their convenience, browse through each of the reports and supporting documents to gain a clearer picture of the organization's financial health.

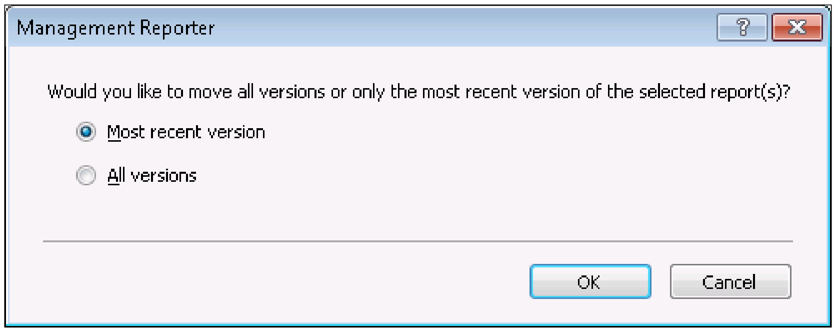
Understanding version control

One thing we should be aware of when using the Report Viewer is that we are viewing a specific version of a particular report. Anytime we select to generate a report from the Report Designer, we are creating a new version of that report specific to the data in our company database at that point in time.

If we continue to generate a report to the same output location, then the existing report at that location will be updated to reflect the newer version. Management Reporter will, however, save the older version of the report as part of the version history of the new report. This continues to occur for successive regenerations of the report.

To see the version history for a particular report, we simply need to right-click on the report in the Report Viewer library, and select **Show Versions**. We can then select and open previous versions to see what has changed since that version.

When we attempt to move a report that contains version history, we are prompted with the following screenshot on whether or not we'd like to move just the **Most recent version** or **All versions** of the report options:



Would you like to move all versions or only the most recent version of the selected report(s)?

[✓ Most recent version]
All versions

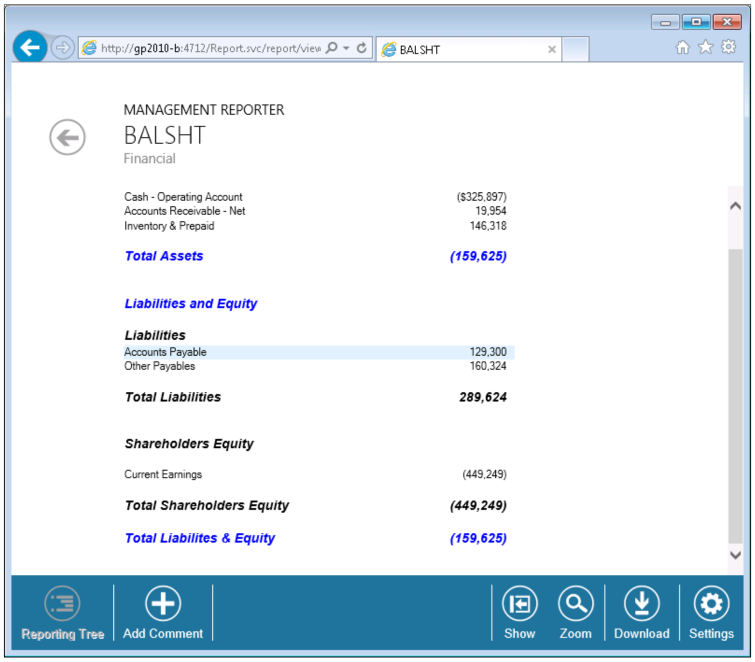
Moving the most recent version of the report creates an orphaned report at the new destination. Successive regenerations of the same report will no longer update the version history of the report we've just moved. Instead, it will create a new report at the designated location and successive regenerations there will create a new string of report versions.

Navigating reports via the Web Viewer

With the release of Management Reporter 2012, when deploying our reports to either a network share, a SharePoint document library, or through e-mail these reports can be viewed via a web browser, without using the desktop viewer. This feature is eloquently called the **Web Viewer**. This feature does require Microsoft Internet Explorer 9 or 10, Mozilla Firefox 14.01 or newer, Google Chrome 22.0.1229.79 or newer, or Apple Safari 5.1 or newer. It's always best to check the current Management Reporter System Requirements on PartnerSource or CustomerSource for changes to these requirements. Some of the other highlights of the Web Viewer are the following:

- Ability to view Management Reporter reports without having a full client installed
- Allows quick export to Excel or XML
- Has the same security as the Report Viewer
- Modern interface with full drill down and tree functionality

If we open one of our reports in the Web Viewer, it will look like the following screenshot:



MANAGEMENT REPORTER
BALSHT
Financial

Cash - Operating Account	($325,897)
Accounts Receivable - Net	19,954
Inventory & Prepaid	146,318
Total Assets	***(159,625)***
Liabilities and Equity	
Liabilities	
Accounts Payable	129,300
Other Payables	160,324
Total Liabilities	***289,624***
Shareholders Equity	
Current Earnings	(449,249)
Total Shareholders Equity	***(449,249)***
Total Liabilites & Equity	***(159,625)***

Across the bottom of the window are the buttons that control additional commands, which are as follows:

- **Reporting Tree**: It displays information for individual units of a reporting tree. We can only view units we have been given access to and this option will only be available if our report is generated with a reporting tree.
- **Add Comment**: It opens a dialog so we can add a comment to the current row we have selected in the report.
- **Go To**: It allows us to view specific rows or levels of a reporting tree. This option will only be available if we generate our report with the Financial level detail.
- **Show**: It opens a submenu that will give us the option to toggle on or off the Charts and Comments and/or the Header and Footer.
- **Zoom**: It increases the magnification of the report data.
- **Download**: It downloads the report as an XPS, Excel, or Management Reporter Report Viewer file.
- **Settings**: It provides information about Management Reporter.

Summary

In this chapter, we expanded our knowledge from the previous chapter of designing our reports and explored generating our reports and storing them in a meaningful manner in the Report Library. We then moved on to navigating the Report Viewer for Management Reporter tool, describing how to use it to organize and store our reports as well as attach those many supporting documents that we all have for our financial reports such as Microsoft Word tables of contents and Microsoft Excel spreadsheets detailing out certain allocation calculations.

In the final chapter, we will bring all of these tools in our reporting toolkit together and show how they can address the challenges and theoretical issues described in the first chapter, by linking them to the practical application of the tools that we have explored throughout this book.

In addition, the final chapter will provide a side-by-side comparison of the pros and cons of each reporting tool in the context of the reporting challenge it may or may not fulfill, with an emphasis on which tool would be the best fit.

10
Bringing it all Together

As more and more reporting tools are made available for Enterprise Resource Planning solutions, such as Dynamics GP, it becomes more of a challenge for us to select the right one for a given circumstance. Every organization has a unique set of requirements when it comes to analyzing its data, and our goal should be to ensure that we select the right reporting application to meet those requirements. Additionally, each tool has its trade-offs, and we must always be aware of these while we are selecting the best one to meet our needs. In the end, we don't want to try to fit the proverbial square peg in a round hole by forcing a reporting tool to meet challenges that simply don't align well with its strengths.

In this chapter, we'll answer some of these questions regarding the selection of the right reporting application to meet our challenges by:

- Recapping what we've discussed in each of the last nine chapters
- Discussing each of the reporting challenges we covered in *Chapter 1, Meeting the Reporting Challenge*, in light of what we've learned about each reporting application
- Taking a look at the future of reporting with Dynamics GP

By the end of this chapter, we should be well versed in the art of selecting the right reporting tool(s) to meet the reporting needs of our organization.

Looking back at what we've covered

How often do we try to force a reporting tool to meet a challenge it is not well suited for? For example, it's not uncommon to witness a user export a **Sales Transaction** SmartList containing thousands of rows into Excel, use Excel formulas to summarize the values from that record set and plug the end result into a single cell into yet another spreadsheet, only to have to repeat this process for several other values. All of this for a single Summary Sales Report by Division that's out-of-date the minute the data is exported to Excel. Rather than continue to waste productive hours watching a SmartList export to Excel on a regular basis, it may be time to consider other reporting tools, such as Excel Reports, which refresh SmartList data directly to Excel or **Analysis Cubes.** Excel or Analysis Cubes provide support for Excel-based dashboards to meet the needs of our report user.

In the end, this book will reduce potential frustrations caused by using the wrong reporting application for the challenges at hand. We have covered the capabilities of each reporting tool. With these in mind, you should be able to apply the right reporting tool to the unique challenges faced by those in your organization.

Reporting trends and challenges

Too often, an end user approaches us with a request for a report, and before he or she has even finished telling us about the requirements, we've selected the reporting tool that we're going to use. While most of us are experienced enough to make such a quick judgment, we should remain careful not to jump to hasty conclusions. These hasty decisions can lead to selecting a reporting tool that does a poor job of meeting the end user's requirements. Furthermore, this can lead to difficult problems down the road when the user finally gives in and says "Enough! This is not working for me!"

So what do we need to think about when a user approaches us about a new set of reporting requirements?

To answer this, *Chapter 1*, *Meeting the Reporting Challenge*, uncovers some of the more common challenges in the process of selecting, creating, and maintaining new reports and reporting applications. As successful report developers and consultants, we must be prepared to consider the myriad of challenges associated with report development so that we can design the most efficient and effective report possible.

Reporting tools for Dynamics GP

Before we can cover any specific reporting tools, we needed to discuss some basic concepts for reporting in Dynamics GP as seen in the rest of the chapters. In *Chapter 2, Where Is My Data and How Do I Get to It?*, we took a look at the techniques that can be used on the database and SQL server level, as well as those that can be used from within the GP application.

With a basic understanding of these concepts, we moved on to the following reporting tools:

- SmartLists and SmartList Builder
- Excel Reports and Excel Reports Builder
- Report Writer, including Word Templates
- Predefined **SQL Server Reporting Services** (**SSRS**) reports and Business Analyzer
- Analysis Cubes for Excel
- Management Reporter

While a host of other reporting applications can be used with Dynamics GP, we selected these for our discussion because they are among the most well known and widely used by GP users. For the most part, each one is representative of several other third-party reporting tools in terms of capability and usefulness. As long as we focus on understanding how the pros and cons of these can be used to meet our challenges, we can extend the same concepts to other similar tools.

Viewing our reporting tools in light of the reporting challenges

Now that you have been given a brief overview of what has been covered thus far, we now turn our attention to reviewing our reporting tools in light of the challenges that we covered in the first chapter. You should remember from *Chapter 1, Meeting the Reporting Challenge*, that each challenge contains diametrically opposing conditions. For example, as we consider the intended audience of our report, we must decide if our report should meet the needs of the day-to-day operations personnel or if it should be designed for external stakeholders who may be interested in a larger time range than a day. In comparing two reporting tools side by side, we may discover that one reporting tool perfectly meets the set of requirements imposed by our day-to-day operations personnel; however, the other tool is more suited for external stakeholders. We will find that other tools fit more towards the middle of these two extreme conditions.

Our goal is to select the reporting tool that has the best ability to meet our required condition(s). Of course, in the perfect world of this chapter, we will consider all of our reporting tools against each individual challenge. In reality, however, we will have to prioritize the varying conditions we need to meet with each challenge, and accept some trade-offs in our reporting tool's ability to meet all of our circumstances.

Intended audience

A report almost always turns out useless when it hasn't been designed and created with the eventual end user in mind. It is imperative that we select the right reporting application to ensure our user receives accurate information with the right functionality to meet his or her needs. Failure to do so will lead to wastage of time and resources with little to no return for our efforts.

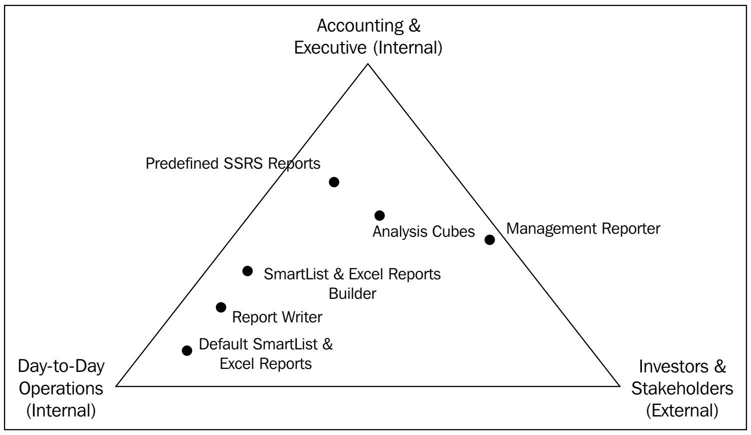
For the most part, our reporting audiences can be classified as either external or internal consumers of our organization's data. External consumers include auditors who require insight into the financial health of our organization, potential investors, and current stakeholders. Internal consumers can range from the operations personnel, such as a warehouse manager, members of the accounting team, as well as members of the executive team.

For external auditors who require insight into our organization's transaction journals, the default reports that come with tools, such as Report Writer and SmartList Builder can easily be modified to provide auditors with the exact information they need. Although these reports are internal to GP, we can convert them to printouts, or in the case of SmartLists, export them to Excel and send the finished document over to the auditor. Other external consumers, such as potential investors and current stakeholders are primarily interested in seeing a summary of our organization's financial statement. This can be provided via numerous types of financial statements and fortunately, **Management Reporter** can be used to create a wide variety of professional-looking financial statements that can then be distributed externally.

One thing to keep in mind about most of our internal users is that they are more likely to operate within GP on a daily basis. We can assume that most of these users already have a user login and are familiar with navigating through the environment. By creating reports with a built-in reporting tool, we can add a new dimension to the user's GP experience without having to introduce the user to another application and another environment. Tools such as SmartList Builder and Report Writer allow us to develop such reports, and users will more than likely appreciate the ease of access that comes with them.

However, this is not to say that all internal users want to see their new reports in GP. It's entirely likely that members of our executive team will not have access to GP, much less the time or desire to navigate through the raw output that comes from SmartLists or posting journals designed in Report Writer. Instead, our executive team members want to see the summarized information in a single, easy-to-reach location. Reporting tools such as the predefined SSRS Reports Library, Analysis Cubes for Excel, and Management Reporter are all alike in that they can provide these team members with a central repository of summary reports that do not require a GP user login to view.

For a better, more visual understanding of the relationship between internal and external report consumers and the tools that they would most likely benefit from, take a look at the following figure:


Accounting & Executive (Internal)
Predefined SSRS Reports
Analysis Cubes
Management Reporter
SmartList & Excel Reports Builder
Report Writer
Default SmartList & Excel Reports
Day-to-Day Operations (Internal)
Investors & Stakeholders (External)

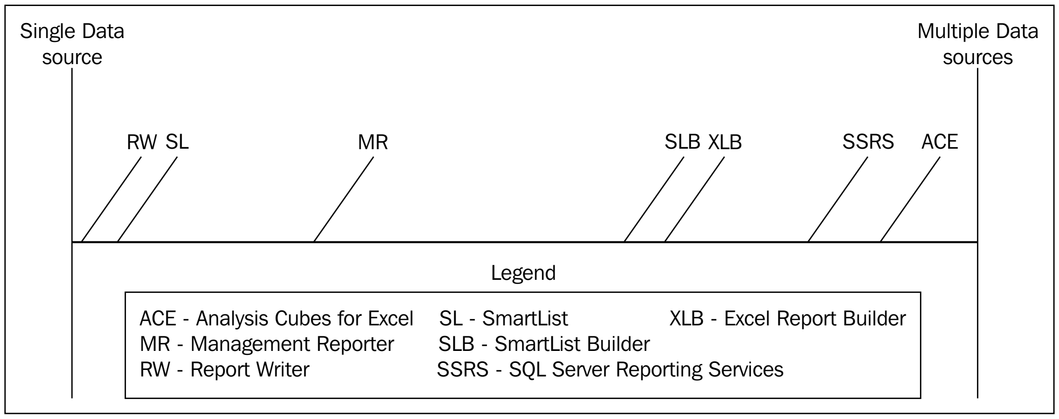
Data sources

Since the end result of the kind of reports we've discussed in this book is to present the end user with data from our ERP solution, we should be concerned with the source of that data. At first glance, the answer seems obvious—if we're trying to provide data from our ERP environment, shouldn't we pull it straight from the underlying company database itself? Unfortunately, the answer is not always that simple. We must be aware of potential performance impacts against our company databases that might result from our reporting processes. Additionally, even though our focus is primarily on GP 2013, we must keep in mind that data can reside in other, non-GP related data marts as well. These separate data sources may exist due to other proprietary applications. If our end user wants to include information from separate data sources in his or her report, we need to be sure that we have a reporting tool that can access this data and use it in conjunction with what we are pulling from our GP database.

By their very nature, some of the reporting tools that we've discussed in this book can easily be used to combine data from multiple data sources. One tool that lends itself to this kind of reporting is the Analysis Cubes for Excel product. The data warehouse provides a readymade location in which data extracted from multiple data sources can be transformed for consistency purposes and then loaded into the warehouse. The transformation piece of this is critical as one of the biggest challenges of using disparate data sources is finding a way to join the data together through some common key value(s). For example, one data source may refer to customers via a unique eight digit numeric customer code, whereas the other data source may refer to the same customer with a ten digit alpha-numeric code. In order to combine these two sets of data, we need to find a way to relate the eight digit codes to their corresponding ten digit codes from the other data source.

Although Analysis Cubes for Excel offers a unique environment in which data from multiple data sources can be staged, other reporting tools that we've discussed can also pull data from multiple data sources. This is largely due to the flexibility they offer in the form of using SQL queries to extract data for reports. For example, in SSRS, we can use SQL queries to create the data set from which our report is built. These SQL queries offer us flexibility to pull data from multiple tables, databases, and servers. Unlike Analysis Cubes, however, the transformation required to create commonality between these different data sources must be done entirely within the SQL query. While this can be achieved, it is usually the domain of those with more advanced technical skills, and it can also cause an unnecessary drain on system resources each time the report is generated. SQL queries can also be used to generate data for SmartList Builder and Excel Report Builder reports. Although this does create some limitations, they may or may not be an issue for our end users.

The following figure shows where each reporting tool falls in the spectrum of reporting against a single data source or multiple data sources:



Latency

As discussed in *Chapter 1*, *Meeting the Reporting Challenge*, latency deals with the idea that depending on the reporting tool selected, the data in our report may lag a bit behind the actual data that exists in our ERP environment. Although it is likely that users will profess a need for real-time data in all of their reports, in reality, this is not always easy to achieve. While it may be relatively easy to display up-to-the-minute transactional data with certain reporting tools, this is not always the case. In order to provide reports capable of analyzing large data sets and offering insight into trends in our data, we cannot always rely on simple reporting tools that return nothing but raw transactional data back at us. Instead, we must use reporting tools that can analyze large amounts of data and provide a summary of trends and movements within our organization. To do this requires a blend of time, money, and resources.

The effort to provide reporting tools that can truly help us aggregate and make sense of our large data sets is increasingly gaining importance. For example, reporting tools, such as Management Reporter allows us to create a structure in which bits of data can be summarized into meaningful financial statements that provide both inside and outside stakeholders a look at the health of our organization. With such an analytical tool we can see the big picture or the trends in our data, even though this data originally came to us in the form of individual journal entries. As we saw in *Chapter 8*, *Designing Financial Reports in Management Reporter* and *Chapter 9*, *Viewing Financial Reports in Management Reporter*, the process of creating financial statements in Management Reporter is twofold as follows:

- Creating the structure or building blocks of our report
- Generating the content of our report

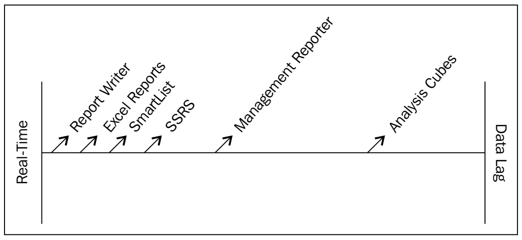
Although this is usually a fairly quick process, made even faster by the fact that we are using only standardized information from our general ledger, it still requires some time and resources to ensure that our reports are up-to-date. Here, then, we see an example of a great reporting tool that provides up-to-date data without too much of a drain on resources; however, we are limited to viewing this data in the rigid structure of a financial statement.

But what happens when users tell us they need analytical capabilities outside of the GP Financials module? These users will likely benefit from the flexible and analytical capabilities of Analysis Cubes for Excel, but they will have to accept a trade-off when using the cubes to report against other modules. The trade-off comes in the form of extraordinarily large time delays between the data found in the cubes and what actually exists in production databases.

This occurs because Analysis Cubes stages data in a data warehouse and then processes the data into cubes for faster querying by the end user. The end result means users will spend less time sorting through copious amounts of transactional data and more time analyzing organizational trends that can be used to provide organizational advantage. While this data may not be up-to-the-minute, this should not scare our report users away from this tool. Instead, we need to remind them that trends found in our data are often the result of a length of time far greater than the latency of our data. In fact, the cost and impact of reacting to every minute shift in a trend analysis caused by the addition of last minute data to our data set can often have far-reaching and costly consequences on our overall business plan.

Finally, let's not overstate the importance of tools, such as SmartList, SmartList Builder, and Excel Reports Builder in the face of far more sophisticated reporting applications, such as Management Reporter and Analysis Cubes. SmartList, SmartList Builder, and Excel Reports Builder do a great job of providing transactional data quickly and accurately. Because they don't offer much in the way of analysis, they do the best job of any reporting tool of providing up-to-the-minute data without a major drain on costs or performance. They also provide search mechanisms that make it easier to navigate to a specific transaction among hundreds and thousands of others. While this is probably not much use to an executive team member, they can be incredibly valuable to the day-to-day operations personnel, such as customer service representatives who must quickly navigate to a customer's order within GP. Using SmartList, for example, our user can quickly find the right entry and use the drilldown functionality to actually open the detail window directly in GP. So, while presenting real-time data can be achieved, we must determine from our end user whether the goal is to see purely transactional data that can be retrieved quickly and easily, or if there is a larger goal of using a report for trend analysis or to provide a summary of a large set of data.

As we've seen, the opposing concepts of this reporting challenge consist of real-time reporting versus potential data lag. While most of the tools we've discussed tend to lean towards real time reporting, we can see from the following figure that some tools do a better job of providing it than others. In the following figure, we have placed Management Reporter in the middle of the two extremes as some users may not have report generation capabilities; therefore, they must use the Report Library to view previously generated data:



Formatting and presentation

Often, we don't really put formatting and presentation very high on the priority list when we begin to develop our reports. In fact, it is usually the last thing we consider. With this being said, it is actually a critical piece to consider when developing our reports.

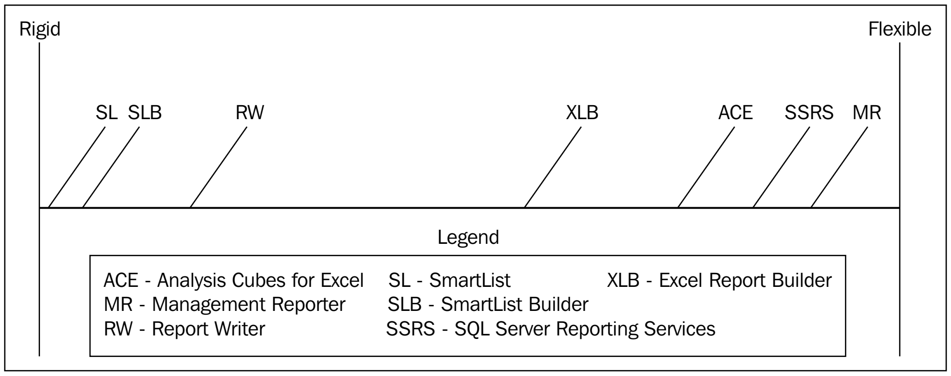
In most cases, our formatting and presentation needs fit with our intended audience. For example, if this is a financial statement that is being presented to auditors or our board of directors, it should be professional looking and easy to follow. Functionalities such as company logos, graphs, proper page numbering, and reports laid out in sections can all add to the presentation of our reports. Reporting tools such as Management Reporter, Analysis Cubes for Excel, and SSRS are all excellent tools to accomplish this type of formatting.

As we have seen in the previous chapters covering these tools, we can easily add logos and graphs as well as groupings so that the report has a logical flow and is easy for our audience to follow. For instance, we can use SSRS to send professional looking invoices to our customers with our company logo included. Management Reporter and Analysis Cubes for Excel allow us to lay out our data in a meaningful manner. We can use Management Reporter for creating properly formatted financial statements; while in Analysis Cubes for Excel, we can create company dashboards for our executive team.

Tools, such as Report Writer, SmartList Builder, and Excel Report Builder; while great at pulling raw data, have very limited formatting options available. These tools are best used when presentation is not necessary and the report will be primarily used by a user who needs to crunch through raw data and analyze things such as an auditor needing a list of all journal entries posted in a given year.

The challenge of selecting a reporting program that provides support for graphics, multiple fonts, and the ability to easily modify the appearance of the report can be broken down diametrically into tools that are either very flexible or are structured and rigid. Flexible reporting tools provide us with the ability to easily modify the appearance of a report and can support the inclusion of graphics and logos. On the other hand, rigid reports are difficult to change, and we should not expect them to provide us with much in the way of formatting and clean presentation.

As the following figure shows, the reporting applications we've discussed cross the gamut of rigidity versus flexibility:



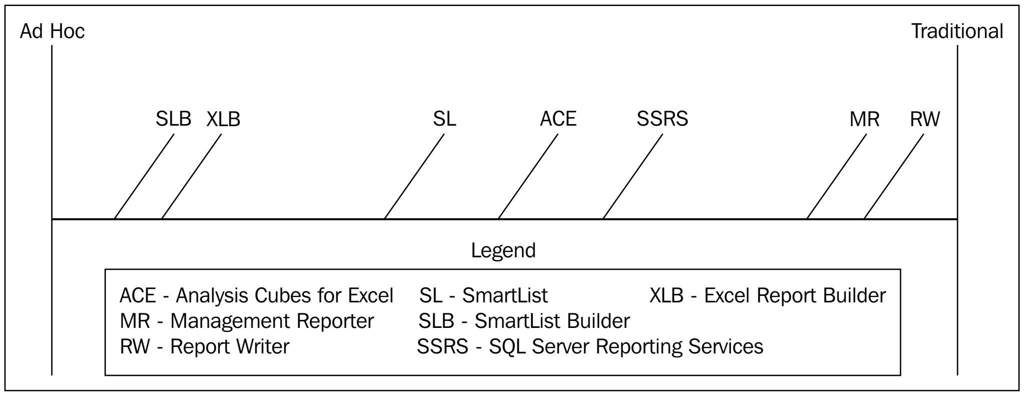
Ad-hoc queries versus traditional reports

In report design, one of the major challenges is deciding whether to use ad-hoc report or some type of traditional report. As we mentioned in *Chapter 1, Meeting the Reporting Challenge*, this decision can be made easier when we learn from our users what the output of the report needs to be and how often that report needs to be generated. We can use the example of an auditor requesting journal entries for the year as something we would consider ad-hoc. We are not going to save that report and run it on an ongoing basis. However, if our Sales Department needs a list of all open sales orders, that would lend itself more towards some type of traditional report in that it could be run every day, every week, and so on. This would give our Sales Department a method of analyzing what type of backlog they have or which products are generating the most demand.

Most of the reporting applications contain elements of both ad-hoc reporting and traditional reporting. The challenge is figuring out when to use which tool in the right circumstance. While reports created through SmartList Builder and Excel Report Builder can be saved and used over and over again, they are typically best suited for ad-hoc reporting. The same thing can be said for SSRS and Analysis Cubes for Excel. Although they can be used to do some level of ad-hoc reporting; typically, they are better fashioned for creating traditional reports that will be used over and over again with intervals.

In addition to this, another thing to consider is the output. Do we need to e-mail our report? Does the reporting tool allow us to export to PDF? Is there an ability for us to subscribe to a report and have it delivered automatically? These are all questions we should answer before deciding whether to use an ad-hoc report or a traditional report.

As the following figure shows, we see that some reporting tools, such as SmartList and Excel Reports Builder are reporting tools through which temporary reports can quickly and easily be generated. If we want more traditional reports, such as financial statements or sales invoices, we can use tools, such as Management Reporter or Report Writer to accomplish this objective.



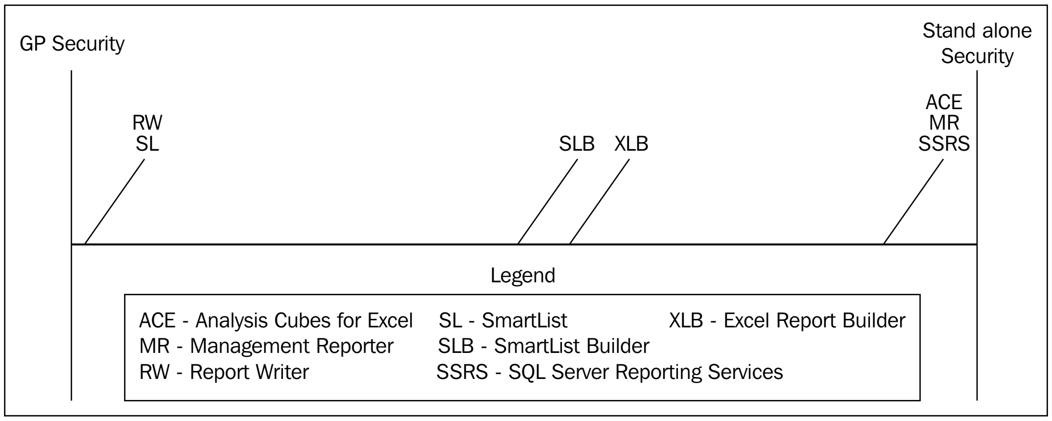
Security

Many of the reporting tools discussed have some type of security model. In deciding on the most effective tool to use for a particular report, we need to be aware and take care to thoroughly think through what type of security model we will need. As we mentioned in *Chapter 1*, *Meeting the Reporting Challenge*, depending on the end user of our report, certain security will need to be set.

Reporting tools, such as Report Writer, SmartList Builder, and Excel Report Builder can easily be controlled from within the Dynamics GP application and can therefore be managed by a Dynamics GP Administrator. SSRS, Analysis Cubes for Excel, and Management Reporter have a different security model in that they use the user's network logins or SQL server accounts. This can lend itself to needing the support of an organization's IT department or some other power user who has the appropriate permissions.

In addition to being able to have the proper user access for the eventual end users of our reports, we also need to think about what permissions are needed for us as report developers utilizing each of these reporting tools so that we can access the necessary data to develop the reports. All of these considerations must be made while keeping in mind the proprietary and sensitive nature of the data.

In general, we can break down security as that which is controlled through the Dynamics GP application and that which is maintained via some separate application. In the case of Report Writer and SmartLists, security is maintained entirely within the Dynamics GP application. But, as we see in the following figure, security for reporting tools, such as Analysis Cubes and SSRS is maintained outside the Dynamics GP application, even though the data found within each tool is based on Dynamics GP data:



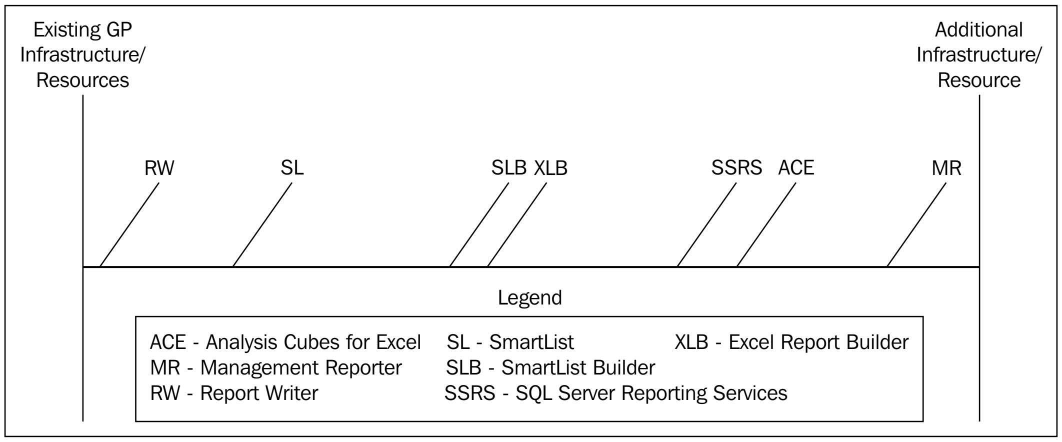
Network access and general IT infrastructure

As we have covered each of these reporting tools throughout this book, we have only briefly discussed the IT infrastructure necessary for each tool's installation and use. While determining which tool is best to use, we must ensure that the IT infrastructure can support it. The last thing we want to do as a report developer is design reports that bring an organization's infrastructure to a crawl or even worse, bring down a server completely.

Although some tools, such as SSRS and Analysis Cubes for Excel can work on the same server as the Dynamics GP application or SQL server, they usually work much more efficiently when they are on a dedicated reporting server.

SmartList Builder and Excel Report Builder, due to their tendency to extract large amounts of raw data through the GP application, may require us to increase RAM, disk space, and other system resources on our servers.

Although the general IT infrastructure may vary from company to company, we can generally break this challenge down to reporting tools that can fit within the existing infrastructure required for Dynamics GP, and reporting tools that require additional resources such as new servers or additional SQL Server components. For example, organizations utilizing only the core GP modules are not likely to need SSAS and SSRS components installed on their servers. Larger organizations, on the other hand, may want to take advantage of these components and can expect to require more in terms of additional infrastructure and resources. As the following figure shows, some reporting tools fit well within the general GP infrastructure while others will require a greater investment in resources:


Existing GP Infrastructure/ Resources
Additional Infrastructure/ Resource
RW
SL
SLB
XLB
SSRS
ACE
MR
Legend
ACE - Analysis Cubes for Excel
MR - Management Reporter
RW - Report Writer
SL - SmartList
SLB - SmartList Builder
SSRS - SQL Server Reporting Services
XLB - Excel Report Builder

Developer resources

The final, but important challenge we must consider when selecting a reporting tool is: what sort of drain will it have on developers? More than likely, as report developers and consultants, we can consider ourselves under the umbrella of developer resources. Ideally, we would like to select a foolproof tool that anyone can use, thus freeing us up from having to do any work. The reality is that this may be a bit unrealistic, but we do need to consider the skill sets of our end users. We need to determine whether or not our end users will have the technical know-how to work with, generate, and possibly make some modifications to reports. Additionally, if our IT department is small or underfunded, we don't want to select a reporting tool that adds even more in the way of developer interaction for creating and managing reports. While some developer expertise over the reporting environment is needed, we do want to free our developers from devoting extra time to troubleshoot minor problems each time an end user encounters something he or she doesn't understand.

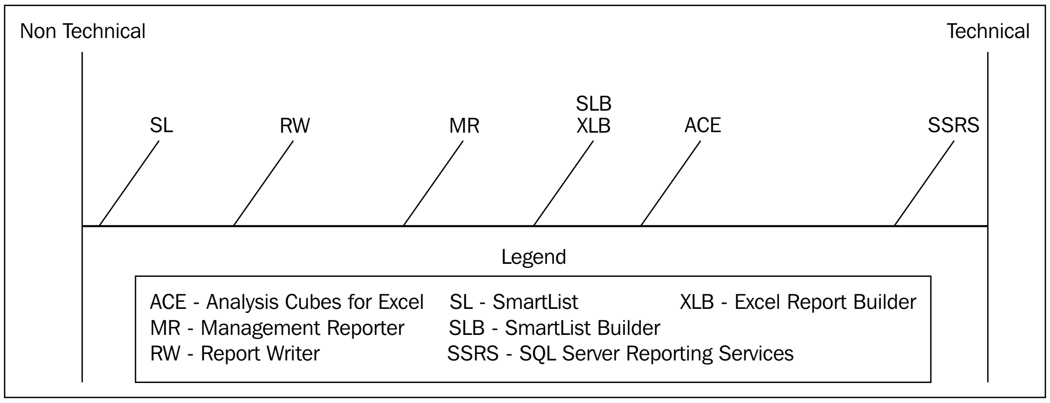
One of the ways that we can minimize the drain on our development staff is to find and train a power user on the use of our selected reporting tool. A power user is someone who possesses a bit of technical knowledge (one tip for finding this person: look for the user in your company who is an Excel guru) and can be a champion for our reporting tool. This power user is someone who has a better grasp of the organization's business processes and can thus relate more easily to end users who will be utilizing our reporting tool. This power user can stand as the gatekeeper to the development team and ensure that only the most pressing questions and concerns are brought to the staff. This person may even be able to handle some of the more basic report requests without the need for assistance from the development team.

As we look back over our reporting toolbox, we see a few tools that require minimum interaction from our development team. For example, with a little training upfront, the basic SmartLists, predeployed Excel Reports, and predeployed SSRS reports can be used by just about anyone who has been trained on Dynamics GP. Depending on the strength of our Dynamics GP power user, our development resource(s) may even be comfortable ceding control of report development via SmartList Builder and Report Writer to the power user. Because these tools are contained within Dynamics GP, our development staff can be rest assured that new silos of data are not being created and maintained elsewhere.

Finally, we have some reporting applications that require extra attention from development resources. These reporting tools include SSRS and Analysis Cubes. Creating custom reports via SSRS and Analysis Cubes for Excel require developers who are experienced with all aspects of SQL including T-SQL, relational databases, SSAS, SSIS, and SSRS. Without some kind of experience in some of these areas, it will be very difficult to deploy, much less maintain, the environments required to support custom SSRS reports and cubes. While it may be possible to deploy the predefined SSRS reports and Analysis Cubes with a modicum of SQL Server knowledge, extending the usefulness of the product will require the kind of advanced knowledge we've just discussed. These two tools, in particular, can also lead us down the slippery slope of creating new silos of information outside of Dynamics GP. Without attention, our organization may become a bit more lax in ensuring that this external data is safeguarded properly and that it does not give rise to a new silo of information.

Some reporting tools do not require users to possess a great degree of technical skill. SmartList and Management Reporter are among the end user's favorites because they can create and generate their own reports with little or no developer help. Many of these users will shy away from reporting applications that require knowledge of technical concepts, such as writing queries against relational databases and effective joining of tables.

As the following figure shows, some tools are well suited to the average Dynamics GP users, while others require the support of a more technical resource such as a developer.



The future of reporting for Dynamics GP

Before we bring this book to a close, let's pause and consider the future of reporting for Dynamics GP. Back when we published the GP 2010 version of this book, and still today in *Chapter 1, Meeting the Reporting Challenge,* of the GP 2013 version, we covered three major trends in reporting as follows:

- Increased flexibility
- Reporting through all levels of an organization
- Increased access to the Report Generation Process

We still continue to expect (and hope) that future reporting in Dynamics GP will be geared towards these trends. Dynamics GP continues to enable users to work smarter and in a more productive fashion. The Dynamics GP 2013 release brought with it new enhancements to reporting. Dynamics GP users at all levels of the organization continue to experience an increasing flexibility and availability of reports that they can generate themselves. As users find ways to cope with the increasing amounts of data in their ERP systems, we expect reporting tools that can easily handle large and varied data sets, such as Management Reporter, SSRS, and Analysis Cubes for Excel. Moreover, this will continue to grow in relevance and importance in the Dynamics GP space. In fact, in an attempt to make it easier for users to view reports that summarize this vast quantity of data, Microsoft released the Business Analyzer tool with GP 2010 R2. Users can quickly cycle through the KPI and charts that are created and stored on their SSRS site for a quick overview of their area.

Moving forward in this ultra-competitive environment, we look to Dynamics GP to continuously provide cutting-edge reporting that empowers users with the necessary information, allowing them to stay ahead of the curve and find a true competitive advantage for their organization.

Improved features with each successive release of Dynamics GP, such as those included with GP 2013, along with updates to standalone reporting tools (such as Management Reporter and additional cubes for Analysis Cubes for Excel), continue to indicate that Microsoft is aware how critical the reporting space is for continued success in the ERP market. As users, we are happy to reap the benefits.

Summary

In the end, we can consider ourselves fortunate that we can choose from such a wide variety of reporting tools for Dynamics GP. Each reporting tool meets a unique set of needs. And, if we are willing to be open and honest about the challenges we face with our organization's reporting requirements, we can select the most appropriate tool for the challenge at hand. While some tools may appear to work in the short term, we must consider the long term during this process as well. As users begin to develop reports and grow more comfortable with using a particular reporting tool, it becomes more and more difficult to instigate a change in tools. If we do opt for the quick fix, with the short-term solution, we may find ourselves devoting even more time and energy in the near future implementing a new set of reporting tools that are more appropriate than our first choice.

Additionally, as we have seen in this chapter, the challenges we face with report development are interrelated. We cannot expect to consider each on its own as the answer to one challenge may dictate the answer to another. For example, as we consider our intended audience, we may realize that we are developing a report for an executive team or external stakeholders. More than likely, these report consumers will not have access to Dynamics GP so we know we do not have to concern ourselves with a reporting tool for which security can be controlled via the Dynamics GP application. Additionally, they are likely to have strict requirements to how a report is formatted, and we find ourselves needing a reporting tool that offers excellent graphics and formatting tools. Even with high-quality visuals, we will need to ensure that we do not stray too far from the traditional layouts these users expect from financial reports.

This book serves as an invaluable tool in the form of a part-reference manual and part-user guide. As we've said quite a few times already, each reporting tool has its place, and we believe this book helps our readers understand what each tool is capable of achieving for our organization. By using this book as a starting point, you should be capable of understanding where to begin and be able to draw upon the resources therein as well as the other countless Dynamics GP resources found throughout the Dynamics community. All of these resources provide reporters with every bit of information required to build accurate and useful reports for our organization.

As the mountain of data in our organization continues to grow, we can be well assured that we have the tools available at our disposal to help make sense of that data. We now have the means at our disposal to help everyone from an Accounts Payable Coordinator to a Chief Financial Officer access the data they need to make effective and practical decisions that will help provide a competitive advantage.

Happy reporting!

Comparing the Dynamics GP Reporting Tools Against Different Reporting Challenges

As we've seen, one of the most critical aspects of report design and development is selecting the appropriate reporting tool for a given set of challenges. But, with so many reporting tools available to us and so many challenges that we are likely to face, it can be difficult to know which reporting tool is the best one for the job. The following tables summarize this information in a helpful, quick-reference format that can help us answer the question: which reporting tool is best for the current situation?

	SmartList Builder (SLB)	**Excel Report Builder (ERB)**	**Report Writer (RW) and Word Templates (WT)**
Intended Audience	Day-to-day operations personnel; GP Power Users	Day-to-day operations personnel; GP Power Users	Day-to-day operations personnel; GP Power Users
Data Sources	Production Database; Additional customizations may include additional databases	Production Database; Additional customizations may include additional databases	Production Database
Latency	Real-Time; Report must be regenerated to include up-to-date data	Real-Time; Report must be regenerated to include up-to-date data	Real-Time; Report must be regenerated to include up-to-date data
Formatting/ Presentation	Very few formatting options; Record-by-record view; Best used for internal reporting	Slightly improved formatting options (over SLB) with Excel; Record-by-record view; Best used for internal reporting	RW has limited formatting options and is better suited for internal use; WT have a more modern look and works well for external reporting
Ad-hoc/Traditional	Ad-hoc; Internal to GP; Larger datasets may be slow in regenerating (when compared to Excel Reports)	Ad-hoc; Reporting in Excel workbook; Instantly refreshable (when compared to SmartList)	Traditional; GP reporting engine (internal); Used for many reports; Reports are not easily modified on an as-needed basis
Security	Integrated with standard GP security	Not integrated with standard GP security; Folder level and data source considerations required	Integrated with standard GP security; WT now offers password protection to post-generation edits
Network Access/ IT Infrastructure	Fits within existing GP application	Requires Microsoft Excel and access to production server/databases	Fits within existing GP application
Developer Resources	New SL can be created by GP Power Users; Advanced SmartLists can use SQL or Extender, which may require more technical skills	Excel Reports are based on SQL views; Modifications to existing Excel Reports require technical knowledge	Can be challenging to learn, but is possible for GP Power Users; Otherwise, additional developers may be required

	SSRS Reports Library (SSRS)	**Analysis Cubes (AC)**	**Management Reporter (MR)**
Intended Audience	Day-to-day operations personnel; GP Power Users; External users with no GP access	Day-to-day operations personnel; GP Power Users; External users with no GP access	Accounting team; GP Power Users; Executive team members (Report Viewer or Web Viewer only)
Data Sources	Production Database; Additional customizations may include additional databases	Refreshable Data Warehouse	Production Database
Latency	Real-Time; Report must be regenerated to include up-to-date data	Dependent on number of times cube load and refresh job is run; Typically run once a day	Real-Time; Report must be regenerated to include up-to-date data; Can schedule publication to SharePoint
Formatting/ Presentation	Capable of producing colorful and professional-looking reports; Multiple font options; External and internal reports; Use Business Analyzer with SSRS for a dashboard feel	Excel PivotTables provide numerous formatting opportunities; PivotTables structure can limit how data appears in spreadsheet	Capable of producing financial reports worthy of a board-room presentation; Supports logos, multiple fonts, and other formatting options; Web Viewer offers a more modern look and feel
Ad-hoc/ Traditional	Traditional; Reports viewed via web browser; Rigid report structure (requires Visual Studio to modify)	Ad-hoc; Easy to create new reports on-the-fly; Existing reports can be refreshed and manipulated quickly	Traditional; Reports viewed in MR Report Viewer or Web Viewer; Board-room style reports; Often saved and re-used at the end of each financial period
Security	Folder-level security within SSRS; Data source security controls access to database(s)	Managed through SSAS; Access can be restricted by cube, dimension or cell	Security is controlled by role for each MR user in the application; Different levels of security such as Content Manager or Viewer; Tree Security can control access to units of a tree via the Web Viewer
Network Access/IT Infrastructure	SSRS requires additional memory; May require a separate reporting server	Data warehouse requires additional space; SSAS requires additional memory; May require a separate reporting server	Requires domain for install; Use DataMart to make report generation faster; Additional database requires some extra space
Developer Resources	Accessing pre-deployed reports requires little technical experience; Modifying or adding custom reports requires knowledge of SSRS	Use of out-of-the-box cubes requires minimal developer resources; Customizing cubes requires knowledge of SQL (DB Engine, SSAS, SSIS)	Based on chart of accounts; Easy to use, even for non-technical users; Knowledge of SQL not required

Index