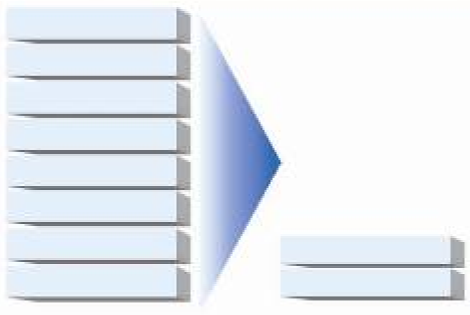
CONSOLIDATION



CONSOLIDATING SQL SERVER™ 2000 DATABASES FROM HP™ PROLIANT™ DL585 SERVERS TO A DELL™ POWEREDGE™ R905 SERVER

REPORT

A Principled Technologies report commissioned by Dell

Table of contents

Executive summary

Consolidating multiple database servers into a single one may let an organization save hardware, software, and operational costs. To help quantify the potential savings, Principled Technologies examined an example test case: How many HP™ ProLiant™ DL585s with a Modular Smart Array 30 (MSA30) can you consolidate onto a single Dell™ PowerEdge™ R905 with a Dell PowerVault™ MD1000 attached? We defined a consolidation methodology, which we detail later in this report; conducted multiple rounds of testing; and analyzed the results.

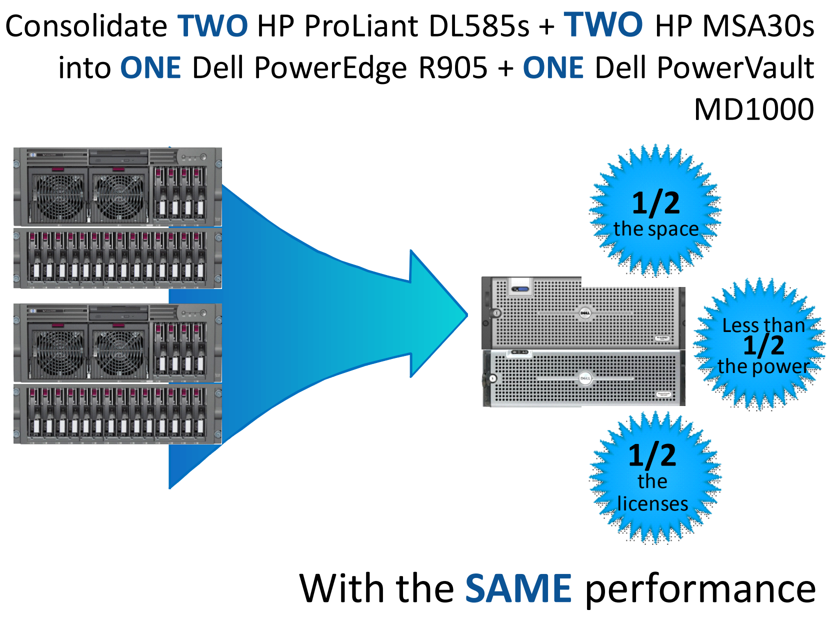

Consolidate TWO HP ProLiant DL585s + TWO HP MSA30s into ONE Dell PowerEdge R905 + ONE Dell PowerVault MD1000
1/2 the space
Less than 1/2 the power
1/2 the licenses
With the SAME performance

Figure 1. The benefits of consolidating two HP ProLiant DL585 with two MSA30 arrays onto a single Dell PowerEdge R905 with one Dell PowerVault MD1000.

As Figure 1 shows, the savings were substantial, with a single PowerEdge R905 and PowerVault MD1000 combination able to replace two legacy HP ProLiant DL585 and MSA30 arrays—and deliver the same performance.

About the Dell consolidation reports

This consolidation report is part of a series of documents that provide concepts, procedures, and hard performance data to help you successfully consolidate your Microsoft® SQL Server® 2000 and SQL Server 2005 databases from multiple machines onto a single SQL Server 2008 server system. In this report, we use the legacy configuration HP ProLiant DL585 with MSA30 array running SQL Server 2000 as the legacy server and the Dell PowerEdge R905 as a target consolidation server. Upcoming reports will feature other Dell PowerEdge servers.

This report presents hard performance data that our hands-on benchmarking has yielded. To learn more about important overall consolidation concepts, see our earlier guide "Consolidating SQL Server 2000 and SQL Server 2005 databases to SQL Server 2008 on Windows Server 2008 Enterprise on Dell Servers" (www.dell.com/sql). That Guide reviews in detail the approach that PT and Dell recommend for a tested and validated consolidation of SQL Server 2000 and SQL Server 2005 databases to SQL Server 2008 using the Dell PowerEdge 2950 running Microsoft Windows Server® 2008 Enterprise.

Consolidation overview

In this section, we briefly define **consolidation**, provide reasons that motivate organizations to consider consolidation, define the term ***consolidation factor***, which we use extensively here, and explain at a high level our approach to the testing we conducted for this report.

What is consolidation?

Generally speaking, *consolidation* is the process of combining multiple things to make a single, more effective unit. In an IT context, you can consolidate several types of things, including the following:

Physical servers. After a successful consolidation, all applications should run on fewer servers than before. Ideally, those applications should run at least as well as they did previously.
Storage. Depending on your setup, consolidating servers may let you also consolidate storage by moving data from a number of servers to a large, locally attached, disk storage subsystem in a new server.
Space. As you consolidate servers, you might also reduce the number of racks or even the number of locations that house servers.

A note on virtualization

It's easy to confuse virtualization with consolidation, because many organizations use virtualization as a means of consolidation. As we note above, consolidation is the process of combining multiple things to make a single, more effective unit. Virtualization, using virtual machines to let one physical server appear as multiple logical servers, is one way to consolidate servers, but we will not address it in this report.

Instead, we will focus on a different consolidation technique: running one or more instances of SQL Server on a single server, and running multiple databases in each instance. This method is relatively simple to set up and administer, so it lets many organizations meet their goals without moving to virtualized servers.

Why consolidate?

An effective server consolidation effort has the potential to yield an environment with more consistent management practices and improved reliability, security, and hardware utilization—all while maintaining the previous level of application performance.

Consolidation can also yield a variety of cost savings:

Hardware savings. Buying, powering, and supporting fewer servers brings obvious savings. Other potential hardware cost savings include a need for fewer racks and network switches: as the number of servers decreases, these costs decrease as well.

Software license savings. Consolidation can save an organization significant money in software licenses. We present a detailed example of potential license savings in our earlier guide, "Consolidating SQL Server 2000 and SQL Server 2005 databases to SQL Server 2008 on Windows Server 2008 Enterprise on Dell Servers" (www.dell.com/sql).

Maintenance and staff savings. A consolidated infrastructure offers many opportunities for maintenance, support, and staffing cost savings. Less hardware and associated equipment means fewer servers that require security patches, monitoring, and other ongoing maintenance.

Reduced support costs. The cost of a given level of support is typically proportional to the size of the installation. By reducing the number of servers, support costs are also likely to go down.

Power and cooling savings. Consolidating servers will usually save power in several ways:

- **Fewer servers.** Obviously, fewer servers consume less power.
- **More efficient servers.** Modern servers are typically more efficient than those of a few years ago, so the power consumption per server is likely to decrease.
- **Less air conditioning.** Fewer, more efficient servers produce less heat and consumer less space. Thus, cooling costs should decrease.

Regulatory compliance savings. With fewer physical devices storing data and more uniform management practices, a consolidated environment can help make the process of complying with regulatory requirements, such as Sarbanes-Oxley and the Health Insurance Portability and Accountability Act (HIPAA), less expensive, easier, and more secure.

Consolidation factor

How many old servers can a new server replace? The answer to this question is a number we call the ***consolidation factor***. The process of determining this number is what we call ***sizing***.

Our first step in sizing was to quantify the amount of work an old server was performing. Using that information, we defined a standardized workload that was one older server’s worth of work.

Our next step was to run multiple copies of the standardized workload against a new server to determine how many concurrent workloads the new server could handle while delivering at least the same performance as the old server on each workload. This number was the consolidation factor.

How we tested: An overview

In this report we present hard performance data and the consolidation factor from testing one set of specific hardware configurations.

For the legacy server, we used a server typical of those in use roughly 3 years ago: an HP ProLiant DL585 G1 with four dual-core AMD Opteron 875 2.2 GHz processors running Windows Server 2003 R2 and SQL Server 2000. We also used an MSA30 U320 external storage array.

For our target consolidation server, we used the Dell PowerEdge R905 with four Quad-Core AMD 2.0 GHz processors, running Windows Server 2008 Enterprise Edition SP1 x64 and SQL Server 2008 Enterprise Edition x64. Like the HP ProLiant DL585, the PowerEdge R905 is a 4U, quad-socket server.

NOTE: We set up the ProLiant DL585 and PowerEdge R905 in configurations typical at the time of their release. The ProLiant DL585 had four dual-core processors, 8 GB of RAM, two gigabit NICs, and all four internal drive bays occupied. The PowerEdge R905 had four quad-core processors, 64 GB of RAM, four gigabit NICs, and all eight internal drive bays occupied. We gave both servers external storage: the ProLiant DL585 used the MSA30, while the PowerEdge R905 used the PowerVault MD1000. Greater detail on these systems' configurations appears in the Defining our environment section.

We used the DVD Store™ Version 2 (DS2) test tool, which is freely available from http://www.delltechcenter.com/page/DVD+Store. DS2 is an open-source simulation of an online e-commerce site. Its main reporting metric is orders per minute, or OPM. Unless we state otherwise, all results in this report are in orders per minute.

To create demand on the servers, we set up several workload-generating desktop-class systems on our network, each running a single instance of DS2.

We ran the same workload and used the same settings as in the tests for our previous consolidation guides. Our goal was to represent the consolidation of a server facing a demanding workload, so we made sure the DVD Store test saturated at least one of the major components of the system. Using the parameters specified in Appendix F, we saturated the main data volume on the MSA30 so that the disk was about 90 percent busy, and most cores were above 50 percent throughout the test.

To avoid any external bottlenecks, we made sure that both the network capacity and the performance of the load-generating machines never limited the performance of the server. To simulate a heavily loaded environment, the load-generating systems ran with no think time, blasting requests as quickly as the server could handle them.

The bottom line

When we consolidated the ProLiant DL585 server and MSA30 array onto the Dell PowerEdge R905 server with a PowerVault MD 1000 providing extra storage, the consolidation factor was **two**.

We reached this conclusion after conducting a wide range of work load tests. We used both external and internal storage on both systems, and we tested with two different database sizes: 10 GB and 20 GB. In this section, we present our findings for both of the scenarios that support the consolidation factors we cite above:

- two DS2 instances running against the PowerEdge R905 with the PowerVault™ MD 1000 attached (with 10GB databases)
- two DS2 instances running against the PowerEdge R905 with the PowerVault MD 1000 attached (with 20GB databases)

Standard scenario

As we note earlier, based on experience and discussions with database administrators, we determined that a common database size for this class of server is 10 GB. We loaded two databases, each 10 GB in size, on the Dell PowerEdge R905 and PowerVault MD 1000. We set up two desktop-class systems on the same network to generate the test workloads. (For details on testing, see Appendices F, G, and H.)

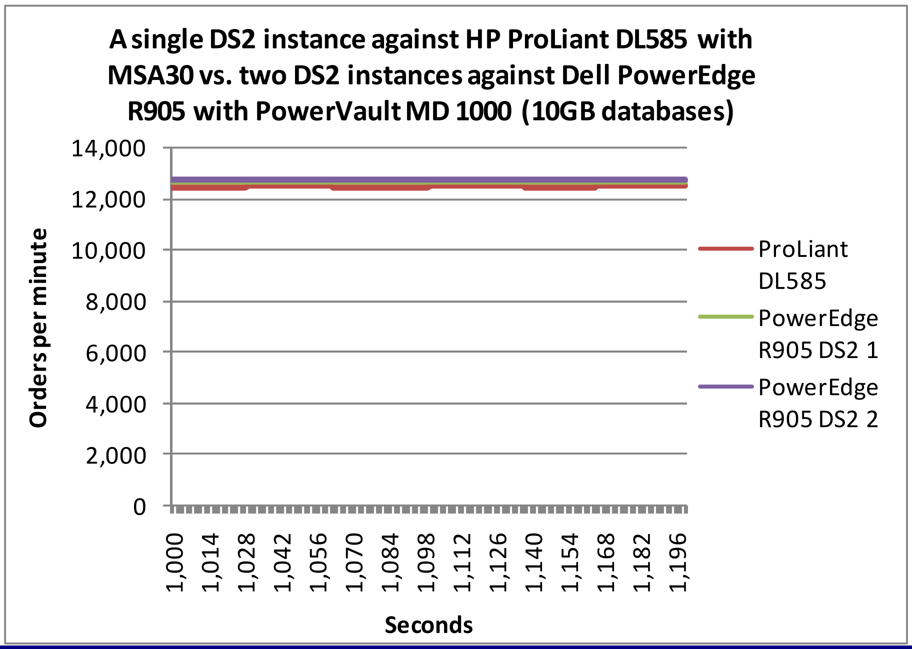


Figure 2. A single DS2 instance running on an HP ProLiant DL585 with MSA30 vs. two DS2 instances running on a Dell PowerEdge R905 with a PowerVault MD 1000, during a period of steady activity, using two DS2 10GB databases. Higher numbers of orders per minute are better.

Each of the lines in Figure 2 represents the performance of a single workload. As we discuss above, each workload represents the work a single ProLiant DL585 is capable of performing. As you can see, each of the two workloads running on the PowerEdge 905 is performing at least as well as a single workload running on the ProLiant DL585 and MSA30. This means that you could consolidate two ProLiant DL585 servers with MSA30 storage onto a single PowerEdge R905 server with a PowerVault MD 1000, reaping great cost savings, without sacrificing any performance at all.

NOTE: We generated all of these results without any special tuning or file layouts. Such tuning would almost certainly have increased the consolidation factor.

High-demand scenario

While we consider 10 GB to be a common database size for this class server, we also ran a more demanding scenario using 20GB databases to see whether the consolidation factor varied.

As Figure 3 shows, the relative performance of the PowerEdge R905 and PowerVault MD 1000 was basically the same, though slightly lower. Although the OPM score for the two workloads on the PowerEdge R905 was a tiny bit under the performance of the ProLiant DL585 server and MSA30 array, it was so close that we from a user's perspective the PowerEdge R905 would still be delivering the performance of two of the HP setups. We thus still consider the consolidation factor to be two.

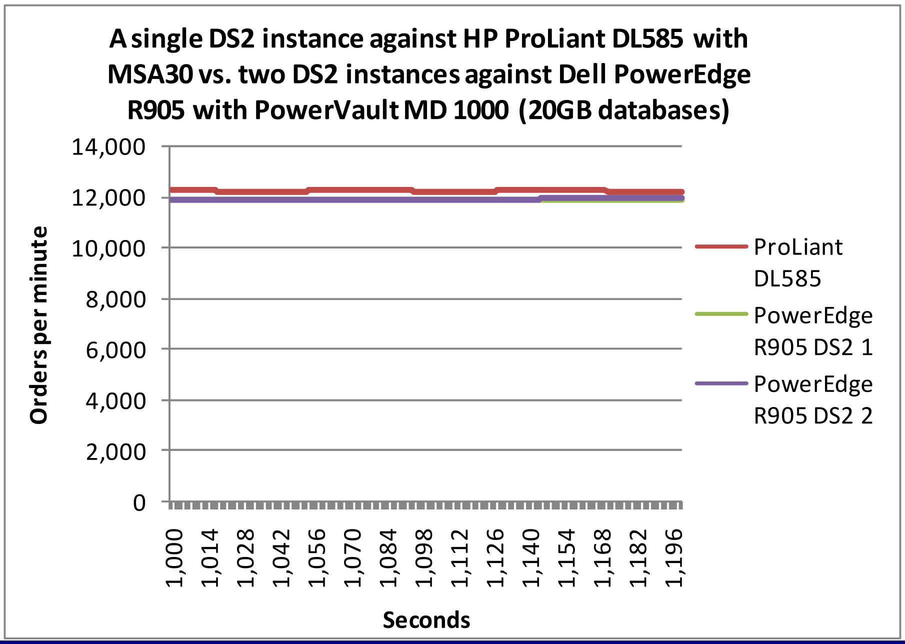


Figure 3. A single DS2 instance running on a HP ProLiant DL585 with MSA30 array vs. two DS2 instances running on a Dell PowerEdge R905 and PowerVault MD 1000, during a period of steady activity, using two DS2 20GB databases. Higher numbers of orders per minute are better.

Power savings

As you might expect, the power savings after consolidation were also dramatic. As Figure 4 shows, the combined power requirements of a Dell PowerEdge R905 and a PowerVault MD 1000 are less than one-half of those of the two HP ProLiant DL585 servers with MSA30 arrays they could replace.

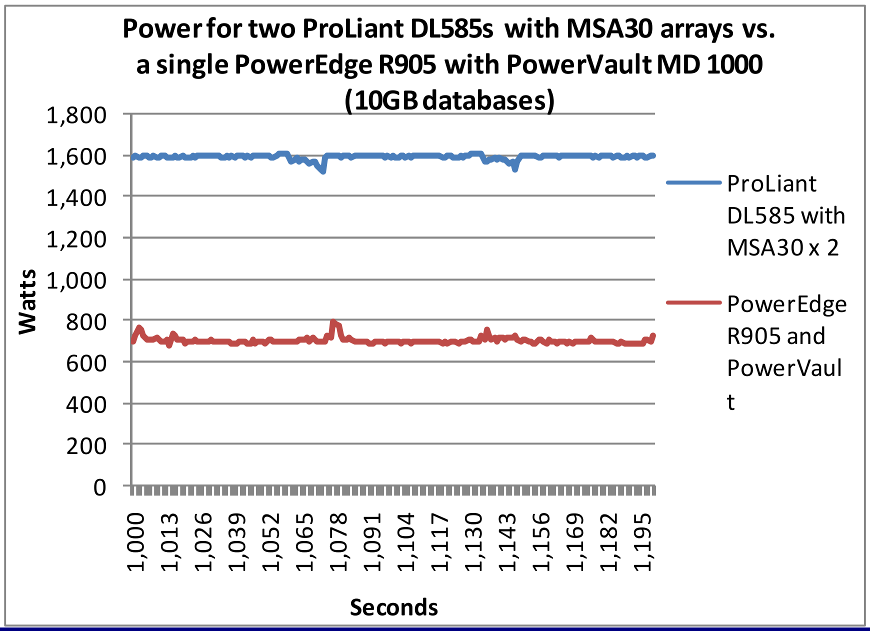


Figure 4. Power usage for two HP ProLiant DL585 servers and MSA30 arrays vs. for a single PowerEdge R905 server with PowerVault MD 1000, in Watts, when using two DS2 10GB databases. Lower power usage is better.

Figure 5 shows that, as you might expect, increasing the database size to 20 GB did not heavily change the relative power consumption of the two servers.

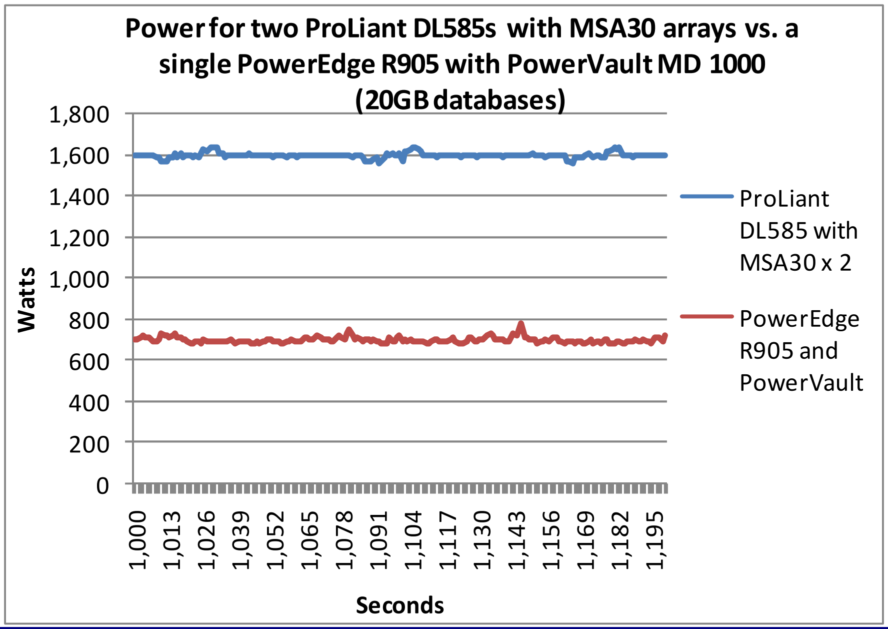


Figure 5. Power usage for two HP ProLiant DL585 servers and MSA30 arrays vs. a single PowerEdge R905 server with PowerVault MD 1000, in Watts, when using two DS2 20GB databases. Lower power usage is better.

Defining our environment

We used a Windows domain containing a SQL Server 2000 database server, a new SQL Server 2008 database server, two workload-generating systems, and an Active Directory server.

To be specific, our example consolidation database server was a Dell PowerEdge R905 running Windows Server 2008 Enterprise Edition SP1 x64 and SQL Server 2008 Enterprise Edition x64. Our legacy server was an HP ProLiant DL585 running Windows Server 2003 R2 Enterprise Edition SP2 x86 and SQL Server 2000 Enterprise Edition SP4. As we note above, both servers used arrays to provide extra storage for the testing. The ProLiant DL585 used an MSA30 array, while the PowerEdge R905 used a PowerVault MD 1000. Our Active Directory server was a Dell PowerEdge 1950 running Windows Server 2003 R2 Enterprise Edition SP2. We connected all of the components via a gigabit Ethernet switch. Figure 6 illustrates our setup.

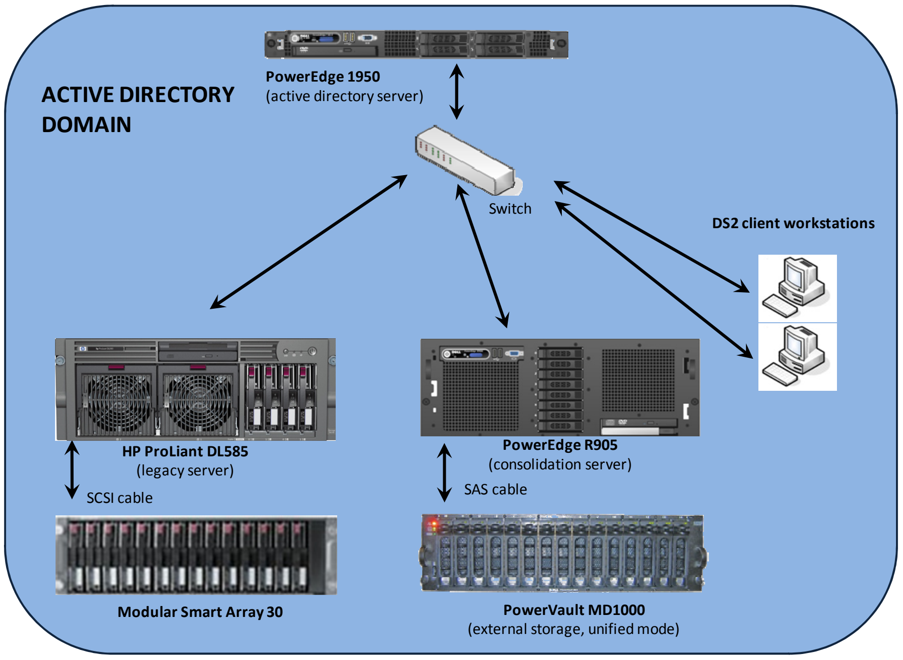


Figure 6. The setup we used in our hands-on testing and research.

Figures 7 and 8 present the hardware and software we used to simulate an Active Directory domain and associated SQL Server 2000, and SQL Server 2008 database servers.

Server	Processor	Memory	Disk drives
Dell 1950 (Active Directory server)	2 x Quad-Core E5440 Intel® Xeon® (2.83 GHz)	16 GB	2 x SAS, 15K RPM, 146 GB
HP ProLiant DL585 G1 (SQL Server 2000 database server)	4 x dual-core AMD Opteron 875 (2.2 GHz)	8 GB	4 x Ultra320 SCSI, 10K RPM, 73 GB
Dell PowerEdge 905 (consolidation database server)	4 x Quad-Core AMD Opteron 8350 (2.0 GHz)	64 GB	8 x SAS, 15K RPM, 73 GB

Figure 7. Servers we used in our hands-on testing and research for this report.

The drive array we used for the ProLiant DL585 was a Modular Smart Array 30 (MSA30), which contained fourteen (14) 73 GB, 10K RPM U320 SCSI drives. The drive array we used for the PowerEdge R905 was a Dell PowerVault MD1000, which contained fifteen (15) 146 GB, 15K RPM SAS drives.

The workload-generating systems were desktop-class machines running Windows Vista 32-bit. All had a minimum of 1 GB of RAM and processors of speed 2.0 GHz or higher. We connected these two systems to the network via gigabit switches and verified that their performance was never a bottleneck during the test. We used a 16-port gigabit Ethernet switch for networking.

Server	Server operating system	SQL server version
Dell 1950 (Active Directory server)	Windows Server 2003 Enterprise Edition R2 SP2 x86	N/A (Active Directory Server)
HP ProLiant DL585 (SQL Server 2000 database server)	Windows Server 2003 Enterprise Edition R2 SP2 x86	SQL Server 2000 Enterprise Edition SP4
Dell PowerEdge 905 (consolidation database server)	Windows Server 2008 Enterprise Edition SP1 x64	SQL Server 2008 Enterprise Edition x64

Figure 8. Software we used in our hands-on testing and research.

BEST PRACTICE: Use the latest tested and validated software, firmware, and driver versions for NICs, storage arrays, and other components. You can find these software components at http://support.dell.com/support/downloads/index.aspx?c=us&l=en&s=gen.

NOTE: Because the ProLiant DL585 was running 32-bit software, we enabled AWE to make the extra RAM available to SQL Server 2000; in conjunction with using AWE, we set the maximum server memory setting for SQL Server 2000 to 6 GB.

Configuring the RAID drives: PowerEdge R905 with PowerVault

We used a basic file layout that followed Microsoft's recommended best practices. We separated the SQL Server transaction logs and the user database files in our drive configuration for the PowerEdge R905. During experimentation, we found that our tempdb usage was very low, so we allowed the tempdb to remain on the volume with the other system databases.

For a description of Microsoft's best practices, see our earlier guide "Consolidating SQL Server 2000 and SQL Server 2005 databases to SQL Server 2008 on Windows Server 2008 Enterprise on Dell Servers" (www.dell.com/sql).

Below we illustrate the exact drive layout we used (see Figure 9) and briefly describe each disk group.

Internal disk drives

Our Dell PowerEdge R905 server contained eight drives. We configured the first two internal server drives using RAID 1 for the operating system, SQL Server 2008 software, and SQL Server system databases. We configured the next six internal drives as a RAID 10 volume and used this volume to hold SQL Server transaction logs.

External disk drives

The PowerVault MD1000 contained 15 drives. We created one large disk group using a RAID 10 configuration. This volume held user data and indices. We left one drive unallocated and assigned it as the hot spare.

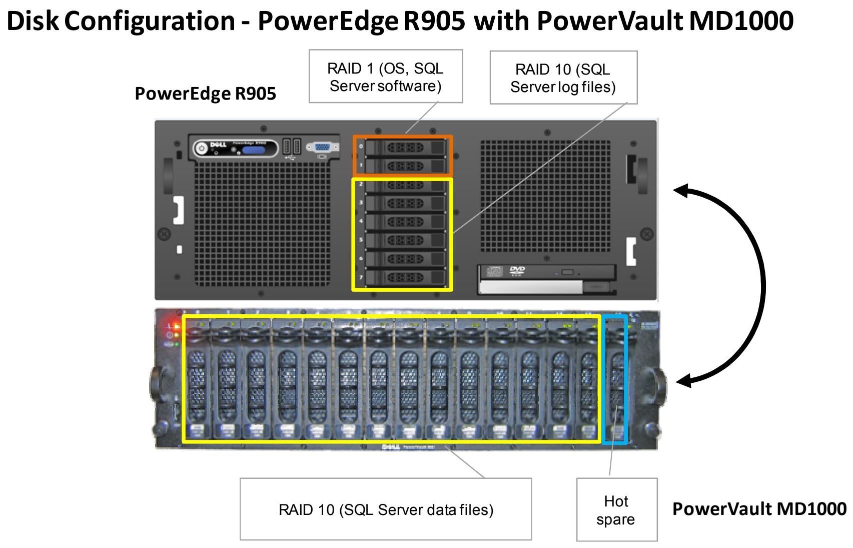

Disk Configuration - PowerEdge R905 with PowerVault MD1000
PowerEdge R905
RAID 1 (OS, SQL Server software)
RAID 10 (SQL Server log files)
RAID 10 (SQL Server data files)
Hot spare
PowerVault MD1000

Figure 9. The drive configuration we used in the consolidated server in our hands-on testing and research.

Configuring the RAID drives: ProLiant DL585 with MSA30 array

Below we illustrate the exact drive layout we used (see Figure 10) and briefly describe each disk group.

Internal disk drives

In the internal storage configuration for the ProLiant DL585, we had only four drives. We configured the first two internal server drives using RAID 1 for the operating system, SQL Server 2000 software, and SQL Server system databases. We configured the next two drives as RAID 1 to hold the SQL Server transaction logs.

External disk drives

The MSA30 contained 14 drives. We created one large disk group using a RAID 10 configuration. This volume held user data and indices.

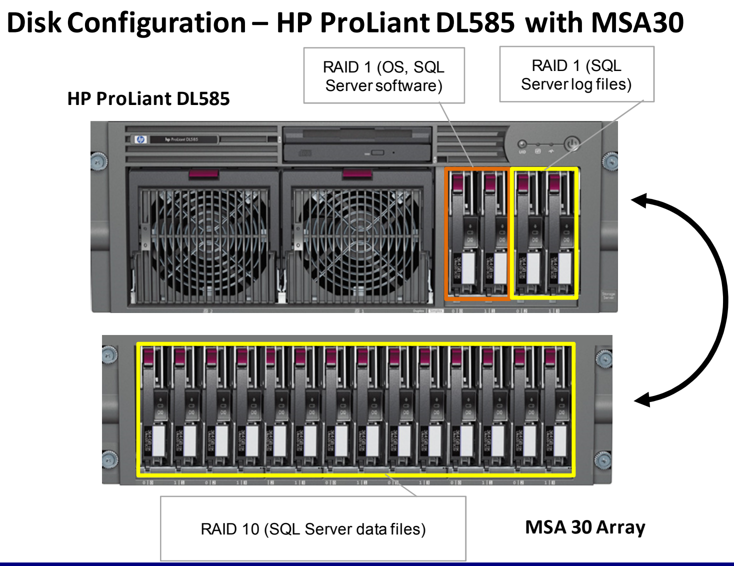

Disk Configuration – HP ProLiant DL585 with MSA30
HP ProLiant DL585
RAID 1 (OS, SQL Server software)
RAID 1 (SQL Server log files)
RAID 10 (SQL Server data files)
MSA 30 Array

Figure 10. The drive configuration we used in the HP ProLiant DL585 server and MSA30 array in our hands-on testing and research.

How we tested: The details

In this section, we discuss in depth how we conducted our testing. We review the tool we used and how we set up and run the test.

Our testing tool

We conducted our testing using the Dell DVD Store application, or DS2, an open-source application with a backend database component, a frontend Web application layer, and a driver layer that operates as the middle tier and actually executes the workload.

Because our goal was to isolate and test a database server, we did not use the Web application layer. Instead, we ran the driver application directly via its command line interface.

DS2 models an online DVD store. Virtual customers log in; browse movies by actor, title, or category; and purchase movies. The workload also creates new customers. Browsing movies involves select operations, some of which use full-text search and some of which do not. The purchase, login, and new customer procedures involve updates and inserts, as well as selects.

For more details about the DS2 stress tool, see http://www.delltechcenter.com/page/DVD+Store.

Setting up the test

The DS2 stress tool provides options to generate 10MB, 1GB, or 100GB datasets. To get the tool to generate the 10 GB and 20 GB of user data we used in this test we had to make a few straightforward changes to the source code. We detail these changes and explain how to generate the data in Appendix A.

NOTE: The database files, including indices and other supporting metadata, require approximately 10 GB of storage space. We sized the database files considerably larger than this minimum, however, for the same reason real database administrators would: to ensure that file growth during testing was not an issue.

We built the database schema on both SQL Server 2000 and SQL Server 2008 using the scripts Dell provided in the DS2 distribution package, though we made a few minor modifications. For details on the database build process, see Appendix B. After loading the data, we built the indices and full-text catalogs. We then performed a full backup of the database. This backup allowed us to restore the server to a pristine state relatively quickly between tests.

Because full-text search is not integrated into the SQL storage engine in SQL Server 2000, restoring the database does not restore the full-text indexing. Therefore, between tests, we restored the database to the ProLiant DL585 without the full text catalog and then recreated the full text catalog each time. See Appendix C for details on how we restored the SQL Server 2000 databases on the ProLiant DL585.

Improvements in SQL Server 2008 made restoring the databases on the PowerEdge R905 a simpler process. See Appendix D for details on restoring the SQL Server 2008 databases on the PowerEdge R905.

As it ships, DS2 accesses only the database named DS2. To run against multiple databases simultaneously, we added the ability to specify the database name on the command line. The details of this change appear in Appendix E.

We also had to make a few other minor modifications to the DVD Store application’s scripts. The details of these modifications appear in Appendix F.

Workload-generating systems

We used a set of two desktop-class systems, which we connected to our network via a gigabit switch, to generate the workload for our tests. Each system contained the DS2 driver application and executed a workload against a single database on the server. An instance of the DS2 driver can access only a single database. Each instance of DS2 spawned 32 threads and ran without think time. Appendix F lists the exact parameters we used.

As we explain in the How we tested: An overview section above, we made sure that the performance of the workload-generating system and the network did not limit the test in any way.

We used PSEXEC to coordinate the workload-generating systems (see technet.microsoft.com at http://technet.microsoft.com/en-us/sysinternals/bb897553.aspx). One system gathered performance statistics data from the server. Our experiments showed that this function did not affect the performance of the test.

To record each server's power consumption during testing, we used two Extech Instruments Power Analyzers/Dataloggers. We used one Extech to measure the power draw of the server, and another to measure the power draw of the array.

In addition to recording power, we used a Digi Watchport sensor to ensure a consistent temperature and humidity level.

See Appendix G for details of how we measured power.

Running the test

Before starting a test, we rebooted the workload-generating systems and the server. We allowed the server to sit idle for at least 10 minutes to ensure it was finished with all aspects of the boot process.

Before starting each test, we recorded power at an idle state for 1 minute. At the 1-minute mark, we started the data collector sets on all active workload-generating systems to gather performance counters. All workload-generating systems collected the orders per minute metric. Workload-generating system number 1 remotely collected performance data from the server. We waited 2 minutes before starting database activity to ensure that all workload-generating systems' collector sets were functioning.

At the 3-minute mark, we initiated the DS2 driver application on all active workload-generating systems. At the beginning of each DS2 run, the driver connected all threads before it initiated the actual workload. We used 32 threads per workload-generating system in our testing. The connection time for all threads was approximately 15 seconds.

After the DS2 driver application established all connections, it executed the workload; the workload typically commenced at approximately the 3:15 mark. The workload included browsing products, customer logins, new customers, and purchases. By default, DS2 allows 1 minute for the server activity to stabilize. So, at the 4:15 mark, the DS2 driver reset its statistics. It then ran the test. We specified 20 minutes for the test. See Appendix F for details on the parameters we selected.

After the workloads completed at approximately the 24:15 mark, we allowed the data collector sets on the relevant workload-generating systems to continue for 1 minute more, and then stopped the performance counters as well as the power measurements.

Summing up

This report shows that you can consolidate two HP ProLiant DL585 servers with MSA30 storage arrays on a Dell PowerEdge R905 server with a PowerVault MD1000, a move that would let you use half the rack space, less than half the power, and half the number of SQL Server licenses, all while enjoying the same performance as before.

In upcoming consolidation reports, we will present hard performance data for other configurations to help you size the servers you need for consolidation efforts and to see the potential savings from those efforts. For conceptual and how-to instruction on consolidation, see our earlier guide "Consolidating SQL Server 2000 and SQL Server 2005 databases to SQL Server 2008 on Windows Server 2008 Enterprise on Dell Servers" (www.dell.com/sql).

Appendix A. Changes we made to the data generation code and how we generated test data

The DS2 readme file encouraged using a Linux system rather than a Windows system when creating large databases. The reason for this is the much larger RAND_MAX constant in Linux. On Windows, RAND_MAX is 32,767. On Linux, it is 2,147,483,647. Because the code uses the C rand() function to generate IDs, the much larger range on Linux is an advantage. We consequently created our test databases on a Linux system.

Code changes to the ds2_create_orders.c module

The module ds2_create_orders.c defines constants that define the bounds of the customer ID and the product ID. The constants for the 10GB and 20GB database size did not exist. We added the constants for both those sizes.

On the command line for the ds2_create_orders module, we specified the size. The available options were S (small), M (medium), and L (large). We added the cases T for using the 10GB databases and W for the 20GB databases. In the switch statement that sets the values for the variables max_cust_id and max_prod_id, we added cases that assigned them the proper values for the 10GB or 20GB database sizes.

We recompiled the ds2_create_orders.c module on Linux, following the instructions in the header comments. We used the following command line: `gcc -o ds2_create_orders ds2_create_orders.c -lm`

Code changes to the ds2_create_cust.c module

We had to make the same changes to the ds2_create_cust.c module that be made to the ds2_create_orders.c module. The module ds2_create_cust.c defines constants that define the bounds of the product ID. The constants for the 10GB and 20GB database size did not exist. We added the constants for both those sizes.

On the command line for the ds2_create_cust module, we specified the size. The available options were S (small), M (medium), and L (large). We added the case T for using the 10GB databases and W for the 20GB databases. In the switch statement that sets the values for the variables max_cust_id and max_prod_id, we added cases that assigned them the proper values for the 10GB or 20GB database sizes.

We recompiled the ds2_create_cust.c module on Linux, following the instructions in the header comments. We used the following command line: `gcc -o ds2_create_cust ds2_create_cust.c -lm`

The scripts we used to generate the data

We used scripts to run all four executables. The distribution did not include scripts for the 10GB and 20GB sizes. We wrote scripts based on the ds2_create_cust_large.sh and ds2_create_orders_large.sh scripts. The ds2_create_prod and ds2_create_inv executables did not ship with associated scripts, so we created scripts using the instructions in the readme files.

We found two dependencies:

- In the file ds2_create_prod_readme.txt, it says:
```
In ../orders, after creating orderlines files,
run ds2_create_inv to total up sales by product
and create inventory load file:
ds2_create_inv n_prods > ../prod/inv.csv

Then in this directory:
ds2_create_prod n_prods > prod.csv
```
- The parameter you give to ds2_create_inv.sh must match the number you gave to ds2_create_orders.sh. The inv program uses the product id as an array index. You must have runds2_create_orders.sh first for the lookup to work.

Therefore, we had to run the scripts in a specific order. We ran them in the following order to create the 10GB database:

- ds2_create_orders_10gb.sh
- ds2_create_inv_10gb.sh
- ds2_create_prod_10gb.sh
- ds2_create_cust_10gb.sh

The scripts launch all their tasks in the background. Because there are order dependencies, we had to check and make sure the processes finished before we moved to the next step. On Linux, we used the command `ps -A | grep ds2`. Because the scripts create only processes with the "ds2 prefix" in the process name, this command displayed the list of currently running processes that the last script we launched had created.

Appendix B. Building the database

We used the scripts Dell provided in the DVD Store distribution package to build the database schema, which includes the file structure, tables, indices, stored procedures, triggers, and so on. We built a master copy of the databases, both the 10GB and 20GB versions, once for SQL Server 2000 and once for SQL Server 2008, and then used those master copies to restore our test databases to the relevant target server between each test run.

For more information on the scripts one uses to build the DS2 database, see the Dell Tech Center site at http://www.delltechcenter.com/page/DVD+Store.

We followed these steps to create the database:

1. We created the database and file structure using scripts Dell provided. We made size modifications specific to our 10GB or 20GB database and the appropriate changes to drive letters.
2. We created tables, stored procedures, and objects.
3. We converted the raw text files to use carriage return/line feed combinations that are the standard Windows format. (Because the text files are in Linux format, a line feed indicated a new line; on Windows, a carriage return/line feed indicates a new line.)
4. We set the database recovery model to bulk-logged to prevent excess logging.
5. We loaded the generated data. For data loading, we used a simple SQL Server Integration Services (SSIS) package. This approach provided the flexibility we needed when moving from system to system. The SSIS package used the same options, such as KEEPIDENTITY and TABLOCK options, as the original scripts.
6. We created indices, full-text catalogs, primary keys, and foreign keys.
7. We updated statistics on each table according to Dell scripts, which sample 18 percent of the table data.
8. We created ds2user SQL Server login and user for testing.

We made the following several changes in the build scripts:

- Because we varied the size of the datasets, we sized the files in our scripts to reflect the database size and the number of files per filegroup. We allowed for approximately 40 percent free space in our database files to ensure that filegrowth activity did not occur during the testing.
- On both the ProLiant DL585 and PowerEdge R905, we followed Microsoft's recommendation of having 0.25 to 1 file per filegroup per core. On the ProLiant DL585, which had eight cores, we used six files per filegroup. On the PowerEdge 905, which had 16 cores, we used eight files per filegroup.
- In the Dell DVD Store distribution, the following indices used the INCLUDE keyword, which was not a feature until SQL 2005. We could not use this keyword when building the SQL Server 2000 databases.

  IX_PROD_PRODID ON PRODUCTS
  IX_PROD_PRODID_COMMON_PRODID
  IX_PROD_SPECIAL_CATEGORY_PRODID

As a result, we modified the scripts for these indices to the following form for SQL Server 2000:

```
/******
On this index, the INCLUDE keyword cannot be used in SQL 2000
******/
CREATE INDEX IX_PROD_PRODID ON PRODUCTS
   (
  PROD_ID ASC
   )
  --INCLUDE (TITLE)
  ON DS_IND_FG
GO

/******
On this index, the INCLUDE keyword cannot be used in SQL 2000
******/
CREATE INDEX IX_PROD_PRODID_COMMON_PRODID ON PRODUCTS
   (
  PROD_ID ASC,
  COMMON_PROD_ID ASC
   )
  --INCLUDE (TITLE, ACTOR)
  ON DS_IND_FG
GO

/******
On this index, the INCLUDE keyword cannot be used in SQL 2000
******/
CREATE INDEX IX_PROD_SPECIAL_CATEGORY_PRODID ON PRODUCTS
   (
  SPECIAL ASC,
  CATEGORY ASC,
  PROD_ID ASC
  )
  --INCLUDE (TITLE, ACTOR, PRICE, COMMON_PROD_ID)
  ON DS_IND_FG
GO
```

- We did not use the DBCC PINTABLE command for the CATEGORIES and PRODUCTS tables, both because Microsoft recommends against this practice and because the commands do nothing in SQL Server 2008.
- In SQL 2008, we added the FORCESEEK query hint to the BROWSE_BY_ACTOR stored procedure, to force SQL Server 2008 to use an index seek, instead of an index scan, in its query execution plan. We made this change because our initial tests showed that SQL Server was using a highly inefficient index scan. (For more information about using FORCESEEK to override an inefficient query plan, see http://msdn.microsoft.com/en-us/library/bb510478(SQL.100).aspx.) We did not have this issue in SQL Server 2000. Therefore, we created the SQL Server 2008 BROWSE_BY_ACTOR procedure as follows:

```
CREATE PROCEDURE BROWSE_BY_ACTOR
  (
  @batch_size_in            INT,
  @actor_in                 VARCHAR(50)
  )

  AS

  SET ROWCOUNT @batch_size_in
  SELECT * FROM PRODUCTS

  --added to force index seek
  WITH (FORCESEEK)

  WHERE CONTAINS(ACTOR, @actor_in)
  SET ROWCOUNT 0
GO
```

- We created a SQL Server login called ds2user and a database user mapped to this login for each database. We made each such user a member of the db_owner fixed database role.
- Because Microsoft modified full-text indexing features in SQL Server 2008, using the DVD Store scripts as a reference we created the full-text catalog and index on the PRODUCTS table manually in Management Studio. For SQL Server 2000, we used the DVD Store scripts.

Appendix C. Restoring the database on SQL Server 2000

This appendix details the procedures we used to reinitialize the SQL Server 2000 DS2 databases between test runs.

We used the traditional backup and restore method to restore the DS2 database to the ProLiant DL585 between test runs. Because SQL Server 2000 does not fully integrate full-text catalogs with the SQL Server storage engine, as SQL Server 2008 does, we had to build our DS2 database without the full text catalog and index, then recreate it upon each database restore.

Then we restored the database and rebuilt the full text catalog.

1. We restored the DS database to the ProLiant DL585 as follows:
   a. We logged into the DL585 with Administrative access
   b. We restored the database in Enterprise Manager by right-clicking "Databases", then choosing "All tasks", "Restore Database…"
   c. We named our database appropriately, chose "From Device", and then select the backup file
   d. We clicked OK to complete the restore
2. We restored the full text catalog by then running the following script to recreate it. This script is included with the DVDStore distribution. We waited for the full population of the full text index before proceeding with testing.

```
EXEC sp_fulltext_database 'enable'
EXEC sp_fulltext_catalog  'FULLTEXTCAT_DSPROD', 'create', 'C:\Program
Files\Microsoft SQL Server\MSSQL\FTDATA'
EXEC sp_fulltext_table    'PRODUCTS',           'create',
'FULLTEXTCAT_DSPROD', 'PK_PRODUCTS'
EXEC sp_fulltext_column   'PRODUCTS',           'ACTOR', 'add'
EXEC sp_fulltext_column   'PRODUCTS',           'TITLE', 'add'
EXEC sp_fulltext_table    'PRODUCTS',           'activate'
EXEC sp_fulltext_catalog  'FULLTEXTCAT_DSPROD', 'start_full'
GO
```

Appendix D. Restoring the database on SQL Server 2008

We followed the procedures below when dropping and restoring the SQL Server 2008 databases between test runs.

Multi-database note: When we executed multiple workloads against the PowerEdge 905, we had to restore multiple copies of the same database. Between test runs, we removed and then re-added all the databases. For each of the databases in a test, we followed these steps:

1. We dropped any existing database(s):
    a. We logged in to Management Studio.
    b. In Object Explorer, we browsed to Databases in the left pane.
    c. We right-clicked the database we wanted, and selected Delete.
    d. In the Delete Object window, we checked the Close existing connections box.
    e. We clicked OK to delete.
2. We restored fresh database(s):
    a. We right-clicked Databases in Management Studio, and selected Restore Database.
    b. We entered the appropriate database name. If we were running a multi-database test, we named the databases sequentially (i.e., DS2_1 and DS2_2) and used these sequential database names to point our two workload-generating systems at the correct database.
    c. We clicked From Device, and then browsed to the appropriate path. On the options page of the restore configuration, we ensured that all paths pointed to the correct locations.
    d. We clicked OK to start the restore.

       Note: If you are testing with more than one database, repeat step 2 as necessary.
3. We dropped and recreated the account ds2user to properly map the ds2user SQL Server login to the database user.

The default version of the DS2 software uses the "SA" SQL Server account, but we chose to change the user context to another user, ds2user.

NOTE: In the scripts below, we use DS2 as the database name. In the shipping version of the DVD Store software, DS2 accesses only that hardcoded database name. As we note above, we changed the code to let a tester specify a different database name in the script as appropriate.

```
USE [DS2]
GO

IF  EXISTS (SELECT * FROM sys.database_principals WHERE name =
N'ds2user')
DROP USER [ds2user]
GO

CREATE USER [ds2user] FOR LOGIN [ds2user] WITH
DEFAULT_SCHEMA=[dbo]

GO
EXEC sp_addrolemember N'db_owner', N'ds2user'
GO
```

4. We rebooted the server.

Appendix E. Code changes to DS2

Changes to the ds2xdriver.cs module

To use the 10GB and 20GB databases we created earlier, we had to change the following constants:

- In the routine Controller(), we changed the string "sizes". We added the T option for the 10GB database size and the W option for the 20GB database size. DS2 uses the sizes string to interpret the db_size_str option.
- In the class Controller, we changed the arrays MAX_CUSTOMER and MAX_PRODUCT. To each, we added values specifying the bounds for the customer and product IDs. The Controller() routine uses these arrays.
- We added a command line parameter for the database name, "—database_name". As it ships, DS2 accesses only the database named DS2. We needed the name to be a parameter so we could run against multiple databases on the same server.

Changes to the ds2sqlserverfns.cs module

We changed the connection string to increase the number of available connections and to not use the default administrator ("sa") account. The default version of DS2 capped the number of connections at 100. We raised the limit to 200 to allow room for experimentation. As we note above, the default version used the sa account for its operations. We created a user account called ds2User and used that account. The connection string is called sConnectionString.

We also changed the connection string to use a command line parameter for the database name.

The ds2connect routine defines ConnectionString. We used the following string; the changes we made appear **in bold**.

```
string sConnectionString = "User
ID=ds2User;Initial Catalog="+dbname+";Max Pool
Size=200;Connection Timeout=120;Data Source=" +
Controller.target;
```

Building the ds2sqlserverdriver.exe executable

We recompiled the ds2xdriver.cs and ds2sqlserverfns.cs module on Windows by following the instructions in the header comments. Because the instructions were for compiling from the command line, we used the following steps:

1. We opened a command prompt.
2. We used the `cd` command to change to the directory containing our sources.
3. We ran the batch file C:\Program Files\Microsoft Visual Studio 9.0\Common7\Tools\vsvars32.bat. This set up the environment variables for us.
4. We executed the following command:

```
csc /out:ds2sqlserverdriver.exe
ds2xdriver.cs ds2sqlserverfns.cs
/d:USE_WIN32_TIMER /d:GEN_PERF_CTRS
```

Appendix F. Running DS2 to support multiple databases

Test topology

On the HP ProLiant DL585, we ran only one database, called DS2, and we executed only one workload against it. We used one workload-generating system per database workload, so in all tests involving the ProLiant DL585 we used only one system.

On the Dell PowerEdge R905, however, we ran two databases (DS2_1 and DS2_2). For each database, we used a dedicated workload-generating system, each with its own copy of the DS2 driver application. Both workload-generating systems were member machines of our Windows domain. Figure 11 shows an overview of the test topology for the ProLiant DL585. Figure 12 shows a sample topology for the PowerEdge R905 running with two workload-generating systems.

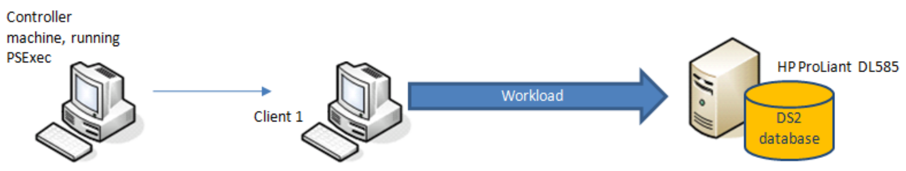


Figure 11. Test topology for the HP ProLiant DL585.

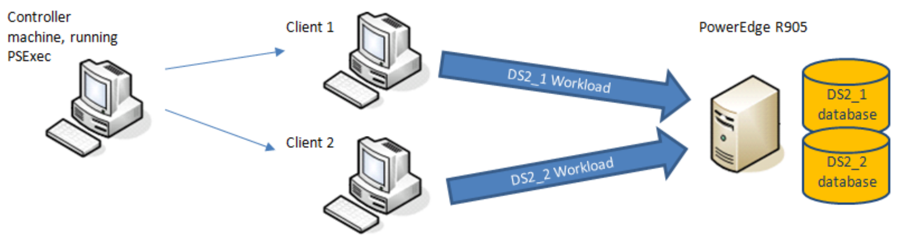


Figure 12. Sample test topology for the Dell PowerEdge R905 with two workload-generating systems.

Performance statistics

Our workload-generating systems were desktop-class machines running the 32-bit version of Windows Vista. DS2 requires .NET 2.0 on all systems.

The DS2 driver application creates an orders per minute (OPM) performance counter on all systems. We created data collector sets on all systems to collect the OPM statistics once every second.

In addition, we remotely collected server performance statistics via a data collector set in Reliability and Performance Monitor on workload-generating system number 1 for all tests. On that system, we created extra data collectors sets to collect on the target server the performance statistics in Figure 13.

Object	Counter
Memory	Available MBytes
Physical disk	% idle time (all drives)
Processor	% processor time (all cores)
Network interface card	Total Bytes/sec
SQL Server:Buffer Manager	Buffer cache hit ratio
SQL Server:Buffer Manager	Checkpoint pages/sec

Figure 13. Performance statistics we monitored during testing.

NOTE: By default, the Reliability and Performance Monitor stores its results in binary format. Before testing, we specified that the Reliability and Performance Monitor use comma-separated values (CSV) format.

Test timing

We restarted all physical machines, both servers and workload-generating systems, prior to each test and allowed them to rest for a minimum of 10 minutes before we began the test. Figure 14 provides the timeline for the testing process.

Time	Event
00:00	We started the power measurements.
01:00	We used PSExec to remotely start the logman utility to initiate the data collector sets on all the active workload-generating systems.
03:00	We used PSExec to remotely start the DS2 workload on all the active workload-generating systems. The beginning of the workload involves threads connecting to the database, but it does not execute the test itself yet.
03:15	All threads are connected, and the workload execution begins.
04:15	The DS2 driver application resets its internal statistics. The workload continues executing against the database(s).
24:15	The DS2 workload ends, and the threads exit. We chose to run our tests for 20-minute periods because the workload had stabilized sufficiently.
25:15	We used PSExec to remotely halt the data collector sets on all the active workload-generating systems. We stopped the power measurements and archived the data extracts.

Figure 14. Testing timeline.

Test parameters

The DS2 driver application allows for multiple parameters, all of which Dell documents at their Tech Center site at http://www.delltechcenter.com/page/DVD+Store. Figure 15 provides the parameters we used on all the workload-generating systems.

All other parameters were at their default values.

To show how we call the DS2 driver application on a workload-generating system, the following sample code shows that call on system 2 using the DS2_2 database.

```
c:\ds2run\ds2sqlserverdriver --target=R905-SQL --
ramp_rate=10 --run_time=20 --n_threads=32 --
db_size_str=W --think_time=0 --database_name=DS2_2
```

DS2 parameter	Value we used
--target	Either DL585-SQL or R905-SQL, which were the machine names of our servers.
--ramp_rate	10
--run_time	20
--n_threads	32
--db_size_str	T to use the 10GB database or W to use the 20GB database. These databases are not part of the default distribution. See Appendix E for details.
--think_time	0
--database_name	DS2 for the single database test; DS2_1 or DS2_2 for the multi-database tests. We added this parameter into the DS2 driver application. See Appendix E for details.

Figure 15. Parameters we used on all the workload-generating systems.

Appendix G. Power measurement

To record each system's power consumption during testing, we used two Extech Instruments (www.extech.com) 380803 Power Analyzers/Dataloggers. Both the ProLiant DL585 and the PowerEdge R905 have two power supplies, so we measured the power draw of each server by using a single Extech with a splitter cable.

Both servers also used a storage array, so we connected a second Extech to measure the power draw of the drive arrays. The PowerVault MD 1000 and MSA30 also each had dual power supplies, so we used a splitter cable to measure the power draw through a single meter.

We connected all the Extech Power Analyzers to one monitoring system to record the power draw of the systems. We used the Power Analyzer's Data Acquisition Software (version 2.11) to capture all the recordings. We installed the software on a separate PC, to which we connected all the Power Analyzers via a separate RS-232 cable for each one.

We captured power consumption at 1-second intervals.

To gauge the idle power usage, we recorded the power usage for 1 minute while each server was running the operating system but otherwise idle.

Because we were using a second Extech for the test, we summed the wattage of both meters. We averaged the power usage during the period the server was running the benchmark. We call this time the *power measurement interval*.

While recording power, we used a Digi Watchport sensor and the Watchport Manager software (version 1.19.07) to monitor the temperature and humidity levels.

1. We started the Power Analyzer software on the power monitoring PC. We started a second instance of the Power Analyzer software to monitor the second Extech meter connected to the relevant storage array.
2. Before beginning the test, we recorded the power usage for 1 minute while each server was running the operating system but otherwise idle.

3. We started the started the test and recorded power during the run.
4. We recorded for one minute past the end of the test.
5. Power Analyzer saves its data in a comma separated format. We imported the CSV files into Microsoft Excel®.
6. Because we used two meters, we summed the power draw for both meters at all data points.

Appendix H. How we report results

Identifying a period of heavy activity

For all of our metrics, we use the results of the time period from 1,000 seconds to 1,200 seconds into the test. This is a period of steady activity and heavy load, and it suffers from neither ramp-up nor ramp-down effects.

During ramp-up, the server is adjusting to a sudden and dramatic increase in demands on SQL Server. Until the system stabilizes, the results do not represent the true capacities of the server. One particular concern is the possibility of some threads returning highly inflated rates. These inflated rates are not representative of the system under full load and including them can distort the results.

During the ramp-down period, overall performance eventually drops as more and more threads terminate. Including data from the period after threads start terminating can consequently distort results.

In both the ramp-up and the ramp-down periods, power consumption can be lower than when the system is under full load.

Of course, stable does not necessarily mean flat. A number of factors, such as SQL Server's checkpointing behavior, can create some fluctuation during the stable period.

Based on our observations as we tested, we determined that the period from 1,000 seconds to 1,200 seconds represented stable system behavior.

Determining the median run

For each configuration, we ran three runs. We report the median run, which we determined as follows:

1. We copied the results for all three runs from the workload-generating systems to a single results-processing system.
2. We imported the results into Microsoft Excel.

NOTE: By default, the Reliability and Performance Monitor stores its results in binary format. Before testing, we specified that the Reliability and Performance Monitor use comma-separated values (CSV) format.

3. For each of the three runs, we determined the average OPM rate during the period from 1,000 seconds to 1,200 seconds into the test by taking the OPM at 1-second intervals and calculating the arithmetic mean.
4. We looked at the three averages and identified the median; we reported the results of this run.

Reporting DS2 results

To report the DS2 OPM data, we followed these steps:

1. For the median run of each configuration, we copied the OPM rates for each workload-generating system in the test into a single Excel worksheet. For the HP ProLiant DL585, there was always only one workload-generating system. For the Dell PowerEdge R905, there were two workload-generating systems.
2. We looked at only the range of data from 1,000 to 1,200 seconds into the test, i.e., the same data we used to compute the averages.
3. We plotted the results on a single chart. The charts had three lines, with a single black line representing the ProLiant DL585 results and two colored lines representing the PowerEdge R905 results.

Reporting the power results

To report the power usage of each test configuration, we followed these steps:

1. For the median run of each configuration, we copied the power draw, in watts, to an Excel worksheet.
2. Because all configurations used external storage arrays, we summed the power for the server and the array. As we explain above, we used separate Extech meters for the server and the drive array.

NOTE: The output from the Power Analyzer software reports the power with a 'W' suffix, as in "217W". We used the Excel replace feature to delete the Ws.

3. We looked at only the range of data from 1,000 to 1,200 seconds into the test, i.e., the same time period over which we computed the OPM averages.

4. We plotted the results on a single chart. In this case, there were always only two lines: one for the HP ProLiant DL585, and one for the PowerEdge R905.
5. Additionally, we computed the average power draw during the period from 1,000 to 1,200 seconds into the test. We computed the average of the combined power draw of the server and the storage.
6. We multiplied the power draw of the ProLiant DL585 and MSA30 by the consolidation factor for this set of results (two).
7. We then computed the power draw of the PowerEdge 905 with PowerVault, as a percentage of the consolidated power for the number of ProLiant DL585s it could replace. We refer to these percentages in the report.

About Principled Technologies

We provide industry-leading technology assessment and fact-based marketing services. We bring to every assignment extensive experience with and expertise in all aspects of technology testing and analysis, from researching new technologies, to developing new methodologies, to testing with existing and new tools.

When the assessment is complete, we know how to present the results to a broad range of target audiences. We provide our clients with the materials they need, from market-focused data to use in their own collateral to custom sales aids, such as test reports, performance assessments, and white papers. Every document reflects the results of our trusted independent analysis.

Principled Technologies, Inc.
1007 Slater Road, Suite 250
Durham, NC, 27703
www.principledtechnologies.com

We provide customized services that focus on our clients' individual requirements. Whether the technology involves hardware, software, Web sites, or services, we offer the experience, expertise, and tools to help you assess how it will fare against its competition, its performance, whether it's ready to go to market, and its quality and reliability.

Our founders, Mark L. Van Name and Bill Catchings, have worked together in technology assessment for over 20 years. As journalists they published over a thousand articles on a wide array of technology subjects. They created and led the Ziff-Davis Benchmark Operation, which developed such industry-standard benchmarks as Ziff Davis Media's Winstone and WebBench. They founded and led eTesting Labs, and after the acquisition of that company by Lionbridge Technologies were the head and CTO of VeriTest.