

# SQL Server 2008 R2 Manageability

SQL Server Technical Article

**Writers:** Graeme Malcolm, Ann Weber, and Martin Ellis (Content Master)

**Technical Reviewer:** Dan Jones, Bill Ramos, and Oscar Naim

**Project Editor:** Joanne Hodgins

**Published:** January 2008

**Updated:** January 2010

**Applies To:** SQL Server 2008 R2

Summary: Microsoft® SQL Server® 2008 R2 builds on the management capabilities of previous versions and provides a variety of tools and frameworks to help you manage your enterprise data environment. SQL Server helps you efficiently manage at scale with SQL Server Management Studio, a single console to manage instances, database applications, services, and resource utilization. SQL Server 2008 R2 provides centralized insights into resource utilization through application and multi-instance management and improved efficiency for the development and deployment of data-tier applications. End-to-end performance, health monitoring and analysis capabilities, and Policy-Based Management help you to optimize your environment and ensure compliance.

### Copyright

This is a preliminary document and may be changed substantially prior to final commercial release of the software described herein.

The information contained in this document represents the current view of Microsoft Corporation on the issues discussed as of the date of publication. Because Microsoft must respond to changing market conditions, it should not be interpreted to be a commitment on the part of Microsoft, and Microsoft cannot guarantee the accuracy of any information presented after the date of publication.

This White Paper is for informational purposes only. MICROSOFT MAKES NO WARRANTIES, EXPRESS, IMPLIED OR STATUTORY, AS TO THE INFORMATION IN THIS DOCUMENT.

Complying with all applicable copyright laws is the responsibility of the user. Without limiting the rights under copyright, no part of this document may be reproduced, stored in or introduced into a retrieval system, or transmitted in any form or by any means (electronic, mechanical, photocopying, recording, or otherwise), or for any purpose, without the express written permission of Microsoft Corporation.

Microsoft may have patents, patent applications, trademarks, copyrights, or other intellectual property rights covering subject matter in this document. Except as expressly provided in any written license agreement from Microsoft, the furnishing of this document does not give you any license to these patents, trademarks, copyrights, or other intellectual property.

2010 Microsoft Corporation. All rights reserved.

Microsoft and SQL Serverare 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.

### Table of Contents

[Introduction 1](#_Toc251068775)

[Manage Efficiently at Scale 1](#_Toc251068776)

[Centralized Data Services Administration 1](#_Toc251068777)

[Integrated Configuration 3](#_Toc251068778)

[Multi-Instance Management 4](#_Toc251068779)

[Single Unit of Deployment 8](#_Toc251068780)

[Enterprise Administration with System Center 10](#_Toc251068781)

[Effective Automation 10](#_Toc251068782)

[Configuring SQL Server with Policy-Based Management 10](#_Toc251068783)

[Policy-based Framework Facets 11](#_Toc251068784)

[Distributing Policies across the Enterprise 14](#_Toc251068785)

[Applying Policies 15](#_Toc251068786)

[Automated Maintenance 15](#_Toc251068787)

[Maintenance Plans 15](#_Toc251068788)

[Programmatic Management Interfaces 16](#_Toc251068789)

[SQLCMD 16](#_Toc251068790)

[Server Management Objects 16](#_Toc251068791)

[Automate Administration and Deployment 16](#_Toc251068792)

[Streamlined Troubleshooting and Performance Optimization 17](#_Toc251068793)

[Data Collection and Monitoring with SQL Server Performance Data Collection 17](#_Toc251068794)

[Low Overhead Data Collection 17](#_Toc251068795)

[Centralized Storage of Performance Data 18](#_Toc251068796)

[Analytical Reporting 18](#_Toc251068797)

[Optimizing Database Performance 19](#_Toc251068798)

[SQL Server Profiler 19](#_Toc251068799)

[Database Engine Tuning Advisor 20](#_Toc251068800)

[Conclusion 20](#_Toc251068801)

#### Introduction

Each release of Microsoft SQL Server has brought innovative manageability features that help reduce the total cost of ownership of your data services solution by automating administrative tasks, centralizing management, and enabling administrators to focus on higher value, strategic activities. Microsoft SQL Server 2008 R2 builds on the enterprise-class manageability capabilities of earlier releases, and introduces application and multi-instance management for improved efficiencies around managing resources at scale and data-tier application deployments and upgrades.

This paper introduces the manageability capabilities of SQL Server 2008 R2 and describes how you can use them to:

* Scale administrative operations easily and manage data services of all types throughout the organization.
* Automate maintenance, best practices, and configuration management to reduce administrative overhead.
* Streamline troubleshooting and analysis and optimize data services performance across the enterprise.

#### Manage Efficiently at Scale

Today’s organizations need to store and manage ever-increasing volumes of data, and support a growing number of software applications and services that access that data. This trend has led to database applications that span the enterprise, which dramatically increases the workload for database administrators and other operational support personnel.

IT staff must manage a variety of data services and database applications such as online transaction processing (OLTP) databases, online analytical processing (OLAP) databases, departmental apps, reporting solutions, extract, transform, and load (ETL) operations, and occasionally connected data solutions on laptops and mobile devices. As the number of database applications and services continuously grow, data management functions must help IT continue to incrementally scale across the enterprise.

##### Centralized Data Services Administration

SQL Server Management Studio, shown in Figure 1, is a comprehensive, centralized management tool that enables you to manage all of your SQL Server instances through a single interface. With SQL Server Management Studio, you can manage all of your database related services including SQL Server database engine, Analysis Services, Reporting Services, Integration Services and SQL Server Compact Edition. Database administrators can use the extensive standard and customizable reports to provide a concise and graphical view of the state of the database environment.

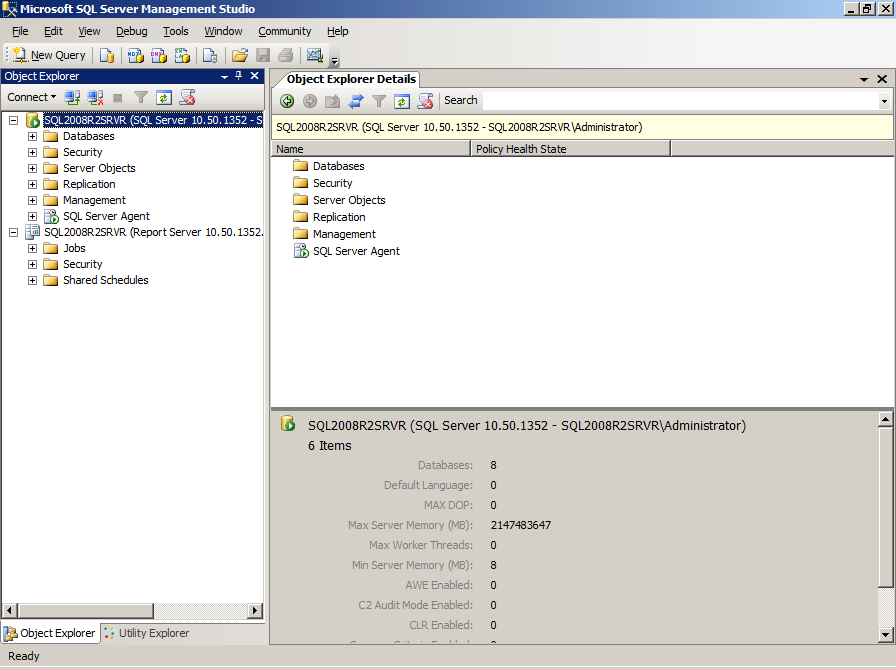


Figure 1: SQL Server Management Studio

SQL Server Management Studio includes a number of standard reports at both the server level and database level that you can use to view performance statistics and live activity. You can use these reports to troubleshoot performance problems that are caused by excessive resource usage or concurrency issues. Figure 2 shows the Memory Consumption standard report.

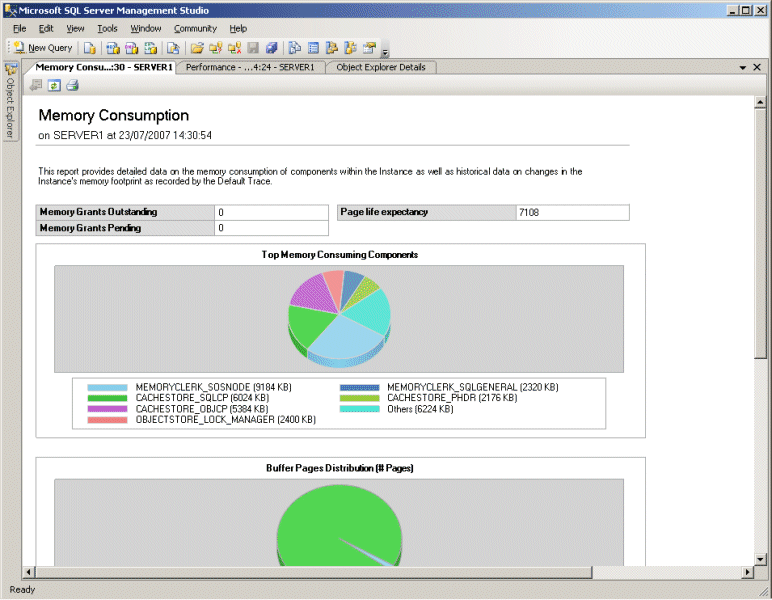


Figure 2: Memory Consumption standard report

##### Integrated Configuration

SQL Server Configuration Manager, shown in Figure 3, enables system administrators to manage SQL Server services and network libraries from a single location. You can add the SQL Server Configuration Tool as a snap-in to the Microsoft Management Console (MMC). By creating a custom management console that is specifically designed to support their job functions, system administrators can save time by not having to open multiple tools. Additionally, the SQL Server configuration tool helps system administrators to manage portions of the services that relate to their functions, which relieves the database administrator to manage databases and database performance.

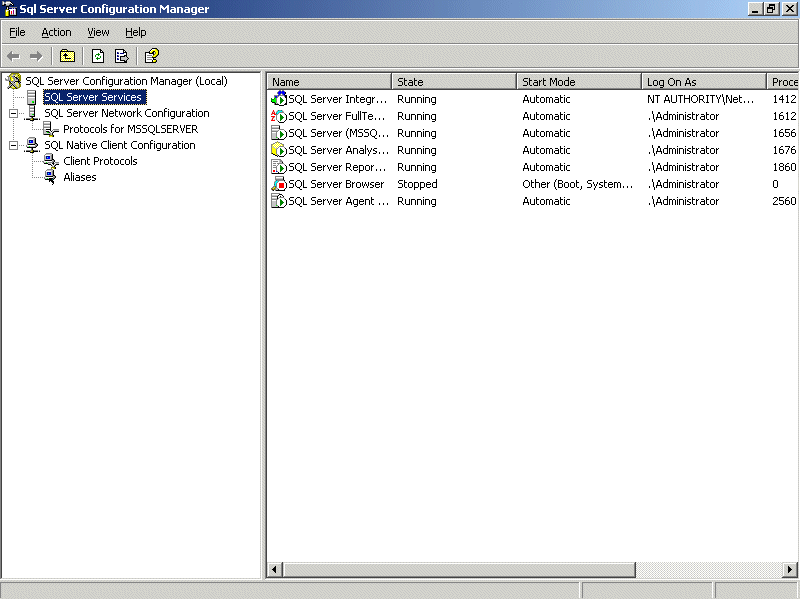


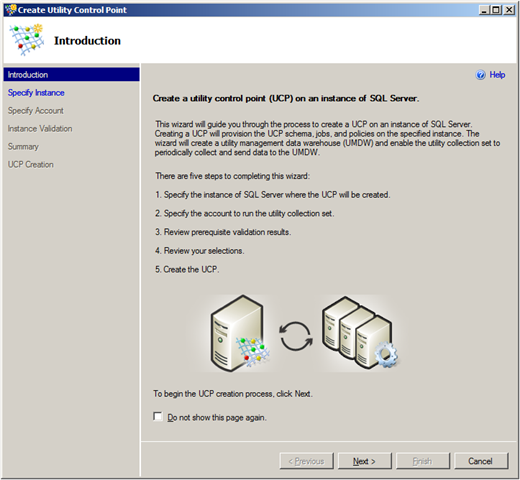
Figure 3: SQL Server Configuration Manager

##### Multi-Instance Management

SQL Server 2008 R2 delivers a new way to monitor and manage the utilization of multiple SQL Server instances and data-tier applications by introducing a centralized *utility control point* and enrolling SQL Server instances and applications from across your enterprise. Default policies evaluate the resource utilization levels of the enrolled instances and assess violations of these policies to give an overall, enterprise-wide view of database server utilization.

Setting up a Control Point is very straightforward. A new wizard in SQL Server Management Studio guides you through a simple five-step process of creating the Control Point on your chosen SQL Server 2008 R2 instance (see Figure 4), and additional wizards enable you to enroll instances and register data-tier applications to be managed by the Control Point . With SQL Server 2008 R2, you have the flexibility to enroll instances one at a time or batch-enroll using PowerShell integration.

Control Point also supports management of data-tier applications. A data-tier application is a container that defines and bundles the database schema, reference data, and deployment requirements of an application together so you can manage them as a single entity. You can manage data-tier applications through the Control Point and evaluate resource utilization just as you can for SQL Server instances.

**Figure 4: Create Utility Control Point wizard**

Instances and data-tier applications enrolled into the Control Point send configuration and performance information to the Control Point every fifteen minutes. This data is stored in a specially created database called the utility management data warehouse (UMDW), which is hosted on the Control Point. The data from each enrolled instance uses approximately 2GB in the UMDW per year, so you must ensure that the instance that hosts this database has enough spare storage capacity to handle this. You can also configure the retention period for this data, with values ranging from one month to two years. The default is that the data is retained for a period of one year.

To manage the Control Point and enrolled instances and registered data-tier applications, SQL Server 2008 R2 Management Studio has a new feature called Utility Explorer. A sister feature to Object Explorer that helps you manage individual databases, Utility Explorer is the management interface for multi-instance management. Utility Explorer allows you to access each Control Point, and it displays summary and detail information about enrolled instances and registered data-tier applications, including database file utilization, storage volume utilization, and Control Point utilization. From the Utility Explorer you can also adjust policies and security settings for the Control Point. It features an intuitive dashboard view with color-coded charts that enable administrators to see at a glance which SQL Server instances and data-tier applications are overutilized or underutilized, helping them to make informed, proactive decisions about server consolidation or expansion to optimize hardware resource utilization. Figure 5 shows the dashboard view.

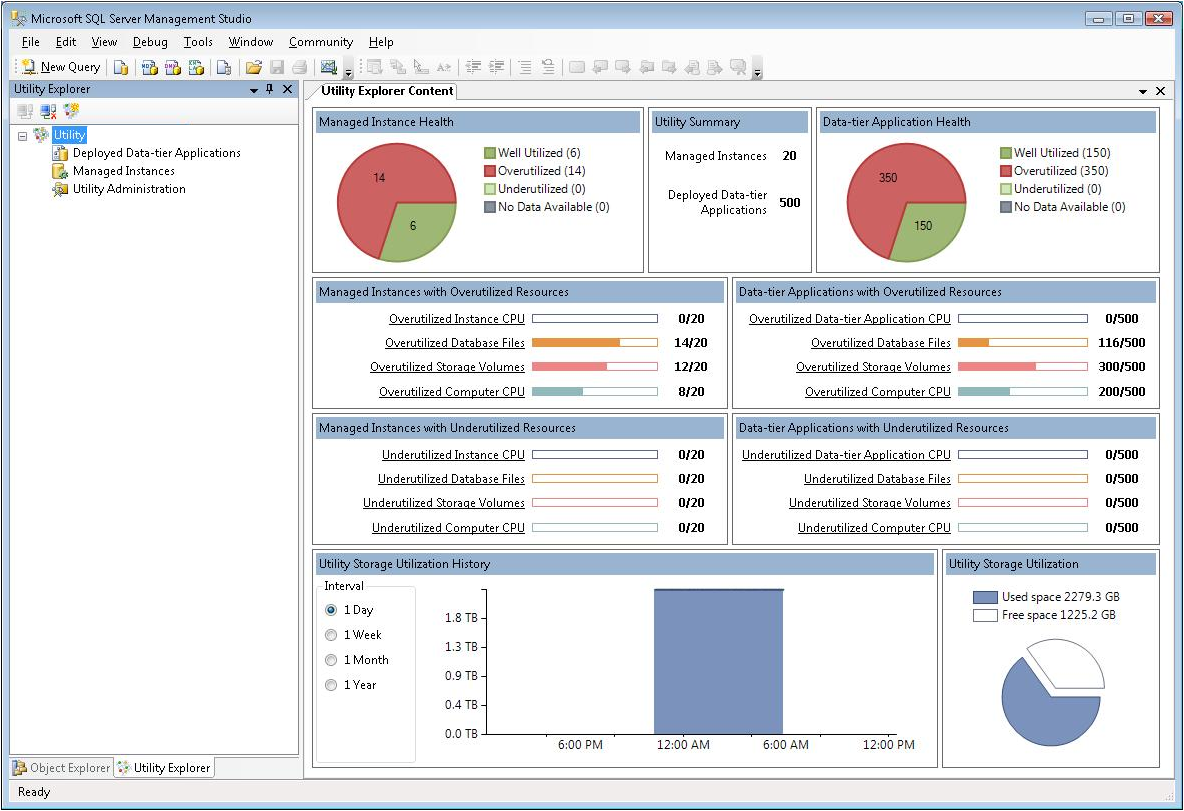
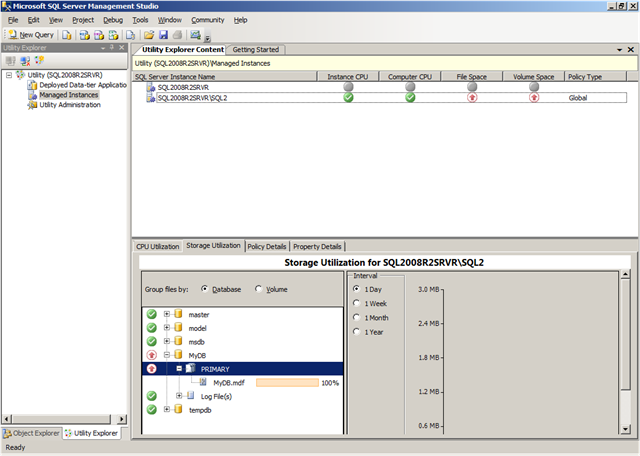


Figure 5: Utility control point dashboard

You can use Utility Explorer to adjust global policies that specify thresholds for underutilization and overutilization of resources. For example, you can specify that 80 percent represents the point at which instance processor utilization is overutilized, and 20 percent as the point where it is underutilized. Performance figures collected from enrolled instances are compared to these values, and any instances that exceed or fall below these figures will be immediately obvious in the dashboard view in Utility Explorer.

After you have identified an overutilized or underutilized instance, you can drill down from the summary information offered in the dashboard view to a more detailed breakdown for that instance by selecting the instance under the **Managed Instances** node in the Utility Explorer, as shown in Figure 6.

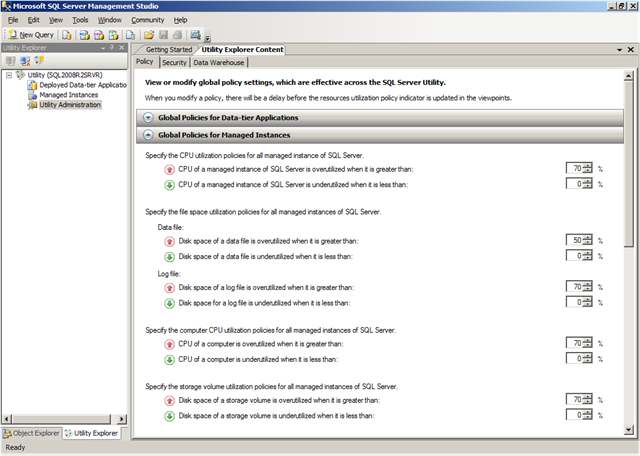
**Figure 6: Utility Explorer Content detail view for a managed instance**

Utilization policies can define maximum and minimum thresholds for:

* Instance processor utilization
* Computer processor utilization
* File space utilization for database and log files
* Computer volume space utilization

As well as setting global policies, you can also define individual policy settings for individual instances, giving you greater flexibility in your management of multiple SQL Server instances. Any policies defined individually in this way will override the global settings where configured thresholds differ.

Figure 7 shows the options for configuring global policies for managed instances.

**Figure 7: Global Policies for Managed Instances**

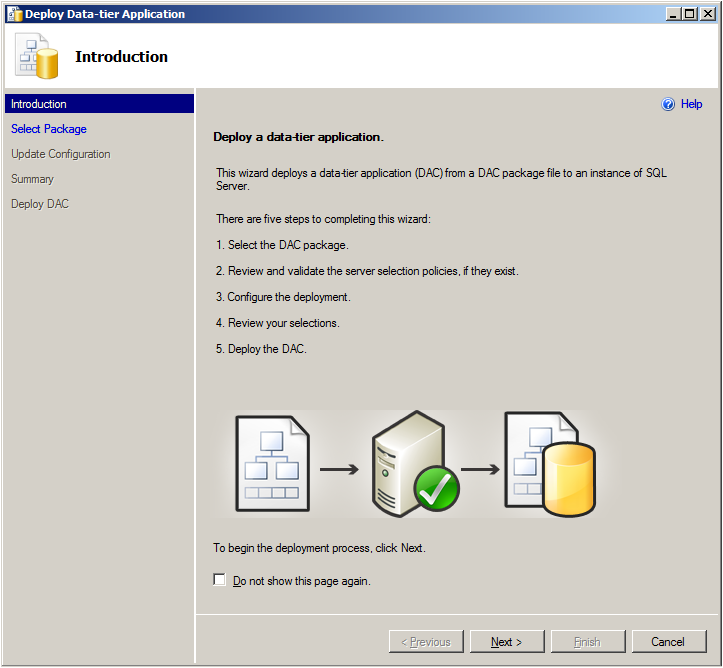
##### Single Unit of Deployment

A key challenge for many database developers and administrators is deploying new applications or upgrading or migrating database applications from one server instance to another—for example, from a staging server to a production server, from one SQL Server instance to another in a server consolidation scenario, or from an older instance of SQL Server to SQL Server 2008 R2. Moving an application from the development environment to a test server and ultimately to the production environment can be problematic for a variety of reasons. For example, the newly deployed database may use a different storage configuration from the development database, or it may rely on dependent server-level objects such as logins that must also be created on the production server for the application to function correctly.

To address these problems, and to help make deployments go more smoothly, SQL Server 2008 R2 introduces the concept of the *data-tier application*. The data tier of an application includes the logical database structure, physical data objects, and dependent server-level objects, such as logins and database and log files. A data-tier application encapsulates these elements into a single unit of deployment called a data-tier application package. Developers can add deployment requirement policies to the data-tier application package as they create or modify it. For example, deployment requirements can be as basic as “VersionMajor is greater than or equal to 10.” These built-in requirements help reduce the loss of information during transition from development to IT. Developers and administrators can create these packages by using SQL Server Management Studio and Microsoft Visual Studio® 2010.

In SQL Server Management Studio, you can use the Extract Data-tier Application wizard to extract information from existing databases to build a package, which can be redeployed elsewhere. If you need to modify the database application before redeployment, you can do so by opening the extracted package in Visual Studio 2010, where you can create new objects such as tables, logins, and users, or add other new deployment requirements. You can also use Visual Studio 2010 to create new data-tier application packages from scratch. Packages that developers create or update by using Visual Studio 2010 can then be passed to database administrators for deployment.

You can deploy new or updated data application component packages by using the Deploy Data-tier Application wizard in SQL Server Management Studio, which is shown in Figure 8. This easy-to-use interface greatly simplifies the deployment and migration of database applications, reducing the time and effort administrators need to spend on this task. You can also create scripts to deploy packages by using Microsoft Powershell™, which can be useful if you need to deploy the same application multiple times.

**Figure 8: Deploying a data-tier application from a data-tier application package**

Data application component packages also simplify the process of migrating existing applications from older platforms, including SQL Server 2000, SQL Server 2005, and SQL Server 2008, to the SQL Server 2008 R2 platform. You can use the Extract Data-tier Application wizard in SQL Server 2008 R2 Management Studio to capture the required information from the database on the older platform into a data-tier application package, and then redeploy it onto a SQL Server 2008 R2 instance as previously described.

The new centralized application and multi-server management capabilities of SQL Server 2008 R2 make the management of complex enterprise environments easier, enabling you to manage server sprawl, optimize resource usage, and manage applications more easily across the application life cycle from a single location. Straightforward wizards enable you to quickly set up dedicated SQL Server Control Point instances, and the new Utility Explorer provides an intuitive, easy-to-use interface for administering Control Point instances in the familiar environment of SQL Server Management Studio.

##### Enterprise Administration with System Center

As well as the native management tools and services provided with SQL Server 2008 R2, administrators can use the SQL Server Management Pack for Microsoft System Center Operations Manager to enable broader system administration that is scalable across the enterprise. With System Center Operations Manager, administrators can implement end-to-end service management of line-of-business (LOB) applications for greater visibility and they can use rich reports and an easy-to-customize reporting environment to bring additional insights when troubleshooting and planning.

#### Effective Automation

SQL Server 2008 R2 reduces the amount of time a database administrator or operator needs to spend performing administrative functions through extensive, policy-based configuration management and automation of maintenance tasks.

##### Configuring SQL Server with Policy-Based Management

The Policy-based Framework provides the ability to define policies that apply to servers, databases, and other objects in your data environment. Well-defined policies can help administrators to control and manage change proactively within the data services environment. The framework delivers the following benefits:

* **Management by Intent**. The framework provides a logical view of the system configuration, which enables administrators to define the desired configuration of data services proactively instead of making changes in response to issues when they arise.
* **Intelligent Monitoring**. Policy-based Framework policies can monitor and prevent changes to the system that deviate from the desired configuration.
* **Virtualized Management**. With the framework, you can scale management across multiple servers, which makes it easy to enforce consistent configuration policies across the enterprise.

The Policy-based Framework uses the following objects when configuring policy management:

* **Facets**. Contain properties that expose options related to an area of policy management. For example, the **Surface Area** includes properties that relate to specific SQL Server features, such as **Database Mail Enabled** and **CLR Integration Enabled**.
* **Conditions**. Express the state of a facet. are based on a single facet, and can be used in one or more policies. For example, you could create a condition named **Minimal Surface Area** in which all of the properties in the **Surface Area** facet are assigned the value **False**.
* **Policies**. Contain a single condition that is enforced for one or more targets. For example, you could create a policy named **Locked Down Server** that assigns the **Minimal Surface Area** condition to the server.
* **Categories**. Contain one or more policies. Database owners can subscribe a database to one or more category. For example, you could create a category named **Corporate DB Policies** that contains a policy to enforce database object naming conventions and another policy to enforce a specific database compatibility level, and subscribe line of business databases to the category. By default, all databases implicitly subscribe to the default category. Additionally, categories can be marked as *Active* or *Inactive* at the server or database level, which enables administrators to control policy enforcement.
* **Targets**. Are entities, such as servers, databases, logins, tables, or other database objects to which policies are assigned. All targets in a SQL Server instance form a target hierarchy. Within a policy, a target set is defined when a set of filters are applied to the target hierarchy. For example, a target set can include all of the indexes owned by the **Production** schema.

###### Policy-based Framework Facets

Each framework facet represents an aspect of SQL Server management in which one or more related configuration options is defined. SQL Server 2008 R2 includes a number of pre-defined facets, which you can use to define policies that enforce specific configuration option values. For example, SQL Server 2008 R2 includes the **Surface Area** facet, shown in Figure 9.

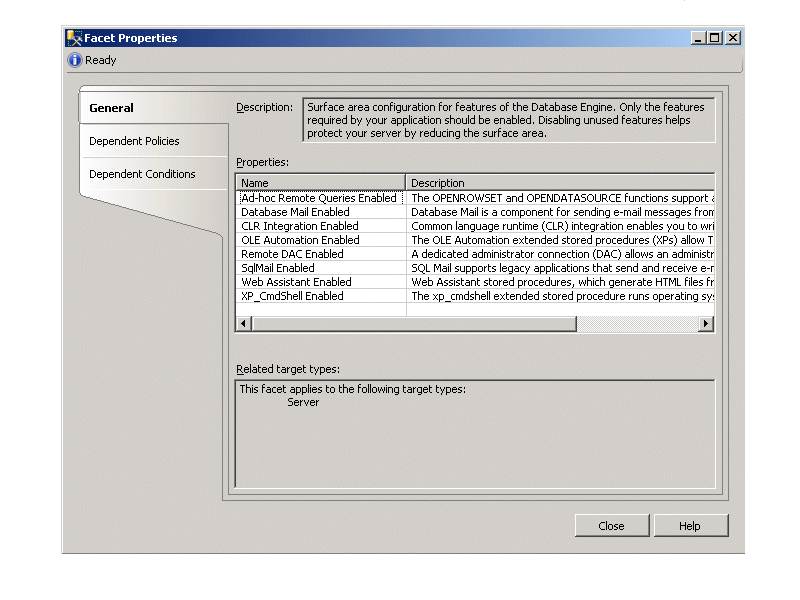


Figure 9: Surface Area Facet

You can create conditions to define a desired configuration that is based on the settings in a facet, as shown in Figure 10.

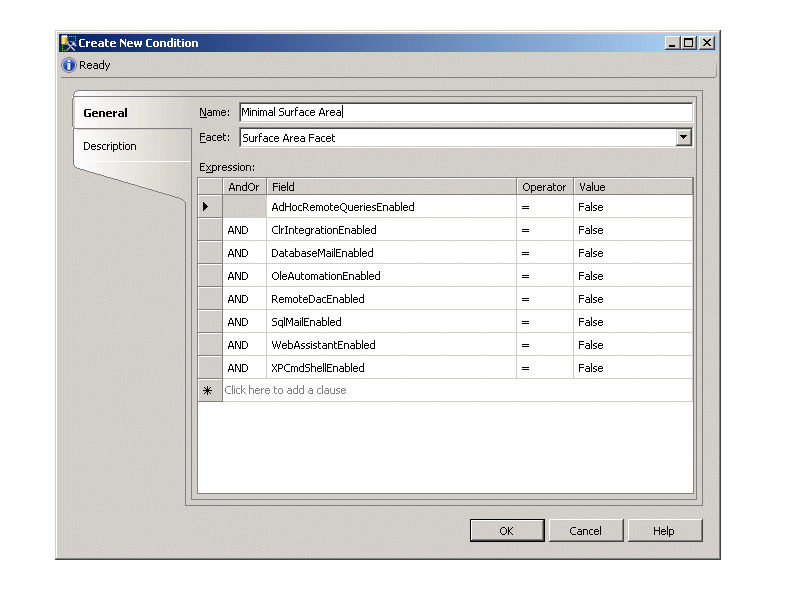


Figure 10: Defining a condition

Finally, to define a policy, you simply specify the condition that you want to enforce, as shown in Figure 11.

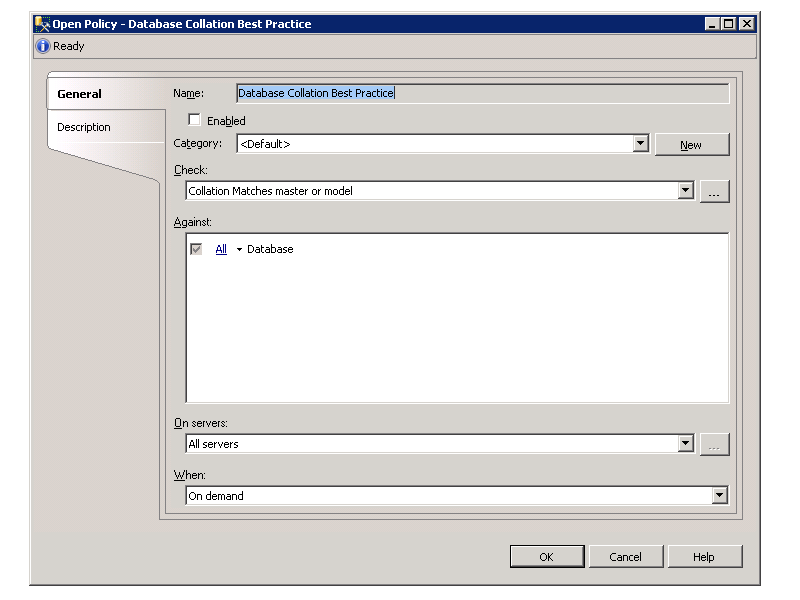


Figure 11: Defining a policy

SQL Server 2008 R2 provides a comprehensive set of facets that you can use to specify configuration settings and rules for your SQL Server 2008 R2 implementation. Some examples of ways in which you can use the provided facets to define policies include:

* Using the **Server** facet to enforce specific server configuration settings such as the login authentication mode.
* Using the **Surface Area** facet to control which features are enabled, and reduce the surface area of the server.
* Using the **Database** facet to enforce specific database settings such as the compatibility level.
* Using the **Multipart Name** facet to enforce naming conventions for tables views, and other schema-bound database objects.
* Using a range of facets to enforce known best practices for your database solution, such as ensuring that data files are stored on a separate drive from log files.

###### Distributing Policies across the Enterprise

The framework provides the capability to publish policy categories to additional SQL Servers in your environment. You can define policies on a SQL Server instance (known as the *configuration server*) and then replicate those policies to other servers (known as *configuration targets*). Any changes made to the policies or the categories on the configuration server (such as marking a particular category as inactive) are automatically propagated to all of the configuration targets, which significantly reduces the overhead that is associated with enforcing configuration policy compliance across the enterprise. In addition to the distribution of policies, you can execute queries against the set of configuration targets by using the Query Editor.

###### Applying Policies

When you define a policy, you select an execution mode for the policy, which will determine whether the policy is enforced proactively, on a scheduled basis, or manually through administrative intervention. The following four execution mode options are in the policy properties:

* **On Demand**. An administrator manually invokes policy checking.
* **On Schedule, log out-of-compliance**. A SQL Server Agent job periodically executes the policy and records any configuration settings that do not comply with active policies.
* **Changes are attempted, prevent out-of-compliance**. DDL triggers roll-back transactions that do not comply with active policies.
* **Changes are attempted, log out-of-compliance**. Event notification evaluates a policy when a relevant change occurs and records any configuration settings that do not comply with active policies.

##### Automated Maintenance

Through the SQL Server Agent service, SQL Server 2008 R2 gives administrators the flexibility to provide an automated and proactive management and maintenance strategy for their database environment. The SQL Server Agent provides the following components to configure your automated environment:

* **Jobs**. A series of one or more steps to be completed. These steps can consist of Transact SQL statements, SQL Server Integration Services package, or command-line statements.
* **Alerts**. Preconfigured messages that are generated automatically when a particular event occurs on a SQL Server.
* **Operators**. Users defined with additional contact information that may include an e-mail address, pager number, or *net send* address. Alerts are sent to these operators when preconfigured events occur, or to notify them of success, failure, or completion of a job or job step.
* **Schedules**. Specifies when a job runs. A job can be associated with more than one schedule.

Multi-server jobs can be maintained at a centralized server and used to manage SQL Server instances across your organization.

###### Maintenance Plans

Maintenance plans enable administrators to quickly configure standard maintenance policies, such as regular database and transaction log backups, and index maintenance to be implemented on SQL Server instances. The Maintenance Plan Wizard provides a quick and user-friendly way to define maintenance plans for one or more databases. Similarly to jobs, multi-server maintenance plans can be managed from a centralized server in a *master server/target server* configuration. In addition to the Maintenance Plan Wizard, SQL Server Management Studio provides an extensive design surface with which you can define an enhanced workflow for maintenance plan tasks.

##### Programmatic Management Interfaces

You can customize and extend your management capabilities by creating applications and scripts to perform common administrative tasks. This approach can significantly simplify the operation and management of your data services by delegating management tasks to less experienced administrators and providing a simple utility that contains only the functionality that is required for those tasks.

###### SQLCMD

To facilitate multi-script automation through batch files, SQL Server 2008 includes SQLCMD. SQLCMD is a comprehensive command-line tool that you can use to run parameterized and multi-server scripts that perform administrative tasks. By incorporating SQLCMD operations in a batch file or script, you can achieve a high level of administrative automation without requiring operators to use complex management interfaces.

###### Server Management Objects

Server Management Objects (SMO) is an enhanced Application Programming Interface (API) that exposes the full management environment of SQL Server to developers. You can use SMO to build powerful custom applications to manage SQL Server–based solutions.

##### Automate Administration and Deployment

Use the PowerShell language to build more robust administrator scripts and manage across multiple Microsoft products. Use the SQL Server provider to quickly navigate paths to locate objects within your servers. Use the set of cmdlets to take actions against the system such as running scripts containing T-SQL or XQuery statements.

Microsoft has incorporated PowerShell into its Common Engineering Criteria for server products, and SQL Server 2008 included PowerShell in its management toolset.  PowerShell by design allows the inclusion of “snap-ins” that provide access to the application that wants to allow a PowerShell interface into the application.  The most notable of these applications is Microsoft Exchange 2007, which rebuilt the application from the ground up with administrative interfaces built for PowerShell, including hundreds of cmdlets for various functions, and the Exchange administrative application executes the PowerShell cmdlets.

The SQL Server PowerShell extensions delivered new cmdlets in 2008, beyond the basic set of available cmdlets.  The cmdlets are:

* Invoke-Sqlcmd
* Invoke-PolicyEvaluation
* Encode-SqlName
* Decode-SqlName
* Convert-UrnToPath

For more information on PowerShell support in SQL Server 2008 R2, please visit: <http://msdn.microsoft.com/en-us/library/dd938892.aspx>

#### Streamlined Troubleshooting and Performance Optimization

A significant aspect of data services management is to ensure that databases and applications provide the required level of performance. SQL Server 2008 R2 provides various tools that you can use to collect and analyze performance data, and to optimize database performance to meet the requirements of your common data access workloads.

##### Data Collection and Monitoring with SQL Server Performance Data Collection

SQL Server 2008 R2 provides Performance Data Collection that enables you to collect, analyze, troubleshoot, and store SQL Server diagnostics information. Performance Data Collection provides an end-to-end solution for performance monitoring that includes low overhead collection, centralized storage, and analytical reporting of performance data. You can use SQL Server Management Studio to manage collection tasks, such as enabling the data collector, starting a collection set, and viewing system collection set reports as a performance dashboard. You can also use system-stored procedures and the Performance Studio API to build your own performance management utilities based on Performance Data Collection.

###### Low Overhead Data Collection

Performance Data Collection provides a unified data collection infrastructure that consists of a data collector in each SQL Server instance that you want to monitor. The data collector is flexible and enables you to manage the scope of data collection to fit development, test, and production environments. You can easily collect both performance and general diagnostic data with the data collection framework.

The data collector infrastructure introduces the following new concepts and definitions:

* **Data Provider**. Sources of performance or diagnostic information that can include SQL Trace, Performance counters, and Transact-SQL queries (for example, to retrieve data from Distributed Management Views).
* **Collector Type**. A logical wrapper that provides the mechanism for collecting the data from the data provider.
* **Collection Item**. An instance of a collector type. When you create a collection item, you define the input properties and collection frequency for the item. A collection item cannot exist on its own.
* **Collection Set**.The basic unit of data collection. A collection set is a group of collection items that are defined and deployed on a SQL Server instance. Collection sets can run independently of each other.
* **Collection Mode**. The manner in which the data in a collection set is collected and stored. The collection mode can be set to cached or non-cached. The collection mode affects the type of jobs and schedules that exist for the collection set.

The data collector is extensible and supports the addition of new data providers.

###### Centralized Storage of Performance Data

When the data collector is configured, a relational database with the default name **MDW** is created as a management data warehouse in which to store the collected data. This database can reside on the same system as the data collector, or on a separate server. Objects in the management data warehouse are grouped into the following three preconfigured schemas, each with a different purpose:

* The **Core** schema includes tables and stored procedures for organizing and identifying the collected data.
* The **Snapshot** schema includes data tables, views, and other objects to support the data that is collected from the standard collector types.
* The **Custom\_Snapshot** schema enables the creation of new data tables to support user-defined collection sets that are created from standard and extended collector types.

For more information about Performance Data Collector and the Management Data Warehouse, please visit: <http://msdn.microsoft.com/en-us/library/dd939169.aspx>

###### Analytical Reporting

Performance Data Collection helps you to analyze your collected data quickly with a robust set of preconfigured system collection sets, including **Server Activity**, **Query Statistics**, and **Disk Usage**. The **Server Activity** system collection set is the entry point for most monitoring and troubleshooting scenarios. A set of reports that is associated with each system collection set is published in SQL Server Management Studio, and you can use these reports as a performance dashboard to help you analyze the performance of your database systems as shown in Figure 12.

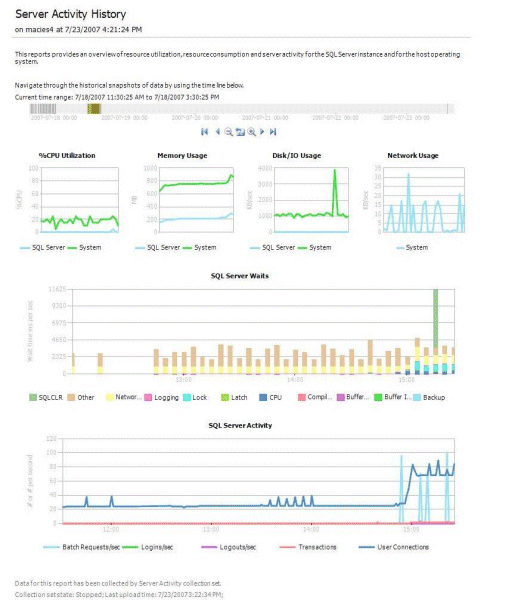


Figure 12: A Performance Data Collection report

##### Optimizing Database Performance

SQL Server 2008 R2 provides a variety of well-integrated tools to optimize your database environment. These tools give operators and administrators the ability to collect statistics and other data so that they can provide optimal database environment performance in the enterprise.

###### SQL Server Profiler

SQL Server Profiler shows how SQL Server internally resolves queries by providing a graphical interface for the functionality provided through SQL Trace. With SQL Server Profiler, you can capture the activity generated by a workload on a SQL Server instance. The events that are captured and displayed can be defined by using predefined templates, and you can modify the system provided templates to create your own templates. You can save the captured workload to a table or to a file for future evaluation. You can immediately start a new trace from SQL Server Management Studio based on your connection context. For example, if you want to trace statement execution for a particular Query Editor session, the new **Trace Query in SQL Server Profiler** command immediately starts a trace that uses the query window SPID as a filter.

After you have collected a workload from your production system, you can replay the workload on a new test system to verify the performance with an actual workload. You can also correlate these workloads with System Monitor Performance Counter logs to view how the workload affects the system performance.

In SQL Server 2008 R2, you can use SQL Server Profiler to capture activity on Microsoft SQL Server™ 2000, Microsoft SQL Server™ 2005, SQL Server 2008, and SQL Server 2008 R2 database engine instances, and on SQL Server 2005, SQL Server 2008, and SQL Server 2008 R2 Analysis Services instances.

###### Database Engine Tuning Advisor

You can use the Database Engine Tuning Advisor (DTA) to gain insight into the existing indexing and partitioning structure of your databases, and to get recommendations for how to improve database performance by creating appropriate indexing and partitioning structures. In addition to optimizing your indexing structure, DTA can recommend new physical data structures, including partitioning. DTA also offers you the ability to tune across multiple servers and limit the amount of time the tuning algorithms run. DTA is available as both a command-line and graphical utility. The command-line utility enables you to take advantage of advanced scripting options.

#### Conclusion

SQL Server 2008 R2 builds on the management capabilities of previous versions of SQL Server and provides a greater variety of tools and frameworks to help you manage your enterprise data environment. New multi-server management capabilities make enterprise administration more efficient through a centralized, intuitive interface, and application deployment is much simpler. SQL Server 2008 R2 provides the tools that help you to scale your data environment to meet the growing needs of your company and to optimize the performance of these databases across your enterprise. It also provides functionality to relieve database administrators of day-to-day maintenance tasks and reduces management overhead through automation.

For more information:

<http://www.microsoft.com/sql>.

Did this paper help you? Please give us your feedback. Tell us on a scale of 1 (poor) to 5 (excellent), how would you rate this paper and why have you given it this rating? For example:

* Are you rating it high due to having good examples, excellent screenshots, clear writing, or another reason?
* Are you rating it low due to poor examples, fuzzy screenshots, unclear writing?

This feedback will help us improve the quality of white papers we release. [Send feedback](mailto:sqlfback@microsoft.com?subject=White%20Paper%20Feedback:%20SQL%20Server%202008%20R2%20Manageability).