CHAPTER 3

Planning Solutions

He who fails to plan is planning to fail.

—Winston Churchill

The most important first step in designing a data warehouse (DW)/business intelligence (BI) system, paradoxically, is to stop.

—Ralph Kimball

Planning is fundamentally important in any undertaking. This is no less true in the case of creating a BI solution. Although no plan is perfect and a protracted planning process is the bane of many projects, even simple BI solutions can benefit from some planning. The trick is to find a balance.

In this chapter, we show you techniques to plan a BI solution. We include tips on conducting interviews, data identification, and documenting your plan as well as examples of documentation that is easy to create within a minimal amount of time. We also give you tips on building a BI solution team, defining the different roles the team will play, determining the infrastructure needs of your BI solution, and estimating the cost.

By the end of this chapter, you will have preliminary plans to start implementation on the demo BI solution used throughout this book.

Note It is not our intent to turn this chapter into a project management book, especially because documentation alone isn't going to create the BI solution. We would much rather get to the part where we are creating our project. One problem that repeatedly presents itself, however, is stumbling across a project with no documentation at all. This is likely because many developers do not even know where to begin, from estimating how long the project is going to take to what simple documentation should look like. Most of the books we have found do not discuss this process. Therefore, we decided that we would change that with our book and give you an example of how to create some basic and relatively painless documentation.

Outline the Steps in the Process

When you start a solution, it is important to create a simple outline of what you are trying to accomplish. For example, you may even want to make yourself a flowchart, similar to one shown in Figure 3-1. We have created Figure 3-1 using Microsoft's PowerPoint, but you can use Microsoft's Visio or simply draw one with pen and paper and it will work just as well. What you use to create the outline is not as important as making sure to take the time to do it before you begin developing.

The planning process typically begins by interviewing the client (or whoever the BI solution is being developed for) and documenting what they want. In Figure 3-1, you can see that the interview process has been included at the top of the flowchart.

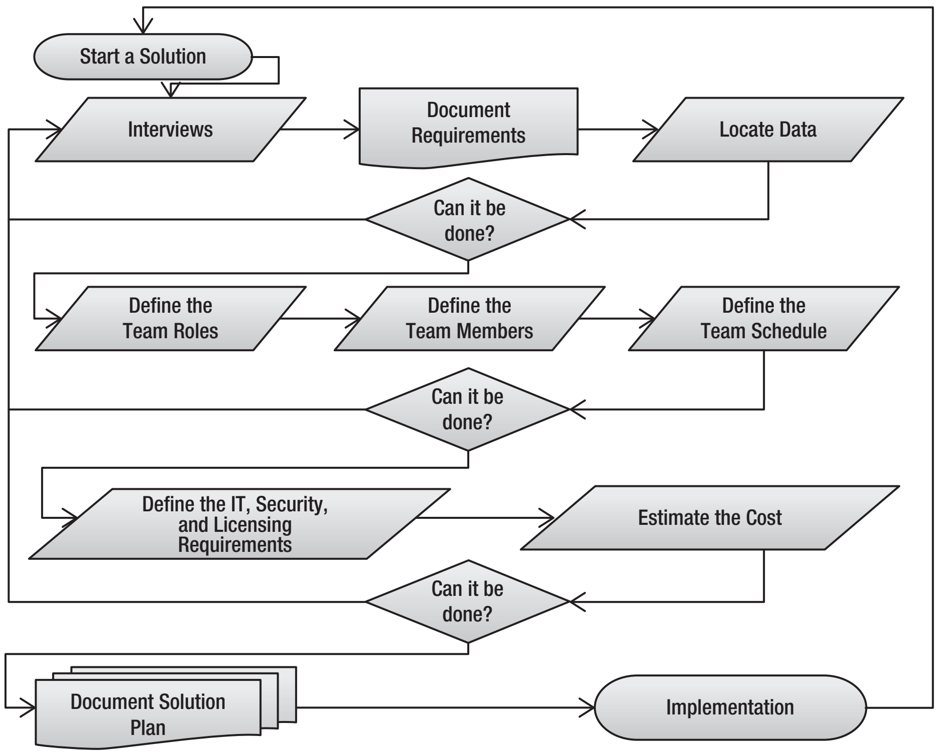


Figure 3-1. *A typical planning workflow*

Next, you need to see whether or not what they are asking for can even be accomplished. A good place to begin is to look for the data required to build the type of solution they are requesting. It is common for clients to want your BI solution to produce information for them that is not supported by their data. This is one of the first things to check for because it is such a common obstacle. If you cannot provide them with the information they want, letting them know early on will save everyone a lot of wasted effort, and it will allow you to work with the clients to revise their expectations while helping to avoid disappointment.

Also, keep in mind other important deal breakers that might kill the solution before it begins. These change for every solution but may include the following:

- Can you complete the solution with the number of staff you have?
- Can you complete the solution in the timeframe allotted?

- Can you complete the solution with the current infrastructure, software, and security?
- Can you complete the solution within the allocated budget?

If the answer is no to any of these questions, then you need to go back to the beginning and redefine the requirements of the BI solution. The process repeats until you have created a plan that balances the customer's needs with the resources at hand. As we have mentioned before, it is best to find out at the beginning whether time spent on the solution will be worthwhile and affordable for the client.

Once you have a working plan, you need to document it. The complexity of the documentation is determined by how complex the BI solution is projected to be. Common items will appear in every solution. The amount of documentation may also depend on how much documentation is required by law or by a company's business practice.

In every case, getting information into those documents is determined by how much you can extract from the objects or events on which you are modeling your solution. Therefore, the best place to start gathering data is through an interview process.

Interviewing

The term *interview* is typically thought of as a meeting where questions are asked. The purpose of the interview itself can vary; some examples are to enable a hiring decision to be made, to provide facts for a story to be written or even to provide leads on an investigation. In this case, considering this type of interview to be like an investigation might be the most accurate means of approaching this for our purposes.

We do not want to limit this process to a single conversation, nor do we want the interview itself to be our only source of information. Clients may not always know how to voice their needs, particularly when they do not know everything you are capable of doing for them.

Do a little research by taking a look at their preexisting documentation or solutions to familiarize yourself with their situation and to help you see potential solutions ahead of time. Past letters and emails with the client may contain facts that the client is assuming you are already taking into consideration or that they simply forgot to bring up at this stage of the process.

Reviewing past correspondence and asking about specifics from concerns they have already voiced can be vital to keeping your client happy. Get as much information as you can while keeping in mind that this process is not limited to verbal communication. The goal is to pinpoint the client's needs and to determine what will best help them with their business. This is also the time to determine the assumptions the client is making about what to expect. Just be sure to avoid treating the client as a hostile witness!

However you plan to get your information, you will want a list of questions answered before you proceed. Here are some that will work for most occasions:

- Why do we need it?
  - What is the goal of the BI solution?
  - What is the hoped-for outcome of the BI solution?
  - Is the project worth the estimated cost?
  - Who will use the BI solution?
- What are we building?
  - What must be in the BI solution?
  - What will be nice to have in the BI solution?
  - What will *not* be in the BI solution?

- How will we build it?
    - Can we release it in increments?
    - Will it need to have all features before it is released?
    - How will the BI solution plan be distributed to the developers?
    - How will progress be monitored?
- Who will we get to build it?
    - Who will be involved in the BI solution?
    - What roles will be needed on the project?
    - Do we have team members who can fulfill those roles?
    - Does the development team have the necessary skills?
- When will we need it?
    - What is the timeframe for the BI solution?
    - When will the BI solution be completed?
    - Who will monitor the progress of the BI solution?
    - Who will sign off on the BI solution completion?
- How will we finish it?
    - Who will document the outcome of the solution?
    - Who will test and approve the BI solution?
    - Who will train the users?
    - How can users submit questions, comments, or requests?
    - What system will be used for bug tracking?

In a perfect world, you will get all of your questions answered. In the real world, you will have to settle for a bit less. Any questions you can get answers to are extremely valuable to the outcome of your BI solution. Each answered question will help you decide how to continue or if you should continue.

Why Do We Need It?

This may be the most important question you can ask. If you do not have a clear understanding of why the BI solution is needed, then perhaps it is not needed at all. It is important that you define this need so that you can validate it against the BI solution you create. A successful BI solution is one where you manage expectations and prove that these were met. If that does not happen, you need to explain why it did not happen.

We recommend looking at what the client is currently using to stay organized and what they are using to help make proper business decisions, as well as how they manage their data. Ask the client what is working well for them within their current system. This can help you determine what is not working for them and what might be added to improve the current system. Be aware that you may have to interview a number of people before you get a clear understanding of what is going on. You can speed up the process by calling a meeting, but in a group setting, only the most vocal members of the group will give you feedback. This can be a productive setting, but sometimes it turns into an opportunity to air past grievances. Take the time to talk with individuals regardless of whether you opt for a group meeting, and you will likely get a much better understanding of what is needed. This is one of those times where technical skills are less important than people skills.

What Are We Building?

Once you have established that the BI solution is beneficial to the client, begin figuring out what is going to be a part of the solution and what is not. Create a list of all the features that your client requested to be included in the solution, and then examine existing documentation and reports for inspiration about what else should be added to the solution. It is common that the interviews with the people involved will not identify all the requirements of the solution.

When your list is created, prioritize what has to be in this version of the solution. One method of doing this is to use a technique known as *four-quadrant prioritizing*. An example of this is shown in Figure 3-2.

Identify what is the value to your client versus the difficulty and cost of an item. When an item provides substantial benefit to users at a low to medium cost during the development cycle, you can classify this item as a Must Have.

If, on the other hand, an item does not provide much benefit to the users and is difficult or costly to implement, then it should be excluded from the project at this time. It does not mean that in the future it will not be included; it just means that right now it is a Not Now item.

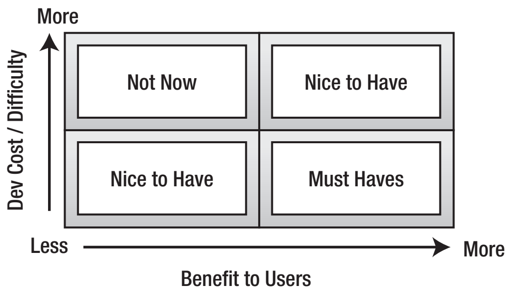


Figure 3-2. *A four-quadrant prioritizing matrix*

If an item provides a lot of benefit to the users but may be costly or difficult, then it is something that will be considered Nice to Have but may not be absolutely necessary to the solution. As such, evaluate these Nice to Haves carefully before including them in the current solution. Once again, this does not necessarily mean it will never be part of the BI solution; it just may not be part of the current version.

Be careful of the final quadrant, in the lower left of Figure 3-2. The items in this quadrant are very easy to implement but provide little to no value for the users. This is a very attractive quadrant because these items will keep you quite busy implementing them and will make you feel as if you are really accomplishing something. In the end, however, it will not benefit your BI solution much, and most clients find it frivolous. Few will feel it pertains to their needs; and consider the fact that if they are paying for your work, they do not want to pay for something for which they did not ask. This particular quadrant is the one that is responsible for the dreaded demon of developers known as *feature creep*. As we say in the field, “When in doubt, leave them out!”

Additional Considerations for Determining What You Will Build

Both the interview process and the examination of current reports and documentation should provide you with a good understanding of what will benefit the users. Identifying the cost involved, however, can be quite a bit more difficult to determine. For example, you may have to consider the ease of use of the current networking infrastructure and hardware that a client is using. Another consideration is the security and compliance restrictions or licensing issues that may be required by the company or by law.

Patient records in the medical industry, for example, must be made available but also held securely. Your ability to access these records may be severely restricted, greatly increasing the cost and difficulty of working with this type of sensitive data. Another consideration is the accessibility of the data you need. You may encounter situations where the data is accessible for only small windows of time. If that is the case, then you must plan accordingly. If accessibility is sporadic, you will find it very difficult to achieve success. You need to find out about this early on and then budget time and costs to account for this.

Another consideration is the conformity of the data. In larger companies, you may find that the same data is recorded in a number of places within the same organization or that the same information is represented in a number of ways. For example, if a company uses descriptors such as Good, Better, or Best for a line of products but then attempts to record the same information in another database using the numbers 1, 2 or 3, you must get a consensus on how this information is expected to be presented in your BI solution before you can continue. And that takes time.

Latency issues, such as the time it takes for the data to change in one part of the organization versus the time it shows up as information in your BI solution is another cost consideration. In projects where a large amount of time or high degree of latency is acceptable, it is much easier to develop a cost-efficient solution. When working with companies where only a small degree of latency can be tolerated, the development costs and urgency required for a completed BI solution may be too exorbitant to continue with your current solution design.

Planning around these obstacles can be quite challenging and may in turn reveal other obstacles such as the following:

- Do the users have the skillsets to extract information from the solution you build for them?
- Will the solution fit the corporate culture of the company with which you are working?
- Do you (or your team) have the skills to manage these types of complications?

As you can see, the interviewing process and taking the time to review the data in depth can determine the success of your BI solution. Taking the time to document a plan will often bring these types of considerations to light.

Determining Your Ability to Complete the Solution

In the end, each BI solution has its own challenges and benefits. You must do your best to evaluate what these are and realize that you will make mistakes just like every other human before you. Do not get bogged down and frozen by indecision. Just do the best you can with the tools you have available. As with all things, your first attempts will contain more errors than your later ones. But, if you never start because you are too afraid of missing something, you will never become experienced enough to know how to avoid most errors. Document your mistakes, learn from them, and move on.

One of the biggest ways to mitigate the number of mistakes you will make is to restrict the complexity of the solution. A solution should always be as complex as it needs to be to get the information the customer needs but as simple as you can possibly make it. The “keep it simple” rule of design will make your life easier and your solutions more profitable.

When you find a solution that cannot be simplified enough for your team to accomplish it, you may want to consider passing up the offer to create the solution. It may be that the solution needs a larger and more experienced team. That may be your team in the future, but perhaps not today.

If you do decide to tackle the solution, be resigned to the fact that changes to the plan are likely. It is imperative that you communicate these changes as they occur during the solution cycle. Communicating with the client and managing the users’ expectations are vital to the success of any BI solution. It is better to disappoint clients expecting a particular feature at the beginning of the project than to have them wait indefinitely for the feature to become available.

Explain to the users that this version of the solution will not include a particular set of features. Then explain to them what will be included. Also let them know when the solution will be delivered. Explain that this delivery date would be unattainable if the features that are being left out were added back in. If the users push back, demanding a particular feature, then it is time to reevaluate your ability to do the BI solution at this current time in the given timeframe. Once again, this evaluation is very important to make at the beginning of the solution. It benefits no one if you start, rack up a lot of hours, and then fail to deliver what you said you could. Be up front and truthful about what you can and cannot do, and your client will appreciate your honesty.

How Long Will It Take to Build?

Close on the heels of the question, "Should we build it?" is the question, "How long will it take to build?" Ideally, the project should begin and end in the shortest time possible. That way, users can have access to the information needed, and you can move on to creating the next version of the solution. However, the definition of the words "shortest time possible" is open to interpretation. Are we talking months or simply weeks? Will a portion of the solution be available sooner, as an incremental release? Or, will users have to wait until all the components are completed before they can start utilizing it?

One means of determining how long it takes to complete a solution is to break it into its constituent parts and assign an estimated number of hours/days it will take to complete those parts. Using rapid application development (RAD) as a model, you can estimate that an employee can accomplish approximately six hours of production in each eight-hour day. (If your team is accustomed to working more or fewer hours per day, then change this number accordingly.) Therefore, for each set of six hours estimated, you can record it as one day of work on the solution. After you have totaled up the days for the different tasks, divide it by the team members you have working on the solution to find an estimated time of completion. On a very small solution, you might estimate something like this:

- Create the data warehouse (6 hours)
- Create the ETL process (24 hours)
- Create the cubes (12 hours)
- Create the reports (12 hours)
- Test the solution (12 hours)

Then add the hours for each project together to give an overall idea of days the solution will take.

- Days 11 = (6 hr + 24 hr + 12 hr + 12 hr + 12 hr)/6-hour days

Of course, we have not taken into account the time it takes to perform the interview and identify the data, plan the solution, obtain the final approval, or release tasks. Therefore, those items will have to be added in as well, in addition to special considerations with your project that are unique and are not listed here.

With regard to the development team, it is unlikely that everyone will work on the same part of the solution at the same time. Ideally, it should be a coordinated effort with the data warehouse, ETL process, cubes, and reports all developed as close to simultaneously as possible. This can be impractical, but when achievable, the turnaround between the planning phase and the sign-off phase is substantially shorter. For example, when the data warehouse is created, a few test entries can be inserted into each table. From these, a cube could be built, and reports can then be created on both the data warehouse and the cube. Keep in mind that these would be preliminary creations. Until you have data within the data warehouse, it is unwise to consider testing and sign-off preparations. However, even with these test values, you may uncover unforeseen deterrents to completing your solution as you build the preliminary prototypes. That is a good thing, because you can let the customer know about these roadblocks early on.

Transparency is crucial to team members' ability to do their jobs efficiently and in coordination with all other stakeholders. Try to set up some mechanism for distributing information about solution progress to all involved. Make sure to monitor the solution's progress and publish these results. We have used basic web pages, bulletin boards, and wiki pages in the past. All of these are simple, inexpensive, and effective.

How Will We Build It?

Building a BI solution can be approached in two ways: a top-down approach or bottom-up approach.

In the top-down approach, the needs of the company as a whole are determined, including all the data that a company uses, with the ultimate goal of creating a solution that will cover all the needs of all users as soon as the entire BI solution is published. In some cases, this is necessary. In these cases, decision makers need all available information through a single viewpoint to make decisions. Interim reports and partial updates are insufficient. Looking at a half-dozen viewpoints is not feasible either. The information from the BI solution is not just an additional tool used in the decision-making process; it is the primary tool. Here, all of the components must be present before the solution is of use.

In the bottom-up approach, the focus is on defining a particular business process and building a solution around this process. The idea is that you can add onto the solution by focusing on a different process in the next version. You continually refine the solution by analyzing the process and incorporating it in the current solution. Your BI solution is only one tool used in the decision-making process, and the solution can be functional even when it is only partially complete.

■ **Tip** The top-down versus bottom-up approaches have been defined as the two paths to choose from. Each is commonly associated with two different developers, Bill Inmon and Ralph Kimball, who are both early (and often competitive) authors on the subject of data warehousing. In the end, both are correct approaches, just not for all solutions.

Over the years, the top-down approach has lost many of its adherents. This is mainly because the approach is often extremely costly and impractical. Therefore, the recommended practice is to use the bottom-up approach whenever possible. Conversely, when the bottom-up approach is impractical, switch to the top-down approach. The trick is asking the question of what is necessary for this particular BI solution. As a consultant, you may not initially know the business you are building the solution for well enough to predetermine what is necessary, so make sure to conduct interviews with this question in mind.

As you may have guessed, in this book we focus on the bottom-up approach, using techniques associated with RAD. (We first mentioned RAD when discussing how to determine the hours it will take to build your solution.) With RAD development, the focus is on supplying the top 80% of the requested requirements. Quite often, this 80% can be accomplished in about 20% of the time it would take to do a full implementation of all requirements. Although this is not always the case, you might be surprised by how often it is accurate. When it is, you are able to supply your customers with a satisfactory BI solution at a fraction of the cost.

Who Will We Get to Build It?

Finding good people to work on a solution is easier said than done. It is a wonderful fantasy to imagine you have all the experience and all the knowledge and all the free time to perform every task the solution requires within a timely manner; however, it is unrealistic to believe that you are capable of accomplishing all of this on your own. What can be done to mitigate the reality of human frailty? In a word: teamwork!

Putting together a team of developers with compatible skills and personalities is a rewarding and lucrative endeavor. In addition to the developers, a team should consist of competent managers, testers, and interested stakeholders. We highly recommend looking for compatible personalities first and sufficient skills second. Although others may argue that we have this reversed, experience has shown that skilled team members who can work together will achieve much more than a group of even better skilled individuals who refuse to cooperate with one another. Failure to build a compatible team costs money and time, two commodities that are often in short supply.

Beyond these soft skills, development teams can fail for two reasons. The first is inexperience with the technology. This failing can be addressed through study and research. For example, learning how to use the SQL Server BI tools by reading and performing the exercises in this book greatly increases your solution success rate. (And telling you this gives us a chance to promote our book!)

The other common failure is harder to prepare for, that is, failure to understand the business process that is being reported. For example, if your solution consists of reporting against the repair rate of computers, you will need to know about the repair centers, the warranty terms, the components that are in the computer, the manufacturers that made the different components, and any internal designation of a group (generation) of components. You will also need to know whether the computers start aging once they are shipped from the factory or whether your business considers their lifetime to begin the first time they are turned on. The users expect the reports to be correct, meaning they answer the question people believe is being asked. If you misunderstand the question even though you provide a legitimate answer, you will not get credit for your answer because it is not the question that was asked.

From this single example, you can see how easy it can be for a solution to fail if your client's business processes are not understood. Once again, transparency is fundamental to the success of your solution. Keep stakeholders informed of your progress, ask for their validation, and—with the client's input—properly determine whether the generated reports do indeed meet the needs of the company. If you can catch these errors early in the development cycle, you will save both time and money.

Stakeholders can be a good choice as project sponsors. A project sponsor is responsible for communicating the progress of your solution to the end users and in turn is able to provide specific feedback to the development team about any misunderstandings. A good stream of communication with a sponsor will affect the final user acceptance at the release of your solution.

Unfortunately, this communication highway can also deliver change requests. Resist the desire to make everyone happy by changing the definition of the solution while it is in progress. If indeed you do find that one of the questions your solution is trying to answer was incorrectly interpreted, then this must be fixed, but you are not looking for new or different questions to answer. New questions can be asked in the next version of your BI solution.

In RAD development, you typically focus on small teams consisting of between six and eight people. Often you try to find team members who have a diverse set of skills so that they are assigned several roles. Some key roles to define include the following:

- Solution sponsor
- Solution manager
- User acceptance coordinator
- Solution planner
- Technical writer
- Programmer writer
- Data warehouse developer
- ETL developer
- Cube developer
- Report developer
- Tester
- Technical trainer

If you research these titles, you will find that there are many different names for these roles as well as additional roles beyond what we have listed. You will need to define your own list depending on the type of solution you are trying to create. Simple solutions do not demand as many roles as more complex ones.

Keep in mind that these represent roles rather than a specific number of people. One person may play many roles. The solution manager may also be in charge of coordinating user acceptance and working with the solution sponsor.

When Will We Need It?

Nobody likes to wait for things that they want, but it is an unpleasant fact that we usually have to. In our industry, every year seems to bring more opportunities for instant gratification. If somebody doesn't answer an email by the end of the day, we can feel slighted. It is hard to remember that only ten years ago people expected to wait for letters to come by mail over a few days' time. Because you cannot change our culture, you will need to plan for the inevitable desire for the project to be done sooner rather than later. Estimate how long the project will take, track its progress, and disclose your adherence or delinquency to the given timeframe.

Key things that you need to define at the beginning of the project include how you are going to monitor the progress and who will sign off on the solution's completion. In a fully staffed team, project managers will be involved in tracking development progress, and you may even have a liaison to interact with customers to get a sign-off document. On smaller projects, team members tend to shoulder many responsibilities that cross these boundaries. When the boundaries of job responsibilities blur, it can be easy to forget which roles are assigned to which team member. Establish these roles early, and make sure that the actions associated with these roles are carried out within the timeframe allotted for the task.

How Will We Finish It?

The completion of a BI solution is just as important as its beginning. In the beginning, you should document what you are trying to accomplish. At the end, you will need to document what has been accomplished. With this document in hand, you will be able to validate that the end users also feel you have accomplished your goal. We have found a 100% buy-in on projects to be an unrealistic target, but the 80/20 rule works pretty well. If you can get everyone to agree that you have accomplished 80% of the most important aspects of the solution and that the other 20% will be worked on in the next version of the solution, chances are that you will have happy clients. Remember that if you set expectations at the beginning of the project correctly, the end of the project is much more likely to be deemed successful.

Like most developers, including one of the authors of this book, you may hate documenting what you are doing. (Of course, this means that the other author enjoys this sort of thing.) Nevertheless, for the sake of everyone involved, even if this task is disliked, it is quite necessary. Creating good documentation can help you track what you did right and what you did wrong on a project. This information is vital for your long-term success as a BI consultant. Think of it as an investment that will pay off over the course of many years, because that is exactly what it is. The good news is that you will see dividends immediately when you begin your next solution and are able to avoid many of the pitfalls that were uncovered in the previous solution.

Another task many developers are not wildly enthusiastic about is testing. Yet testing is fundamentally important to your success. It can be very tempting to skip this portion of the BI solution. Resist this temptation! Undoubtedly, you have found mistakes already in this book. Can you imagine how many there would be if we never had an editor review it?

Even with the best intentions, an author will make mistakes. Without a good editor, these mistakes will be missed and passed on to you, the reader! This analogy mirrors the relationship between a developer and tester. The developer creates the content and the tester reviews and approves it. The feedback given from the tester to the developer makes a project better and more efficient, an investment that will pay off long-term.

The next aspect involves making sure that the end users are trained to use the BI solution you created. Making users happy is very important to finishing your solution. Moreover, their happiness is likely to be determined more by perception than fact. Even if your solution is well-made and consistent, if the client perceives that the program is unwieldy, then all the preparation and hard work you did may be for nothing. Do not skip this step as you complete the BI solution. Take some time to instruct the users how to "use" your solution efficiently, and everyone will be happier.

Striving for perfection can keep you on a never-ending chase for the unattainable. One way to avoid this is to allow users an opportunity to submit questions, comments, or requests. As always, be sure to let users know that the question and comment process is not a means of revising the previously agreed-upon solution. The purpose is to prepare for future revisions of the current solution. If users believe that their comments will change the current version, they will be dissatisfied when they find out it did not. If you let them affect the current solution, you will begin a tug-of-war where you are pulled from the left and the right to the point where you can get nothing done. Although some may be disappointed that their needs are not being met immediately, most will understand that if they are patient, you will address them as soon as possible.

Additionally, you may need someone to give the final approval on a project and advertise that approval. This person is your sponsor. Sponsors are usually managers working for the client. They filter information from your team to the users and from the users to your team. This is similar to how a wedding coordinator works with caterers, the florist, and the rental company. In this analogy, the BI solution team (including you) are the caterers, the florist, and the rental company; the sponsor is the wedding coordinator; and the report users are the happy couple.

You will need to protect the credibility of this sponsor (or sponsors) by making sure that your projects provide clear and accurate information. Not having enough information, not getting it to perform fast enough, and not making it look professional will kill your solution and the sponsor's credibility. Make sure you coordinate with the people approving your project and get their honest feedback before you publish your solution.

■ **Note** Remember, planning a BI solution is very complex. As mentioned earlier, this book focuses on simple solutions that can be used as building blocks for larger solutions. The planning process discussed here is not meant to be an entire course on data warehouse project management. It will, however, get you going in the right direction. Nevertheless, for very large or complex projects, you may need more research to become fully proficient in BI solution planning. You can find many excellent and in-depth articles specific to data warehousing by the Kimball Group at `www.ralphkimball.com`.

"Hey, Wait! I'm a Developer, Not a Manager"

We know, we know . . . This is supposed to be a developer book. So, why are we rattling on so much about project management, documentation, salesmanship, and testing? Well, the fact of the matter is, to be successful in creating BI solutions, you need team members who have these skills. In some cases, you may have sufficient skills in these areas to perform all the tasks yourself. If not, just knowing what these task are will allow you to look for and find team members who can help you in these areas or at least to understand your role on a team.

It is time to move away from all this theory and focus on creating a BI solution as an example. We keep it simple so that you are not spending the next week doing nothing but planning when what you really want is to get started creating a working example. However, we get into enough detail that you are able to get a good idea of what it takes to plan a BI solution. Don't worry, it won't hurt.

EXERCISE 3-1. THE PLANNING PHASE

In this exercise, you take on the role of a consultant performing interviews. You then outline the goals of the BI solution you are being asked to create. There is not much to do in this activity, because for the sake of providing a simple scenario, we have to give you both the questions and the answers. Your job is to follow along to get an understanding of what needs to be accomplished.

Scenario

Your new client (let's call them Publication Industries, Inc.) has a small business. They are booksellers that manage books produced by various publishers and sales to various stores. Although they do not publish books themselves, they do act as an intermediary in selling the books wholesale.

Their current method of managing their data is to track the sales of boxes of books to various stores in a SQL database. They also track additional information about the sales supplied to them from the publishers they represent.

All reports are made in an ad hoc fashion (meaning there was no prior existing system, and there is not a specific standard as to how they collect and manage the information), and clients are finding that they often create reports with conflicting information. The company wants a new business intelligence reporting solution that will provide them with accurate information about their company sales.

Some Questions Asked of the Client

Q. Why do we need it?

A. Our current reporting solution is inaccurate, and we need something better.

Q. What is the BI solution's goal?

A. To provide accurate information about our sales.

Q. What is the hoped outcome of the solution?

A. Our company will end up with one location where all the information is consistent so that everyone who needs it can access it for accurate information.

Q. Who will use the solution?

A. We have managers who read most of the reports. It is rare that all employees will read these reports.

Q. What are we building?

A. We would like several reports involving sales information. Ideally this solution will include a much cleaner database that we can develop our reports from and more accurately track our sales information.

Q. What must be in the solution?

A. Our database is a mess. We need to get it cleaned up and make it consistent.

Q. What would be nice to have in the solution?

A. We know that other companies are using cubes for their reports. We would like to have a cube also.

Q. What will *not* be in the solution?

A. We don't want anything that costs a lot of money to develop or maintain.

Q. How will we build it?

A. We already have a SQL Server that hosts our current database. We would like to use the existing server for all items.

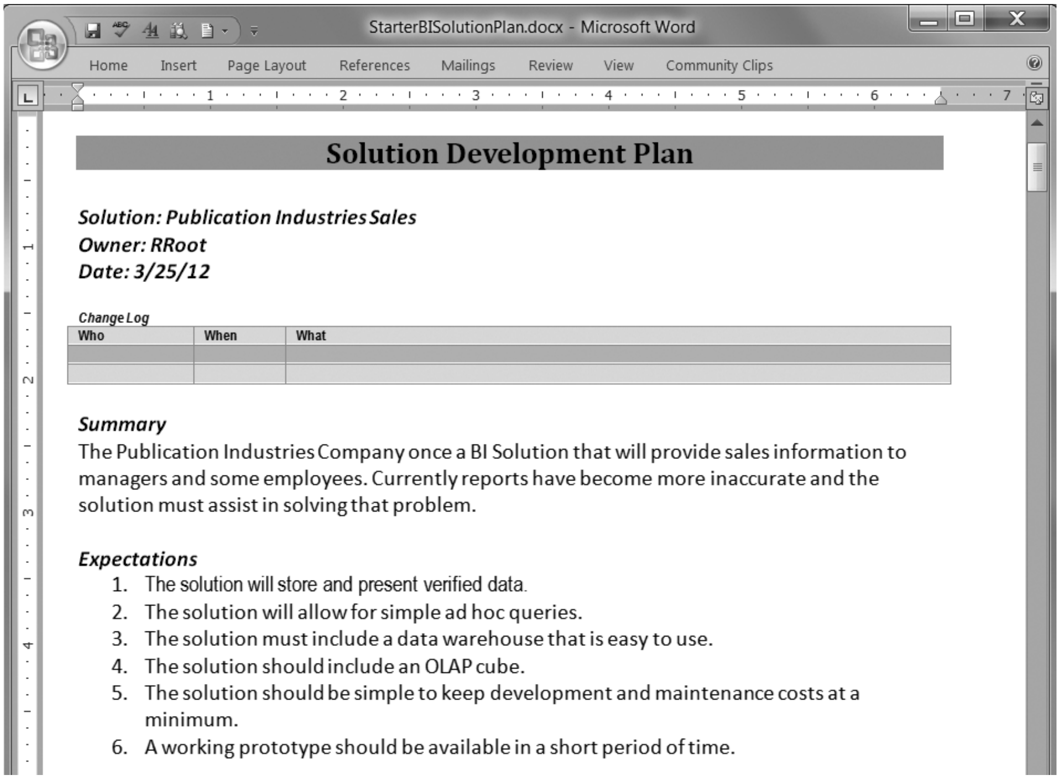
Q. Can we release it in increments?

A. That will be acceptable as long as we start seeing a practical outcome as soon as possible.

In this exercise, you imagined having the role of an interviewer asking questions about the nature of the BI solution you are proposing to create. In the next exercise, we look at defining what will be included in and excluded from the BI solution.

Documenting the Requirements

With the interview process complete, it is time to make some notes about what you found. A simple Word document is sufficient to record these findings. You could simply start by making an itemized list, or you could use a document with tables, diagrams, and section headings. Let's look at an example of this more complex type of document, which you can see in Figure 3-3.



Solution Development Plan

Solution: Publication Industries Sales
Owner: RRoot
Date: 3/25/12

Change Log

Who	When	What

Summary
The Publication Industries Company once a BI Solution that will provide sales information to managers and some employees. Currently reports have become more inaccurate and the solution must assist in solving that problem.

Expectations
1. The solution will store and present verified data.
2. The solution will allow for simple ad hoc queries.
3. The solution must include a data warehouse that is easy to use.
4. The solution should include an OLAP cube.
5. The solution should be simple to keep development and maintenance costs at a minimum.
6. A working prototype should be available in a short period of time.

Figure 3-3. *An example of a "Solution Development Plan" document*

When you look over the documents shown in Figure 3-3, you see it has several sections. (You can also find this document in the Chapter 3 folder of the downloadable content files for a clearer view.)

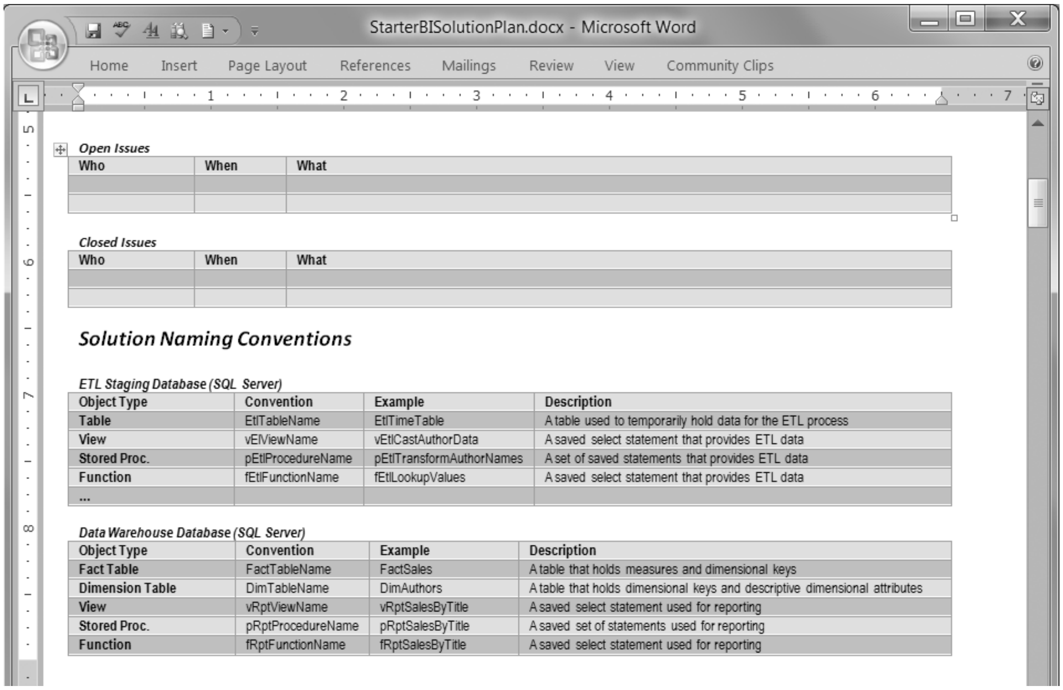
At the top, it identifies the name of the solution, the owner, and the date the document started. Of course, there are other items that could be added, but in an effort to keep things simple, we have reduced this content to just these three items.

Beneath that is a section for recording changes to the document. This area is used after your initial writing of the document, which at this point will not take place until other steps are completed first.

The "Change Log" section is followed by the summary of the solutions you are going to build. This is a parenthetical description of the solution.

Following that is an outline of expectations. It is important to list what will satisfy the client once the solution is completed. You will compare the success or failure of the solution against these expectations once it is completed. Reviewing your successes and failures will allow you to plan for future solutions, allowing you to avoid the same mistakes. This is also a section used by testers to verify that your solution has accomplished its goal.

In Figure 3-4, you can see that the document includes open and closed issues sections. Developers, testers, and managers can use these sections to track questions about the solution and the answers that go with them.



Open Issues

Who	When	What

Closed Issues

Who	When	What

Solution Naming Conventions

ETL Staging Database (SQL Server)

Object Type	Convention	Example	Description
Table	EtlTableName	EtlTimeTable	A table used to temporarily hold data for the ETL process
View	vElViewName	vEtlCastAuthorData	A saved select statement that provides ETL data
Stored Proc.	pEtlProcedureName	pEtlTransformAuthorNames	A set of saved statements that provides ETL data
Function	fEtlFunctionName	fEtlLookupValues	A saved select statement that provides ETL data
...			

Data Warehouse Database (SQL Server)

Object Type	Convention	Example	Description
Fact Table	FactTableName	FactSales	A table that holds measures and dimensional keys
Dimension Table	DimTableName	DimAuthors	A table that holds dimensional keys and descriptive dimensional attributes
View	vRptViewName	vRptSalesByTitle	A saved select statement used for reporting
Stored Proc.	pRptProcedureName	pRptSalesByTitle	A saved set of statements used for reporting
Function	fRptFunctionName	fRptSalesByTitle	A saved select statement used for reporting

***Figure 3-4.** The second page of the "Solution Development Plan" document*

As questions come up, they are recorded in the "Open Issues" section. Answers are recorded in the "Closed Issues" section. When the solution is completed, examining both these sections will help you plan for future solutions.

One section that is often omitted is the one that records the naming conventions used in your solution. Although this section is not strictly necessary, it is more professional looking when all the objects in your BI solution are named consistently. It also makes a difference in the cost of maintenance, as items are easier to find and interpret based solely on their name. For example, should the dimension tables in the data warehouse start with the prefix *Dim*, or should there be no prefix? Most developers have found that including the prefix on the dimension tables makes them easier to find and interpret their use. Therefore, the table that held a list of authors might be named DimAuthors. You would record this preference in the "Solution Naming Conventions" section. Testers of your solution will be asked to verify adherence to the naming conventions.

There are still several more sections to review, but we get to those as we start working on the items that pertain to those sections. For now, let's get organized by completing this next exercise.

EXERCISE 3-2. THE DOCUMENTS

In this exercise, you review a document that describes the BI solution you are creating and copy it from its current location to the solution folder using Windows Explorer.

Before beginning this exercise, you should have downloaded and unzipped the book files from the Apress website. If you did not do so in Chapter 2, please do so now. When the file is unzipped, you will have a folder called `_BookFiles`. Copy this folder to the root of your `C:\ drive`.

1. Navigate to the `_BookFiles` folder on your `C:\ drive`, locate the `Chapter03Files` subfolder, and open it (Figure 3-5).

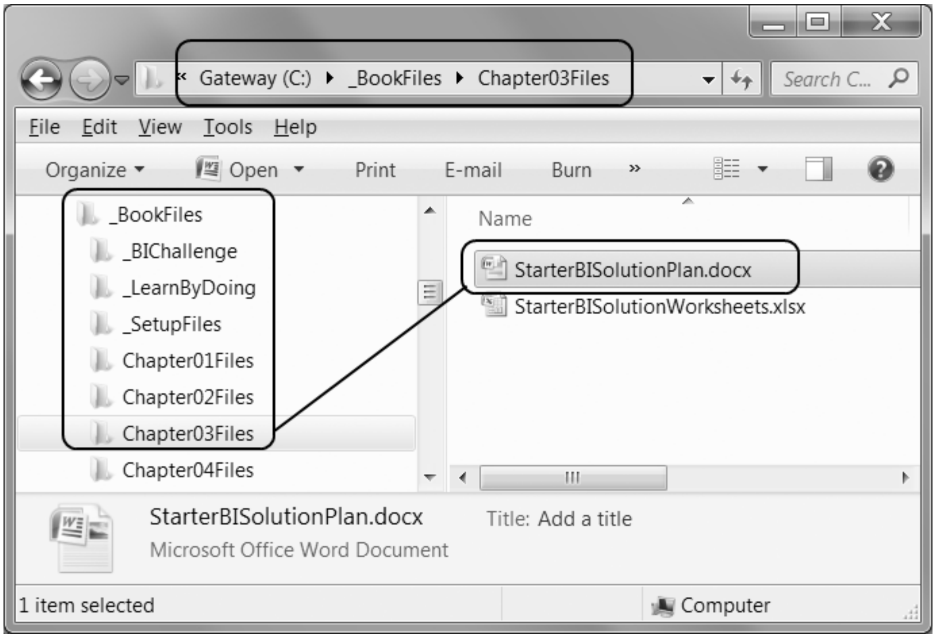

Gateway (C:) ▸ _BookFiles ▸ Chapter03Files
Search C...
File Edit View Tools Help
Organize Open Print E-mail Burn
_BookFiles
_BIChallenge
_LearnByDoing
_SetupFiles
Chapter01Files
Chapter02Files
Chapter03Files
Chapter04Files
Name
StarterBISolutionPlan.docx
StarterBISolutionWorksheets.xlsx
StarterBISolutionPlan.docx
Microsoft Office Word Document
Title: Add a title
1 item selected
Computer

Figure 3-5. *Locating the documents in the chapter folder*

2. Locate the file titled `StarterBISolutionPlan.docx` within the `Chapter03Files` subfolder (Figure 3-5) and double-click it to open it in Microsoft Word.
3. Review the contents of this document noting the location of each section. This document is used in later exercises.
4. Close the document when you have finished reviewing it.
5. Next, while you are still in the `Chapter03Files` folder (`C:\_BookFiles\Chapter03Files`), right-click the `StarterBISolutionPlan.docx` file, and select Copy.
6. Navigate to the `C:\_BISolutions` folder (that you created in Chapter 2) and paste it into the `PublicationIndustries` folder (Figure 3-6).

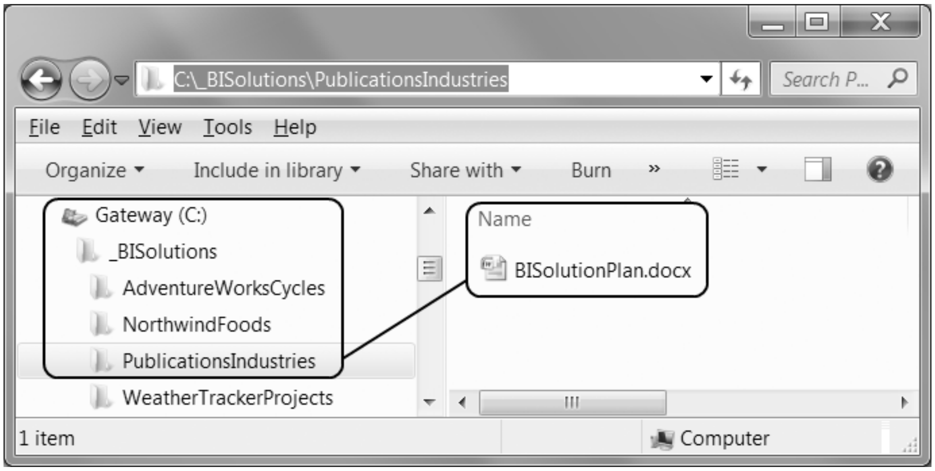


Figure 3-6. *Copying the "Solution Development Plan" document to its new location*

7. Right-click the copied file and select Rename from the context menu. Rename the file `BISolutionPlan.docx`, as shown in Figure 3-6.

In this exercise, you began to build the basic documents needed at the beginning of your BI solution. The next step is to locate the data and verify whether it will be feasible to create the solution. In Exercise 3-4 within this chapter, you add the file to a new Visual Studio solution using Solution Explorer.

Locating Data

Once you have created a basic outline of the BI solution, locate the data necessary to create it. This data may be in many different places, including simple text files, emails, or more typically, a database.

No matter where the data is located, you need to review what is available and decide what to include and what to discard from the current solution. As always, try to keep things as simple as possible while satisfying the requirements of the solution.

In the case of text files, review each one and decide whether the text file as a whole will be part of your BI solution. After you have categorized which files will be included and which ones will not, closely scrutinize each field within the file to determine whether it is valuable to the current solution or whether it should be ignored during this iteration.

In the case of a database, the process is quite similar. First examine the tables available and then decide whether they are important to the current BI solution. Once you have determined which ones are important and which ones are not, closely review each field within the tables in order to decide which of these should be included.

This process is much easier to learn by performing it than reading about it. So, let's do that now in this next exercise.

■ **Note** If you are like us, you want to make your own decisions about something you are creating. Nevertheless, because these exercises do not have a real team or client and because all the chapters in this book have to work together from start to finish, we make the decisions for you. We include information from the fictitious client as we go along, as though you were in communication with them.

We are sorry for the inconvenience and know that you may not agree with some of our decisions or thought processes, but we think you will agree that it is still helpful to see how someone else approaches these tasks. In addition, it will help you understand what we are doing in future exercises of this book.

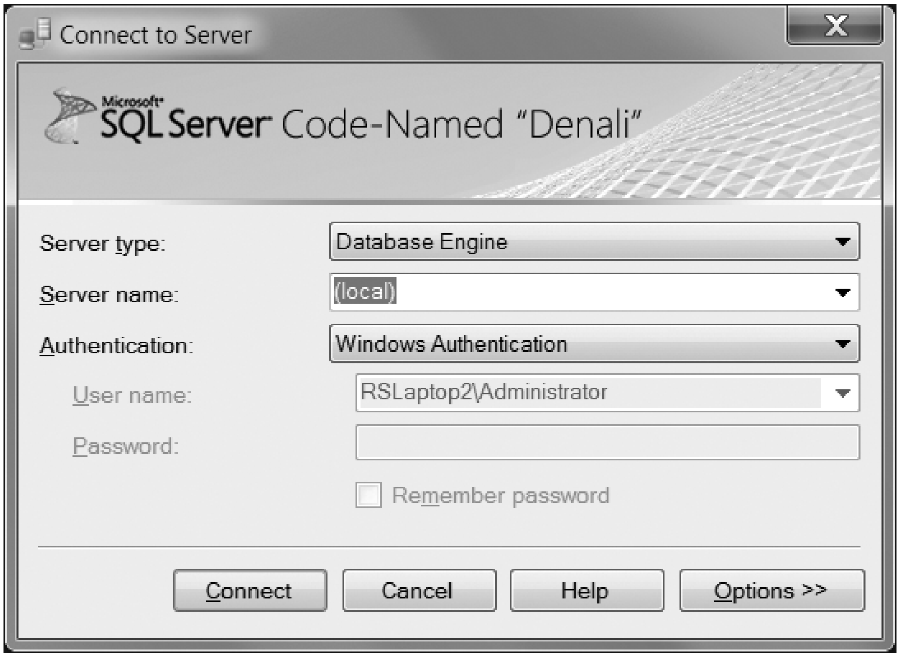
In a real situation, it is best to meet with pertinent members of your team, as well as the client for the review process. Each will often see things that the others do not, and the client will have more perspectives on what is necessary to include than you will without them.

EXERCISE 3-3. REFINING THE PLAN

In this exercise, you take on the role of a solution planner and data warehouse developer. You examine the data currently available and decide whether you will be able to create an effective data warehouse from the data. You further refine the current definition of the BI solution by removing requirements that cannot be accomplished during this iteration of the development cycle.

Reviewing the Current Plan

1. Open SQL Server Management Studio and run as administrator. This is done by clicking the Start button at the bottom left of your screen and navigating to All Programs ➤ Microsoft SQL Server 2011. Then right-click SQL Server Management Studio and select Run as Administrator from the context menu. (You will always run these programs as administrator, because of the permissions required to access databases, and so on.)
2. When SQL Server Management Studio opens, connect to your computer's database engine, and type in the name of your SQL Server. Then click the Connect button to connect to the database engine (Figure 3-7).



Connect to Server

Microsoft SQL Server Code-Named "Denali"

Server type: Database Engine

Server name: (local)

Authentication: Windows Authentication

User name: RSLaptop2\Administrator

Password:

Remember password

Figure 3.*7. Connecting to the database engine in SQL Server Management Studio*

Usually this will be (local) or localhost, but remember that if you are using a named instance of SQL Server, you will have to use the full name in the format Computer-Name\NamedInstance. For example, Randal uses (local)\SQL2012 and Caryn uses (local)\Denali because we each have installed SQL Server under those names, we are using a beta version, and Caryn never got around to giving her computer a real name. For further information on connecting to your computer's database, see Chapter 5 and search the Web for *SQL Server Named Instances*.

3. After the connection is made, you should see the databases icon in the Object Explorer window. Locate this icon and expand the treeview by clicking the plus (+) sign next to the word *Databases* (Figure 3-8).

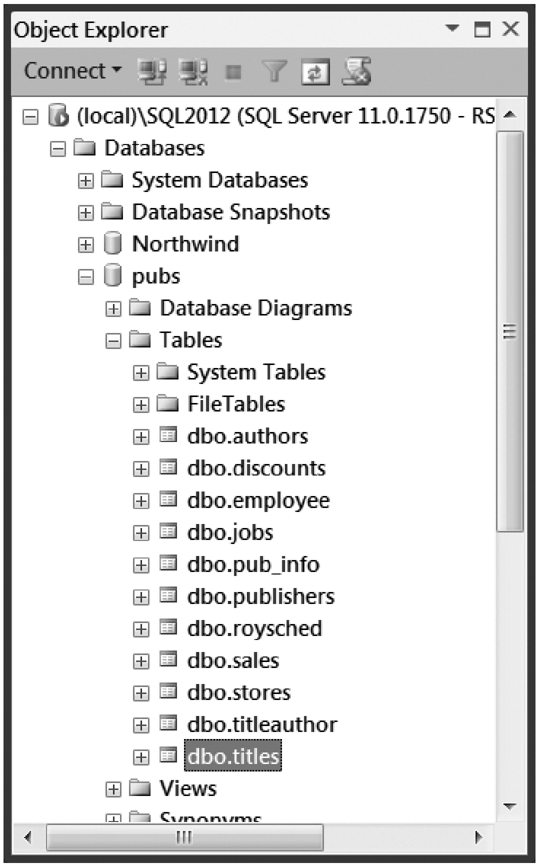


Figure 3-8. *The pubs database and its tables in Object Explorer*

4. In the list of databases, locate the pubs database and expand the treeview by clicking the plus (+) sign next to the pubs database name to see the `Tables` folder (Figure 3-8).

5. The pubs database was one of the databases you added as part of the setup process. If it does not exist, please review the instructions found in the folder `C:\_BookFiles\_SetupFiles`.

6. Expand the treeview by clicking the plug (+) sign next to the `Tables` folder to see the list of tables (Figure 3-8).

7. Review the tables noting the subject matter of each one. Figure 3-9 shows a diagram of these tables and their relation to each other.

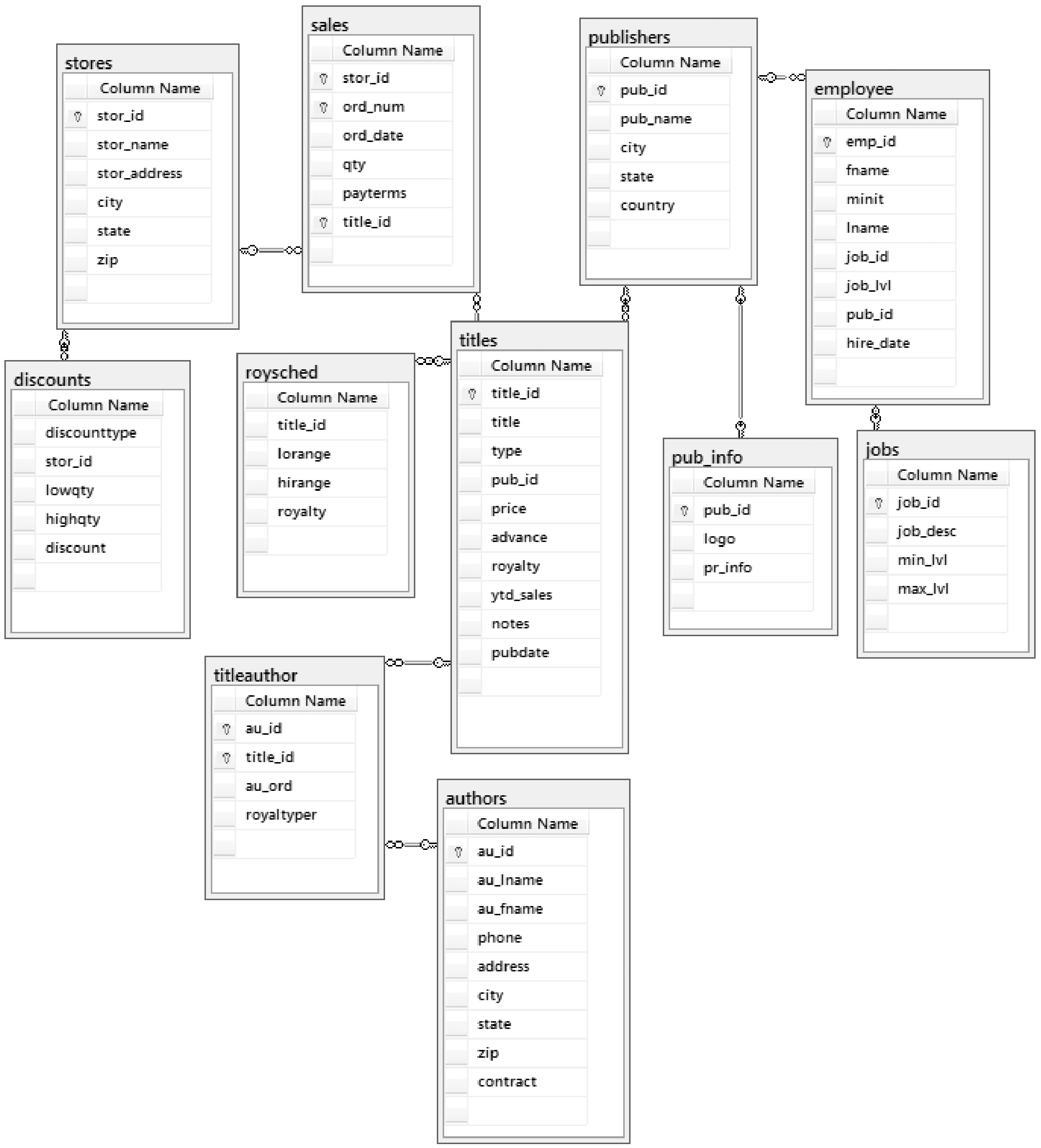


Figure 3-9. *The current pubs database*

Deciding Which Tables to Include

Although the titles table is the center of the diagram, the sales table provides us with information about the company sales. Therefore, this table is the focus of our reporting efforts. As you examine this diagram, notice that the other tables seem to support the sales information. Each table at some point was considered important, but this is not the same as being important to our BI solution. It is a best practice not to include everything from the OLTP environment into the data warehouse. Instead, each table should be reviewed to determine its relevance to the current BI solution.

At an absolute minimum, you need the information found in the sales table. Now, let's look at what else to include.

In addition to the sales information, you need descriptors of each sale. These descriptors are also referred to as *dimensional attributes*. Start by reviewing the other tables associated with the sales table; then determine whether the data inside these tables are a Must Have, a Nice to Have, or a Not Needed item. You can use the four-quadrant prioritizing technique we discussed earlier in this chapter (Figure 3-2).

The following steps take you through each of the tables and outline facts that help determine what to include:

1. Examine the dbo.Stores and dbo.Titles tables. These tables include information about which books are being sold to which stores. Therefore, both stores and titles need to be included. These tables provide a great deal of value and are easy to implement. These are easily determined to be a Must Have.
2. Examine the dbo.Publishers table. Because this fictitious company happens to wholesale books from many publishers, we also need the publisher information. The publisher's information is easy to obtain from this table and is of great use. Therefore, dbo.Publishers is categorized as a Must Have.
3. Examine the dbo.Authors table. Information about which authors write which books may not be as important to the sales reports as the name of the titles may be, but it would be nice to know. Therefore, you might consider including information about the authors as well. This information is easy to obtain and provides somewhat useful information. Therefore, dbo.Authors should be classified as Nice to Have.
4. Examine the dbo.Employee table. For some reason, the database records information about the employees known to work at various publishers. This seems to have little to do with the event of making a sale. The information is easy to obtain but of little value. As such, dbo.Employee qualifies as Not Needed.
5. Examine the dbo.Jobs table. Oddly enough, the database has information about which jobs are held by publisher employees. Exactly how this information was obtained is unimportant. What is important is that it is not necessary to our data warehouse design. So, let's classify dbo.Jobs as a Not Needed item.
6. Examine the dbo.TitleAuthors table. It tracks the royalty percentages given to various authors in the dbo.TitleAuthors table and the order in which each author is listed on each book in this same table. Royalties may have little to do with making a sale, and we doubt that this company—which is acting as a middleman wholesaler—even finds it useful. The order in which the authors' names appear can have an impact on the sales of the book, because only the first author's name may appear in certain listings. The important factor to consider is that this table represents a many-to-many relationship between dbo.Titles and dbo.Authors. Therefore, dbo.TitleAuthors is required to enable our solution to map the relationship between these tables effectively. We must classify this table as a Must Have item.
7. Examine the dbo.RoySched table. The royalty schedule table, dbo.RoySched, tracks ranges of royalties based on the amount of books sold. As more books are sold, the royalties to the authors increase. Although this influences profit, our BI solution

focuses on sales volumes. In the future we may want this, but currently it is easy to obtain but of little value to us. As such, dbo.RoySched is classified as Not Needed.

8. Examine the dbo.Discounts table. We see that some stores seem to have discounts tracked in the discounts table. This would seem to be something we would like to include as well. And yet, let's say that we asked the company owner about how the discounts were calculated. The company owner then informed us that not only are discounts no longer tracked but the data in these fields are considered iffy at best. In real life, it is common to discover something in the database that was once tracked in the past but somehow got put away in the background and ignored. As time goes by, the reason for this information is lost. This means that it is difficult to verify that the data is truly valid, and even if the table would provide good information for reports, it still should be classified as Not Needed.

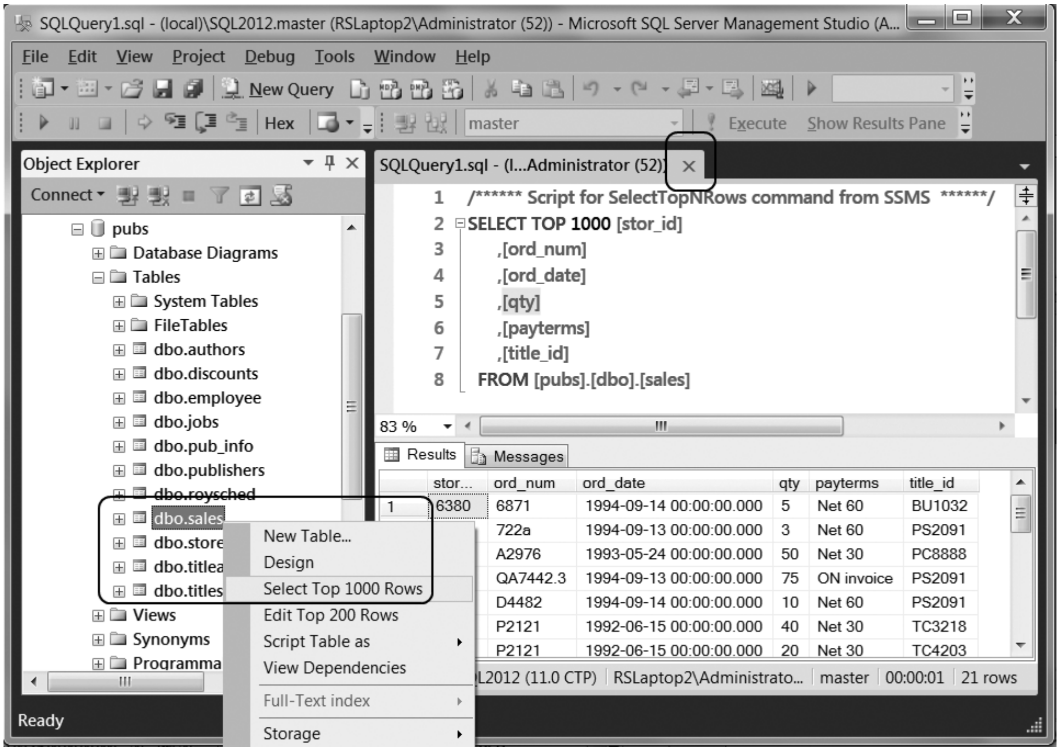
9. Examine the dbo.Pub_Info table. The publisher information table, dbo.Pub_Info, would seem to be useful at first sight. On closer inspection, we see that it is probably not useful to us. In real life, it is often the case that a table looks like it will be of use at first glimpse, but upon further examination, it is not as pertinent as it first seemed. In this case, we classify this table as Nice to Have.

In the next section of this exercise, we take a look at the data in these tables to determine which columns will be included.

Deciding Which Columns to Include from Sales

At this point, you have examined the tables and categorized them by priority. It is time to do a closer inspection of the data in each of these tables. In SQL Server Management Studio, you can quickly look at the contents of a table by right-clicking the table and selecting Select Top 1000 Rows from the context menu (Figure 3-10). We start with the sales table because it is central to our BI solution.

1. Right-click the dbo.Sales table in the Object Explorer treeview and choose the Select Top 1000 Rows menu item from the context menu. SQL Server Management Studio will create and execute a SQL select statement for you and display the results in a query window, as shown in Figure 3-10.



```
/****** Script for SelectTopNRows command from SSMS ******/
SELECT TOP 1000 [stor_id]
      ,[ord_num]
      ,[ord_date]
      ,[qty]
      ,[payterms]
      ,[title_id]
  FROM [pubs].[dbo].[sales]
```

	stor...	ord_num	ord_date	qty	payterms	title_id
1	6380	6871	1994-09-14 00:00:00.000	5	Net 60	BU1032
		722a	1994-09-13 00:00:00.000	3	Net 60	PS2091
		A2976	1993-05-24 00:00:00.000	50	Net 30	PC8888
		QA7442.3	1994-09-13 00:00:00.000	75	ON invoice	PS2091
		D4482	1994-09-14 00:00:00.000	10	Net 60	PS2091
		P2121	1992-06-15 00:00:00.000	40	Net 30	TC3218
		P2121	1992-06-15 00:00:00.000	20	Net 30	TC4203

Figure 3-10. *Displaying dbo.Sales table data in SQL Server Management Studio*

2. Examine the data in each column to determine whether the data represents a measured value or a descriptive value. In the data warehouse, measured values are translated into measures, and descriptive values are translated into dimensional attributes.

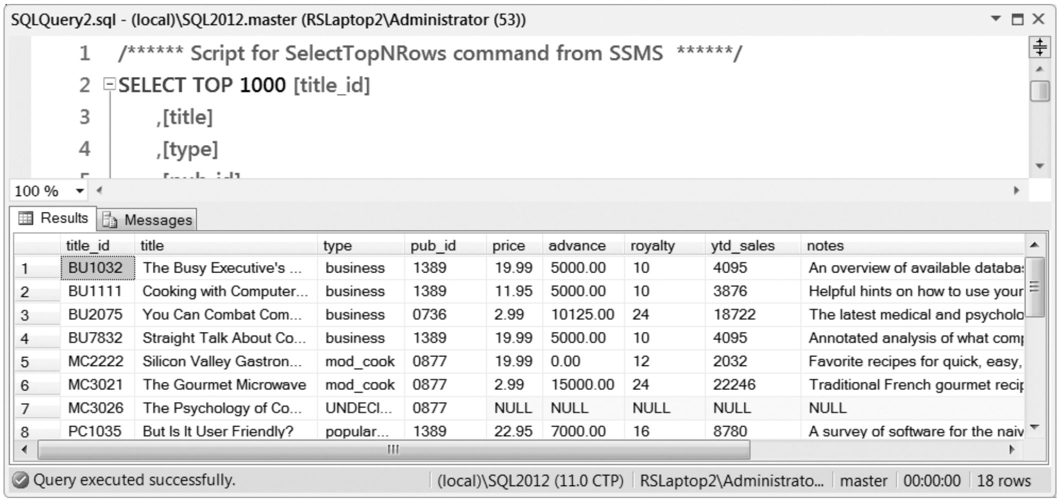
3. Close the query window by clicking the *x* on this window's tab.

The obvious measure in the sales table is the sales quantities. All other columns represent descriptive values. These current descriptive values, such as the title_id, in and of themselves hold little meaning to most clients using the reports. Therefore, you need to add dimensional attributes to the data warehouse in order to clarify what items such as title_id really indicate. It is important to include these columns as dimensional keys within your fact table.

■ **Note** The term *dimensional key* defines a column used to identify an individual row of dimensional data. This is usually used as a primary key in the dimension tables and a foreign key in the fact table. An example here is the title_id.

Reviewing the Data in the Titles Table

We need to look at the data in the various supporting tables. We take a look at the dbo.Titles table next (Figure 3-11).



SQLQuery2.sql - (local)\SQL2012.master (RSLaptop2\Administrator (53))

```
/****** Script for SelectTopNRows command from SSMS  ******/
SELECT TOP 1000 [title_id]
      ,[title]
      ,[type]
```

100 %

Results | Messages

	title_id	title	type	pub_id	price	advance	royalty	ytd_sales	notes
1	BU1032	The Busy Executive's ...	business	1389	19.99	5000.00	10	4095	An overview of available databa:
2	BU1111	Cooking with Computer...	business	1389	11.95	5000.00	10	3876	Helpful hints on how to use your
3	BU2075	You Can Combat Com...	business	0736	2.99	10125.00	24	18722	The latest medical and psycholo
4	BU7832	Straight Talk About Co...	business	1389	19.99	5000.00	10	4095	Annotated analysis of what comp
5	MC2222	Silicon Valley Gastron...	mod_cook	0877	19.99	0.00	12	2032	Favorite recipes for quick, easy,
6	MC3021	The Gourmet Microwave	mod_cook	0877	2.99	15000.00	24	22246	Traditional French gourmet recip
7	MC3026	The Psychology of Co...	UNDECI...	0877	NULL	NULL	NULL	NULL	NULL
8	PC1035	But Is It User Friendly?	popular...	1389	22.95	7000.00	16	8780	A survey of software for the naiv

Query executed successfully. | (local)\SQL2012 (11.0 CTP) | RSLaptop2\Administrato... | master | 00:00:00 | 18 rows

Figure 3-11. *The titles table*

4. Locate the dbo. Titles table in the treeview list, and right-click this table to access the context menu.

5. Choose Select Top 1000 Rows from the context menu. SQL Management Studio will create a query window, execute the query, and show you the results (Figure 3-11).

In the titles table, notice that along with the title there is a type for each title. This information would be quite useful in a report because it provides a way to group the titles collectively. Let's classify title type as a Must Have item.

A publisher ID is included in this table that could also be used to group titles. Let's classify this as a Must Have item.

In addition, the price of each title is listed here and could prove useful for making measured calculations. Although it does seem odd that this information was not included in the sales table when we created the data warehouse, we can rectify this. Let's classified this as a Must Have item.

Other columns may or may not be as useful. For example, it is unlikely that sales reports would need to categorize sales based on the type of advance given to each author for a given title. Let's classify this as Not Needed.

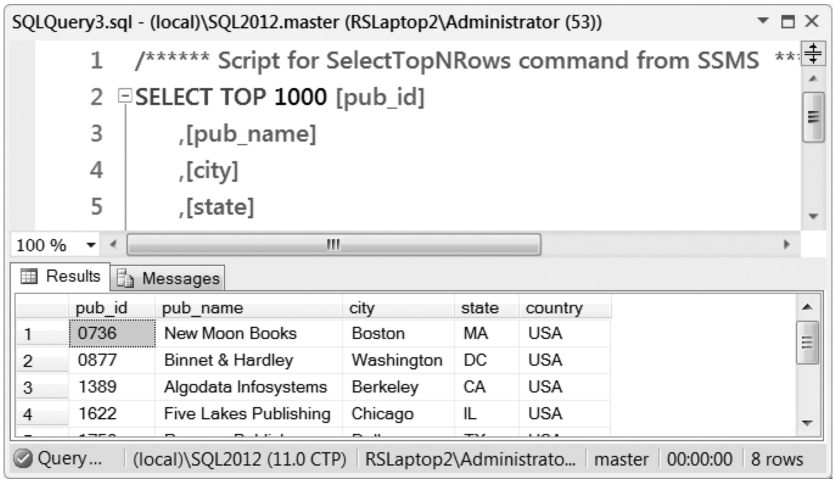
Also, the year-to-date column looks temptingly like a measure but provides aggregate values, and as we show later, aggregate values do not go into a fact table holding measured data. Let's classify this as Not Needed.

Additional auxiliary information includes a set of notes about each title and the date the titles were published. It seems somewhat obvious that the notes can be dismissed as being superficial to the sales reports. However, inclusion of the published dates is a little less clear. It may be useful to know how many sales have occurred since its published date, and this is easy to obtain. So, we include the published date in our design by classifying it as Nice to Have and exclude the notes by classifying them as Not Needed.

6. Close the query window by clicking the *x* on this window's tab.

Reviewing the Data in the Publishers Table

With the dbo.Titles table examined, let's turn our attention to the dbo.Publishers table (Figure 3-12).


SQLQuery3.sql - (local)\SQL2012.master (RSLaptop2\Administrator (53))

```
/****** Script for SelectTopNRows command from SSMS **
SELECT TOP 1000 [pub_id]
      ,[pub_name]
      ,[city]
      ,[state]
```

	pub_id	pub_name	city	state	country
1	0736	New Moon Books	Boston	MA	USA
2	0877	Binnet & Hardley	Washington	DC	USA
3	1389	Algodata Infosystems	Berkeley	CA	USA
4	1622	Five Lakes Publishing	Chicago	IL	USA

Query... | (local)\SQL2012 (11.0 CTP) | RSLaptop2\Administrato... | master | 00:00:00 | 8 rows

Figure 3-12. *The publishers table*

The following are the steps to open the dbo.Publishers table:

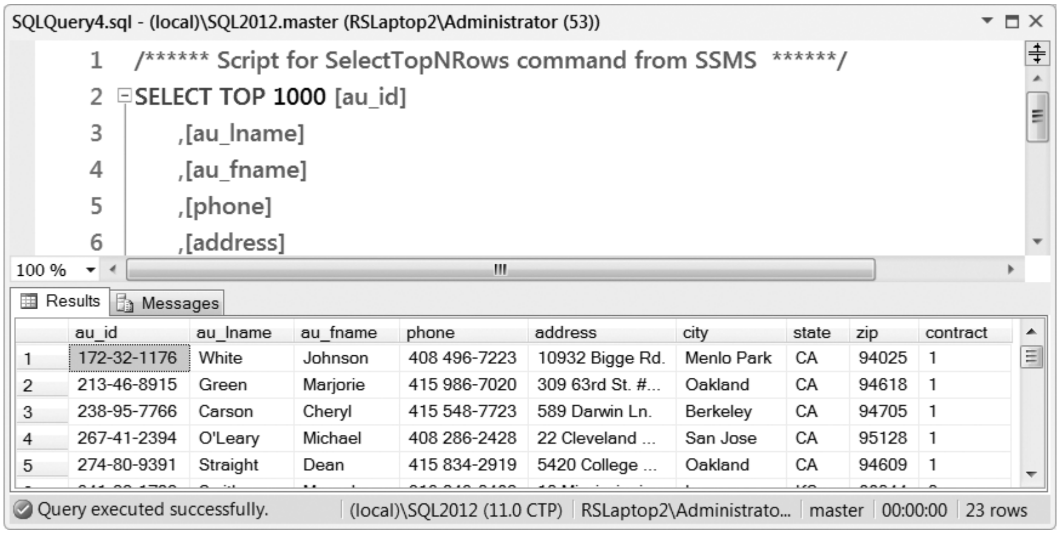
1. Locate the dbo.Publishers table in the treeview list, and right-click this table to access the context menu.
2. Choose Select Top 1000 Rows from the context menu. SQL Management Studio will create a query window, execute the query, and show you the results (Figure 3-12).

The dbo.Publishers table contains the name of the publisher, which will be useful in our sales reports.

3. This table also includes the city, state, and country where those publishers are located. At first you would likely decide that these additional columns provide information that would be useful to include in our data warehouse as dimensional attributes. Yet, looking at the values, you may notice that it is never the case that the cities or states are repeated. You would not utilize either of these columns to group publishers in your sales reports. However, there are a number of repeating values in the country column, so grouping publishers by country might prove useful. Therefore, from a practical sense, the publisher name is a Must Have item whereas country, state, and city are only Nice to Have but not required. Close the query window by clicking the *x* on this window's tab.

Reviewing the Data in the Authors Table

Let's see if something similar is happening in the dbo.Authors table (Figure 3-13).



SQLQuery4.sql - (local)\SQL2012.master (RSLaptop2\Administrator (53))

```
/****** Script for SelectTopNRows command from SSMS  ******/
SELECT TOP 1000 [au_id]
      ,[au_lname]
      ,[au_fname]
      ,[phone]
      ,[address]
```

	au_id	au_lname	au_fname	phone	address	city	state	zip	contract
1	172-32-1176	White	Johnson	408 496-7223	10932 Bigge Rd.	Menlo Park	CA	94025	1
2	213-46-8915	Green	Marjorie	415 986-7020	309 63rd St. #...	Oakland	CA	94618	1
3	238-95-7766	Carson	Cheryl	415 548-7723	589 Darwin Ln.	Berkeley	CA	94705	1
4	267-41-2394	O'Leary	Michael	408 286-2428	22 Cleveland ...	San Jose	CA	95128	1
5	274-80-9391	Straight	Dean	415 834-2919	5420 College ...	Oakland	CA	94609	1

Query executed successfully. | (local)\SQL2012 (11.0 CTP) | RSLaptop2\Administrato... | master | 00:00:00 | 23 rows

Figure 3-13. *The authors table*

Here are the steps to open the dbo.Authors table:

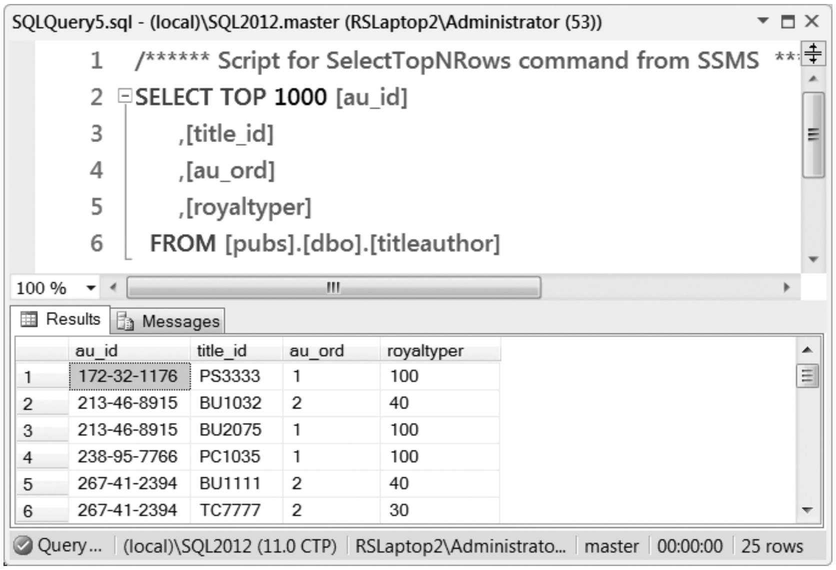
1. Locate the dbo.Authors table in the treeview list and right-click this table to access the context menu.
2. Choose Select Top 1000 Rows from the context menu. SQL Management Studio will create a query window, execute the query, and show you the results (Figure 3-13).

In the authors table it is pretty clear that the author's first and last name are Must Have attributes. However, the author's phone number and address data are less so. Keeping things simple, you might want to leave out these unnecessary columns. They do not take up much space; however, when you are trying to keep things as clean and uncluttered as possible, removing things that are unnecessary is the easiest way to accomplish this goal. Therefore, let's keep the author's name columns as Must Have and the other columns as only Nice to Have. (It may be that in another version of the data warehouse this priority list will change.)

3. Close the query window by clicking the *x* on this window's tab.

Reviewing the Data in the TitleAuthor Table

The table that connects the titles and authors together in a many-to-many relationship is the dbo.TitleAuthor table. Let's look at this table next (Figure 3-14).


SQLQuery5.sql - (local)\SQL2012.master (RSLaptop2\Administrator (53))

```
/****** Script for SelectTopNRows command from SSMS  **
SELECT TOP 1000 [au_id]
      ,[title_id]
      ,[au_ord]
      ,[royaltyper]
  FROM [pubs].[dbo].[titleauthor]
```

	au_id	title_id	au_ord	royaltyper
1	172-32-1176	PS3333	1	100
2	213-46-8915	BU1032	2	40
3	213-46-8915	BU2075	1	100
4	238-95-7766	PC1035	1	100
5	267-41-2394	BU1111	2	40
6	267-41-2394	TC7777	2	30

Query... | (local)\SQL2012 (11.0 CTP) | RSLaptop2\Administrato... | master | 00:00:00 | 25 rows

Figure 3-14. *The titleauthor table*

Here are the steps to open the titleauthor table:

1. Locate the dbo.TitleAuthor table in the treeview list and right-click this table to access the context menu.

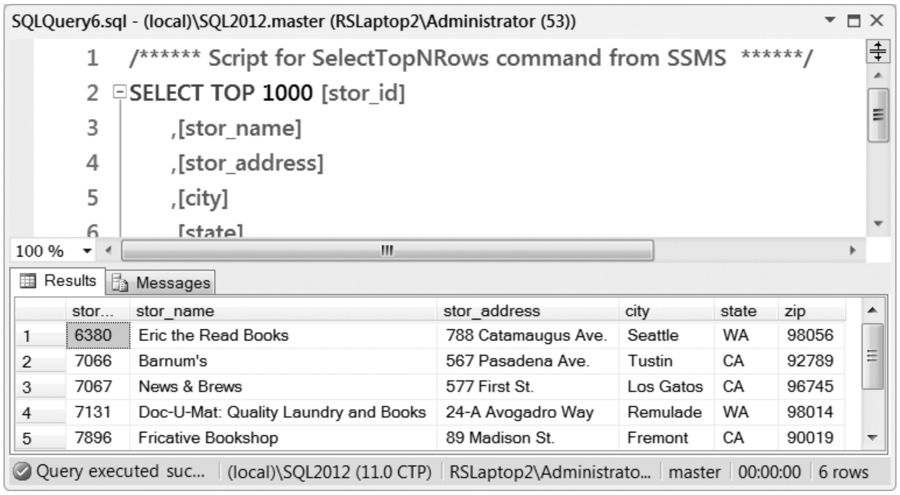
2. Choose Select Top 1000 Rows from the context menu. SQL Management Studio will create a query window, execute the query, and show you the results (Figure 3-14).

This table is a Must Have for its ability to connect a many-to-many relationship between titles and authors. In order for that to be accomplished, you must include the title_id and au_id columns. However, the royalty percentages may not be useful to us and should be excluded. The author order column, au_ord, which may have an impact on sales, can be classified as a Nice to Have item.

3. Close the query window by clicking the *x* on this window's tab.

Reviewing the Data in the Stores Table

Next up is the dbo.Stores table shown in Figure 3-15.


SQLQuery6.sql - (local)\SQL2012.master (RSLaptop2\Administrator (53))

```
/****** Script for SelectTopNRows command from SSMS  ******/
SELECT TOP 1000 [stor_id]
      ,[stor_name]
      ,[stor_address]
      ,[city]
      ,[state]
```

	stor...	stor_name	stor_address	city	state	zip
1	6380	Eric the Read Books	788 Catamaugus Ave.	Seattle	WA	98056
2	7066	Barnum's	567 Pasadena Ave.	Tustin	CA	92789
3	7067	News & Brews	577 First St.	Los Gatos	CA	96745
4	7131	Doc-U-Mat: Quality Laundry and Books	24-A Avogadro Way	Remulade	WA	98014
5	7896	Fricative Bookshop	89 Madison St.	Fremont	CA	90019

Query executed suc... | (local)\SQL2012 (11.0 CTP) | RSLaptop2\Administrato... | master | 00:00:00 | 6 rows

Figure 3-15. *The stores table*

Here are the steps to open the stores table:

1. Locate the dbo.Stores table in the treeview list, and right-click this table to access the context menu.

2. Choose Select Top 1000 Rows from the context menu. SQL Management Studio will create a query window, open an SQL query window, execute the query, and show you the results (Figure 3-15).

Similar to the authors table, the stores table contains the names and addresses of the stores. Once again, the names qualify as Must Have data, whereas the address information would be Nice to Have but not required. It may be interesting to examine sales on a state-by-state basis, but it is much less likely that sales would be examined based on street addresses. Cities fall somewhere in between the two, but if you look closely, you see that cities are never repeated. Therefore, it is unlikely that useful information can be extracted regarding sales based on the city in which the stores are located.

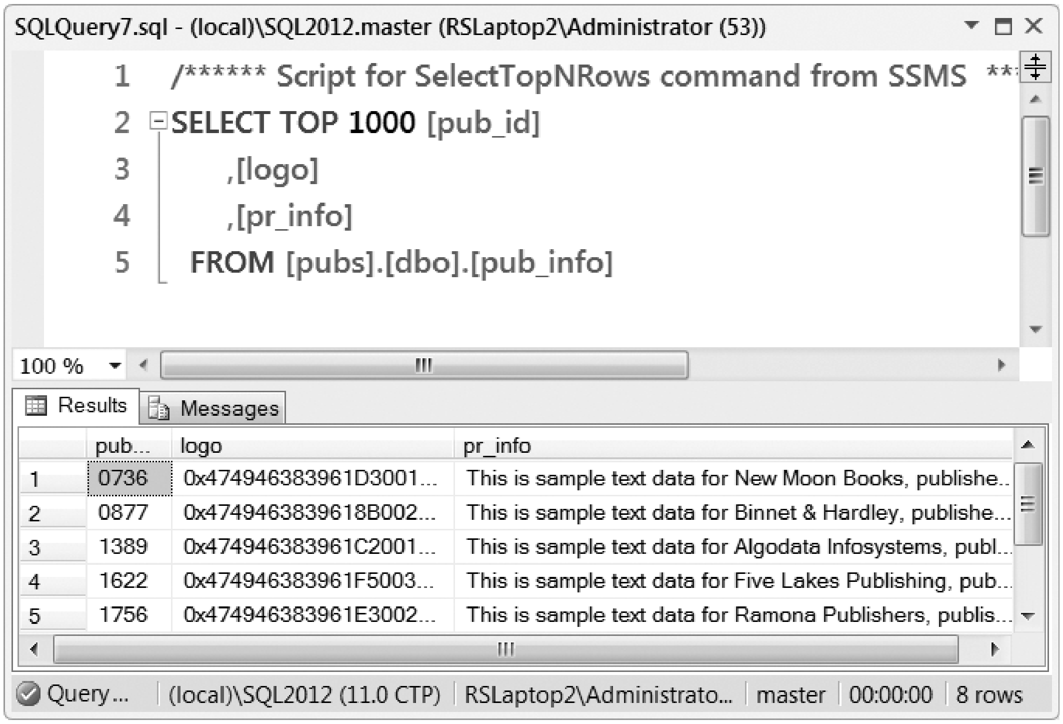
One question we forgot to ask during the interview process was how long the client has been collecting this particular set of data. If you went back and asked this question, you would find that the number of stores, publishers, and authors seldom changed over the last few years. If a lot of change had been occurring, we might have decided that the city column would become more useful as time went on. However, because change is slow or nonexistent, all the city column will do is provide additional data in the data warehouse without much additional information. Using the “keep it simple” rule, you might choose to exclude the street address, city, and zip code from this first version of the data warehouse. As time goes by, if you decide that this data becomes useful, you can add it during the creation of future versions.

Note: We are leaving each store’s state data out of the first version of the design. As we see later in the book, we will come to regret this decision. But, it allows us a chance to examine what to do if something is missing from an initial version.

3. Close the query window by clicking the *x* on this window’s tab.

Reviewing the Data in the Pub_Info Table

A number of columns in various supporting tables are not necessarily useful for this version but could be useful in the future. For example, the publishers’ information table, or pub_info as it is called, has little in it that is of use to us at this time (Figure 3-16).


SQLQuery7.sql - (local)\SQL2012.master (RSLaptop2\Administrator (53))

```
/****** Script for SelectTopNRows command from SSMS **
SELECT TOP 1000 [pub_id]
      ,[logo]
      ,[pr_info]
  FROM [pubs].[dbo].[pub_info]
```

	pub...	logo	pr_info
1	0736	0x474946383961D3001...	This is sample text data for New Moon Books, publishe...
2	0877	0x4749463839618B002...	This is sample text data for Binnet & Hardley, publishe...
3	1389	0x474946383961C2001...	This is sample text data for Algodata Infosystems, publ...
4	1622	0x474946383961F5003...	This is sample text data for Five Lakes Publishing, pub...
5	1756	0x474946383961E3002...	This is sample text data for Ramona Publishers, publis...

Query... | (local)\SQL2012 (11.0 CTP) | RSLaptop2\Administrato... | master | 00:00:00 | 8 rows

Figure 3-16. *The pub_info table*

Here are the steps to open the pub_info table:

1. Locate the dbo.Pub_Info table in the treeview list, and right-click this table to access the context menu.
2. Choose Select Top 1000 Rows from the context menu. SQL Management Studio will create a query window, execute the query, and show you the results (Figure 3-16).

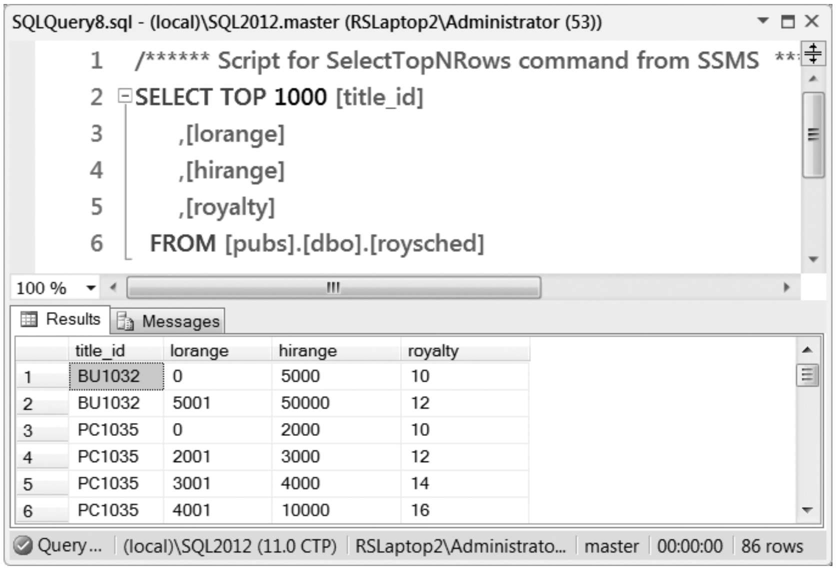
Note: This table includes binary data that represents an image of the publisher companies' logo. It also holds some nonsensical text as a placeholder that has never been filled in retroactively.

3. Close the query window by clicking the *x* on this window's tab.

Reviewing the Data in the RoySched Table

If you remember, we excluded a few tables at the beginning of this exercise. While we are here, let's look at them again to make sure we made the right decision.

One of these excluded tables was the royalty schedule table, otherwise known as *roysched* (Figure 3-17).


SQLQuery8.sql - (local)\SQL2012.master (RSLaptop2\Administrator (53))
100 %
Results
Messages

```
/****** Script for SelectTopNRows command from SSMS **
SELECT TOP 1000 [title_id]
      ,[lorange]
      ,[hirange]
      ,[royalty]
  FROM [pubs].[dbo].[roysched]
```

	title_id	lorange	hirange	royalty
1	BU1032	0	5000	10
2	BU1032	5001	50000	12
3	PC1035	0	2000	10
4	PC1035	2001	3000	12
5	PC1035	3001	4000	14
6	PC1035	4001	10000	16

Query ... | (local)\SQL2012 (11.0 CTP) | RSLaptop2\Administrato... | master | 00:00:00 | 86 rows

Figure 3-17. *The royalty schedule table*

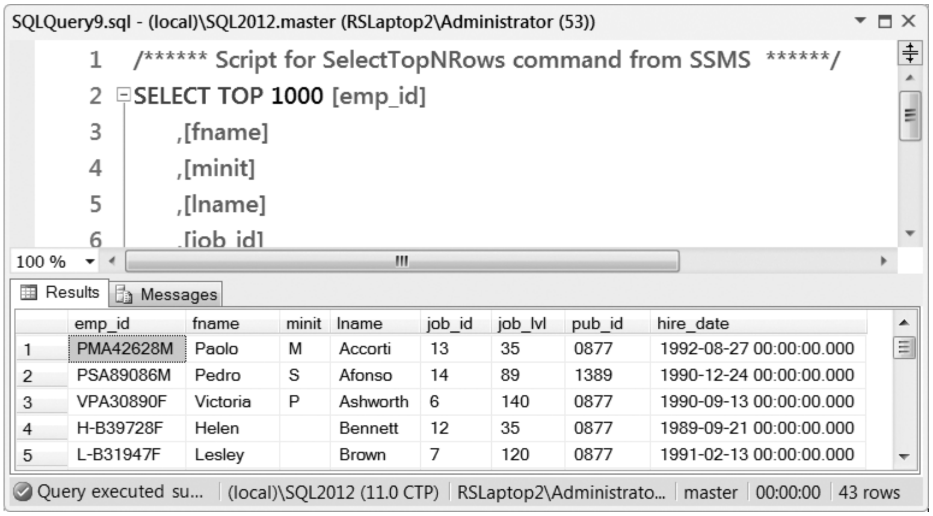
Here are the steps to open the roysched table:

1. Locate the dbo.RoySched table in the treeview list and right-click this table to access the context menu.
2. Choose Select Top 1000 Rows from the context menu. SQL Management Studio will create a query window, execute the query, and show you the results (Figure 3-17).

In this table you see data that represents the amount of royalties paid on a particular book based on the range of sales. When a book sells from 0 to 5,000 copies the royalty paid is 10%. However, when book sales reach between 5,001 and 50,000, the royalty percentage goes up to 12%. Because royalties are paid by the publisher to the authors and not by the wholesaling company who we are building the BI solution for, this information is irrelevant.

Reviewing the Data in the Employee Table

Another excluded table was the dbo.Employee table. At first glance, information about employees could prove useful to providing sales information. In most cases, this would be a true assumption; however, in the pubs database this is not the case. On closer inspection, we see that the employee table (Figure 3-18) provides a list of employees who only work at a particular publishing house, and the table is classified as Not Needed.



SQLQuery9.sql - (local)\SQL2012.master (RSLaptop2\Administrator (53))

```
/****** Script for SelectTopNRows command from SSMS  ******/
SELECT TOP 1000 [emp_id]
      ,[fname]
      ,[minit]
      ,[lname]
      ,[job_id]
```

	emp_id	fname	minit	lname	job_id	job_lvl	pub_id	hire_date
1	PMA42628M	Paolo	M	Accorti	13	35	0877	1992-08-27 00:00:00.000
2	PSA89086M	Pedro	S	Afonso	14	89	1389	1990-12-24 00:00:00.000
3	VPA30890F	Victoria	P	Ashworth	6	140	0877	1990-09-13 00:00:00.000
4	H-B39728F	Helen		Bennett	12	35	0877	1989-09-21 00:00:00.000
5	L-B31947F	Lesley		Brown	7	120	0877	1991-02-13 00:00:00.000

Query executed su... | (local)\SQL2012 (11.0 CTP) | RSLaptop2\Administrato... | master | 00:00:00 | 43 rows

Figure 3-18. *The employee table*

Here are the steps to open the dbo.Employee table:

1. Locate the dbo.Employee table in the treeview list and right-click this table to access the context menu.
2. Choose Select Top 1000 Rows from the context menu. SQL Management Studio will create a query window, execute the query, and show you the results (Figure 3-18).

Although it may be important to have a list of employee contacts who can be reached by Publication Industries, it is not relevant to the sales of individual titles to individual stores. One wonders how this information was even collected.

Deciding to Continue with the Solution

At this point, it seems obvious that we have sufficient data to create a BI solution that will provide reporting information on sales. We may not have every single detail, but we have enough to make an initial prototype. Once the prototype is created, we can get additional feedback and start working on the next version of the BI solution that will provide even greater functionality and benefit to the customer.

In this exercise, you examined the data available to you and decided what was going to be included in the BI solution and what was going to be excluded. For that matter, you also decided whether the BI solution could even be accomplished. Because it was decided that it can be accomplished, your next step is to define the roles required to create the solution and acquire the team members that will implement these roles.

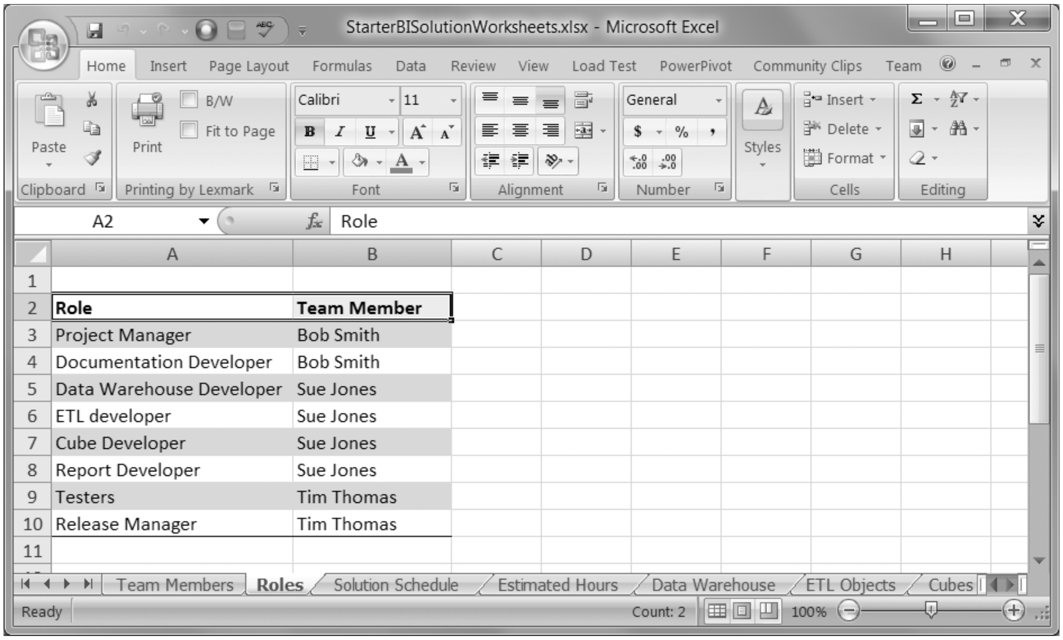
Defining the Roles

Now comes the point of defining the roles needed for the team. At a minimum, you need one team member to build the data warehouse, one team member to fill up the data warehouse, and one team member to make the initial reports. The solution must be documented as well, so you need somebody to build the documentation. In addition, the solution you come up with needs to be tested. Therefore, add a tester role to the list. They asked for a cube as well, so you need somebody to build one.

You could make a formal document describing the role and which task they are to perform; however, let's just add an additional worksheet to our Excel spreadsheet and then make a simple list.

■ **Note** You can find this Excel file in the `C:\BookFiles\Chapter03Files` folder. We called it `StarterBISolutionWorksheets.xslx`.

You may not necessarily know who your team members are in the beginning, but we can make a blank column where their names can be entered as the roles are filled (Figure 3 -19).



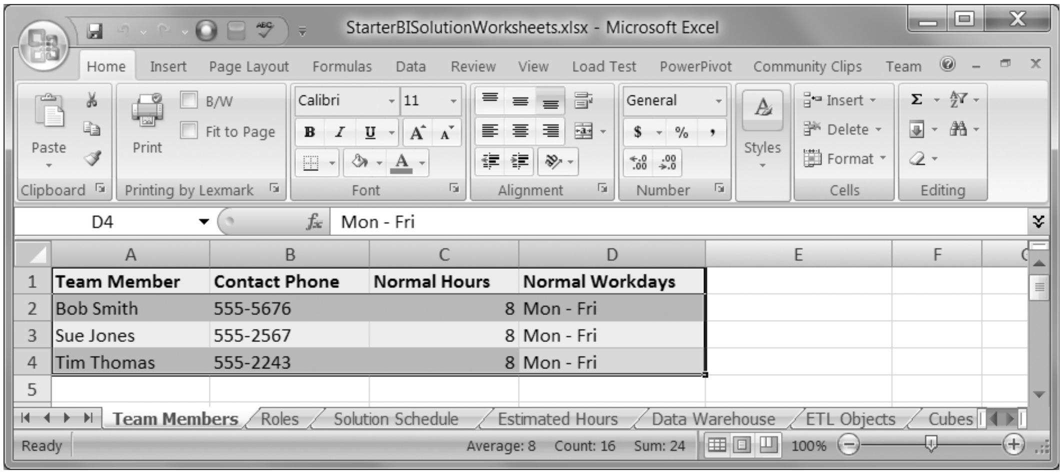
Role	Team Member
Project Manager	Bob Smith
Documentation Developer	Bob Smith
Data Warehouse Developer	Sue Jones
ETL developer	Sue Jones
Cube Developer	Sue Jones
Report Developer	Sue Jones
Testers	Tim Thomas
Release Manager	Tim Thomas

Figure 3-19. *Listing the roles*

Defining the Team

For learning purposes, you are fulfilling each role in this book, and it is your task to create each of the various elements of your BI solution. In real life, you would likely want an entire team to work on your BI solution. The old adage about two heads being better than one really does apply when you are trying to work on something as complex as a BI solution. One individual may miss seeing a possible problem that is quite obvious to another.

As stated before, documentation does not have to be exhaustive to be of use. You can simply list the team members' names, phone numbers, and email addresses, and we also recommend listing the hours they work. Three years after the project is completed, it may be useful for the BI solution users to contact members of the development team to ask questions. Offering something as simple as a list of contacts in the solution documents can make life simpler (Figure 3 -20).


StarterBISolutionWorksheets.xlsx - Microsoft Excel

D4 Mon - Fri

	A	B	C	D
1	Team Member	Contact Phone	Normal Hours	Normal Workdays
2	Bob Smith	555-5676	8	Mon - Fri
3	Sue Jones	555-2567	8	Mon - Fri
4	Tim Thomas	555-2243	8	Mon - Fri
5				

Figure 3-20. *Listing the team members*

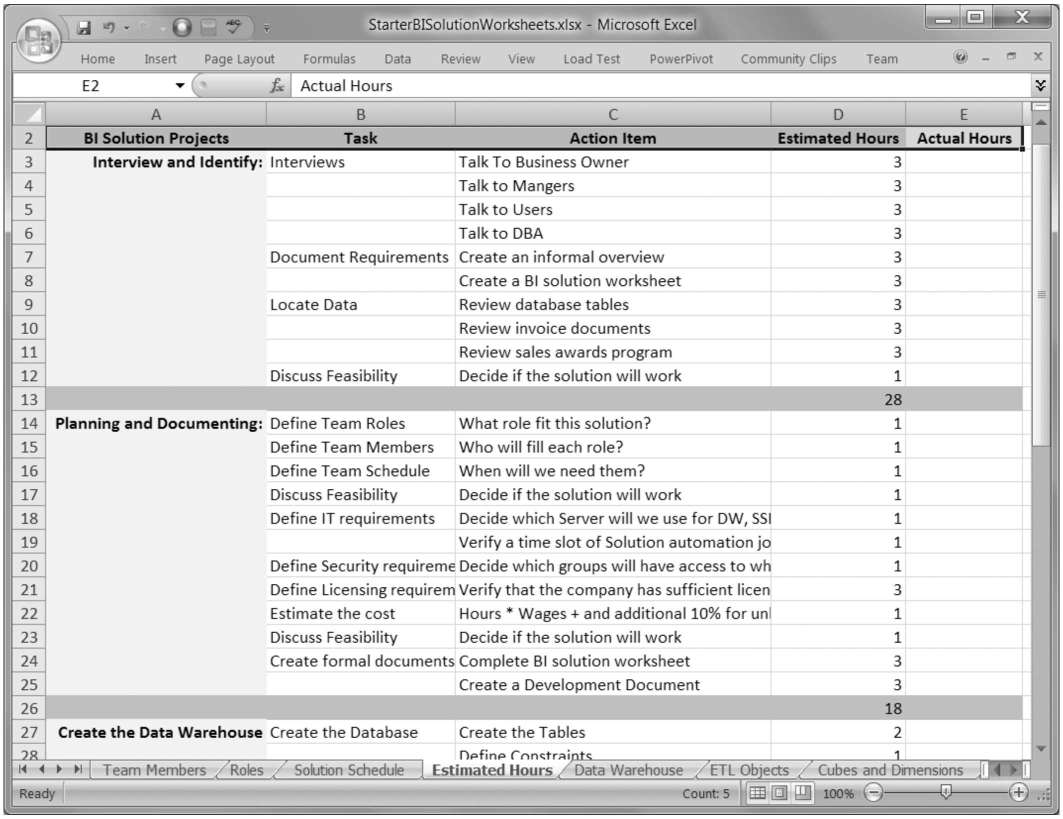
Determining the Schedule

In order to come up with a schedule, you need to have an idea of approximately how long each task will take. If you are not very experienced with these types of tasks, such as developing cubes or ETL processes, coming up with a schedule can be difficult. As you gain experience, it will be easier to make these estimates.

To get started, identify the list of tasks that you think you will need to accomplish for your particular BI solution. All BI solutions share certain similarities; however, it is a given that there will be something distinct about each one.

Try not to micromanage each task, but sum it up into sets of one to three hours of work. In an eight-hour day, consider that breaks will be taken, emails will require answering, teammates will ask questions, phones will ring and lunches will be scheduled. In an eight-hour day, the average person will complete about six hours of work.

As you can see in Figure 3-21, the number of hours is strongly weighted toward the planning phase. This may seem out of proportion, but in reality once you have defined the names of all your tables, columns, dimensions, cubes, basic reports, and so on, their creation will be a much faster process. Therefore, much of the development time is the planning phase. This includes selecting the datatype, setting certain properties, and defining the relationships between various objects. Lastly, be sure to plan time for the communication process, which can take several hours of meetings, depending upon the size of the project.

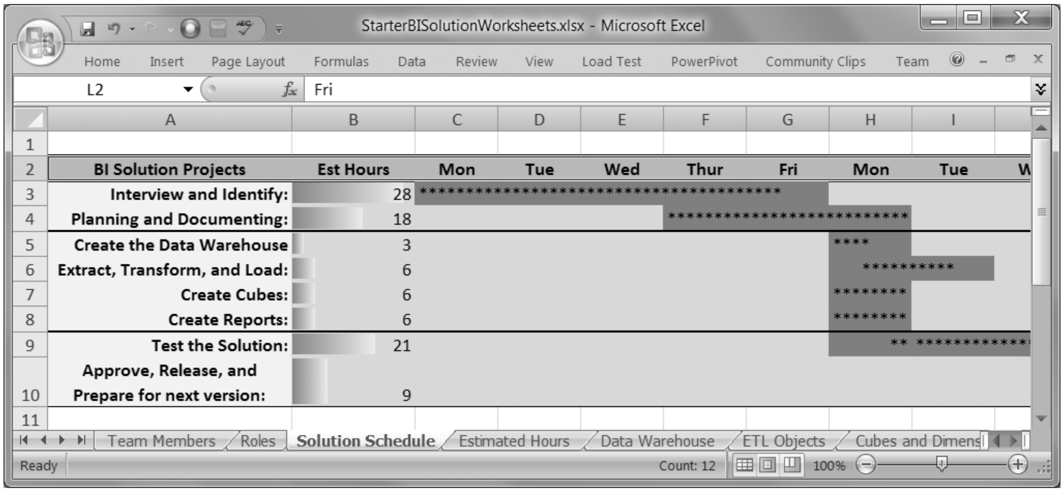

StarterBISolutionWorksheets.xlsx - Microsoft Excel
Home Insert Page Layout Formulas Data Review View Load Test PowerPivot Community Clips Team
E2
Actual Hours
Team Members Roles Solution Schedule Estimated Hours Data Warehouse ETL Objects Cubes and Dimensions
Ready
Count: 5
100%

	A	B	C	D	E
2	BI Solution Projects	Task	Action Item	Estimated Hours	Actual Hours
3	Interview and Identify:	Interviews	Talk To Business Owner	3	
4			Talk to Mangers	3	
5			Talk to Users	3	
6			Talk to DBA	3	
7		Document Requirements	Create an informal overview	3	
8			Create a BI solution worksheet	3	
9		Locate Data	Review database tables	3	
10			Review invoice documents	3	
11			Review sales awards program	3	
12		Discuss Feasibility	Decide if the solution will work	1	
13				28	
14	Planning and Documenting:	Define Team Roles	What role fit this solution?	1	
15		Define Team Members	Who will fill each role?	1	
16		Define Team Schedule	When will we need them?	1	
17		Discuss Feasibility	Decide if the solution will work	1	
18		Define IT requirements	Decide which Server will we use for DW, SSI	1	
19			Verify a time slot of Solution automation jo	1	
20		Define Security requireme	Decide which groups will have access to wh	1	
21		Define Licensing requirem	Verify that the company has sufficient licen	3	
22		Estimate the cost	Hours * Wages + and additional 10% for unl	1	
23		Discuss Feasibility	Decide if the solution will work	1	
24		Create formal documents	Complete BI solution worksheet	3	
25			Create a Development Document	3	
26				18	
27	Create the Data Warehouse	Create the Database	Create the Tables	2	
28			Define Constraints	1	

Figure 3-21. Estimating the hours

Taking the time to preplan how long a project will take and tracking hours spent will provide you with invaluable information for future projects. You will be able to estimate the cost and forecast release dates of future projects more accurately. Adding a column to your spreadsheet that tracks the hours spent is easy to do, and you will find that you refer to it often (Figure 3 -21).

Once you have estimated the hours required to complete the project, creating a timeline is relatively easy. Total up the hours for each section. If you have determined that each day represents about five hours of activity, you can make a calendar to map out how long the project will take. This calendar can be as simple as adding yet another worksheet (Figure 3-22). Keep in mind that most months have holidays, and the like.


StarterBISolutionWorksheets.xlsx - Microsoft Excel

L2 | Fri

	A	B	C	D	E	F	G	H	I
1									
2	BI Solution Projects	Est Hours	Mon	Tue	Wed	Thur	Fri	Mon	Tue
3	Interview and Identify:	28	**						
4	Planning and Documenting:	18				**************************			
5	Create the Data Warehouse	3						****	
6	Extract, Transform, and Load:	6						**********	
7	Create Cubes:	6						********	
8	Create Reports:	6						********	
9	Test the Solution:	21						** ************	
10	Approve, Release, and Prepare for next version:	9							
11									

Team Members / Roles / Solution Schedule / Estimated Hours / Data Warehouse / ETL Objects / Cubes and Dimens

Ready Count: 12 100%

Figure 3-22. *Planning the schedule*

As with anything new, the more you practice it, the more proficient at it you will become. Keep in mind that you can track how accurate you were at planning each solution by comparing what occurred to what you had planned. An additional benefit to scheduling your project is that your teammates, who may be involved with other projects as well, can integrate their schedules with yours. In the end you will find that a simple timeline is easy to create and provides advantages that far outweigh the time it takes to implement.

The IT, Security and Licensing Requirements

As consultants, we have worked with many different companies. Each company has different needs with regard to IT, security, and licensing. It is important that you consider these requirements when you are creating a BI solution.

Defining these IT requirements includes (and is not limited to) the following:

- Network bandwidth
- Which servers are available
- The age of the servers
- The space on the servers
- The people who maintain these servers

Defining security requirements involves evaluating many factors. Everyone wants their data to be secure, but what "secure" means to one company is not at all secure to another. If an industry, such as pharmaceuticals, has governmental regulations, then additional security requirements need to be addressed.

For licensing requirements, there is also a wide range of possible scenarios. One company may have a blanket license for a lot of different software. Another company may need to purchase their software individually. It may be advantageous to use software that is flexible or comes bundled with as many features as you require to complete the project. An example of this is Microsoft SQL Server. With the purchase of SQL Server alone, you gain a whole suite of BI applications. These applications allow you to build the data warehouse, perform the ETL process, create cubes, and even create reports. In fact, it could be argued that one of the best reasons to choose Microsoft SQL Server is that, for many companies, this will be the most effective tool to implement a BI solution.

Estimating the Cost

When estimating the cost of the solution, a good place to start is by evaluating how many hours you thought each task would take and multiplying it by the hourly wages you expect to pay. Keep in mind that issues can come up that could block you from performing your tasks in a particular order or on a particular date. In a perfect world, everything runs smoothly and works as planned. Nevertheless, as we all know, that is not likely to happen for the vast majority of solutions. A common way to mitigate this is to add a percentage to the estimated cost. This additional percentage usually varies between 10% to 20% depending on how risky you feel any given project is in regard to overruns and changes.

Documenting the Solution Plan

You should have a good idea of what the solution consists of after you have examined the tasks your solution will entail. By this time, you should know the following:

- Which tables you need
- Which type of ETL tasks you need to perform
- What the name of your cube might be
- The titles of your basic prototype reports

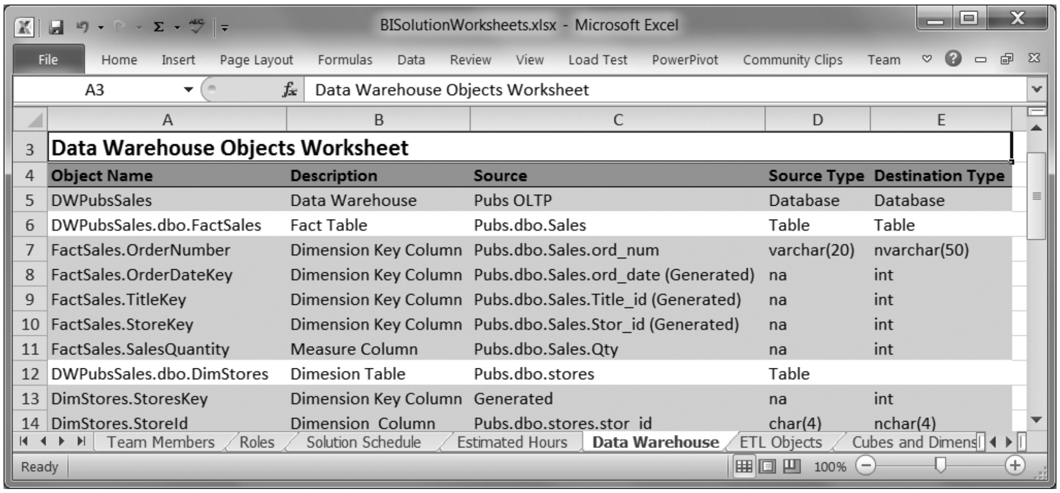
You can choose to put this information directly in your formal Word document or record it in a simple spreadsheet. Either way, it should be recorded. This formal or informal document is then used for the creation of your BI solution. Additionally, this documentation should correspond with what is incorporated into the official contract, if one is necessary.

Assume that you will make mistakes. Assume that you will miss some items. Assume that some of your time estimates may be incorrect. In the end, it does not matter that you get everything perfect the first time out. What does matter is that you learn by your mistakes and become better at creating BI solutions as time goes on.

Let's assume you are going to use an informal Excel spreadsheet to start your solution documentation. You can add another worksheet to the spreadsheet that you have already been using to document your solution plan. Make a list of the tables that you think you need to document the data warehouse.

You can start with the name of database itself. Try to stay true to whatever naming conventions you decided on in the planning throughout the project. For example, in the exercises for this book, we determined that our naming convention indicates the object's type followed by the object's name. In this case, we recorded the data warehouse name as Data Warehouse Publication Sales. Because that title is rather long, we abbreviated it to DWPubsSales.

To continue this example, we needed a fact table to hold our sales information. Therefore, we listed the fact table under the name FactSales (Figure 3-23). Its formal name is the name of the database, followed by the name of the namespace or schema, followed by the name of the object. Therefore, we listed this in the spreadsheet as DWPubsSales.dbo.FactSales.


BISolutionWorksheets.xlsx - Microsoft Excel
File Home Insert Page Layout Formulas Data Review View Load Test PowerPivot Community Clips Team
A3
Data Warehouse Objects Worksheet
Team Members Roles Solution Schedule Estimated Hours Data Warehouse ETL Objects Cubes and Dimens
Ready
100%

Data Warehouse Objects Worksheet

Object Name	Description	Source	Source Type	Destination Type
DWPubsSales	Data Warehouse	Pubs OLTP	Database	Database
DWPubsSales.dbo.FactSales	Fact Table	Pubs.dbo.Sales	Table	Table
FactSales.OrderNumber	Dimension Key Column	Pubs.dbo.Sales.ord_num	varchar(20)	nvarchar(50)
FactSales.OrderDateKey	Dimension Key Column	Pubs.dbo.Sales.ord_date (Generated)	na	int
FactSales.TitleKey	Dimension Key Column	Pubs.dbo.Sales.Title_id (Generated)	na	int
FactSales.StoreKey	Dimension Key Column	Pubs.dbo.Sales.Stor_id (Generated)	na	int
FactSales.SalesQuantity	Measure Column	Pubs.dbo.Sales.Qty	na	int
DWPubsSales.dbo.DimStores	Dimesion Table	Pubs.dbo.stores	Table	
DimStores.StoresKey	Dimension Key Column	Generated	na	int
DimStores.StoreId	Dimension Column	Pubs.dbo.stores.stor_id	char(4)	nchar(4)

Figure 3-23. *Planning the data warehouse*

Our naming conventions may be different from what you are accustomed to, but every company will have their own preferences. Whatever the preferences are, your documentation should reflect this.

In Exercise 3-3, you examined each table in the pubs database and determined which columns to include and which to omit. These can now be listed in the worksheet (Figure 3-23).

After the objects are listed, it can be helpful to list their description, source, source datatype, and destination datatype as well. We have found these columns to be quite useful when building the data warehouse. In fact, we use this list to build the data warehouse in Chapter 4.

We can then continue to list the objects in the SSIS ETL project, the SSAS cubes project, and the SSRS reports project. The list does not have to be perfect. It is simply a way of getting started. It has been our experience that we always miss something in the initial documentation anyway. We recommend that you use the spreadsheet as the initial documentation and update it as you find omissions and mistakes. Later, at the end of the solution development cycle, you can update the formal document that was created using Microsoft Word. Another, better option is to get a technical writer on your team who can update this to the formal document as you go. It is this formal version that is submitted to your client along with other documents throughout the course of your project such as your initial contract, changes to your contract if there are any, technical notes, observations, billing documents, any specific items requested by the client to be submitted, and any thank-you messages. For more information about formal documentation, see Chapter 19.

Implementation

Once you have formulated a plan of action, you must perform that action. In the case of a SQL Server BI solution, that means creating a data warehouse and several Visual Studio projects. As we saw in Chapter 2, one Visual Studio solution can hold many projects and even solution documents. So, perhaps the first thing to do is to create a solution in Visual Studio and add our documents to it. Let's do that now.

EXERCISE 3-4. ADDING THE DOCUMENTS TO YOUR SOLUTION

In this exercise, copy the Excel spreadsheet file to your `_BISolutions` folder, create a new Visual Studio solution to hold both documents you examined in this chapter and add these document to your Visual Studio solution.

Copying the Excel File to the Solution Folder

1. Inside the `C:\_BookFiles\Chapter03Files` folder, locate the `StarterBISolutionWorksheets.xlsx` file.
2. Copy this file to the `C:\_BISolutions\PublicationsIndustries` folder (Figure 3-24).

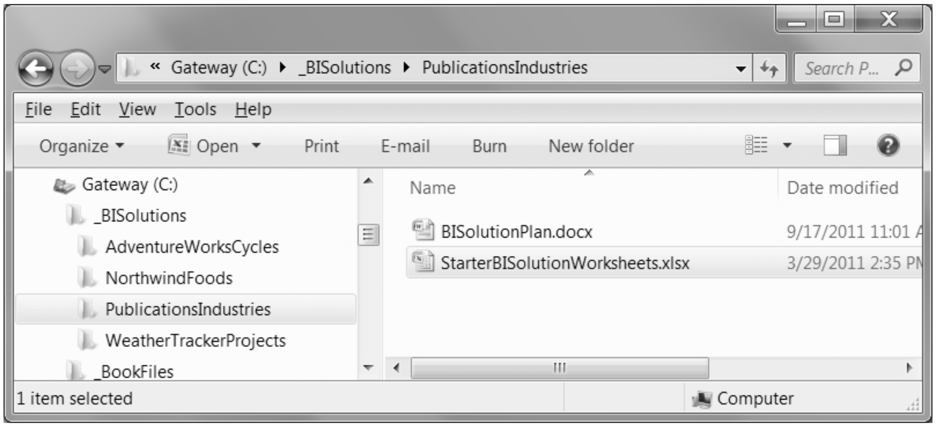

« Gateway (C:) ▸ _BISolutions ▸ PublicationsIndustries
Search P...
File Edit View Tools Help
Organize Open Print E-mail Burn New folder
Gateway (C:)
_BISolutions
AdventureWorksCycles
NorthwindFoods
PublicationsIndustries
WeatherTrackerProjects
_BookFiles
Name
Date modified
BISolutionPlan.docx 9/17/2011 11:01 A
StarterBISolutionWorksheets.xlsx 3/29/2011 2:35 PM
1 item selected
Computer

***Figure 3-24.** Now both both BI planning documents are in the `PublicationsIndustries` folder.*

3. Rename the `StarterBISolutionWorksheets.xlsx` file to `BISolutionWorksheets.xlsx`.

Placing these files directly on your `C:\` drive makes it much easier to navigate to the files you need for these exercises and leaves less room for confusion later, because we access this particular folder quite often throughout this book.

Creating a Empty Visual Studio Solution

We now need a new Visual Studio solution to hold our BI documents and projects. We do that by first creating a blank solution and then adding documents and projects to it.

1. Open Visual Studio 2010. You can do so by clicking the Start button and navigating to All Programs ➤ Microsoft Visual Studio 2010 ➤ Microsoft Visual Studio 2010; then right-click this menu item to see an additional context menu (Figure 3-25). Click the “Run as administrator” menu item. If the UAC message box appears, click Continue to accept this request.

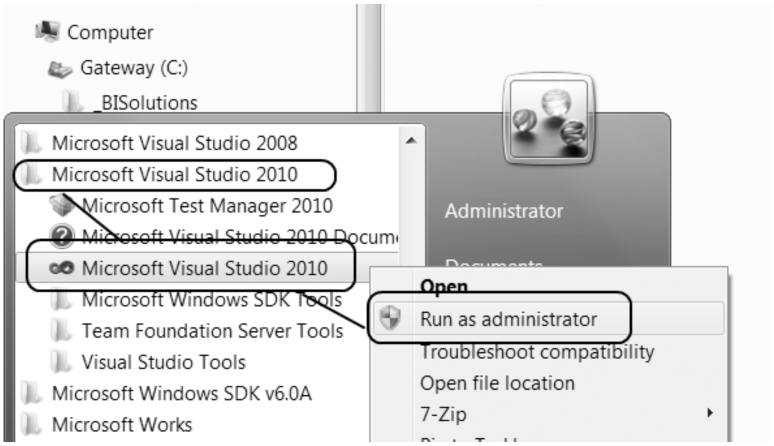
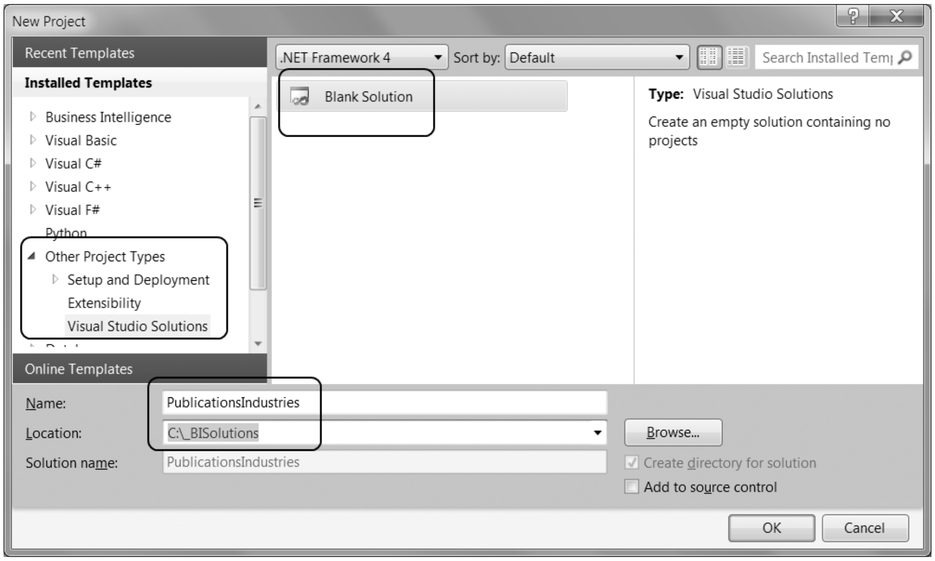

Computer
Gateway (C:)
_BISolutions
Microsoft Visual Studio 2008
Microsoft Visual Studio 2010
Microsoft Test Manager 2010
Microsoft Visual Studio 2010 Docum
Microsoft Visual Studio 2010
Microsoft Windows SDK Tools
Team Foundation Server Tools
Visual Studio Tools
Microsoft Windows SDK v6.0A
Microsoft Works
Administrator
Open
Run as administrator
Troubleshoot compatibility
Open file location
7-Zip

Figure 3-25. *Opening Visual Studio and running as admininstrator*

2. When Visual Studio opens, select File ➤ New ➤ Project from the menu. (Do not use the Create: Project option from the Start Page as you may have done in the past with other types of solutions.)

3. When the New Project dialog window opens on the left side of your screen, click the arrow to expand Other Project Types and select Visual Studio Solutions, as you shown in Figure 3-26.


New Project

Type: Visual Studio Solutions

Create an empty solution containing no projects

Name: PublicationsIndustries

Location: C:_BISolutions

Solution name: PublicationsIndustries

Figure 3-26. *Creating a new blank solution*

4. In the templates section, select Blank Solution, as shown in Figure 3-26. The Name and Location textboxes will be filled in with a default name, but we change these in the next step.

5. In the Name textbox, type the name **PublicationsIndustries**.

This is the same folder where the solution files are located. When Visual Studio makes the solution, it will use this already created folder.

6. In the Location textbox, type **C:_BISolutions**, as shown in Figure 3-26, and finally click OK.

Once the solution is created, it will appear in Solution Explorer. This will be on the right side of your screen. (This is its default location for the Solution Explorer window, but it can be moved.)

7. Right-click the solution PublicationsIndustries icon and select the Open Folder in the Windows Explorer context menu item. Windows Explorer will open to the location of your solution folder (Figure 3-27).

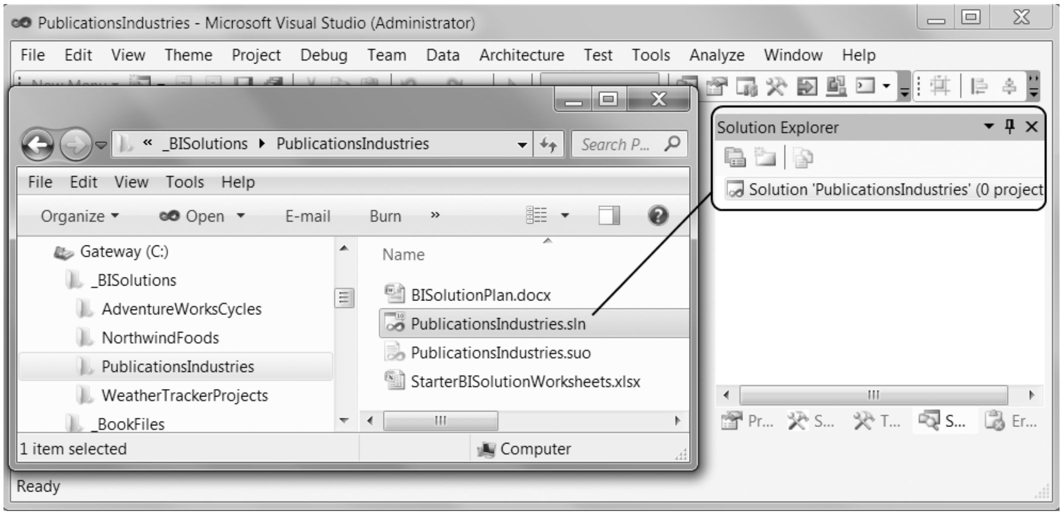

PublicationsIndustries - Microsoft Visual Studio (Administrator)
File Edit View Theme Project Debug Team Data Architecture Test Tools Analyze Window Help
« _BISolutions ▸ PublicationsIndustries
Search P...
File Edit View Tools Help
Organize Open E-mail Burn
Gateway (C:)
_BISolutions
AdventureWorksCycles
NorthwindFoods
PublicationsIndustries
WeatherTrackerProjects
_BookFiles
Name
BISolutionPlan.docx
PublicationsIndustries.sln
PublicationsIndustries.suo
StarterBISolutionWorksheets.xlsx
1 item selected
Computer
Solution Explorer
Solution 'PublicationsIndustries' (0 project
Pr... S... T... S... Er...
Ready

Figure 3-27. *Viewing the files in the solution folder*

8. Review the files in this folder and notice that they are not currently showing in the Solution Explorer window. Solution Explorer shows only a minimum of files from a solution or project folder, but we can change this by adding the file to the solution as existing items. (Depending on your computer settings you may not see the `.suo` hidden file.)

9. Close Windows Explorer.

Adding the Solution Documents

1. Add a new solution folder to your Visual Studio solution by clicking the new folder button at the top of the Solution Explorer window.
2. Rename the new folder as **SolutionDocuments**. It will appear in the Solution Explorer.
3. Right-click the new SolutionDocuments folder, and select Add ➤ Existing Items from the context menu, as shown in Figure 3-28. A dialog window will open.

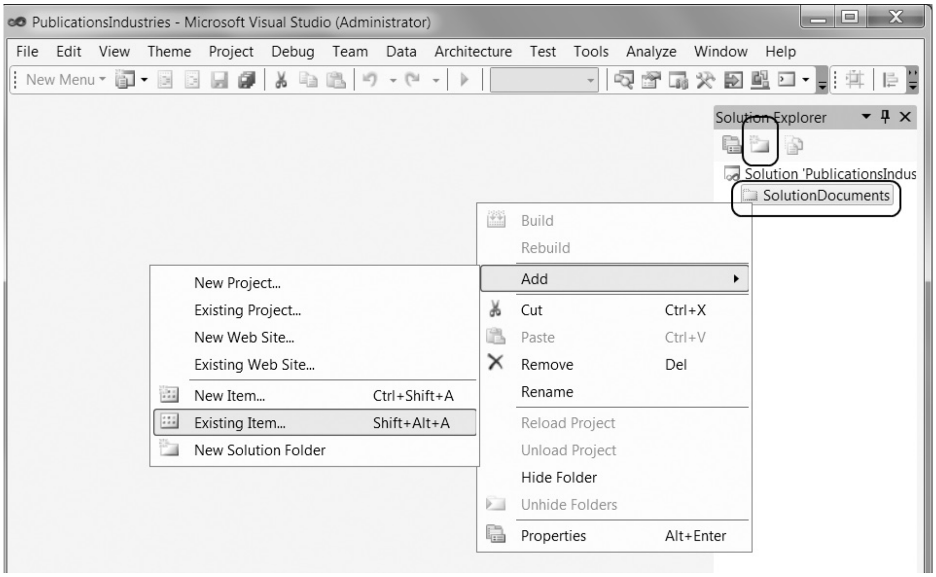
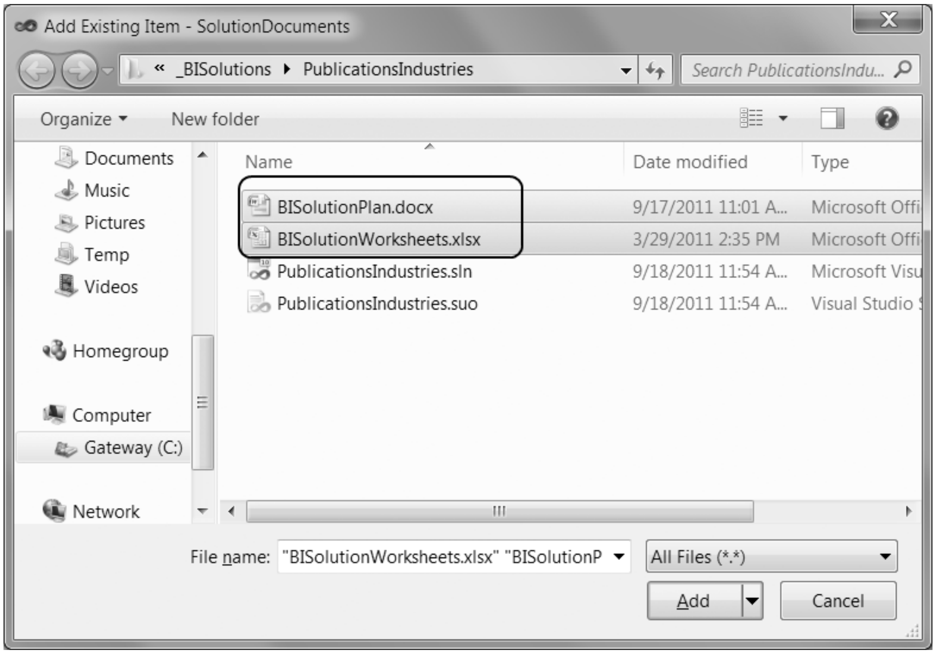


Figure 3-28. *Adding existing items to the solution folder*

4. When the Add Existing Item dialog window appears, select Local Disk (C:), and then navigate to the `C:\_BISolutions\PublicationIndustries` folder.
5. While holding down the Control button, click to select the following files: `BISolutionWorksheets.xlsx` and `BISolutionPlan.docx` (Figure 3-29).



Name	Date modified	Type
BISolutionPlan.docx	9/17/2011 11:01 A...	Microsoft Offi
BISolutionWorksheets.xlsx	3/29/2011 2:35 PM	Microsoft Offi
PublicationsIndustries.sln	9/18/2011 11:54 A...	Microsoft Visu
PublicationsIndustries.suo	9/18/2011 11:54 A...	Visual Studio

File name: "BISolutionWorksheets.xlsx" "BISolutionP

Figure 3-29. *Viewing the files in the solution folder*

Note: Windows by default hides file extensions. So, you may only see the first part of the name of each of these files. We recommend turning off this feature when you get the chance. We often find it very helpful to know the extension of all the files. To find out more about this feature, search the Web for *Windows Show Extensions*.

6. Click the Add button to add them to the `SolutionDocuments` folder. (Visual Studio uncharacteristically creates a reference instead of a copy when you add an existing item to a solution folder.)
7. Visual Studio will try to open each of the files so that you can see their content. We are not making any changes to these files. Review them if you like, but then close Excel and Word afterward.
8. Use Visual Studio's File menu to save your work by selecting the Save All option.
9. Close Visual Studio.

In this exercise, you created a blank solution and added documents used for creating your SSIS, SSAS, and SSRS projects. We refer to these documents in future exercises throughout the book.

Moving On

In this chapter, we have seen how to come up with a basic plan to implement a BI solution. We reviewed the basic workflow that outlines the various tasks involved in deciding whether the BI solution would be a viable project. The workflow incorporated items such as interviewing the customer, defining what would be included in the solution, reviewing the available data, determining which roles each team member would play, and creating simple documentation for tracking and planning the project.

In the next chapter, we create the data warehouse for the BI solution based on our plan. We tend to think that creating the data warehouse is a lot more fun than all this planning. But good planning can make the difference between a profitable, effective BI solution and a chaotic expensive mess. Hence, Winston Churchill's wise words on planning, once again, come to mind.

LEARN BY DOING

In this "Learn by Doing" exercise, you perform the process defined in this chapter using the Northwind database. We have included an outline of the steps you performed in this chapter and an example of how the authors handled them in two Word documents. These documents are found in the folder `C:\_BISolutionsBookFiles\_LearnByDoing\Chapter03Files`. Please see the `ReadMe.doc` file for more instructions.

What's Next?

Entire books are written about solution planning, and one chapter in this book is insufficient to make anyone an expert on the subject. By the same token, reading all the books ever written about the subject will not necessarily make you proficient at performing it, either. Becoming proficient involves a combination of practical experience and researched knowledge.

For further instruction on solution planning, we recommend these books: *The Kimball Group Reader: Relentlessly Practical Tools for Data Warehousing and Business Intelligence* by Ralph Kimball and Margy Ross (Wiley), and *Lessons in Project Management* by Jeff Mochal (Apress).