

Learn
SQL SERVER 2012
IN A MONTH OF LUNCHES



GRANT B. FRITCHEY

MANNING



MEAP Edition
Manning Early Access Program
Learn SQL Server 2012 in a Month of Lunches version 6

Copyright 2012 Manning Publications

For more information on this and other Manning titles go to
www.manning.com

Table of Contents

Part I: Getting started

1. Dive into SQL Server
2. Creating a database
3. Configuring the server
4. Securing the server
5. Securing the database

Part II: Protecting your data

6. Database backups
7. Restores using database backups
8. Checking the database storage
9. Tools to control necessary maintenance
10. Scheduling operations in SQL Server

Part III: Creating databases, structure, and code

11. Create a table
12. Adding data to the table
13. Reading data out of the table

Part IV: Creating Data Structures

14. Relating tables to each other
15. Controlling how data is stored
16. Speed access to the data with an index
17. Masking the data with views
18. Storing SQL Server queries using stored procedures

Part V: Tuning and tweaking SQL Server

19. What is the server doing?
20. Which queries are called most often?
21. Ways to make the machine work for you
22. How did that query run?
23. More ways to control SQL Server
24. Things to make your life easier
25. What to do when things go wrong
26. How and Where to Get Help

Appendix: Installing SQL Server

1

Dive into SQL Server

SQL Server is a generic name for a fairly large and diverse set of software that manages various different aspects of information within a business. Most of the time when someone refers to SQL Server, what they're actually talking about is a particular piece of the software called the relational storage engine, software that stores data in tables that can be related to each other.

The idea behind the book is to get you started using SQL Server with all its complexity and functionality. And the idea behind this chapter is to get you acquainted with the fundamental concepts around SQL Server and some of the concepts behind the theory of relational storage. But before we can get into the real work of managing SQL Server, we're going to start with the fundamentals of servers, databases, and database management so we're all on the same page.

1.1 What Is SQL Server?

Microsoft SQL Server, as a term, can be pretty misleading. First off, what is SQL? It stands for Structured Query Language, and it's the scripting language used to access the data within a database, which is part of the relational storage engine, described in detail in the next section. For the rest of the book, when referring to SQL Server, we will be talking about the relational storage engine and not all the other tools that Microsoft has crammed into the software. Let's quickly cover the various tools available to you.

1.1.1 SQL Server Tools

There are four large pieces of software that are marketed and sold as SQL Server.

- SQL Server
- SQL Server Integration Services (SSIS)
- SQL Server Analysis Services (SSAS)

©Manning Publications Co. Please post comments or corrections to the Author Online forum:

<http://www.manning-sandbox.com/forum.jspa?forumID=798>

- SQL Server Reporting Services (SSRS)

These different pieces of software are not necessarily used together at all times. Figure 1.1 shows what they do and how they relate to one another.

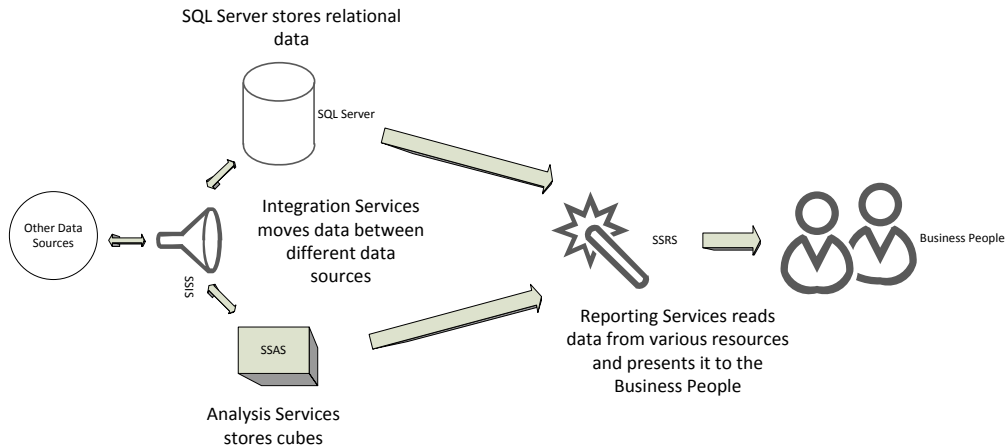


Figure 1.1 The different services that make up SQL Server

Just so we're clear as to which piece of software is for which task, let's review them.

SQL SERVER RELATIONAL ENGINE

SQL Server is a relational database engine. As a storage engine, SQL Server stores information in the form of tables. As a relational storage engine, these tables are related to each other and these relationships define what are called data structures. These data structures mirror the things that your business manages, uses or constructs.

The idea of storing data in a relational way goes all the way back to 1970 when the original scientific white paper outlining it was published by Ted Codd at IBM. Computing has come a very long way since then, but the concepts are still the same. The core functionality of the relational engine relies on the concept that information can be related to other information. This concept is in evidence all over the place in the real world. For example, look at Figure 1.2, your company's physical address can be related to your company or to you. Both you and your company are located at that address. Further, you can be related to your company as an employee or contractor. Also, you can be related to the other people at the company, you have peers and supervisors who also have peers and supervisors, all related to one another. This is the concept that relational databases attempt to model.

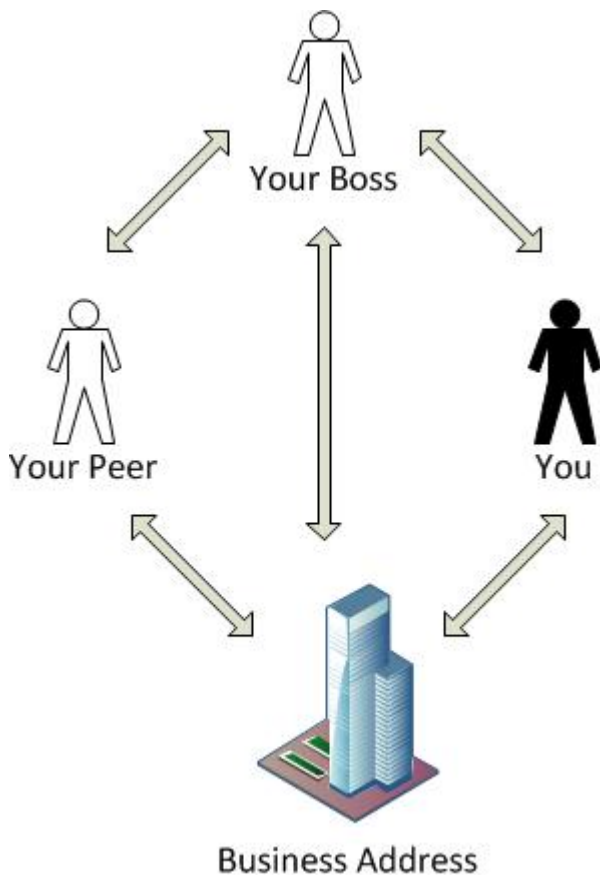


Figure 1.2: You have a relationship to a business address and to your boss. Your boss has a relationship to you, your peers and a business address.

REPORTING SERVICES

Reporting Services is a whole series of tools for creating, distributing and securing reports. These reports can be simple little grids that look a lot like Excel, or they can be extremely complex and rich multi-media experiences with maps and interactive graphics that look like a full blown application.

INTEGRATIONS SERVICES

Integration Services is Microsoft's tool for moving data into and out of various locations where data is stored. This can include SQL Server databases, and frequently does. But it can also include files downloaded from a mainframe or Excel spreadsheets, Access data tables, or even other database systems entirely. There are ways of connecting to almost any data

source to read from or write to through Integration Services. Integration Services is frequently referred to as an Extract, Transform, and Load (ETL) tool.

ANALYSIS SERVICES

Analysis Services is Microsoft's data cube processing engine. Cubes are an entirely different way of storing and extracting data completely separate from relational data engines. Cubes are principally a way of reading and examining data. They are not a data input location. Analysis Services is much more focused on complex mechanisms of presenting data and data mining methods.

With the parts of SQL Server out and on the page, let's go a bit deeper in the SQL Server relational engine. As I've mentioned before, SQL Server is a relational database engine, which means that it stores data that relates to other data. Figuring out how to define these relationships and what is stored is a major part of designing a database. Even if you're not planning to design a database from scratch, you're going to need to read data from databases that were designed by someone using some type of relational model.

In order to work with SQL Server, you need to know how the different storage mechanisms are arrived at. This may feel like a lot of theory in a chapter one and for a book that wants to get you up and running quickly, but I promise it will be helpful to you as you continue through the book.

1.2 Data Structures

If you wanted to store all the information about a person that might be of interest to an insurance company, how would you go about it? The first thing you might do is think about the information you need. You're going to want to track information about the person such as their name, birth date, height, weight, and address. You'll also want to keep track of their billing information, how much they owe, how much they've paid, any claims you've paid out to them. As you start to think of this information you could write it all down like you see in Figure 1.3:

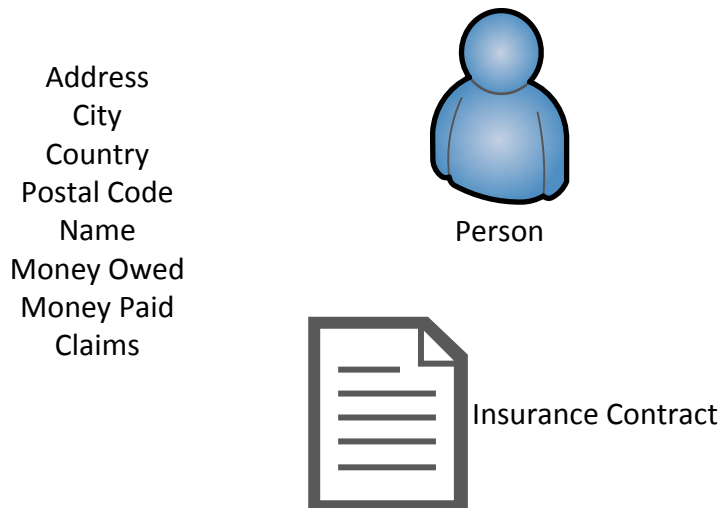


Figure 1.3: The objects and properties that might make up a database for storing insurance information

Then the real thinking begins. Take addresses for an example. Do most people live forever at the same address? Some do, but most do not. They may move several times. Do you want to have to type in the address over and over even though it may be the same address for multiple people? The address will need to be stored separately from the person because of this.

You've now established your first relationship and you can keep going from there. The good news is there are several different ways of laying out data in relationships so you don't have to try to figure it all out on your own. There are a large number of very well defined approaches to this, but you can focus on two general approaches: relational models and dimensional models. The more well known approach is the relational model.

1.2.1 Relational Model

The relational model is defined through an undertaking known as data normalization. Data normalization is a process of following a series of rules to rearrange the information you've defined into a data structure that is considered to be normal or normalized. The goals of normalization are to get to the point where you have good information stored in your system that is accurate and not duplicated. A properly normalized database makes putting data into the system easier and makes getting data out of the system easier. If you find that either of these simple rules are not being met, there's a good chance that the normalization process is off track.

There are a number of different types and levels of normalization, but it's best to focus on the simplest, Codd Third Normal Form. The basic rules of this type of normalization are:

1. No repeating groups
2. Meets 1 and No data in the table dependent on another key
3. Meets 1 and 2 and all data in the table is dependent on the key and not on any other columns

Yes, this is somewhat hard to understand, but take it back to the insurance data talked about earlier.

NO REPEATING GROUPS

A repeating group would be something that occurs over and over again, repeating data. What might be a repeating group from the information above? The Money Paid stands out as one example. This means that this information would be removed from the definition of the Person table that we're beginning to define.

NO DATA DEPENDENT ON ANOTHER KEY

Now you have to think about information that might be dependent on another key. Defining a key would be the most important part here. A key is a unique identifier, some piece of data that completely identifies all the other data associated with it. In some situations you won't be able to identify a unique value. In these instances you may have to use what is called an artificial key. It's just assigning a number, like a student ID number in college, that will go everywhere with that piece of data wherever it goes within the system.

Back to the example of the Person of interest to the insurance company, information like the address would not be dependent on the key for that person. It would need it's own key. This information would also have to be moved to a different table.

ALL DATA DEPENDENT ON THE KEY

Finally you have to think about all the other pieces of information that are in the table and try to figure out if they are not applicable to the key, don't belong in that particular table but maybe should be stored elsewhere. In this case the PolicyNumber is directly related to that person, but it's not a fundamental part of the person and it's something that could be changed or replaced as business transactions occur, so you actually might need more than one of these. That means that this piece of data is not dependent on the key.

After breaking down the information this way, comparing your data to the rules of normalization, you arrive at the data structure shown earlier in Figure 1.4, that is to say, a set of definitions about the data that begins to describe what it might look like if it were to be stored in a database. Figure 1.4 represents a normalized data model:

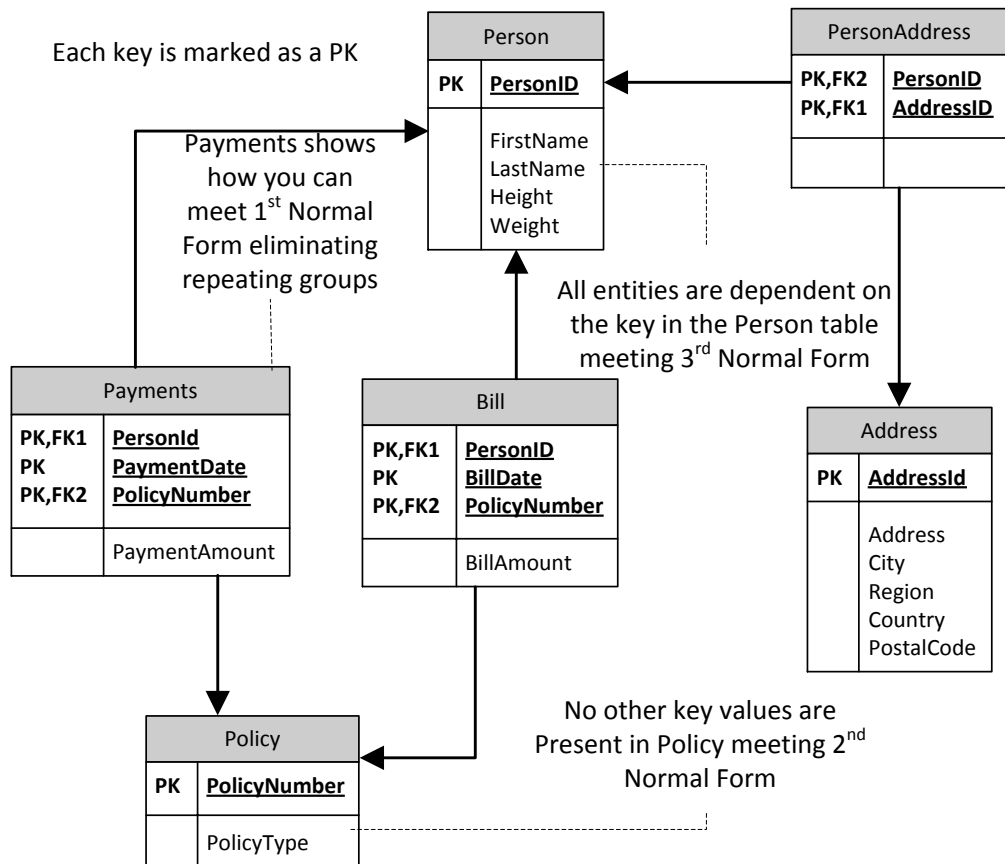


Figure 1.4: The beginnings of a normalized relational model. The solid lines are relationships. The dashed lines are to comments.

While this is a relational representation of the data, it is possible to come up with other methods of representing the same information. The relational style above is well suited to applications focused on data entry. But systems that want to just report on the data can benefit from other types of designs.

1.2.2 Dimensional Models

Collecting data through an application is not the only use for databases. Another major use for databases is to feed that information back to business people in order to support their decision making processes. While you can write queries against the more traditional

©Manning Publications Co. Please post comments or corrections to the Author Online forum:

<http://www.manning-sandbox.com/forum.jspa?forumID=798>

relational structures of a normalized database, those queries may not always be terribly efficient. As the amount of data you have to maintain grows and the number of different types of reports grows with the data, you'll find that your relational structure just is not enough. There are alternatives to the relational structure that are better suited to the specialized purposes of reporting. It's good to understand what these structures look like so that you're not surprised when the databases under your charge don't follow that relational methodology, you can recognize the other type.

A dimensional structure is based off of two general concepts:

- 1) you have a fact
- 2) that fact has dimensions

The most common dimensional model type is the star schema. The star schema is so called because the shape roughly resembles a star with a center point and arms point out in all directions. This model is better suited to the types of queries that reporting does, aggregations and searches on criteria other than the primary key of a table.

Taking the same example above, of a person with their insurance policy payments and bills, you would need to rearrange the data. You can see the Fact at the center of Figure 1.3 as the circle. It describes information about the Insurance Bill.

A dimension table is the information that describes everything around the piece of business data that the fact represents. You can see Dimensions around the Fact table such as the Person or Date dimensions.

Figure 1.5 shows a dimensional model:

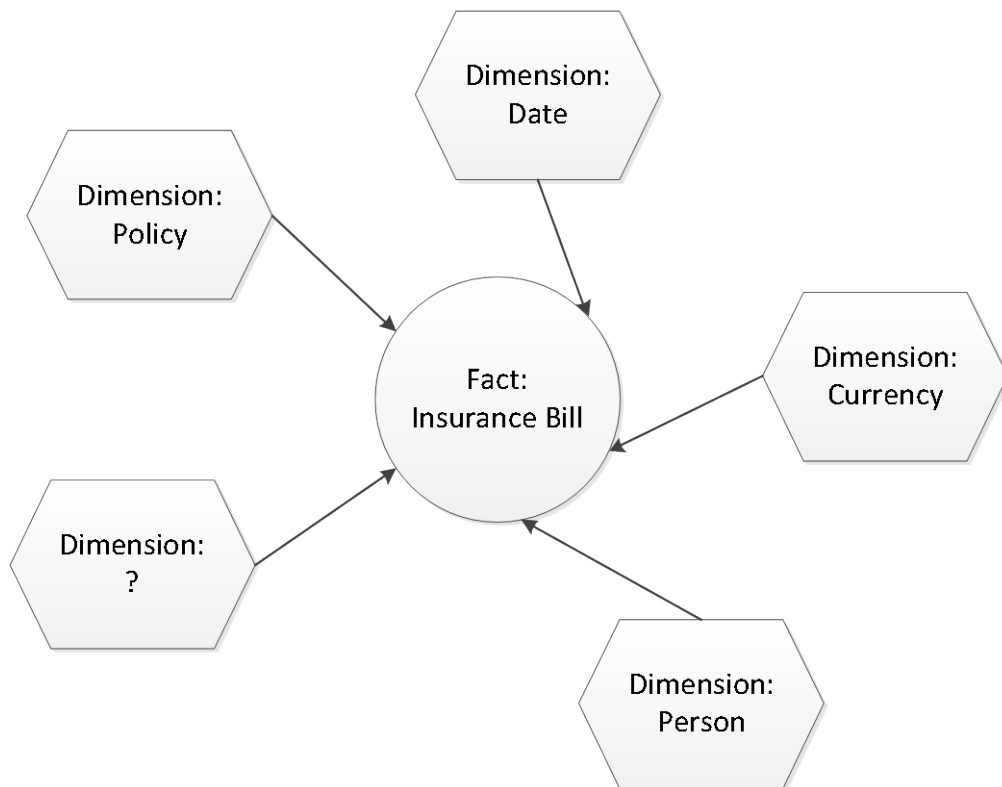


Figure 1.5: Example of a dimensional model showing the fact and dimension tables laid out in a star schema

The thing to remember about the dimensional model is that it doesn't represent all the possible data in a system, but rather a specified set of data that defines the fact. This means, when picking the address or addresses that might be included, you would take into account the that the address of an insured house is not nearly as important as the billing address, although they maybe one and the same. You might also want an employer address if the insurance is through an employer.

When all put together the star schema looks something like Figure 1.6:

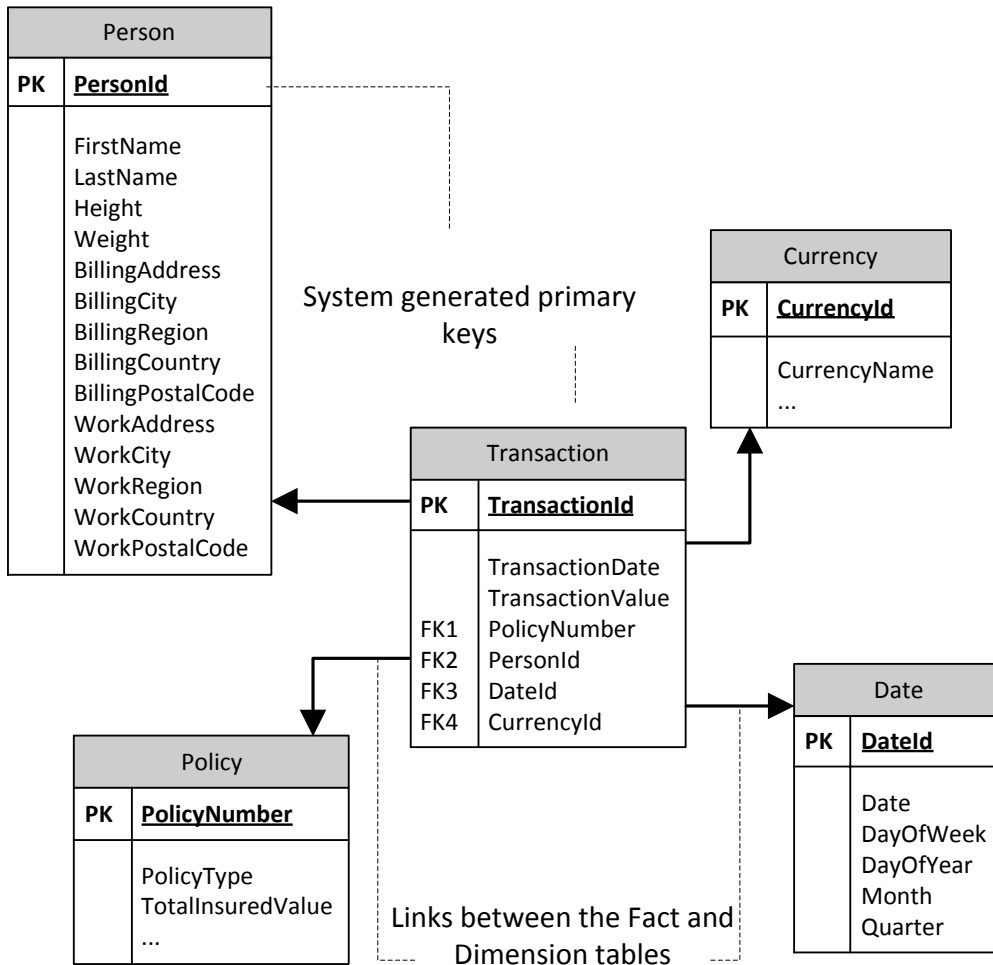


Figure 1.6: The dimensional model as it would look in a database.

You can see how the information is stored differently than with the relational structure, but you can also see how it still takes advantage of the relational nature of SQL Server. It's worth noting again, you wouldn't use a structure like this for data entry because you're far too likely to end up with different spellings in the addresses or who knows what kind of errors in the data.

That's more than enough theory. You have some idea now of what you can expect to see in the databases that you're going to be managing. It's time to move on to getting your hands dirty, in a figurative sense, with the SQL Server software itself.

1.3 Working With SQL Server

The best way to learn any software is to start using it. There are a bunch of software tools in the SQL Server toolbox, but the biggest and most important is SQL Server Management Studio (SSMS). SSMS is where you'll spend most of your time when you start to work with SQL Server. SSMS provides a variety of different graphical user interfaces for creating databases, setting up security, reading data out of the database, and all sorts of other functions you'll need to work with your SQL Server instances, the databases stored there, and all the stuff inside those databases. SSMS also supplies you with an interface to the basic scripting language of SQL Server, called Transact Structured Query Language, or Transact-SQL, or just T-SQL for short. In this section we're going to connect to SQL Server for the first time and start to explore the SSMS interface.

1.3.1 Connecting To SQL Server

To get started with SQL Server after you have installed it (See Appendix A for an overview on installing SQL Server), open SQL Server Management Studio from the Start menu on your computer. When it opens you'll be presented with a connection window. Depending on how you installed SQL Server you have choices and decisions to make at this point.

The installation may have been a default install, which means that the server name is all you need to connect to SQL Server. If not, then an instance name may have been specified, with a backslash between the server name and the instance name. In addition to the server and/or instance name, you also have to worry about how security was configured. Most installations will be using Windows Integrated Security. This means that your Windows login gives you access to SQL Server. In some places you'll be given a user name and password, this is known as a SQL Server Login. Most of the time your connection to SQL Server will look something like Figure 1.7:

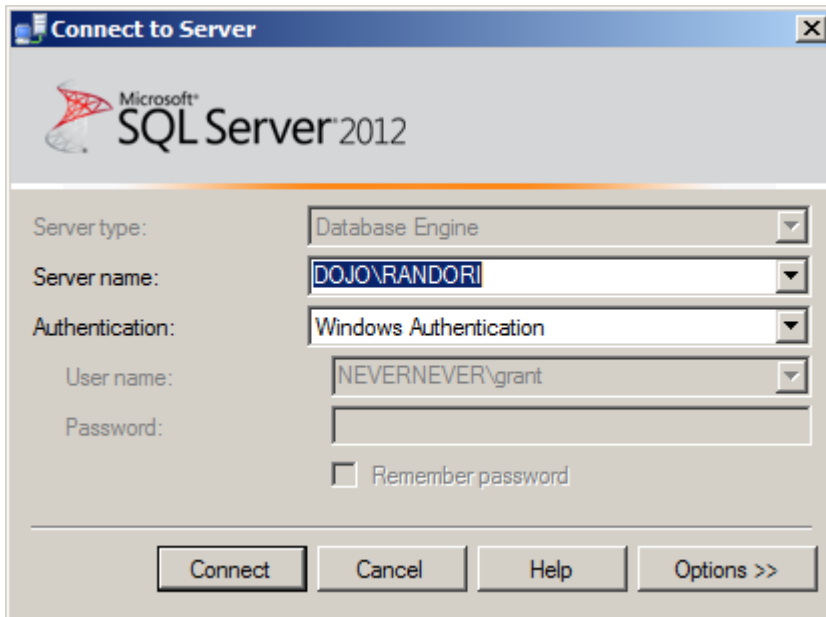


Figure 1.7: SQL Server's connection window filled out for an instance and Windows authentication

After you configure the connection correctly you can click on the Connect button. This will connect you to SSMS directly. If you have been following along, you're now connected to your SQL Server instance. I'll run through some of the basic functionality of the various windows in preparation for the work you'll be doing with this software in the future.

TRY IT NOW OPEN SSMS AND CONNECT TO A SERVER

1.3.2 Working With the Object Explorer

When SQL Server Management Studio (SSMS) opens, by default it will be laid out as you see in Figure

1.8:

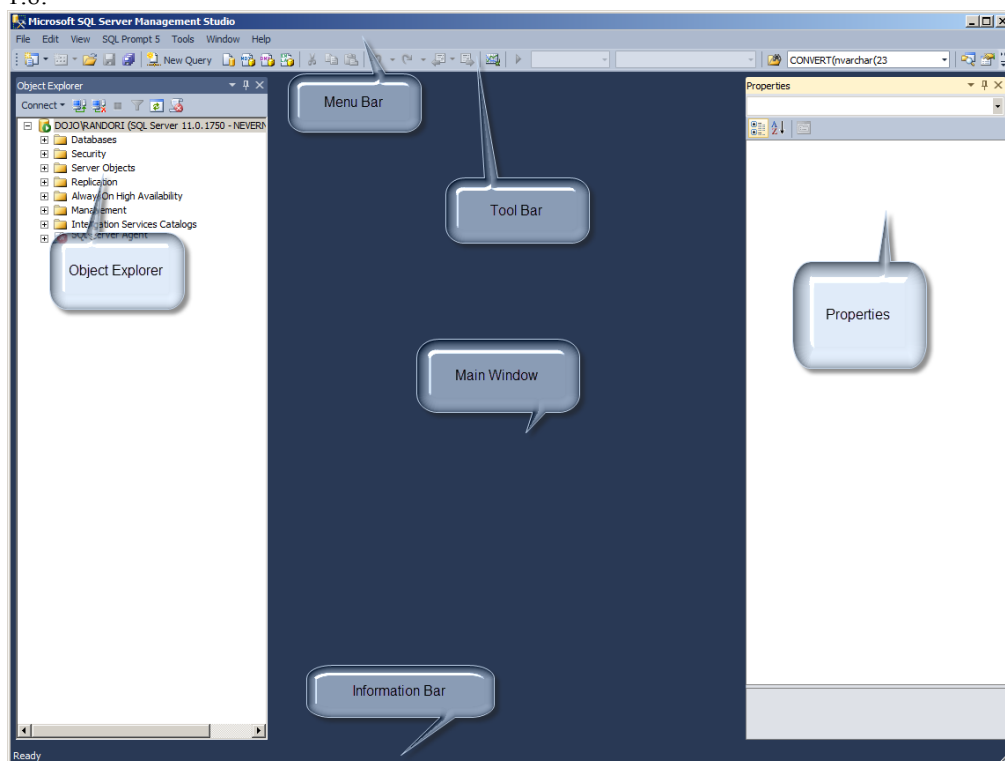


Figure 1.8: The main parts of SSMS identified

SSMS is built out of Microsoft's programming environment, Visual Studio, and shares a lot of the same functionality. Windows are very customizable:

- You can have the windows float independently of each other
- You can mount them in various locations on the screen, creating tabbed environments if that's what you prefer
- You can also pin and unpin the windows causing them to float in and out of view as your mouse pointer gets close to them by clicking on the push-pin icon on the window.

Try clicking on the push pin visible in the Object Explorer window. The window should disappear leaving a little tab on the left side of your screen that is labeled Object Explorer. If you move your mouse pointer over that tab and leave it there in what is called a hover, the Object Explorer window will pop back out. Click on the pushpin icon again and the Object Explorer window will be docked back on the left side of the screen as it was before.

©Manning Publications Co. Please post comments or corrections to the Author Online forum:

<http://www.manning-sandbox.com/forum.jspa?forumID=798>

TRY IT NOW SEE IF YOU CAN CHANGE HOW THE OBJECT EXPLORER WINDOW BEHAVES BY CLICKING ON THE PUSH PIN ICON.

. You'll be spending a lot of time working with the Object Explorer so it makes sense to become familiar with it first. SSMS will allow you to connect to multiple SQL Server instances. As you connect to each instance, you'll see them listed in the Object Explorer. Each server will have a little plus sign next to it that opens the tree for that server. The tree is a list of the different types of objects that you can begin to manage from SSMS. When you connect to a server, the server name is at the top of the Object Explorer windows. Below that are the different parts of the tree (Figure 1.9). I've named my server DOJO after a martial arts training hall. And my SQL Server instance is named RANDORI, which means wild testing.

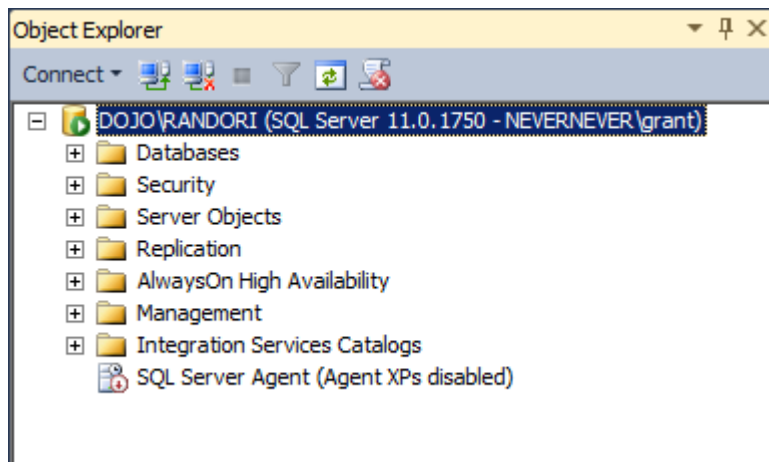


Figure 1.9: Default view of the Object Explorer tree for a new server connection

1.3.3 Navigating Within the Object Explorer

Learning the behavior of the SSMS tool is going to be an ongoing process because of all the options available to you. Get started by clicking on the plus sign next to the folder labeled Databases. This opens the list of Databases for your server. Since there are no databases installed by default and you probably haven't created any, you should only see two more folders on the tree; System Databases and Database Snapshots.

Click on the plus next to System Databases and you'll see four of the databases that SQL Server uses to manage its own operations: master, model msdb and tempdb. Don't worry about what these do yet. Click the minus sign that is now next to the System Databases label. This closes that list. Click the minus sign next to the Databases label and you'll close that list as well. This is how all the various objects can be accessed through the Object Explorer.

TRY IT NOW FIND DIFFERENT OBJECTS IN THE OBJECT EXPLORER WINDOW. YOU'RE GOING TO BE SPENDING LOTS OF TIME USING THIS SOFTWARE, YOU MAY AS WELL GET FAMILIAR WITH IT

There are other ways to manipulate objects in the Object Explorer as well. Right click with your mouse on the Databases label inside Object Explorer. This action opens a context window that provides other functionality for you. You can see the context menu in Figure 1.10. You'll be using a number of these functions outlined in the menu throughout the book.

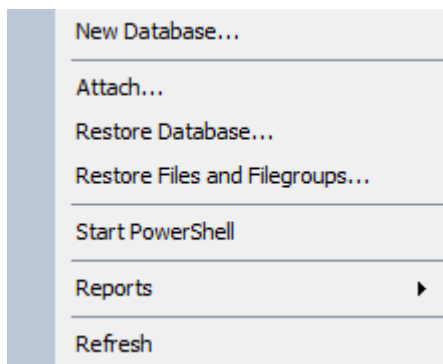


Figure 1.10: The context menu for databases in Object Explorer

Across the top of the Object Explorer are a number of icons that allow you to perform other actions. On the left is a drop down menu that lets you connect to another instance of SQL Server, labeled as Database Engine, Analysis Services, Integration Services or Reporting Services. Next to this, as you can see in Figure 1.9, are a row of icons. The first is for connecting to another instance of SQL Server, although, if you hover the mouse of the icon it will show a label that says Connect Object Explorer. Next to that is an icon that will disconnect you from the server that is currently selected in the Object Explorer window. Try clicking on that icon. The tree in the window of the Object Explorer will go away. To get it back you'll need to reconnect to your server, as you did before. But you can open the connection window using one of the two methods explained above in section 1.3.1

TRY IT NOW EXPERIMENT WITH USING SOME OF THE BUTTONS AT THE TOP OF THE OBJECT EXPLORER WINDOW

1.3.4 Using Menus and Shortcuts

Take a look at the top of the screen where the menu bar is located in Figure 1.8. There are a number of options here, but the one that should concern you the most is all the way on the right, Help. Click on the Help menu and you'll see a number of different choices. The most important one is at the top, View Help. Click on this menu choice. If this is the first time you've tried to open help, you'll be presented with a choice, do you want to connect to online help or not. Online help is more up to date than any help files installed on your system. This means that you'll get much more accurate responses, but it is slower and a

©Manning Publications Co. Please post comments or corrections to the Author Online forum:

<http://www.manning-sandbox.com/forum.jspa?forumID=798>

little more difficult to manipulate than the local help file which is called Books Online. Production servers in some companies can't connect to the Internet either, so this decision depends on your situation.

The most important thing to learn about SQL Server is that there are always a lot of ways to get things done. Getting to the Help window is important. Even more important is getting to the right kind of help. You can click on the menu as we did above, but you also have the ability to open help in the context of the window you're currently working in. Any time you'd like to see help for the window you're on, press the F1 key on your keyboard. That will open the help screen.

The menus across the top of the screen can change depending on what you're doing and which windows are open in the screen below. Don't be surprised when you see new menu items sometimes appearing as you work through the examples in the book. The menus are taken from Visual Studio so they may not always seem applicable to an environment for managing databases, but you're given a lot of functionality through these menus. Some of it is duplicates of the functionality you've already seen. For example, clicking on the File menu choice will show a number of menu items. At the top is the option to Connect Object Explorer with an ellipsis. This will open the connection window that you've already seen.

TRY IT NOW TRY USING SOME OF THE COMMANDS WE JUST TALKED ABOUT AND EXPLORE SOME OF THE OTHER MENU CHOICES SO THAT YOU UNDERSTAND SOME OF WHAT'S AVAILABLE TO YOU. IT'S ALWAYS A GOOD IDEA IN SQL SERVER TO KNOW MORE THAN ONE WAY TO GET SOMETHING DONE.

1.3.5 Gathering Additional Information With Object Explorer Details

Click on the View menu. This shows you a very large number of things that you can bring up for viewing or modification on your screen. Click on Object Explorer Details. You should see a second window open to the right of your Object Explorer window. Depending on what is currently selected in the Object Explorer, you may see things displayed in the Object Explorer Details window. You'll notice that this window is slightly different at the very top. It has a tab. This larger area within SSMS is set up for tabbed viewing by default. Like the other windows you can customize this. There are a number of different types of windows that will be displayed here and they will all remain open until you close them, arrayed in a series of tabs across the top of the screen.

To see the Object Explorer Details at work, click on the server and instance name at the top of your Object Explorer window. This will bring up a list of objects in the Object Explorer Detail window that looks basically identical to the tree list in Object Explorer. You can select these objects in this window, but unless you double click them, you won't open any of the folders. Don't explore just yet. If you have something selected in the Object Explorer detail window, click in the white space to the right or below the list of objects. Look down near the bottom of the window. You should see a gray box stretching across the whole Object Explorer Detail window that looks something like Figure 1.7. With the server and instance selected in Object Explorer and no other objects selected in Object Explorer Details, this gray box should show you information about your server and instance.

TRY IT NOW LOOK AT THE OBJECT EXPLORER DETAILS. YOU MAY NEED TO DRAG THE DARK GREY BORDER OF THE BOX UP TO BE ABLE TO READ EVERYTHING, OR YOU CAN USE THE SCROLL BAR ON THE RIGHT.

Scrolling around you can see various pieces of information about your server such as the current version of SQL Server that is installed there, the number of databases, which is probably four, and other information such as the Max Server Memory. All this information is available to you, and more, as you select different objects in the Object Explorer window. Try selecting a few and see how the information on display in the Object Explorer Details window changes.

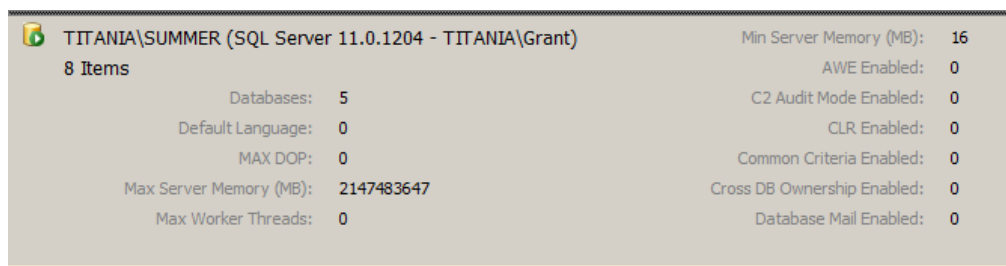


Figure 1.11: Server properties in the Object Explorer Details window

As you can see there is a lot of information available to you in SQL Server in different forms. Learning all the different ways you can see this information and knowing what it means are just part of what you need to learn in order to get good at manipulating the server.

There are a number of other windows and views into the data that will be explored in the rest of the book. You now have the basics to enable you to begin to explore your system.

1.4 Summary

In this chapter we spent a little time explaining some basic database theory in order to establish just what is meant by a relational storage engine. You learned about data normalization and dimensional models as different mechanisms for using the relational engine for storing data. We also dipped our toe in the water and started to use the SQL Server Management Studio, the most important tool in the SQL Server toolbox. All this is just to get you ready for your very first action within SQL Server, creating a database in Chapter 2.