

Infrastructure Planning and Design

Microsoft® SQL Server® 2008   
and SQL Server 2008 R2

Version 1.1

Published: February 2009

Updated: May 2010

For the latest information, please see [www.microsoft.com/IPD](http://go.microsoft.com/fwlink/)

Copyright © 2010 Microsoft Corporation. This documentation is licensed to you under the Creative Commons Attribution License.  To view a copy of this license, visit <http://creativecommons.org/licenses/by/3.0/us/> or send a letter to Creative Commons, 543 Howard Street, 5th Floor, San Francisco, California, 94105, USA.  When using this documentation, provide the following attribution: Infrastructure Planning and Design is provided with permission from Microsoft Corporation.

This documentation is provided to you for informational purposes only, and is provided to you entirely "AS IS". Your use of the documentation cannot be understood as substituting for customized service and information that might be developed by Microsoft Corporation for a particular user based upon that user’s particular environment. To the extent permitted by law, MICROSOFT MAKES NO WARRANTY OF ANY KIND, DISCLAIMS ALL EXPRESS, IMPLIED AND STATUTORY WARRANTIES, AND ASSUMES NO LIABILITY TO YOU FOR ANY DAMAGES OF ANY TYPE IN CONNECTION WITH THESE MATERIALS OR ANY INTELLECTUAL PROPERTY IN THEM.

Microsoft may have patents, patent applications, trademarks, or other intellectual property rights covering subject matter within this documentation. Except as provided in a separate agreement from Microsoft, your use of this document does not give you any license to these patents, trademarks or other intellectual property.

Information in this document, including URL and other Internet website references, is subject to change without notice. Unless otherwise noted, the example companies, organizations, products, domain names, email addresses, logos, people, places and events depicted herein are fictitious.

Microsoft, Excel, Hyper-V, SharePoint, SQL Azure, SQL Server, Windows, and Windows Server are either registered trademarks or trademarks of Microsoft Corporation in the United States and/or other countries.

The names of actual companies and products mentioned herein may be the trademarks of their respective owners.

You have no obligation to give Microsoft any suggestions, comments or other feedback ("Feedback") relating to the documentation. However, if you do provide any Feedback to Microsoft then you provide to Microsoft, without charge, the right to use, share and commercialize your Feedback in any way and for any purpose. You also give to third parties, without charge, any patent rights needed for their products, technologies and services to use or interface with any specific parts of a Microsoft software or service that includes the Feedback. You will not give Feedback that is subject to a license that requires Microsoft to license its software or documentation to third parties because we include your Feedback in them.

Contents

[The Planning and Design Series Approach 1](#_Toc262479040)

[Introduction to the SQL Server 2008 Guide 2](#_Toc262479041)

[IPD in Microsoft Operations Framework (MOF 4.0) 7](#_Toc262479042)

[SQL Server 2008 Design Process 8](#_Toc262479043)

[Step 1: Determine the Project Scope 10](#_Toc262479044)

[Step 2: Determine Which Roles Will Be Required 11](#_Toc262479045)

[Step 3: Design the SQL Server Database Engine Infrastructure 16](#_Toc262479046)

[Step 4: Design the SQL Server Integration Services Infrastructure 25](#_Toc262479047)

[Step 5: Design the SQL Server Analysis Services Infrastructure 28](#_Toc262479048)

[Step 6: Design the SQL Server Reporting Services Infrastructure 32](#_Toc262479049)

[Step 7: Design the SQL Server Master Data Services Infrastructure 36](#_Toc262479050)

[Conclusion 41](#_Toc262479051)

[Appendix A: Job Aids 42](#_Toc262479052)

[Appendix B: Server Performance Analyzing and Scaling 46](#_Toc262479053)

[Version History 57](#_Toc262479054)

[Acknowledgments 58](#_Toc262479055)

# 

# The Planning and Design Series Approach

This guide is one in a series of planning and design guides that clarify and streamline the planning and design process for Microsoft® infrastructure technologies.

Each guide in the series addresses a unique infrastructure technology or scenario. These guides include the following topics:

* Defining the technical decision flow (flow chart) through the planning process.
* Describing the decisions to be made and the commonly available options to consider in making the decisions.
* Relating the decisions and options to the business in terms of cost, complexity, and other characteristics.
* Framing the decision in terms of additional questions to the business to ensure a comprehensive understanding of the appropriate business landscape.

The guides in this series are intended to complement and augment the product documentation.

## Benefits of Using This Guide

Using this guide will help an organization to plan the best architecture for the business and to deliver the most cost-effective Microsoft SQL Server® 2008 technology.

Benefits for Business Stakeholders/Decision Makers:

* Most cost-effective design solution for an implementation. Infrastructure Planning and Design (IPD) eliminates over-architecting and overspending by precisely matching the technology solution to the business needs.
* Alignment between the business and IT from the beginning of the design process to the end.

Benefits for Infrastructure Stakeholders/Decision Makers:

* Authoritative guidance. Microsoft is the best source for guidance about the design of Microsoft products.
* Business validation questions to ensure the solution meets the requirements of both business and infrastructure stakeholders.
* High integrity design criteria that includes product limitations.
* Fault-tolerant infrastructure, where necessary.
* Proportionate system and network availability to meet business requirements. Infrastructure that is sized appropriately to meet business requirements.

**Benefits for** Consultants or Partners:

* Rapid readiness for consulting engagements.
* Planning and design template to standardize design and peer reviews.
* A “leave-behind” for pre- and post-sales visits to customer sites.
* General classroom instruction/preparation.

Benefits for the Entire Organization:

Using this guide should result in a design that will be sized, configured, and appropriately placed to deliver a solution for achieving stated business requirements, while considering the performance, capacity, manageability, and fault tolerance of the system.

# Introduction to the SQL Server 2008 Guide

This guide describes the process of planning a SQL Server 2008 or SQL Server 2008 R2 infrastructure. The guide addresses the following fundamental decisions and tasks:

* Identifying which SQL Server features will be needed.
* Design decisions related to the components, layout, and connectivity of the SQL Server infrastructure.

Following the instructions in this guide will result in a design that is sized, configured, and appropriately placed to deliver the stated business benefits, while also considering the performance, capacity, and fault tolerance of the system.

This guide addresses the scenarios most likely to be encountered by someone designing a SQL Server infrastructure. Customers should consider having their architecture reviewed by Microsoft Customer Service and Support prior to implementation as that organization is best able to comment on the supportability of a particular design.

SQL Server 2008 includes four primary components that can be implemented separately or as a group to form a scalable data platform:

* The Database Engine. The Database Engine is the core service for storing, processing, and securing data.
* Integration Services. Microsoft SQL Server Integration Services (SSIS) is a platform for building data integration solutions from heterogeneous sources, including packages that provide extract, transform, and load (ETL) processing for data warehousing.
* Analysis Services. Microsoft SQL Server Analysis Services (SSAS) supports OLAP (online analytical processing) and data mining functionalities. This allows a database administrator to design and create multidimensional structures that contain data aggregated from other data sources, such as relational databases.
* Reporting Services. Microsoft SQL Server Reporting Services (SSRS) delivers enterprise reporting functionality for creating reports that gather content from a variety of data sources, publishing the reports in various formats, and centrally managing their security and subscriptions.

Additionally, SQL Server 2008 R2 introduces a fifth component:

* **Master Data Services.** Master Data Services is a master data management (MDM) application built from platform components that may be deployed as an application or extended by use of the platform components to consistently define and manage the critical data entities of an organization. Master Data Services is an any-domain hub that supports but is not limited to domains such as product, customer, location, cost center, equipment, employee, and vendor.

For readers who might not be familiar with functionality other than the core database services, Figure 1 illustrates a possible set of relationships between the components that may comprise a SQL Server 2008 infrastructure. Note that the figure does not provide a comprehensive look at all possible data pathways; it is a single representation for illustrative purposes.

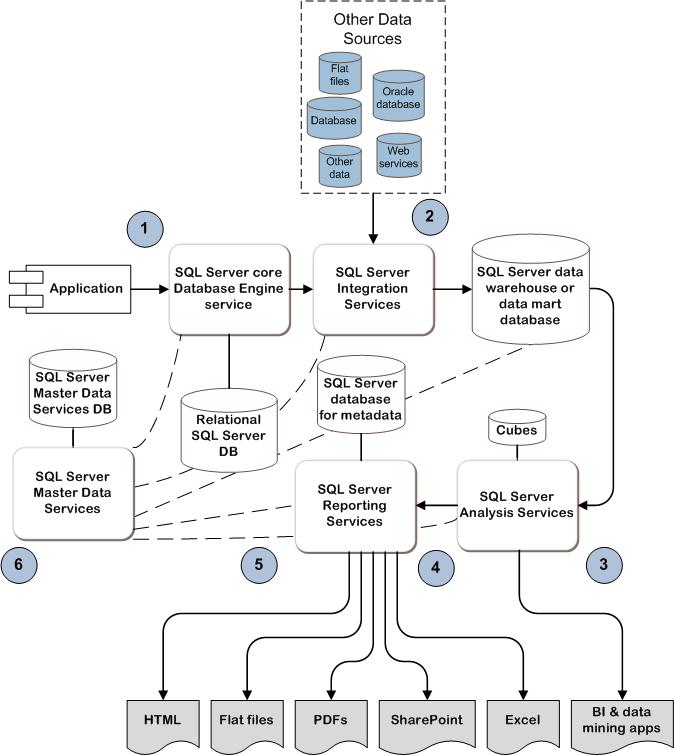


Figure 1. Example of one possible SQL Server infrastructure

To provide a perspective on how the different SQL Server components complement each other, a possible scenario (steps 1–6) is pictured above and described below:

1. An application takes input from users and writes entries to an OLTP database as it occurs.
2. Periodically, SQL Server Integration Services (SSIS) extracts the data from the OLTP database, combines it with other data existing in the organization (perhaps another database system or some flat file exports from legacy systems). SSIS then transforms the data, standardizes the formats of the data—for example, state names versus abbreviations—and then loads the data into another database—in this case, a SQL Server data warehouse.
3. SQL Server Analysis Services (SSAS) is then used to extract data from the data warehouse and place it into OLAP cubes. Cubes allow for complex analytical and as-needed queries with a rapid execution time.
4. Managerial decision makers can use Microsoft Excel® spreadsheet software or other applications to perform data mining to detect and predict trends in the business.
5. SQL Server Reporting Services (SSRS) is used to publish reports to other users in the organization, for example, salespeople who need to know current production levels. These reports are generated either on-demand or on a scheduled basis.
6. Master Data Services is a master data management application used to consistently define and manage the critical data entities of an organization.

Different editions of SQL Server are available to meet the various needs of an organization. Depending on what the organization wants to accomplish will determine which version of SQL Server will be implemented to meet those needs. The following provides an overview of the different SQL Server editions available:

Core Editions:

* Standard. A full-featured data platform for running small-scale to medium-scale online transaction processing (OLTP) applications and basic reporting and analytics.
* Enterprise. Provides additional features for scalability and performance, high availability (HA), security, data warehousing, and business intelligence (BI) tools, and supports a maximum of 50 instances.

Premium Editions (new for SQL Server 2008 R2):

* Datacenter. Built on the existing Enterprise edition code base, it is designed to support large SQL Server implementations by providing support for up to 256 logical processors, 25 instances of SQL Server, and up to 2 terabytes of RAM. (SQL Server 2008 R2 only.)
* Parallel Data Warehouse. A data warehouse hardware appliance that provides the ability to scale data warehousing services from tens to hundreds of terabytes. (SQL Server 2008 R2 only.)

Specialized Editions:

* Workgroup. Provides secure, remote synchronization and capabilities for running branch applications. It includes the core database features of the SQL Server product line and is easy to upgrade to Standard or Enterprise.
* Web. Specifically designed for highly available, Internet-facing, web-serving environments.
* Developer. Includes all the features and functionality found in Datacenter; however, it is strictly meant to be used for development, testing, and demonstration purposes only.
* Express. This free edition is for independent software vendors, nonprofessional developers, and hobbyists building client applications.
* Compact. This edition is typically used to develop mobile and small desktop applications. It is free to use and is commonly redistributed with embedded and mobile independent software vendor applications.

This guide will focus on the Standard and Enterprise editions, although the infrastructure principles discussed may still apply to other editions.

Note   See <http://msdn.microsoft.com/en-us/library/cc645993.aspx> for more detailed information on features in specific versions of SQL Server 2008 and <http://msdn.microsoft.com/en-us/library/cc645993(SQL.105).aspx> for SQL Server 2008 R2.

## What’s New in SQL Server 2008 R2

This guide’s design process is valid for both SQL Server 2008 and SQL Server 2008 R2 environments. Where appropriate, functionality only available in SQL Server 2008 R2 will be identified.

Built on SQL Server 2008, SQL Server 2008 R2 has the following expanded reporting and analytics functionalities through self-service business intelligence:

* **Master Data Services.** With Master Data Services, IT organizations can centrally manage critical data assets companywide and across diverse systems, enable more people to securely manage master data directly, and ensure the integrity of information over time.
* **Improved application and multi-server management capabilities:**
* The SQL Server Utility allows DBAs to centrally manage and view instances of SQL Server, data-tier applications, database files, and volumes.
* The Utility Control Point (UCP) collects configuration and performance information from managed instances of SQL Server every 15 minutes, and provides dashboard views of health summary of SQL Server resources.
* Data-tier applications (DAC) contain all of the database’s schema, dependent objects, and deployment requirements used by an application to improve the deployment of data applications and the collaboration between data-tier developers and DBAs.
* Utility Explorer dashboards to monitor resource utilization and health states.
* **Two new premium editions.** SQL Server 2008 R2 introduces two new premium editions to meet the needs of large scale data centers and data warehouses: Datacenter and Parallel Data Warehouse.
* **Integration with Microsoft SQL Azure™.** The client tools included with Microsoft SQL Server 2008 R2 allows DBAs to connect to SQL Azure, a cloud-based service that offers a flexible and fully relational database solution in the cloud.
* **Integration of SQL Server with Sysprep.** Allows DBAs to automate the deployment of SQL Server.
* **Analysis Services integration with SharePoint®.** SQL Server PowerPivot for SharePoint is a new role-based installation option in which PowerPivot for SharePoint will be installed on a new or an existing SharePoint 2010 server to support PowerPivot data access in the farm.

## Assumptions

In writing this guide, the following assumptions have been made:

* This guide does not address the business or technical case for choosing a database solution.
* The reader is familiar with Microsoft infrastructure solutions. This guide does not attempt to educate the reader on the features and capabilities of Microsoft products as the product documentation covers that information.
* The information in this guide applies to SQL Server 2008 only. Earlier versions are not included.
* The organization’s database administrator (DBA) will provide the business and technical requirements for the design.

## Feedback

Please direct questions and comments about this guide to [satfdbk@microsoft.com](http://www.microsoft.com/IPD).

We value your feedback on the usefulness of this guide. Please complete the following **Solution Accelerators Satisfaction Survey**, available at [http://go.microsoft.com/fwlink/?LinkID=132579](http://www.microsoft.com/IPD), and help us build better guidance and tools.

# IPD in Microsoft Operations Framework (MOF 4.0)

Microsoft Operations Framework (MOF) offers integrated best practices, principles, and activities to assist an organization in achieving reliable solutions and services. MOF provides guidance to help individuals and organizations create, operate, and support technology services, while helping to ensure the investment in technology delivers expected business value at an acceptable level of risk. MOF’s question-based guidance helps to determine what is needed for an organization now, as well as providing activities that will keep the organization running efficiently and effectively in the future.

Use MOF with IPD guides to ensure that people and process considerations are addressed when changes to an organization’s technology services are being planned.

* Use the Plan Phase to maintain focus on meeting business needs, consider business requirements and constraints, and align business strategy with the technology strategy. IPD helps to define an architecture that delivers the right solution as determined in the Plan Phase.
* Use the Deliver Phase to build solutions and deploy updated technology. In this phase, IPD helps IT pros design their technology infrastructures.
* Use the Operate Phase to plan for operations, service monitoring and control, as well as troubleshooting. The appropriate infrastructure, built with the help of IPD guides, can increase the efficiency and effectiveness of operating activities.
* Use the Manage Layer to work effectively and efficiently to make decisions that are in compliance with management objectives. The full value of sound architectural practices embodied in IPD will help deliver value to the top levels of a business.



Figure 2. The architecture of Microsoft Operations Framework (MOF) 4.0

# SQL Server 2008 Design Process

This guide addresses the following activities that must occur in planning the design for SQL Server 2008. The six steps that follow represent the most critical design elements in a well-planned SQL Server 2008 design:

* Step 1: Determine the Project Scope
* Step 2: Determine Which Roles Will Be Required
* Step 3: Design the SQL Server Database Engine Infrastructure
* Step 4: Design the SQL Server Integration Services Infrastructure
* Step 5: Design the SQL Server Analysis Services Infrastructure
* Step 6: Design the SQL Server Reporting Services Infrastructure
* Step 7: Design the SQL Server Master Data Services Infrastructure

The items in this list represent tasks that must be carried out. These types of items are addressed because their presence is significant in order to complete the infrastructure design.

## Decision Flow

Figure 3 provides a graphic overview of the steps involved in designing a SQL Server 2008 infrastructure.

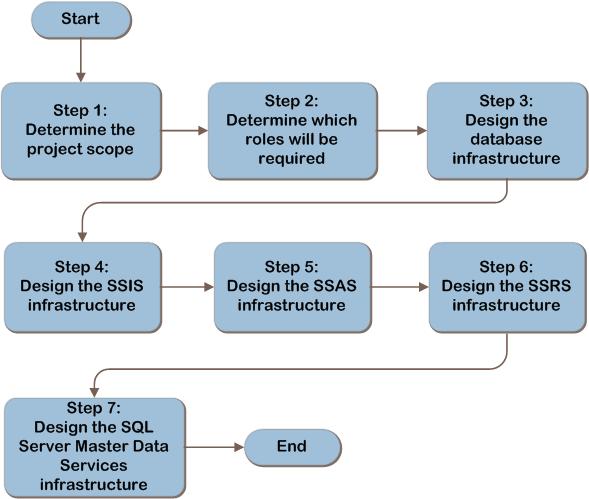


Figure 3. SQL Server 2008 R2 infrastructure decision flow

## Guide Audience

This guide addresses considerations that are related to planning and designing the necessary components for a successful SQL Server infrastructure and is written to address the needs of the following groups:

* Organizations that are implementing SQL Server for a specific application.
* Organizations that are implementing SQL Server to provide a business intelligence platform for managers and decision makers to conduct data integration, reporting, and analysis.
* Organizations that are implementing SQL Server as a service to be available to multiple business units.
* Organizations that are upgrading an existing SQL Server implementation by migrating to new hardware.

## Out of Scope

This guide does not address the following:

* In-place upgrades, where an older instance of SQL Server is upgraded to SQL Server 2008 on the existing server; therefore, no infrastructure changes occur.
* Side-by-side upgrades, where a new instance of SQL Server 2008 is installed on the same server as an instance of SQL Server 2005, and then the data is moved, with no infrastructure changes occurring.
* Developer, Express, Compact, Evaluation, and Parallel Data Warehouse editions of SQL Server. The Developer edition has all the capabilities of the Enterprise edition but is not licensed for any form of production use. The Express edition is an entry-level database for learning and ISV redistribution. The Compact edition is an embedded database for developing desktop and mobile applications. The Evaluation edition is used for evaluating SQL Server. The Parallel Data Warehouse solution comes preassembled from certified hardware vendors.
* Reporting Services can integrate with Microsoft Office SharePoint Server 2007 to provide a user interface (UI) to administer, secure, manage, view, and deliver reports. SharePoint integration will not be covered in this guide.
* Database design as it addresses the structure of the actual database.

# Step 1: Determine the Project Scope

In Step 1, the project scope will be determined in order to align the goals of the project with the business motivation. The results of this step will be used in the next step to determine which SQL Server roles will be required to meet the business goals.

Understanding the needs of the business enables the designer to focus the design to meet the business requirements. The database administrator (DBA) can serve as a trusted advisor and major influence for the design team given the DBA’s deeper working knowledge of SQL Server features beyond the core infrastructure.

There are two primary architecture approaches—application-specific or as a service—for designing a SQL Server implementation.

In the application-specific approach, the SQL Server infrastructure is designed to support a specific application. This approach involves gathering the requirements about the application, determining which SQL Server roles will be required, and optimizing the server design for the specific applications. This is the approach used in this guide.

A second approach in designing a SQL Server architecture is to consider the SQL Server needs of many applications across the enterprise and to design SQL Server as a service. The SQL Server as a service approach can be made available as a general platform for the business units, which can deliver standardization and the associated economies of scale. The IT department can determine the SQL Server needs of specific applications based on data throughput, storage, memory, fault tolerance, security, and availability requirements, then balance incoming requests against existing capacity and performance, and allocate the appropriate resources. This guide is not written for this approach; however, all of the tasks will be applicable so it may provide a starting point. With this service-oriented approach, several steps in this guide may need to be repeated as each role is designed.

## Task 1: Determine Applications in Scope

Before beginning to plan and design a SQL Server infrastructure, an organization needs to identify the applications requiring a SQL Server database or the application sources for business intelligence analysis and reporting. This document guides the reader through the design process for a single application; therefore, the steps will need to be repeated for any additional applications.

Use Table A-1 in Appendix A: “Job Aids” to record the name of the applications or business needs in scope for this SQL Server design project.

## Step Summary

In Step 1, the project scope was determined in order to align the goals of the project with the business motivation. The results of this step will be used in the next step to determine which SQL Server roles will be required to meet the business goals.

# Step 2: Determine Which Roles Will Be Required

In the previous step, the scope of the project was determined. In this step, the product roles required to deliver the business requirements and desired functionality will be identified.

## Task 1: Determine If a Database Engine Will Be Required

The Database Engine is the core service for storing, processing, and securing data.   
It supports relational, or OLTP, and data warehouse databases. If the application being implemented requires one or more OLTP databases, record the database names in Table A-2 in Appendix A. If the application requires one or more data warehouse databases, record the names of the databases in Table A-2 in Appendix A.

Other services that may be selected in later tasks could determine the need for the Database Engine. Reporting Services requires access to a SQL Server-based database server to store metadata; this metadata database can be on the same server as Reporting Services or on another database server. The need for Reporting Services will be determined in Task 4. Integration Services can store packages in the msdb database or on the file system. The storage location will be determined in Step 4 and, depending on the outcome of that decision, a database may be required for Integration Services.

## Task 2: Determine If SQL Server Integration Services Will Be Required

SQL Server Integration Services (SSIS) can connect to a wide variety of data sources in order to extract the data, transform the data to compatible formats, merge it into one dataset, and then load it into one or more destinations, including flat files, raw files, and relational databases. Integration Services may be required if the organization needs to:

* Merge data from heterogeneous data sources.
* Populate data warehouses and data marts.
* Cleanse and standardize data.
* Automate administrative functions and data loading.

Figure 4 provides a graphic overview of many of the possible inputs and outputs for SSIS in a SQL Server 2008 infrastructure.

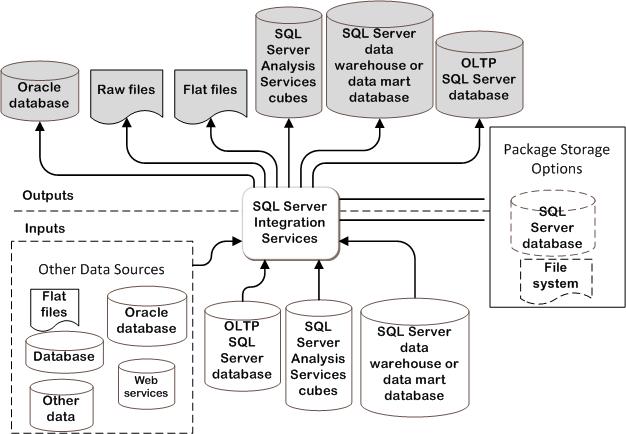


Figure 4. SSIS architecture, indicating many of the possibilities for inputs and outputs

Record whether Integration Services will be required in the organization in Table A-1 in Appendix A.

## Task 3: Determine If SQL Server Analysis Services Will Be Required

SQL Server Analysis Services (SSAS) supports OLAP (online analytical processing) and data mining functionalities. This allows a database administrator to design and create multidimensional structures that contain data aggregated from other data sources, such as relational databases.

Analysis Services may be needed if the organization needs to rapidly access reports with varying degrees of granularity (for example, yearly totals, monthly totals, quarterly totals, and individual transactions) and requires that the yearly totals appear just as quickly as daily totals. An online transaction processing (OLTP) system would have to add up these values, while an SSAS system, in many cases, will already have the answers pre-calculated.

Figure 5 provides a graphic overview of the inputs and outputs for SSAS in a SQL Server 2008 infrastructure.

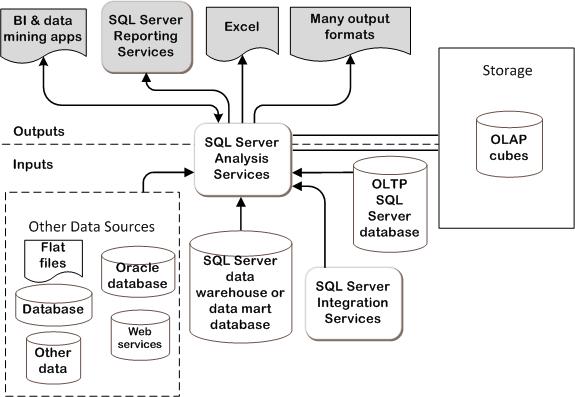


Figure 5. Illustration of the varied inputs and outputs of SSAS

Record whether Analysis Services will be required in the organization in Table A-1 in Appendix A.

## Task 4: Determine If SQL Server Reporting Services Will Be Required

SQL Server Reporting Services (SSRS) delivers enterprise reporting functionality for creating reports that gather content from a variety of data sources, publishing the reports in various formats, and centrally managing their security and subscriptions. Reports can be delivered on a scheduled basis to users, accessed via a web interface from SharePoint, or from custom applications including Windows® Forms applications that call via web services.

Reporting Services can be used to generate reports on OLTP databases, SSAS cubes, data warehouses, data marts, or third-party data sources such as flat files, Oracle databases, or web services.

Figure 6 provides a graphic overview of the inputs and outputs for SSRS in a SQL Server 2008 infrastructure.

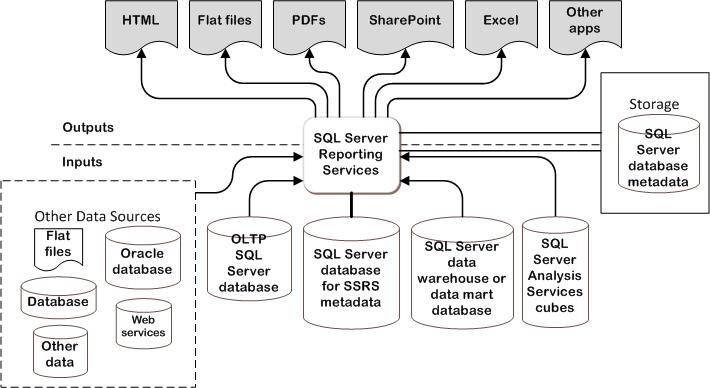


Figure 6. SSRS can be used to deliver reports in a multitude of formats from a variety of data sources.

Record whether Reporting Services will be required in the organization in Table A-1 in Appendix A.

If SSRS is required, the Database Engine will also be required and should also be recorded in the same table.

## Task 5: Determine If SQL Server Master Data Services Will Be Required

SQL Server Master Data Services provides a master data hub to centrally organize, maintain, and manage master data within the organization. By integrating disparate operational and analytic systems with Master Data Services, all applications across the organization will rely on a central, accurate source of information. Using Master Data Services, a single source of master data is created, and an auditable record of that data is maintained as it changes over time.

**Note**Master Data Services is only available in the Enterprise, Datacenter, or Developer editions of SQL Server 2008 R2.

Figure 7 provides a graphic overview of the inputs and outputs for Master Data Services in a SQL Server 2008 R2 infrastructure.

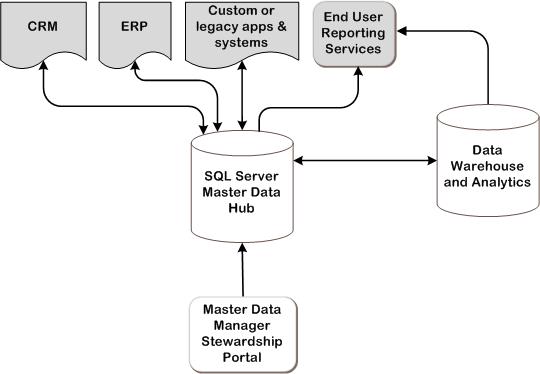


Figure 7. Master Data Services is an any-domain hub that supports domains such as product, customer, location, cost center, equipment, employee, and vendor.

Record whether Master Data Services will be required in the organization in Table A-2 in Appendix A.

If Master Data Services is required, the Database Engine will also be required and should also be recorded in the same table.

## Step Summary

In Step 2, it was determined which SQL Server roles will be required to support the business goals. Note that the Database Engine may be required by the application or by other SQL Server roles in certain circumstances.

# Step 3: Design the SQL Server Database Engine Infrastructure

In the previous step, the roles required to support the organization’s needs were determined. If it was determined that an OLTP database is not required, go to Step 4.   
In Step 3, the database requirements will be gathered, and the database infrastructure will be designed from those requirements.

A database has a minimum of two files associated with it: a data file and a log file, and it can also have secondary data files. Databases can range in size from 1 MB to a theoretical limit of 1,048,516 terabytes.

An instance of SQL Server is a SQL Server executable program running on a server. Each instance has a name, a sqlservr.exe process in memory, buffer memory, its own copies of the system databases, and its own set of user databases. A default instance is identified by the name of the computer on which the SQL Server 2008 Database Engine is running; it does not have a separate instance name. All instances of SQL Server other than the default instance are identified by the instance name that was set during installation. Applications connect to a named instance by specifying the computer name and the instance name in the format computer\_name\instance\_name.

A single database server instance can have up to 32,767 databases; the number of instances that can exist on a server is dependent on the product limitations for each edition (16 instances for SQL Server 2008 Standard, 50 instances for SQL Server 2008 Enterprise).

## Task 1: Determine Capacity and Performance Requirements

Before the database infrastructure can be designed, storage and performance requirements will need to be gathered. The product group does not provide guidance on physical design, so consult the database administrator to obtain the following information.

### Disk Storage Required

For databases that don’t yet exist, an estimate will need to be made of the disk storage required. Storage needs should be calculated for the database, transaction log, indexes, and tempdb database.

To generally estimate the database’s storage needs, multiply the average row length by the number of records per month that will be entered into the database. Multiply that result by the number of days’, weeks’, or months’ worth of data that will be kept in the database to obtain the approximate amount of storage required. Add about 5 percent to this for database overhead. This could be larger or smaller for some record layouts, but 5 percent is a good number to use for an estimate. For a more detailed size estimate, see “Estimating the Size of a Database” at <http://msdn.microsoft.com/en-us/library/ms187445.aspx>. Record the disk storage requirements in Table A-3 in Appendix A.

### IOPS and Throughput Required

Disk throughput is a measure of the data transfer rate through an I/O (input/output) subsystem. Since the main function of SQL Server is to manipulate data, and that data resides either in memory or on the I/O subsystem, any I/O performance problems will result in performance degradation of SQL Server.

Although it may not be possible to calculate the required IOPS (IOs per second) in advance, benchmarks for some workloads may be available from SAN and disk vendors that may provide a baseline for estimating the required performance and the disk storage configuration required to deliver that performance level. Additionally, most vendors, including Microsoft, have test labs that can be scheduled to simulate the production database scenario using the selected disk subsystem and the projected IOPS. If determining the required IOPS manually, see Appendix B: “Server Performance Analyzing and Scaling” for a detailed discussion on the calculation. Record the IOPS and throughput required in Table A-3 in Appendix A.

In general, additional smaller drives are better than one big one for database systems. For example, it is better to have nine or ten 128-GB drives than one terabyte drive.

## Task 2: Determine Whether to Place the Database in a New or Existing Instance

For each database identified in Task 1 of Step 2, it must be determined whether to place the database in an existing instance or to create a new one. If the server hosting an existing instance has excess capacity to support the database, it may be simpler to place the database within an existing instance. However, there are potential reasons why a database might not be able to be located within an existing instance:

* Regulatory requirements. Corporate policies or government regulations may require separate storage for certain types of data or for certain parts of the organization.
* Memory isolation. There may be a database application with a memory leak, and thus it may be desirable to place it in its own instance and cap the memory available to this instance to protect other databases on the server from being affected.
* Fault tolerance. Clustering is on the instance level; thus it will be necessary to place a database in an instance with similar fault-tolerance requirements.
* Authentication. Authentication is set at the instance level to be either SQL Server authentication or Windows authentication. The new database’s authentication requirements will need to be compatible with the instance’s authentication setting.
* Security concerns. Total database encryption can be used to add security to a database. The service master key is for all databases in an instance. To minimize exposure if a key is compromised, it may be desirable to place the database in a separate instance.
* Support requirements. The vendor or developer may require that the database runs in a separate instance.

Record the placement of each SQL Server database in Table A-3 in Appendix A.

## Task 3: Determine Whether to Place the Instance on a New or Existing Server Running SQL Server

If it was determined in Task 2 that a new instance needs to be created for the database,   
it next needs to be decided whether the instance will be created on an existing SQL Server-based server that already has one or more instances on it.

Running multiple instances on a given server is ideal in the following circumstances:

* To support multiple test and development environments on a single large server.
* To run multiple applications on a server, and to have each application install its own instance of the SQL Server 2008 Database Engine.
* To securely isolate the databases available on a single server.

Multiple instances of SQL Server 2008 Database Engines have their own set of system and user databases. Each instance has separate SQL Server and SQL Server Agent services. All other components and services are shared, which adds to the overhead on the server due to managing the shared resources.

Record the placement of each SQL Server database instance in Table A-3 in Appendix A.

## Task 4: Determine the Number of Servers

The next task in designing the database infrastructure will be to determine the number of servers required.

Two factors that may change the number of servers running SQL Server are:

* Whether scaling out will be implemented.
* Whether fault tolerance will be implemented at the database level or instance level.

### Scaling Out

Scaling out is the process of distributing a SQL Server load across more than one server; thus the decision to scale out will increase the number of servers required. Multiple scale-out options are available, and this section will provide a brief overview of them. Scale-out options require design changes that occur at the database level and are outside the scope of this guide; therefore, it is expected that any decisions relative to scaling out will have been made in conjunction with the database administrator. Note that a new server, by definition, is a new instance.

Scale-out options include:

* Scalable shared databases. With scalable shared databases, a read-only copy of the database on a shared storage area network (SAN) drive is shared between several servers to spread out querying or reporting loads.
* Peer-to-peer replication. Instead of several database engines accessing a single copy of the database, multiple copies of the database are made, with each database engine maintaining its own copy of the database. Replication provides scale out, while also allowing updates to the data. Replication is used to propagate changes to all copies of the data.
* Linked servers and distributed queries. SQL Server has the ability to query remote database objects as if they were local. This means that a scaled-out database can look like a single large database to an application. The major change to the SQL queries is that the table names will now include the name of the linked server where the data is located.
* Distributed partition views. The data in a table is partitioned among tables in several distributed databases, based on a partitioning key. For example, a customer table might be partitioned on customer number ranges: 1–10,000 in one database, 10,001–20,000 in a second database, 20,001–30,000 in a third, and so on. A query that accesses a particular customer number will be executed only on the database that contains the desired customer number. Queries that don't reference a particular customer number will need to be run on all the partitions.
* Data dependent routing (DDR). With DDR, the data is partitioned among databases, and either the application or some middleware services route queries to the correct database. DDR is usually combined with distributed partitioned views to form a complete solution.
* Service-Oriented Data Architecture (SODA). For scaling out, SODA data can be split in multiple ways. The most common way is to divide services among databases and split the data so that each database contains the data for the services it contains. Some data will be shared across service boundaries. The preferred way for the service to access data that it doesn't own is to call the service that owns the data, but this method can be inefficient in some cases. Instead, replication or linked servers can be used to access data that a service doesn't own but uses extensively. Direct access of data in another service should be avoided, if possible. The main exception to this is reference data that can be replicated to all services that require access to the reference data.

If splitting data on service boundaries doesn't provide enough scale out, multiple copies of each service database can be created, with each copy handling a data partition. This works the same way as DDR, with a front-end service that looks at the message data and routes the message to the database that owns the required data.

* Service Broker. Service Broker is a queued management system implemented in the database. It can be useful to scale out data collection and data processing tasks. Consider using Service Broker when an application can asynchronously handle the data flow.

The decision to use a scale-out solution is influenced by several factors. The table below summarizes the importance of these factors for each solution. For further information relative to scaling out, see “Scaling Out SQL Server 2005” at <http://msdn.microsoft.com/en-us/library/aa479364.aspx>.

Each of the scale-out options will increase the number of SQL Server-based servers in the organization. Determine whether any of the scale-out options will be implemented and, if so, record the number of servers required to support the selected option in Table A-3 in Appendix A.

Table 1. Factors Influencing the Selection of Scale-Out Solutions

| Scale-out solution | Update frequency | Ability to change application | Data partitionability | Data coupling |
| --- | --- | --- | --- | --- |
| Scalable shared databases | Read only | Limited or no change required | No requirement | No requirement |
| Peer-to-peer replication | Read mostly, no conflicts | Limited or no change required | No requirement | No requirement |
| Linked servers | Minimize cross-database updates | Minor changes | Not typically required | Important to have coupling |
| Distributed partition views | Intended mostly for read | Some changes may be required | Very important | Limited impacts |
| Data-dependent routing | Frequent updates okay | Significant changes possible | Very important | Low coupling may help some applications |
| Service-Oriented Data Architecture | Frequent updates okay | Changes usually not required | Not required, unless combined with DDR | Low coupling between services required |
| Service Broker | Frequent updates okay | Significant changes possible | No requirement | No requirement |

### Database-Level Fault Tolerance

Three types of fault tolerance are available to provide protection at the database level:

* Database mirroring
* Log shipping
* Replication

#### Option 1: Database Mirroring

Database mirroring maintains two copies of a single database that must reside on different server instances (principal and mirror) of SQL Server Database Engine. A principal instance serves the database to clients. The mirror instance acts as a hot or warm standby server, depending on the configuration and state of the mirroring session. When a database mirroring session is synchronized, database mirroring provides a hot standby server that supports rapid failover without a loss of data from committed transactions as long as the applications are configured to support it. When the session is not synchronized, the mirror server is typically available as a warm standby server, with possible data loss.

Database mirroring offers the following benefits:

* Uses a single, duplicate copy of the database. If additional copies are required, log shipping can be used on the database in addition to database mirroring.
* When database mirroring operates synchronously, provides for zero work loss through delayed commit on the principal database.

#### Option 2: Log Shipping

Log shipping provides the ability to automatically send transaction log backups from a primary database on a primary server instance to one or more secondary databases on separate secondary server instances. The transaction log backups are applied to each of the secondary databases individually. An optional third server instance, known as the monitor server, records the history and status of backup and restore operations. Optionally, the monitor server provides alerts if these operations fail to occur as scheduled.

Log shipping can be a supplement or an alternative to database mirroring. Although similar in concept, asynchronous database mirroring and log shipping have key differences. Log shipping offers the following distinct capabilities:

* Supports multiple secondary databases on multiple server instances for a single primary database.
* Allows a user-specified delay during the time when the primary server backs up the log of the primary database and when the secondary servers must restore the log backup. A longer delay can be useful, for example, if data is accidentally changed on the primary database. If the accidental change is noticed quickly, a delay will allow retrieval of still unchanged data from a secondary database before the change is reflected there.

Asynchronous database mirroring has the potential advantage over log shipping in that there is a shorter time between when a given change is made in the primary database and when that change is reflected to the mirror database.

An advantage of database mirroring over log shipping is that high-safety mode is a no-data-loss configuration that is supported as a simple failover strategy.

#### Option 3: Replication

Replication uses a publish-subscribe model. Using this replication model allows a primary server, referred to as the Publisher, to distribute data to one or more secondary servers, or Subscribers. Replication was not originally built as a fault-tolerant option but can be used as such.

Replication delivers the following benefits:

* Allows filtering in the database to provide a subset of data at the secondary databases because it operates at the database level.
* Allows more than one redundant copy of the database.
* Allows real-time availability and scalability across multiple databases, supporting partitioned updates.
* Allows complete availability of the secondary databases for reporting or other functions, without query recovery.

Each of the database-level fault-tolerance options will increase the number of servers running SQL Server in the organization. Determine whether any of the database-level fault-tolerance options will be implemented and, if so, record the number of servers required to support the selected option in Table A-3 in Appendix A.

### Instance-Level Fault Tolerance

Failover clustering is the only fault-tolerance option that provides high availability for the instance namespace, which is how applications are configured to connect to the SQL Server database. It provides fault tolerance in the case of a single server failure and supplies these benefits:

* Automatic detection and failover
* Manual failover
* Transparent client redirect

Failover clustering has these constraints:

* Operates at the server instance level.
* Has no reporting on standby.
* Utilizes a single copy of the database.
* Does not protect against disk failure.
* Does not protect against data corruption.

Note that failover clustering of guests is not supported in a Microsoft Hyper-V™ environment. Therefore, if clustering is desired, a physical server will be required.

Decide whether failover clustering protection is desired, and record this in Table A-3 in Appendix A.

## Task 5: Determine Placement of Each New Instance

In this task, the placement of each new instance of SQL Server will be determined. For each new instance identified in Task 2 of this step, determine:

* The location of the instance. Decide in which office or geographical location the instance will be placed.
* Whether the instance will be on a physical or virtual server. Virtualization introduces flexibility into an environment by allowing virtual machines (VMs) to be moved easily between hosts. Services that may not be compatible with each other can run on the same host because of the isolation that VM provides. However, virtualization overhead may decrease the number of clients that a given VM can support.
* Whether the instance will be on existing hardware or new hardware. It will be important to ensure in either case that the performance requirements are met.

If virtualization is selected, additional planning is required for the physical computers that will host the VMs. The Infrastructure Planning and Design Guide for Windows Server Virtualization at <http://technet.microsoft.com/en-us/library/bb897507.aspx> provides the planning process for server virtualization.

Record the placement of each SQL Server instance in Table A-3 in Appendix A.

## Task 6: Select the Server Hardware

This task will be used to provide hardware considerations for each physical server on which a SQL Server database will be deployed based on the requirements gathered in Task 1 of this step.

Little specific architectural information is available to use in determining the hardware for the servers. However, sizing calculators are available from hardware vendors to help in designing the server hardware. See the “Additional Reading” section for links to these tools. In addition to using a calculator, considerations for the hardware are given below.

### CPU

It can be challenging to predict the CPU consumption of a database that doesn’t yet exist, and many variables can affect CPU utilization. The following are general guidelines when selecting new hardware:

* 64-bit processors. 64-bit hardware can address additional memory space. A 32-bit processor can only address 4 GB of memory natively. Address Windowing Extensions (AWE) can be used to access more memory, but there are limits to what the memory addressed above 4 GB can be used for within a 32-bit processor running SQL Server.
* L2 or L3 cache. Larger L2 or L3 processor caches generally provide better performance and often play a bigger role than raw CPU frequency.
* Multi-core and multiple processors. Research data has shown that two CPUs are not as fast as one CPU that is twice as fast. Because it is not always possible to get a CPU that is twice as fast, doubling the number of CPUs is preferred, but this does not guarantee twice the performance. However, SQL Server is efficient at parallelizing operations where the workload can be divided into multiple steps to provide increased performance on multiple CPUs.

Record the number of CPUs, architecture, and processor speed expected to be required for handling the required loads in Table A-3 in Appendix A.

Additional information on CPU scaling can be found in Appendix B: “Server Performance Analyzing and Scaling.”

### Memory

By design, SQL Server uses all available memory to cache data and query plans from the disk, so it isn’t necessary to recreate the data every time a query is run. In the case of a small database—for example, 2 GB on the disk—it isn’t necessary to have 16 GB of memory. However, the size of the database will be greater than the amount of available memory in most cases.

For OLTP applications, the product group recommends 2–4 GB of memory per processor core, and for data warehouses, 4–8 GB of memory per processor core. Determine the amount of memory that will be used and record this in Table A-3 in Appendix A.

### Disk Subsystem

Design the disk subsystem to meet the requirements that were determined in Task 1, in the following areas:

* Provide the required storage space.
* Deliver the performance to support the required number of IOPS and throughput.
* Provide the desired level of protection against hardware failures. If a SAN is used, this may include redundant host bus adapters (HBAs) and switches.

Record the disk subsystem configuration in Table A-3 in Appendix A.

For more information on calculating disk performance, see Appendix B.

### Network Adapters

Network adapters and accompanying drivers are available from some hardware manufacturers to provide fault tolerance or teaming of network ports for increased throughput. Hardware and driver support for the fault tolerance and teaming technology is provided by the hardware manufacturer. Determine the number of network adapters that will be used and record this in Table A-1 in Appendix A.

## Validating with the Business

Review the decisions made for each application with the affected business units by considering the following questions:

* **Have all geopolitical issues been appropriately addressed in the placement of the databases, instances, and servers?** For example, if two business units have their own IT departments, it may be necessary to have separate instances for each business unit.
* Are the server placement decisions in accordance with regulatory and security compliance requirements? Organizations might have a business need to store certain types of data and applications within the corporate data center.

## Step Summary

In Step 3, the database requirements were gathered and the database infrastructure was designed from those requirements. The capacity and performance requirements were determined, and decisions were made whether to place the database in a new or existing instance and whether to place the instance on a new or existing server running SQL Server.

In addition, the scale-out and fault-tolerance requirements were used to determine the number of servers required, and a determination was made as to the placement of each new instance. Finally, the server hardware was selected.

The following Step 3 outputs were recorded in Table A-3 in Appendix A:

* The disk capacity and performance and IOPS and throughput requirements, and the disk subsystem configuration to support the requirements.
* The number of CPUs, architecture, and processor speed.
* The amount of memory required.
* The number of network adapters.
* Whether the database will be placed in a new or existing instance, and on a new or existing SQL Server-based server.
* The number of servers required to support scale-out and fault-tolerance requirements.
* The office or geographical location where each new instance will be placed, and whether an instance will be on a physical server or in a virtualized environment.

## Additional Reading

* “Estimating the Size of a Database”: <http://msdn.microsoft.com/en-us/library/ms187445.aspx>
* “Determining SQL Server database storage requirements”: <http://searchsqlserver.techtarget.com/tip/0,289483,sid87_gci1289528_mem1,00.html>
* “Analyzing I/O Characteristics and Sizing Storage Systems for SQL Server Database Applications”: <http://msdn.microsoft.com/en-us/library/ee410782.aspx>
* “Optimize disk configuration in SQL Server”: <http://searchsqlserver.techtarget.com/tip/0,289483,sid87_gci1262122,00.html>
* “Maximum Capacity Specifications for SQL Server”: <http://msdn.microsoft.com/en-us/library/ms143432.aspx>
* “Predeployment I/O Best Practices: SQL Server Best Practices Article”: <http://www.microsoft.com/technet/prodtechnol/sql/bestpractice/pdpliobp.mspx>
* “Hardware and Software Requirements for Installing SQL Server 2008”: <http://technet.microsoft.com/en-us/library/ms143506.aspx>
* “Running SQL Server 2008 in a Hyper-V Environment”: <http://download.microsoft.com/download/d/9/4/d948f981-926e-40fa-a026-5bfcf076d9b9/SQL2008inHyperV2008.docx>
* SQLIO Disk Subsystem Benchmark Tool: <http://www.microsoft.com/downloads/details.aspx?familyid=9a8b005b-84e4-4f24-8d65-cb53442d9e19&displaylang=en>
* “Network adapter teaming and server clustering”: <http://support.microsoft.com/kb/254101/>
* Dell SQL Server Advisor tool: <http://advisors.dell.com/advisorweb/Advisor.aspx?advisor=214e88fe-eb6e-4d1c-86bf-b7d7dd092c38&c=us&l=en&cs=555>
* HP Transaction Processing Storage Planning Calculator for Microsoft SQL Server: <http://h71019.www7.hp.com/ActiveAnswers/cache/127077-0-0-0-121.html>
* IBM Systems Advisor Tool: [http://www-03.ibm.com/systems/express/systemsadvisor/index.html?ca=eserver&me=w&p\_creative=s&met=flash&p\_site=servershp#](http://www-03.ibm.com/systems/express/systemsadvisor/index.html?ca=eserver&me=w&p_creative=s&met=flash&p_site=servershp%23%20)
* “Distributed Partitioned Views / Federated Databases: Lessons Learned”: <http://sqlcat.com/technicalnotes/archive/2007/09/11/distributed-partitioned-views-federated-databases-lessons-learned.aspx>

# Step 4: Design the SQL Server Integration Services Infrastructure

In Step 2, the roles that need to be implemented were determined. If it was determined that SQL Server Integration Services (SSIS) is required in the organization, the SSIS infrastructure will be designed in this step. If it was determined that SSIS is not required, go to the next step.

## Task 1: Determine Resource Requirements

Data cannot be transformed faster than it can be read or written, and since there isn’t a calculator available to determine capacity and performance for the SSIS role, this task presents several items for consideration.

The following figure represents a high-level data path for SSIS.



Figure 8. Data path for SSIS

SSIS takes data from one source and moves it to another, so the areas to consider when determining the resource requirements include:

* The data source.
* The network from the data source to the SSIS server.
* The SSIS server.
* The network to the destination.
* The destination server.

I/O and storage needs for SSIS itself are minimal; however, the sources and targets will be affected as data is read and written, respectively.

Although the extract and load phases will access disk when reading from source and writing to the destination servers, the transformation itself should process in memory; therefore, ensure that the SSIS system has enough memory to prevent any performance degradation that might occur if the transformation swaps to disk. The product group recommends a memory range of 2–8 GB per processor core, with an average of 4 GB per processor core.

The total volume of data should be calculated to the best of the organization’s ability, and an estimate made of whether the network and I/O subsystems will perform as required. Ensure that each component of the system meets the capacity and performance requirements of the business and record this information in Table A-4 in Appendix A.

Record all source and destination data types in Table A-4 in Appendix A. This will be used to determine the required edition and version of SQL Server. Record the version that will be required for each SSIS server to support the business requirements to connect to other systems in Table A-4 in Appendix A.

SQL Server is available in 32-bit and 64-bit versions. Depending on the Open Database Connectivity (ODBC) drivers available for the source and destination, a particular version may be required. Many legacy ODBC drivers are only available in 32-bit versions.

This task should be repeated for each additional instance required for SSIS.

## Task 2: Decide Where the Integration Services Packages Will Be Stored

An Integration Services package is the set of instructions that is retrieved and executed against the data. Packages can be stored in the SQL Server msdb database or on the file system.

If it is decided that an msdb database will be used to store the packages, it is recommended that a local instance of the database engine for the msdb database be installed. A remote msdb database can be used, but it does not use the dedicated SSIS server as effectively as a local msdb database does.

Record in Table A-4 in Appendix A the name of the instance of SQL Server or file share where the packages are stored. If it is determined that the packages will be stored in a SQL Server msdb database, the Database Engine will be required. If the Database Engine is required, record this in Table A-4 in Appendix A.

## Task 3: Determine the Number of SSIS Servers Required

SSIS does not support clustering or provide for scaling out to automatically load balance. If the load exceeds the capacity of the server to meet the business’s requirements, such as having the data transformed within a period of time, add additional servers to perform parallel loads or processing and manually divide the SSIS tasks among the servers.

Record the number of SSIS servers required in Table A-4 in Appendix A.

## Task 4: Determine Placement

Network throughput is important with the SSIS role as it is moving data from one system to another. Factors such as politics, policy, network constraints, proximity to data sources, and geography may also affect the decision of where to place the SSIS. For example, if external storage systems that connect to the SSIS server using either iSCSI or fiber channel are used, some of the load on the production network may be reduced.

Determine where to place each SSIS server. SSIS can be run in a virtualized environment if the memory, disk, and network requirements do not exceed the throughput capabilities of the virtual machine.

SQL Server roles can coexist with other SQL Server roles on the same server, so if imports are expected to only happen occasionally or at times or load levels where they will not affect other services, the SSIS role can be installed on an existing server. However, recall that SSIS does not support clustering. Decide whether the SSIS role will coexist with other roles on an existing server or if it will reside on a dedicated server.

Record in Table A-4 in Appendix A whether SSIS will be placed in a virtualized or physical environment and whether the SQL Server role will coexist with other SQL Server roles on the same server or reside on a dedicated server.

This task should be repeated for each additional instance required for SSIS.

## Step Summary

If it was determined in Step 2 that SQL Server Integration Services (SSIS) is required in the organization, the SSIS infrastructure was designed in this step. The resource requirements were determined and the location where the Integration Services packages will be stored was also determined. The number of SSIS servers required was determined as well where to place the SSIS server and SSIS roles. If it was determined that SSIS is not required, the reader was advised to go to the next step.

The following Step 4 outputs were recorded in Table A-4 in Appendix A:

* Capacity and performance requirements.
* All of the data sources and destinations for the various data types.
* The SSIS version.
* Name of SQL Server instance or file share where packages will be stored.
* Whether the Database Engine will be required.
* The number of required SSIS servers and the placement of each server.

## Additional Reading

* “Top 10 SQL Server Integration Services Best Practices”: <http://sqlcat.com/top10lists/archive/2008/10/01/top-10-sql-server-integration-services-best-practices.aspx>
* “Typical Uses of Integration Services”: <http://technet.microsoft.com/en-us/library/ms137795.aspx>
* “SQL Server Integration Services” in SQL Server 2008 Books Online: <http://technet.microsoft.com/en-us/library/ms141026.aspx>

# Step 5: Design the SQL Server Analysis Services Infrastructure

In Step 2, the roles that need to be implemented were determined. If it was determined that SQL Server Analysis Services (SSAS) is required in the organization, the SSAS infrastructure will be designed in this step. If it was determined that SSAS is not required, go to the next step.

## Task 1: Determine Resource Requirements

SSAS uses online analytical processing (OLAP) multi-dimensional databases, also called cubes, which are stored in a folder on the file system.

Before deploying SSAS, consider the resource requirements of the installation. In particular, consider disk storage requirements and memory and processor needs.

### Disk Storage Requirements

Cube sizes depend on the size of the fact tables and dimension members. If no other data is available, a good starting point may be to plan to allocate approximately 20–30 percent of the amount of storage required for the same data stored in the underlying relational database, if that’s where the data is originating.

In addition to the storage needed for the cube, consider the size of the aggregations. Aggregations are typically less than 10 percent of the size of the data stored in the underlying relational database, but they can vary based on the number of aggregations.

During processing, SSAS stores copies of the objects in the processing transaction on disk until it is finished, and then the processed copies of the objects replace the original objects. Therefore, sufficient additional disk storage must be provided for a second copy of each object. For example, if a whole cube will be processed in a single transaction, sufficient hard disk space to store a second copy of the whole cube will be required.

Determine the disk storage required, and record this in Table A-5 in Appendix A.

### IOPS and Throughput Requirements

Disk throughput is a measure of the data transfer rate through an I/O subsystem. The processing of OLAP databases is read and write-intensive, while queries will result in random reads. Design the I/O subsystem to support the expected IOPS and throughput that will be required.

### Memory and Processor Requirements

The amount of memory and processor resources that are available to Analysis Services varies depending on which version of the Windows operating system is installed on the server. For more information on running SSAS on Windows Server® 2008, see <http://technet.microsoft.com/en-us/library/ms175672.aspx>.

The product group recommends 4–8 GB of memory per processor core, but this can be more if queries return very large result sets.

## Task 2: Determine SQL Server Version

Evaluate the application for which SQL Server is being designed to select the required version. SQL Server is available in 32-bit and 64-bit versions. Depending on the ODBC drivers available for the source and destination, a specific version may be required. Many legacy ODBC drivers are only available in 32-bit versions. Even though most 32-bit ODBC versions will work on a 64-bit machine, the performance will not be as efficient as using a 64-bit version of the driver.

Use Table A-5 in Appendix A to record the version of each SSAS server that will be required in order to support the business requirements to connect to other systems.

## Task 3: Decide Whether Scalable Shared Databases Will Be Used

Scalable shared databases can apply to both the relational database and the multi-dimensional databases (cubes). These can be used to scale out querying or reporting loads. With scalable shared OLAP databases, a read-only copy of the database on a shared drive is shared between several SSAS servers.

The benefits of using shared databases include an increase in the number of processors available and more memory for processing queries; however, proper disk configuration is important for the disk containing the database.

The restrictions of using shared databases include:

* The database must be on a read-only volume.
* The databases are supported by Windows Storage running only on Windows Server 2003 SP1 or later.
* Scalable shared databases do not support database snapshots.

Determine whether scalable shared databases will be used and record this in Table A-5 in Appendix A.

Refer to the “Additional Reading” section to find links to white papers on scalable shared databases.

## Task 4: Determine Scaling Needs

If cube processing affects query performance or if processing can’t occur during times of reduced query load, consider moving processing tasks to a staging server and then performing an online synchronization of the production server and the staging server. For more information, see “Synchronizing Analysis Services Databases” in SQL Server Books Online at <http://msdn.microsoft.com/en-us/library/ms174928.aspx>.

Processing can also be distributed across multiple instances of Analysis Services by using remote partitions. Processing remote partitions uses the processor and memory resources on the remote server instead of the resources on the local computer. For information on remote partitions management, see “Managing Analysis Services Partitions” in SQL Server Books Online at <http://msdn.microsoft.com/en-us/library/ms175604.aspx>.

Determine whether to scale up or scale out for SSAS and record this information in Table A-5 in Appendix A.

## Task 5: Decide Whether to Cluster

SSAS natively supports failover clusters (formerly known as server clusters or as MSCS) in Windows Server. Failover clustering enables recovery from a service failure in the shortest time possible. However, it is important to note the following considerations:

* When an instance fails over to the next available node, all connections are lost, with the corresponding loss of uncommitted work. Most client applications should be able to handle this situation. Often, performing a refresh of the application will restore the results.
* Administrative applications that process Analysis Services objects should be designed to handle failover situations.
* SQL Server 2008 Enterprise enables up to 16 nodes in the failover clusters, while SQL Server 2008 Standard is limited to two nodes.

Record in Table A-5 in Appendix A whether clustering will be used for the SSAS server.

## Task 6: Determine Placement

Factors such as politics, policy, network constraints, proximity to data sources, and geography can determine where to place the SSAS servers. In this task, it will be determined where to place each SSAS server.

SSAS can be run in a virtualized environment if the memory, disk, and network requirements do not exceed the throughput capabilities of the virtual machine.

SQL Server roles can coexist with other SQL Server roles on the same server. Decide whether the SSAS role will coexist with other roles on an existing server or if it will reside on a dedicated server.

Record in Table A-5 in Appendix A whether SSAS will be placed in a virtualized or physical environment and whether the SQL Server role will coexist with other SQL Server roles on the same server or reside on a dedicated server.

This task should be repeated for each additional instance required for SSAS.

## Step Summary

In Step 5, the SSAS infrastructure was designed if it was determined that SSAS is required in the organization. The resource requirements, the SQL Server version, whether scalable shared databases will be used, and the scaling needs were determined. Additionally, whether to cluster and where to place the SSAS server and roles were determined. If it was determined that SSAS is not required, the reader was advised to go to the next step.

The following Step 5 outputs were recorded in Table A-5 in Appendix A:

* The resource requirements.
* The SQL Server version.
* Whether scalable shared databases will be used, the scaling requirements, and whether clustering will be used to support fault tolerance.
* The SSAS placement and whether the SQL Server role will coexist with other SQL Server roles on the same server or reside on a dedicated server.

## Additional Reading

* “Server Clusters”: <http://technet.microsoft.com/en-us/library/cc783714.aspx>
* “Getting Started with SQL Server 2008 Failover Clustering”: <http://msdn.microsoft.com/en-us/library/ms189134.aspx>
* “Requirements and Considerations for Analysis Services Deployment”: <http://msdn.microsoft.com/en-us/library/ms175672.aspx>
* “Scaling out an Analysis Services Solution”: <http://technet.microsoft.com/en-us/library/cc280669.aspx>
* “Technical Notes: Running Microsoft SQL Server 2008 Analysis Services on Windows Server 2008 vs. Windows Server 2003 and Memory Pre-allocation: Lessons Learned”: <http://sqlcat.com/technicalnotes/archive/2008/07/16/running-microsoft-sql-server-2008-analysis-services-on-windows-server-2008-vs-windows-server-2003-and-memory-preallocation-lessons-learned.aspx>

# Step 6: Design the SQL Server Reporting Services Infrastructure

In Step 2, the roles that need to be implemented were determined. If it was determined that SQL Server Reporting Services (SSRS) is required in the organization, the SSRS infrastructure will be designed in this step. If it was determined that SSRS is not required, go to the next step.

## Task 1: Determine Resource Requirements

In this task, the performance requirements will be assessed. No hard guidance is available for determining the processor, disk performance, or networking requirements for a typical scenario as they are all different, but there are a few factors that can be taken into consideration:

* Size and complexity of report definitions. Complex reports that process large numbers of rows require significantly more resources than a simple report that processes a few rows.
* Data source. This refers to whether reports are executed from cached or snapshot data or from live data. Reports that are executed by using cached or snapshot data can consume significantly fewer resources than reports that are run with live data.   
  In addition, using cached reports reduces the load placed on the source system from which the data is queried.
* Format requested when rendering a report. Formats such as PDF or Excel are more resource-intensive than HTML. Automated emailing of reports containing large-image format reports can negatively affect network performance.

The following sections provide information that will assist with determining resource requirements.

### Disk Storage Requirements

A report server database, or catalog, provides internal storage to one or more report servers. Each SSRS server must connect to two SQL Server databases, named ReportServer and ReportServerTempDB by default. The ReportServer database stores the report definitions, resources, and configurations, while the ReportServerTempDB database stores all of the temporary snapshots while reports are running.

#### ReportServer Database

Generally, a moderately sized report definition will take about 100–200 KB of disk space, which is larger than the actual size of the RDL (Report Definition Language) file where the specifications of the report are stored. This is because Reporting Services will persist both the RDL and the compiled binary representation in order to improve report execution performance. In the case of history snapshot sizes, this is a function of the amount of data that is in the report (that is, the larger the report and datasets, the larger the snapshot). A general guideline is that Reporting Services has an approximate 5:1 compression ratio; therefore, if a report has 10 MB of data, then the snapshot will be about 2 MB in size. Note that reports that do not perform aggregations will be slightly larger than reports that do.

#### ReportServerTempDB Database

The size of this database varies depending on the number of users who are concurrently using the report servers. Each live report execution generates a report snapshot that is persisted in the ReportServerTempDB for the duration of the user’s session. Because of the high degree of variability in the number of users who are accessing the report servers at any one time, the sizing estimate should be based on the maximum number of concurrent users that access the Reporting Services environment. A general guideline here is that there will typically be a maximum of 10–20 percent concurrency of the user base.

For example, if there are 1,000 users, then expect up to 200 concurrent users. If most of these users are accessing a 10-MB report, then there will need to be at least 400 MB of storage provided when taking compression into account. Of course, when users are no longer querying for reports and/or report sessions are timing out, the space will be reclaimed and be made available for new users. But at the same time, to size properly, the calculation must be based on the maximum number of concurrent users.

Record the estimated size of the ReportServer and ReportServerTempDB databases in Table A-6 in Appendix A.

### Memory Requirements

The product group recommends 2–4 GB of memory per processor core.

## Task 2: Determine Placement of the Report Server Databases

The report server databases can be hosted either on the Reporting Services server or on a remote database server.

If the report server databases are hosted locally, the SQL Server relational database will compete with the report server for available machine resources.

Hosting the report server databases on a remote database server eliminates competition for machine resources between the report server databases and the SQL Server-based server hosting the database. However, adequate network bandwidth must be provided between the report server databases and the database server.

For installations that are being planned for scalability, the remote database implementation is the first step toward a scale-out configuration.

Record in Table A-6 in Appendix A whether the databases will be hosted locally or on a remote database server and, if so, which server. Repeat Step 3 to design the database structure.

## Task 3: Determine the Scaling and Fault-Tolerance Approach

Scale-out deployments are used to increase scalability of report servers in order to handle additional concurrent users and larger report execution loads, and placing a network load balancer (NLB) in front of the report servers can add fault tolerance. Scale-out deployments can also be used to dedicate specific servers to process interactive or scheduled reports.

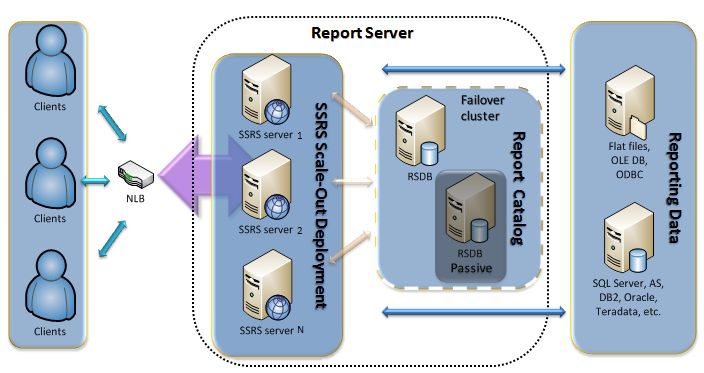


Figure 9. An example Reporting Services scale-out architecture

Although moving to a scale-out configuration requires a move to the Enterprise edition of Reporting Services, doing so has a number of advantages in addition to raw capacity. These advantages include the following:

* Multiple report servers offer better availability; if a single server fails, the overall system continues to answer queries.
* Additional machines can take advantage of dedicated memory address space, without having to move to 64-bit hardware.

Note that in a scale-out implementation, all servers must have the same type of authentication extension. Because of this restriction, a scale-out deployment cannot support simultaneous Intranet and extranet access to the same report server database through separate report server instances, where each instance is configured for different authentication types and authorization rules.

While SSRS is not supported in an MSCS cluster configuration, the server hosting the report server database can be set up in a failover cluster configuration.

Determine the number of servers required to support the reporting users, and record this information in Table A-6 in Appendix A.

## Task 4: Determine Placement of the SSRS Server

For smaller environments, the SSRS server can be implemented on the same system as other SQL Server services. However, note that SSRS does not support clustering as the Database Engine role and SSAS do.

SSRS is supported in a virtualized environment if the memory, disk, and network requirements do not exceed the throughput capabilities of the virtual machine host.

For each SSRS server, determine whether to have the SSRS server coexist with other SQL Server services or by itself. Also, determine whether the SSRS will be in a physical environment or exist in a virtualized environment. Record these decisions in Table A-6 in Appendix A.

## Step Summary

In Step 6, the SSRS infrastructure was designed if it had been determined in Step 2 that SSRS is required in the organization. The resource requirements and placement of the report server databases were determined. Additionally, the scaling and fault-tolerance approach were determined, as well as the placement of the SSRS server.

The following Step 6 outputs were recorded in Table A-6 in Appendix A:

* The database requirements (size of the ReportServer and ReportServerTempDB databases), and whether the databases will be hosted locally or on a remote database server and, if so, which ones.
* The number of servers to support SSRS.
* Whether to have the SSRS SQL Server-based server coexist with other SQL Server roles on the same server, and whether to place the SSRS server in a virtualized or physical environment.

## Additional Reading

* “Reporting Services Performance Optimizations”: <http://sqlcat.com/technicalnotes/archive/2009/01/14/reporting-services-performance-optimizations.aspx>
* “Report and Snapshot Size Limits”: <http://msdn.microsoft.com/en-us/library/ms156002.aspx>
* “Estimating Report Server Database Requirements”: <http://msdn.microsoft.com/en-us/library/ms159758.aspx>
* “Planning for Scalability and Performance with Reporting Services”: <http://www.microsoft.com/technet/prodtechnol/sql/2005/pspsqlrs.mspx>

# Step 7: Design the SQL Server Master Data Services Infrastructure

If it was determined in Step 2 that the SQL Server Master Data Services (MDS) will be required in the organization, its infrastructure will be designed in this step. If Master Data Services is not required, the SQL Server infrastructure planning is complete.

Master Data Services includes the following server-side components:

* A web server, composed of:
* Master Data Manager, an ASP.NET web application used to manage the master data and perform other administrative tasks.
* Optionally, the MDS web service, a service hosted in IIS in which developers can use to programmatically extend or develop custom solutions for MDS in their environments.
* An MDS database, which contains all of the information for the MDS system, including:
* Settings, database objects, and data required by the MDS system.
* Staging tables that are used to process data from source systems.
* A schema and database objects to store master data from source systems.
* Versioning functionality, including business rule validation and email notifications.
* Views for subscribing systems that need to retrieve data from the database.

**Note**   Master Data Services is a feature of the versions and editions of SQL Server listed in Setup Requirements (Master Data Services) at <http://msdn.microsoft.com/en-us/library/ee633742(v=SQL.105).aspx>. Any computer where you install Master Data Services must be licensed accordingly. For more information, refer to the End User License Agreement (EULA).

The web and database components can be installed together on a single server or on separate servers. The following diagram illustrates the communication pathway for Master Data Services.

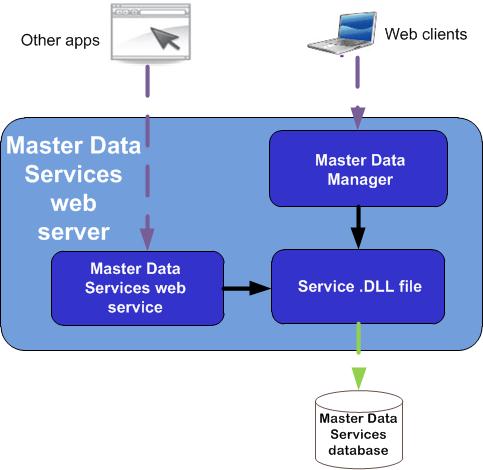


Figure 10. Master Data Services architecture

## Task 1: Determine Resource Requirements

In this task, the performance requirements will be assessed. No hard guidance is available for determining the processor, disk performance, or networking requirements for a typical scenario as they are all different, but there are a few factors that can be taken into consideration:

* The amount of Master Data Services data that will be tracked. Organizations will often implement master data management to solve the business problem in a single domain area, and then expand the system from there. The SQL Server Master Data Services product group has tested the system up to 10 million unique members.
* How often the data will be versioned. An accounting domain area may only need to be updated once per month, but a product domain area could require more frequent updates.

### Master Data Services Database

Master Data Services requires a database to support the Master Data Manager web application and web services. The computer that hosts the Master Data Services database must run an instance of the SQL Server Database Engine. The product group reports that even a fairly large implementation only resulted in a database that was 2 GB in size as Master Data Services is not transactional in nature; therefore, the disk space requirements for the database are not necessarily significant.

### Master Data Services Web Server

Master Data Services will have different categories of users, such as system administrators, data architects, data stewards, data editors, and data consumers. Depending on the implementation, the data consumers may not be directly interacting with the system, so they would not count as concurrent users. A rough estimate provided by the product group is that a 10,000-employee company may have 50–100 MDS users.

Determine the number of concurrent users that will be accessing the system. This information will be used to estimate the load on the web server.

Record the estimated size of the Master Data Services database and number of concurrent users in Table A-7 in Appendix A.

## Task 2: Determine the Scaling and Fault-Tolerance Approach

The Master Data Services web server can be placed in a load-balanced web farm to provide scaling and/or fault tolerance. The deployment is based on the feature provided by the underlying IIS, such as hardware load balancers, software load balancing via NLB, or Application Request Routing (ARR).

Multiple web servers in a load-balanced farm may be required for the following functions:

* **Scale out.** As a rough estimate, the product group suggests that a single web server can support approximately 100 concurrent users. Refer to the number of users determined in Task 1 to decide the number of web servers required in the deployment.
* **Fault tolerance.** The smallest implementation that will provide fault tolerance is two web servers with either hardware or software load balancing. Implementing a Master Data Services web farm using ARR requires at least three IIS web servers. One of these servers will be the IIS ARR server and the remaining servers will serve as Master Data Services web application servers. More information on this deployment configuration can be found at <http://blogs.msdn.com/mds/archive/2010/02/10/configuring-a-mds-load-balanced-web-farm-using-iis-v7-arr.aspx>.

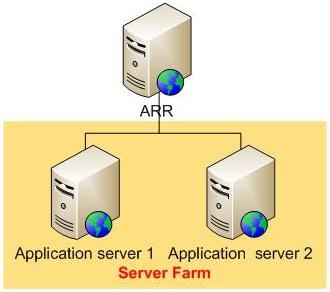


Figure 11. An example of scale-out for the Master Data Services web farm using IIS’s ARR

A web farm configuration will require a centrally located Master Data Services database that all web servers are configured to use.

**Note**The IIS product group recommends NLB as the primary and preferred method for improving the scalability and availability of web applications running IIS, as opposed to failover clustering. For more information, see “Configuring IIS 7.0 World Wide Web Publishing Service in a Microsoft Windows Server 2008 failover cluster” at <http://support.microsoft.com/kb/970759/>.

Determine the number of servers that will be required to support the fault tolerance and scaling requirements, and record this information in Table A-7 in Appendix A. Then consult the *Infrastructure Planning and Design Guide for Internet Information Services* at <http://go.microsoft.com/fwlink/?LinkId=157703> to design the Master Data Services web server infrastructure.

## Task 3: Determine Placement of the Master Data Services Database

The Master Data Services database can be hosted either on the Master Data Services web server or on a remote database server that runs a supported version and edition of SQL Server. If the Master Data Services database is hosted locally on the web server, adequate hardware resources should be allocated to support both database and web operations. It can also coexist with other databases and SQL Server services and is supported in a virtualized environment.

Reasons for hosting the Master Data Services database on a remote server include:

* To conform with the organization’s IT or security policies. The Master Data Services web service and Master Data Services web application both run on top of IIS, so the organization’s policies may dictate using a separate database server rather than co-locating with IIS.
* To support an implementation of the Master Data Services web server in a load-balanced web farm for fault tolerance (determined in Task 2).

Any of the fault tolerance options for a SQL Server database server described in Step 3 of this guide are valid for the Master Data Services database.

Record in Table A-7 in Appendix A whether the databases will be hosted locally or on a remote database server and, if so, which server. Repeat Step 3 to design the database structure.

## Task 4: Determine the Placement of the Master Data Services Web Server

According to the Product Group, for smaller environments, the Master Data Services web server can be implemented on the same system as other SQL Server services. The Master Data Services web server is supported in a virtualized environment if the memory, disk, and network requirements do not exceed the throughput capabilities of the virtual machine host.

For each Master Data Services web server, determine whether to have the server coexist with other SQL Server services or by itself. Also, determine whether the server will be in a physical environment or exist in a virtualized environment.

There are no requirements for physical placement of the web server as long as there is sufficient throughput to the database. There are also no hardware requirements specific to Master Data Services other than what is required for IIS, which is outlined here for Windows Server 2008 (<http://msdn.microsoft.com/en-us/windowsserver/cc196364.aspx>) and here for Windows Server 2008 R2 (<http://www.microsoft.com/windowsserver2008/en/us/system-requirements.aspx>).

Information on the roles, role services, and features from IIS, .NET Framework, and PowerShell that are required for MDS are documented in <http://msdn.microsoft.com/en-us/library/ee633744(v=SQL.105).aspx>.

Record these decisions in Table A-7 in Appendix A.

## Step Summary

In Step 7, the Master Data Services infrastructure was designed. The resource requirements were assessed, and then the fault tolerance and scaling requirements were planned. Based partly on these decisions, the database was placed either on the same server as the Master Data Services web service or on a remote database server.

The following Step 7 outputs were recorded in Table A-7 in Appendix A:

* The database requirements (size of the Master Data Services database), and whether the databases will be hosted locally or on a remote database server and, if so, which ones.
* The number of servers to support the Master Data Services web service.
* Whether to have the Master Data Services web Service server coexist with other SQL Server roles on the same server, and whether to place the servers in a virtualized or physical environment.

## Additional Reading

* “Master Data Services Team: Configuring a Master Data Services load-balanced web farm using IIS v7 ARR”: <http://sqlblog.com/blogs/mds_team/archive/2010/02/05/configuring-a-mds-load-balanced-web-farm-using-iis-v7-arr.aspx>
* “Microsoft Windows Server 2008 System Requirements”: <http://msdn.microsoft.com/en-us/windowsserver/cc196364.aspx>
* “Windows Server 2008 R2: System Requirements”: <http://www.microsoft.com/windowsserver2008/en/us/system-requirements.aspx>
* “The What, Why, and How of Master Data Management”: <http://msdn.microsoft.com/en-us/library/bb190163.aspx>
* “Master Data Services”: <http://msdn.microsoft.com/en-us/library/ee633763(SQL.105).aspx>
* “Organizational Approaches to Master Data Management”: <http://msdn.microsoft.com/en-us/library/ff626496(v=SQL.100).aspx>
* “SharePoint Workflow Integration with Master Data Services”: <http://msdn.microsoft.com/en-us/library/ff459274(v=SQL.100).aspx>

# Conclusion

This guide has outlined the step-by-step process for planning a SQL Server 2008 infrastructure. In each step, major decisions relative to the SQL Server infrastructure were determined and explained. The guide has explained how to record choices of roles needed, server resources, scaling, and fault tolerance, which can then be made available to the infrastructure planners. In addition, updated features available in SQL Server 2008 R2 were discussed.

Using the information recorded from the steps completed in this guide, the organization can help ensure that they meet business and technical requirements for a successful SQL Server 2008 deployment.

## Additional Reading

* SQL Server 2008 Upgrade Technical Reference Guide: <http://www.microsoft.com/downloads/details.aspx?familyid=66D3E6F5-6902-4FDD-AF75-9975AEA5BEA7&displaylang=en>
* SQL Server Customer Advisory Team page: [www.sqlcat.com](http://www.sqlcat.com)
* Microsoft SQL Server page: <http://www.microsoft.com/sqlserver/2008/en/us/default.aspx>

## Feedback

Please direct questions and comments about this guide to [satfdbk@microsoft.com](http://www.microsoft.com/IPD).

We value your feedback on the usefulness of this guide. Please complete the following **Solution Accelerators Satisfaction Survey**, available at [http://go.microsoft.com/fwlink/?LinkID=132579](http://www.microsoft.com/IPD), and help us build better guidance and tools.

# Appendix A: Job Aids

This section provides job aid examples to record data while planning the SQL Server 2008 infrastructure.

**Step 1.** Use the table below to record business applications in the project scope.

Table A-1. Requirements Gathering

|  |  |
| --- | --- |
| Requirement description | Requirement results |
| Name of business applications |  |

**Step 2.** Use the table below to record information relative to the required SQL Server roles.

Table A-2. SQL Server Roles Data Gathering

| **Requirement description** | **Requirement results** |
| --- | --- |
| Names of relational or OLTP databases required |  |
| Names of data warehouse databases required |  |
| SSIS required? |  |
| SSAS required? |  |
| SSRS required? \* requires Database Engine also |  |
| Master Data Services required? \* requires Database Engine also |  |

Step 3. Use the table below to record information relative to the SQL Server Database Engine.

Table A-3. SQL Server Database Engine Data Gathering

| Requirement description | Requirement results |
| --- | --- |
| Disk storage requirements |  |
| IOPS and throughput requirements |  |
| Database in new or existing instance? |  |
| Will instance reside on existing server running SQL Server or new server? |  |
| Number of servers required to support scale-out options, if selected |  |
| Number of servers required to support fault-tolerance option, if selected |  |
| Failover clustering protection required? |  |
| In which office or geographical location will instance be placed? |  |
| Will instance be on physical server or virtualized environment? |  |
| Number of CPUs required |  |
| Required architecture |  |
| Required processor speed |  |
| Amount of memory required |  |
| Disk subsystem configuration |  |
| Number of network adapters |  |

Step 4. Use the table below to record information relative to the SSIS infrastructure.

Table A-4. SQL Server SSIS Infrastructure Data Gathering

| **Requirement description** | **Requirement results** |
| --- | --- |
| Does each component of the system meet the capacity and performance requirements? |  |
| Source data type |  |
| Destination data type |  |
| SSIS server version |  |
| Name of instance of SQL Server or file share where packages are stored |  |
| Database Engine required? |  |
| Number of SSIS servers |  |
| SSIS server: virtualized or physical environment? |  |
| SSIS - SQL Server role coexist with other SQL Server roles on same server? |  |

Step 5. Use the table below to record information relative to the SSAS infrastructure.

Table A-5. SQL Server SSAS Infrastructure Data Gathering

| **Requirement description** | **Requirement results** |
| --- | --- |
| Disk storage space requirements |  |
| SSAS server version |  |
| Will scalable shared databases be used? |  |
| What are scaling needs? (scale up or scale out) |  |
| Will failover clustering be used? |  |
| SSAS server: virtualized or physical environment? |  |
| SSAS - SQL Server role coexist with other SQL Server roles on same server? |  |

Step 6. Use the table below to record information relative to the SSRS infrastructure.

Table A-6. SQL Server SSRS Infrastructure Data Gathering

| **Requirement description** | **Requirement results** |
| --- | --- |
| Record database requirements (size of the ReportServer and ReportServerTempDB databases) |  |
| Databases hosted locally or on a remote database server? |  |
| Which databases are hosted locally or on a remote server? |  |
| Number of servers to support SSRS |  |
| SSRS – SQL Server coexist with other SQL Server roles on same server? |  |
| SSRS server: virtualized or physical environment? |  |

Step 7. Use the table below to record information relative to the Master Data Services infrastructure.

Table A-7. SQL Server Master Data Services Infrastructure Data Gathering

| **Requirement description** | **Requirement results** |
| --- | --- |
| Estimated Master Data Services database size |  |
| Number of concurrent users |  |
| Number of servers to support fault-tolerance and scaling requirements |  |
| Databases hosted locally or on a remote database server? |  |
| Which databases are hosted locally or on a remote server? |  |
| Master Data Services web server coexist with other SQL Server roles on same server? |  |
| Master Data Services web server: virtualized or physical environment? |  |

# Appendix B: Server Performance Analyzing and Scaling

The following information identifies important monitoring counters used for capacity planning and performance monitoring of a system.

## Processor Utilization

Over-committing CPU resources can adversely affect all the workloads on the same server, causing significant performance issues for a larger number of users. Because CPU resource use patterns can vary significantly, no single metric or unit can quantify total resource requirements. At the highest level, measurements can be taken to see how the processor is utilized within the entire system and whether threads are being delayed. The following table lists the performance counters for capturing the overall average processor utilization and the number of threads waiting in the processor Ready Queue over the measurement interval.

Table B-1. Performance Monitor Counters for Processor Utilization

|  |  |  |
| --- | --- | --- |
| Object | Counter | Instance |
| Processor | % Processor Time | \_Total |
| System | Processor Queue Length | N/A |

### Processor\% Processor Time

As a general rule, processors that are running for sustained periods at greater than 90 percent busy are running at their CPU capacity limits. Processors running regularly in the 75–90 percent range are near their capacity constraints and should be closely monitored. Processors regularly reporting 20 percent or less utilization can make good candidates for consolidation.

For response-oriented workloads, sustained periods of utilization above 80 percent should be investigated closely as this can affect the responsiveness of the system. For throughput-oriented workloads, extended periods of high utilization are seldom a concern, except as a capacity constraint.

Unique hardware factors in multiprocessor configurations and the use of Hyper-threaded logical processors raise difficult interpretation issues that are beyond the scope of this document. Additionally, comparing results between 32-bit and 64-bit versions of the processor are not as straightforward as comparing performance characteristics across like hardware and processor families. A discussion of these topics can be found in Chapter 6, “Advanced Performance Topics” in the *Microsoft Windows Server 2003 Performance Guide*.

### System\Processor Queue Length

The Processor Queue Length can be used to identify if processor contention, or high cpu-utilization, is caused by the processor capacity being insufficient to handle the workloads assigned to it. The Processor Queue Length shows the number of threads that are delayed in the processor Ready Queue and are waiting to be scheduled for execution. The value listed is the last observed value at the time the measurement was taken.

On a machine with a single processor, observations where the queue length is greater than 5 is a warning sign that there is frequently more work available than the processor can handle readily. When this number is greater than 10, then it is an extremely strong indicator that the processor is at capacity, particularly when coupled with high CPU utilization.

On systems with multiprocessors, divide the queue length by the number of physical processors. On a multiprocessor system configured using hard processor affinity (that is, processes are assigned to specific CPU cores), which have large values for the queue length, can indicate that the configuration is unbalanced.

Although Processor Queue Length typically is not used for capacity planning, it can be used to identify if systems within the environment are truly capable of running the loads or if additional processors or faster processors should be purchased for future servers.

## Memory Utilization

In order to sufficiently cover memory utilization on a server, both physical and virtual memory usage needs to be monitored. Low memory conditions can lead to performance problems, such as excessive paging when physical memory is low, to catastrophic failures, such as widespread application failures or system crashes when virtual memory becomes exhausted.

Table B-2. Performance Monitor Counters for Memory Utilization

|  |  |  |
| --- | --- | --- |
| Object | Counter | Instance |
| Memory | Pages/sec | N/A |
| Memory | Available Mbytes | N/A |
| Memory | Pool Paged Bytes | N/A |
| Memory | Pool Paged Resident Bytes | N/A |
| Memory | Transition Faults/sec | N/A |
| Memory | Committed Bytes | N/A |
| Process | Working Set | <Process Name> |

### Memory\Pages/sec

As physical RAM becomes scarce, the virtual memory manager will free up RAM by transferring the information in a memory page to a cache on the disk. Excessive paging to disk might consume too much of the available disk bandwidth and slow down applications attempting to access their files on the same disk or disks. The Pages/Sec counter tracks the total paging rates, both read and writes, to disk.

For capacity planning, watch for upward trends in this counter. Excessive paging can usually be reduced by adding additional memory. Add memory when paging operations absorbs more than 20–50 percent of the total disk I/O bandwidth. Because disk bandwidth is finite, capacity used for paging operations is unavailable for application-oriented file operations. The Total Disk I/O Bandwidth is a ratio of the Pages/sec and the Physical Disk\Disk Transfers/sec for all disks in the system:

Memory\Pages/sec ÷ Physical Disk (\_Total)\Disk Transfers/sec

### Memory\Available Mbytes

The Available Mbytes displays the amount of physical memory, in megabytes, that is immediately available for allocation to a process or for system use. The percent Available Megabytes can be used to indicate if additional memory is required. Add memory if this value drops consistently below 10 percent. To calculate the percent of Available Megabytes:

(Memory\Available Mbytes ÷ System RAM in Megabytes) \* 100

This is the primary indicator to determine whether the supply of physical memory is ample. When memory is scarce, Pages/sec is a better indicator of memory contention. Downward trends can indicate a need for additional memory. Counters are available for Available Bytes and Available Kbytes.

### Memory\Pool Paged Bytes and Memory\Pool Paged Resident Bytes

The Pool Paged Bytes is the size, in bytes, of the paged pool, an area of system memory used by the operating system for objects that can be written to disk when they are not being used.

The Pool Paged Resident Bytes is the size, in bytes, of the nonpaged pool, an area of system memory used by the operating system for objects that can never be written to disk, but must remain in physical memory as long as they are allocated.

A ratio of Pool Paged Bytes to Pool Paged Resident Bytes can be calculated by:

Memory\Pool Paged Bytes ÷ Memory\Pool Paged Resident Bytes

This ratio can be used as a memory contention index to help in planning capacity. As this approaches zero, additional memory needs to be added to the system to allow both the Nonpaged pool and Page pool to grow.

The size returned by the Pool Paged Resident Bytes can be used for planning additional TCP connections. Status information for each TCP connection is stored in the Nonpaged pool. By adding memory, additional space can be allocated to the Nonpaged pool to handle additional TCP connections.

### Memory\Transition Faults/sec

The Transition Faults counter returns the number of soft or transition faults during the sampling interval. Transition faults occur when a trimmed page on the Standby list is re-referenced. The page is then returned to the working set. It is important to note that the page was never saved to disk.

An upward trend is an indicator that there may be a developing memory shortage. High rates of transition faults on their own do not indicate a performance problem. However, if the Available Megabytes is at or near its minimum threshold value, usually 10 percent, then it indicates that the operating system has to work to maintain a pool of available pages.

### Memory\Committed Bytes

The Committed Bytes measures the amount of committed virtual memory. Committed memory is allocated memory that the system must reserve space for either in physical RAM or on the paging file so that this memory can be addressed by threads running in the associated process.

A memory contention index called the Committed Bytes:RAM can be calculated to aid in capacity planning and performance. When the ratio is greater than 1, virtual memory exceeds the size of RAM and some memory management will be necessary. As the Committed Bytes:RAM ratio grows above 1.5, paging to disk will usually increase up to a limit imposed by the bandwidth of the paging disks. Memory should be added when the ratio exceeds 1.5. The Committed Bytes:RAM is calculated by:

Memory\Committed Bytes ÷ System RAM in Bytes

### Process\Working Set

The Working Set counter shows the amount of memory allocated to a given process that can be addressed without causing a page fault to occur. To see how much RAM is allocated overall across all process address spaces, use the \_Total instance of Working Set. Watch for upward trends for important applications.

Some server applications, such as IIS, Exchange Server, and SQL Server, manage their own process working sets. To measure their working sets, application-specific counters need to be used.

## Disk Storage Requirements

The process of planning for storage requirements is divided into capacity requirements and disk performance. Although a total capacity requirement can be determined, the performance requirements, as well as fault tolerance requirements, of the system will have an impact on the implementation of the storage subsystem. For example, a single drive could provide enough storage space, but the performance of that single disk may not meet the performance needs of the system.

Due to this, both capacity and performance requirements need to be met, which may alter the decision around the size, speed, and configuration of the drives in the storage subsystem.

Using an I/O stress tool to validate performance and ensure that the system is tuned optimally for SQL Server before deployment helps identify hardware or I/O configuration related issues before the complexity of SQL Server is introduced. SQLIO is a tool provided by Microsoft which can also be used to determine the I/O capacity of a given configuration. A minimum target for SQL Enterprise edition SQLIO is 400 megabytes per second throughput on 64k block size. If the results are lower than that, the SQL Server database will likely display disk I/O issues.

SQLIOSim can be used to validate the basic functionality of an I/O subsystem under stress by simulating SQL Server I/O patterns.

### Disk Space Capacity

The amount of storage capacity required can be calculated based on OS requirements as well as any application-specific data that needs to be stored on the system.

### Disk Performance

Disk performance is typically expressed as a value of the total number of I/O operations that occur per second (IOPS), measured over some period of time during peak usage.

To determine the number of disks needed to meet a system’s IOPS requirement, the IOPS of a given drive needs to be determined. To further complicate matters, IOPS are very dependent upon the access pattern. For example, a drive will typically have a higher IOPS rating for sequential reads than it will for random writes. For this reason, it is normal to calculate a “worst case” IOPS measurement based on short random input/output operations.

To calculate the IOPS of a drive, information about the drive needs to be collected. The following table lists the information required that normally can be found in the manufacturer’s datasheet about the drive.

Table B-3. Information Required for Calculating IOPS

|  |  |
| --- | --- |
| Required Information | Description |
| Spindle Rotational Speed (RPM) | The spindle speed expressed as RPM. |
| Average Read Seek Time (ms) | The average seek time for reads. |
| Average Write Seek Time (ms) | The average seek time for writes. |

The first step to calculating the IOPS is to determine the Average Seek Time in milliseconds that the drive is capable of doing. There is an assumption that there will be a 50/50 mix of read and write operations. If the ratio of read and write operations is modified, then the Average Seek Time will need to be adjusted. For example, if a drive has an Average Read of 4.7 ms and an Average Write of 5.3 ms, the Average Seek Time for this drive will be 5.0 ms:

5.0ms = (4.7ms + 5.3ms) ÷ 2

Next, the IO Latency needs to be calculated. This is calculated by adding the Average Seek Time to the Average Latency. The following table lists the Average Latency of common spindle speeds of drives on the market today.

Table B-4. Average Latency Based on Spindle Rotational Speeds

|  |  |
| --- | --- |
| Spindle Rotational Speed (rpm) | Average Latency (ms) |
| 4,200 | 7.2 |
| 5,400 | 5.6 |
| 7,200 | 4.2 |
| 10,000 | 3.0 |
| 15,000 | 2.0 |

The example drive has a spindle speed of 10,000 RPM. So this drive has an IO Latency of 8.0 ms:

8.0 ms = 5.0ms + 3.0ms

A drive can only perform one IO operation at a time. To calculate the number of IOs that can be performed in a millisecond, 1 is divided by the IO Latency. Finally, this value is converted to IO per Second by multiplying by 1000. The IOPS calculation for the example drive evaluates to 125 IOPS:

125 IOPS = (1 IO ÷ 8.0ms) \* 1000 ms/sec

## Storage Requirements

To determine storage requirements, additional information is needed to be collected around the system being considered. Some of this information is easily identified and self explanatory, while other information may be more difficult to identify due to lack of quantifiable data. All of the following calculations are for a single server; although if shared storage systems are being considered, then the information can be scaled up based on the number of systems sharing that storage. The following table shows the information that needs to be collected.

Table B-5. Information Required for Calculating Storage Requirements

| Required Information | Description | Example |
| --- | --- | --- |
| # Users Per Server | Total number of users hosted by that server. | 700 |
| % Concurrent Users | The percentage of users connected to the server during peak times. | 80% |
| IOPS per User Required | The number of IOPS generated by a user. | 0.5 |
| Storage Capacity in Gigabytes | The calculated disk storage capacity needed. | 450 |
| % Buffer Factor (for growth) | The percentage of disk storage growth allowed by the system. | 20% |
| Read % of IOPS | The percentage of IOPS that are Read operations. | 50% |
| Write % of IOPS | The percentage of IOPS that are Write operations. | 50% |
| Disk Size (GB) | The drive size being considered for the storage system. | 146 |
| Calculated Drive IOPS | The calculated IOPS of the drive being considered for the storage system. | 125 |

The information in the table above is fairly self explanatory with the exception of the IOPS per User Required. This is a measurement of the number of IOPS that a single user will generate on the system. Most venders do not have this information for their applications unless the application is extremely IO intensive. This information may be calculated from observation of a running system, but it is fraught with a number of challenges and is beyond the scope of this guide. For the purpose of this example, this guide will use the value used with Exchange Server, which is 0.5 IOPS per user.

Based on the information in Table B-5, there are a number of derived values that need to be calculated. The following table lists these values.

Table B-6. Derived Information Required for Calculating Storage Requirements

|  |  |  |
| --- | --- | --- |
| Required Information | Description | Example |
| # of Concurrent Users | The number of concurrent users calculated by calculating the percentage of concurrent users and the number of users per server. | 560 |
| IOPS per Server Required | The number of IOPS generated by each user multiplied by the number of concurrent users. | 280 |
| Total Storage Requirements | The Storage Capacity increased by the percentage of growth allowed. | 540 |
| Number of Read IOPS | The percentage of Reads and the IOPS per Server Required. | 140 |
| Number of Write IOPS | The percentage of Reads and the IOPS per Server Required. | 140 |
| Drive Size Actual (GB) | After formatting, drives capacity is roughly 10 percent less than the advertised total capacity of the disk. This value adjusts for this loss of space. This value is calculated by taking 90 percent of the Disk Size (GB). | 132 |

RAID 0+1 and RAID 5 are the most common drive configurations for storing data in a redundant manner on a system. However, these two RAID systems have different IOPS calculations due to how they operate.

### RAID 0+1 Calculations

To calculate the number of drives necessary to meet the storage requirements with RAID 0+1, divide the Total Storage Requirements by the Drive Size Actual. Round the result up and multiply by 2. For this example, 10 drives are needed for RAID 0+1 to meet the storage requirements:

10 = ROUNDUP(540÷132)\*2

To calculate the number of drives required to meet the performance requirements with RAID 0+1, multiply the Number of Write IOPS by 2 and add the Number of Read IOPS. Divide this total by the Calculated Drive IOPS and round the result up. For this example, 4 drives are needed for RAID 0+1 to meet the performance requirements:

4 = ROUNDUP(((140\*2)+140)÷125)

Although RAID 0+1 can meet the performance requirements with just 4 disks, 10 disks are required to meet the capacity requirements.

### RAID 5 Calculations

To calculate the number of drives necessary to meet the storage requirements with RAID 5, the Total Storage Requirements needs to be multiplied by 1.2 to adjust for parity storage requirements. This value is then divided by the Drive Actual Size and then rounded up. For this example, 5 drives are needed for RAID 5 to meet the storage requirements:

5 = ROUNDUP((540\*1.2)÷132)

To calculate the number of drives required to meet the performance requirements with RAID 5, multiply the Number of Write IOPS by 4 and add the Number of Read IOPS. Divide this total by the Calculated Drive IOPS and round the result up. For this example, 6 drives are needed for RAID 5 to meet the performance requirements:

6 = ROUNDUP(((140\*4)+140)÷125)

Although RAID 5 can meet the storage requirements with just 5 disks, 6 disks are required to meet the capacity requirements.

The number derived from the above formula is only an estimate. This number must be adjusted down if there is a write-mostly application because a RAID 5 configuration is affected by heavier write penalties.

### RAID 0+1 versus RAID 5 Calculations

As can be seen in this example, RAID 5 looks to be the better choice when using 10K 146-GB drives. However, it is important to look at different types of drives when doing these calculations. For example, if a drive that has 300 GB is substituted for the 146-GB drives and if all other characteristics remain the same, then the choices drastically change.

Using 300-GB drives, RAID 0+1 requires just 4 drives to meet both capacity and performance characteristics. RAID 5 will require 3 drives to meet capacity but 6 drives to meet performance requirements. By changing the size of the drive, the best choice changed as well.

## Storage Monitoring

IOPS are used to help characterize the performance requirements of the system. However, once the system is up, additional performance monitoring can be utilized to determine if the disk subsystem is slowing the system down.

Table B-7. Performance Monitor Counters for Disk Performance

|  |  |  |
| --- | --- | --- |
| Object | Counter | Instance |
| Physical Disk | % Idle Time | <All Instances> |
| Physical Disk | Disk Transfers/sec | <All Instances> |
| Physical Disk | Avg. Disk secs/Transfers | <All Instances> |
| Physical Disk | Split IO/sec | <All Instances> |

### Physical Disk\% Idle Time

The % Idle Time counter is the percent of time that a disk was idle during the sampling interval. An idle time of less than 20 percent indicates that the disk may be overloaded.

Physical Disk(n)\Disk utilization can be derived by subtracting the Physical Disk(n)\% Idle Time from 100%.

### Physical Disk\Disk Transfers/sec

The Disk Transfers/sec is the number I/O request Packets (IRP) that has been completed during the sampling interval. A disk can only handle one I/O operation at a time. So the number of physical disks attached to the computer serves as an upper bound on the sustainable disk IO rate. Where disk arrays are used, divide the Disk Transfers/sec by the number of disks in the array to estimate individual disk I/O rates.

The Physical Disk(n)\Average Disk Service Time/Transfer can be calculated by taking the Physical Disk(n)\Disk Utilization and dividing it by the Physical Disk(n)\Disk Transfers/sec for each physical disk. This indicates how fast the drive responds to a request. If this climbs above what is specified for the disk, it can indicate that the subsystem is overloaded.

### Physical Disk\Avg. Disk secs/transfers

The Avg. Disk secs/transfer is the overall average of the response time for physical disk requests during the sampling interval. This includes both the time the request was serviced by the device and the time it spent waiting in the queue. If this climbs to over 15–25 disk I/Os per second per disk, then the poor disk response time should be investigated.

The Physical Disk(n)\Average Disk Queue Time/Transfer can be calculated by taking the Physical Disk(n)\Avg. Disk secs/Transfer and subtracting the Physical Disk(n)\Avg. Disk Service Time/Transfer. The Average Disk Queue Time/Transfer indicates the amount of time a request spent waiting in the disk queue for servicing. A high queue time can indicate a poorly responding disk subsystem or specifically a poorly responding physical disk.

### Physical Disk\Split IO/sec

Split IO/sec is the rate physical disk requests were split into multiple disk requests during the sampling interval. A large number of split IO/s indicates that the disk is fragmented and performance is being affected. The percentage of Split IOs can be calculated by the following formula where “n” is a specific disk:

(Physical Disk(n)\Split IO/sec ÷ Physical Disk(n)\Disk Transfers/sec) \* 100

If this percentage is greater than 10 to 20 percent, check to see whether the disk needs to be defragmented.

## Disk Throughput

The table below lays out the primary counters for disk throughput. They measure how many bytes per second are being read from or written to a given disk. Although the “Instance” column reads “All Instances,” it is recommended that this be run on each individual instance so as to identify throughput performance on each disk and then tie that to the database files on those disks.

Table B-8. Performance Monitor Counters for Disk Throughput

|  |  |  |
| --- | --- | --- |
| **Object** | **Counter** | **Instance** |
| Physical Disk | Disk Bytes/sec | <All Instances> |
| Physical Disk | Disk Read Bytes/sec | <All Instances> |
| Physical Disk | Disk Write Bytes/sec | <All Instances> |
|  |  |  |

The following table does not deal directly with throughput, but it can be used to correlate disk throughput to disk throughput capacity. For example, if a disk is reading 100 MB/sec for the “Disk Read Bytes/sec” counter and the “% Disk Read Time” is at 50 percent, it is estimated that the disk will be able to handle around 200 MB/sec for read throughput.

Table B-9. Additional Performance Monitor Counters for Disk Throughput

|  |  |  |
| --- | --- | --- |
| **Object** | **Counter** | **Instance** |
| Physical Disk | % Disk Time | <All Instances> |
| Physical Disk | % Disk Read Time | <All Instances> |
| Physical Disk | % Disk Write Time | <All Instances> |

## Network Performance

Most workloads require access to production networks to ensure communication with other applications and services and to communicate with users. Network requirements include elements such as throughput—that is, the total amount of traffic that passes a given point on a network connection per unit of time.

Other network requirements include the presence of multiple network connections.

Table B-10. Performance Monitor Counters for Network Performance

|  |  |  |
| --- | --- | --- |
| Object | Counter | Instance |
| Network Interface | Bytes Total/sec | (Specific network adapters) |
| Network Interface | Current Bandwidth | (Specific network adapters) |
| Ipv4 & Ipv6 | Datagrams/sec | N/A |
| TCPv4 & TCPv6 | Connections Established | N/A |
| TCPv4 & TCPv6 | Segments Received/sec | N/A |

### Network Interface\Bytes Total/sec and Network Interface\Current Bandwidth

This Bytes Total/sec is the number of bytes transmitted and received over the specified interface per second The Current Bandwidth counter reflects the actual performance level of the network adaptor, not its rated capacity. If a gigabit network adapter card on a segment is forced to revert to a lower speed, the Current Bandwidth counter will reflect the shift from 1 Gps to 100 Mpbs.

Using these two values, the Network interface utilization can be calculated for each interface, designated as “n” with the following equation:

(Network Interface(n)\Bytes Total/sec ÷ Network Interface(n)\Current Bandwidth) \*100

If % Busy for a given adapter exceeds 90 percent, then additional network resources will be necessary. Generally, the maximum achievable bandwidth on a switched link should be close to 90–95 percent of the Current Bandwidth counter.

### Ipv4 & Ipv6\Datagrams/sec

These counters show the total number of IP datagrams transmitted and received per second during the sampling interval. By generating a baseline around this counter, a trending and forecasting analysis of the network usage can be performed.

### TCPv4 & TCPv6\Connections Established

The Connections Established counter is the total number of TCP connections in the ESTABLISHED state at the end of the measurement interval. The number of TCP connections that can be established is constrained by the size of the Nonpaged pool. When the Nonpaged pool is depleted, no new connections can be established.

Trending and forecasting analysis on this counter can be done to ensure that the system is properly scaled to handle future growth. The server can be tuned using the TCP registry entries like *MaxHashTableSize* and *NumTcTablePartitions* based on the number of network users seen on average.

### TCPv4 & TCPv6\Segments Received/sec

The Segments Received/sec is the number of TCP segments received across established connections, averaged over the sampling interval. The average number of segments received per connection can be calculated. This can be used to forecast future load as the number of users grows. The following formula can be used to calculate the average number of segments received per connection:

TCPvn\Segments Received/sec ÷ TCPvn\Connections Established/sec

# Version History

|  |  |  |
| --- | --- | --- |
| **Version** | **Description** | **Date** |
| 1.1 | Added What’s New in SQL Server 2008 R2 section.  Added Step 2, Task 5, “Determine If SQL Server Master Data Services Is Required.”  Added Step 7, “Design the SQL Server Master Data Services Infrastructure.”  Updated decision flow and step listings accordingly. | May 2010 |
| 1.0 | Original release. | February 2009 |

# 

# Acknowledgments

The Solution Accelerators–Management and Infrastructure (SA-MI) team acknowledges and thanks the people who produced the Infrastructure Planning and Design Guide for Microsoft SQL Server 2008 and SQL Server 2008 R2. The following people were either directly responsible for or made a substantial contribution to the writing, development, and testing of this guide.

Contributors:

* Steve Abraham – Xtreme Consulting
* Jude Chosnyk – GrandMasters
* Michael Kaczmarek – Microsoft
* Greg Low – GrandMasters
* Robin Maher – Microsoft
* Fergus Stewart – Microsoft
* Melissa Stowe – Microsoft

Reviewers:

* Kevin Cox – Microsoft
* Robert McPhail
* Reagan Templin – *Microsoft*

Editors:

* Laurie Dunham – Microsoft
* Pat Rytkonen – Volt