Chapter 1

Overview of Microsoft SQL Server 2008

SQL Server 2008 is a very complex product. It definitely deserves a little bit of introduction. Even if you have been working with SQL Server for quite some time, you might find some new features and nuances in the product that may catch you off guard, but they might delight you as well. In this chapter, we will focus on where SQL Server fits into the grand scheme of Microsoft architecture as well as its placement in your own computing enterprise.

In this chapter, you will learn to:

- Use Architect SQL Server services in the typical IT environment
- Install SQL Server 2008
- Use the Microsoft SQL Server toolset
- Implement other useful third-party tools

SQL Server in the Enterprise World

The authors of this book have been working with SQL Server for over a combined 28 years. Back in the early days, it was a struggle to defend the product. We came to SQL Server with the same background as many other PC developers of our age, having worked with Clipper, Access, and other similar products. But SQL Server was different, and it required a change of perspective from the older tools with which we worked. Because Mike had worked for IBM, he had some previous experience with mainframes. Both Mike and Gentry also did some work with Oracle. However, SQL Server was definitely new territory compared to some of these older tools.

We quickly discovered, by playing with the product and trying to put it through its paces, that SQL Server had a lot of potential, but it still had a long way to go. Part of the problem was developer education. Many of the problems that we came across were the result of developers trying to build applications with SQL Server using the same mindset and techniques that served them well during the Clipper days. Obviously, SQL Server was not Clipper. Some things had to change.

SQL Server 2008 has come a very long way since those days when we first started to kick the tires, but one thing has not changed: the need to understand the big picture. SQL Server is a "big picture" product. Somewhat like an aspen grove where all of the trees share a common root system, SQL Server is integrated tightly with many other services and layers in the typical Microsoft-based IT environment. It is not strictly for storing user data any more. The sooner you grasp that, the better you will understand the logic behind the product, its architecture, and the direction that Microsoft is going with SQL Server.

The Need for an Enterprise System

Most people reading this book should have some prior experience with SQL Server. To that end, we do not need to explain the rationale for enterprise data access. Our data is the lifeblood of our business. Some of the authors' work has been done in heavily regulated industries such as insurance and casino gaming. In those environments, the need for data archival, the types of reports produced, and other data requirements are often dictated by governmental and other regulatory bodies. Without the data and programmatic resources to support those requirements, the organization cannot function.

As we stated previously, in the early days of SQL Server, it was somewhat difficult to defend it as a truly enterprise product, but starting with the 2005 release, SQL Server started to challenge that perception. The key, though, is to effectively match your business's enterprise-level requirements with the features that SQL Server brings to the table. The job is always easier if you are using the right tool. Some of the typical enterprise data requirements of an organization include:

Interoperability In many cases, SQL Server data must be available to applications outside of your Microsoft Windows infrastructure. With the strong presence of mainframe systems, UNIX and Linux environments, Java platforms, etc., data interoperability requirements may be extensive.

Performance Performance often means different things to different people. To the end user, it is typically about how fast they can get their data. For an administrator, the primary concern is maximizing overall throughput. These are often conflicting goals, and the database system must be able to balance them.

Security The integrity of your data is only as good as your ability to secure it, but security these days goes beyond the typical authentication/authorization issues of the past. We must also be able to manage encryption processes, backup security, etc., while at the same time being proactive to prevent the security exposures of tomorrow.

High Availability What good is having the data if it is not available to the people who need it when they need it? A good enterprise framework must ensure that we have adequate redundancy and the means to minimize downtime. Every second that the data system is not available can mean lost revenue, and in some cases, legal repercussions.

Automation The more complex and larger a system becomes, the more important it is to be able to automate routine tasks and proactive maintenance to ensure that data availability and integrity is maintained. Otherwise, the typical database administrator (DBA) will be completely overwhelmed with day to day maintenance.

Centralized Reporting Storing data isn't much use if you can't get data out when you need it in the formats that you need. A number of reporting environments are available. Whichever one you use, it is essential that the reporting tier be efficient, be flexible, and have the lowest possible footprint in terms of resource consumption.

This is a pretty short list. Off the top of your head, you could probably double this list right now. In order to be of any real value, a database system must be "enterprise ready" and not just a data engine with a collection of unrelated supporting services. That won't work in today's IT environment.

So, if SQL Server 2008 is our enterprise solution, how does it address these and other issues?

SQL Server Features

All right, we know that this sounds like a marketing speech, but we really do have a rationale for covering features. It is important to have a good understanding of what a system can do if you want to design effective solutions. Besides, some of the features are new, and you have to have some justification for upgrading to the 2008 version when the accountants knock on your door.

SQL Server Editions

Be aware that much of this information is available electronically in SQL Server Books Online, either locally installed or web-based, as well as a variety of other good Internet and published resources. The goal here is to synthesize it for your convenience. You may want to check other online resources, though.

Let's begin by looking at the different editions of SQL Server 2008. SQL Server 2008 is available in a number of different versions. The primary distributions are the Standard and Enterprise editions. Additionally, Microsoft has release four specialized editions of SQL Server, namely Workgroup, Web, Compact, Developer, and Express. Each has its advantages and disadvantages. Because the Compact version specifically targets embedded systems, we will not discuss it here. All of the following editions come in 32-bit and 64-bit versions, although only Enterprise and Developer versions support IA64.

SQL Server Express Edition

- Intended for embeddable, lightweight, and standalone solutions
- Supports one CPU
- 1GB maximum addressable memory
- 4GB maximum database size
- Free download and distribution

SQL Server Workgroup Edition

- Intended for small user loads, low-use web applications, etc.
- Supports two CPUs
- 4GB maximum addressable memory on 64-bit. OS Maximum on 32-bit
- No maximum database size

Web Edition

- Intended for larger traffic web applications
- Supports four CPUs
- OS maximum memory support

Standard Edition

- Intended for most applications
- Supports four CPUs
- Memory up to OS maximum

Enterprise Edition

- Provides the highest level of scalability and enterprise features
- Unlimited CPUs to OS maximum
- Memory up to OS maximum
- Numerous enterprise features not found in the Standard Edition such as:
  - Database mirroring
  - Database snapshots
  - Online indexing
  - Online page restores
  - Distributed partitioned views
- Numerous business intelligence (BI) features not found in the Standard Edition, such as:
  - Scale-out report servers
  - Infinite click-through reports
  - Text mining
  - OLAP dimension and cell writeback

Developer Edition

- Functionally equivalent to Enterprise Edition
- Licensed only for development environments, not to be used in production

Choosing the right edition is important. It needs to satisfy your current needs as well as your projected ones. You need to make sure that your database will support your performance and scalability requirements. You must also consider database size and the need for enterprise data and BI features.

SQL Server Services and Features

The feature list for SQL Server is extensive. Because we didn't want to fill these pages with redundant information, we ask you to refer to the SQL Server Books Online. There you will be able to find a great deal of information regarding the essential and expanded features of SQL Server.

A few specific features are worth mentioning. These are the big ones. Not all of them will fit into your enterprise plan, but some may, and the fact that they all are bundled with SQL Server can be incredibly convenient and cost-effective.

Support for Unstructured Data For the average database administrator (DBA), there are two kinds of data: the data in the database (structured data) and everything else (unstructured data). Unfortunately, most organizations have a significant amount of unstructured data that they must manage ranging from documents to pictures to proprietary data. Through FILESTREAM data, large binary data streams can be stored in the file system without sacrificing the transactional integrity of the database system.

Policy-Based Management Policies make it easier to enforce best practices and standards. You can configure policies such as naming conventions, configuration standards, and coding standards. You can enforce these centrally for the entire organization. Not only does this simplify enforcement of standards, it also allows central modification of policies as needed.

SQL Server Integration Services (SSIS) SQL Server Integration Services (SSIS) provide exchange/transform/load (ETL) functionality for SQL Server 2008. SQL 2005 took a significant step forward from DTS when it was released, providing not only more intuitive designers, but also a native .NET execution environment. SQL Server 2008 continues that process with significant pipeline scalability improvements as well as the ability to provide persistent lookups, which is a significant data warehousing enhancement.

SQL Server Reporting Services (SSRS) Although not a new feature, this is, in our opinion, one of the most important enterprise features that SQL Server provides. The tight integration with SQL Server and the exposure of the reporting engine through a standard web services interface make Reporting Services one of the most compelling reporting solutions available for SQL Server 2008. Enterprise features such as web report distribution, subscriptions, and infinite click-through make this one feature you should definitely evaluate for inclusion in your enterprise data architecture.

Full-Text Search Services Full-text indexing and special query syntax let you effectively search large text blocks for patterns, word forms, proximities, and other elements. SQL Server 2008 provides some compelling enhancements to the full-text set of services. They include features such as improved full-text index management and performance enhancements. If you haven't at least played with Full-Text Search, you owe it to yourself to spend a few minutes getting cozy with it.

Business Intelligence Business Intelligence (BI) content is critical to the success of SQL Server 2008. The BI agenda for SQL Server 2008 was very aggressive. One of the reasons that the schedule slipped for SQL Server 2008 was that the BI features needed a bit more polishing.

The cornerstone of this version's BI functionality is SQL Server Analysis Services. Supporting data warehousing through multidimensional data models (data cubes), Analysis Services provides mechanisms for analyzing and mining the large data volumes that most organizations have collected over the years. It is also highly integrated with other services, relying on SSIS for warehouse loading and SSRS for reporting on multidimensional data, including ad hoc reporting through the report builder.

This is another list that we could extend significantly, but our objective here was to give you a taste for the services available in SQL Server 2008, show you where some of the major enhancements fit, and provide a better framework for your own investigation and research. We strongly recommend that you take the time to review the SQL Server 2008 feature documentation in the SQL Server Books Online. As a system architect, you will not be able to design components into your system if you do not know what is available and how it works.

SQL Server's Place Within Microsoft Architecture

To say that Microsoft's enterprise strategy relies on SQL Server is an understatement. Many of the products in Microsoft's enterprise line are dependent on, or have significant integration with, SQL Server at some level. Data services are required everywhere, and SQL Server is the glue that holds many of them together. Among the Microsoft products with SQL Server dependencies or integrations are

- Team Foundation Server
- Systems Management Server/System Center Configuration Manager
- Office SharePoint Server
- Office Project Server
- Operations Manager Server
- BizTalk Server
- ADO.NET

The logic is sound. Microsoft has already created a very compelling data engine, and when you need data services, there is no sense in reinventing the wheel. Of course, different tools require different levels of service support. For example, it is very common for SharePoint Server to be installed with SQL Server Express Edition, but a utility like Team Foundation Server might require real enterprise scalability.

We added the ADO.NET to this list, and at first glance it might seem a little out of place. The rest of the elements listed are Server components, but ADO.NET is a .NET API, used to provide data access support for .NET applications. We've included it because although ADO.NET can provide connectivity to a wide variety of different database systems through their own .NET Data Providers, the SQL Server Native Client is specifically and tightly integrated with the ADO.NET API itself, providing API-level support for features such as SQL notifications and automatic failover with database mirroring.

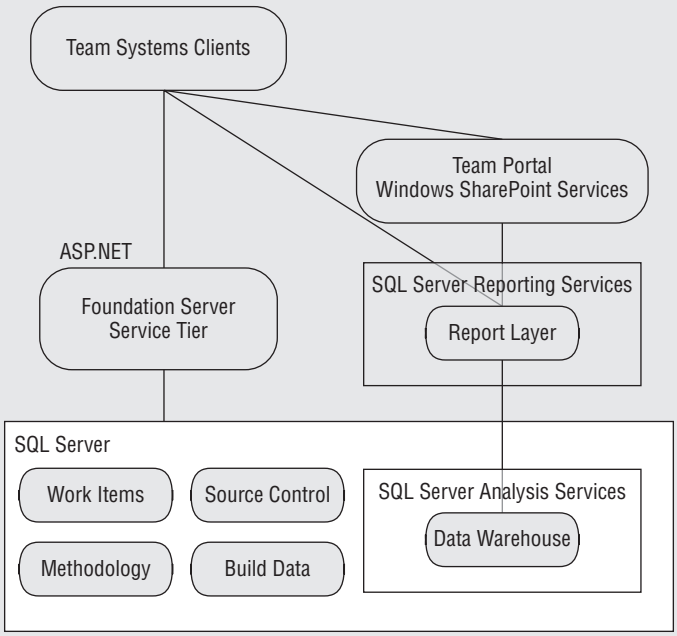
The lesson to learn from this discussion on Microsoft integration, however, is that sometimes it's a good idea to expand our definition of what we consider "data services." Especially now that to SQL Server 2008 supports unstructured data, the SQL Server Data Engine may play an ever more expansive role whenever and wherever any data is needed in any format and for any purpose. If we remember that and learn to take full advantage of the services that SQL Server provides, we will be able to design more comprehensive architectures, comfortable that SQL Server will provide data services at any level necessary, from embedded to enterprise.

Real World Scenario

SQL Server Integration with Team Foundation Server

One of the most extreme real-world examples of SQL Server integration in the Microsoft Architecture is Team Foundation Server. This is a perfect case study in the prevalence of data and how SQL Server can provide data services whenever and wherever they are needed.

When author Mike Lee teaches classes on Team Foundation Server, he frequently uses a diagram similar to the one shown here to illustrate the architecture of Team Foundation Server. Note the extensive use of SQL Server services in the diagram.



To do its work, Team Foundation Server requires significant data services. Rather than provide the data tier as part of the Team Foundation Server product itself, the Foundation Server delegates all of its data requirements to various SQL Server services. Some of these are creative in their application.

Work Items A Work Item in Foundation Server is essentially an assignment to complete a specific unit of work. When a project manager on a team needs work done, they create a Work Item in Foundation Server and assign it to the resource that is responsible for that item. These Work Items are database entries that document the work and its history.

Source Control Repository Traditionally, source control storage has been file system based, but in Team Foundation Server, all source code and all historical versions of that code, along with their labeling, are stored in SQL Server tables. This makes source control significantly more scalable and stable in other environments such as Visual Source Safe.

Methodology Data An important part of Team Foundation Server is its support for development methodologies such as Agile. The choice of methodology affects many factors in the system, such as the types of Work Items, reports, and process guidance documentation. Methodologies are also highly customizable and an organization can create their own methodologies. The is methodology data is imported into SQL Server through Foundation Server and stored there for any future team projects to reference.

Build Data Another feature of Foundation Server is support for Team Build processes. Repeatable builds are an important part of the application development lifecycle. They ensure that all builds are consistent and use the same configurations. To this end, historical build information must be retained so that a build administrator and/or release engineer can track what happened in each build.

Data Warehouse All of the operational data in the Work Items database, the Build Database, and the versioning information in the Source Control Repository are used for progress and status reporting. SQL Agent Jobs are used to periodically aggregate data cubes running in SQL Server Analysis Services, thus providing reporting capabilities directly from the warehouse. You must have Analysis Services installed in order to install Foundation Server because it handles all data processing for reporting.

Report Tier Using SQL Server Reporting Services, the report tier provides reports that interact with the data warehouse, providing information regarding schedules, budgeting, volatility, code churn, etc. Many built-in reports are available, depending on which methodology you use, but custom reports are also common, providing additional custom information regarding timelines and processes.

Looking at the Team Foundation Server architecture can teach us a very important lesson. SQL Server has the capacity to do so much if we let it do its job! Sometimes we are so focused on using SQL Server as a user data repository that we forget its role as an extremely valid cog in the overall architecture. SQL Server's ability to expose functionality through service endpoints is just one feature that makes it highly interoperable.

SQL Server Licensing

Many readers will not have to deal with licensing issues, because those choices are out of your hands, but for those of you who are small business owners or consultants, we'll say a few words on SQL Server licensing options.

What Do We Know?

At this writing, we do not have the specifics on SQL Server 2008 licensing. However, we have been promised that the price will not increase from SQL 2005 licensing costs. The following guidelines are based on 2005 licensing schemes and should be reviewed prior to publication.

SQL Server licensing costs and options typically depend on which edition you use. The good news is that Microsoft has stated that it does not plan to increase licensing costs for SQL Server 2008, so your costs should be similar to those for SQL Server 2005, assuming that you are planning to use comparable editions and licensing options.

The other bit of good news on licensing is that the Express edition is still free. There is, of course, a license agreement, and you cannot take advantage of free distribution of the Express edition unless you register with Microsoft and agree to its licensing terms for distributing the Express edition with your products. The Compact edition, typically used for handheld devices and embedded solutions, also falls into this category.

The Developer edition of SQL Server is a little different in terms of licensing. This edition is feature-identical to Enterprise edition, but can only be licensed to an individual developer and used for development. No production server can use Developer edition. In earlier versions of SQL Server (those before SQL Server 2000), the Developer edition was performance- and connection-throttled, which prevented its effective use in production. This was problematic because it prevented the developer from doing reasonable stress testing, performance benchmarking, etc. Fortunately, the Developer edition is no longer throttled, and now it is an outstanding option for developing enterprise-capable applications at a reasonable cost.

All other versions are licensed under one of three different options, as indicated in Table 1.1. The obvious conclusion is that the organization should choose the option that will satisfy their needs at the lowest licensing cost. The more feature-rich the edition, the higher the license fees in each category. For example, a single-processor license for the Workgroup edition would be less expensive than a single-processor license for Standard edition.

Table 1.1 SQL Server Licensing Options

Licensing Option	Description	Best Suited For
Per Processor	Each available processor must be licensed. If a processor in a server is configured to be inaccessible to SQL Server, it does not have to be licensed. Licensing is done per socket, not per core/Logical CPU.	High client to server ratios and Internet accessible applications.
Server plus Device CAL	Requires a license for each server, plus a CAL for each device accessing SQL Server within your organization.	Shared devices such as shared workstations or kiosks.
Server plus User CAL	Requires a license for each server, plus a CAL for each user accessing SQL Server within your organization.	Limited number of users needing access to the server, especially if they connect with multiple devices.

Of course, licensing costs will depend on who you are. Microsoft has a variety of licensing programs as well as negotiated costs for specific customers such as large corporations, governments, and educational institutions. For more information regarding specific licensing costs, you can check Microsoft's website for current retail costs or contact your Microsoft Licensing Reseller for volume license information.

Installing SQL Server 2008

Now you should have a good feel for where SQL Server 2008 fits into the typical IT infrastructure. The next step is to look at the installation process. Like any good architecture, it all starts with a plan. When it comes to SQL Server installation, planning is key. You have a number of decisions to make when you perform an installation. It's easier to do it right the first time than to try to modify it later.

DECIDE NOW AND THEN FOREVER HOLD YOUR PEACE

In one organization that Mike Lee consulted with, some decisions regarding character sets and collations were made very early. The organization decided not to go with the default settings, which is not inherently a problem as long as a user has a defendable reason for doing so. These decisions were made early in the organization's existence. Over time, as the organization grew, it distributed databases to many customers and also used them internally.

When the organization later decided to examine and reevaluate its installation decisions because the configuration had become cumbersome, it was too late to change those decisions because many customers as well as internal systems had become dependent on the initial configuration. Ironically, no one could say why the initial decision was made. The morale of this story is to make sure you have reasons for every design choice you make and be sure that you think well into the future.

Defining an Installation Plan

Some of the choices that you must make are obvious. Which editions of SQL Server 2008 will you choose? What licensing schemes will you use? What kind of hardware should you purchase? These are some of the initial decisions. Some of your options are flexible: you can change your mind about them later if needed. For example, you can always add more memory or purchase additional Client Access Licenses (CALs).

Things get trickier when you need to decide what the database architectures will look like. How many databases should you have? Are there performance implications to your decisions? What about partitioning? Should you install multiple instances of SQL Server on a single server? You can best answer these and similar questions by looking at your enterprise data needs. Let's take a closer look at some of the major planning issues such as capacity planning, database consolidation and distribution, and archival requirements.

CAPACITY PLANNING

When planning capacity, the professional DBA must consider the projected requirements of all the major resources that SQL Server will use. More specifically, SQL Server requires resources from four major categories. Think of them as SQL Server's four basic food groups. A server must have adequate resources from each group to remain healthy.

- Disk and I/O Subsystem
- CPU
- Memory
- Network

Just as the human body needs to balance grains and proteins in its diet, SQL Server needs to have an adequate balance of each of these resources to keep operating at its maximum potential. Any one of these resources out of balance can cause a resource starvation problem in another area. Capacity planning will be covered in more detail in Chapter 3, "Managing Databases, Files, and Filegroups," so for now we will be content with a general overview. What are the important things to consider when preparing for a SQL Server installation?

Disk Resources

Databases require disk space. Although a well-normalized database will have lower disk requirements than a denormalized one, there is still a significant disk need for any production system. Data warehousing might require significant disk resources, not only to handle the volume of gathered data, but also to perform the multidimensional data modeling that is typical to data warehousing. It is also essential that the I/O subsystem be able to keep up with requests to read and write to disk, or else the overall performance of the system will suffer.

Disk storage requirements may depend on a number of factors. You have to consider the current data that the business needs to function, as well as archival requirements caused by regulatory concerns, data warehousing/mining strategies, encryption requirements, and data growth estimates. Some of these issues, specifically warehousing and archiving, may necessitate additional servers—not just more disk space.

Data growth is one of the more complex issues. You not only need to consider how much data you have now, but also how much you will have in the future. There are three basic growth models, each of which has numerous variations. You can probably evaluate your growth patterns according to one or more of these models.

Linear Growth This is the simplest growth model. It assumes that the database will grow at a constant rate over time. For example, assume 400MB database will grow at a constant rate of 50MB per month. One year from now, therefore, the database will be 1GB in size. Use the following formula to calculate linear growth:

$$FDisk = CDisk + (GAmt \times NPer)$$

In this formula, use the following:

FDisk = future disk space required

CDisk = current disk space used

GAmt = amount of disk growth per period

NPer = number of periods

Compound Growth This model assumes that the growth rate is based on a percentage growth rate rather than a fixed growth amount. As the size of the database increases, so will its future growth amount because each increment is a percentage of the current database size. A database of 100MB that grows 10 percent per year would be 110MB at the end of the first year, but 121MB at the end of the second year. Use the following formula to calculate compound growth:

$$FDisk = CDisk \times (1 + Rate)^{N}$$

In this formula, use the following:

FDisk = future disk space required

CDisk = current disk space used

Rate = percentage growth rate per period

N = number of periods

Compound Growth with Compounding Growth Rate This model assumes that as the database grows, so does the growth rate. For example, suppose that we expect the database to grow by 10 percent this year, but we expect that the growth rate itself will also increase every year by 20 percent. That means that next year, the growth rate will be 12 percent. The first year, therefore, a 100MB database will grow by 10MB. In this case, the 10MB is called the *initial increment* and the 20 percent is called the *incremental rate of growth*. Use the following formula to calculate this model:

$$FDisk = CDisk + (Init \times (1 - IncRate)^{(N+1)} + (1 - IncRate)$$

In this formula, use the following:

FDisk = future disk space required

CDisk = current disk space used

N = number of periods

Init = initial increment of growth

IncRate = incremental rate of growth per period

Of course, you must consider the I/O subsystem as well. Remember that at some point, any data that you might want to work with in your application must move from disk to memory. The speed at which this takes place will have a direct effect on the performance of your application. The specifics of monitoring and tuning resources will be covered in Chapter 10, "Monitoring SQL Server Activity," and Chapter 16, "Using Resource Governor and Policy-Based Management," respectively.

Processor Resources

When planning processor architecture, you have two primary concerns: what type of processor and how many. The decisions that you make can determine not only the amount of raw power in terms of calculations per second, but also may determine the amount of addressable memory that is available to SQL Server. Because data normally must be in current memory to be used in SQL Server, the size of the data cache may be critical for application performance.

First, for determining the number of processors, if you have an existing database that you can use to benchmark, you're way ahead in the game. Specifically, you want to look at the number of users and how much data they will be using. Most queries are not really CPU bound, especially those that focus strictly on read and write activity, but if you have a lot of procedural logic, number crunching, aggregations, or a large number of users, you will need more CPU resources.

When considering processors, you must also think about licensing costs. Remember that if you use a per-processor licensing scheme, each processor will have to be licensed. The processor affinity settings in SQL Server allow you to specify which processors SQL Server is allowed to use. When allocating processor resources to SQL Server, you must remember to allow sufficient resources for other services that might be running on the same server. You don't want to allow a

situation where SQL Server could be starving other services. Of course, this brings up the question of whether the SQL Servers are dedicated machines or share responsibilities for other services. This in turn will depend on projected server activity levels and the level of resource consumption. As you make these decisions, remember that processor licensing is per socket, not per core.

The Real Cost of Affinity

Remember that the processor affinity settings in SQL Server are very tightly integrated with the licensing system. Make sure that you don't use processor affinity settings that might violate your license agreements.

When deciding on processor architecture, you must also consider your memory requirements. Chapter 2, "Understanding SQL Server Architecture," will cover the specifics of memory configuration and how memory is used, but for now, remember that the primary difference between 32-bit and 64-bit processors is the amount of memory that is addressable by the processor. The 32-bit world limits the server to 4GB addressable memory, only 2GB of which are available for SQL Server unless you use some special switches and creative manipulation. 64-bit processors significantly increase that limit. How much of an increase depends on the processor, but theoretically it could extend to 16 exabytes. This is a theoretical limit only, and all current 64-bit architectures implement limits that are a tiny fraction of this level, averaging around 32GB of addressable memory.

The real story then is not that 64-bit processors are faster. In fact, dealing with a 64-bit address can actually cause a 64-bit processor to run slower than a comparable 32-bit processor. The real issues are the amount of memory that can be addressed and the convenience with which that memory can be addressed without special configuration on the part of the administrator.

Memory Resources

In order for data to be of any use in your applications, in most cases it needs to be brought from disk into memory. The data in memory is SQL Server's real-time view of the database. Data on disk is only as current as the last flush from cache to disk. Because data must be in memory to be useful, it takes a lot of memory to drive your application, and the more data you have to work with, the more memory you need.

Just because you have large databases does not mean that you need a lot of memory. However, you should try to ensure that there is enough memory to store the commonly used data and that there is enough room to bring additional data into memory without having to constantly swap between cache and disk. Don't forget that you also need memory for other resources such as user connections, open objects, stored procedures, and locks.

One Hand Washes the Other

It should be obvious at this point how interdependent all of these resources are. For example, the CPU type will determine your memory architecture. The amount of memory can determine how much work the I/O subsystem will have to do. The more that you tinker with SQL Server, the more you will realize how closely related all these resources really are.

Network Resources

Because the whole idea behind SQL Server, especially from the enterprise perspective, is to make data available to a broad audience of users, it is important that the data be able to get to its ultimate destination as efficiently as possible.

A number of issues are at play. First of all, you need to make sure that your network is capable of handling the volume of data that you will need to push around. This means that you need to pay attention to things like network cards, switches, and cabling. You can only push so much data through a 10-megabit pipe.

The other thing that you need to consider is how your applications are written and whether they are using network resources appropriately. For example, are you properly restricting your result sets so that you are not returning any more data to the client than they really need? You can limit some bandwidth by doing little things like turning `NOCOUNT` on in all of your stored procedures.

Doctor, It Hurts When I Do This

Sometimes the easiest way to solve a performance problem is to say, "Don't do that!" For example, I had a client that was having performance problems with an application that returned a large amount of sorted data to the client. The initial concern was that it was a network problem based on the volume of data that was being returned. It became clear after testing, however, that the problem was related to the amount of extra time that it took the server to do the sorting. Because many clients were requesting sorted data, the throughput was dragging.

Because sorting does not reduce the amount of data that goes across the wire, this task could be relocated without increasing network consumption. We found that in this situation, the sorting could be offloaded to the client because there was excess capacity on the client that was not being used. Sometimes you have to be willing to accept that the server is not the best place to perform a task.

Data Consolidation and Distribution

Depending on how much data you have and how your enterprise's users need to access that data, there may be varying ways to consolidate or distribute data and server resources. There can be significant advantages to each depending on your needs. Let's begin with a discussion of server consolidation.

As licensing costs and hardware costs can be high, it is very important to make sure that you get the most "bang for the buck." Are you really using all of your resources to their fullest potential? It doesn't make sense to spend extra money on servers and licenses if you are not fully utilizing what you have now.

In addition to these issues, you might also benefit from a simplification of administrative processes, auditing requirements, and security management. Fewer boxes to baby-sit means fewer things that can break.

Of course, the downside is that when things do break, it can have a much more serious impact on the organization. You might be introducing a single point of failure by performing a server consolidation. You also have to be careful not to overload your server resources. Consolidating to

the point that there is no room for growth means that even small changes in the way that server resources are used can have significant effects on the overall performance of your servers.

There are many ways to consolidate. Here are a few.

Merge Databases One approach is to create a smaller number of larger databases. Fewer databases means that there are fewer administrative tasks, but these tasks may be more cumbersome and time-consuming. There may also be problems with increased transaction activity and loss of configuration granularity.

Adding More Databases to a SQL Server Instance Although this can solve the problems inherent in managing very large databases, it still requires that all databases running under the given instance share all server-level configurations and address space. However, for administrative reasons, it may be desirable to have many databases running in a single instance. The benefit is that you only have to configure and tune that single instance.

Installing Multiple Instances on a Single Server The primary benefit of having multiple instances is that they can be configured and tuned individually. If databases have different server configuration needs, they do not need to run on separate servers, but instead can run in separate instances. All instances still will share the same server resources, however, and any platform tuning will affect all instances.

Using Server Virtualization Using a tool such as VMware or Microsoft Virtual Server allows multiple virtual or emulated "servers" to coexist on the same hardware. Each one can be configured separately, although they all will still share the same actual hardware resources. This approach can be very effective when you have excess hardware capacity, but requirements that specify individual OS configuration and security.

No matter which consolidation approach you choose, remember that the ultimate goal is to make the best use of all of your hardware resources and licenses. If your consolidation scheme results in unwanted performance degradations or security compliance problems, you should rethink that strategy. Also, always remember to reserve enough capacity for reasonable growth, or you will quickly be required to start distributing some of your load.

Generally, the advantages of consolidation are also the disadvantages of distribution. The reverse is also true. While consolidation can lower licensing and hardware costs, distribution increases them. Conversely, while distribution can increase performance by providing additional resources to your applications, consolidation can result in reduced performance because too few resources are spread across too many databases or instances.

Archival Requirements

Another issue to consider when planning an enterprise installation strategy is archival requirements. Although your organization may have painstakingly collected mounds of data over the years, not every query that you execute has to wade through those mounds of data every single time.

Good indexing strategies can help with the performance problems that arise when you query from very large data stores, but other problems can emerge such as the maintenance costs of those indexes, the need to back up large amounts of non-changing data just to capture a small amount of live data, and so on. If you can separate those large volumes of data into other repositories, whether different tables or entire archival databases, your queries will have to trudge through much less data to complete their tasks.

But What about the Warehouse?

Although this would seem a good time to bring up the subject of data warehousing, the subject is far too vast for inclusion at this point. Appendix C, "Analysis Services Overview," provides a very brief overview of SQL Server Analysis Services, which includes a general review of data warehousing concepts. However, the best place to start reading would be the work of Ralph Kimball and Bill Inmon, the true pioneers in this area. We specifically recommend the books in the Data Warehouse Toolkit series by Kimball and *Building the Data Warehouse* by Inmon, all published by Wiley.

From those readings, you can decide which approach makes the most sense in your organization. Also remember that most organizations that do serious data warehousing in the real world are using a combination of theories based on the practical needs of the moment.

The first real issue when planning archival storage is to decide how to segment your data. Often, data is segmented based on date. Older data is archived while newer data is retained in the operational data store. There may be other ways in which your data should be segmented, however. Identifying the factors that differentiate current from archival data is the most important aspect of defining a successful archival strategy.

Once you have decided how to segment the data, you need to decide the best archival storage approach. In some cases, you might simply archive data into separate tables in the same database. Other cases may require you to completely separate archived data on separate servers. There are many different techniques for supporting these archives including table partitioning, partitioned views, and rowset functions. For now, the important thing is to consider what the resource load will be, based on the amount of needed archival data and what kind of server installation strategy you will need to support it.

SQL Server 2008 Installation Requirements

The official Microsoft SQL Server installation requirements are, to be polite, barely adequate. As you evaluate the true requirements for SQL Server, remember that SQL Server is a resource hog. It loves memory, can use a lot of disk space, and often requires a significant network pipe. That said, the requirements posted on Microsoft's website are barely enough to get the service running, assuming that there are no other primary services running on the same machine.

Table 1.2 lists the standard Microsoft-recommended requirements in abbreviated form. To get the complete listing, please consult the books online.

Table 1.2 Microsoft SQL Server Installation Requirements

Component	Requirement
Processor	64- or 32-bit (version dependent) processor at 1GHz or higher
.NET Framework	.NET 3.5 Framework SP1 (installed by the SQL Server installer)
Operating System	Windows Vista (any version)
Windows Server 2008	Windows Server 2003 SP2 Windows XP SP 2

TABLE 1.2 Microsoft SQL Server Installation Requirements *(CONTINUED)*

Visual Studio 2008	If VS 2008 is installed on the same machine as the server or client tools, it must be updated to SP1. It is not required to have VS 2008 installed in advance.
Other Software	Microsoft Internet Explorer 6 SP1 or higher Microsoft Windows Installer 3.1 or higher MDAC 2.8 SP1 or higher
Memory	1 GB or more
Hard Disk	280 MB - Complete Data Engine Core 90 MB - Complete Analysis Services 120 MB - Complete Reporting Services 120 MB - Complete Integration Services 850 MB - Complete Client Components 240 MB - Complete Books Online

Also be aware that many components will be installed by the SQL Server installer. In addition to the .NET Framework, there will be installations of MSXML support, SQL Native Client installation, and other components.

During the installation process you will be asked to identify how the SQL Server services are going to log in to Windows. You should consider this requirement before you begin installation so you can either identify or create the necessary security accounts *before* the installation process begins. This should be part of your security planning. The choice that you make can have a significant impact on the ability of the installed server to effectively communicate with other SQL Servers and resources on the local machine and in the greater enterprise environment.

The general recommendation is that you should create local or domain user accounts with the permissions necessary to perform the tasks required. Do not use the Network Service account for the Data Engine or SQL Agent services because this approach does not follow the "least privilege" principle, which states that you should only grant an account the minimum permissions necessary to perform the required task. If your SQL Server must communicate with other service to perform remote procedure calls, replication, using remote data sources, or similar tasks, you should use a domain user account rather than a local user account. If your server will access local resources only, the "local service" account may be adequate, because it has the same permissions as members of the local Users group.

The SQL Server Installation Process

The SQL Server installation process is organized into three primary phases:

1. Compatibility Check
2. Data Gathering
3. Software Installation

Compatibility Check

The first phase, Compatibility Check, allows the installer to verify that the hardware/software configuration of the target server meets all minimum requirements for installation. If any of these requirements are not met, the installation will be terminated and the problem must be rectified. It is best to review all of the requirements in advance and make sure that your server is properly configured before you begin. It will definitely make the installation process go more smoothly.

The installation process is wizard-driven and should not be problematic if you have installed all of the required software in advance. On the main installation screen, select Installation from the menu on the left. This dialog is pictured in Figure 1.1.

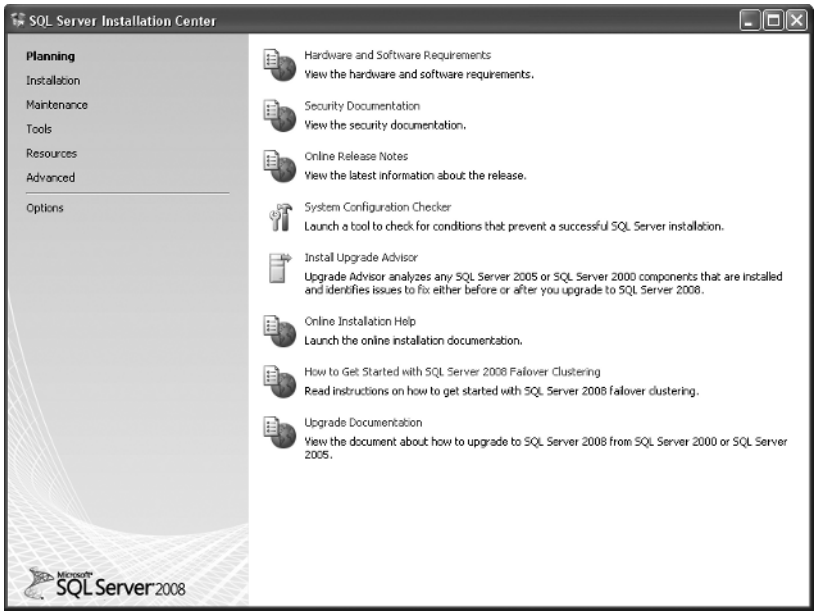


Figure 1.1 The SQL Server 2008 Installation Center

Once you launch the installer, you will be asked to accept the license agreement and install the setup support files. These steps must be completed before the formal installation process can begin.

The next screen in the installation process is the main installation control panel, pictured in Figure 1.2. From here you can perform a new installation, upgrade older servers to SQL Server 2008, or modify a SQL Server cluster.

Click the first option to perform a new standalone installation. In a moment, a new popup window will appear to begin the formal installation. It will automatically perform a compatibility check and report the results in the next screen as shown in Figure 1.3. If your system does not pass the minimum compliance check, you will not be able to continue any further until you resolve the issues. There is a button on the screen labeled Show Details. Click this button to get more information regarding the results of the configuration check. Figure 1.3 shows the detailed information. When you are ready to move on, click the OK button.

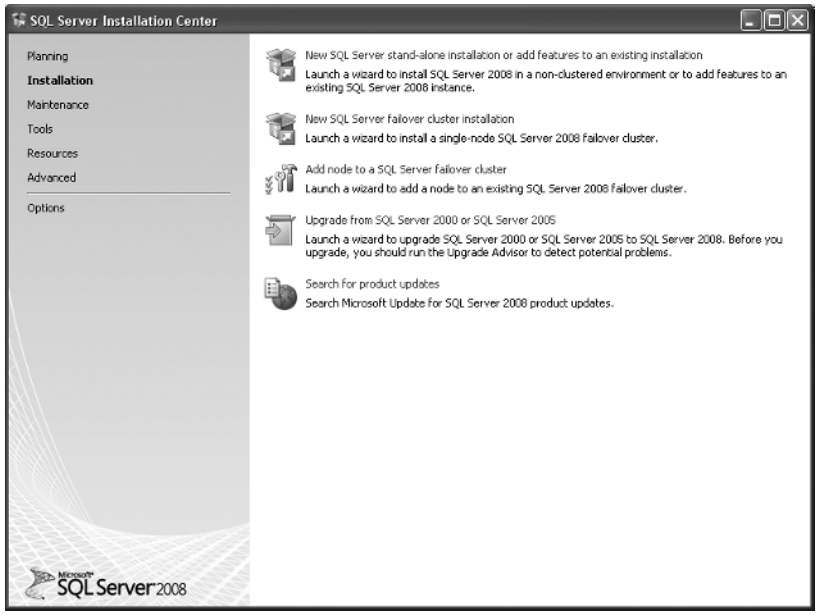


FIGURE 1.2
The SQL Server Installation panel

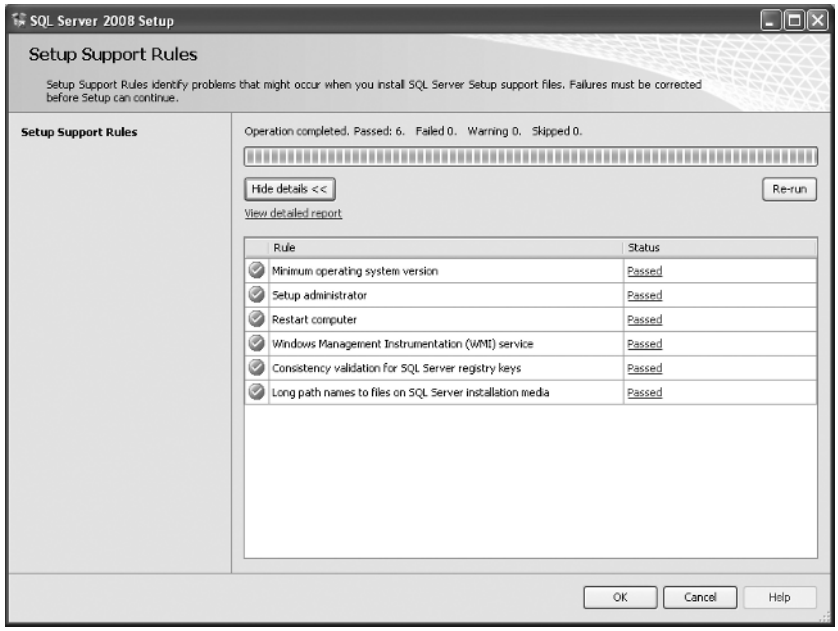
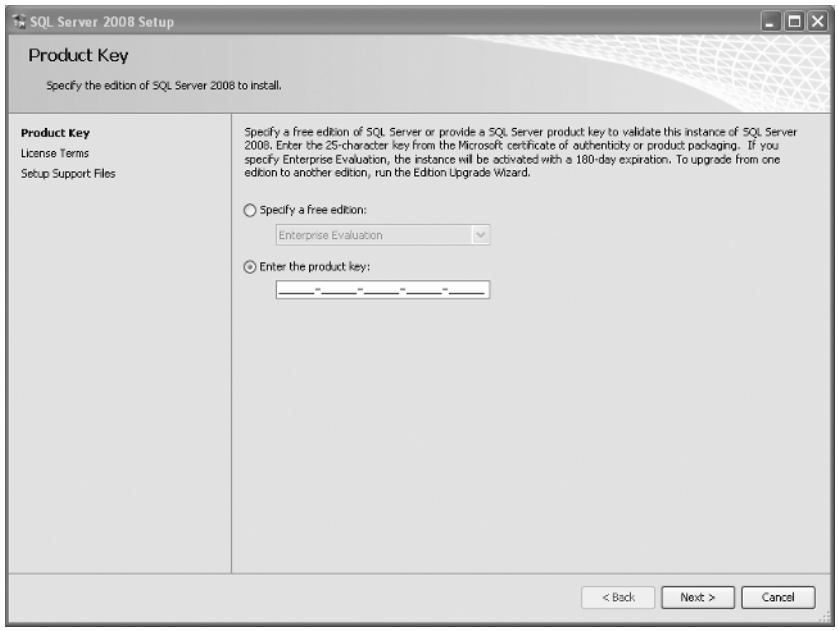


FIGURE 1.3
Viewing the compatibility results

Data Gathering

At this point we enter the data-gathering phase of the installation. This begins with a licensing screen as pictured in Figure 1.4. You will see that you can either enter a key for a licensed version of SQL Server or select a free version such as an evaluation version. Make your selection and enter the key, if necessary. Then click Next to continue.

Figure 1.4
Validating the edition



The next screen is the license terms agreement. It is not pictured here for brevity. Simply accept the license agreement and click Next to continue. The next screen will provide the installation of the setup support files. You must perform this small installation before you can continue. Again, because this is a simple dialog, it is not illustrated. Just click the Install button to advance. When complete, the installer will provide a report like the one pictured in Figure 1.5. Check for errors, and make sure that warnings, like the one in this dialog, are verified before you continue. Click Next to advance.

The next thing that the installer needs to know in this phase is which SQL Server 2008 services you want to install. SQL Server uses a unified installer, which is much more convenient than having separate installers for each service. In Figure 1.6, all options have been selected, but you can choose any combination of installed features that you like depending on your enterprise installation plan. Do not install components that you do not need because this will add additional administration requirements, could result in unnecessary resource consumption, and may be a security risk. Click Next to advance.

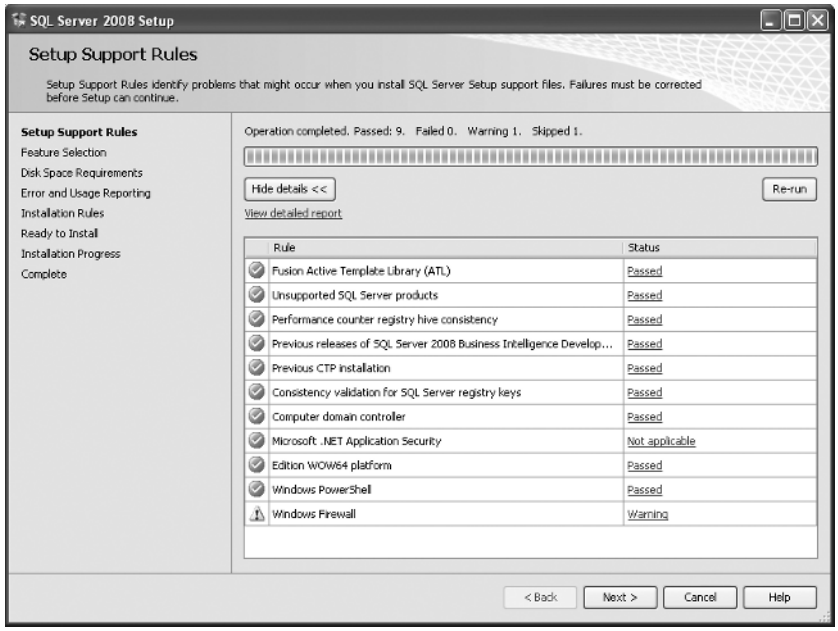


FIGURE 1.5
The Setup Files Installation report

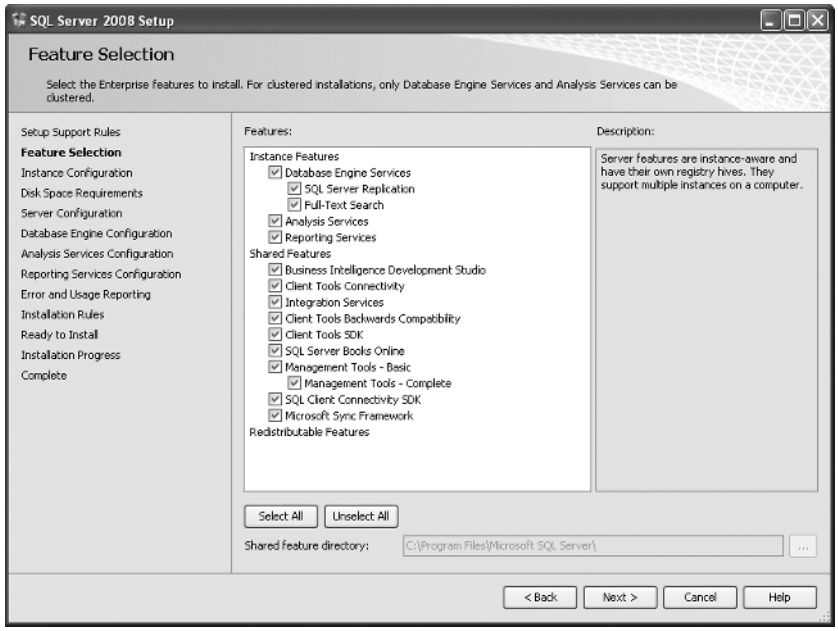
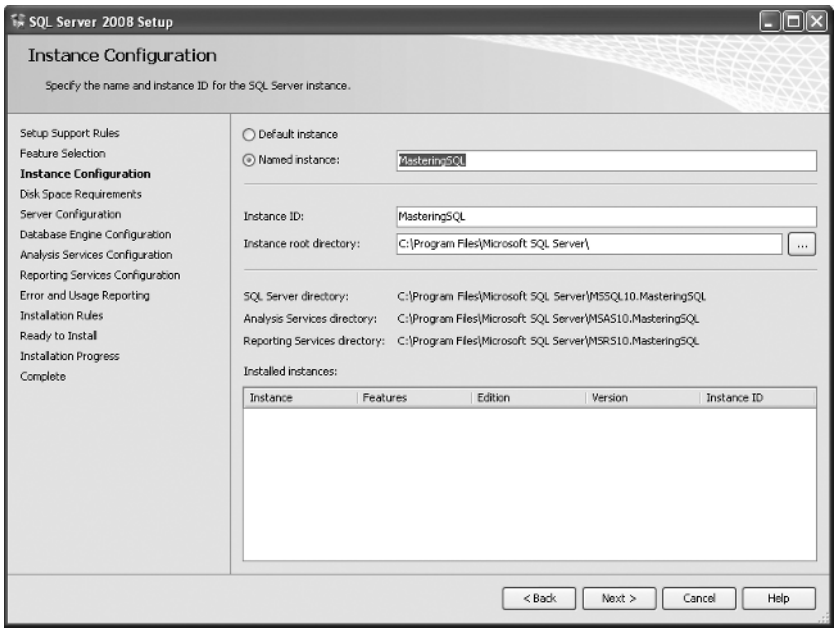


FIGURE 1.6
The Feature Selection dialog

The next page in the wizard, pictured in Figure 1.7, allows you to configure the instance. Is this a named instance or the default instance? Typically, the default instance will resolve to the NetBIOS name of the computer, which is convenient, but you can have only one default instance. If you are not sure what other instances of SQL Server you have installed, this dialog will list them at the bottom. Be sure to configure the correct location for file installation and the desired instance name before you continue. Select either the default instance or provide an instance name and click Next to advance.

FIGURE 1.7 Configuring the instance



The next screen summarizes the disk space requirements. Notice that in this example, all services have been installed and it requires over 2.5GB of disk space. Figure 1.8 provides this illustration. If you do not have sufficient disk space available, you will receive an error at this point. Click Next to advance.

The next dialog, Server Configuration, allows you to configure all of the services in one location. Notice how this dialog, pictured in Figure 1.9, gives you the option of configuring service login accounts and passwords, as well as deciding the startup mode for each service. Now is the time for you to provide the login information for the accounts that you created in advance for the services. If you want, you can simplify the configuration by using a single account for all services and applying this account with one click.

Choose the account wisely. The local system account can work fine if the SQL Server does not have to communicate with other services on the network, otherwise you may want to create a domain account with appropriate permissions. This will give you more control over the service-level security architecture.

Notice also that the Collation tab allows for configuration of default collations for both the Database Engine and Analysis Services. Select your collations carefully and only depart from the defaults if there is a verifiable reason to do so. Although you can create artifacts in a database that

depart from the default collations, the settings that you choose here will define a set of default rules that will be assumed to apply unless you create an exception.

If there is an error condition in any of the configurations, the tab will appear with an error symbol. At the bottom of the screen, below the grid that defines the SQL Browser and Full Text Daemon Launcher, an error bar will appear. You must resolve these errors before continuing. Click Next to continue.

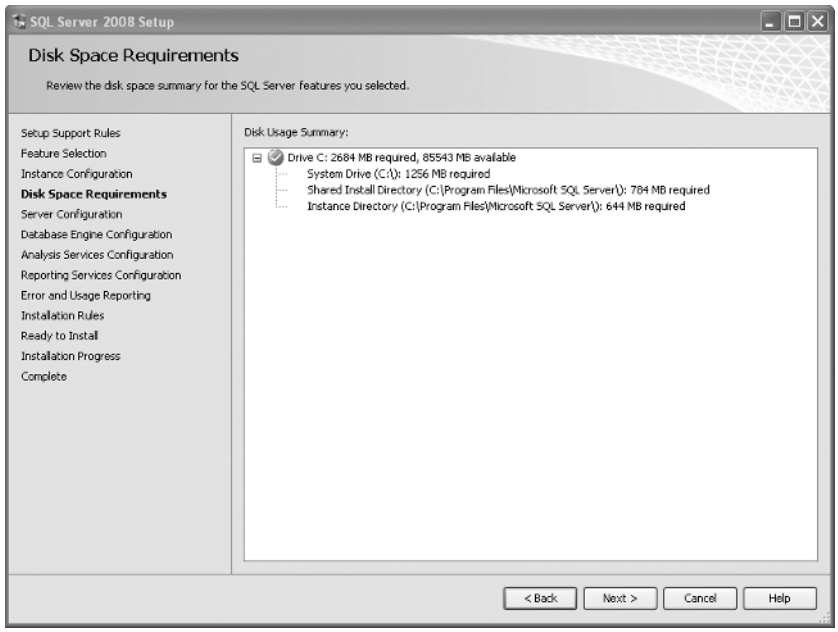


FIGURE 1.8
Disk space summary

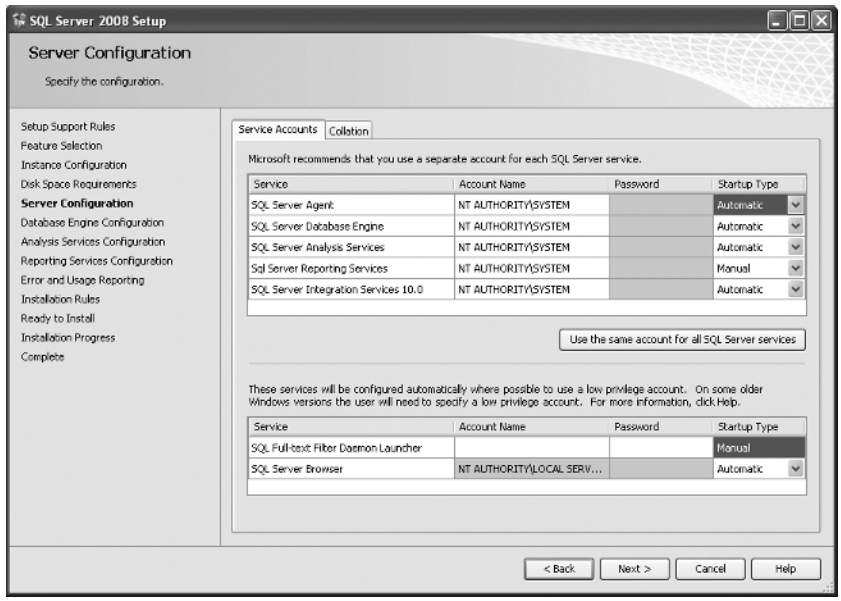


FIGURE 1.9
The Server Configuration dialog

Now the installer will begin with the configuration of each individual service. First, you configure the Database Engine. This includes the security modes configuration as well as the file location and FILESTREAM configuration. On the first tab, specify the security mode. Windows Authentication mode uses the Windows identity of the logged-in user to authenticate that user to SQL Server; the Mixed mode option *also* allows SQL Server to authenticate users with its own internal authentication mechanism and login lists. We are using Windows Authentication mode. You can add yourself as an administrator by clicking the Add Current User button. Otherwise, the add button allows you to browse your accounts directory and add administrative users and groups to the server. You must add at least one administrative user before you will be allowed to continue. Figure 1.10 illustrates the Database Engine Configuration dialog.

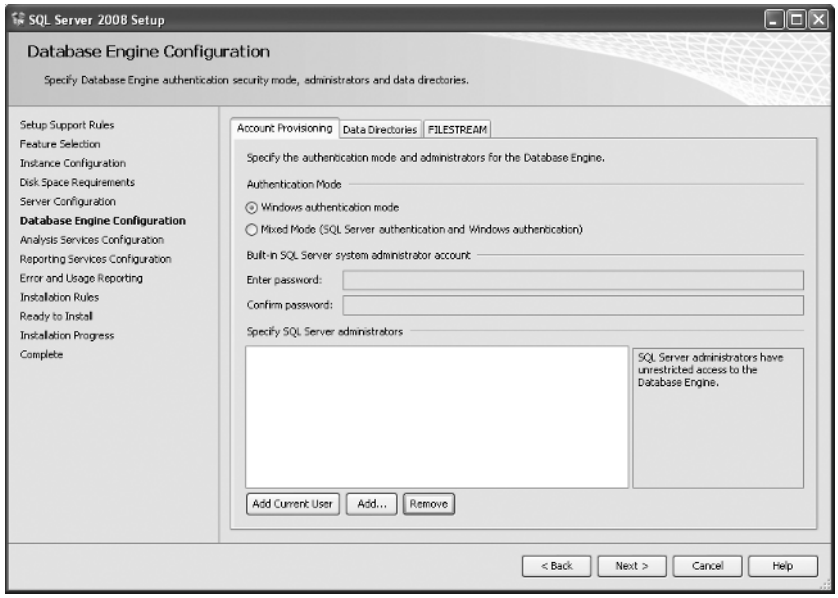


FIGURE 1.10
The Database Engine Authentication Configuration dialog

If you are unsure of which mode you need, you may want to skip ahead to the discussion in Chapter 8, "Managing User Security," and read up on the difference. This is a configuration that you can easily change after installation, however, so you are not stuck forever with the choice that you make here.

If you do choose Mixed mode, you will have to define a password for the system administrative account, which is `sa`. Otherwise, you must add appropriate Windows users or groups as administrators in the bottom part of the dialog so that an administrator will be assigned. If you try to advance beyond this dialog without providing some sort of administrative access, you will get an error message that will prevent you from continuing.

Also, don't forget to look at the other tabs in the dialog, labeled Data Directories and FILESTREAM. The Data Directories tab allows you to define the default locations for all data paths in SQL Server. Although you can override most of these when working with your data, it is a good idea to plan your default locations based on your server resources and configurations.

The FILESTREAM tab provides for the initial configuration of the FILESTREAM storage option in SQL Server. This feature allows you to define a share on the server that will store files that the data engine can stream to the user. This is an alternative to storing the file data directly in the database.

The next dialog, Analysis Services Configuration, is very similar to Database Engine Configuration and works in the same manner. This dialog allows you to specify the security and file location setting for the Analysis Server. This screen is so similar to the previous one that it is not pictured here.

The Reporting Service Configuration dialog is used to configure the deployment mode for SQL Server Reporting Services. You have a choice between Native and SharePoint modes. You can also install an unconfgured server. Make your selection in this dialog, pictured in Figure 1.11 and click Next to advance.

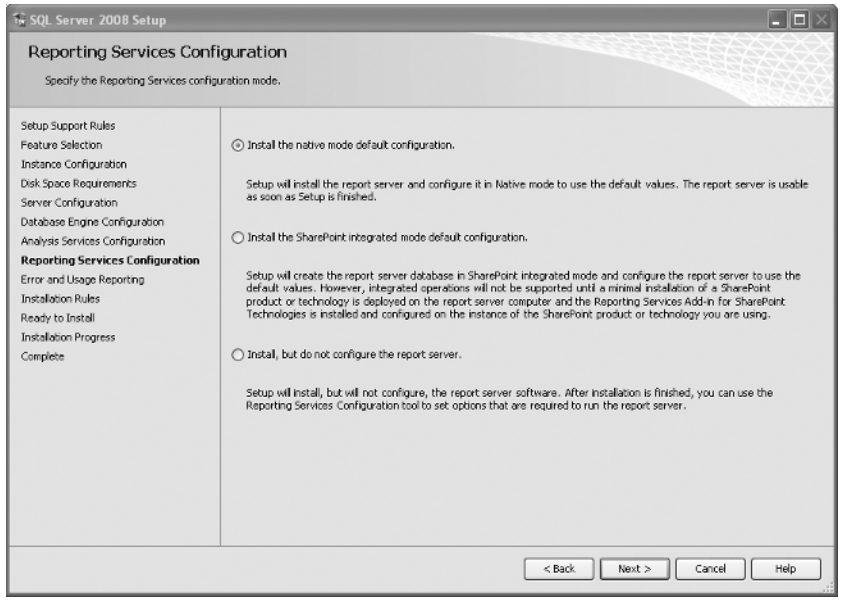


FIGURE 1.11
Configuring the reporting server

Native Mode Reporting Server Native mode allows the reporting server to act as a standalone application server, using IIS and ASP.NET features to handle all configuration, report access, security access, etc. This is the default option and does not require any additional software installed beyond IIS and the .NET Framework.

SharePoint Mode Reporting Server In this mode, the report server is not standalone, but rather is part of a SharePoint application deployment. Reports can be stored in SharePoint libraries and accessed by users through the SharePoint interface. This affects the ability of users to access reports through a URL, because all access must be qualified within the SharePoint hierarchy. Also, appropriate versions of Windows SharePoint Services or Microsoft Office SharePoint Server must be installed to handle the Report Server deployment before the report server can be accessed.

The next dialog, Error and Usage Reporting, gives you the option of sending experience information to Microsoft. This is very simple and is not illustrated here. After this, you get one more report dialog, validating your choices and preparing the installation. As pictured in Figure 1.12, there should be no errors on this report or you will not be able to continue.

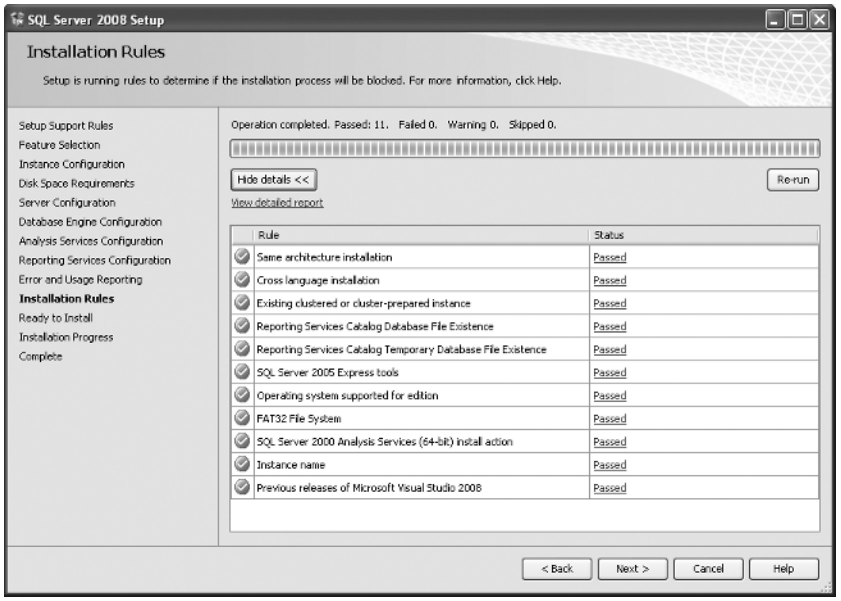


FIGURE 1.12
Final pre-installation report

The last data-gathering dialog (not pictured) presents you with a summary of your choices for your review. Make sure that the configurations listed match your intended installation before you continue. When you are ready, click the Install button to begin the actual installation.

SOFTWARE INSTALLATION

Once you have verified your installation parameters and are ready to begin with the installation, click the Install button to begin the installation process. Installation time depends on the services that you have selected and the configuration of those services.

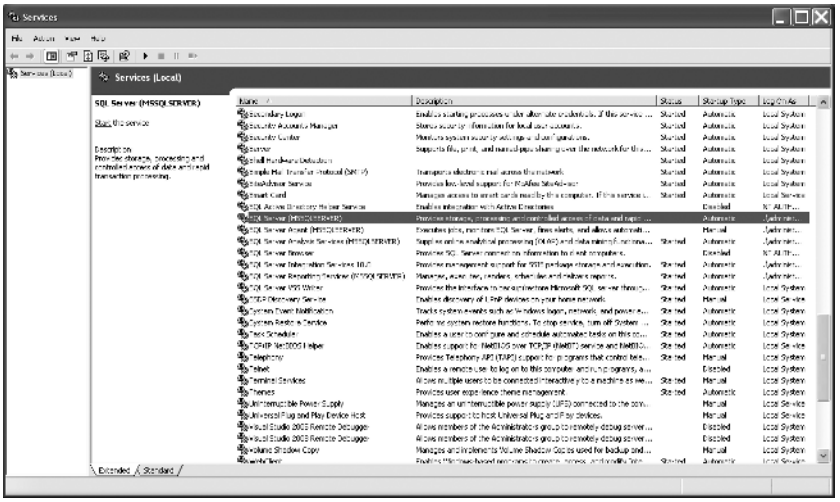
Once the process is complete, all of the installation options should show Success. Clicking the Next button will take you to the final dialog, which will contain a link to the installation log. Make a note of the location of the installation log files. If there are any problems with the installation, you may want to review these logs to troubleshoot your problems.

Verifying the Installation

If you did not get any errors when you ran the installation, you probably will not have any problems, but it is always a good idea to run some quick checks to make sure that everything happened the way you expected. Here is a list of some of the actions users typically take to verify an installation.

Check for the Installed Services Using the tool of your choice, ensure that the services have been correctly installed, configured, and started. Figure 1.13 illustrates the Windows Services applet. Notice that each of the services is listed, along with its login account and current state. Ensure that all services will start as configured and alter any startup options as necessary. In Figure 1.14, the SQL Server service is installed but not started. You must start a service before you can test its installation.

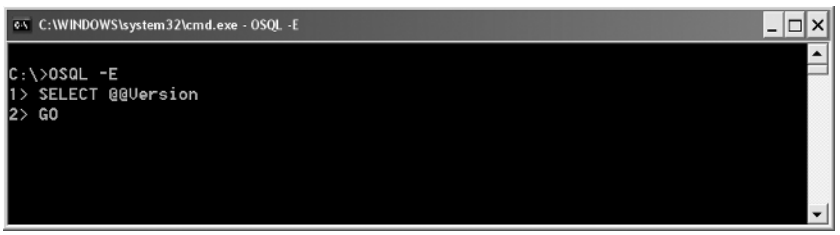
FIGURE 1.13
Reviewing installed Windows Services



Verify Existence of Installation Folders Make sure that the configured default data folders and temp folders are present and that they contain the files that you expect, such as `.mdf` and `.ldf` files for the system databases. The location of these files will depend on the Data Directory configuration that you specified when you installed SQL Server.

Connect to the Server We will cover the SQL Server clients in more detail in a moment, but one of the verifications that you will want to do is to attempt to connect to SQL Server. The simplest way to do this is to open a console window and use the command **`OSQL -E`** to log into the server. This assumes that you are connecting to a default instance through Windows Authentication mode. Once you authenticate, you will see a line prompt on the screen labeled 1>. Type a simple command such as the one pictured in Figure 1.14. Press the Enter key after typing **`GO`** to execute the batch.

FIGURE 1.14
Connecting to SQL Server with OSQL



If all is well, the screen should return version information. If you are unable to connect or if the command returns an error, verify that your services are running, that your account has permissions to access the server, and review the installation logs as necessary for problems. Type QUIT and hit enter to end the OSQL Session. Then type **EXIT** and press Enter to close the Command Prompt window.

Installing Sample Databases

If, after installing the server, you want some sample databases to play with, you can download them from Codeplex.com. Go to the SQL Server section, where you can download installers for the various databases. Examples in this book will use the AdventureWorks sample databases, specifically the OLTP and DW databases, so you may want to have those installed for your reference.

The SQL Server Tool Set

When you installed SQL Server, you had an option to install the client utilities as well. This client tool set consists of numerous configuration clients, command-line utilities, and development tools. In addition, there is also a thriving third-party market for SQL Server client tools. You can accomplish most of your day-to-day tasks with the out-of-the-box SQL Server utilities, but some of the third-party utilities are very convenient and make up for deficiencies in the Microsoft toolkit.

The Microsoft Client Tools

Numerous utilities are installed with SQL Server. You will probably want to review the SQL Server Books Online to get a complete description of each of these and their capabilities, but the primary list is provided in the following text.

The SQL Server Management Studio

This is the primary SQL Server client. Introduced in SQL Server 2005, this utility replaces the Enterprise Manager and Query Analyzer tools found in SQL Server 2000 and earlier versions. Using this one tool, you should be able to handle the majority of administrative and development tasks.

You will find the SQL Server Management Studio in the SQL Server program group in the Start menu. When you launch it, you will be prompted to log in. This is not required, but if you do log in, the tool will populate its Object Browser with catalog information and you will be able to perform tasks within SQL Server Management Studio without further authentication. How you authenticate will depend on the security setting that you chose when you installed the server. After authenticating, you will see a screen like the one pictured in Figure 1.15. If you do not see the details pane, press the F7 key on your keyboard.

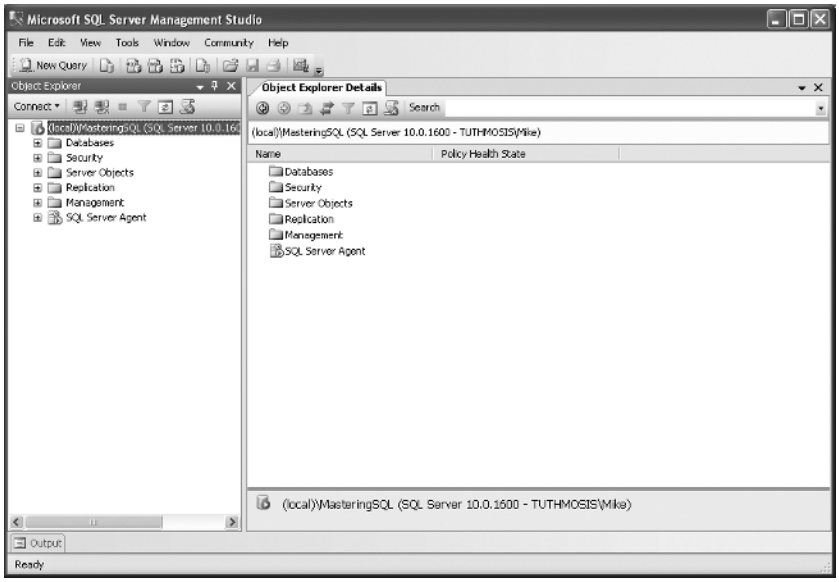


FIGURE 1.15
The SQL Server Management Studio

Notice the tree view on the left. This is the Object Browser. Use this part of the screen to look at and configure your resources. The dialog makes good use of your secondary mouse button, so most of the options you want should be no further than a right click away. Also notice the Connect button in the upper-left portion of the browser screen. Use this to connect to other servers or other services on the current server. For example, you can use it to connect to multiple Data Engines on different servers or instances, as well as to an Analysis Server and a Data Engine on the same instance.

To run a query, click the New Query button. A query window will open based on the current connection. This window works very much like the old Query Analyzer utility. Enter your query and click the Execute button or press the F5 key to execute the query. You will want to play around a little bit with your options in the client as you will use it often while working through this book as well as your daily work as a DBA or developer. We will address the specifics of what you can do with this client as the book progresses.

As you start working with the query editors in the SQL Server Management Studio, you will probably stumble across one of the most-requested new features for the SQL Server development clients, called Intellisense. As you type, Intellisense will prompt you with lists of valid syntax or artifacts that you can use in your queries. It will also inform you of syntax errors and other problems as you type by putting red underlines in your code. Visual Basic and .NET developers have had these tools forever, and they have finally made their way into SQL Server, to the delight of the SQL Server development community.

In With the New, But Not Out With the Old

One of the things that we like about the progression of SQL Server versions is that Microsoft still supports many of the old habits that you may have developed in your early days as a SQL Server developer. For example, back in the old days of SQL Server, when we used to use a tool called ISQL for query development, many of us got in the habit of using Alt+X to execute queries. That was replaced with Ctrl+E, which was ultimately replaced with F5. However, all of the old keystrokes are still functional. Author Mike Lee says, "Be it good or bad, I will probably be an Alt+X man until they play the Windows Shutdown .wav at my funeral someday."

The SQL Server Business Intelligence Development Studio

This tool is really just a set of Microsoft Visual Studio addins. It can cause confusion if you are not aware of what is happening. When you install this client, the Microsoft Visual Studio Development Environment is installed on your computer along with the specific development editors for the SQL Server Business Intelligence artifacts such as reporting, data cubes, and Integration Services packages. Remember that all of these BI artifacts are actually .NET executables, so the development environment for them is wrapped into the .NET container.

If you already have the corresponding version of Visual Studio .NET installed on your computer when you perform the SQL Server client installation, it will not need to install the development environment, but only the environments for BI development. Therefore, to see what you really have installed in Visual Studio, start the tool and select File ➢ New Project from the menu. The list of project templates will indicate what you have installed and available for use. Figure 1.16 illustrates a New Project dialog with just the BI environments available.

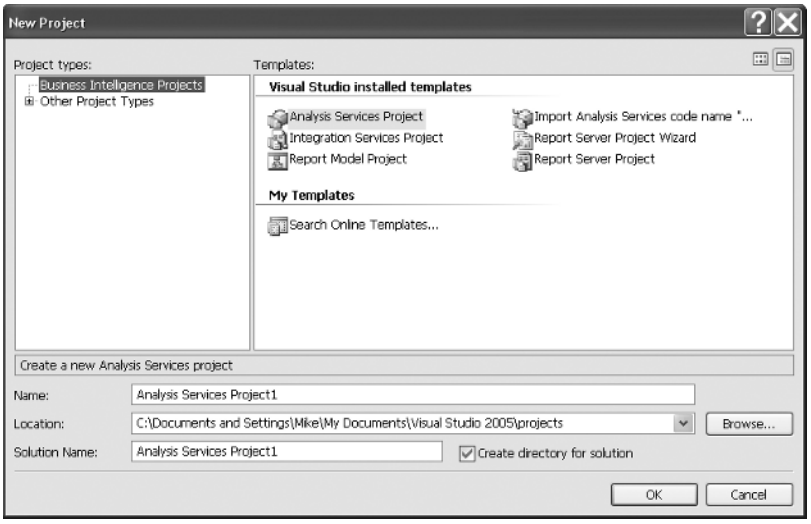


Figure 1.16 The New Project dialog in Visual Studio

The SQL Server Configuration Manager

Another very convenient tool is the SQL Server Configuration Manager illustrated in Figure 1.17. This utility provides a central configuration tool for all SQL Server services. In addition to the actual service configuration, you can also configure .NET libraries, client connectivity, and client aliases in one location. Prior to SQL Server 2005, this was done in multiple utilities. This welcome enhancement is preserved in SQL Server 2008.

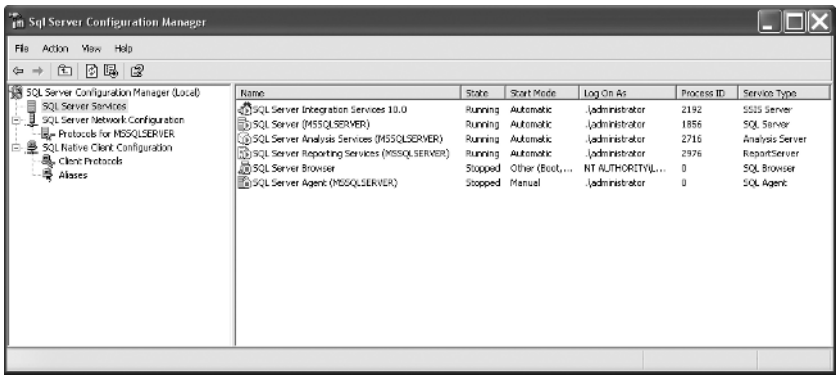


Figure 1.17
The SQL Server Configuration Manager

The interesting thing about the SQL Server Configuration Manger is that it is a WMI client. As you browse though this client, you may be thinking to yourself that it would be nice to be able to control these features and get this information programmatically. This is possible through WMI. With a little bit of skill in .NET, a SQL Server DBA or developer can fully exploit the capabilities of SQL Server (and simplify their life, too).

Performance Tools

If you select Start ➢ All Programs ➢ Microsoft SQL Server 2008 ➢ Performance Tools from the menu, you will see two additional tools of note, the Database Engine Tuning Advisor and the SQL Server Profiler. While each of these will be discussed in more detail in later chapters, it is worth making a quick mention now of what they bring to the toolbox.

The SQL Server Profiler is an indispensable tool for auditing SQL Server activity. In previous versions, the tool was called SQL Trace. In essence, its job is to listen for events that occur within SQL Server and log them to a file or to a table. When defining a profiler log, you can decide exactly which event you are interested in logging and what data about each event you want to log. Although the event and data lists are predefined, they are extensive. You can watch for events such as connections, procedure calls, and T SQL actions. For each of these events, you can track client information, connection information, server values, and other data.

The real value in the SQL Server Profiler, though, is that is can be used for many different purposes, such as security auditing, performance tuning, and diagnostics. One of the most useful features of the SQL Profiler is to log T SQL calls and stored procedure calls. For instance, you can use the Profiler to log calls that are made by unfamiliar client applications. This allows the SQL Server DBA to know exactly what the client is doing even if it is a shrink-wrapped client or one with no documentation. You can use this information for performance tuning client applications or debugging problems that the client may be experiencing.

The Data Engine Tuning Wizard is another tool that has been around for a while, but the average SQL Server DBA probably does not use the tool to its fullest capability. While it is certainly not perfect, if you give it proper information regarding typical workflows, the Data Engine Tuning Wizard will make very reasonable recommendations related to indexing, and other performance optimizations.

The Profiler and the Tuning Wizard work together effectively to improve performance. You can use the Profiler to capture a snapshot of typical database activity and then use this log as the input for the Tuning Wizard to make its recommendations. After altering the data structures based on the Tuning Wizard's recommendations, you can replay the trace to see if the performance metrics improve. If not, you can start the whole process over again. We will address these two tools extensively in later discussions of performance analysis and management.

Third-Party Tools and Utilities

First of all, the authors want to be perfectly clear that they are not attempting to promote, endorse, or recommend—let alone warranty—any third-party tools. There is good and bad in any tool, and just because we like a particular tool does not mean that you will, but there are a few big players in the data tools arena that you should be familiar with, if for no other reason than to give you material to captivate your audience with at the next big DBA cocktail party you attend.

Embarcadero Technologies Embarcadero makes database products that fill quite a few holes in the marketplace. Their flagship tool is called ER/Studio. This is a design utility that puts the focus on good entity design. The concept is that if you can implement a good entity design, then a good data design will certainly follow. This is also a good product to use in a team environment. All design artifacts can be easily stored in a central data repository. Team members can log into the repository through the ER/Studio client and either check out designs for alteration or view them for reference.

Another of the nice features of this tool is that it lets you export a data design structure as an HTML catalog, making it very easy to share the design with other users.

Other Embarcadero products such as Schema Examiner and Rapid SQL aid the database developer by providing a solid platform for SQL development and schema management.

Computer Associates Computer Associates (CA) makes a data modeling tool called ERWin® that is a competitor to the Embarcadero ER/Studio tool. The concepts and feature sets of the two tools are very similar, but the interfaces are different and each tends to create very loyal users who grow accustomed to the methodology behind the tool.

CA has recently been making real strides in the SQL Server user community with the ERWin tool. Although the tool itself is and has always been Window-based, CA has been enhancing the tool's image extensively within the SQL Server community, including partnering with the Professional Organization for SQL Server (PASS) to produce a series of webcasts targeted specifically to SQL Server users.

Red Gate Software Red Gate also provides a good range of utilities, but unlike Embarcadero, Red Gate specializes in tools that are specific to Microsoft SQL Server rather than tools that work with a range of databases. Their SQL Compare and SQL Data Compare Utilities are quite effective for monitoring schema and data lineage changes. However, their SQL Prompt utility, which provided an Intelisense add-in for SQL Server management Studio, is unnecessary now that this feature is native in the Microsoft toolset.

Among the most compelling Red Gate tools is the SQL Backup tool, which handles the compression and encryption of SQL Server backups. Although SQL Server 2008 now supports these features, the ease of use of the Red Gate tool and the monitoring ability that it provides are still strong selling points.

The Red Gate SQL Refactor tool is another real gem for the SQL developer because it gives you a great deal of control when formatting and refactoring code, allowing you to avoid the problems typically associated with copy-and-paste refactoring and reformatting.

Quest Software Quest Software makes a great little development utility called TOAD. If you are an Oracle developer, you have probably heard of it, as it has become the de facto standard for Oracle development. Quest now makes a version for SQL Server.

TOAD is an easy-to-use and intuitive client. It's especially appealing at moments when the SQL Server Management Studio is a little too much to wade through—such as when all you want to do is execute some queries and write a few simple scripts. Best of all, there is a freeware version of TOAD for SQL Server, so you can try it out if you like. The full versions can be a bit expensive, but have a very loyal following.

There are many other tools out there—far too many to review here. If your favorite tool was not on this short list, we apologize; it was not a deliberate snub. The important thing is to find tools that you like and tools with which you can work. With all of the good tools on the market, you should always be able to find the right one to help you get the job done. At the same time, you owe it to yourself to evaluate additional tools when you get the opportunity. You never know what gems you might uncover if you are willing to dig a little to find them. In the end, it's all about getting the result you want as efficiently and cleanly as possible.

The Bottom Line

Utilize the Architect SQL Server services in the typical IT environment. SQL Server provides more than just simple data services to the IT infrastructure. SQL Server is packed with so many features that it quite literally has become a data suite in a box. The key is making sure that you know what SQL Server can do and how to use the appropriate feature once you identify the need for it.

Master It Which SQL Server feature would you use to meet each of the following business requirements?

1. Your company has been collecting sales data for over 10 years. The director of marketing has noticed that over the last two years, sales have been dropping in North America while they have been increasing in Central and South America. You need to see if you can find a reason for the trend.

2. Your company must comply with industry regulations that require the company to provide specific information in a specific format for regulatory inspectors. The inspectors must be able to get the information any time they choose, it must always be organized the way that they want to see it, and it must always contain current data.

3. A significant portion of your firm's sales is handled through the Internet. The marketing department would like to have a mechanism that allows users to search for a product, and if the product is not in stock, they can log in to the system and request to receive an email when the product is available.

4. Much of your company's data is stored as large text fields. This text data must be searchable as efficiently as possible using very fuzzy logic including word forms and proximity searches.

Install SQL Server 2008. While the installation process itself is wizard-driven, SQL Server installation takes some planning to make sure that you have an appropriate installation with the right resources in place for the right purposes.

Master It SQL Server needs the right balance of resources to handle the demands of the data service. What are the four primary resources that SQL Server uses and how do they interact with each other?

Use the Microsoft SQL Server toolset. Microsoft ships a number of high-quality tools with the SQL Server product that suit many different purposes. The key is to use the right tool for the job.

Master It Which of the SQL Server tools would you use for each of the following goals?

1. Writing and executing a query
2. Identifying the optimal indexing structure
3. Performing routine security administration tasks
4. Performing a security audit in which you try to identify failed login attempts
5. Identifying the interaction between SQL Server and a compiled client application
6. Configuring the network protocols that SQL Server can use

Implement other useful third-party tools. With all the tools out there, it is important that you select the right ones. There are a lot of tools from which to choose.

Master It What is the best third-party tool for creating entity-relationship diagrams?